Ατομική Διπλωματική Εργασία

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΚΑΙ ΠΡΟΒΛΕΨΗ ΚΟΣΤΟΥΣ ΑΝΑΠΤΥΞΗΣ ΛΟΓΙΣΜΙΚΟΥ ΜΕΣΩ ΥΠΟΛΟΓΙΣΤΙΚΗΣ ΝΟΗΜΟΣΥΝΗΣ

Έφη Παπαθεοχάρους

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ιούλιος 2004

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μοντελοποίηση και Πρόβλεψη Κόστους Ανάπτυξης Λογισμικού μέσω Υπολογιστικής Νοημοσύνης

Έφη Παπαθεοχάρους

Επιβλέπων Καθηγητής
Ανδρέας Ανδρέου

Η Ατομική αυτή Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Ιούλιος 2004

Ευχαριστίες

Ευχαριστώ τον επιβλέποντα καθηγητή μου, Δρα. Ανδρέα Ανδρέου, για τη βοήθεια και τις συμβουλές που μου παρείχε, κατά τη διάρκεια της εκπόνησης της διπλωματικής μου μελέτης, καθώς και για το υλικό που παρείχε προς βοήθεια για διεκπεραίωση της έρευνας.

Επίσης, ευχαριστώ τον Δρα. Ευθύβουλο Κυριάκου, για την βοήθεια του καθώς και για τις συμβουλές που μου παρείχε.

Επίσης, ευχαριστίες οφείλω και στον καθηγητή Jean-Marc Desharnais, Ph.D. ο οποίος είχε την ευγένεια όταν επικοινώνησα μαζί του, να διαθέσει για την διπλωματική μου μελέτη, το σύνολο δεδομένων που χρησιμοποίησε στην δική του έρευνα.

Περίληψη

Η Υπολογιστική Νοημοσύνη περιλαμβάνει τον συνδυασμό δύο τεχνικών μηχανών εκμάθησης, των Τεχνητών Νευρωνικών Δικτύων και του Γενετικού Προγραμματισμού, σε μια συνεκτική λύση και την ανάπτυξη μιας υβριδικής προσέγγισης. Η διπλωματική αυτή εργασία, ασχολείται με την μοντελοποίηση και την πρόβλεψη του κόστους ανάπτυξης λογισμικού μέσω Υπολογιστικής Νοημοσύνης. Ο υπολογισμός του κόστους ανάπτυξης ενός έργου λογισμικού στον τομέα της Τεχνολογίας Λογισμικού, είναι μια δραστηριότητα πολύ σημαντική στο επίπεδο της διαχείρισης ενός οργανισμού ή μιας επιχείρησης. Αφορά τον καθορισμό της προσπάθειας που καταβάλλεται (effort), που συνήθως υπολογίζεται σε άνθρωπο-μήνες (person-months), για την ανάπτυξη ενός έργου λογισμικού.

Η έρευνα ξεκίνησε με την προσπάθεια μελέτης και κατανόησης της πολύπλοκης λειτουργίας των μεθόδων Υπολογιστικής Νοημοσύνης για επίλυση του προβλήματος. Το ενδιαφέρον εστιάστηκε στα Τεχνητά Νευρωνικά Δίκτυα και στον Γενετικό Προγραμματισμό. Η έρευνα ακολούθησε δύο διόδους. Αρχικά, με το εργαλείο της Ward Systems Group, NeuroShell 2, καθορίστηκε ένα θεμιτό πλαίσιο αποτελεσμάτων της πρόβλεψης του κόστους ανάπτυξης λογισμικού, με βάση το οποίο μπορέσαμε να στηρίξουμε την έρευνα και την αξιολόγηση των αποτελεσμάτων. Με την υλοποίηση ενός εργαλείου διεξαγωγής πειραμάτων, δημιουργήθηκαν Τεχνητά Νευρωνικά Δίκτυα που υπολογίζουν την προσπάθεια ανάπτυξης λογισμικού και η μέθοδος βελτιστοποιήθηκε με την προσαρμογή στο εργαλείο Γενετικού Αλγόριθμου, ο οποίος βοήθησε στην εύρεση της κατάλληλης αρχιτεκτονικής του δικτύου με σκοπό την εύρεση καλύτερων αποτελεσμάτων.

Τα συμπεράσματα έδειξαν ότι οι μέθοδοι μπορούν να προσεγγίζουν με σχετικά μεγάλη ακρίβεια το πραγματικό κόστος, και με την ικανότητά τους να γενικεύουν και να λύνουν προβλήματα μεγάλης πολυπλοκότητας, εμφανίζονται σαν μια από τις πιο βέλτιστες μεθόδους επίλυσης του προβλήματος του υπολογισμού του κόστους ανάπτυξης λογισμικού.

Πίνακας Περιεχομένων

Κεφάλαιο 1

Εισαγωγή

1.1 Γενική Περιγραφή

Ο υπολογισμός του κόστους ανάπτυξης ενός έργου λογισμικού στον τομέα της Τεχνολογίας Λογισμικού είναι μια δραστηριότητα βαρυσήμαντη, ουσιαστική και πολύτιμη στα έργα ανάπτυξης Πληροφοριακών Συστημάτων. Ιδιαίτερα όταν η πρόβλεψη του κόστους ανάπτυξης λογισμικού είναι επιτυχημένη και μπορεί να πραγματοποιηθεί στα αρχικά στάδια της ανάπτυξης ενός έργου λογισμικού, τότε μπορεί να επηρεάσει σημαντικά στην απόδοση, στην παραγωγικότητα, στην λήψη σωστών διοικητικών αποφάσεων στο επίπεδο της διαχείρισης ενός οργανισμού ή μιας επιχείρησης.

Ο υπολογισμός του κόστους ανάπτυξης λογισμικού άρχισε να απασχολεί τον χώρο της Τεχνολογίας Λογισμικού στα τέλη του 1950 και του 1960. Ειδικά όμως κατά τη διάρκεια των τελευταίων τριών δεκαετιών, έχουν αναπτυχθεί και έχουν ερευνηθεί πιο εκτεταμένα διάφορες τεχνικές υπολογισμού για να προσφέρουν βελτιωμένα αποτελέσματα στις προβλέψεις του κόστους ανάπτυξης. Αν έστω αυτά τα μοντέλα μπορούν να συνεισφέρουν μια ελαφρά βελτιωμένη πρόβλεψη του κόστους της σχεδόν χαοτικής και δυναμικά μεταβαλλόμενης διαδικασίας ανάπτυξης λογισμικού, από αυτά που ήδη υλοποιήθηκαν, τότε θα προσφέρουν σημαντικά στα οικονομικά ωφέλη και στη σωστή διαχείριση των έργων.

Η διπλωματική αυτή εργασία ασχολείται με την μοντελοποίηση και την πρόβλεψη του κόστους ανάπτυξης λογισμικού μέσω Υπολογιστικής Νοημοσύνης. Πιο συγκεκριμένα, αποτελεί τον συνδυασμό δύο τεχνικών μηχανών εκμάθησης, των

Τεχνητών Νευρωνικών Δικτύων και του Γενετικού Προγραμματισμού, σε μια συνεκτική λύση και την ανάπτυξη μιας υβριδικής προσέγγισης.

1.2 Σκοπός

Σκοπός της διπλωματικής εργασίας είναι η μελέτη των τεχνικών μοντελοποίησης, η ανάπτυξη μοντέλου για την πρόβλεψη του κόστους ανάπτυξης λογισμικού μέσω Υπολογιστικής Νοημοσύνης, αξιολόγηση του μοντέλου και τέλος η διεξαγωγή συμπερασμάτων. Με την βοήθεια Υπολογιστικής Νοημοσύνης μπορεί να αναπτυχθεί μια δυναμική διαδικασία μάθησης, μέσω της οποίας κάποιο σύστημα, αντιδρώντας σε κάποιο περιβαλλοντικό ερέθισμα, αναδιοργανώνεται έτσι ώστε να έχει πιο αποδοτικά και αξιόλογα αποτελέσματα.

Το μοντέλο που προτείνεται, αναφέρεται και έχει σαν πυρήνα τη μετρική της προσπάθειας που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο. Η προτεινόμενη μεθοδολογία πειραμάτων ακολουθεί δύο διόδους. Αρχικά διεξάγονται τα πειραματικά τρεξίματα με την χρήση ενός εργαλείου, το NeuroShell 2, το οποίο προσομοιώνει την χρήση Τεχνητών Νευρωνικών Δικτύων. Στη συνέχεια, ακολουθεί η υλοποίηση ενός εργαλείου διεξαγωγής πειραμάτων, το οποίο δημιουργεί Τεχνητά Νευρωνικά Δίκτυα και με την χρήση Γενετικού Αλγορίθμου, επιλέγει την βέλτιστη αρχιτεκτονική για το Τεχνητό Νευρωνικό Δίκτυο.

1.3 Ανασκόπηση Κεφαλαίων

Στο πρώτο κεφάλαιο έγινε μια μικρή εισαγωγή για τον υπολογισμό του κόστους ανάπτυξης λογισμικού, για την θεματολογία και για τους στόχους της διπλωματικής εργασίας.

Στο δεύτερο κεφάλαιο γίνεται μια αναλυτική περιγραφή του υπολογισμού του κόστους ανάπτυξης λογισμικού, καθώς και της σημασίας αλλά και της δυσκολίας που εμπερικλείεται γύρω από τον ακριβή αυτό υπολογισμό.

Στο τρίτο κεφάλαιο παρουσιάζονται τα ιστορικά στοιχεία, καθώς και εμπειρικές σχετικές μελέτες που αναφέρονται στην βιβλιογραφία, τα συμπεράσματα και η κατηγοριοποίηση των μοντέλων υπολογισμού του κόστους ανάπτυξης λογισμικού. Ακόμα περιγράφονται οι παράγοντες που επηρεάζουν το κόστος ανάπτυξης λογισμικού

και τα κριτήρια αξιολόγησης της ποιότητας των αποτελεσμάτων του υπολογισμού του κόστους ανάπτυξης λογισμικού.

Στο τέταρτο κεφάλαιο αναλύεται η προτεινόμενη μεθοδολογία υπολογισμού προς μελέτη και υλοποίηση. Αρχικά, περιγράφονται οι δυνατότητες του εργαλείου NeuroShell 2, με το οποίο ακολουθήθηκε μια σειρά από πειράματα για τον υπολογισμό του κόστους και το οποίο βοήθησε να καθοριστεί το πλαίσιο αξιολόγησης των αποτελεσμάτων για το εργαλείο που υλοποιήθηκε στη συνέχεια. Περιγράφονται τα πειράματα, τα αποτελέσματα και τα συμπεράσματα. Στη συνέχεια, περιγράφεται το προτεινόμενο εργαλείο διεξαγωγής πειραμάτων με την χρήση Υπολογιστικής Νοημοσύνης, τα σχετικά αποτελέσματα και τα συμπεράσματα.

Στο πέμπτο κεφάλαιο παρουσιάζονται αναλυτικά τα αποτελέσματα της μελέτης και του προτεινόμενου εργαλείου διεξαγωγής πειραμάτων, καθώς οι παρατηρήσεις που εξάγονται.

Στο έκτο κεφάλαιο περιγράφονται κάποια συμπεράσματα και εισηγήσεις για μελλοντική βελτίωση, αναφορά ή εργασία.

Κεφάλαιο 2

Εισαγωγή στον Υπολογισμό του Κόστους Ανάπτυξης Λογισμικού

2.1 Εισαγωγή

Τα τελευταία χρόνια, η ανάπτυξη λογισμικού έχει γίνει η πιο ουσιαστική, η πιο πολύτιμη και συνεπώς, η πιο ακριβοπληρωμένη διαδικασία στα έργα ανάπτυξης Πληροφορικών Συστημάτων. Η ανάγκη για πρόβλεψη κάποιων βασικών χαρακτηριστικών που διέπουν το κόστος ανάπτυξης λογισμικού, πριν ακόμα αυτό υλοποιηθεί, και η ανάγκη για την ακριβή πρόβλεψη αυτού του μελλοντικού κόστους, ειδικά στα αρχικά στάδια της ανάπτυξης του λογισμικού, έχει οδηγήσει στην δημιουργία Υπολογιστικών Μοντέλων στην Τεχνολογία Λογισμικού, που υπολογίζουν και προβλέπουν παράγοντες, όπως την προσπάθεια ανάπτυξης λογισμικού, την ποιότητα και την αξιοπιστία του λογισμικού αλλά και την παραγωγικότητα των προγραμματιστών.

Στόχος αυτού του κεφαλαίου είναι να αναγνωριστούν οι έννοιες που διέπουν τον υπολογισμό του κόστους ανάπτυξης λογισμικού, καθώς και να τονιστεί η σημασία, αλλά και η δυσκολία της ακριβής μέτρησης του υπολογισμού αυτού.

2.2 Γενική Περιγραφή του Υπολογισμού του Κόστους Ανάπτυξης Λογισμικού

Ο υπολογισμός του κόστους ανάπτυξης λογισμικού στον τομέα της Τεχνολογίας Λογισμικού, μπορεί να αφορά τον καθορισμό ενός ή περισσότερων παραγόντων: της προσπάθειας που καταβάλλεται (effort), που συνήθως υπολογίζεται σε άνθρωπο-μήνες

(person-months), της διάρκειας ανάπτυξης έργου (duration), που μετριέται σε χρονολογικές μονάδες (calendar time) και του χρηματικού κόστους.

Το μοντέλο που προτείνεται στην μελέτη αυτή, όπως και τα περισσότερα μοντέλα που αναπτύχθηκαν στο παρελθόν και σε μελέτες της σχετικής βιβλιογραφίας, αναφέρονται, και έχουν σαν πυρήνα, τη μετρική της προσπάθειας που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο. Η μετρική της προσπάθειας υπολογίζεται σε μονάδες που ονομάζονται άνθρωπο-μήνες (person-months). Ο ορισμός ενός άνθρωπο-μήνα είναι η προσπάθεια που καταβάλλεται μέσα σε ένα μήνα, από έναν άνθρωπο. Η μετρική της προσπάθειας στη συνέχεια, μπορεί να μεταφραστεί στις άλλες μετρικές, όπως είναι η χρονική διάρκεια και το χρηματικό κόστος, με την βοήθεια τύπων μετασχηματισμού, αφού είναι στοιχεία αλληλένδετα μεταξύ τους.

2.3 Σημασία του Υπολογισμού του Κόστους Ανάπτυξης Λογισμικού

Στα μέσα της δεκαετίας του ’90, η ομάδα μελέτης Standish είχε κατορθώσει να επισκοπήσει περισσότερα από 8,000 έργα ανάπτυξης λογισμικού. Τα αποτελέσματα αυτής της μελέτης έδειξαν ότι ένα μέσο έργο υπερβαίνει το υπολογιζόμενο χρηματικό κόστος κατά 90%, υπερβαίνει τον υπολογιζόμενο χρόνο-προγραμματισμό κατά 222%, ενώ περισσότερα από τα μισά ολοκληρωμένα έργα παρέδωσαν λιγότερα από το 50% των αρχικών απαιτήσεων [20]. Αυτές οι στατιστικές παρουσιάζουν τα συμπτώματα και τα βαθιά προβλήματα που εμφανίζονται στην διαδικασία και στην ολοκλήρωση έργων ανάπτυξης λογισμικού και έχουν ως αποτέλεσμα τον λανθασμένο προϋπολογισμό των παραγόντων που επηρεάζουν το κόστος, γεγονός που οφείλεται κυρίως στην έλλειψη γνώσης και εμπειρίας των εμπλεκομένων στην αρχή του κύκλου ζωής ενός έργου.

Ο υπολογισμός του κόστους ανάπτυξης ενός έργου λογισμικού στον τομέα της Τεχνολογίας Λογισμικού, είναι μια δραστηριότητα πολύ σημαντική στο επίπεδο της διαχείρισης ενός οργανισμού ή μιας επιχείρησης, αφού οι λανθασμένοι υπολογισμοί και οι ανακρίβειες στον υπολογισμό του κόστους και του χρόνο-προγραμματισμού, είχαν οδηγήσει επανειλημμένα έργα στην καταστροφή. Ειδικότερα, η παραγωγή εντός χρόνου και εντός χρηματικού κόστους είναι κρίσιμη για την βιωσιμότητα οργανισμών που αναλαμβάνουν την διεκπεραίωση ενός έργου και έχουν άμεση επίδραση στην φήμη τους, στην ανταγωνιστικότητά τους και στην απόδοσή τους.

Στον τομέα της βιομηχανίας έχει αναγνωριστεί ότι αν βελτιωθούν οι μέθοδοι υπολογισμού του κόστους ενός έργου, τότε τα προβλήματα που εμφανίζονται και

αφορούν τον καταμερισμό των πόρων και τον χρόνο-προγραμματισμό, θα εκμηδενίζονταν. Ιδιαίτερα οι διαχειριστές (managers) τονίζουν την σημασία της βελτίωσης του υπολογισμού αυτού, έτσι ώστε να είναι πιο ακριβής, αφού είναι αναγκαίο στοιχείο για την επίτευξη καλύτερου προγραμματισμού, παρακολούθησης, ελέγχου και επιτυχημένης τελικά παράδοσης του έργου.

Πιο συγκεκριμένα, ο ακριβής υπολογισμός της προσπάθειας ανάπτυξης λογισμικού μπορεί να βοηθήσει στην κατηγοριοποίηση και εύρεση μιας σειράς προτεραιότητας των έργων, δίνοντας και ανάλογη σημασία στο γενικό πλαίσιο της επιχείρησης ή του οργανισμού. Επίσης, μπορεί να καθορίσει ποιους και πόσους πόρους είναι καλύτερα να αφοσιωθούν σε κάθε έργο και με ποιο τρόπο θα μπορέσουν να τους εκμεταλλευτούν πιο αποδοτικά. Ακόμα, μπορεί να παίξει καθοριστικό ρόλο στην επίδραση και αντιμετώπιση των απροσδόκητων αλλαγών και να υποστηρίξει τον ενδεχόμενο επαναληπτικό προγραμματισμό του έργου, ενώ η διαχείριση και ο έλεγχος του έργου γίνονται πιο εύκολα και πιο αποδοτικά.

Η διαδικασία ανάπτυξης λογισμικού, ιδιαίτερα για μεγάλης κλίμακας έργα, για να μπορέσει να προβεί σε επιτυχία, είναι αναγκαία η ακριβής πρόβλεψη του κόστους της προσπάθειας ανάπτυξης λογισμικού έγκαιρα, δηλαδή σε πρώιμο στάδιο της ανάπτυξης. Βασίζεται στα χαρακτηριστικά του έργου όπως το μέγεθος, η πολυπλοκότητα, το περιβάλλον ανάπτυξης, η γλώσσα, η αξιοπιστία στις απαιτήσεις του συστήματος και στο προσωπικό.

Τα μοντέλα πρόβλεψης που αφορούν τη διαδικασία ανάπτυξης ενός έργου λογισμικού και ο ακριβής υπολογισμός, είναι ιδιαίτερα σημαντικά και ενδιαφέρουν ένα ευρύ σύνολο ανθρώπων συμπεριλαμβανομένου τους ιδιοκτήτες, τους πελάτες, τους δημιουργούς των συστημάτων και τους διαχειριστές (managers). Τα μοντέλα αυτά μπορούν να χρησιμοποιηθούν για την παραγωγή προτάσεων, για τη διαπραγμάτευση συμβολαίων, για την μελλοντική πρόβλεψη του κόστους, για το καλύτερο συντονισμό, παρακολούθηση και έλεγχο των έργων αλλά και της διαχείρισής τους. Αν έστω αυτά τα μοντέλα μπορούν να συνεισφέρουν μια ελαφρά βελτιωμένη πρόβλεψη του κόστους, της σχεδόν χαοτικής και δυναμικά μεταβαλλόμενης διαδικασίας ανάπτυξης λογισμικού, τότε θα προσφέρουν σημαντικά στα οικονομικά ωφέλη και στη σωστή διαχείριση των έργων.

Είναι σημαντικό να αναφέρουμε, ότι ο υπολογισμός του κόστους δεν πρέπει να υπερεκτιμάται (overestimate) ούτε να υποεκτιμάται (underestimate), αλλά, πρέπει να είναι ακριβής, καθώς έχει προφανή αντίκτυπο σε όλα τα στάδια του κύκλου ανάπτυξης

λογισμικού. Πιο συγκεκριμένα η υπερεκτίμηση του υπολογισμού του κόστους ανάπτυξης λογισμικού μπορεί να προκαλέσει:

- Λανθασμένη επιλογή έργων,
- Λανθασμένη κατανομή των πόρων σε ένα έργο,
- Μη πραγματοποίηση συμβολαίων λόγω υψηλών κοστών,
- Χάσιμο ευκαιριών και χρημάτων,
- Χαμηλή παραγωγικότητα των προγραμματιστών.

Ενώ η υποεκτίμηση του υπολογισμού του κόστους ανάπτυξης λογισμικού μπορεί να προκαλέσει:

- Λανθασμένη επιλογή έργων που ξεπερνούν τελικά τα όρια του κόστους,
- Μείωση της ποιότητας του προϊόντος,
- Λανθασμένη κατανομή των πόρων σε ένα έργο,
- Αργοπορημένη τελικά παράδοση του έργου.

2.4 Δυσκολία του Υπολογισμού του Κόστους Ανάπτυξης Λογισμικού

Η διεξαγωγή ακριβούς και αξιόπιστης πρόβλεψης του υπολογισμού του κόστους που χρειάζεται για την ανάπτυξη ενός λογισμικού είναι προκλητική, ιδιαίτερα πολύπλοκη και δύσκολη διεργασία. Ακόμα, για να θεωρείται χρήσιμη η πρόβλεψη για τους ενδιαφερόμενους, ιδιαίτερα για μεγάλης κλίμακας έργα, εξυπακούεται εκτός από την ακριβή αριθμητικά πρόβλεψη και η έγκαιρη πρόβλεψη, στα αρχικά στάδια της ανάπτυξης.

Η δυσκολία οφείλεται στην μεγάλη πολυπλοκότητα και στην μοναδικότητα της διαδικασίας ανάπτυξης λογισμικού, καθώς ποτέ δύο συστήματα ή έργα δεν είναι πανομοιότυπα. Ακόμα, η γνώση μας είναι περιορισμένη για την πολύπλοκη, δυναμική διαδικασία της ανάπτυξης λογισμικού και για τις συνεχώς μεταβαλλόμενες συσχετίσεις των παραγόντων που επηρεάζουν και προκαλούν τις διαφοροποιήσεις στην παραγωγικότητα. Η πολυπλοκότητα του υπολογισμού του κόστους αυξάνεται, καθώς τα χαρακτηριστικά του έργου που λαμβάνονται υπόψιν και που εμπλέκονται στην διαδικασία αυξάνονται, όπως είναι για παράδειγμα το μέγεθος, το περιβάλλον της ανάπτυξης, η γλώσσα ανάπτυξης, η αξιοπιστία στις απαιτήσεις του συστήματος και το προσωπικό. Η δυσκολία του υπολογισμού εντείνεται ακόμα περισσότερο, αφού στην φάση της ανάπτυξης του λογισμικού, εμπλέκονται διάφοροι συσχετιζόμενοι παράγοντες, οι οποίοι επηρεάζουν την προσπάθεια ανάπτυξης, ενώ οι συσχετίσεις αυτές

είναι δύσκολο να μελετηθούν και να κατανοηθούν. Ακόμα, η έλλειψη εκπαιδευμένου προσωπικού που να κάνει τους σωστούς υπολογισμούς και να έχει εμπειρία στη διαδικασία αυτή εντείνει ακόμα περισσότερο την δυσκολία υπολογισμού του προβλήματος.

Ένα από τα μεγαλύτερα προβλήματα που αντιμετωπίζουμε είναι η έλλειψη ποιοτικών συνόλων δεδομένων που να περιέχουν αριθμητικά το κόστος και τους πόρους που ξοδεύονται για να αναπτυχθεί ένα έργο. Οι πληροφορίες αυτές, θα μπορούσαν να συλλεχτούν από περασμένα έργα που χρειάστηκαν τους πόρους αυτούς για την διεξαγωγή τους, αλλά και θα πρέπει να υπολογιστούν από την αρχή από τους ειδικούς στην ανάπτυξη λογισμικού πριν ακόμα αναλάβουν ένα έργο. Ακόμα, οι υπολογισμοί που γίνονται, βασίζονται πάνω στην προσωπική μνήμη και εμπειρία των διαχειριστών ενός οργανισμού ή μιας επιχείρησης, έτσι δεν μπορούν να θεωρηθούν αντικειμενικές, ενώ υπάρχει και το φαινόμενο της μη ύπαρξης εμπιστοσύνης των εταιριών προς τις ερευνητικές ομάδες, έτσι ώστε με ευκολία να διοχετεύσουν τόσο εμπιστευτικές πληροφορίες προς τα έξω. Επίσης, χρησιμοποιούνται άτυπες διαδικασίες και μέθοδοι, αφού δεν υπάρχουν επίσημοι, σταθεροί, καθολικοί ή αριθμητικοί κανόνες τυποποιημένοι για την διεξαγωγή των υπολογισμών αυτών. Άρα, είναι επιτακτική η ανάγκη εύρεσης ανθρώπων με μεγάλη γνώση και εμπειρία σε πολλά και ποιοτικά έργα ανάπτυξης λογισμικού, έτσι ώστε με βάση την προσωπική τους έμπειρη κρίση να λαμβάνονται υπόψιν οι κατάλληλες μετρικές και οι παράγοντες που επηρεάζουν τους υπολογισμούς. Να λαμβάνουν σωστές αποφάσεις στη διαδικασία και με την χρήση ορθολογιστικών τεχνικών, οι μετρήσεις που διεξάγονται να είναι αξιόπιστες και συνεπής, με αποτέλεσμα όταν χρησιμοποιηθούν για πρόβλεψη να έχουμε όσο το δυνατόν καλύτερα αποτελέσματα. Όπως είναι φυσικό η εύρεση τέτοιων ανθρώπων είναι δύσκολη, συχνά αδύνατη, και έτσι η διαδικασία του υπολογισμού παρουσιάζει πολλά προβλήματα και ανυπέρβλητες δυσκολίες.

Αξίζει να αναφέρουμε ένα από τα σημαντικότερα προβλήματα που εντείνουν ακόμη περισσότερο την δυσκολία του ακριβή υπολογισμού του κόστους ανάπτυξης λογισμικού, που είναι η συλλογή ομογενών και αξιόπιστων δεδομένων, δηλαδή έργων που έγιναν σε ίδια περιβάλλοντα εργασίας. Υπάρχει μεγάλη έλλειψη ποιοτικών ιστορικών στοιχείων μέτρησης του κόστους ανάπτυξης έργων σε μια βάση δεδομένων, και αν υπάρχει, δεν είναι εύκολα διατεθειμένη στο κοινό. Επίσης, τα δεδομένα αυτά είναι πάντοτε υπό αμφισβήτηση αν θα είναι ποιοτικά αξιόλογα και αν θα δουλεύουν για κάποιο υλοποιημένο μοντέλο, αφού είναι συνυφασμένα με στοιχεία όπως είναι η

αποδοτικότητα των προγραμματιστών, το περιβάλλον εργασίας, η εμπειρία, η συνεργασία και η επικοινωνία της ομάδας, ο χώρος και ο χρόνος διεξαγωγής, η τεχνολογική υποστήριξη κ.α. Ακόμα, η συλλογή δεδομένων που θα χρησιμοποιηθούν για τον υπολογισμό του κόστους ανάπτυξης είναι μια ακριβή και χρονοβόρα διαδικασία. Τα δεδομένα είναι διασκορπισμένα και συνήθως όταν βρεθούν, έχουν μικρή ποιότητα και ποσότητα, με αποτέλεσμα να είναι ακόμα πιο δύσκολη η διαμόρφωση θεωριών και η κατασκευή συστημάτων υποστήριξης αποφάσεων με ακριβή αποτελέσματα.

Συμπερασματικά, αφού οι δυσκολίες οφείλονται στην ύπαρξη μεγάλης ποικιλίας πρακτικών, μεθόδων μέτρησης, συλλογής ποσοτικά και ποιοτικά ικανοποιητικών συνόλων δεδομένων και τρόπων μοντελοποίησης, είναι επιτακτική η ανάγκη της συστηματικής προσέγγισης της μεθοδολογίας διεξαγωγής του υπολογισμού του κόστους ανάπτυξης λογισμικού. Ακολούθως, με την επανάληψη του υπολογισμού με τον σωστό τρόπο, μπορούμε να αναπτύξουμε και να υποστηρίξουμε την διαδικασία αυτή με ειδικές τεχνικές, μοντέλα ή / και εργαλεία.

Κεφάλαιο 3

Ιστορικά Στοιχεία του Κόστους Ανάπτυξης Λογισμικού

3.1 Εισαγωγή

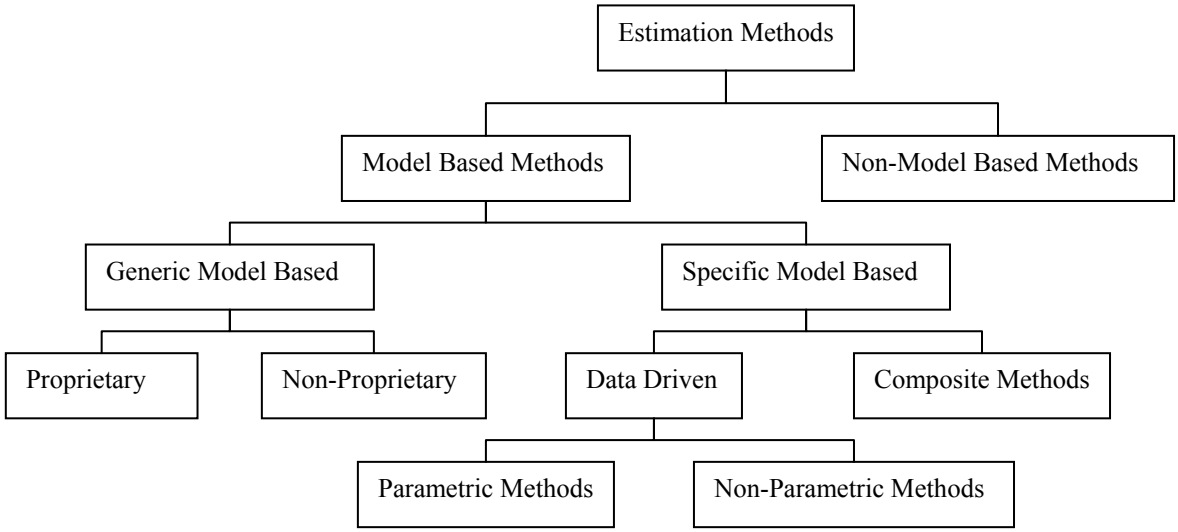
Ο υπολογισμός του κόστους ανάπτυξης λογισμικού άρχισε να απασχολεί τον χώρο της Τεχνολογίας Λογισμικού στα τέλη του 1950 και του 1960. Ειδικά όμως κατά τη διάρκεια των τελευταίων τριών δεκαετιών, έχουν αναπτυχθεί και έχουν ερευνηθεί πιο εκτεταμένα διάφορες τεχνικές υπολογισμού για να προσφέρουν βελτιωμένα αποτελέσματα στις προβλέψεις του κόστους ανάπτυξης. Θα ξεκινήσουμε την ιστορική αναδρομή με την περιγραφή μιας κατηγοριοποίησης των μοντέλων υπολογισμού του κόστους ανάπτυξης λογισμικού που προτείνεται και με την περιγραφή μερικών γνωστών τεχνικών υπολογισμού. Μετά την παρουσίαση μερικών σημαντικότερων ερευνών και αποτελεσμάτων, σκοπός είναι η ανάλυση της μεθοδολογίας της έρευνας του υπολογισμού του κόστους ανάπτυξης λογισμικού, με έμφαση στους παράγοντες που επηρεάζουν το κόστος και στα κριτήρια αξιολόγησης των μοντέλων, και τέλος θα αναλύσουμε τις νέες τάσεις και τις επιρροές σχετικών μελετών που αναφέρονται στην βιβλιογραφία και που αποτέλεσαν και τον γνώμονα της πορείας αυτής της διπλωματικής μελέτης.

3.2 Κατηγοριοποίηση των Μοντέλων Υπολογισμού Κόστους

Σκοπός των ερευνητών, αλλά και της μελέτης αυτής, εκτός της εύρεσης επαρκών υπολογιστικών μεθόδων που να υπολογίζουν με ακρίβεια το κόστος

ανάπτυξης λογισμικού, είναι παράλληλα και η κατηγοριοποίηση των μοντέλων υπολογισμού, έτσι ώστε να καταστεί δυνατή η εκτίμηση και η σύγκριση των μοντέλων μεταξύ τους και των αποτελεσμάτων τους.

Στο πιο κάτω σχήμα παρουσιάζεται η προτεινόμενη διάκριση των μεθόδων υπολογισμού του κόστους, το οποίο αναφέρεται στο άρθρο των Lionel C. Briand and Isabella Wieczorek, Resource Estimation in Software Engineering [14]. Είναι ένα σχήμα ιεραρχικό, ξεκινάει από δύο μεγάλες κατηγορίες, αυτές που βασίζονται σε μοντέλα (Model Based Methods) και αυτές που δεν βασίζονται σε μοντέλα (Non-Model Based Methods), ενώ στη συνέχεια αυτά διακρίνονται σε υποκατηγορίες. Μέχρι στιγμής τα σχήματα κατηγοριοποίησης θεωρούνται υποκειμενικά, καθώς δεν έχει υπάρξει κάποια συνολική συμφωνία σχετικά με αυτά.



Σχήμα 3.1: Διάκριση των Μεθόδων Υπολογισμού του Κόστους [14].

Η πρώτη μεγάλη κατηγορία των Model Based Methods γενικά περιλαμβάνει τουλάχιστον μια μέθοδο μοντελοποίησης, ένα μοντέλο και μια μέθοδο εφαρμογής μοντέλου. Συνήθως, ένα μοντέλο υπολογισμού του κόστους της προσπάθειας που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα λογισμικό, παίρνει ένα σύνολο από εισόδους (συνήθως οι παράγοντες παραγωγικότητας και έναν αριθμητικό υπολογισμό του μεγέθους του συστήματος) και παράγει σαν έξοδο έναν υπολογισμό της προσπάθειας (effort) που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο.

Ξεκινώντας από τα General Model Based, στα οποία εμπερικλείονται τα μοντέλα εκείνα που μπορούν γενικά να εφαρμοστούν σε πολλά διαφορετικά περιβάλλοντα (context) και τα οποία διαχωρίζονται σε δύο κατηγορίες:

- Proprietary:

  Είναι οι μέθοδοι μοντελοποίησης και τα μοντέλα εκείνα, τα οποία δεν είναι πλήρως τεκμηριωμένα και δεν βρίσκονται για διάθεση του κοινού.

- Non-Proprietary:

  Είναι οι μέθοδοι μοντελοποίησης και τα μοντέλα εκείνα τα οποία είναι πλήρως τεκμηριωμένα και βρίσκονται για διάθεση του κοινού.

Παραδείγματα τέτοιων μεθόδων αποτελούν τα COCOMO I, COCOMO II και SLIM.

Στη συνέχεια, στο σχήμα κατηγοριοποίησης υπάρχει η κατηγορία Specific Model Based, στις οποίες μεθόδους περιλαμβάνονται τα τοπικά εκείνα μοντέλα των οποίων η εγκυρότητα είναι άμεσα συνδεδεμένη και εξαρτάται από το περιβάλλον (context) στο οποίο βρίσκονται. Διαχωρίζονται σε δύο κατηγορίες:

- Data Driven

  Είναι οι μέθοδοι που βασίζονται στην ανάλυση δεδομένων, δηλαδή στα μοντέλα που είναι αποτέλεσμα και εξάγονται βασικά από τα δεδομένα. Παραδείγματα τέτοιων μεθόδων αποτελούν τα CART, OSR, ANOVA, OLS και Robust Regression. Διακρίνονται σε δύο ακόμα υποκατηγορίες:

  - Parametric Methods: οι οποίες απαιτούν τον προκαθορισμό της λειτουργικής συσχέτισης μεταξύ των χαρακτηριστικών του έργου και του κόστους και χρησιμοποιώντας την, προσπαθεί να προσαρμόσει τα υποκείμενα δεδομένα με τον καλύτερο δυνατό τρόπο,
  - Non-Parametric Methods: αποτελούνται από τα μοντέλα εκείνα που δεν απαιτούν προκαθορισμό της λειτουργικής συσχέτισης μεταξύ των χαρακτηριστικών του έργου και του κόστους, αν και σε κάποιο σημείο θα λάβουν υπόψιν κάποιες υποθέσεις.

- Composite Methods

  Είναι τα μοντέλα που κατασκευάζονται συνδυάζοντας την άποψη ειδικών, την εξειδικευμένη γνώση και τεχνικές που βασίζονται στην ανάλυση δεδομένων, δηλαδή, σε μοντέλα εξαγόμενα από τα δεδομένα. Η μέθοδος περιγράφει τον τρόπο που μπορεί να εφαρμοστεί και να κατασκευαστεί ένα τελικό βέλτιστο υπολογιστικό

μοντέλο. Παραδείγματα αυτής της μεθόδου αποτελούν τα COBRA, Analogy Based Methods.

Η δεύτερη μεγάλη κατηγορία είναι αυτή των Non-Model Based μεθόδων, η οποία αποτελείται από τον συνδυασμό μιας ή περισσοτέρων τεχνικών, ενώ καθορίζεται για αυτές ο τρόπος εφαρμογής τους κάτω από ένα συγκεκριμένο περιβάλλον. Οι μέθοδοι αυτοί δεν περιλαμβάνουν την κατασκευή μοντέλων, αλλά απλά την διεξαγωγή ενός άμεσου υπολογισμού. Συνήθως, οι μέθοδοι αυτοί καταστούν απαραίτητη την συμβουλή από έναν ή περισσότερους ειδήμονες, έτσι ώστε να μπορεί να παραχθεί ένας υποκειμενικός υπολογισμός της προσπάθειας ανάπτυξης λογισμικού. Ο υπολογισμός αυτός μπορεί να πραγματοποιηθεί με δύο προσεγγίσεις, την κάτω-προς-τα-πάνω (bottom-up) και την πάνω-προς-τα-κάτω (top-down).

- Top-down: η μέθοδος αυτή αφορά τον υπολογισμό της προσπάθειας για ολόκληρο το έργο και στη συνέχεια μοιράζεται αυτός ο υπολογισμός ανάμεσα σε όλα τα μέρη (components) και / ή δραστηριότητες του συστήματος.
- Bottom-up: η μέθοδος αυτή αφορά τον υπολογισμό της κάθε δραστηριότητας ή / και του κάθε μέρους του συστήματος ξεχωριστά και το συνολικό κόστος της προσπάθειας είναι ο συνδυασμός των ξεχωριστών αυτών πράξεων.

Στη συνέχεια θα παρουσιάσουμε και θα μελετήσουμε επιλεκτικά μερικά παραδείγματα από τις σημαντικότερες και πιο γνωστές μεθόδους που αναφέραμε πιο πάνω.

3.3 Γνωστές Τεχνικές Υπολογισμού του Κόστους Ανάπτυξης Λογισμικού

3.3.1 Σύντομη Περιγραφή Γνωστών Τεχνικών Υπολογισμού του Κόστους

Σκοπός είναι η παρουσίαση των πιο κλασικών τεχνικών που εμφανίζονται πιο συχνά στην προτεινόμενη βιβλιογραφία που μελετήθηκε. Οι μέθοδοι που χρησιμοποιούνται για τον υπολογισμό του κόστους παραγωγής έργου ενός λογισμικού ποικίλουν από Αλγοριθμική Μοντελοποίηση του Κόστους, Εξειδικευμένη Κρίση και Γνώση, Υπολογισμός με Αναλογία, Parkinson's Law, Pricing to Win μέχρι το Κλασικό Regression, Τεχνητά Νευρωνικά Δίκτυα, Ασαφή Λογισμό και τελευταία Γενετικό Προγραμματισμό.

Η Αλγοριθμική Μοντελοποίηση του Κόστους (Algorithmic Cost Modelling) είναι η ανάπτυξη μοντέλου χρησιμοποιώντας από το παρελθόν πληροφορίες όσον

αφορά το κόστος και το οποίο συσχετίζεται συνήθως με κάποια μετρική (συνήθως αυτήν του μεγέθους). Γίνεται μια εκτίμηση της μετρικής (μέγεθος του έργου) και το μοντέλο προβλέπει την προσπάθεια που θα χρειαστεί για να παραχθεί το έργο αυτού του μεγέθους. Μπορεί να γίνει αφού είναι γνωστά τα χαρακτηριστικά και τα κόστη σε αριθμητικούς υπολογισμούς, ήδη τελειωμένων έργων.

Υπάρχουν ένα μεγάλο σύνολο από Αλγοριθμικά Μοντέλα Υπολογισμού. Εδώ κρίθηκε ότι είναι αναγκαίος ο συνδυασμός αναλυτικών εξισώσεων, προσαρμογής δεδομένων και εξειδικευμένης γνώσης. Μια από τις πιο καλές και γνωστές μεθόδους, προτάθηκε από τον Boehm το 1981 και είναι το COnstructive COst MOdel (COCOMO). Προσφέρει εξισώσεις που έχουν το μέγεθος του συστήματος σαν το πρωταρχικό στοιχείο που καθορίζει την προσπάθεια. Η προβλεπόμενη προσπάθεια ανάπτυξης λογισμικού τότε προσαρμόζεται για να φιλοξενήσει την επιρροή άλλων επιπλέον 15 παραγόντων που επηρεάζουν το κόστος. Άλλα παραδείγματα που ακολουθούν την ίδια μεθοδολογία είναι το SLIM (Putnam, 1978) και COPMO (Conte et al., 1986). Η μεθοδολογία αυτή ανήκει στην κατηγορία των Generic Model Based και Non-proprietary μεθόδων. Θα μελετήσουμε αναλυτικότερα και μεθόδους της κατηγορίας Specific Model Based και Data Driven, όπως είναι το CART και στην κατηγορία των Composite Methods η Analogy-Based μεθοδολογία, καθώς είναι και οι πιο γνωστές.

Η Εξειδικευμένη Κρίση και Γνώση (Expert Judgement) συμπεριλαμβάνει τις τεχνικές εκείνες που χρειάζονται την γνώμη και την βοήθεια ενός έμπειρου, ειδικού πάνω στην τεχνική, οι οποίοι εκτιμούν το κόστος. Αυτές οι εκτιμήσεις συγκρίνονται και συζητιούνται και η διαδικασία γίνεται σε επανάληψη μέχρι να υπάρξει κάποια συμφωνία στις εκτιμήσεις.

Ένα παράδειγμα τέτοιας μεθόδου, σε συνδυασμό με Αλγοριθμική Μοντελοποίηση είναι η τεχνική COBRA (Cost Estimation Benchmarking and Risk Analysis). Αποτελείται από δύο μέρη, αρχικά ένα causal model το οποίο λαμβάνει υπόψιν παράγοντες που επηρεάζουν το κόστος και λαμβάνεται με ερωτήσεις που απευθύνονται σε έμπειρους διαχειριστές έργων (project managers) και στην συνέχεια χρησιμοποιεί δεδομένα του παρελθόντος, από παρόμοια έργα για να εξαγάγει την συσχέτιση μεταξύ του κόστους και της παραγωγικότητας.

Ο Υπολογισμός Βασισμένος σε Αναλογίες (Estimation by Analogy), αφορά τον χαρακτηρισμό των έργων μέχρι τη χρονική στιγμή που ζητείται να εκτελεστεί ο υπολογισμός. Ένα έργο δηλαδή, θα μπορέσει να χαρακτηριστεί ανάλογα με τον αριθμό

των εισόδων, των εξόδων, ανάλογα με τον αριθμό των οθονών του συστήματος ή και την γλώσσα προγραμματισμού του και έτσι, η επιλογή των μεταβλητών που θα ληφθούν υπόψιν εξαρτάται από τα δεδομένα που είναι διαθέσιμα. Όταν γίνουν αυτά, τότε θα καταστεί δυνατή η πρόβλεψη ενός άγνωστου, νέου έργου κοιτάζοντας για παρόμοια ή ανάλογα έργα αποθηκευμένα στα δεδομένα. Ακόμα μια από τις πιο γνωστές μεθόδους που αναφέρονται στα ιστορικά της Τεχνολογίας Λογισμικού είναι η CART.

Η μέθοδος CART (Classification and Regression Trees), προτείνει την χρήση δύο τύπων δέντρων αποφάσεων. Τα Classification Trees, τα οποία παράγουν μια πρόβλεψη για μια κατηγορηματική (nominal, ordinal) μεταβλητή και τα Regression Trees, τα οποία παράγουν πρόβλεψη ανάμεσα σε ένα διάστημα ή σε μια κλίμακα. Διακρίνει τα έργα σε κατηγορίες, ως προς την παραγωγικότητα με βάση μια συλλογή από κανόνες. Ακόμα ένα παράδειγμα στην ίδια κατηγορία είναι το OSR (Optimized Set Reduction), το οποίο με βάση μιας συλλογής από λογικές εκφράσεις καθορίζει τα σύνολα δεδομένων εκπαίδευσης που δίνουν τα πιο βέλτιστα αποτελέσματα πρόβλεψης.

Ο Νόμος του Parkinson, υποστηρίζει ότι η δουλειά είναι μια επέκταση για να γεμίσει ο διαθέσιμος χρόνος και το κόστος καθορίζεται σε σχέση με τους διαθέσιμους πόρους. Αν για παράδειγμα ένα έργο πρέπει να παραδοθεί εντός δώδεκα μηνών και είναι διαθέσιμοι για εργασία πέντε άνθρωποι, τότε η προσπάθεια που πρέπει να καταβληθεί υπολογίζεται σε 60 άνθρωπο-μήνες (person-months).

Η τεχνική Pricing to Win, υπολογίζει ότι το κόστος του λογισμικού καθορίζεται από το ποσό που διαθέτει ο πελάτης να ξοδέψει για το έργο και έτσι η υπολογιζόμενη προσπάθεια δεν εξαρτάται από την λειτουργικότητα του συστήματος, αλλά από το χρηματικό ποσό που διαθέτει ο πελάτης.

Στην συνέχεια, θα παρουσιάσουμε και θα συγκρίνουμε τα αποτελέσματα αυτών των μεθόδων και θα καταλήξουμε στην χρήση των Data-Driven και Non-parametric Methods. Οι τεχνικές των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγορίθμων, αν και αναφέρονται σε αυτό το τμήμα, δεν παρουσιάζονται αφού αναλύονται σε λεπτομέρεια τόσο οι τεχνικές όσο και εμπειρικές μελέτες σε επόμενο τμήμα της μελέτης.

3.3.2 Παρατηρήσεις και Συγκρίσεις Τεχνικών

Στόχος αυτού του τμήματος, μετά την σύντομη περιγραφή των τεχνικών που αναφέρονται πιο συχνά στην βιβλιογραφία, είναι η παρουσίαση των συμπερασμάτων

μερικών πιο γνωστών ερευνών που επίσης αναφέρονται στην βιβλιογραφία και οι οποίες χρησιμοποίησαν τις κλασικές αυτές τεχνικές. Τα συμπεράσματα των ερευνών αξιολογούν ως την πιο βέλτιστη λύση για τον υπολογισμό του κόστους ανάπτυξης λογισμικού, αλλά και γενικά των πολύπλοκων προβλημάτων πρόβλεψης, την επιλογή χρήσης της τεχνικής των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγορίθμων.

Ξεκινώντας από το COCOMO, συμπερασματικά μπορεί να χρησιμοποιηθεί για έργα στα οποία έχουν αναγνωριστεί τα βασικά μέρη, αφού ο υπολογισμός γίνεται με βάση τα μέρη αυτά. Τα αποτελέσματα που λήφθηκαν από το ενδιάμεσο (intermediate) και το λεπτομερές (detailed) COCOMO είναι πολύ καλά, αν και ίσως να υποφέρουν από αποστήθιση (over fitting), ενώ συγκριτικά τα αποτελέσματα του βασικού (basic) COCOMO είναι αρκετά φτωχά. Το μειονέκτημα της μεθόδου είναι το γεγονός ότι δεν μπορεί το μοντέλο του COCOMO να χρησιμοποιηθεί σε άλλα περιβάλλοντα, λόγω του ότι οι συσχετίσεις μεταξύ των παραγόντων και το κόστος ισχύουν μόνο υπό ένα συγκεκριμένο περιβάλλον (Kitchenham and Taylor, 1985).

Το COCOMO II έχει παρουσιάσει σημαντικές βελτιστοποιήσεις σε σχέση με το COCOMO I, όπως την μέτρηση του μεγέθους με βάση KLOC, Function Points ή Object Points. Μπορεί να χρησιμοποιηθεί ξανά, μειώνει τις δυσκολίες που αφορούν την διαφορετική κλίμακα των δεδομένων και οι παράγοντες που επηρεάζουν το κόστος έχουν αναπροσαρμοστεί, έχουν καθοριστεί κάποιοι νέοι, έχουν αφαιρεθεί άλλοι, έχουν γίνει δηλαδή αλλαγές σε σχέση με πριν (Boehm et al., 1996) και επομένως παρατηρήθηκαν βελτιώσεις στα αποτελέσματα.

Η τεχνική COBRA (Cost Estimation Benchmarking and Risk Analysis) εμφανίζει πλεονεκτήματα όσον αφορά το ότι μπορεί να εφαρμοστεί σε περιβάλλοντα στα οποία είναι διαθέσιμα λίγα δεδομένα. Η εξειδικευμένη γνώση που χρησιμοποιήθηκε στο μοντέλο μπορεί να χρησιμοποιηθεί ξανά σε μια νέα μοντελοποίηση, ενώ τα μεγαλύτερα μειονεκτήματα της τεχνικής είναι η ανάγκη που υπάρχει για εύρεση ατόμων με μεγάλη πείρα, το οποίο είναι δύσκολο να γίνει. Ακόμα και αν βρεθούν τα άτομα αυτά, η λήψη και η απομόνωση της εξειδικευμένης αυτής γνώσης είναι δύσκολο να γίνει, καθώς χρειάζεται μεγάλη εμπειρία και εκπαίδευση.

Η τεχνική των Regression Trees παρατηρούμε ότι κάτω από συγκεκριμένο περιβάλλον, όπου τα δεδομένα είναι κατηγορηματικά (nominal, ordinal), αν οι παράγοντες που επηρεάζουν το κόστος αλληλεπιδρούν μεταξύ τους και συσχετίζονται δυνατά (strongly) και μη γραμμικοί, τότε οι παράγοντες λειτουργούν σωστά. Είναι

γενικά μια καλή εναλλακτική τεχνική μοντελοποίησης λόγω της απλότητας και της διεξαγωγής ανάμεικτων αποτελεσμάτων. Ενώ όσες τεχνικές χρησιμοποίησαν OLS (Ordinary Least-Square Regression) είχαν σχετικά ικανοποιητική επίδοση αν και δεν έχουν γίνει πολλές συγκρίσεις με την μέθοδο αυτή και άλλες τεχνικές, εντούτοις είναι θετική η ευκολία που παρουσιάζουν στην διερμηνεία των αποτελεσμάτων.

3.4 Μεθοδολογία Έρευνας Υπολογισμού του Κόστους Ανάπτυξης Λογισμικού

Ο υπολογισμός του κόστους ανάπτυξης λογισμικού είναι η διαδικασία διεξαγωγής πρόβλεψης που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα λογισμικό σύστημα (software system). Η πρόβλεψη αυτή είναι πολύ σημαντική και χρήσιμη κατά την διάρκεια του κύκλου ζωής ενός έργου, ενώ είναι ιδιαίτερα χρήσιμη και αναγκαία στα πρώτα στάδια του κύκλου ανάπτυξης. Όταν η πρόβλεψη του κόστους ανάπτυξης λογισμικού προσεγγίσει τις πραγματικές τιμές, τότε μπορεί να βοηθήσει στην λήψη σωστών διοικητικών και διαχειριστικών αποφάσεων τόσο στην αρχή, όσο και κατά την διάρκεια της ανάπτυξης του έργου. Στο σημείο αυτό ακολουθεί η περιγραφή των παραγόντων που επηρεάζουν τον υπολογισμό του κόστους ανάπτυξης λογισμικού και η περιγραφή των κριτηρίων αξιολόγησης που λαμβάνονται υπόψιν τόσο στις εμπειρικές μελέτες που αναφέρονται στην βιβλιογραφία, τόσο και στην παρούσα μελέτη. Η μεθοδολογία της έρευνας συμπληρώνεται με τα κριτήρια αξιολόγησης των αποτελεσμάτων που εξάγονται από τα Υπολογιστικά Μοντέλα.

3.4.1 Παράγοντες που επηρεάζουν το Κόστος Ανάπτυξης Λογισμικού

Η ιστορική ανάλυση και έρευνα για την διαδικασία ανάπτυξης λογισμικών έργων που αναπτύχθηκαν στο παρελθόν, ανέδειξε ότι το κόστος επηρεάζεται και έχει άμεση συσχέτιση με συγκεκριμένους μετρήσιμους παράγοντες. Αυτή η παρατήρηση αναχαίτισε την μελέτη και την δημιουργία ενός ευρύ συνόλου από μοντέλα, τα οποία χρησιμοποιούνται για να αποκτήσουν πρόσβαση, πρόβλεψη και έλεγχο στο κόστος ανάπτυξης λογισμικού βασισμένα σε πραγματικό χρόνο. Τα μοντέλα αυτά, συνήθως προτείνουν έναν ή περισσότερους μαθηματικούς αλγόριθμους για να υπολογίσουν το κόστος ανάπτυξης λογισμικού, σε συνάρτηση με ένα σύνολο από μεταβλητές.

Μερικοί από τους βασικότερους παράγοντες που επηρεάζουν την παραγωγικότητα και το κόστος της ανάπτυξης ενός λογισμικού από τους

κατασκευαστές συγκεντρώθηκαν στην συνέχεια, όπως περιγράφονται στο βιβλίο Software Engineering, Ian Sommerville [1]. Αρχικά, η ποιότητα της διαδικασίας ανάπτυξης λογισμικού παίζει καθοριστικό ρόλο στο καθορισμό του κόστους. Όσο καλύτερη είναι ποιοτικά η διαδικασία αυτή, τόσο μεγαλύτερη είναι η παραγωγικότητα και κατά συνέπεια τόσο μικρότερο το κόστος. Η γνώση και η πείρα στο συγκεκριμένο πεδίο του λογισμικού που θα αναπτυχθεί είναι ένα βασικό στοιχείο που συμβάλει στην βελτίωση της διαδικασίας ανάπτυξης λογισμικού. Μεγαλύτερη πείρα συνεπάγεται και λιγότερο κόστος. Το μέγεθος του έργου που θα αναπτυχθεί παίζει σημαντικό ρόλο στο κόστος, αφού όσο μεγαλύτερο είναι το μέγεθος τόσο περισσότερος χρόνος απαιτείται για την επικοινωνία της ομάδας και άρα τόσο λιγότερος χρόνος απομένει για την ανάπτυξη της εφαρμογής. Ακόμα, η τεχνολογική υποστήριξη είναι ένας σημαντικός παράγοντας που μπορεί να βελτιώσει την παραγωγικότητα και κατ' επέκταση να μειώσει το κόστος ανάπτυξης λογισμικού. Τέλος, όσο πιο ιδανικό είναι το περιβάλλον εργασίας τόσο μεγαλύτερη είναι η απόδοση και η παραγωγικότητα της ομάδας εργασίας και επομένως τόσο λιγότερο είναι το κόστος.

Ο πρωταρχικός και σημαντικότερος παράγοντας που παρουσιάζεται να επηρεάζει το κόστος ανάπτυξης λογισμικού, αλλά και μοναδικός παράγοντας που επιλέχτηκε να χρησιμοποιηθεί σε αυτή την διπλωματική εργασία, λόγω του ότι είναι ένας παράγοντας που μπορεί να υπολογιστεί από τα αρχικά στάδια της εφαρμογής, με σχετική ακρίβεια, είναι το μέγεθος του λογισμικού που θα αναπτυχθεί. Υπάρχουν δύο συνήθεις τρόποι που χρησιμοποιούνται για να μετρηθεί το μέγεθος ενός λογισμικού: οι γραμμές του κώδικα (lines of code) και τα λειτουργικά σημεία (function points).

Ο πιο συνηθισμένος τρόπος μέτρησης του μεγέθους του κώδικα πηγής (source code) είναι οι γραμμές του κώδικα (lines of code (LOC)). Συχνά εμφανίζεται η σύντμηση NCLOC, η οποία χρησιμοποιείται για να αντιπροσωπεύσει τον κώδικα πηγής χωρίς σχόλια, ενώ μερικές φορές εμφανίζεται και σαν λειτουργικές γραμμές κώδικα (effective lines of code) με την σύντμηση ELOC. Αξίζει να αναφέρουμε ότι το μέγεθος του κώδικα των σχολίων μπορεί να χρησιμοποιηθεί και σαν έγκυρη μετρική, αν θεωρείται η τεκμηρίωση του κώδικα σαν μέρος της προγραμματιστικής προσπάθειας που θα εκπονηθεί. Συμπερασματικά, η σύντμηση CLOC χρησιμοποιείται για να αντιπροσωπεύσει το μέγεθος των γραμμών που αντιπροσωπεύουν σχόλια, την τεκμηρίωση του κώδικα. Συνεπώς, αν υπολογιστούν τα NCLOC και CLOC μπορούμε να ορίσουμε σαν τον συνολικό μέγεθος γραμμών κώδικα:

$$total\_length(LOC) = NCLOC + CLOC$$

Η σύντμηση KLOC χρησιμοποιείται για να αντιπροσωπεύσει τις χιλιάδες αριθμού γραμμών κώδικα.

Εντούτοις, αν και οι γραμμές του κώδικα είναι η πιο γνωστή και κοινά αποδεκτή μέθοδος που χρησιμοποιείται στις περισσότερες μελέτες για τον υπολογισμό του κόστους ανάπτυξης λογισμικού, παρουσιάζει κάποιους περιορισμούς τους οποίους αναφέρουμε στη συνέχεια. Ο αριθμός των γραμμών του κώδικα που θα παραχθεί είναι άμεσα συνυφασμένος με την γλώσσα προγραμματισμού και το προσωπικό στιλ προγραμματισμού της ομάδας ανάπτυξης. Ακόμα, μια έμπειρη εταιρία που έχει υψηλό βαθμό ωριμότητας (capability maturity level) μπορεί να γνωρίζει περίπου το μέγεθος του έργου που αναλαμβάνει ή και που μπορεί να αναλάβει, εντούτοις δεν μπορεί να είναι γνωστή η ακριβής τιμή της, από την αρχή του κύκλου ζωής του έργου, πριν τελειώσει η υλοποίηση. Αν και έχουν προταθεί πρότυπα (standards) για το πως θα πρέπει να γίνεται η καταμέτρηση των γραμμών του κώδικα σε κάθε μια γλώσσα προγραμματισμού, έτσι ώστε να είναι συγκρίσιμες (Park, 1992) δεν έγιναν ευρέως αποδεκτές ακόμα και έτσι δεν χρησιμοποιούνται.

Τα λειτουργικά σημεία (function points (FP)) χρησιμοποιούνται για να καθορίσουν τον συνολικό αριθμό των λειτουργιών σε ένα σύστημα. Υπολογίζονται αφού πρώτα μετρηθούν τα μη-προσαρμοσμένα λειτουργικά σημεία (unadjusted function point count (UFC)) των πιο κάτω σημείων:

- Εξωτερικοί Είσοδοι

  Αντιπροσωπεύουν τα αντικείμενα τα οποία παρέχονται από τον χρήστη και περιγράφουν τα δεδομένα τα οποία ακολουθούνται από κάποιο ξεχωριστό application, όπως είναι για παράδειγμα τα file names και τα menu selections.
- Εξωτερικοί Έξοδοι

  Αντιπροσωπεύουν τα αντικείμενα τα οποία παρέχονται στον χρήστη και τα οποία παράγουν δεδομένα βασισμένα σε ξεχωριστά applications, όπως είναι για παράδειγμα τα reports και τα μηνύματα.
- Εξωτερικές Αναζητήσεις

  Αντιπροσωπεύουν τις εισόδους που είναι διαδραστικές και που εκτελούν μια ερώτηση και αναμένουν στη συνέχεια κάποια απάντηση.
- Εξωτερικά Αρχεία

  Αντιπροσωπεύουν τα περιβάλλοντα διαπροσωπείας, τα οποία μπορούν να διαβαστούν από μηχανή και από άλλα συστήματα.

- Εσωτερικά Αρχεία

  Αντιπροσωπεύουν τα κύρια λογικά αρχεία του συστήματος.

Αφού συλλεχτούν τα δεδομένα σχετικά με τα σημεία αυτά, αξιολογούνται σε σχέση με την πολυπλοκότητά τους, ζυγίζονται δηλαδή ακόμα περισσότερο σαν "simple", "average" ή "complex" ανάλογα με τα χαρακτηριστικά τους, σύμφωνα με τον πίνακα 3.1 (Symons, 1988) [19].

Πίνακας 3.1: Βάρη των Αντικειμένων [19]

	Συναπτικά Βάρη		
Αντικείμενα	**Simple**	**Average**	**Complex**
Εξωτερικοί Είσοδοι	3	4	6
Εξωτερικοί Έξοδοι	4	5	7
Εξωτερικές Αναζητήσεις	3	4	6
Εξωτερικά Αρχεία	7	10	15
Εσωτερικά Αρχεία	5	7	10

Κάθε μέτρηση πολλαπλασιάζεται με το ανάλογο συναπτικό βάρος και τα αποτελέσματα προστίθενται και υπολογίζουν τα μη-προσαρμοσμένα λειτουργικά σημεία (unadjusted function point count (UFC)). Τότε τα προσαρμοσμένα λειτουργικά σημεία (adjusted point count (FP)), υπολογίζονται αν πολλαπλασιαστούν τα μη-προσαρμοσμένα λειτουργικά σημεία με ένα παράγοντα τεχνικής πολυπλοκότητας (technical complexity factor (TFC)), τα οποία επηρεάζονται από τα εξής στοιχεία:

F1 – Συνέπεια σε θέματα back-up και recovery (Reliable back-up and recovery)

F2 – Data communications

F3 – Κατανεμημένες Λειτουργικότητες (Distributed Functions)

F4 – Απόδοση (Performance)

F5 – Heavily used configuration

F6 – Άμεση είσοδος δεδομένων (Online data entry)

F7 – Ευκολία στη Λειτουργία (Operational Ease)

F8 – Άμεσες αλλαγές (Online update)

F9 – Πολυπλοκότητα διαπροσωπείας (Complex interface)

F10 – Πολυπλοκότητα επεξεργασίας (Complex processing)

F11 – Επαναχρησιμοποίηση (Reusability)

F12 – Ευκολία στην εγκατάσταση (Installation ease)

F13 – Multiple sites

F14 – Facilitate change

Κάθε στοιχείο αξιολογείται στα πλαίσια του 0 μέχρι και το 5, όπου το μηδέν υποδεικνύει ότι το στοιχείο δεν επηρεάζει καθόλου το σύστημα ενώ το 5 υποδεικνύει ότι το στοιχείο επηρεάζει και είναι αναπόσπαστο μέρος του συστήματος.

Στη συνέχεια ο παράγοντας τεχνικής πολυπλοκότητας (technical complexity factor (TFC)), υπολογίζεται ως εξής:

$$TCF = 0.65 + 0.01(SUM(F_i))$$

Τέλος, τα λειτουργικά σημεία (function points (FP)) υπολογίζονται ως εξής:

$$FP = UCF * TCF$$

Τα λειτουργικά σημεία (function points (FP)) υπερβαίνουν τους περιορισμούς που αναφέραμε προηγουμένως των γραμμών κώδικα (lines of code (LOC)), όμως παρουσιάζουν κάποιους άλλους. Οι περιορισμοί αφορούν ανακρίβειες στο σχήμα των βαρών, με αποτέλεσμα να μην ακολουθείται από όλες τις έρευνες, αλλά συνήθως να προτείνεται και να εφαρμόζεται κάποιο νέο σχήμα βαρών. Ακόμα, λόγω του γεγονότος ότι τα λειτουργικά σημεία εφαρμόζονται σε Συστήματα Διαχείρισης Πληροφοριών, MIS (Management Information Systems), οι πέντε λειτουργικοί τύποι του πίνακα 3.1, μπορούν να διεξαχθούν από τα Διαγράμματα Σχέσεων-Οντοτήτων ERD (Entity-Relationship Diagram) και τα Διαγράμματα Ροής Δεδομένων (Data Flow Diagram). Όμως, σε περιβάλλοντα ή σε συστήματα πραγματικού χρόνου υπάρχουν μερικά επιπλέον σημεία που έχουν μεγάλη σημασία και τα οποία δεν λαμβάνονται υπόψιν ανάμεσα στους πέντε λειτουργικούς τύπους, όπως είναι για παράδειγμα οι αλγόριθμοι και οι μεταβάσεις από μια κατάσταση σε μια άλλη.

3.4.2 Κριτήρια Αξιολόγησης

Η ικανότητα διεξαγωγής ακριβής ή όχι πρόβλεψης των Μοντέλων Υπολογισμού αξιολογείται με βάση κάποια κριτήρια τα οποία παρουσιάζονται σε αυτό το τμήμα της μελέτης. Τα περισσότερα από τα κριτήρια που αξιολογούν τα μοντέλα υπολογισμού, προτάθηκαν από το Boehm το 1981 και αναφέρονται στην συνέχεια, ενώ

αναλύονται μόνο τα κριτήρια της ακρίβειας τα οποία χρησιμοποιήθηκαν στην διπλωματική αυτή εργασία.

- Definition – Σαφήνεια

  Το μοντέλο πρέπει να καθορίζει ξεκάθαρα το κόστος που υπολογίζει.

- Fidelity – Πιστότητα

  Οι υπολογισμοί του μοντέλου για το κόστος ανάπτυξης λογισμικού πρέπει να πλησιάζουν το πραγματικό κόστος.

- Objectivity – Αντικειμενικότητα

  Το σύνολο των μεταβλητών δεν πρέπει να περιλαμβάνουν υποκειμενικούς παράγοντες των οποίων οι αξιολόγηση γίνεται δύσκολα (όπως για παράδειγμα η πολυπλοκότητα).

- Constructiveness – Εποικοδομητικό

  Ο χρήστης πρέπει να μπορεί να ερμηνεύσει τα αποτελέσματα που δίνει το μοντέλο.

- Detail – Λεπτομέρεια

  Το μοντέλο πρέπει λαμβάνει υπόψη τυχών υποσυστήματα ή άλλες υπομονάδες του συστήματος.

- Stability – Σταθερότητα

  Το μοντέλο πρέπει να μπορεί να διατηρεί τη σταθερότητα.

- Scope – Φάσμα

  Οι εκτιμήσεις που δίνει το μοντέλο πρέπει να αφορούν ολόκληρο το φάσμα από έργα που επιθυμεί ο χρήστης.

- Easy of use – Ευκολία χρήσης

  Οι επιλογές και οι είσοδοι του μοντέλο πρέπει να είναι κατανοητοί και εύκολοι να προσδιοριστούν.

- Prospectiveness – Προοπτική

  Το μοντέλο δεν πρέπει να χρησιμοποιεί πληροφορίες οι οποίες δεν θα είναι γνωστές παρά μόνο αφού τελειώσουν τα έργα.

- Parsimony – Φειδωλότητα

  Το μοντέλο πρέπει να αποφεύγει την χρήση πολύ περιττών παραγόντων ή παραγόντων των οποίων η επιρροή είναι μικρή.

- Accuracy – Ακρίβεια

Η διαδικασία αυτής της αξιολόγησης γίνεται με βάση το κριτήριο της ακρίβειας. Δηλαδή με βάση την διαφορά των τιμών των πραγματικών παρατηρήσεων της προσπάθειας ανάπτυξης λογισμικού σε σύγκριση με τις τιμές των προβλεπόμενων παρατηρήσεων, το αποτέλεσμα της μεθόδου. Οι αξιολογήσεις των τιμών αυτών αποτελούν σημαντικά κριτήρια για την καταλληλότητα των μοντέλων και στην σχετική βιβλιογραφία προτείνονται διάφορα μέτρα κατάλληλα για την αξιολόγηση αυτή. Τα κριτήρια που επιλέχτηκαν για την αξιολόγηση του αποτελέσματος του κόστους ανάπτυξης λογισμικού σε αυτή την διπλωματική εργασία, είναι η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (Correlation Coefficient) και Μέσο Σφάλμα Συσχέτισης MRE (Mean Relative Error). Η επιλογή των κατάλληλων κριτηρίων αξιολόγησης πρέπει να γίνεται ανάλογα με το πως επιθυμεί ο ερευνητής να αξιολογήσει τα αποτελέσματα του, για παράδειγμα αν θέλει να μελετήσει την ακρίβεια των αποτελεσμάτων, την σωστή ανοδική ή πτωτική τάση και εξαρτάται άμεσα από το είδος των δεδομένων που θα εισαχθούν στο μοντέλο. Είναι ιδιαίτερα σημαντικό ο ερευνητής κάθε φορά που χρησιμοποιεί τα κριτήρια αξιολόγησης να είναι προσεκτικός και να ελέγχει, όπου έχει σημασία, την κλίμακα των δεδομένων προτού λάβει τα αποτελέσματα.

Παραθέτουμε το πρώτο κριτήριο αξιολόγησης λάθους την Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE, Normalized Root Mean Squared Error), το οποίο είναι ένα μέτρο αξιολόγησης που ανιχνεύει την ποιότητα των προβλέψεων επικεντρώνοντας καθαρά στην σύγκρισή της με αυτήν ενός κινητού μέσου. Για τον υπολογισμό αυτού του σφάλματος χρησιμοποιούμε τη Ρίζα Μέσου Τετραγωνικού Σφάλματος (RMSE, Root Mean Squared Error) το οποίο ορίζεται ως εξής:

$$RMSE(n) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[x_{pred}(i) - x_{act}(i)\right]^2}$$

Η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE, Normalized Root Mean Squared Error), ορίζεται κανονικοποιώντας την στη συνέχεια, με την απόκλιση των αυθεντικών τιμών:

$$NRMSE(n) = \frac{RMSE(n)}{\sigma_{\Delta}} = \frac{RMSE(n)}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[x_{act}(i) - \bar{x}_n\right]^2}}$$

Αν NRMSE=0, τότε η πρόβλεψη είναι 100% επιτυχής. Ενώ η τιμή NRMSE=1, δείχνει ότι η πρόβλεψη δεν είναι καλύτερη από το να παίρναμε σαν επόμενη προβλεπόμενη τιμή x_{pred} την μέση τιμή των n αυθεντικών δειγμάτων. Το πρόβλημα που αντιμετωπίζει το Normalized RMSE εμφανίζεται όταν η απόκλιση των αυθεντικών τιμών των δειγμάτων είναι ένας πολύ μικρός αριθμός και όταν τα δείγματα είναι τιμές πολύ κοντά στο μέσο της σειράς, με πολύ μικρές διακυμάνσεις. Αυτό το φαινόμενο, έχει ως αποτέλεσμα το NRMSE να παίρνει πολύ ψηλές τιμές, ακόμη και όταν η απόκλιση είναι πολύ μικρή, έως αμελητέα των προβλεπόμενων τιμών από αυτών των πραγματικών τιμών. Συνεπώς, η χρήση του μέτρου αυτού για την αποτίμηση της επιτυχίας των προβλέψεων πρέπει να γίνεται μόνο σε επιλεγμένες περιπτώσεις και για να είμαστε εκ του ασφαλούς πρέπει να συνδυάζεται με άλλα μέτρα σφάλματος.

Επόμενο κριτήριο αξιολόγησης λάθους είναι ο Συντελεστής Συσχέτισης (CC, Correlation Coefficient), ο οποίος μας επιτρέπει να παρακολουθήσουμε την εξέλιξη της τροχιάς των σημείων των πραγματικών τιμών και των προβλεπόμενων, όσων αφορά την συσχέτισή τους. Μπορούμε δηλαδή να ανιχνεύσουμε αν η προβλεπόμενες τιμές ακολουθούν την αυξητική ή την μειωτική εναλλαγή των πραγματικών τιμών. Ο Συντελεστής Συσχέτισης ορίζεται από την εξής εξίσωση:

$$CC = \frac{\sum_{i=1}^{n}\left[\left(x_{act}(i) - \bar{x}_{act,n}\right) - \left(x_{pred}(i) - \bar{x}_{pred,n}\right)\right]}{\sqrt{\left[\sum_{i=1}^{n}\left(x_{act}(i) - \bar{x}_{act,n}\right)^2\right]\left[\sum_{i=1}^{n}\left(x_{pred}(i) - \bar{x}_{pred,n}\right)^2\right]}}$$

Όπου $\bar{x}_{act,n}$ είναι οι μέσες τιμές των n αυθεντικών, πραγματικών δειγμάτων και $\bar{x}_{pred,n}$ οι μέσες τιμές των n προβλεπόμενων.

Οι τιμές που μπορεί να πάρει ο Συντελεστής Συσχέτισης είναι από -1 έως +1. Όταν η τιμή του Συντελεστή Συσχέτισης είναι κοντά στην μονάδα (είτε κοντά στο -1 είτε κοντά στο +1), τότε ερμηνεύεται ότι οι προβλέψεις μιμούνται την κίνηση, την τάση των πραγματικών τιμών σε κλίση. Οι αρνητικές τιμές του Συντελεστή Συσχέτισης ερμηνεύονται σαν τη καθρεφτική αντίδραση των πραγματικών και των προβλεπόμενων τιμών, δηλαδή η προβλεπόμενη σειρά τιμών ακολουθεί τις εναλλαγές της αυθεντικής σειράς τιμών με τρόπο αντίστροφο. Με τρόπο αντίστροφο εννοούμε ότι εκεί που υπάρχει αυξητική τάση στην πραγματική σειρά, υπάρχει μειωτική τάση στην προβλεπόμενη σειρά και το αντίστροφο. Ενώ όταν η τιμή του Συντελεστή Συσχέτισης δεν είναι κοντά στην μονάδα (κοντά στο μηδέν), τότε δείχνει την ανικανότητα της

μεθόδου να παράγει προβλέψεις που να αναπαράγουν την κίνηση της αυθεντικής σειράς, των πραγματικών τιμών.

Ένα από τα πιο διαδεδομένα κριτήρια αξιολόγησης είναι το Μέσο Τετραγωνικό Σφάλμα (MSE, Mean Squared Error), το οποίο δίνει το σφάλμα των προβλέψεων εκφρασμένο ως την μέση τιμή της τετραγωνικής απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές. Το Μέσο Τετραγωνικό Σφάλμα ορίζεται από την εξίσωση:

$$MSE(n) = \frac{1}{n}\sum_{i=1}^{n}\left[x_{act}(i) - x_{pred}(i)\right]^2$$

Όπου n, ο αριθμός των προβλέψεων, $x_{act}(i)$ η αυθεντική πραγματική τιμή του δείγματος i και $x_{pred}(i)$ η προβλεπόμενη τιμή του δείγματος i.

Όπως φαίνεται, το κριτήριο αυτό αξιολόγησης είναι εξαρτώμενο από την κλίμακα των δεδομένων, αφού το λάθος υπολογίζεται από την τιμή της διαφοράς ανάμεσα στις τιμές των δειγμάτων.

Επόμενο κριτήριο αξιολόγησης που παραθέτουμε είναι το Σχετικό Μέσο Σφάλμα (MRE, Mean Relative Error), το οποίο δίνει το σφάλμα των προβλέψεων εκφρασμένο ως την απόλυτη τιμή της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές και ορίζεται από την εξίσωση:

$$MRE(n) = \frac{\frac{1}{n}\sum_{i=1}^{n}\left|x_{act}(i) - x_{pred}(i)\right|}{x_{act}(i)}$$

Τέλος, παραθέτουμε το κριτήριο αξιολόγησης MAE, το Μέσο Απόλυτο Σφάλμα (Mean Absolute Error), το οποίο δίνει το σφάλμα των προβλέψεων εκφρασμένο ως την απόλυτη τιμή της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές, με αποτέλεσμα η τιμή του να είναι ψηλότερη από αυτή του MSE σε πολύ μικρά δεδομένα και ψηλότερη για μεγάλες τιμές δειγμάτων. Το Μέσο Απόλυτο Σφάλμα ορίζεται από την εξίσωση:

$$MAE(n) = \frac{1}{n}\sum_{i=1}^{n}\left|x_{act}(i) - x_{pred}(i)\right|$$

Όπου n, ο αριθμός των προβλέψεων, $x_{act}(i)$ η αυθεντική πραγματική τιμή του δείγματος i και $x_{pred}(i)$ η προβλεπόμενη τιμή του δείγματος i.

Όπως φαίνεται, και το κριτήριο αυτό αξιολόγησης, όπως και το MSE είναι εξαρτώμενο από την κλίμακα των δεδομένων, σε μικρότερη όμως ευαισθησία αφού το

λάθος υπολογίζεται από την απόλυτη τιμή της διαφοράς ανάμεσα στις τιμές των δειγμάτων.

3.5 Περιγραφή και Ανάλυση Σχετικών Εμπειρικών Μελετών

Κατά την διάρκεια των είκοσι τελευταίων χρόνων, έχουνε δημοσιευθεί πολλές μελέτες που αφορούν την σύγκριση των τεχνικών και των αποτελεσμάτων υπολογισμού, καθώς και των διαφορετικών τεχνικών μοντελοποίησης, για την πρόβλεψη του κόστους παραγωγής λογισμικού. Μετά από προσεκτική μελέτη των σχετικών δημοσιεύσεων, παρατηρούμε ότι ο υπολογισμός του κόστους ανάπτυξης λογισμικού παραμένει ένα πολύπλοκο και ακόμα άλυτο πρόβλημα, αφού δεν έχει αποδεχτεί κανένα μοντέλο που να προβλέπει με ακρίβεια. Οι ερευνητές στην προσπάθεια τους να το προσεγγίσουν όσο το δυνατόν καλύτερα, πρόσφατα έχουν καταφύγει σε μοντέλα βασισμένα στον τομέα της Τεχνητής Νοημοσύνης, με την χρήση Τεχνητών Νευρωνικών Δικτύων και έχουν παρουσιάσει αρκετά ενθαρρυντικά αποτελέσματα.

Ο σκοπός αυτού του τμήματος της μελέτης είναι να παρουσιάσει την ανάλυση και τα αποτελέσματα διάφορων δημοσιευμένων ερευνών που αναφέρονται στην βιβλιογραφία και που έχουν κυρίως χρησιμοποιήσει Τεχνητά Νευρωνικά Δίκτυα, Γενετικούς Αλγορίθμους ή τον συνδυασμό τους. Έτσι, στη συνέχεια θα μπορέσουμε να συγκρίνουμε και να αξιολογήσουμε τα αποτελέσματα και το ποσοστό ακριβείας της πρόβλεψης υπολογισμού του κόστους για την ανάπτυξη ενός λογισμικού, που έχουν διεξαχθεί από τα πειραματικά τρεξίματα χρησιμοποιώντας το εργαλείο NeuroShell 2 και το προτεινόμενο εργαλείο που υλοποιήθηκε για πραγματοποίηση των στόχων της διπλωματικής εργασίας. Επίσης, αξίζει να αναφέρουμε ότι οι μελέτες αυτές επηρέασαν την ανάπτυξη της μεθοδολογίας και του μοντέλου που υλοποιήθηκε, βοήθησαν στην αποτροπή από αδιέξοδα και λάθη και γενικότερα έχουν τροφοδοτήσει πολλές από τις ιδέες που υλοποιήθηκαν στην συνέχεια. Σε αυτό το σημείο παρουσιάζονται αναλυτικά οι μελέτες, καθώς τα αποτελέσματα και τα συμπεράσματα που εξάγονται από αυτές.

3.5.1 The S.G. MacDonell &A.R. Gray Study

Σε αυτή τη μελέτη [17] έχουν χρησιμοποιηθεί κάποιοι μέθοδοι μοντελοποίησης λιγότερο συνηθισμένοι σε σχέση με αυτές που προηγήθηκαν στην ανάλυση των

κατηγοριών των Υπολογιστικών Μοντέλων. Περιλαμβάνουν μοντέλα Νευρωνικών Δικτύων και Ασαφούς Λογικής, με σκοπό να ξεπεραστούν τα προβλήματα και οι περιορισμοί που παρουσιάζονταν παλιά, και που αφορούσαν τα χαρακτηριστικά των συνόλων δεδομένων (data sets).

Ο υπολογισμός της ακριβείας των αποτελεσμάτων γίνεται με την χρήση μερικών από των πιο συνηθισμένων δεικτών ακριβείας μετρήσεων ή κριτηρίων αξιολόγησης, το CC (Correlation Coefficient) και το MRE (Magnitude of Relative Error). Το MRE (Magnitude of Relative Error), ορίζεται σαν την διαφορά των κανονικοποιημένων μετρήσεων μεταξύ των πραγματικών τιμών (V_A) των δεδομένων και των υπολογιζόμενων (V_F) τιμών:

$$MRE = \frac{|V_A - V_F|}{V_A}$$

Η σύγκριση των τεχνικών μοντελοποίησης είναι βασισμένη στην ανάλυση ενός συνόλου δεδομένων, που αφορά παρατηρήσεις για περισσότερα από 80 συστήματα ανάπτυξης και που συγκεντρώθηκαν μέσα σε τέσσερα χρόνια [Desharnais 1989]. Τα δεδομένα αυτά χρησιμοποιήθηκαν για την πρόβλεψη της προσπάθειας ή του κόστους ανάπτυξης και περιλαμβάνουν τις μετρήσεις της προσπάθειας που χρειάζεται για την ανάπτυξη ενός έργου (project effort), την διάρκεια του έργου (project duration), το επίπεδο της εμπειρίας του προσωπικού ανάπτυξης (level of experience) σε σχέση με τον εξοπλισμό ανάπτυξης, την τεχνολογική υποστήριξη και την διοίκηση έργου, τον αριθμό των βασικών διεργασιών και των οντοτήτων (basic transactions and data entities) και τις ακατέργαστες μετρήσεις των λειτουργικών σημείων (function points (FP)).

Τα αριθμητικά αποτελέσματα που παρουσιάζονται αφορούν μόνο την ανάλυση βασισμένη σε μοντέλο Μονοκατευθυντικών Νευρωνικών Δικτύων (Feedforward Neural Network), με χρήση Multi-Layer Perceptron (MLP). Υπάρχουν διαθέσιμες 81 παρατηρήσεις, από τις οποίες επιλέγονται με τυχαίο τρόπο για το κάθε σενάριο οι 54. Αυτές χρησιμοποιούνται για την εκπαίδευση (training) και την δοκιμή (testing) του δικτύου, ενώ οι υπόλοιπες 20 παρατηρήσεις χρησιμοποιούνται για την επικύρωση (validation) του. Επίσης, τα αποτελέσματα αυτά αποτελούν την καλύτερη απόδοση του δικτύου με επιλογή της βέλτιστης αρχιτεκτονικής δικτύου. Η εκπαίδευση του δικτύου σταμάτησε όταν ελαχιστοποιήθηκε το λάθος δοκιμής και αυτό χρησιμοποιήθηκε για να επιλεγεί η βέλτιστη αρχιτεκτονική δικτύου.

Παρατηρούμε ότι η απόδοση του δικτύου δεν είναι ιδιαίτερα εντυπωσιακή, αφού 7 από τις 27 περιπτώσεις οι τιμές δεν προβλέπουν μέσα στο 50% των πραγματικών τιμών. Όμως σε σχέση με τις άλλες μεθόδους μοντελοποίησης, το μοντέλο Νευρωνικού Δικτύου δίνει πιο ακριβή αποτελέσματα.

3.5.2 The Ali Idri, Taghi M. Khoshgoftaar, Alain Abran Study

Σε αυτή τη μελέτη [5] εκφράζεται ο τρόπος που έγινε η επιλογή της καλύτερης μοντελοποίησης, για τον υπολογισμό της προσπάθειας ανάπτυξης λογισμικού με την χρήση Τεχνητών Νευρωνικών Δικτύων. Έστω και αν αυτή η μέθοδος έχει αποδείξει στο παρελθόν την δύναμή της στην επίλυση πολύπλοκων προβλημάτων, το μειονέκτημα της εξακολουθεί να έγκειται στο γεγονός ότι τα Νευρωνικά Δίκτυα θεωρούνται «μαύρα κουτιά», τα οποία δεν μπορούμε να ερευνήσουμε και να αλλάξουμε στο εσωτερικό επίπεδο, έτσι ώστε να τα κατανοήσουμε καλύτερα και να επιτύχουμε καλύτερα αποτελέσματα. Εντούτοις, τα πλεονεκτήματα που προσφέρουν τα Τεχνητά Νευρωνικά Δίκτυα, ειδικά όταν χρησιμοποιούνται για τον υπολογισμό κάποιας πρόβλεψης, είχαν ως αποτέλεσμα την επιτακτική ανάγκη να συμπεριληφθούν στην έρευνα αυτή.

Πιο συγκεκριμένα, ο σκοπός είναι να υπολογιστεί το κόστος ανάπτυξης λογισμικού με την χρήση ενός μοντέλου Back Propagation 3 Multi-layer Perceptron Δικτύου. Το πείραμα βασίστηκε στην προσομοίωση των Τεχνητών Νευρωνικών Δικτύων στο σύνολο δεδομένων COCOMO ’81, το οποίο περιέχει 63 έργα ανάπτυξης λογισμικού. Κάθε ένα έργο περιγράφεται από 17 χαρακτηριστικά. Επίσης, οι συναρτήσεις δραστηριοποίησης που χρησιμοποιήθηκαν για το κρυφό και το εξωτερικό επίπεδο είναι η σιγμοειδής και η συνάρτηση ταυτοποίησης αντίστοιχα. Έχουν αγνοηθεί μερικά από τα χαρακτηριστικά και έχουν παραμείνει στο πείραμα τα 13 χαρακτηριστικά που μεταβάλουν το κόστος και έτσι το Τεχνητό Νευρωνικό Δίκτυο έχει 13 εισόδους, τα COCOMO cost drivers, και μία έξοδο, την προσπάθεια ανάπτυξης λογισμικού (effort). Επίσης, τα δεδομένα εισόδου είναι όλα κανονικοποιημένα, έτσι ώστε να επιτευχθεί η διαδικασία εκπαίδευσης του δικτύου, ενώ τα βάρη είναι αρχικοποιημένα σε μικρές και τυχαίες τιμές.

Τα αποτελέσματα που παρήχθησαν εκφράστηκαν στις εξής τέσσερις ποσότητες: Min of MRE, η μικρότερη τιμή που εμφανίστηκε στα λάθη, Max of MRE, η μεγαλύτερη τιμή που εμφανίστηκε στα λάθη, Median of MRE η μέση τιμή που

εμφανίστηκε στα λάθη και τέλος Mean of MRE ο μέσος όρος των τιμών που εμφανίζονται στα λάθη.

Όπου και πάλι το MRE ορίζεται σαν η απόλυτη τιμή της διαφοράς της πραγματικής προσπάθειας με την υπολογιζόμενη διά την πραγματική τιμή.

$$MRE = \frac{\left| Effort_{actual} - Effort_{estimated} \right|}{Effort_{actual}}$$

Στο πρώτο πείραμα που έγινε, χρειάστηκε η είσοδος ολόκληρου του συνόλου των COCOMO δεδομένων για την εκπαίδευση και για την δοκιμή του δικτύου. Τα αποδεκτά αποτελέσματα φαίνονται στον πίνακα 3.2 και αυτά πραγματοποιήθηκαν με αρχιτεκτονική δικτύου που είχε 13 κρυμμένες μονάδες (units) στο εσωτερικό επίπεδο και με 300,000 επαναλήψεις μάθησης.

Παρατηρούμε ότι σε αυτό το πείραμα, εκτός από το γεγονός ότι το δίκτυο οδηγείται σε over fitting δηλαδή προσαρμόζει υπερβολικά τις τιμές του, τα αποτελέσματα δεν μπορούν να θεωρηθούν σωστά και έγκυρα γιατί έχουν χρησιμοποιηθεί τα ίδια δεδομένα για εκπαίδευση και για δοκιμή του δικτύου. Αυτό σίγουρα θα οδηγήσει σε αποτελέσματα που δεν μπορούμε να εμπιστευτούμε, καθώς συνήθως τα έργα που θα χρησιμοποιούνται για εκπαίδευση και το άγνωστο έργο που θα προσπαθήσουμε να προβλέψουμε είναι διαφορετικά μεταξύ τους. Ακόμα, με αυτό τον τρόπο δεν γίνεται η δοκιμή βασισμένη σε στοιχεία που δεν έχει επεξεργαστεί ξανά το δίκτυο, τα έργα που χρησιμοποιούνται για την εκπαίδευση είναι ανεπαρκή και επίσης, δεν υπάρχει συσχέτιση μεταξύ των εισόδων και των εξόδων του δικτύου.

Έτσι, μετά από αυτές τις παρατηρήσεις κρίθηκε αναγκαίο να διεξαχθούν κάποια επιπλέον πειράματα.

Στο δεύτερο πείραμα, αφαιρέθηκαν τυχαία 23 έργα από το σύνολο, για να χρησιμοποιηθούν για την δοκιμή του δικτύου, ενώ τα υπόλοιπα 40 χρησιμοποιήθηκαν για την εκπαίδευσή του.

Στο τρίτο πείραμα αφαιρέθηκε μόνο ένα έργο από το σύνολο και χρησιμοποιήθηκε για την δοκιμή του δικτύου, ενώ τα υπόλοιπα 62 χρησιμοποιήθηκαν για την εκπαίδευσή του. Το ένα έργο που αφαιρέθηκε αρχικά, ήταν ένα από τα 23 έργα που αφαιρέθηκαν στο δεύτερο πείραμα. Το πείραμα αυτό επαναλήφθηκε 23 φορές.

Τα αποτελέσματα φαίνονται και αυτά στον πίνακα 3.2. Από αυτά συμπεραίνουμε ότι η ακρίβεια στους υπολογισμούς αυξάνεται ανάλογα με τον αριθμό των έργων που χρησιμοποιούμε στην φάση της εκπαίδευσης του δικτύου. Η ακρίβεια των αποτελεσμάτων θεωρείται αποδεκτή.

3.5.3 The Wittig & Finnie Study

Στην έρευνα αυτή έχει χρησιμοποιηθεί η μέθοδος ανάπτυξης ενός Back Propagation Νευρωνικού Δικτύου με χρήση Multi-layer Perceptron. Σκοπός ήταν η πρόβλεψη της προσπάθειας που χρειάζεται για την ανάπτυξη λογισμικού και έχουν χρησιμοποιηθεί τα σύνολα δεδομένων Desharnais και ASMA. Το πρώτο είναι ένα σύνολο δεδομένων που αποτελείται από 81 έργα παραγωγής λογισμικού τα οποία έχουν παρθεί από ένα Καναδικό Οίκο Ανάπτυξης Λογισμικού στα τέλη της δεκαετίας του 80, ενώ το δεύτερο σύνολο δεδομένων περιλαμβάνει 136 έργα ανάπτυξης και έχει παρθεί από το Australian Software Metrics Association.

Για το σύνολο δεδομένων Desharnais, είχανε χωρίσει τα έργα ανάπτυξης τρεις φορές, μεταξύ των οποίων χρησιμοποίησαν τα 10 για δοκιμή του δικτύου και 71 για εκπαίδευση του δικτύου. Τα αποτελέσματα από τα τρία σύνολα δεδομένων που χρησιμοποιήθηκαν για επικύρωση, έχουν συγκεντρωθεί και η απόδοση τους έχουν θεωρηθεί ότι βρίσκονται σε υψηλό επίπεδο ακρίβειας, αν και μερικές τιμές έχουν αποκλειστεί. Τα αποτελέσματα αυτά ήταν τα πιο βέλτιστα και πραγματοποιήθηκαν με 0.3 ποσοστό μάθησης (learning rate) και με coefficient 0.6 σε ένα Νευρωνικό Δίκτυο με ένα κρυμμένο επίπεδο. Οι συναρτήσεις μεταφοράς Gaussian και Sigmoid έχουν επιτύχει τα καλύτερα αποτελέσματα, με τα αποτελέσματα στα λάθη των προβλέψεων για την Sigmoid συνάρτηση. Ακόμα μια παρατήρηση είναι ότι το σύνολο των έργων που χρησιμοποιήθηκαν για εκπαίδευση ήταν αρκετά μεγάλα, ενώ για επικύρωση χρησιμοποιήθηκαν πολύ μικρός αριθμός έργων, γεγονός που οδηγεί στο συμπέρασμα ότι για να έχουμε καλές προβλέψεις χρειαζόμαστε και ένα μεγάλο όγκο δεδομένων εισόδου στο δίκτυο.

3.5.4 The Jorgenson Study

Σε αυτή τη μελέτη έχουν συγκριθεί τέσσερις διαφορετικές προσεγγίσεις μοντελοποίησης, το Regression, τα Τεχνητά Νευρωνικά Δίκτυα, μια μορφή αναγνώρισης προτύπων και ένα thumb μοντέλο που υποστηρίζει ότι η προσπάθεια είναι ίση με το μέγεθος του έργου ανάπτυξης δια τον μέσο όρο παραγωγικότητας. Η περίπτωση που παρουσιάζουμε είναι αυτή της εφαρμογής Νευρωνικού Δικτύου και

κατά την οποία έχει χρησιμοποιηθεί ο αλγόριθμος Back Propagation πάνω στο σύνολο δεδομένων Jorgenson, το οποίο αποτελείται από 109 έργα συντήρησης.

Τα αποτελέσματα ήταν λιγότερο ενθαρρυντικά, καθώς το Νευρωνικό Δίκτυο έδινε μεγαλύτερα λάθη MMRE σε σχέση με τα καλύτερα αποτελέσματα που έδινε το μοντέλο Regression. Εντούτοις, παρατηρήθηκε ότι σε ποσοστιαία αναλογία, δηλαδή πόσα έργα από το σύνολο των έργων έχουν χαμηλότερα αποτελέσματα λάθους MMRE από το 25%, είχε ικανοποιητικά αποτελέσματα. Ένα από τα μειονεκτήματα της προσέγγισης παρουσιάστηκε όταν μετρήθηκε η ευρωστία της, η οποία κρίθηκε από τις πιο χαμηλές, αφού ανάλογα με το σύνολο των δεδομένων που είναι υπό εξέταση τόσο πιο απομακρυσμένα είναι τα αποτελέσματα που λαμβάνονται στην πρόβλεψη.

3.5.5 The Serluca Study

Σε αυτή την μελέτη έχουν συγκριθεί τα αποτελέσματα των τριών μεθόδων μεταξύ των οποίων το Regression, το Analogy και τα Neural Networks. Τα αποτελέσματα που παρουσιάζονται στον πίνακα 3.2 αφορούν την περίπτωση που έχει χρησιμοποιηθεί ένα Νευρωνικό Δίκτυο με αλγόριθμο Back Propagation πάνω στο σύνολο δεδομένων Mermaid-2.

Όταν χρησιμοποιήθηκε για πρόβλεψη ολόκληρο το σύνολο των δεδομένων παρατηρήθηκε ότι το αποτέλεσμα ήταν σαφώς καλύτερο από αυτό της μεθόδου Regression και της μεθόδου Analogy. Ενώ, όταν χώρισαν το σύνολο δεδομένων Mermaid-2, σε δύο περισσότερο ομοιογενή και μικρότερα κομμάτια, παρατήρησαν ότι το δίκτυο είχε φτωχά αποτελέσματα, οδηγώντας στο συμπέρασμα ότι τα Νευρωνικά Δίκτυα χρειάζονται μεγάλους όγκους πληροφορίας σαν σύνολα για εκπαίδευση για να δώσουν καλά αποτελέσματα με ακρίβεια.

3.5.6 The Samson et al. Study

Σε αυτή τη μελέτη έχει αναπτυχθεί ένα Albus Multi-layer Perceptron για την πρόβλεψη της προσπάθειας ή του κόστους ανάπτυξης ενός λογισμικού. Τα καλύτερα αποτελέσματα, διαφαίνονται στον πίνακα 3.2 για την εξαγωγή των οποίων χρησιμοποιήθηκε στο σύνολο δεδομένων (data set) COCOMO του Boehm. Αν και τα αποτελέσματα είναι καλύτερα από αυτά που παρουσιάζουν άλλες μέθοδοι, εξακολουθούν να μην θεωρούνται πειστικά.

3.5.7 The Srinivasan & Fisher Study

Σε αυτή τη μελέτη έχει συνδυαστεί ο αλγόριθμος μάθησης Back Propagation με ένα Multi-layer Νευρωνικό Δίκτυο για την πρόβλεψη των τιμών του συνόλου δεδομένων Kemerer. Λαμβάνοντας υπόψη το μικρό μέγεθος του συνόλου δεδομένων Kemerer, αφού περιέχει πληροφορίες για μόλις 15 έργα ανάπτυξης λογισμικού, έχει θεωρηθεί πρέπον να χρησιμοποιηθεί ολόκληρο αυτό το σύνολο δεδομένων για την δοκιμή του δικτύου, ενώ η εκπαίδευση του έγινε με το COCOMO data set.

Τα αποτελέσματα αυτών των πειραμάτων είναι πολύ θετικά και υποστηρίζουν την χρήση των Τεχνητών Νευρωνικών Δικτύων για προβλέψεις. Ωστόσο, θα πρέπει να σημειωθεί ότι τα αποτελέσματα αυτά διεξήχθησαν με ανορθόδοξο τρόπο, καθώς χρησιμοποιήθηκε διαφορετικό σύνολο δεδομένων για την εκπαίδευση και για την δοκιμή του δικτύου. Αυτό φυσικά δεν σημαίνει ότι τα αποτελέσματα είναι αναξιόπιστα, αλλά απλά ότι οι προσεγγίσεις γύρω από τα Νευρωνικά Δίκτυα μπορούν να αξιοποιηθούν ανάμεσα σε διαφορετικά περιβάλλοντα χρησιμοποιώντας μη ομογενή δεδομένα. Μία άλλη παρατήρηση είναι το γεγονός ότι τα αποτελέσματα είναι ιδιαίτερα ευαίσθητα στον αριθμό των επιπέδων και των νευρώνων στο εσωτερικό επίπεδο. Η εισήγηση που υπάρχει για να βρεθεί η καλύτερη δυνατή αρχιτεκτονική του δικτύου, που να εξάγει τα καλύτερα αποτελέσματα είναι ο διαχωρισμός του συνόλου των δεδομένων που χρησιμοποιούνται για εκπαίδευση και ο πειραματισμός με διάφορες αρχιτεκτονικές μέχρις ότου να ξεκινήσει η διαδικασία της μάθησης.

3.5.8 The Hughes Study

Σε αυτή τη μελέτη έχουν συγκριθεί τρεις διαφορετικές προσεγγίσεις μοντελοποίησης, συμπεριλαμβανομένου το Regression, το Analogy και τα Τεχνητά Νευρωνικά Δίκτυα. Τα αποτελέσματα στον πίνακα 3.2 αφορούν την τελευταία προσέγγιση καθώς αυτή μας ενδιαφέρει.

Το σύνολο των δεδομένων που χρησιμοποιήθηκε αφορά 33 έργα τηλεπικοινωνιών (telecommunication), τα οποία όταν αρχικά διαχωρίστηκαν σε δύο ομοιογενείς ομάδες είχαν πολύ φτωχά αποτελέσματα, ενώ όταν στην συνέχεια χρησιμοποιήθηκε ολόκληρο το σύνολο τότε βελτιώθηκαν και αυτά φαίνονται στον πίνακα 3.2. Αυτό ενδυναμώνει την άποψη ότι τα Τεχνητά Νευρωνικά Δίκτυα μπορούν

να έχουν πολύ βελτιωμένα αποτελέσματα όταν χρησιμοποιούνται μεγαλύτερα σύνολα δεδομένων, ενώ ταυτόχρονα οι υπόλοιπες τεχνικές περιθωριοποιούνται.

3.5.9 The J. Javier Dolado Study

Σε αυτή τη μελέτη [11] παρουσιάζονται τρεις τεχνικές μοντελοποίησης για τον υπολογισμό του κόστους ανάπτυξης λογισμικού. Σε αυτές συμπεριλαμβάνονται το Linear Regression (LR), τα Τεχνητά Νευρωνικά Δίκτυα (NN) και ο Γενετικός Προγραμματισμός (GA), ενώ τα αποτελέσματα που παρουσιάζουμε στον πίνακα 3.2 αφορούν μόνο τις τελευταίες δύο τεχνικές.

Το κλασικό Regression, εμφανίζεται να έχει πολλά προβλήματα που αφορούν τις υποθέσεις που πρέπει να ληφθούν υπόψη για την κατανομή και την διαδικασία κανονικοποίησης (normalization) των δεδομένων. Τα Νευρωνικά Δίκτυα προσφέρονται για να επιλύσουν τέτοιου είδους προβλήματα, αφού παρουσιάζουν την ικανότητα να προσεγγίζουν οποιαδήποτε συνάρτηση, ενώ οι Γενετικοί Αλγόριθμοι αποτελούν μία νέα τεχνική που εφαρμόζεται στο πρόβλημα και η οποία αποδείχτηκε αποτελεσματική στην μοντελοποίηση των δεδομένων και στην παραγωγή εξισώσεων που περιγράφουν τις διαδικασίες.

Στον τομέα των Νευρωνικών Δικτύων, έχει χρησιμοποιηθεί ένα Μονοκατευθυντικό Νευρωνικό Δίκτυο (Feedforward Neural Network), με δύο επίπεδα και δύο νευρώνες στο κρυμμένο επίπεδο, που χρησιμοποιούν μια logistic-sigmoid συνάρτηση, ενώ στο εσωτερικό επίπεδο υπάρχει μια γραμμική συνάρτηση μεταφοράς (linear transfer function). Οι προσομοιώσεις έχουν διεξαχθεί με την χρήση του Neural Network Toolbox της MATLAB®. Η προσέγγιση βασίζεται στο γεγονός ότι αν υπάρχει μια υποκείμενη συσχέτιση μεταξύ των εξεταζόμενων δεδομένων, τότε το Νευρωνικό Δίκτυο θα πρέπει να είναι ικανό να την εντοπίσει. Οι αλγόριθμοι μάθησης που δοκιμάστηκαν ήταν οι Lenenberg-Marquardt και Back Propagation και κρίθηκε ο δεύτερος σαν πιο βέλτιστος και πιο γρήγορος. Τα αρχικά βάρη και biases είχαν αρχικοποιηθεί τυχαία, ενώ χρειάστηκε να πραγματοποιηθούν πολλές δοκιμές έτσι ώστε να βρεθούν οι κατάλληλες τιμές και ρυθμίσεις στις παραμέτρους του δικτύου.

Στον τομέα του Γενετικού Προγραμματισμού και των Γενετικών Αλγορίθμων, ακολουθείται μια μη παραμετρική μέθοδος, αφού όπως παρατηρούμε δεν γίνεται καμιά υπόθεση για την κατανομή των δεδομένων, αλλά οι εξισώσεις που αποκομίζονται γίνονται σύμφωνα με μόνιμες, σταθερές μόνο τιμές. Η εξίσωση που έχει δώσει τα

αποτελέσματα που εμφανίζονται στον πίνακα 3.2, είναι αυτή που έχει σαν αποτέλεσμα την πιο κατάλληλη προσέγγιση, στην τελευταία γενεά για τα δεδομένα. Παρατηρούμε ότι στην προκείμενη περίπτωση λήφθηκαν τα καλύτερα αποτελέσματα όταν περιορίστηκε ο αριθμός των γενεών.

Τα δεδομένα που χρησιμοποιήθηκαν για τις προσομοιώσεις ήταν πρώτα το σύνολο δεδομένων του Belady το οποίο περιείχε 33 έργα ανάπτυξης. Δεύτερο ήταν το σύνολο δεδομένων του Boehm (COCOMO), τρίτο ήταν το σύνολο δεδομένων των Albrecht και Gaffney 1983, τα οποία περιείχαν 24 έργα ανάπτυξης και Kemerer 1987, το οποίο περιείχε 15 έργα ανάπτυξης. Τέταρτο ήταν το σύνολο δεδομένων 48 έργων ανάπτυξης λογισμικών από φοιτητές σε ακαδημαϊκό περιβάλλον, ενώ πέμπτο ήταν το σύνολο δεδομένων που χρησιμοποιήθηκε από τον Matson et al. αφού τα αυθεντικά δεδομένα δεν ήταν διαθέσιμα.

Τα αποτελέσματα που λάβαμε με την εισαγωγή των ίδιων δεδομένων κατά την εκπαίδευση και κατά την δοκιμή του δικτύου, δείχνουν ότι οι Γενετικοί Αλγόριθμοι είχαν καλύτερα αποτελέσματα από τις άλλες τεχνικές στο τρίτο και στο τέταρτο σύνολο δεδομένων, ενώ είχαν χειρότερα στο πρώτο και στο δεύτερο σύνολο δεδομένων. Επίσης, τα Τεχνητά Νευρωνικά Δίκτυα παρουσιάζουν παρεμφερή αποτελέσματα ότι δηλαδή είχαν καλύτερα αποτελέσματα από τις άλλες τεχνικές στο τρίτο και στο τέταρτο σύνολο δεδομένων, ενώ είχαν χειρότερα στο πρώτο και στο δεύτερο σύνολο δεδομένων. Παρατηρούμε όμως ότι οι Γενετικοί Αλγόριθμοι πάντοτε ξεπερνούν σε θέματα απόδοσης τα Τεχνητά Νευρωνικά Δίκτυα.

Τα αποτελέσματα που λάβαμε παίρνοντας δείγματα από τα σύνολα δεδομένων και χρησιμοποιώντας τα για διαφορετικό σκοπό κάθε φορά, για την εκπαίδευση, την επικύρωση και την δοκιμή του δικτύου δεν παρουσιάζει να έχει καλύτερα αποτελέσματα, ούτε κάποια εμφανή διαφοροποίηση από πριν, δεν έχουν κανένα ενδιαφέρον και έτσι δεν εμφανίζονται στον πίνακα.

Συμπερασματικά, παρατηρούμε ότι τα Τεχνητά Νευρωνικά Δίκτυα και οι Γενετικοί Αλγόριθμοι είναι πιο ελαστικές σαν τεχνικές και έχουν ελαφρώς καλύτερα αποτελέσματα από τις κλασικές μεθόδους υπολογισμού. Ωστόσο, μέχρι σήμερα δεν έχει ξεχωρίσει κάποια τεχνική που να αποδίδει τα βέλτιστα αποτελέσματα.

Πίνακας 3.2 Αποτελέσματα και Πληροφορίες των Εμπειρικών Μελετών [17], [5], [6], [11].

AUTHOR NAME	NAME OF PROJECT / GOAL OF ESTIMATE	METHOD	DATASET	NN ARCHI TECTURE	CC	MRE	
S.G. MacDonell &A.R. Gray	A Comparison of Modeling Techniques for Software Development Effort Prediction	Feedforward NN MPL	[Desharnais 1989]		0.8896	0.2968	**Training Data**
					0.7745	0.4586	**Testing Data**
					0.7379	0.43508	**Validation Data**
Ali Idri, Taghi M. Khoshgoftaar, Alain Abran	Can Neural Networks be easily Interpreted in Software Cost Estimation	Backpropagation MLP	COCOMO '81	13-13-1		1,5%	
				13-13-1		203,66%	
				13-13-1		84,35%	
Wittig & Finnie	Effort	NN backpropagation	Desharnais			27%	
			ASMA			17%	
Jorgenson	Maintenance Effort	NN backpropagation	Jorgenson			100%	
Serluca	Development Effort	NN backpropagation	Mermaid-2			76%	
Samson et al.	Development Effort	NN backpropagation	COCOMO[63]			428%	
Srinivasan & Fisher	Development Effort	NN backpropagation	Kemerer & COCOMO[63]			70%	
Hughes	Development Effort	NN backpropagation	Hughes			55%	
J. Javier Dolado	Limits to the Methods in Software Cost Estimation / Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation	Feed-forward NN	Belady's Data	x*-2-2-1		1.2733	
		GP	Belady and Lehman, 1979	x*-2-2-1		0.848	
		Feed-forward NN	Boehm,1981	x*-2-2-1		5.8576	
		GP	Boehm,1981	x*-2-2-1		3.227	
		Feed-forward NN	Albrecht, Gaffney 1983&Kemerer,1987	x*-2-2-1		2.123	
		GP	Albrecht, Gaffney 1983&Kemerer,1987	x*-2-2-1		0.684	
		Feed-forward NN	Dolado, 1997	x*-2-2-1		0.4211	
		GP	Dolado, 1997	x*-2-2-1		0.433	
		Feed-forward NN	Abran and Robillard, 1996	x*-2-2-1		0.279	
		GP	Abran and Robillard, 1996	x*-2-2-1		0.221	
		Feed-forward NN	Miyazaki et al., 1994	x*-2-2-1		0.5886	
		GP	Miyazaki et al., 1994	x*-2-2-1		0.643	
		Feed-forward NN	Approximation to the dataset of Matson et al., 1994	x*-2-2-1		1.1448	
		GP	Approximation to the dataset of Matson et al., 1994	x*-2-2-1		0.958	

* "x": ερμηνεύεται σαν άγνωστος αριθμός εισόδων

"NN ARCHITECTURE": π.χ. "i-h_i -…- h_n-o" ερμηνεύεται σαν i αριθμός νευρώνων εισόδου, h_i αριθμός των νευρώνων στο i κρυμμένο επίπεδο και o αριθμός των νευρώνων εξόδου.

Στον πίνακα 3.2 παρουσιάζονται συγκεντρωμένα τα αποτελέσματα των εμπειρικών μελετών που εμφανίζονται στα άρθρα και στην σχετική βιβλιογραφία, με τους συγγραφείς τους, την αρχιτεκτονική του Νευρωνικού Δικτύου που υιοθέτησαν, το σύνολο δεδομένων (data set), καθώς και την μέθοδο που χρησιμοποίησαν. Τα αποτελέσματα εμφανίζονται σε στήλες που αντιπροσωπεύουν τα λάθη και το ποσοστό ακριβείας υπολογισμού της προσπάθειας ανάπτυξης, δηλαδή του κόστος ανάπτυξης ενός λογισμικού. Τα στοιχεία που απουσιάζουν από τον εν λόγω πίνακα, οφείλονται σε παραλείψεις στις συγκεκριμένες μελέτες.

Στη συνέχεια, μετά από την μελέτη των σχετικών εμπειρικών μελετών που παρουσιάστηκαν προηγουμένως, αναφέρουμε κάποια συμπεράσματα και κάποιες διαπιστώσεις, που βοήθησαν στην λήψη σχετικών αποφάσεων στην υλοποίηση του προτεινόμενου εργαλείου και στην αποτροπή από αδιέξοδα και λάθη, αλλά και που γενικότερα έχουν τροφοδοτήσει την διπλωματική εργασία με ιδέες.

Η καταφυγή σε μοντέλα βασισμένα στον τομέα της Τεχνητής Νοημοσύνης, με την χρήση Τεχνητών Νευρωνικών Δικτύων έχουν παρουσιάσει αρκετά ενθαρρυντικά και πιο ακριβή αποτελέσματα σε σχέση με άλλες μεθόδους. Θα πρέπει να προσέξουμε το δίκτυο να μην οδηγείται σε over fitting, δηλαδή να προσαρμόζει υπερβολικά τις τιμές του. Ακόμα, πολλές μελέτες αναφέρουν το κύριο μειονέκτημα των Τεχνητών Νευρωνικών Δικτύων ότι δεν μπορούμε να τα ερευνήσουμε και να τα αλλάξουμε στο εσωτερικό επίπεδο, έτσι ώστε να τα κατανοήσουμε καλύτερα και να επιτύχουμε καλύτερα αποτελέσματα. Εντούτοις τα πλεονεκτήματα, είναι περισσότερα και έτσι προσφέρονται για την επίλυση τέτοιων προβλημάτων. Είναι σημαντικό να αναφέρουμε ότι τα δεδομένα πρέπει να εισάγονται κανονικοποιημένα στο δίκτυο. Ακόμα, δεν θεωρείται σωστή και απορρίπτεται η προσέγγιση που προτάθηκε από μια μελέτη που χρησιμοποίησε το ίδιο σύνολο δεδομένων για την εκπαίδευση και για την δοκιμή του δικτύου, αφού εξάγει αποτελέσματα που δείχνουν ότι το δίκτυο αποστηθίζει τις τιμές. Επιπλέον, φαίνεται ότι η ακρίβεια στους υπολογισμούς αυξάνεται ανάλογα με τον αριθμό των έργων που χρησιμοποιούνται στην εκπαίδευση και άρα το σύνολο δεδομένων που χρησιμοποιείται πρέπει περίπου αναλογικά, να χρησιμοποιείται το 70% για εκπαίδευση του δικτύου. Οι προσεγγίσεις στα Τεχνητά Νευρωνικά Δίκτυα μπορούν να αξιοποιηθούν ανάμεσα σε διαφορετικά περιβάλλοντα, ανάλογα δηλαδή με την τοπολογία του δικτύου, τον αριθμό των επαναλήψεων μάθησης και τα βάρη των νευρώνων, ενώ τα αποτελέσματα είναι ιδιαίτερα ευαίσθητα στον αριθμό των επιπέδων και των νευρώνων στο εσωτερικό επίπεδο.

Κεφάλαιο 4

Πειραματική Μελέτη και Προτεινόμενο Σύστημα

4.1 Εισαγωγή

Στόχος αυτού του κεφαλαίου είναι η παρουσίαση της μεθοδικής πειραματικής μελέτης και του προτεινόμενου συστήματος που υποβοηθά στην διεξαγωγή επιπλέον πειραμάτων, έτσι ώστε να μπορέσει να υπολογιστεί το κόστος της προσπάθειας που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο λογισμικού. Η διεξαγωγή ακριβής και αξιόπιστης πρόβλεψης του υπολογισμού του κόστους ανάπτυξης ενός λογισμικού είναι προκλητική και ιδιαίτερα πολύπλοκη και δύσκολη διεργασία.

Η προτεινόμενη μεθοδολογία πειραμάτων ακολουθεί δύο διόδους. Αρχικά διεξάγονται ένα σύνολο από πειραματικά τρεξίματα με την χρήση ενός εργαλείου, το οποίο προσομοιώνει την χρήση Τεχνητών Νευρωνικών Δικτύων. Στη συνέχεια ακολουθούν τα πειραματικά αποτελέσματα του προτεινόμενου εργαλείου που υλοποιήθηκε για να εξυπηρετήσει τους στόχους αυτής της μελέτης, το οποίο δημιουργεί Τεχνητό Νευρωνικό Δίκτυο και με την χρήση Γενετικού Αλγορίθμου, επιλέγει την πιο βέλτιστη αρχιτεκτονική του Τεχνητού Νευρωνικού Δικτύου.

4.2 Γενική Περιγραφή

Η MATLAB® είναι μια υψηλής απόδοσης γλώσσα που χρησιμοποιείται για Technical Computing και η οποία μπορεί να εκφράζει λύσεις στα προβλήματα με

μαθηματικό τρόπο. Υποστηρίζει την δημιουργία Τεχνητών Νευρωνικών Δικτύων με την βοήθεια των οποίων μπορούμε να λύσουμε προβλήματα, όπως αυτό που προσπαθεί να προσεγγίσει η διπλωματική εργασία. Κάποια αρχικά πειράματα δημιουργίας, εκπαίδευσης και δοκιμής ενός δικτύου στην MATLAB®, έδειξαν ότι δεν μπορούσαν να είναι καθοριστικά στην επίλυση του προβλήματος υπολογισμού του κόστους ανάπτυξης λογισμικού, χωρίς να υπάρχουν κάποια επιπλέον αποτελέσματα με τα οποία θα γινόταν κάποια σύγκριση έτσι ώστε για να μπορέσουν να ξεπεραστούν οι περιορισμοί.

Οι περιορισμοί της MATLAB®, οδήγησαν στην χρήση του εργαλείου της Ward Systems Group, NeuroShell 2, με το οποίο διεξήχθησαν μεθοδικά τριακόσια πειράματα. Αρχικά, δημιουργήθηκαν τα αναγκαία σύνολα που αποτέλεσαν την είσοδο του Τεχνητού Νευρωνικού Δικτύου, χωρίστηκαν σε τρία υποσύνολα αυτό της εκπαίδευσης, της επικύρωσης και της δοκιμής και πήραμε σαν αποτέλεσμα την πρόβλεψη της προσπάθειας ανάπτυξης λογισμικού. Η μεθοδολογία συνεχίστηκε με την ανάπτυξη εργαλείου δημιουργίας Νευρωνικών Δικτύων και στο οποίο εφαρμόστηκε Γενετικός Αλγόριθμος σε μια προσπάθεια βελτιστοποίησης των αποτελεσμάτων.

4.3 Περιγραφή Προτεινόμενης Μεθοδολογίας της Έρευνας με την Χρήση Τεχνητών Νευρωνικών Δικτύων και Γενετικό Προγραμματισμό

Η μοντελοποίηση για την πρόβλεψη του κόστους ανάπτυξης λογισμικού μέσω Υπολογιστικής Νοημοσύνης αποτελεί τον συνδυασμό δύο τεχνικών μηχανών εκμάθησης, των Τεχνητών Νευρωνικών Δικτύων και του Γενετικού Προγραμματισμού.

4.3.1 Τεχνητά Νευρωνικά Δίκτυα

Τα Τεχνητά Νευρωνικά Δίκτυα είναι μια αρκετά διαδεδομένη μεθοδολογία εκμάθησης προτύπων δεδομένων και εξαγωγής υπολογισμών και συμπερασμάτων. Έχουν την γενική ικανότητα που εκφράζεται μέσα από τις διαδικασίες υπολογισμών, συλλογισμών, λογικής, διακρίβωσης, μάθησης, χρήσης γλώσσας, αντίληψης του περιβάλλοντος σε διαφορετικούς βαθμούς λεπτομέρειας, εξοικείωσης σε νέο περιβάλλον, αυτοδιόρθωσης και επινόησης. Μπορούν να προσεγγίζουν οποιανδήποτε συνάρτηση, και με την ικανότητά τους να γενικεύουν και να λύνουν προβλήματα μεγάλης πολυπλοκότητας, εμφανίζονται σαν μια από τις πιο βέλτιστες μεθόδους επίλυσης του προβλήματος του υπολογισμού του κόστους ανάπτυξης λογισμικού.

Στόχος τους είναι η υποστήριξη ενός μηχανισμού αυτόματης μάθησης. Η μάθηση αυτή συνήθως γίνεται με διαδικασίες μειώσεως του σφάλματος μεταξύ της πραγματικής και της επιθυμητής εξόδου, όπως γίνεται σε προβλήματα βελτιστοποίησης, όπως είναι και ο στόχος του ακριβή υπολογισμού του κόστους ανάπτυξης λογισμικού. Ο μηχανισμός της αυτόματης μάθησης είναι υπεύθυνος για τη εκπαίδευση και την προσαρμογή του συστήματος σύμφωνα με τις μεταβαλλόμενες συνθήκες στην ίδια υπό εξέταση περίπτωση και το περιβάλλον της.

Τα Τεχνητά Νευρωνικά Δίκτυα υποστηρίζουν την προσπάθεια ανάπτυξης μηχανών με την ικανότητα να σκέφτονται και ακολουθούν μια μεθοδολογία, η οποία μιμείται τον τρόπο με τον οποίο έχει δημιουργηθεί ο ανθρώπινος εγκέφαλος. Είναι ένα σύστημα κατασκευασμένο με πολλούς κόμβους επεξεργασίας (processing elements), διασυνδεδεμένους μεταξύ τους με μεγάλο βαθμό συνεκτικότητας, τα οποία επεξεργάζονται τα δεδομένα εισόδου δυναμικά, προσομοιώνοντας την λειτουργία του ανθρώπινου εγκεφάλου. Οι κόμβοι αυτοί ονομάζονται τεχνητά νευρώνια (artificial neurons) ή μονάδες (units). Ελπίζεται ότι με την αυτόνομη ανάπτυξη αυτών των συστημάτων, θα εκδηλωθεί κάποιας μορφής νοημοσύνη, και σταδιακά ευφυία.

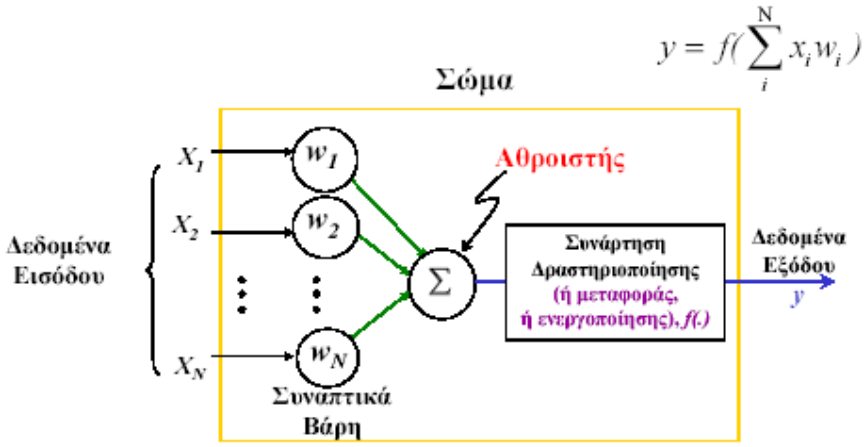
Τα Τεχνητά Νευρωνικά Δίκτυα αποτελούνται από απλά στοιχεία τα οποία λειτουργούν σε παραλληλία. Αποτελούνται από πολλούς τεχνητούς νευρώνες που είναι παράλληλα κατανεμημένοι σε ομάδες επεξεργαστών, σε στοιβάδες ή στρώματα (ή επίπεδα) και είναι συνδεδεμένοι σε ένα οργανωμένο σύνολο, στο οποίο υπάρχει επικοινωνία και αλληλεπίδραση μεταξύ τους. Λόγω της παράλληλης λειτουργίας, η επεξεργασία γίνεται πολύ γρήγορα και η σταθερότητα (robustness) του συστήματος είναι μεγάλη.

Στη συνέχεια αναφέρουμε μερικές σημαντικές ιδιότητες των Τεχνητών Νευρωνικών Δικτύων. Συνήθως εκπαιδεύονται με διδασκαλία (supervised learning), είναι πολύ ανεκτικά στα λάθη, μπορούν εύκολα να προβούν σε συμπεράσματα για κάτι που έχουν διδαχτεί επαρκώς και έχουν την ιδιότητα της γενίκευσης. Αν τα δεδομένα αλλάξουν τότε χρειάζονται επανεκπαίδευση, μπορεί να γίνει ανάπτυξη πολλών μοντέλων ή και αλλαγές εσωτερικά στο δίκτυο για την ίδια εφαρμογή. Οι κανόνες είναι κωδικοποιημένοι σε μια κατάσταση κατανομής συνδέσεων και βαρών. Τα κύρια χαρακτηριστικά των Τεχνητών Νευρωνικών Δικτύων είναι η δυνατότητα ύπαρξης πολλών εισόδων και ενός εξόδου (MISO), η μη-γραμμικότητα (non-linearity) και η προσαρμοστικότητα (adaptivity).

Η μάθηση όπως προσδιορίστηκε από τους Mendel και McClaren (1970), είναι η διαδικασία κατά την οποία οι ελεύθεροι παράμετροι του Τεχνητού Νευρωνικού Δικτύου υιοθετούνται μέσω μιας διαδικασίας ερεθισμού από το περιβάλλον, ενώ ο τύπος της μάθησης καθορίζεται από τον τρόπο που οι αλλαγές στις παραμέτρους συμβαίνουν. Ο τύπος της μάθησης μπορεί να είναι είτε με επίβλεψη (supervised), είτε χωρίς επίβλεψη (unsupervised), είτε με ενίσχυση (reinforced), είτε με συναγωνισμό (competitive). Η μάθηση με επίβλεψη γίνεται όταν κατά την διάρκεια της εκπαίδευσης ενός Τεχνητού Νευρωνικού Δικτύου εφαρμόζονται στο δίκτυο, ένα σύνολο ερεθισμάτων και λαμβάνουμε μια απάντηση. Η απάντηση συγκρίνεται με ένα προκαθορισμένο επιθυμητό στόχο (target) και αν η πραγματική απάντηση του δικτύου διαφέρει από τον στόχο τότε το Νευρωνικό Δίκτυο παράγει ένα σήμα λάθους (error signal), το οποίο στην συνέχεια θα χρησιμοποιηθεί για να υπολογιστεί η προσαρμογή που πρέπει να γίνει στα συναπτικά βάρη (synaptic weights) του δικτύου, έτσι ώστε η απάντηση του δικτύου να ταυτίζεται με τον επιθυμητό στόχο. Η μάθηση χωρίς επίβλεψη γίνεται χωρίς την ύπαρξη του στόχου εξόδου, αλλά τα πρότυπα εισόδου οργανώνονται σε κατηγορίες και το δίκτυο παράγει σαν απάντηση την κατηγορία στην οποία ανήκει το ερέθισμα. Αν η κατηγορία αυτή δεν υπάρχει τότε δημιουργείται. Η μάθηση με ενίσχυση απαιτεί έναν ή περισσότερους νευρώνες στο εξωτερικό επίπεδο και ένα δάσκαλο ο οποίος καθορίζει αν η απάντηση ταυτίζεται με την πραγματική απάντηση ή όχι, με μια ένδειξη pass or fail. Μέχρι να πετύχει το δίκτυο απάντηση pass, δοκιμάζει ξανά αναπροσαρμόζοντας τις παραμέτρους του. Η μάθηση με συναγωνισμό γίνεται αφού τοποθετηθούν μερικά νευρώνια στο εξωτερικό επίπεδο, από τα οποία αξιολογούνται οι διαφορετικές τους απαντήσεις και κυριαρχεί ένα. Έτσι σε κάθε επανάληψη με διαφορετικά ερεθίσματα καθορίζεται ένα νευρώνιο σαν το πιο βέλτιστο και το οποίο εκπαιδεύεται να δίνει τα αποτελέσματα για αυτά τα ερεθίσματα. Για την επίλυση του προβλήματος της πρόβλεψης του κόστους ανάπτυξης λογισμικού επιλέγουμε η μάθηση να είναι με επίβλεψη, με δάσκαλο.

Στην συνέχεια, στο σχήμα 4.1 παραθέτουμε μια αναπαράσταση απλού μοντέλου τεχνητού νευρώνα. Κατά την διάρκεια της εκπαίδευσης ενός Τεχνητού Νευρωνικού Δικτύου, ένα σύνολο ερεθισμάτων, τα δεδομένα εισόδου, εφαρμόζονται στο δίκτυο και λαμβάνουμε μια απάντηση, ένα σύνολο από δεδομένα εξόδου. Ο κόμβος του αθροιστή στο μοντέλο υπολογίζει ένα γραμμικό συνδυασμό των εισόδων που εφαρμόζονται στα βάρη του. Η απάντηση που λαμβάνεται τότε συγκρίνεται με ένα προκαθορισμένο επιθυμητό σήμα εξόδου, τον στόχο (target). Αν η πραγματική απάντηση του δικτύου

διαφέρει από τον στόχο, την επιθυμητή απάντηση, τότε το Νευρωνικό Δίκτυο παράγει ένα σήμα λάθους (error signal), το οποίο στην συνέχεια θα χρησιμοποιηθεί για να υπολογιστεί η προσαρμογή που πρέπει να γίνει στα συναπτικά βάρη (synaptic weights) του δικτύου, έτσι ώστε η απάντηση του δικτύου να ταυτίζεται με τον επιθυμητό στόχο.



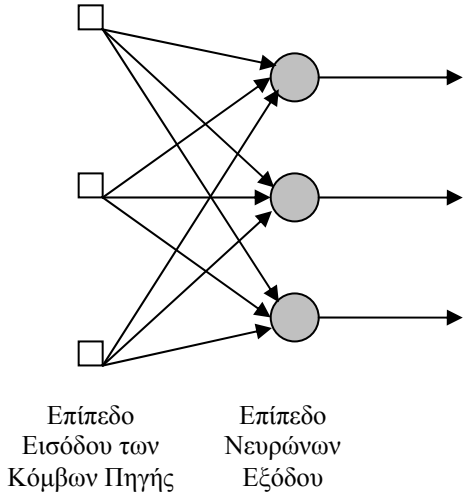
Σχήμα 4.1: Αναπαράσταση Απλού Μοντέλου Τεχνητού Νευρώνα

Παραθέτουμε μερικές σημαντικές παραμέτρους των Τεχνητών Νευρωνικών Δικτύων:

1. Τοπολογία ή Αρχιτεκτονική του Δικτύου.
2. Αριθμός των Επιπέδων.
3. Αριθμός των Νευρώνων ή των Κόμβων σε κάθε επίπεδο.
4. Αλγόριθμος Εκμάθησης.
5. Αριθμός των Επαναλήψεων ανά Πρότυπο στην φάση της Εκπαίδευσης.
6. Αριθμός των Υπολογισμών ανά Επανάληψη.
7. Απόδοση του Δικτύου.
8. Πλαστικότητα του Δικτύου (ο αριθμός των νευρώνων που αποτυγχάνουν και ο βαθμός λειτουργικότητας του Δικτύου).
9. Ικανότητα του Δικτύου (ο μεγαλύτερος αριθμός των προτύπων που μπορεί να ανακαλέσει το Δίκτυο).
10. Βαθμός Προσαρμοστικότητας του Δικτύου.
11. Όροι bias (προκατάληψης, συνήθως καθορίζεται στην σταθερή τιμή +1).
12. Κατώφλι (threshold, συνήθως καθορίζεται σε μια σταθερή τιμή όπως το 0 ή το 1).
13. Όρια των Συναπτικών Βαρών.

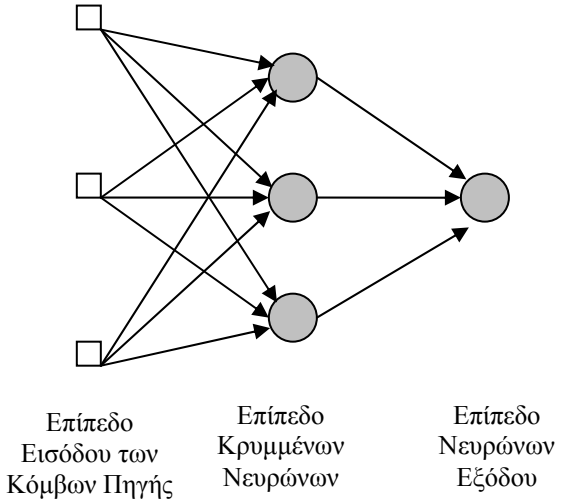
Στην συνέχεια , θα δούμε μερικές από τις πιο βασικές αρχιτεκτονικές (ή τοπολογίες) Τεχνητών Νευρωνικών Δικτύων [4], που μπορούν να χρησιμοποιηθούν για τη δημιουργία άλλων, πιο σύνθετων δομών. Αυτές είναι τα Adaline, Madaline τα Perceptrons και σταδιακά τα Multi-layer Perceptrons (MLP).

Τα Τεχνητά Νευρωνικά Δίκτυα αποτελούνται από πολλά νευρώνια συνδεδεμένα με διαφορετικές αρχιτεκτονικές. Πρώτη κατηγορία είναι τα Single-Layer Feedforward Networks, η οποία είναι η πιο απλή κατηγορία στην οποία έχουμε ένα επίπεδο εισόδου και ένα επίπεδο από νευρώνες εξόδου και στο οποίο το δίκτυο έχει μόνο μια κατεύθυνση από τις εισόδους στις εξόδους. Ένα παράδειγμα με 3 νευρώνες φαίνεται στο σχήμα 4.2.

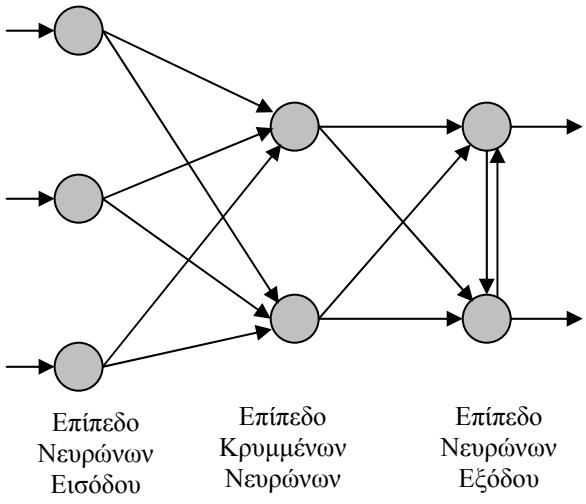


Σχήμα 4.2: Single-Layer Feedforward Network

Η επόμενη κατηγορία τοπολογίας Τεχνητών Νευρωνικών Δικτύων είναι τα Multilayer Feedforward Networks, τα οποία διακρίνονται από την ύπαρξη ενός ή περισσοτέρων κρυμμένων ή εσωτερικών επιπέδων. Οι νευρώνες στα επίπεδα αυτά ονομάζονται κρυμμένοι και έχουν την ευθύνη να μεσολαβήσουν μεταξύ του εξωτερικού επιπέδου εισόδων και της εξόδου με τέτοιο τρόπο ώστε να βοηθήσουν στους υπολογισμούς. Ένα παράδειγμα Multilayer Feedforward Network φαίνεται στο σχήμα 4.3, ενώ στο σχήμα 4.4 παρουσιάζεται ένα ακόμα παράδειγμα Multilayer Feedforward Network Cooperative / Competitive.

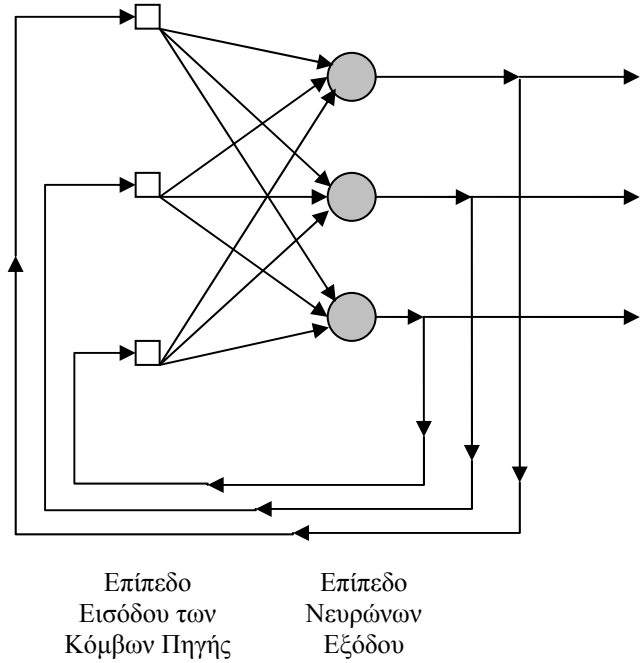


Σχήμα 4.3: Multilayer Feedforward Network



Σχήμα 4.4: Multilayer Feedforward Network Cooperative / Competitive

Τρίτη και τελευταία κατηγορία είναι τα Recurrent Networks τα οποία διακρίνονται από την ύπαρξη τουλάχιστον ενός Feedforward loop επανάληψης. Για παράδειγμα, ένα τέτοιο δίκτυο θα μπορούσε να αποτελείται από ένα επίπεδο από νευρώνες με κάθε ένα νευρώνα να τροφοδοτεί με την έξοδο του νευρώνα προς τα πίσω την είσοδο του κάθε νευρώνα. Ένα παράδειγμα Recurrent Network φαίνεται στο σχήμα 4.5.



Σχήμα 4.5: Recurrent Network

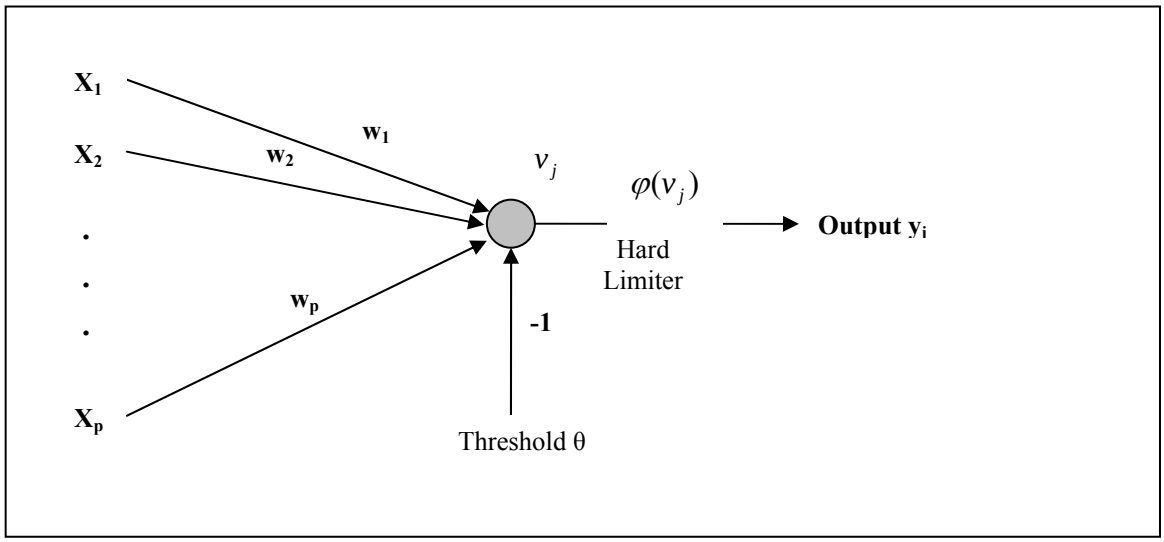
Το Perceptron είναι η απλούστερη μορφή Τεχνητού Νευρωνικού Δικτύου, το οποίο χρησιμοποιείται για την ταξινόμηση ενός ειδικού τύπου προτύπων, που λέγεται γραμμικά διαχωριζόμενα, δηλαδή πρότυπα τα οποία βρίσκονται στις αντίθετες πλευρές ενός υπερεπιπέδου. Ένα τέτοιο δίκτυο αποτελείται βασικά από ένα απλό νευρώνα, με προσαρμοζόμενα βάρη και κατώφλι.

Το απλό μονοεπίπεδο (single layer) Perceptron εμφανίστηκε πρώτα σε μια διαδικασία μάθησης που αναπτύχθηκε από τον Rosenblatt το 1950, στο Perceptron μοντέλο του εγκεφάλου και χρησιμοποιήθηκε για να προσαρμόσει τις ελεύθερες παραμέτρους του δικτύου και είχε μεγάλη επίδραση στην ανάπτυξη των Τεχνητών Νευρωνικών Δικτύων. Δημιουργήθηκε με έχοντας σαν βάση το μοντέλο McCulloch-Pitts για έναν τεχνητό νευρώνα. Ένας τέτοιος νευρώνας αποτελείται από ένα γραμμικό συνδυαστή, που ακολουθείται από ένα περιοριστή (ή ψαλιδιστή) ή στοιχείο κατωφλιού (hard limiter) το οποίο εφαρμόζει την συνάρτηση signum. Το μοντέλο McCulloch-Pitts φαίνεται στο σχήμα 4.6. Ο κόμβος του αθροιστή στο μοντέλο υπολογίζει ένα γραμμικό συνδυασμό των εισόδων που εφαρμόζονται στα βάρη του και το ίδιο συμβαίνει για το εξωτερικά εφαρμοζόμενο κατώφλι. Το άθροισμα που προκύπτει εφαρμόζεται στον ψαλιδιστή. Στη συνέχεια, ο νευρώνας παράγει μια έξοδο ίσον με +1, αν η είσοδος του ψαλιδιστή είναι θετική και -1 αν είναι αρνητική. Στο σχήμα 4.6 τα συνοπτικά βάρη του

Perceptron ενός επιπέδου, συμβολίζονται με $w_1, w_2, ..., w_p$ και αντίστοιχα οι είσοδοι που εφαρμόζονται στο Perceptron συμβολίζονται σαν $x_1, x_2, ..., x_p$ ενώ το εξωτερικά εφαρμοζόμενο κατώφλι συμβολίζεται με θ. Έτσι στο μοντέλο του σχήματος 4.6 φαίνεται ότι η έξοδος του γραμμικού συνδυαστή, δηλαδή η είσοδος του στοιχείου κατωφλίου είναι:

$$u = \sum_{i=1}^{p} w_i x_i - \theta$$

Το Perceptron ενός επιπέδου απέδειξε ότι αν τα πρότυπα που χρησιμοποιούνται για την εκπαίδευση του δικτύου, προέρχονται από δύο γραμμικά διαχωριζόμενες κλάσεις, τότε ο αλγόριθμος συγκλίνει.



Σχήμα 4.6: Μοντέλο McCulloch-Pitts Perceptron ενός επιπέδου

Τα Multi-layer Perceptrons (MLP), είναι δίκτυα Perceptron πολλών επιπέδων. Είναι δίκτυα εμπρός τροφοδότησης (Feedforward) και αποτελούνται από ένα σύνολο αισθητηρίων, το επίπεδο εισόδων, ένα ή περισσότερα κρυφά επίπεδα υπολογιστικών κόμβων και ένα επίπεδο υπολογιστικών κόμβων εξόδου. Συνήθως εκπαιδεύονται, μαθαίνουν με επίβλεψη (supervised learning), χρησιμοποιώντας ένα πολύ δημοφιλή αλγόριθμο ανατροφοδότησης σφάλματος (Error Backpropagation - BP), που είναι επίσης γνωστός ως Γενικευμένος Κανόνας Δέλτα (Generalized Delta Rule). Ο αλγόριθμος βασίζεται σε κανόνα μάθησης που χρησιμοποιεί το σφάλμα στην έξοδο για να οδηγήσει το δίκτυο σε σταδιακά καλύτερη απόδοση (Error Correction Learning Rule). Αποτελείται από δύο περάσματα διαμέσου των επιπέδων του δικτύου. Ένα

πέρασμα προς τα εμπρός, ενεργοποίησης (forward pass, activation pass) και ένα πέρασμα προς τα πίσω, ανατροφοδότησης, ανάδρασης (backward pass, error back propagation, error feedback). Στην ενεργοποίηση, τα βάρη στο δίκτυο διατηρούνται σταθερά. Κατά τη διάρκεια της ανατροφοδότησης τα βάρη προσαρμόζονται σύμφωνα με τον κανόνα διόρθωσης σφάλματος, έτσι ώστε σταδιακά να μειωθεί το συνολικό σφάλμα (logistic function). Η κατανεμημένη μορφή της μη-γραμμικότητας και η υψηλή διασύνδεση του δικτύου κάνουν την θεωρητική ανάλυση ενός MLP εξαιρετικά δύσκολη. Επίσης, η χρήση κρυμμένων νευρώνων κάνει την διαδικασία μάθησης δύσκολη και δυσνόητη.

Ο αλγόριθμος BP είναι ένας γρήγορος τρόπος για την επίτευξη μάθησης τέτοιων νευρωνικών δικτύων, όπου βοηθά στη κατανομή της αναγκαίας διόρθωσης στους νευρώνες στο κρυμμένο επίπεδο, ώστε το συνολικό σφάλμα να μειώνεται σταδιακά. Ο κλασικός αλγόριθμος BP εφαρμόζεται σε δίκτυα με πολλά επίπεδα, που έχουν ένα ή περισσότερα κρυμμένα επίπεδα, που είναι πλήρως συνδεδεμένα. Οι υπολογισμοί από την είσοδο προς την έξοδο, λέγονται συνήθως σήματα ενεργοποίησης ή συναρτησιακά σήματα (function signals) ή σήματα δραστηριοποίησης, ενώ τα σήματα που στέλνονται πίσω για να διορθώσουν τα βάρη λέγονται σήματα διόρθωσης βαρών (error correction signals) ή ανατροφοδότησης (feedback signals). Έρευνες έδειξαν ότι ένα δίκτυο που έχει μόνο δύο κρυμμένα επίπεδα που έχουν λογιστική-σιγμοειδή δραστηριοποίηση και είναι αρκετό για να επιλύσει τα πλείστα μη γραμμικά προβλήματα απεικόνισης εισόδου – εξόδου (Kolmogorov, White, Hecht-Nielsen).

4.3.2 Γενετικός Αλγόριθμος

Ο Γενετικός Προγραμματισμός είναι μια σχετικά νέα τεχνική, που έχει αποδειχτεί αποτελεσματική στην μοντελοποίηση δεδομένων και στην παραγωγή εξισώσεων που περιγράφουν με επιτυχία τις διαδικασίες, και που τελικά είναι πολύ χρήσιμος στην αναζήτηση λύσεων σε πρακτικά προβλήματα βελτιστοποίησης και εκμάθησης. Είναι ένας τύπος εξελικτικής τεχνικής υπολογισμού, ο οποίος χρησιμοποιήθηκε σε ένα ευρύ σύνολο από τομείς με επιτυχία. Πιο συγκεκριμένα, είναι μια μη παραμετρική μέθοδος, αφού δεν κάνει υποθέσεις όσο αφορά την κατανομή των δεδομένων, αλλά παράγει εξισώσεις σύμφωνα με τις τιμές που προσαρμόζονται. Οι Γενετικοί Αλγόριθμοι αναφέρονται ανάμεσα σε ένα σύνολο από μεθόδων που ονομάζονται σαν “Evolutionary Computation Techniques” και χαρακτηρίζονται από το

γεγονός ότι η λύση αναδεικνύεται μέσα από ένα σύνολο από επαναλήψεις γενεών υποψήφιων λύσεων και από τις οποίες επιλέγεται και επιβιώνει η καλύτερη.

Τα πλεονεκτήματα των Γενετικών Αλγορίθμων, σε συνδυασμό με τις παρατηρήσεις ότι υπάρχουν μεγάλες διαφοροποιήσεις στην απόδοση των Τεχνητών Νευρωνικών Δικτύων, όσο μεταβάλλονται διάφοροι εσωτερικοί παράμετροι των δικτύων, όπως είναι η αρχιτεκτονική, ο αριθμός των εισόδων κ.α., οδήγησαν στην συχνή εφαρμογή τους. Αποτελούν αλγόριθμους καθολικής βελτιστοποίησης (Global Optimization) και αναζήτησης γενικής εφαρμογής. Θεμελιώνονται σε αρχές εμπνευσμένες από τους πραγματικούς μηχανισμούς της Γενετικής που διέπουν τα βιολογικά συστήματα και την εξέλιξη των ειδών σύμφωνα με τη θεωρία του Δαρβίνου. Στα βιολογικά συστήματα ισχύει ο νόμος της φυσικής επιλογής, σύμφωνα με τον οποίο επιβιώνουν μόνο τα ισχυρότερα άτομα (survival of the fittest), τα οποία είναι κατάλληλα για αναπαραγωγή. Η αναπαραγωγή σε τέτοια συστήματα γίνεται συνήθως με την ανάμιξη των γενετικών χαρακτηριστικών δύο γονέων, που επιλέγονται με βάση τον παραπάνω νόμο, και οδηγεί στην γένεση νέων ατόμων που συμπληρώνουν τον υπάρχοντα πληθυσμό. Ο ανασυνδυασμός των γενετικών χαρακτηριστικών των γονέων, μαζί με την μικρή αλλά όχι μηδενική πιθανότητα μετάλλαξης τους από εξωτερικούς παράγοντες, δημιουργούν προϋποθέσεις παραγωγής νέων βελτιωμένων χαρακτηριστικών (αλλά και την πιθανότητα δημιουργίας χειρότερων χαρακτηριστικών) στον πληθυσμό των απογόνων. Ωστόσο, σύμφωνα με τον νόμο της επιβίωσης των καλύτερων ατόμων, η εξέλιξη που πραγματοποιείται είναι προσανατολισμένη προς την κατεύθυνση της σταδιακής βελτίωσης των χαρακτηριστικών του πληθυσμού και την ολοένα καλύτερη προσαρμογή τους στο περιβάλλον. Αυτή η δυναμική προσαρμογής και βελτίωσης, καθιστά τους πληθυσμούς βιώσιμους στις μεταβολές του περιβάλλοντος και συμβάλλει στην βελτίωση των χαρακτηριστικών των ειδών.

Η διαδικασία εφαρμογής Γενετικού Αλγόριθμου, όπως υλοποιήθηκε στο εργαλείο, αφορά αρχικά την κωδικοποίηση των ατόμων λύσεων με μια σειρά από bits που αντιπροσωπεύουν την αρχιτεκτονική, την τοπολογία του Τεχνητού Νευρωνικού Δικτύου.

Τα βήματα εφαρμογής του αλγορίθμου περιγράφονται στην συνέχεια.

- Βήμα 1: Αρχικοποίηση

Η διαδικασία εξέλιξης της τοπολογίας του Τεχνητού Νευρωνικού Δικτύου, ξεκινά με τον καθορισμό μιας τυχαίας δομής, με τυχαίο αριθμό νευρώνων στην είσοδο και τυχαίο αριθμό στο κάθε ένα από τα εσωτερικά επίπεδα, ενώ ο αριθμός των εξόδων

παραμένει σταθερός στο ένα. Αρχικά δημιουργείται ένα σύνολο ατόμων με τυχαίες τοπολογίες, τα οποία αποτελούν το σύνολο του αρχικού πληθυσμού. Ο αριθμός των ατόμων που δημιουργούνται είναι ένας σημαντικός παράγοντας και επηρεάζει τα αποτελέσματα του αλγορίθμου. Γενικά προτιμάται ένα μεγάλο μέγεθος ατόμων του πληθυσμού για την εξαγωγή καλύτερων αποτελεσμάτων. Στα πειράματα που αναφέρονται στη συνέχεια, λόγω του μεγάλου υπολογιστικού κόστους που έχει σαν συνέπεια την μεγάλη χρονική επεξεργασία του αλγορίθμου, παράγουμε 30 άτομα για τον κάθε πληθυσμό. Οι επαναλήψεις του Γενετικού Αλγόριθμου προτείνονται στις 200 επαναλήψεις. Αυτές οι επιλογές, αλλά και άλλες επιπλέον επιλογές που αφορούν το σχετικό Νευρωνικό Δίκτυο μπορούν να δοθούν από τον χρήστη του εργαλείου.

- Βήμα 2: Επιλογή

Η επιλογή των κατάλληλων ατόμων είναι μια αναγκαία διαδικασία στους Γενετικούς Αλγόριθμους. Για κάθε ένα άτομο στον παρών πληθυσμό μετράμε την απόδοση του, με την χρήση της συνάρτησης καταλληλότητας (fitness function), η οποία είναι μια συνάρτηση που βασίζεται στο Μέσο Σχετικό Σφάλμα MRE (Mean Relative Error) σε σχέση με την κατανομή πιθανοτήτων. Η συνάρτηση αυτή ορίζεται ως εξής:

$$eval(I_i) = \frac{1}{1 + MRE}$$

Όπου I_i αντιπροσωπεύει το κάθε ένα άτομο του πληθυσμού και το Μέσο Σχετικό Σφάλμα MRE (Mean Relative Error) ορίζεται σαν:

$$MRE(n) = \frac{\frac{1}{n}\sum_{i=1}^{n}\left|x_{act}(i) - x_{pred}(i)\right|}{x_{act}(i)}$$

Όπου n, ο αριθμός των προβλέψεων, $x_{act}(i)$ η αυθεντική πραγματική τιμή του δείγματος i και $x_{pred}(i)$ η προβλεπόμενη τιμή του δείγματος i.

Το συνολικό fitness του πληθυσμού είναι το άθροισμα της συνάρτησης:

$$F = \sum_{i=1}^{NSIZE} eval(I_i)$$

Η συνάρτηση καταλληλότητας (fitness) του κάθε ατόμου, αλλά και οι καλύτερες τιμές που λαμβάνονται για κάθε άτομο συγκεντρώνονται και σχεδιάζονται μετά το τέλος του αλγόριθμου σε γραφικές παραστάσεις.

Η πιθανότητα επιλογής p_i για κάθε ένα άτομο I_i σε σχέση με τον υπόλοιπο πληθυσμό ορίζεται ως εξής:

$$p_i = \frac{eval(I_i)}{F}$$

Ενώ η συσσωρευτική πιθανότητα επιλογής q_i για κάθε άτομο I_i είναι:

$$q_i = \sum_{j=1}^{i} p_j$$

Με βάση την συνάρτηση καταλληλότητας επιλέγεται το βέλτιστο άτομο του πληθυσμού και αποθηκεύεται. Στην συνέχεια, παράγουμε ένα τυχαίο πραγματικό αριθμό r στο διάστημα [0,1] και με βάση την ομοιόμορφη κατανομή, αν $r < q_1$ τότε επιλέγεται το άτομο I_1, αλλιώς επιλέγεται το i-οστό άτομο I_i, $2 \le i \le NSIZE$, έτσι ώστε $q_{i-1} < r < q_i$.

- Βήμα 3: Διασταύρωση

Η διασταύρωση εφαρμόζεται αφού επιλεχτούν δύο γονείς και με τυχαίο σημείο στην κωδικοποίηση των ατόμων, παράγεται ένα νέο άτομο, το οποίο είναι ο συνδυασμός των δύο γονέων στο σημείο της διασταύρωσης. Πιο συγκεκριμένα, για κάθε άτομο του νέου πληθυσμού παράγουμε έναν τυχαίο πραγματικό αριθμό r στο διάστημα [0,1] και με βάση την ομοιόμορφη κατανομή, αν $r < p_c$ τότε επιλέγεται το άτομο για διασταύρωση. Επιλέγοντας δύο άτομα, ένα ζευγάρι ατόμων, τους γονείς (parents) παράγουμε για κάθε ένα άτομο έναν τυχαίο ακέραιο αριθμό στο διάστημα [1, (αριθμός των κρυμμένων κόμβων-1)], ο οποίος ορίζει το σημείο διασταύρωσης.

- Βήμα 4: Μετάλλαξη

Με την εφαρμογή μετάλλαξης επιλέγεται τυχαία ένα άτομο από τον πληθυσμό και αλλάζει τον αριθμό των εισόδων ή / και τον αριθμό των νευρώνων στα εσωτερικά επίπεδα, προσθέτοντας ή αφαιρώντας έναν τυχαίο αριθμό νευρώνων.

Η εξέλιξη που πραγματοποιείται είναι προσανατολισμένη προς την κατεύθυνση της σταδιακής βελτίωσης των χαρακτηριστικών του πληθυσμού και έτσι μετά την επανειλημμένη εφαρμογή του αλγορίθμου λαμβάνουμε βελτιωμένες αρχιτεκτονικές.

4.3.3 Περιγραφή Χρήσης Νευρωνικών Δικτύων και Γενετικού Αλγόριθμου

Στην πειραματική μελέτη και στην υλοποίηση του εργαλείου χρησιμοποίησα αρχιτεκτονική Μονοκατευθυντικού Νευρωνικού Δικτύου (Feedforward Neural Network) με χρήση BackPropagation και με ένα επίπεδο νευρώνων εισόδου, πηγής

δεδομένων, τρία κρυμμένα επίπεδα με αριθμό υπολογιστικών νευρώνων καθορισμένο είτε από τον χρήστη, είτε με τον Γενετικό Αλγόριθμο και μία έξοδο.

Κατά την διάρκεια των πειραμάτων αποφάσισα να περιορίσω τους παράγοντες που επηρεάζουν το κόστος ανάπτυξης λογισμικού και να μην λάβω καθόλου υπόψιν δεύτερης σημασίας παράγοντες. Έτσι από τα σύνολα δεδομένων χρησιμοποίησα μόνο τις μετρικές του μεγέθους και της προσπάθειας ανάπτυξης λογισμικού που επηρεάζουν άμεσα το κόστος και είναι κοινά αποδεκτές σαν πρωταρχικοί παράγοντες. Έτσι χρησιμοποιήθηκε από το σύνολο δεδομένων του COCOMO και του Kemerer οι γραμμές του κώδικα (lines of code (LOC)) και από το σύνολο δεδομένων του Desharnais και του Albrecht τα λειτουργικά σημεία (function points (FP)).

Τα κριτήρια αξιολόγησης που χρησιμοποιήθηκαν για τα πειράματα και για το προτεινόμενο εργαλείο που υλοποιήθηκε, είναι η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (Correlation Coefficient) και το Σχετικό Μέσο Σφάλμα (MRE).

Ο συνδυασμός αυτών των κριτηρίων αξιολόγησης διασφαλίζει ότι η αξιολόγηση του μοντέλου θα είναι έγκυρη και ορθή. Αρχικά η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος αξιολογεί και ανιχνεύει την ποιότητα των προβλέψεων, αλλά λόγω των περιορισμών που αναφέρθηκαν προηγουμένως, η χρήση του μέτρου αυτού πρέπει να γίνεται μόνο σε επιλεγμένες περιπτώσεις και για να είμαστε εκ του ασφαλούς πρέπει να συνδυάζεται με άλλα μέτρα σφάλματος. Έτσι, αναγκαία κρίθηκε η χρήση του Συντελεστή Συσχέτισης, ο οποίος, εκτός του ότι μας επιτρέπει να παρακολουθήσουμε την εξέλιξη της τροχιάς των σημείων των πραγματικών τιμών και των προβλεπόμενων, όσων αφορά την συσχέτισή τους, αποτρέπει την πιθανόν λανθασμένη μας αξιολόγηση να εκλάβουμε τις προβλέψεις που κάνει το μοντέλο και είναι κοντά στο μέσο όρο σαν καλές προβλέψεις. Αν για παράδειγμα παραχθούν προβλέψεις με μικρή απόκλιση από τις πραγματικές τιμές, αλλά στην ουσία οι τιμές της πρόβλεψης σχηματίζουν μια ευθεία γραμμή κοντά στη μέση τιμή των αυθεντικών, πραγματικών δειγμάτων, ο Συντελεστής Συσχέτισης θα δείξει φτωχά αποτελέσματα σηματοδοτώντας ότι η μέθοδος πρόβλεψης κάνει μια προσέγγιση στην μέση τιμή, άρα και τίποτα αξιόλογο. Τέλος, για την αξιολόγηση των αποτελεσμάτων του μοντέλου χρησιμοποιήθηκε και ένα τρίτο κριτήριο αξιολόγησης το Σχετικό Μέσο Σφάλμα (Mean Relative Error), το οποίο δίνει το σφάλμα των προβλέψεων εκφρασμένο ως την απόλυτη τιμή της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές.

4.4 Πειραματική Μελέτη

4.4.1 Σκοπός

Σκοπός αυτής της ενότητας είναι η περιγραφή των λειτουργιών και δυνατοτήτων του εργαλείου της Ward Systems Group, NeuroShell 2, καθώς και η αναλυτική παρουσίαση των πειραματικών τρεξιμάτων που διεξήχθησαν με την βοήθεια του εργαλείου αυτού. Στη συνέχεια ακολουθεί η παρουσίαση των αποτελεσμάτων και των συμπερασμάτων.

Το εργαλείο NeuroShell 2 παρέχει τα εξής επίπεδα χρήσης:

α. Για αρχάριους χρήστες, για δημιουργία μιας απλής αρχιτεκτονικής Νευρωνικού Δικτύου και εφαρμογή εκπαίδευσης και δοκιμής του δικτύου.

β. Για προχωρημένους χρήστες, για δημιουργία μιας πολύπλοκης αρχιτεκτονικής Νευρωνικού Δικτύου και εφαρμογή εκπαίδευσης και δοκιμής του δικτύου.

Στα πειράματα που διεξήχθησαν επιλέξαμε το δεύτερο επίπεδο, αφού απαιτούμε μια πιο εξειδικευμένη αρχιτεκτονική και χρήση Τεχνητού Νευρωνικού Δικτύου.

4.4.2 Περιγραφή του Προβλήματος

Χρησιμοποιώντας διάφορα διαθέσιμα σύνολα δεδομένων, που περιέχουν τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ή τα λειτουργικά σημεία ενός λογισμικού και τις τιμές της προσπάθειας που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού, προσπαθούμε να προβλέψουμε την προσπάθεια ή το κόστος ανάπτυξης λογισμικού κάποιων έργων που δεν γνωρίζει το Τεχνητό Νευρωνικό Δίκτυο. Μπορούμε να μεταβάλουμε τον αριθμό των εισόδων, τον αριθμό των νευρώνων και την συνάρτηση δραστηριοποίησης σε κάθε πλάκα (slab) στο εσωτερικό, κρυμμένο επίπεδο, δηλαδή την αρχιτεκτονική του δικτύου. Επίσης, μπορούμε να ορίσουμε ειδικές παραμέτρους και κριτήρια για την εκπαίδευση του δικτύου και να προσπαθήσουμε με διάφορους συνδυασμούς και με διάφορους τρόπους επιλογής των δεδομένων εισόδου να πετύχουμε καλύτερα αποτελέσματα.

4.4.3 Περιγραφή Συνόλων Δεδομένων

Για να μπορέσει να υπολογιστεί με ακρίβεια το κόστος ανάπτυξης λογισμικού είναι σημαντική η συλλογή ενός συνόλου δεδομένων που να είναι αμιγές, συνεπή, ομοιογενή, να παρουσιάζει τα ίδια μεγέθη μετρικής και να προέρχεται από ένα ομογενή περιβάλλον, όπως για παράδειγμα μια εταιρία, έτσι ώστε να μπορέσει να αποτελέσει μια καλή και αξιόπιστη βάση του υπολογισμού. Τα δεδομένα όμως δυστυχώς είναι διασκορπισμένα και συνήθως είχαν μικρή ποιότητα και ποσότητα, με αποτέλεσμα να είναι ακόμα πιο δύσκολη η διαμόρφωση θεωριών και η κατασκευή συστημάτων υποστήριξης αποφάσεων με ακριβή αποτελέσματα. Αυτό ενισχύεται και από τα συμπεράσματα μιας μελέτης που χρησιμοποιήθηκαν παραμετρικοί μέθοδοι (Putnam 1978, Albrecht 1979, Boehm 1981, Bailey and Basili 1981) στην οποία συγκρίθηκαν σύνολα δεδομένων διαφόρων μεγεθών, τα οποία λήφθηκαν από διαφορετικά περιβάλλοντα και παρατηρήθηκαν πολύ χαμηλές επιδόσεις. Άρα υπάρχει ανάγκη για εύρεση συνόλων δεδομένων που να μην είναι διφορούμενα, ελλιπή και να μην έχουν ληφθεί ούτε σε διαφορετική χρονολογική περίοδο αλλά ούτε και σε διαφορετικά περιβάλλοντα.

Η δυσκολία εύρεσης αξιόπιστων και ποιοτικών συνόλων δεδομένων είναι πολύ μεγάλη, καθώς δεν υπάρχουν δημοσιευμένα ή προσβάσιμα προς το κοινό, και για αυτό η μελέτη αυτή περιορίστηκε στην χρήση των πέντε συνόλων δεδομένων που βρέθηκαν και που περιγράφονται στη συνέχεια. Τα σύνολα αυτά δεδομένων λήφθηκαν μετά από εκτενή αναζήτηση στο διαδίκτυο, με εξαίρεση το πέμπτο σύνολο δεδομένων, το οποίο βρέθηκε μετά από επικοινωνία με τον καθηγητή Jean-Marc Desharnais, Ph.D., ο οποίος είχε την ευγένεια να διαθέσει το σύνολο δεδομένων που χρησιμοποίησε στην σχετική έρευνα του [7], η οποία αναφέρεται στην βιβλιογραφία. Τα σύνολα δεδομένων που βρέθηκαν είναι τα εξής:

- [COCOMO ’81]
- [Kemerer 1987]
- [COCOMO ’81&Kemerer 1987]
- [Albrecht, Gaffney 1983]
- [Desharnais 1989]

Η λογική που ακολουθήθηκε για την διεξαγωγή των πειραμάτων χωρίζεται σε τρεις κατηγορίες όπως εξηγούνται πιο κάτω:

- Με χρήση μιας μετρικής LOC(i) ή FP(i) για την πρόβλεψη της μετρικής EFF(i).

  Μπορεί να χρησιμοποιηθεί αν διαθέτουμε γνώση για την προσπάθεια που χρειάστηκε για να αναπτυχθούν οι γραμμές του κώδικα ή τα λειτουργικά σημεία και για την προσπάθεια ανάπτυξης περασμένων έργων. Δίνουμε τις γραμμές του κώδικα ή τα λειτουργικά σημεία ενός άγνωστου έργου και προβλέπουμε την προσπάθεια ανάπτυξης του έργου αυτού.

- Με χρήση δύο μετρικών LOC(i) ή FP(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

  Μπορεί να χρησιμοποιηθεί αν διαθέτουμε γνώση για την προσπάθεια που χρειάστηκε για να αναπτυχθούν οι γραμμές του κώδικα ή τα λειτουργικά σημεία και για την προσπάθεια ανάπτυξης περασμένων έργων και προβλέπουμε την προσπάθεια ανάπτυξης ενός άγνωστου έργου, χωρίς να γνωρίζουμε πληροφορίες για το έργο αυτό. Ωφελεί η χρήση σε περίπτωση που έχουμε διεξαγωγή παρόμοιων μεγέθους έργων.

- Με χρήση δύο μετρικών LOC(i), LOC(i+1) ή FP(i+1), FP(i+1) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

  Μπορεί να χρησιμοποιηθεί αν διαθέτουμε γνώση για την προσπάθεια που χρειάστηκε για να αναπτυχθούν οι γραμμές του κώδικα ή τα λειτουργικά σημεία και για την προσπάθεια ανάπτυξης περασμένων έργων και προβλέπουμε την προσπάθεια ανάπτυξης ενός άγνωστου έργου, αν δώσουμε σαν είσοδο τις γραμμές του κώδικα ή τα λειτουργικά σημεία του έργου αυτού.

Πίνακας 4.1: Επεξήγηση Συμβόλων

	Επεξήγηση Συμβόλων
LOC(i)	Ο αριθμός των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού. Αν για παράδειγμα ο χρήστης δώσει για i την τιμή 3, τότε θα χρησιμοποιηθούν το project της σειράς 1, το project της σειράς 2 και το project της σειράς 3 από τα αρχεία των συνόλων δεδομένων. Στην επόμενη επανάληψη, χρησιμοποιούνται το project της σειράς 2, το project της σειράς 3 και το project της σειράς 4. Η λήψη δειγμάτων συνεχίζεται με τον ίδιο τρόπο.
FP(i)	Ο αριθμός των λειτουργικών σημείων που περιέχονται σε ένα σύστημα λογισμικού. Το παράδειγμα είναι ανάλογο του παραδείγματος του LOC.
EFF(i+1)	Η προσπάθεια που καταβάλλεται ή το κόστος ανάπτυξης ενός έργου λογισμικού, που συνήθως υπολογίζεται σε άνθρωπο-μήνες (person-months). Αν για παράδειγμα ο χρήστης δώσει για i την τιμή 3, τότε η πρόβλεψη θα αποτελέσει την $4^{η}$ τιμή της προσπάθειας, δηλαδή του project της σειράς 4 από τα αρχεία των συνόλων δεδομένων. Στην επόμενη επανάληψη, η πρόβλεψη θα αποτελέσει την $5^{η}$ τιμή της προσπάθειας, δηλαδή του project της σειράς 5. Η διεξαγωγή πρόβλεψης συνεχίζεται με αυτό τον τρόπο.
i	Ανάλογα με την επιλογή και την κρίση του χρήστη, δίνεται μια ακέραια τιμή στο i, το οποίο αντιπροσωπεύει τον αριθμό των έργων που επιθυμεί να δοθεί σαν είσοδος και στην συνέχεια σαν έξοδος του Τεχνητού Νευρωνικού Δικτύου. Δηλαδή αποτελεί τον αριθμό των σειρών, των projects που θα επιλεχτούν από τα αρχεία των συνόλων δεδομένων.
i+1	Ανάλογα με την επιλογή και την κρίση του χρήστη, όταν δίνεται μια ακέραια τιμή στο i, τότε αντιπροσωπεύει την αμέσως επόμενη τιμή στον αριθμό των σειρών ή των έργων που επιθυμεί να δοθεί σαν είσοδος και στην συνέχεια σαν έξοδος του Τεχνητού Νευρωνικού Δικτύου.

4.4.3.1 Σύνολο Δεδομένων COCOMO

Το COCOMO (Boehm 1981) σύνολο δεδομένων περιέχει 63 έργα ανάπτυξης λογισμικού, τα οποία λήφθηκαν από έργα ανάπτυξης ενός Οίκου Ανάπτυξης Λογισμικών [21]. Κάθε ένα έργο περιγράφεται από 17 χαρακτηριστικά που μεταβάλουν το κόστος (cost drivers) και τα οποία παρουσιάζονται στον πίνακα 4.2. Αυτό το σύνολο δεδομένων χρησιμοποιήθηκε στην έρευνα του Boehm COCOMO.

Πίνακας 4.2: Χαρακτηριστικά που μεταβάλουν το κόστος στο COCOMO [1], [21].

Reliability	Programmer Capability
Data Base Size	Applications Experience
Complexity	Platform Experience
Required Reusability	Language & Tool Experience
Documentation	Personnel Continuity
Execution Time Constraint	Use of Software Tools
Main Storage Constraint	Multi-site Development
Platform Volatility	Required Schedule
Analyst Capability	

Τα 75 πρώτα τρεξίματα (τα οποία υπάρχουν σε λεπτομέρεια σε ηλεκτρονική μορφή) έγιναν με το σύνολο δεδομένων του COCOMO, του οποίου χρησιμοποιήθηκαν μόνο οι δύο στήλες των δεδομένων που αφορούν τις τιμές της προσπάθειας (EFF) και τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC). Αποφάσισα να ακολουθήσω μια συγκεκριμένη μεθοδολογία εκτέλεσης των πειραμάτων, έτσι ώστε να μπορέσουν να εξαχθούν κάποια συμπεράσματα και να μπορέσουν να εφαρμοστούν κάποιες βελτιστοποιήσεις στη συνέχεια. Στα πρώτα 25 αποτελέσματα χρησιμοποιώ μόνο τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για τα συγκεκριμένα έργα. Στα επόμενα 25 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και την δεδομένη προσπάθεια (EFF) για το λογισμικό αυτό και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για το επόμενο έργο. Στα επόμενα 25 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC), μαζί με τον κώδικα για το αμέσως επόμενο έργο και την δεδομένη προσπάθεια για τα έργα και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για το αμέσως επόμενο έργο, του οποίου δόθηκαν σαν είσοδος οι γραμμές του κώδικα.

4.4.3.2 Σύνολο Δεδομένων KEMERER

Το σύνολο δεδομένων KEMERER περιλαμβάνει 15 έργα ανάπτυξης λογισμικού, τα οποία λήφθηκαν από μια εθνική εταιρία που παρέχει συμβουλές και υπηρεσίες ηλεκτρονικών υπολογιστών, που ειδικεύεται στην σχεδίαση και στην ανάπτυξη λογισμικών που επεξεργάζονται δεδομένα. Πηγή αυτού του συνόλου δεδομένων είναι το άρθρο Kemerer, C.F. 'An Empirical Validation of Software Cost Estimation Models', CACM, 30(5), pp416-429,1987. Τα χαρακτηριστικά που αναφέρονται παρουσιάζονται στον πίνακα 4.3.

Πίνακας 4.3: Χαρακτηριστικά στο KEMERER [22].

Actual Effort	Actual Project Effort (Measured in Man Months)
Duration	Project Duration (Measured in Months)
KSLOC	Thousand Source lines of Code
UA_FP	Unadjusted Function Point Count
FP	Adjusted Function Points

Για τα επόμενα 30 τρεξίματα (τα οποία υπάρχουν σε λεπτομέρεια σε ηλεκτρονική μορφή) χρησιμοποιήθηκε το σύνολο δεδομένων του KEMERER, το οποίο χρησιμοποιήθηκαν μόνο οι δύο στήλες των δεδομένων που αφορούν τις τιμές της προσπάθειας (EFF) και τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC). Αποφάσισα να ακολουθήσω μια συγκεκριμένη μεθοδολογία εκτέλεσης των πειραμάτων, έτσι ώστε να μπορέσουν να εξαχθούν κάποια συμπεράσματα και να μπορέσουν να εφαρμοστούν κάποιες βελτιστοποιήσεις στη συνέχεια. Στα πρώτα 10 αποτελέσματα χρησιμοποιώ μόνο τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα συγκεκριμένα έργα (EFF). Στα επόμενα 10 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και την δεδομένη προσπάθεια για το λογισμικό αυτό και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα επόμενα έργα. . Στα επόμενα 10 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC), μαζί με τον κώδικα για το

αμέσως επόμενο έργο και την δεδομένη προσπάθεια για τα έργα και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για το αμέσως επόμενο έργο, του οποίου δόθηκαν σαν είσοδος οι γραμμές του κώδικα.

4.4.3.3 Σύνολο Δεδομένων COCOMO&KEMERER

Στην επόμενη σειρά πειραμάτων (τα οποία υπάρχουν σε λεπτομέρεια σε ηλεκτρονική μορφή) έχει ακολουθηθεί μια μεθοδολογία εμπνευσμένη από την μελέτη Srinivasan & Fisher κατά την οποία έχει χρησιμοποιηθεί το σύνολο δεδομένων COCOMO για να προβλεφθούν οι τιμές του συνόλου δεδομένων KEMERER. Πιο συγκεκριμένα, το σύνολο δεδομένων που χρησιμοποιήσαμε για τα πειράματα αποτελείται από τα δεδομένα του συνόλου COCOMO και είναι συνδυασμένα με τα δεδομένα του συνόλου KEMERER. Για τα 50 τρεξίματα χρησιμοποιήθηκαν 63 δείγματα για την εκπαίδευση του δικτύου, ενώ τα υπόλοιπα που απομέναν από το σύνολο δεδομένων KEMERER χρησιμοποιήθηκαν για την δοκιμή του δικτύου. Αξίζει να επισημάνουμε πως χρησιμοποιήθηκαν μόνο οι δύο στήλες των δεδομένων που αφορούν τις τιμές της προσπάθειας (EFF) και τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC).Αποφάσισα να ακολουθήσω μια συγκεκριμένη μεθοδολογία της σειράς εκτέλεσης των πειραμάτων, έτσι ώστε να μπορέσουν να εξαχθούν κάποια συμπεράσματα και να μπορέσουν να εφαρμοστούν κάποιες βελτιστοποιήσεις στη συνέχεια. Στα πρώτα 25 αποτελέσματα χρησιμοποιώ μόνο τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα συγκεκριμένα έργα. Στα επόμενα 25 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ενός λογισμικού (LOC) και την δεδομένη προσπάθεια για το λογισμικό αυτό και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα επόμενα έργα.

4.4.3.4 Σύνολο Δεδομένων ALBRECHT

Το σύνολο δεδομένων Albrecht, 1983 περιέχει 24 έργα ανάπτυξης τα οποία αναπτύχθηκαν από την IBM DP οργανισμό υπηρεσιών. Πηγή του συνόλου είναι το άρθρο των Albrecht, A.J. and J.R. Gaffney, 'Software Function Source Lines of Code,

and Development Effort Prediction: A Software Science Validation', IEEE trans. on Softw. Eng., 9(6), pp639-648,1983. Στον πίνακα 4.4 αναφέρονται τα χαρακτηριστικά του συνόλου δεδομένων Albrecht.

Πίνακας 4.4: Χαρακτηριστικά στο ALBRECHT [23]

Actual Effort	Actual Project Effort (Measured in Thousand Work Hours)
SLOC	Thousand Source Lines of Code
IN	No. of Inputs
OUT	No. of Outputs
FILE	No. of Master Files
INQ	No. of Inquiries
Function Points	Function Point Count

Για τα επόμενα 30 τρεξίματα (τα οποία υπάρχουν σε λεπτομέρεια σε ηλεκτρονική μορφή) χρησιμοποιήθηκε το σύνολο δεδομένων του ALBRECHT, το οποίο χρησιμοποιήθηκαν μόνο οι δύο στήλες των δεδομένων που αφορούν τις τιμές της προσπάθειας (EFF) και τον αριθμό των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP). Αποφάσισα να ακολουθήσω μια συγκεκριμένη μεθοδολογία εκτέλεσης των πειραμάτων, έτσι ώστε να μπορέσουν να εξαχθούν κάποια συμπεράσματα και να μπορέσουν να εφαρμοστούν κάποιες βελτιστοποιήσεις στη συνέχεια. Στα πρώτα 10 αποτελέσματα χρησιμοποιώ μόνο τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP) και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα συγκεκριμένα έργα (EFF). Στα επόμενα 10 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP) και την δεδομένη προσπάθεια για το λογισμικό αυτό και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα επόμενα έργα. . Στα επόμενα 10 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP), μαζί με τον κώδικα για το αμέσως επόμενο έργο και την δεδομένη προσπάθεια για τα έργα και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για το αμέσως επόμενο έργο.

4.4.3.5 Σύνολο Δεδομένων DESHARNAIS

Το σύνολο δεδομένων [Desharnais 1989] περιέχει παρατηρήσεις για περισσότερα από 80 συστήματα ανάπτυξης, τα οποία έχουν παρθεί από ένα Καναδικό Οίκο Ανάπτυξης Λογισμικού στα τέλη της δεκαετίας του 1980 και που συγκεντρώθηκαν μέσα σε τέσσερα χρόνια. Τα δεδομένα αυτά χρησιμοποιήθηκαν σε πολλές μελέτες για την πρόβλεψη της προσπάθειας ή του κόστους ανάπτυξης και περιλαμβάνουν μετρήσεις της προσπάθειας που χρειάζεται για την ανάπτυξη ενός έργου (project effort), την διάρκεια του έργου (project duration), το επίπεδο της εμπειρίας του προσωπικού ανάπτυξης (level of experience) σε σχέση με τον εξοπλισμό ανάπτυξης και την διοίκηση έργου, τον αριθμό των βασικών διεργασιών και των οντοτήτων (basic transactions and data entities) και τις ακατέργαστες μετρήσεις των Function Points (FP). Στον πίνακα 4.5 περιγράφονται τα χαρακτηριστικά του συνόλου δεδομένων DESHARNAIS.

Πίνακας 4.5: Χαρακτηριστικά στο DESHARNAIS [7].

Project name	Numeric identifier
Effort	Measured in hours
ExpEquip	Team experience in years
ExpProjMan	Project manager’s experience in years
Trans	Number of transactions processed
Entities	Number of entities
RawFPs	Unadjusted function points
AdjFPs	Adjusted function points
DevEnv	Development environment
YearFin	Year of completion

Για τα επόμενα 75 τρεξίματα (τα οποία υπάρχουν σε λεπτομέρεια σε ηλεκτρονική μορφή) χρησιμοποιήθηκε το σύνολο δεδομένων του DESHARNAIS, από το οποίο χρησιμοποιήθηκαν μόνο οι δύο στήλες των δεδομένων που αφορούν τις τιμές της προσπάθειας (EFF) και τον αριθμό των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP). Αποφάσισα να ακολουθήσω μια συγκεκριμένη

μεθοδολογία εκτέλεσης των πειραμάτων, έτσι ώστε να μπορέσουν να εξαχθούν κάποια συμπεράσματα και να μπορέσουν να εφαρμοστούν κάποιες βελτιστοποιήσεις στη συνέχεια. Στα πρώτα 25 αποτελέσματα χρησιμοποιώ μόνο τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP) και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα συγκεκριμένα έργα (EFF). Στα επόμενα 25 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP) και την δεδομένη προσπάθεια για το λογισμικό αυτό και εξάγω σαν αποτέλεσμα την προσπάθεια που χρειάζεται για τα επόμενα έργα. . Στα επόμενα 25 αποτελέσματα χρησιμοποιώ τα δεδομένα του αριθμού των ακατέργαστων μετρήσεων των λειτουργικών σημείων (FP), μαζί με τα λειτουργικά σημεία για το αμέσως επόμενο έργο και την δεδομένη προσπάθεια για τα έργα και εξάγω σαν αποτέλεσμα την προσπάθεια (EFF) που χρειάζεται για το αμέσως επόμενο έργο, του οποίου δόθηκαν σαν είσοδος τα FP.

4.4.4 Λεπτομερής Περιγραφή Πειραματικών Τρεξιμάτων

4.4.4.1 Χρήση Σύνολου Δεδομένων COCOMO

- Με χρήση μιας μετρικής LOC(i) για την πρόβλεψη της μετρικής EFF(i).

Στον πρώτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ την LOC(i) τιμή για να προβλέψω το EFF(i). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 13.

Στον δεύτερο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων

σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 12.

Στον τρίτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 11.

Στον τέταρτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2), LOC(i+3) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον πέμπτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2), LOC(i+3), LOC(i+4) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 19.

- Με χρήση δύο μετρικών LOC(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον έκτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 12.

Στον έβδομο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 11.

Στον όγδοο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον ένατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 8, σε 12, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 9.

Στον δέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3), LOC(i+4), EFF(i+4) για να προβλέψω το EFF(i+5). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 8.

- Με χρήση δύο μετρικών LOC(i+1) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον ενδέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για

εκπαίδευση του δικτύου είναι 46, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον δωδέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 45, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον δέκατο τρίτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 44, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον δέκατο τέταρτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3), LOC(i+4) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 6, σε 12, σε 15 και σε 20. Ο αριθμός των έργων

(projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 43, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Στον δέκατο πέμπτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3), LOC(i+4), EFF(i+4), LOC(i+5) για να προβλέψω το EFF(i+5). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 42 για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 10.

Τα καλύτερα αποτελέσματα που είχαν οι σειρές των πειραμάτων που διεξήχθησαν φαίνονται με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων που ακολουθούν. Τα αποτελέσματα αυτά συλλέχτηκαν και παρουσιάζονται στον πίνακα 4.7 μαζί με τα συμπεράσματα και τις παρατηρήσεις.

Πίνακας 4.2.1: Αποτελέσματα Συνόλου Δεδομένων COCOMO

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO	1-3-3-3-1	LOC (1)	EFF	45-5-13	0.8213	0.64293	2753765.8509	12.7297	723.2369	1.6839	0.53892	184218.3343	12.7875	425.4908
COCOMO	1-6-6-6-1				0.78408	0.6404	2509864.1001	7.8266	582.8054	1.1613	0.539	87619.4121	7.7633	275.485
COCOMO	1-9-9-9-1				0.81929	0.64392	2740315.5632	8.7561	636.4714	1.2483	0.53882	101237.7738	8.719	303.5144
COCOMO	1-15-15-15-1				0.86163	0.66331	3030889.5322	9.6446	694.5725	1.3361	0.53874	115986.2982	9.6222	329.853
COCOMO	1-20-20-20-1				0.79892	0.6496	2605755.3633	4.8265	546.2267	0.95346	0.53885	59063.6007	4.6713	185.6626
COCOMO	2-3-3-3-1	LOC (2)	EFF	45-5-12	0.72949	0.68498	2161062.1668	3.5736	425.1072	0.92732	0.4188	60931.7572	3.4542	160.2676
COCOMO	2-6-6-6-1				0.71468	0.70047	2074210.6153	1.8943	399.2909	0.92453	0.45385	60565.3566	1.6324	119.0607
COCOMO	**2-9-9-9-1**				**0.71759**	**0.69648**	**2091156.8185**	**0.87066**	**408.998**	**0.9627**	**0.44737**	**65669.9967**	**0.5254**	**101.6436**
COCOMO	2-15-15-15-1				0.6944	0.72046	1958154.4929	0.73263	424.4816	1.0772	NaN	82226.9433	0.82342	131.4333
COCOMO	2-20-20-20-1				0.72186	0.69643	2116122.1525	0.6435	431.0357	1.0167	0.50176	73244.8485	0.67192	115.711
COCOMO	3-3-3-3-1	LOC (3)	EFF	45-5-11	0.78926	0.66754	2527812.7978	5.6291	517.4098	0.98046	0.26911	73953.1649	5.62	211.2001
COCOMO	3-6-6-6-1				0.83312	0.64982	2816549.6543	8.6017	610.4929	1.1527	0.47968	102217.0002	9.0442	299.7655
COCOMO	3-9-9-9-1				0.87348	0.62406	3096052.7516	5.4735	594.4936	0.98151	0.11871	74112.1435	5.4366	206.1339
COCOMO	3-15-15-15-1				0.80472	0.68566	2627799.7551	2.7315	483.3246	0.93228	0.36662	66863.5571	2.5787	144.7559
COCOMO	3-20-20-20-1				0.76779	0.68775	2392157.212	1.9346	428.4629	0.94326	0.37305	68448.244	1.729	127.795
COCOMO	4-3-3-3-1	LOC (4)	EFF	45-5-10	0.44721	0.9192	812716.9154	2.6342	330.2229	1.073	-0.62515	95952.8314	1.7701	155.1867
COCOMO	4-6-6-6-1				0.51106	0.87376	1061375.5118	0.70944	374.2806	1.0874	NaN	98542.57	0.84464	153.4
COCOMO	4-9-9-9-1				0.507	0.86964	1044566.6979	0.64102	357.594	1.0129	0.63806	85507.4626	0.55573	132.0872

Πίνακας 4.2.2: Αποτελέσματα Συνόλου Δεδομένων COCOMO

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO	4-15-15-15-1				0.45731	0.93173	849840.7081	1.2075	318.9161	1.0857	-0.36418	98244.7501	0.72839	148.3723
COCOMO	4-20-20-20-1				0.31796	0.96189	410846.3463	1.1028	278.6718	1.0867	-0.29029	98432.1083	0.73167	150.7033
COCOMO	5-3-3-3-1	LOC (5)	EFF	45-5-9	0.539	0.84794	1182209.929	0.68449	366.2428	1.0729	0.48324	104824.3837	0.75389	161.8526
COCOMO	**5-6-6-6-1**				**0.57749**	**0.82066**	**1357087.7339**	**0.83338**	**345.5678**	**0.95258**	**0.4972**	**82636.863**	**0.74621**	**133.4605**
COCOMO	5-9-9-9-1				0.55621	0.82867	1258919.7944	2.9266	394.742	0.83376	0.59698	63307.617	2.9327	175.1248
COCOMO	5-15-15-15-1				0.5681	0.82324	1313313.338	2.1811	357.5909	0.87071	0.51363	69042.2636	1.9241	154.4712
COCOMO	**5-20-20-20-1**				**0.58427**	**0.83076**	**1389156.1593**	**1.2695**	**341.4701**	**0.99617**	**0.63815**	**90372.4277**	**0.80458**	**138.7993**
COCOMO	2-3-3-3-1	LOC(1), EFF(1)	EFF (2)	45-5-12	0.79082	0.60393	2539708.4301	11.5902	544.6581	1.5017	0.20435	159783.0867	12.4747	394.6974
COCOMO	2-6-6-6-1				0.78881	0.60473	2526832.2823	13.4875	574.2725	1.7224	0.19026	210214.4189	14.597	455.9329
COCOMO	2-9-9-9-1				0.78552	0.60919	2505829.1369	16.309	618.5617	1.9951	0.15775	282030.7199	17.8026	517.0152
COCOMO	2-15-15-15-1				0.78859	0.6073	2525421.7713	13.128	571.5558	1.6322	0.15255	188764.6178	14.2383	425.5751
COCOMO	2-20-20-20-1				0.78923	0.60514	2529505.3225	13.3659	580.6526	1.715	0.18615	208401.3805	14.5008	454.1608
COCOMO	4-3-3-3-1	LOC(2), EFF(2)	EFF(3)	45-5-11	0.78382	0.61082	2493093.0689	14.2525	550.4643	1.7115	0.31742	225350.6853	15.6639	470.4169
COCOMO	4-6-6-6-1				0.78532	0.60975	2502598.0736	14.3456	569.9831	1.7078	0.38585	224365.7623	15.6896	468.5709
COCOMO	4-9-9-9-1				0.78447	0.61005	2497205.4617	13.8553	548.3645	1.6612	0.36436	212300.6457	15.1807	456.5512
COCOMO	4-15-15-15-1				0.78429	0.61039	2496047.4657	13.6689	545.9722	1.6372	0.38745	206201.2382	14.9897	449.1384
COCOMO	4-20-20-20-1				0.78333	0.61164	2489935.842	14.4575	555.0533	1.729	0.38755	229970.9171	15.9291	474.1844

Πίνακας 4.2.3: Αποτελέσματα Συνόλου Δεδομένων COCOMO

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO	6-3-3-3-1	LOC(3), EFF(3)	EFF(+1)	45-5-10	0.78652	0.62059	2513897.9351	6.2955	427.2301	0.98814	0.40014	81380.7761	6.7883	248.5209
COCOMO	6-6-6-6-1				0.7916	0.61739	2546489.8849	5.5382	427.124	0.93505	0.46398	72870.6111	5.7179	218.1669
COCOMO	6-9-9-9-1				0.79602	0.61021	2574984.2327	6.0298	450.0137	0.97755	0.26521	79645.0473	5.5588	231.6961
COCOMO	6-15-15-15-1				0.78512	0.61771	2504936.2027	8.7624	458.3558	1.1239	0.45204	105279.4862	8.7473	307.6881
COCOMO	6-20-20-20-1				0.78777	0.61575	2521900.8399	8.1122	450.1908	1.0705	0.44386	95504.5685	7.8942	281.5499
COCOMO	8-4-4-4-1	LOC(4), EFF(4)	EFF(+1)	45-5-9	0.79878	0.60126	2596394.7827	5.5553	446.2442	1.0028	0.065476	91573.7246	4.7571	251.9387
COCOMO	8-8-8-8-1				0.79862	0.60954	2595371.2149	5.0664	441.9378	0.95188	0.18218	82515.2466	4.3841	225.2665
COCOMO	8-12-12-12-1				0.78777	0.61575	2521900.8399	8.1122	450.1908	1.0705	0.44386	95504.5685	7.8942	281.5499
COCOMO	8-15-15-15-1				0.79933	0.60442	2599997.269	4.3101	410.9713	0.98193	0.068136	87807.6637	3.7311	223.1763
COCOMO	8-20-20-20-1				0.79952	0.60378	2601246.8459	4.0306	403.8066	0.98447	0.051619	88261.5243	3.5355	219.4737
COCOMO	10-3-3-3-1	LOC(5), EFF(5)	EFF(+1)	45-5-8	0.81048	0.58848	2673306.239	5.7866	492.814	0.96609	0.0568	93452.8233	3.9018	245.513
COCOMO	10-6-6-6-1				0.80411	0.60011	2631469.7496	4.8162	452.0421	0.97227	0.082103	94651.4575	3.6171	245.8542
COCOMO	10-9-9-9-1				0.80779	0.60054	2655633.6739	3.0594	425.9241	0.97482	-0.02963	95150.1594	2.4998	209.8303
COCOMO	10-15-15-15-1				0.89987	0.48775	3295503.0874	4.245	571.8904	0.97668	-0.46181	95512.867	2.7572	201.9601
COCOMO	10-20-20-20-1				0.88883	0.5108	3215157.4185	3.2846	558.3913	0.96131	-0.10672	92529.2768	2.4171	191.1592
COCOMO	3-3-3-3-1	LOC(2), EFF(1)	EFF (2)	46-5-10	0.75934	0.65341	2301037.0417	5.283	451.7594	0.96333	0.21235	77375.6716	4.7494	218.7979
COCOMO	3-6-6-6-1				0.74104	0.69111	2191471.5979	5.6044	458.0825	0.95912	0.39521	76699.5603	5.0915	222.1002

Πίνακας 4.2.4: Αποτελέσματα Συνόλου Δεδομένων COCOMO

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO	3-9-9-9-1				0.70799	0.71064	2000334.5214	3.7544	404.3019	0.90613	0.38352	68459.2749	3.3068	175.1478
COCOMO	**3-15-15-15-1**				**0.71569**	**0.70877**	**2044066.0647**	**0.92923**	**353.7588**	**0.9522**	**0.40724**	**75597.3066**	**0.55149**	**111.9895**
COCOMO	**3-20-20-20-1**				**0.72121**	**0.7053**	**2075729.8117**	**0.70128**	**382.8061**	**0.98873**	**0.4354**	**81508.7374**	**0.5229**	**120.5357**
COCOMO	5-3-3-3-1	LOC(3), EFF(2)	EFF(3)	45-5-10	0.75415	0.66912	2267874.7576	4.049	412.8627	0.92296	0.28988	77409.0623	2.6728	177.6606
COCOMO	5-6-6-6-1				0.74234	0.67031	2197416.2571	4.5311	430.1806	0.92327	0.26271	77460.9672	3.0206	190.9269
COCOMO	5-9-9-9-1				0.72823	0.68285	2114664.8602	4.507	411.5421	0.91286	0.37849	75724.2151	3.2094	194.2562
COCOMO	**5-15-15-15-1**				**0.75723**	**0.65835**	**2286430.0032**	**1.1601**	**361.9915**	**0.97982**	**0.19506**	**87241.5028**	**0.56702**	**123.4747**
COCOMO	5-20-20-20-1				0.73605	0.69001	2160321.6292	0.67515	387.1454	1.083	0.0065195	106581.8039	0.81248	159.8386
COCOMO	7-3-3-3-1	LOC(4), EFF(3)	EFF(4)	44-5-10	0.795	0.61732	2523672.4498	27.689	775.0908	2.9962	0.41734	897519.9941	18.7533	908.2327
COCOMO	7-6-6-6-1				0.79376	0.61842	2515806.1299	28.0818	768.8194	2.996	0.45878	897400.4705	18.9065	910.0609
COCOMO	7-9-9-9-1				0.77975	0.62156	2427780.4435	21.8403	657.9961	2.2531	0.45626	507537.707	14.4627	661.6893
COCOMO	7-15-15-15-1				0.78302	0.61841	2448169.8172	22.6942	675.074	2.2889	0.50226	523773.9459	14.832	674.5178
COCOMO	7-20-20-20-1				0.78193	0.61766	2441371.9405	20.9916	644.7783	2.0407	0.4992	416337.4458	13.3783	611.5862
COCOMO	9-4-4-4-1	LOC(5), EFF(4)	EFF(5)	43-5-10	0.78151	0.62263	2442034.2887	20.3703	674.4633	2.0816	0.4318	479603.6007	9.6019	649.9768
COCOMO	9-6-6-6-1				0.78131	0.62508	2440739.5619	22.1773	700.3733	2.4051	0.42135	640271.307	10.7866	749.7276
COCOMO	9-12-12-12-1				0.77921	0.62846	2427651.889	22.4051	709.2545	2.4967	0.4551	689966.4397	11.161	784.0484
COCOMO	9-15-15-15-1				0.78261	0.62235	2448911.9936	21.4207	692.9015	2.191	0.51131	531351.0545	10.0558	675.329
COCOMO	9-20-20-20-1				0.78872	0.62105	2487306.8413	23.2625	738.3448	2.3068	0.55157	589003.3411	10.685	716.3354
COCOMO	11-3-3-3-1	LOC(6), EFF(5)	EFF(6)	42-5-10	0.79451	0.62053	2518151.0916	24.9868	768.4439	12.3605	0.97461	877211.5959	13.8688	936.024
COCOMO	11-6-6-6-1				0.78534	0.62956	2460400.6804	24.3933	751.7504	12.4041	0.76176	883409.0264	13.6001	934.1268

Πίνακας 4.2.5: Αποτελέσματα Συνόλου Δεδομένων COCOMO

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO	11-9-9-9-1				0.7892	0.6253	2484600.4933	25.2632	760.4215	12.1182	0.94579	843162.0271	13.4781	916.3473
COCOMO	11-15-15-15-1				0.78929	0.6297	2485216.1175	26.8726	787.135	13.4012	0.85941	1031153.8366	14.5283	1008.5675
COCOMO	11-20-20-20-1				0.82045	0.62209	2685324.0855	33.4818	946.8561	17.1196	0.80321	1682751.1953	18.4196	1287.0363

Παρατηρήσεις:

Από τα αποτελέσματα των πειραμάτων με το σύνολο δεδομένων COCOMO φαίνονται έξι ικανοποιητικά αποτελέσματα, τα οποία παρουσιάζονται συνήθως σε τρεξίματα όπου υπήρχαν πολλοί νευρώνες στο εσωτερικό κρυμμένο επίπεδο. Επίσης, μια ακόμα αξιόλογη παρατήρηση είναι το γεγονός ότι οι τρεις τιμές των καλύτερων αποτελεσμάτων παρουσιάζονται στην μέθοδο όπου χρησιμοποιούνται οι γραμμές του κώδικα για να προβλεφτεί η προσπάθεια και οι άλλες τρεις εμφανίζονται μαζεμένες, χρησιμοποιώντας 15 ή 20 νευρώνες στο κρυμμένο επίπεδο στην τρίτη μέθοδο, η οποία αν κρίνουμε και από τις τιμές των αποτελεσμάτων σαν σύνολο, είναι καλύτερη και στην οποία χρησιμοποιούνται οι γραμμές του κώδικα δύο έργων και η προσπάθεια του ενός έργου για να προβλεφτεί η προσπάθεια για το δεύτερο έργο.

Τα καλύτερα αποτελέσματα που πετυχαίνονται στο σύνολο μεμονωμένα είναι τα εξής: Normalized RMSE = 0,83376 το σφάλμα στη τιμή, CC = 0,97461 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 52,29% η απόκλιση ή αλλιώς 47,71% η σύγκλιση. Ενώ το καλύτερο αποτέλεσμα πειράματος είναι αυτό που έγινε με τοπολογία δικτύου 2-9-9-9-1 (αριθμός νευρώνων εισόδου – κρυμμένους σε κάθε ένα από τα τρία εσωτερικά επίπεδα – εξόδου) και είχε αποτελέσματα:
Normalized RMSE = 0,9627 το σφάλμα στη τιμή, CC = 0,44737 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 52,54% η απόκλιση ή αλλιώς 47,46% η σύγκλιση στις πραγματικές τιμές, μια ικανοποιητική σχετικά επιτυχία

4.4.4.2 Χρήση Σύνολου Δεδομένων KEMERER

- Με χρήση μιας μετρικής LOC(i) για την πρόβλεψη της μετρικής EFF(i).

Στον πρώτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ την LOC(i) τιμή για να προβλέψω το EFF(i). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 9, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

Στον δεύτερο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1)για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 8, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

- Με χρήση δύο μετρικών LOC(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον τρίτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 8, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

Στον τέταρτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 7, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

- Με χρήση δύο μετρικών LOC(i+1) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον πέμπτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 8, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

Στον έκτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων

(projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 7, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 3.

Τα καλύτερα αποτελέσματα που είχαν οι σειρές των πειραμάτων που διεξήχθησαν φαίνονται με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων που ακολουθούν. Τα αποτελέσματα αυτά συλλέχτηκαν και παρουσιάζονται στον πίνακα 4.7 μαζί με τα συμπεράσματα και τις παρατηρήσεις.

Πίνακας 4.3.1: Αποτελέσματα Συνόλου Δεδομένων KEMERER

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
KEMERER	**1-3-3-3-1**	**LOC (1)**	**EFF(1)**	**9-3-3**	**0.76079**	**0.75026**	**49409.0639**	**0.9635**	**126.1515**	**0.63212**	**0.87793**	**3129.8062**	**0.48763**	**53.4137**
KEMERER	**1-6-6-6-1**				**0.82501**	**0.77251**	**58102.7317**	**0.91535**	**130.4524**	**0.70659**	**0.87961**	**3910.7112**	**0.4585**	**52.9389**
KEMERER	**1-9-9-9-1**				**0.69446**	**0.76166**	**41169.3677**	**0.53244**	**112.0366**	**0.77448**	**0.87848**	**4698.333**	**0.24051**	**49.5276**
KEMERER	**1-15-15-15-1**				**0.77384**	**0.75904**	**51119.0082**	**0.49384**	**110.3905**	**0.83044**	**0.87824**	**5401.7461**	**0.25624**	**53.5632**
KEMERER	1-20-20-20-1				0.71321	0.76786	43422.5823	0.5087	122.6271	1.0921	0.87876	9341.6689	0.52135	83.9711
KEMERER	2-3-3-3-1	LOC (2)	EFF(2)	8-3-3	0.36822	0.92603	12691.0471	0.49053	78.6495	1.2623	0.98595	12480.547	0.61187	100.012
KEMERER	2-6-6-6-1				0.31702	0.94635	9406.7915	0.50144	72.4681	1.2969	0.96893	13175.395	0.66582	105.0648
KEMERER	2-9-9-9-1				0.31181	0.94752	9100.4992	0.4963	70.9275	1.0252	0.98366	8231.9953	0.46137	79.1473
KEMERER	2-15-15-15-1				0.88093	0.45347	78650.8967	1.6663	171.7917	1.1028	0.9987	9525.5016	0.52898	86.9047
KEMERER	2-20-20-20-1				0.26115	0.96294	6383.4511	0.50167	62.7729	1.3173	0.92139	13591.4555	0.6246	103.7553
KEMERER	2-3-3-3-1	LOC(1), EFF(1)	EFF (2)	8-3-3	0.88202	0.39475	72817.3399	1.6731	163.9089	0.89737	0.20148	6307.6879	0.69179	56.3686
KEMERER	2-6-6-6-1				0.91471	0.37935	78315.1689	1.6095	169.564	0.96104	0.9044	7234.4002	0.7912	71.2884
KEMERER	2-9-9-9-1				0.90548	0.39903	76743.2791	1.5016	162.5963	0.83978	0.27512	5524.0584	0.64459	56.8845
KEMERER	2-15-15-15-1				0.925	0.3639	80087.1143	1.2762	153.4792	0.815	0.30725	5202.779	0.61411	58.2357
KEMERER	2-20-20-20-1				0.93029	0.39949	81005.7453	1.1378	148.0115	0.84594	0.2188	5605.3459	0.57787	72.604
KEMERER	4-3-3-3-1	LOC(2), EFF(2)	EFF(3)	7-3-3	0.86987	0.44735	76688.3216	1.6086	159.1268	0.9164	0.17942	6577.9683	0.71738	59.9793

Πίνακας 4.3.2: Αποτελέσματα Συνόλου Δεδομένων KEMERER

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
KEMERER	4-6-6-6-1				0.86603	0.45436	76012.9002	1.5065	153.903	0.87332	0.049942	5974.0991	0.67931	66.1835
KEMERER	4-9-9-9-1				0.88093	0.45347	78650.8967	1.6663	171.7917	1.1735	0.98441	10786.0849	0.94831	83.4485
KEMERER	4-15-15-15-1				0.88268	0.44126	78963.102	1.4018	153.5036	0.80912	0.14733	5127.9588	0.603	64.0431
KEMERER	**4-20-20-20-1**				**0.86013**	**0.5222**	**74980.155**	**1.4207**	**156.0341**	**0.74941**	**0.66681**	**4399.0665**	**0.5694**	**43.8099**
KEMERER	3-3-3-3-1	LOC(2), EFF(1)	EFF (2)	8-3-3	0.31683	0.95274	9395.8511	0.485	73.1016	1.1914	0.97231	11118.3308	0.58701	95.1163
KEMERER	3-6-6-6-1				0.33937	0.93812	10780.225	0.52965	76.28	1.0506	0.9971	8645.6996	0.49567	82.3492
KEMERER	3-9-9-9-1				0.30563	0.94979	8743.077	0.54798	74.6545	1.2692	0.80433	12618.4463	0.61025	100.5923
KEMERER	3-15-15-15-1				0.26616	0.96185	6630.9632	0.45471	60.8768	1.3499	0.85684	14273.7761	0.67733	108.6519
KEMERER	3-20-20-20-1				0.2724	0.95965	6945.3582	0.51098	65.4451	1.2014	0.88457	11305.203	0.57817	95.2862
KEMERER	**5-3-3-3-1**	**LOC(3), EFF(2)**	**EFF(3)**	**7-3-3**	**0.58434**	**0.89168**	**34606.2374**	**1.2118**	**128.477**	**0.632**	**0.9991**	**3128.6915**	**0.50663**	**44.795**
KEMERER	**5-6-6-6-1**				**0.73439**	**0.75875**	**54660.399**	**1.3746**	**148.7307**	**0.73792**	**0.98486**	**4265.1859**	**0.5988**	**51.6993**
KEMERER	**5-9-9-9-1**				**0.67789**	**0.8142**	**46573.908**	**1.0555**	**122.352**	**0.58642**	**0.93271**	**2693.6594**	**0.47352**	**49.7018**
KEMERER	**5-15-15-15-1**				**0.48461**	**0.90991**	**23801.3446**	**0.88118**	**105.256**	**0.47411**	**0.987**	**1760.6478**	**0.27859**	**39.0628**
KEMERER	**5-20-20-20-1**				**0.38442**	**0.93928**	**14977.4907**	**0.75894**	**86.8048**	**0.5027**	**0.98801**	**1979.4266**	**0.23201**	**39.0124**

Παρατηρήσεις:

Από τα αποτελέσματα των πειραμάτων με το σύνολο δεδομένων KEMERER φαίνονται δέκα πολύ επιτυχημένα αποτελέσματα, με ιδιαίτερα χαμηλές τιμές στο MRE σφάλμα. Τα καλύτερα αποτελέσματα που πετυχαίνονται στο σύνολο είναι τα εξής: Normalized RMSE = 0,47411 το σφάλμα, CC = 0,9991 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 23,20% η απόκλιση ή αλλιώς 76,80% η σύγκλιση. Ενώ το καλύτερο πείραμα εκφράστηκε στην τοπολογία δικτύου 5-20-20-1 με Normalized RMSE = 0,5027 το σφάλμα, CC = 0,98801 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 23,20% η απόκλιση ή αλλιώς 76,80% η επιτυχία.

4.4.4.3 Χρήση Σύνολου Δεδομένων COCOMO&KEMERER

- Με χρήση μιας μετρικής LOC(i) για την πρόβλεψη της μετρικής EFF(i).

Στον πρώτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ την LOC(i) τιμή για να προβλέψω το EFF(i). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 15 και είναι από το KEMERER data set.

Στον δεύτερο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 14 και είναι από το KEMERER data set.

Στον τρίτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 13 και είναι από το KEMERER data set.

Στον τέταρτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2), LOC(i+3) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 12 και είναι από το KEMERER data set.

Στον πέμπτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), LOC(i+1), LOC(i+2) , LOC(i+3), LOC(i+4) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 11 και είναι από το KEMERER data set.

- Με χρήση δύο μετρικών LOC(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον έκτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση

του δικτύου 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 14 και είναι από το KEMERER data set.

Στον έβδομο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 13 και είναι από το KEMERER data set.

Στον όγδοο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 12 και είναι από το KEMERER data set.

Στον ένατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων

σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 8, σε 12, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 11 και είναι από το KEMERER data set.

Στον δέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του LOC (Lines of Code), τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές LOC(i), EFF(i), LOC(i+1), EFF(i+1), LOC(i+2), EFF(i+2), LOC(i+3), EFF(i+3), LOC(i+4), EFF(i+4) για να προβλέψω το EFF(i+5). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 13 και είναι από το COCOMO data set και για δοκιμή-παραγωγή είναι 10 και είναι από το KEMERER data set.

Τα καλύτερα αποτελέσματα που είχαν οι σειρές των πειραμάτων που διεξήχθησαν φαίνονται με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων που ακολουθούν. Τα αποτελέσματα αυτά συλλέχτηκαν και παρουσιάζονται στον πίνακα 4.7 μαζί με τα συμπεράσματα και τις παρατηρήσεις.

Πίνακας 4.4.1: Αποτελέσματα Συνόλου Δεδομένων COCOMO&KEMERER

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO& KEMERER	1-3-3-3-1	LOC (1)	EFF(1)	50-13-15	0.66224	0.75701	1433180.0018	2.311	524.8111	11.4803	0.73826	9771877.5407	12.8027	2278.6926
COCOMO& KEMERER	1-6-6-6-1				0.6678	0.74327	1457345.2256	2.2071	494.7607	9.6374	0.74912	6886302.0972	10.5256	1886.3628
COCOMO& KEMERER	1-9-9-9-1				0.66859	0.75171	1460796.6238	2.6781	535.5464	10.9395	0.73685	8872788.242	12.6721	2206.3693
COCOMO& KEMERER	1-15-15-15-1				0.67882	0.75674	1505866.9623	2.5624	553.1039	12.5765	0.7333	11727060.8251	14.2983	2521.7387
COCOMO& KEMERER	1-20-20-20-1				0.68593	0.75086	1537546.9381	2.3854	560.9031	11.8293	0.7289	10375057.0759	14.0734	2422.5255
COCOMO& KEMERER	2-3-3-3-1	LOC (2)	EFF(1)	50-13-14	0.2784	0.95985	255008.857	3.6493	250.8773	12.3718	0.91707	11348402.9599	7.5714	1844.6624
COCOMO& KEMERER	2-6-6-6-1				0.22453	0.97468	165876.7692	2.3383	202.9069	11.1694	0.90506	9249646.4408	7.4371	1620.7915
COCOMO& KEMERER	2-9-9-9-1				0.33574	0.94482	370884.6974	2.8704	242.8704	11.5867	0.95173	9953709.1671	7.9673	1781.3539
COCOMO& KEMERER	2-15-15-15-1				0.21927	0.97558	158193.9349	2.7986	193.7063	11.1719	0.90364	9253823.1766	7.7918	1649.4637
COCOMO& KEMERER	2-20-20-20-1				0.2115	0.97734	147182.3713	2.6006	193.9809	11.1906	0.90757	9284799.4088	7.7178	1664.3854
COCOMO& KEMERER	3-3-3-3-1	LOC (3)	EFF(1)	50-13-13	0.12359	0.99227	50856.4242	3.8251	157.6072	5.807	0.073503	2500158.816	7.5325	764.4829
COCOMO& KEMERER	3-6-6-6-1				0.10974	0.99412	40091.3105	2.7559	137.4735	1.7926	-0.21453	238253.9857	2.6456	367.6366
COCOMO& KEMERER	3-9-9-9-1				0.16407	0.98968	89616.6714	2.8683	173.5332	2.0413	-0.078814	308950.3836	4.1482	440.3336
COCOMO& KEMERER	3-15-15-15-1				0.1468	0.99142	71747.8135	2.8223	178.1249	1.8537	-0.33031	254761.0426	2.8846	374.2467
COCOMO& KEMERER	3-20-20-20-1				0.17321	0.98876	99890.2725	2.5921	186.0062	1.8397	0.21451	250933.2117	3.1174	364.7082
COCOMO& KEMERER	4-3-3-3-1	LOC (4)	EFF(1)	50-13-12	0.12666	0.99182	54255.8983	2.873	162.0012	4.4102	0.21972	1442062.5197	6.038	830.7567
COCOMO& KEMERER	4-6-6-6-1				0.84108	0.6811	2392533.9829	7.4718	541.4564	2.6859	0.60361	534877.0558	5.7036	632.1911
COCOMO& KEMERER	4-9-9-9-1				0.82908	0.65441	2324803.5491	4.5598	479.7233	3.0283	0.29553	679927.7285	6.8925	714.3941
COCOMO& KEMERER	4-15-15-15-1				0.14013	0.99138	66411.9155	3.2553	171.2213	2.3702	0.19024	416528.5471	4.074	367.7487
COCOMO& KEMERER	4-20-20-20-1				0.15644	0.99036	82775.1805	2.7981	177.9079	2.5404	0.13822	478482.3681	4.4689	416.7856

Πίνακας 4.4.2: Αποτελέσματα Συνόλου Δεδομένων COCOMO&KEMERER

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO& KEMERER	5-3-3-3-1	LOC (5)	EFF(1)	50-13-11	0.87054	0.69103	2604360.1379	12.9992	700.8064	2.5633	0.64655	487140.1757	7.0507	664.6866
COCOMO& KEMERER	5-6-6-6-1				0.10629	0.99436	38821.986	3.4784	141.4264	1.8608	0.61838	256723.6068	3.9015	427.0603
COCOMO& KEMERER	5-9-9-9-1				0.96493	0.40556	3199777.849	12.8337	743.5235	1.7441	0.090773	225528.7169	5.3029	462.7772
COCOMO& KEMERER	5-15-15-15-1				0.10029	0.99505	34563.6135	2.5847	127.1374	1.4904	0.3714	164698.5851	4.2836	304.71
COCOMO& KEMERER	5-20-20-20-1				0.10998	0.99443	41566.7856	1.2334	126.656	1.3311	0.36278	131374.0119	4.1325	283.3899
COCOMO& KEMERER	2-3-3-3-1	LOC(1), EFF(1)	EFF(2)	50-13-14	0.79804	0.60577	2095450.7969	5.6115	413.6657	0.96815	0.048577	69495.1678	1.3613	155.5727
COCOMO& KEMERER	2-6-6-6-1				0.80045	0.60741	2108102.8058	4.4419	396.3454	1.0778	0.30747	86130.4604	1.0278	178.6377
COCOMO& KEMERER	2-9-9-9-1				0.80148	0.6076	2113535.4802	4.1802	394.8194	1.0986	0.27026	89480.8633	1.0508	185.3912
COCOMO& KEMERER	2-15-15-15-1				0.80312	0.6085	2122202.3238	4.2366	409.006	1.1352	0.4387	95548.8397	0.95913	191.7556
COCOMO& KEMERER	2-20-20-20-1				0.80297	0.60898	2121383.1153	3.973	400.1767	1.1661	0.012603	100820.0558	0.9149	197.9907
COCOMO& KEMERER	4-3-3-3-1	LOC(2), EFF(2)	EFF(3)	50-13-13	0.79591	0.61046	2109029.9799	5.0517	394.88	1.1767	0.25681	102651.7219	0.76817	175.4042
COCOMO& KEMERER	4-6-6-6-1				0.79026	0.61696	2079173.9585	5.461	378.6303	1.5588	0.16984	180165.0136	2.3068	320.0812
COCOMO& KEMERER	4-9-9-9-1				0.7917	0.61732	2086750.4146	5.6219	397.8664	1.3522	0.17136	135564.1213	1.418	225.6612
COCOMO& KEMERER	4-15-15-15-1				0.7952	0.61975	2105282.12	4.7531	405.8318	1.2129	0.15918	109078.211	1.0615	201.7199
COCOMO& KEMERER	4-20-20-20-1				0.792	0.61746	2088365.7585	4.8318	373.3116	1.7112	0.21411	217111.0128	2.2763	312.797
COCOMO& KEMERER	6-4-4-4-1	LOC(3), EFF(3)	EFF(4)	50-13-12	0.81599	0.58527	2251969.131	5.1913	435.5638	1.5368	0.23059	175096.0183	3.6475	355.5274
COCOMO& KEMERER	6-6-6-6-1				0.78638	0.61855	2091497.3369	6.1585	392.6322	1.193	0.0084285	105515.5503	1.416	217.0671

Πίνακας 4.4.3: Αποτελέσματα Συνόλου Δεδομένων COCOMO&KEMERER

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
COCOMO& KEMERER	6-9-9-9-1				0.78709	0.61856	2095276.6758	6.0481	393.5359	1.0863	0.19855	87485.7196	1.2321	199.0333
COCOMO& KEMERER	6-15-15-15-1				0.78829	0.61797	2101626.5414	5.9281	393.3544	1.1028	0.65492	90166.5638	0.93971	195.5541
COCOMO& KEMERER	6-20-20-20-1				0.78966	0.61852	2108972.0216	5.2814	386.5285	1.1939	0.13079	105679.0029	1.2746	218.3975
COCOMO& KEMERER	8-4-4-4-1	LOC(4), EFF(4)	EFF(5)	50-13-11	0.79064	0.61361	2148239.7596	5.391	392.2304	1.1461	0.082886	97391.969	0.82979	186.5051
COCOMO& KEMERER	8-8-8-8-1				0.921	0.40686	2915037.9608	7.2862	615.2216	1.2858	0.33553	122578.7718	3.2434	285.9022
COCOMO& KEMERER	8-12-12-12-1				0.77999	0.62607	2090762.4414	5.2059	383.9038	1.5898	0.222	187397.151	4.7362	315.9763
COCOMO& KEMERER	8-15-15-15-1				0.78418	0.62567	2113290.5709	4.2661	384.4807	1.2482	0.12401	115514.3433	1.9787	240.5429
COCOMO KEMERER	**8-20-20-20-1**				**0.78133**	**0.62643**	**2097968.6715**	**5.038**	**388.7748**	**0.94758**	**0.43199**	**66572.816**	**0.95111**	**182.8779**
COCOMO& KEMERER	10-3-3-3-1	LOC(5), EFF(5)	EFF(6)	50-13-10	0.78098	0.62406	2128106.2833	5.7801	400.9829	1.096	0.60627	89061.1326	0.91914	195.8996
COCOMO& KEMERER	10-6-6-6-1				0.787	0.61487	2161019.5743	6.6354	416.2271	1.1237	0.23442	93616.0711	0.91991	195.4352
COCOMO& KEMERER	10-9-9-9-1				0.78695	0.61459	2160779.9368	6.4676	403.7197	1.0744	0.30857	85584.664	0.93209	189.7124
COCOMO& KEMERER	10-15-15-15-1				0.78942	0.61284	2174336.3344	5.6171	396.4637	1.104	0.23967	90358.6111	1.0298	203.4697
COCOMO& KEMERER	10-20-20-20-1				0.77867	0.6265	2115530.0733	6.2358	397.1823	1.0161	0.94698	76549.652	0.91962	189.6636

Παρατηρήσεις:

Από τα αποτελέσματα των πειραμάτων με τον συνδυασμό του σύνολου δεδομένων COCOMO και KEMERER εμφανίζεται μόνο μια τιμή με ικανοποιητικά σφάλματα, αφού τα πειράματα αφορούν τιμές από διαφορετικά περιβάλλοντα. Τα καλύτερα αποτελέσματα που πετυχαίνονται σε πείραμα είναι τα εξής: Normalized RMSE = 0,94758 το σφάλμα στη τιμή, CC = 0,43199 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 95,11% η απόκλιση ή αλλιώς 4,89% η σύγκλιση. Τα αποτελέσματα δεν είναι ικανοποιητικά

4.4.4.4 Χρήση Σύνολου Δεδομένων ALBRECHT

- Με χρήση μιας μετρικής FP(i) για την πρόβλεψη της μετρικής EFF(i).

Στον πρώτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ την FP(i) τιμή για να προβλέψω το EFF(i). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 12, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

Στον δεύτερο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), FP(i+1)για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 11, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

- Με χρήση δύο μετρικών FP(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον τρίτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default

συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 11, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

Στον τέταρτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 10, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

- Με χρήση δύο μετρικών FP(i+1) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον πέμπτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 11, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

Στον έκτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i,

χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 10, για επικύρωση-δοκιμή είναι 3 και για δοκιμή-παραγωγή είναι 9.

Τα καλύτερα αποτελέσματα που είχαν οι σειρές των πειραμάτων που διεξήχθησαν φαίνονται με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων που ακολουθούν. Τα αποτελέσματα αυτά συλλέχτηκαν και παρουσιάζονται στον πίνακα 4.7 μαζί με τα συμπεράσματα και τις παρατηρήσεις.

Πίνακας 4.5.1: Αποτελέσματα Συνόλου Δεδομένων ALBRECHT

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
ALBRECHT	1-3-3-3-1	FP(1)	EFF(1)	12-3-9	0.1688	0.98493	31.2086	0.39126	3.5222	0.58403	0.96682	130.0848	2.1201	8.1766
ALBRECHT	1-6-6-6-1				0.17305	0.98463	32.8015	0.32001	3.3021	0.59984	0.96696	137.2229	1.9043	8.4645
ALBRECHT	1-9-9-9-1				0.17126	0.98481	32.1262	0.35086	3.4251	0.54868	0.96765	114.8142	1.9651	7.79
ALBRECHT	1-15-15-15-1				0.17295	0.98483	32.762	0.34456	3.4798	0.53791	0.96751	110.35	1.9099	7.6752
ALBRECHT	1-20-20-20-1				0.17734	0.98453	34.4464	0.33312	3.6092	0.51574	0.96814	101.4447	1.7798	7.4705
ALBRECHT	2-3-3-3-1	LOC (2)	EFF(2)	11-3-9	0.21662	0.97505	31.6068	0.40291	3.9348	0.57111	0.94052	124.3948	2.0602	7.9619
ALBRECHT	2-6-6-6-1				0.21847	0.97521	32.1511	0.37318	3.8333	0.51343	0.92215	100.5353	2.115	7.2374
ALBRECHT	2-9-9-9-1				0.22082	0.97498	32.8446	0.36769	3.8349	0.51909	0.91451	102.7634	2.1728	7.5416
ALBRECHT	2-15-15-15-1				0.22338	0.97519	33.613	0.36134	3.7812	0.47178	0.92471	84.8866	1.9534	7.0937
ALBRECHT	2-20-20-20-1				0.23087	0.97542	35.902	0.33547	3.7106	0.4149	0.9358	65.6535	1.6399	6.5738
ALBRECHT	2-3-3-3-1	FP(1), EFF(1)	EFF (2)	11-3-9	0.78042	0.66757	410.2554	0.7246	11.4189	0.99041	0.22003	374.1048	3.2943	14.3656
ALBRECHT	2-6-6-6-1				0.48507	0.94653	158.492	0.60591	8.3894	1.8706	0.37201	1334.4611	5.3528	21.8216
ALBRECHT	2-9-9-9-1				0.77686	0.6419	406.5269	0.7468	11.5346	1.0025	0.16167	383.3049	2.7377	14.0283
ALBRECHT	2-15-15-15-1				0.24507	0.97007	40.4552	0.49638	4.9755	1.9201	0.34552	1406.1164	5.6255	22.3649
ALBRECHT	2-20-20-20-1				0.79143	0.66762	421.9175	0.67605	11.8451	1.0362	0.17717	409.4907	2.3451	14.5336
ALBRECHT	4-3-3-3-1	FP(2), EFF(2)	EFF(3)	10-3-9	0.61793	0.80985	20.7574	0.47128	3.2078	0.90775	0.64191	314.2609	2.0363	11.6885

Πίνακας 4.5.2: Αποτελέσματα Συνόλου Δεδομένων ALBRECHT

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
ALBRECHT	4-6-6-6-1				0.93517	0.41427	47.5416	0.65019	5.1058	1.0176	0.24117	394.8902	2.8902	12.8384
ALBRECHT	4-9-9-9-1				0.92273	0.34976	46.2851	0.62131	5.1384	0.96831	0.43064	357.5939	2.5652	12.7826
ALBRECHT	4-15-15-15-1				0.86326	0.67866	40.5114	0.57135	4.6441	1.0089	0.53053	388.1705	2.4347	12.6781
ALBRECHT	4-20-20-20-1				0.95695	0.30092	49.7826	0.5794	5.0733	0.99512	0.32637	377.6713	2.6513	13.0515
ALBRECHT	3-3-3-3-1	FP(2), EFF(1)	EFF (2)	11-3-9	0.15318	0.98736	15.8053	0.30051	2.5844	0.9398	0.7547	336.847	3.3747	12.8718
ALBRECHT	3-6-6-6-1				0.16544	0.98576	18.4362	0.29002	2.6968	0.67436	0.84524	173.4401	2.6624	9.8718
ALBRECHT	3-9-9-9-1				0.15007	0.98792	15.1692	0.25919	2.4469	0.88479	0.81997	298.5674	3.1379	12.7588
ALBRECHT	3-15-15-15-1				0.15164	0.98759	15.4896	0.25826	2.4198	1.0551	0.72661	424.5305	3.6403	14.0676
ALBRECHT	3-20-20-20-1				0.15267	0.98746	15.6997	0.27651	2.5127	0.86153	0.74329	283.0767	3.3194	12.1962
ALBRECHT	5-3-3-3-1	FP(3), EFF(2)	EFF(3)	10-3-9	0.52706	0.87316	15.1016	0.37092	2.9438	0.64854	0.81747	160.4115	1.1422	9.592
ALBRECHT	5-6-6-6-1				0.92846	0.29141	46.8617	0.6402	5.0792	0.99836	0.31399	380.1319	2.979	12.5977
ALBRECHT	5-9-9-9-1				0.91883	0.41081	45.8948	0.61463	4.9951	1.0148	0.064041	392.7875	2.6626	13.1849
ALBRECHT	5-15-15-15-1				0.83323	0.81642	37.742	0.5138	4.2281	0.89443	0.8903	305.1108	2.4781	11.7737
ALBRECHT	5-20-20-20-1				0.89798	0.43568	43.8361	0.55512	4.6452	0.99108	0.51773	374.6122	2.3575	11.7785

Παρατηρήσεις

Δεν υπάρχει κανένα ικανοποιητικό αποτέλεσμα, αφού το σφάλμα MRE είναι πολύ ψηλό. Τα καλύτερα αποτελέσματα που πετυχαίνονται στο σύνολο είναι τα εξής: Normalized RMSE = 0,4149 το σφάλμα στη τιμή, CC = 0,96814 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω.

4.4.4.5 Χρήση Σύνολου Δεδομένων DESHARNAIS

- Με χρήση μιας μετρικής FP(i) για την πρόβλεψη της μετρικής EFF(i).

Στον πρώτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ την FP(i) τιμή για να προβλέψω το EFF(i). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 55, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δεύτερο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), FP(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 54, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον τρίτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), FP(i+1), FP(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 53, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον τέταρτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), FP(i+1), FP(i+2), FP(i+3) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 52, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον πέμπτο κύκλο δοκιμών, χρησιμοποίησα μόνο την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), FP(i+1), FP(i+2), FP(i+3), FP(i+4) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 51, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

- Με χρήση δύο μετρικών FP(i) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον έκτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των

έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 54, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον έβδομο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 53, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον όγδοο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 52, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον ένατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2), FP(i+3), EFF(i+3) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά

μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 8, σε 12, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 51, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2), FP(i+3), EFF(i+3), FP(i+4), EFF(i+4) για να προβλέψω το EFF(i+5). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

- Με χρήση δύο μετρικών FP(i+1) και EFF(i) για την πρόβλεψη της μετρικής EFF(i+1).

Στον ενδέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1) για να προβλέψω το EFF(i+1). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 54, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δωδέκατο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να

καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2) για να προβλέψω το EFF(i+2). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 53, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δέκατο τρίτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2), FP(i+3) για να προβλέψω το EFF(i+3). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 52, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δέκατο τέταρτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2), FP(i+3), EFF(i+3), FP(i+4) για να προβλέψω το EFF(i+4). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 4, σε 8, σε 12, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 51, για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Στον δέκατο πέμπτο κύκλο δοκιμών, χρησιμοποίησα την στήλη του FP (Function Points), τον αριθμό των λειτουργικών σημείων που περιέχονται σε έναν κώδικα προγραμματισμού και την στήλη του EFF την προσπάθεια που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού και την χρονική στιγμή i, χρησιμοποιώ τις τιμές FP(i), EFF(i), FP(i+1), EFF(i+1), FP(i+2), EFF(i+2), FP(i+3), EFF(i+3), FP(i+4), EFF(i+4), FP(i+5), για να προβλέψω το EFF(i+5). Σε κάθε πείραμα χρησιμοποιώ αρχιτεκτονική δικτύου Back Propagation with 3 Hidden Slabs having Different Activation Functions Selected, αφήνοντας κάθε φορά τις default συναρτήσεις δραστηριοποίησης αλλά μεταβάλλω τον αριθμό των νευρώνων σε κάθε πλάκα (slab) στο εσωτερικό επίπεδο από 3, σε 6, σε 9, σε 15 και σε 20. Ο αριθμός των έργων (projects) που χρησιμοποιείται για εκπαίδευση του δικτύου είναι 50 για επικύρωση-δοκιμή είναι 5 και για δοκιμή-παραγωγή είναι 21.

Τα καλύτερα αποτελέσματα που είχαν οι σειρές των πειραμάτων που διεξήχθησαν φαίνονται με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων που ακολουθούν. Τα αποτελέσματα αυτά συλλέχτηκαν και παρουσιάζονται στον πίνακα 4.7 μαζί με τα συμπεράσματα και τις παρατηρήσεις.

Πίνακας 4.6.1: Αποτελέσματα Συνόλου Δεδομένων DESHARNAIS

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
DESHARNAIS	1-3-3-3-1	FP(1)	EFF(1)	55-5-21	0.61462	0.7919	5822403.2382	0.60543	1754.5363	1.195	0.16496	46540208.5142	1.1618	4139.7913
DESHARNAIS	1-6-6-6-1				0.64325	0.77711	6377490.4414	0.64112	1886.3345	1.118	0.20852	40740651.3181	1.1246	3763.223
DESHARNAIS	1-9-9-9-1				0.62392	0.79016	5999881.848	0.63119	1802.2514	1.2104	0.19883	47752430.3537	1.1987	4173.1999
DESHARNAIS	1-15-15-15-1				0.61812	0.79141	5888871.232	0.61646	1780.0107	1.213	0.20031	47957343.227	1.1739	4151.5546
DESHARNAIS	1-20-20-20-1				0.61935	0.79279	5912250.6565	0.62351	1775.3994	1.2256	0.17823	48957540.0825	1.2094	4286.0454
DESHARNAIS	2-3-3-3-1	FP (2)	EFF(2)	54-5-21	0.67159	0.74202	7071418.0444	0.64412	1985.475	0.94695	0.33359	29226584.3234	1.0711	2962.4522
DESHARNAIS	2-6-6-6-1				0.66899	0.74764	7016855.0724	0.63872	1982.4619	0.96429	0.29193	30307121.0415	1.1607	3142.9257
DESHARNAIS	2-9-9-9-1				0.67914	0.73201	7231197.3159	0.6275	1975.6409	0.88813	0.42877	25708955.3549	1.0624	2801.2516
DESHARNAIS	2-15-15-15-1				0.6609	0.74845	6848146.8523	0.62583	1949.8835	0.99918	0.23569	32540121.6932	1.0291	3109.5815
DESHARNAIS	2-20-20-20-1				0.67026	0.74467	7043360.7552	0.66192	2001.664	0.98913	0.27976	31888398.0424	1.0433	3000.7681
DESHARNAIS	3-3-3-3-1	FP (3)	EFF(3)	53-5-21	0.62518	0.77902	6232410.5843	0.55145	1785.0652	1.0582	0.00056626	36494821.7299	1.3883	3696.9138
DESHARNAIS	3-6-6-6-1				0.62774	0.77928	6283574.0121	0.5691	1827.2283	1.0372	0.10754	35064831.3931	1.4112	3690.9125
DESHARNAIS	3-9-9-9-1				0.64022	0.76929	6535774.5687	0.61561	1869.5258	1.0984	0.12437	39321524.4599	1.6155	4032.0784
DESHARNAIS	3-15-15-15-1				0.64614	0.77033	6657356.6428	0.64067	1931.1945	1.0827	0.12061	38203833.6438	1.5267	3944.1031
DESHARNAIS	3-20-20-20-1				0.61795	0.78873	6089080.2081	0.59827	1821.8774	1.1983	0.052392	46797725.6257	1.5286	4360.1951
DESHARNAIS	4-3-3-3-1	FP (4)	EFF(4)	52-5-21	0.60078	0.81349	5744308.7891	0.60534	1828.7474	1.051	0.1346	36000472.9035	1.3374	3610.6132
DESHARNAIS	4-6-6-6-1				0.59976	0.82298	5724722.9396	0.61523	1832.0725	1.0696	0.19856	37287826.9275	1.5097	4068.9885
DESHARNAIS	4-9-9-9-1				0.58654	0.81304	5475061.828	0.57155	1786.5819	1.106	0.11395	39872649.8504	1.4867	4017.9127

Πίνακας 4.6.2: Αποτελέσματα Συνόλου Δεδομένων DESHARNAIS

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
DESHARNAIS	4-15-15-15-1				0.60613	0.81273	5847035.2648	0.63417	1890.0393	1.2405	0.054799	50156726.0904	1.9728	5112.1482
DESHARNAIS	4-20-20-20-1				0.57957	0.81984	5345831.312	0.56429	1775.343	1.2102	0.066504	47734317.4107	1.8912	4864.0403
DESHARNAIS	5-3-3-3-1	FP (5)	EFF(5)	51-5-21	0.39202	0.92029	2486001.7052	0.42793	1107.9602	1.0498	0.28156	35918467.9233	1.4538	3541.1811
DESHARNAIS	5-6-6-6-1				0.34234	0.94141	1895874.5029	0.41444	999.8655	1.3563	0.11521	59955849.017	2.4511	5743.4994
DESHARNAIS	5-9-9-9-1				0.33026	0.94324	1764428.7059	0.38757	962.5953	1.2434	0.23817	50393275.9655	2.2384	4725.7544
DESHARNAIS	5-15-15-15-1				0.41701	0.90881	2813133.5145	0.43355	1186.8047	1.0596	0.36237	36594059.4445	1.839	4086.9048
DESHARNAIS	5-20-20-20-1				0.36201	0.9333	2119995.6842	0.42433	1060.509	1.2359	0.1797	49783290.0537	1.9537	4830.8988
DESHARNAIS	2-3-3-3-1	FP(1), EFF(1)	EFF (2)	54-5-21	0.98539	0.35177	15223323.3189	0.65896	2572.174	1.0867	0.2803	38492898.5128	1.6081	4501.4813
DESHARNAIS	2-6-6-6-1				1.0185	0.36052	16264332.5982	0.59067	2606.865	1.168	0.36713	44467066.7762	1.2946	4553.8508
DESHARNAIS	2-9-9-9-1				1.0081	0.33289	15932560.3404	0.63445	2517.0852	1.114	0.5876	40445448.8703	1.5737	4452.675
DESHARNAIS	2-15-15-15-1				1.0091	0.35071	15964297.6375	0.62421	2550.087	1.1556	0.58633	43529096.1876	1.4156	4486.9034
DESHARNAIS	2-20-20-20-1				1.0166	0.25605	16202926.6001	0.69503	2632.1122	1.0656	0.33129	37011696.6431	1.2927	4045.6851
DESHARNAIS	4-3-3-3-1	FP(2), EFF(2)	EFF(3)	53-5-21	0.69804	0.73531	7769759.9651	0.49925	1964.2981	1.2077	0.55001	47540878.3287	2.5647	5762.5354
DESHARNAIS	4-6-6-6-1				0.63499	0.79238	6429536.368	0.44734	1775.2098	1.3109	0.66712	56011194.8393	2.7503	6037.4404
DESHARNAIS	**4-9-9-9-1**				**0.35458**	**0.93454**	**2004852.7334**	**0.29805**	**830.299**	**0.24746**	**0.96996**	**1995890.8337**	**0.48074**	**828.043**
DESHARNAIS	4-15-15-15-1				0.66564	0.76702	7065178.5207	0.46853	1843.8165	1.2949	0.69767	54652880.3071	2.7902	6126.3418
DESHARNAIS	4-20-20-20-1				0.65141	0.7943	6766358.4708	0.43447	1850.2071	1.3486	0.60262	59278561.0113	2.5287	6257.985
DESHARNAIS	6-4-4-4-1	FP(3), EFF(3)	EFF(4)	52-5-21	0.58925	0.80959	5525946.8798	0.4657	1634.6599	1.1784	0.05563	45259371.2866	2.2256	5276.0862
DESHARNAIS	6-6-6-6-1				0.72504	0.70877	8366162.5006	0.49954	2001.303	1.2909	0.1983	54313157.4572	1.8095	5380.5164

Πίνακας 4.6.3: Αποτελέσματα Συνόλου Δεδομένων DESHARNAIS

					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
DESHARNAIS	**6-9-9-9-1**				**0.041726**	**0.99919**	**27708.9043**	**0.031155**	**97.5483**	**0.032424**	**0.99951**	**34266.779**	**0.051066**	**116.1206**
DESHARNAIS	6-15-15-15-1				0.74864	0.68134	8919526.0533	0.62205	2163.5419	1.1097	0.083645	40139866.1241	1.6831	4697.3804
DESHARNAIS	6-20-20-20-1				0.73512	0.70436	8600480.542	0.55489	2022.5987	1.0614	0.023494	36720543.1831	1.6904	4450.434
DESHARNAIS	8-4-4-4-1	FP(4), EFF(4)	EFF(5)	51-5-21	0.47386	0.88911	3632357.5481	0.35005	1316.8201	1.0981	0.096364	39301636.1378	2.0811	4861.0447
DESHARNAIS	8-8-8-8-1				0.45253	0.89706	3312764.5574	0.34379	1239.0603	1.2146	0.28293	48081636.1543	1.9345	5105.5347
DESHARNAIS	8-12-12-12-1				0.54628	0.85926	4827554.6168	0.39857	1483.9623	1.1612	0.30423	43951204.2109	1.8762	4615.9952
DESHARNAIS	8-15-15-15-1				0.42123	0.91961	2870354.0403	0.27854	1159.3164	1.2753	0.21493	53013451.8324	1.7204	5125.9528
DESHARNAIS	8-20-20-20-1				0.53898	0.87066	4699340.6969	0.33457	1395.444	1.1716	0.24412	44739557.0869	1.5203	4434.0961
DESHARNAIS	10-3-3-3-1	FP(5), EFF(5)	EFF(6)	50-5-21	0.5919	0.8134	5716020.8905	0.53427	1658.682	1.1298	0.051779	41601964.6317	2.0378	4948.3735
DESHARNAIS	10-6-6-6-1				0.41459	0.91788	2804466.4437	0.34882	1250.2574	1.1904	0.15619	46184833.2655	2.4058	5081.7651
DESHARNAIS	10-9-9-9-1				0.13718	0.991	307018.1472	0.10377	335.3647	1.4307	0.15512	66714463.869	1.819	5473.2847
DESHARNAIS	10-15-15-15-1				0.3675	0.93669	2203551.3951	0.30689	1113.7675	1.1967	0.15405	46677841.4495	2.0779	4982.4886
DESHARNAIS	10-20-20-20-1				0.48946	0.89807	3908816.853	0.36849	1484.5951	1.2571	0.049916	51506123.0424	2.333	5085.5647
DESHARNAIS	3-3-3-3-1	FP(2), EFF(1)	EFF (2)	54-5-21	0.65247	0.75683	6674479.2999	0.59305	1906.9515	1.0233	0.164	34130950.8666	1.258	3472.5591
DESHARNAIS	3-6-6-6-1				0.67225	0.7548	7085371.32	0.67294	2050.161	1.0165	0.21922	33680634.6137	1.3977	3362.8061
DESHARNAIS	3-9-9-9-1				0.48344	0.89885	3664261.2016	0.33652	1244.1014	1.1795	0.069319	45344846.0052	0.81518	3928.7751
DESHARNAIS	3-15-15-15-1				0.68265	0.73208	7306330.8373	0.64451	2015.7702	0.98547	0.29828	31652756.1264	1.1686	3085.6689
DESHARNAIS	3-20-20-20-1				0.68725	0.75019	7405065.1314	0.68389	2109.4375	1.0349	0.25374	34910393.2272	1.3522	3190.055
DESHARNAIS	5-3-3-3-1	FP(3), EFF(2)	EFF(3)	53-5-21	0.35726	0.93443	2035263.4429	0.24249	862.4416	1.0263	0.15197	34328752.9791	1.8126	4053.8034
DESHARNAIS	5-6-6-6-1				0.51065	0.85803	4158071.1936	0.41136	1496.1743	1.0199	0.175	33903504.862	1.4459	3558.9703

Πίνακας 4.6.4: Αποτελέσματα Συνόλου Δεδομένων DESHARNAIS

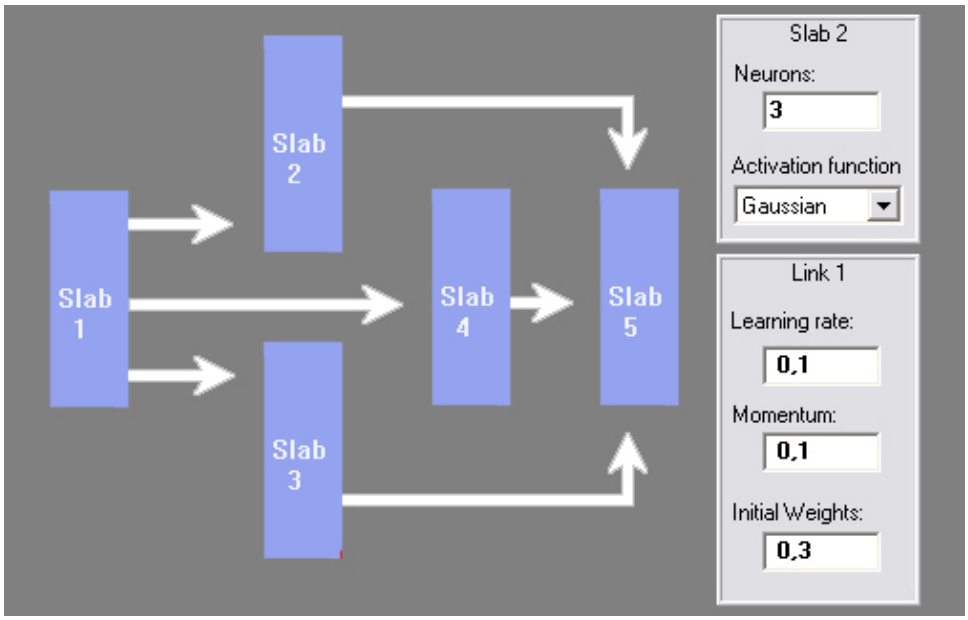
					TRAINING					TESTING				
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**MRE error**	**MAE error**
DESHARNAIS	5-9-9-9-1				0.34826	0.9413	1933923.4652	0.22368	818.0813	0.78302	0.62706	19983530.9891	1.3374	3420.4781
DESHARNAIS	5-15-15-15-1				0.51546	0.855	4236734.2596	0.41307	1488.3348	1.0658	0.18643	37023946.446	1.5239	3640.9124
DESHARNAIS	5-20-20-20-1				0.33872	0.94538	1829521.4746	0.23199	828.8804	1.0756	0.12383	37708607.7621	1.3924	3977.9458
DESHARNAIS	7-3-3-3-1	FP(4), EFF(3)	EFF(4)	52-5-21	0.55269	0.83985	4861386.4183	0.41708	1576.4909	0.71905	0.68873	16851621.8759	1.0208	2739.6547
DESHARNAIS	7-6-6-6-1				0.51991	0.86941	4301813.6909	0.32682	1421.829	0.72379	0.69364	17074732.375	1.1955	3049.1936
DESHARNAIS	7-9-9-9-1				0.56448	0.84422	5071022.3938	0.35436	1562.353	0.79132	0.58762	20409695.2842	1.056	2737.4646
DESHARNAIS	7-15-15-15-1				0.51277	0.8702	4184454.3831	0.32385	1385.5746	0.85235	0.53163	23678983.442	1.1799	3248.4067
DESHARNAIS	7-20-20-20-1				0.4499	0.89357	3221298.2603	0.34028	1282.6266	1.0015	0.4163	32690055.1137	1.5996	4125.2081
DESHARNAIS	9-4-4-4-1	FP(5), EFF(4)	EFF(5)	51-5-21	0.54072	0.85755	4729746.0222	0.42251	1467.5851	0.75815	0.64803	18734351.6981	1.3611	3137.2312
DESHARNAIS	9-8-8-8-1				0.3941	0.92005	2512527.979	0.38427	1169.9235	0.86222	0.56296	24230728.6584	1.6203	3665.8564
DESHARNAIS	9-12-12-12-1				0.37272	0.92916	2247215.796	0.33418	1053.7469	0.93379	0.48834	28419906.5854	1.6377	3903.012
DESHARNAIS	9-15-15-15-1				0.35181	0.93848	2002231.1456	0.32231	1037.1567	1.009	0.37643	33181192.3052	1.5128	3945.8444
DESHARNAIS	9-20-20-20-1				0.44704	0.90947	3232805.9898	0.35042	1251.8933	0.81434	0.57327	21614155.9205	1.2827	3094.852
DESHARNAIS	11-3-3-3-1	FP(6), EFF(5)	EFF(6)	50-5-21	0.28598	0.96222	1334384.1207	0.23311	823.7715	0.85625	0.57219	23896107.1074	1.4474	3572.1615
DESHARNAIS	11-6-6-6-1				0.25655	0.96903	1073871.3853	0.20809	713.8299	0.9256	0.67331	27923892.3805	1.5038	4199.2261
DESHARNAIS	11-9-9-9-1				0.17265	0.98613	486327.9303	0.11737	450.7839	0.94093	0.53116	28856675.8262	1.3967	4238.2669
DESHARNAIS	11-15-15-15-1				0.18855	0.98463	580036.9468	0.14324	503.4091	0.74587	0.66864	18132424.9887	1.4919	3437.6106
DESHARNAIS	11-20-20-20-1				0.22422	0.97915	820252.8492	0.19211	717.8415	0.98529	0.36894	31641181.7824	2.0641	4308.9311

Παρατηρήσεις

Εμφανίζονται δύο πολύ καλές τιμές στα σφάλματα, όπου εμφανίζεται το δίκτυο να προβλέπει ορθά και ικανοποιητικά, με τις γραφικές παραστάσεις της πραγματικής και της προβλεπόμενης προσπάθειας, όπως παρουσιάζονται στην συνέχεια να είναι πανομοιότυπες.
Τα καλύτερα αποτελέσματα που πετυχαίνονται στο σύνολο είναι τα εξής: Normalized RMSE = 0,24746 το σφάλμα στη τιμή, CC = 0,99951 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 0,5% η απόκλιση ή αλλιώς 99,5% η σύγκλιση των τιμών της πρόβλεψης της προσπάθειας σε σχέση με την πραγματική προσπάθεια, που είναι σημαντικά βελτιωμένα και ελπιδοφόρα αποτελέσματα.

Πιο συγκεκριμένα, τα δύο πολύ επιτυχημένα πειράματα παρουσιάστηκαν με την επιλογή της τοπολογίας του δικτύου 4-9-9-9-1 με τα εξής αποτελέσματα Normalized RMSE = 0,24746 το σφάλμα στη τιμή, CC = 0,96996 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 4,80% η απόκλιση ή αλλιώς 95,20% η σύγκλιση της επιτυχίας που είναι πολύ σημαντική επιτυχία και την συνέχεια με τοπολογία δικτύου 6-9-9-9-1 με τα εξής αποτελέσματα: Normalized RMSE = 0,032424 το σφάλμα στη τιμή που είναι πραγματικά πολύ μειωμένο, CC = 0,99951 η πιστότητα στις τάση προς τα κάτω ή προς τα πάνω των τιμών, MRE = 0,051066% η απόκλιση ή αλλιώς 99,49% η σύγκλιση της επιτυχίας των τιμών που είναι μια μεγάλης ακρίβειας πρόβλεψη.

Στους πίνακες 4.2.1 έως 4.6.4, φαίνονται τα αποτελέσματα από τα πειράματα που διεξήχθησαν με τα πέντε σύνολα δεδομένων. Οι τιμές της πραγματικής προσπάθειας και της προβλεπόμενης προσπάθειας που εξάγουν τα πειράματα, καθώς και οι γραφικές τους παραστάσεις και οι τιμές των λαθών Normalized RMSE, Correlation Coefficient, Mean Square Error, Relative Mean Absolute Error και Mean Absolute Error, κατά την φάση της εκπαίδευσης και της δοκιμής για τα σύνολα δεδομένων εμφανίζονται με λεπτομέρεια στο παράρτημα Α. Η αρχιτεκτονική του Τεχνητού Νευρωνικού Δικτύου αντιπροσωπεύεται στην εικόνα 4.1.



Εικόνα 4.1: Αρχιτεκτονική Νευρωνικού Δικτύου

Οι συναρτήσεις δραστηριοποίησης που χρησιμοποιήθηκαν είναι οι εξής:

Slab 1: linear [-1,1]

Slab 2: Gaussian

Slab 3: tanh

Slab 4: Gaussian comp.

Slab 5: logistic

Ενώ και οι υπόλοιποι παράμετροι ορίζονται σαν:

Ρυθμός Μάθησης : Learning Rate = 0, 1

Ορμή : Momentum = 0, 1

Αρχικά Βάρη : Initial Weights = 0, 3.

Επίσης, το κριτήριο λήξης εκπαίδευσης του δικτύου είναι οι 10,000 επαναλήψεις.

4.4.5 Ενδεικτικά Αποτελέσματα

Σκοπός αυτού του μέρους είναι η παρουσίαση των ενδεικτικών πιο βέλτιστων αποτελεσμάτων που εμφανίστηκαν με πιο έντονο χαρακτήρα στους πίνακες των αποτελεσμάτων των συνόλων δεδομένων, που παρουσιάστηκαν στο προηγούμενο μέρος. Ακόμα, θα παρουσιαστούν επιλεκτικά μερικές γραφικές παραστάσεις, μερικά σχόλια και συμπεράσματα για τα αποτελέσματα αυτά. Σημειώνουμε ότι οι προσομοιώσεις των αποτελεσμάτων, καθώς και η λεπτομερής ανάλυση τους, υπάρχουν σε ηλεκτρονική μορφή, ενώ οι λεπτομέρειες όσον αφορά τα ενδεικτικά και πιο βέλτιστα αποτελέσματα βρίσκονται στο Παράρτημα Α.

Γενικά οι παρατηρήσεις και τα συμπεράσματα, αυτού του τομέα της μελέτης δεν αναφέρονται σε ολόκληρο το σύνολο των πειραμάτων, αλλά μόνο στα ενδεικτικά βέλτιστα αποτελέσματα που εμφανίζονται στον πίνακα 4.7. Τα κριτήρια αξιολόγησης που χρησιμοποιήθηκαν είναι μόνο η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (Correlation Coefficient) και Μέσο Σχετικό Σφάλμα (MRE), καθώς ο συνδυασμός αυτών των κριτηρίων αξιολόγησης διασφαλίζει ότι η αξιολόγηση των πειραμάτων θα είναι έγκυρη και ορθή.

Πίνακας 4.7: Ενδεικτικά Αποτελέσματα πειραμάτων με το εργαλείο NeuroShell 2.

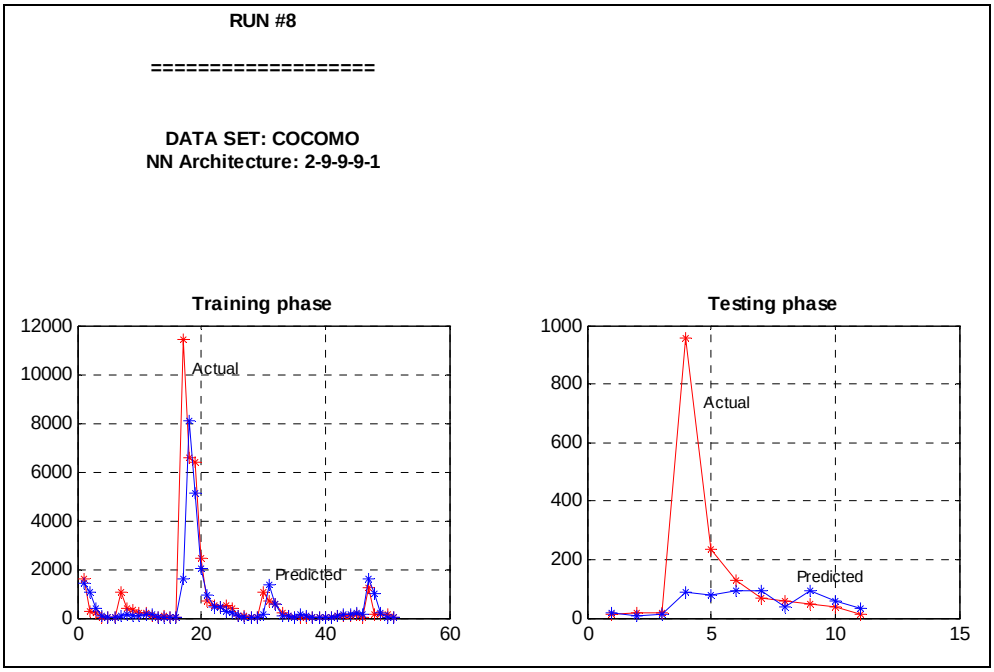
					TRAINING			**TESTING**		
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**
COCOMO	2-9-9-9-1	LOC (2)	EFF	45-5-12	0.71759	0.69648	0.87066	0.9627	0.44737	0.5254
COCOMO	5-6-6-6-1	LOC (5)	EFF	45-5-9	0.57749	0.82066	0.83338	0.95258	0.4972	0.74621
COCOMO	5-20-20-20-1	LOC (5)	EFF	45-5-9	0.58427	0.83076	1.2695	0.99617	0.63815	0.80458
COCOMO	3-15-15-15-1	LOC(2), EFF(1)	EFF (2)	46-5-10	0.71569	0.70877	0.92923	0.9522	0.40724	0.55149
COCOMO	3-20-20-20-1	LOC(2), EFF(1)	EFF (2)	46-5-10	0.72121	0.7053	0.70128	0.98873	0.4354	0.5229
COCOMO	5-15-15-15-1	LOC(3), EFF(2)	EFF (3)	45-5-10	0.75723	0.65835	1.1601	0.97982	0.19506	0.56702
KEMERER	1-3-3-3-1	LOC (1)	EFF (1)	9-3-3	0.76079	0.75026	0.9635	0.63212	0.87793	0.48763
KEMERER	1-6-6-6-1	LOC (1)	EFF (1)	9-3-3	0.82501	0.77251	0.91535	0.70659	0.87961	0.4585
KEMERER	1-9-9-9-1	LOC (1)	EFF (1)	9-3-3	0.69446	0.76166	0.53244	0.77448	0.87848	0.24051
KEMERER	1-15-15-15-1	LOC (1)	EFF (1)	9-3-3	0.77384	0.75904	0.49384	0.83044	0.87824	0.25624
KEMERER	4-20-20-20-1	LOC(2), EFF(2)	EFF (3)	7-3-3	0.86013	0.5222	1.4207	0.74941	0.66681	0.5694

					TRAINING			TESTING		
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**
KEMERER	5-3-3-3-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.58434	0.89168	1.2118	0.632	0.9991	0.50663
KEMERER	5-6-6-6-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.73439	0.75875	1.3746	0.73792	0.98486	0.5988
KEMERER	5-9-9-9-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.67789	0.8142	1.0555	0.58642	0.93271	0.47352
KEMERER	5-15-15-15-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.48461	0.90991	0.88118	0.47411	0.987	0.27859
KEMERER	5-20-20-20-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.38442	0.93928	0.75894	0.5027	0.98801	0.23201
COCOMO& KEMERER	8-20-20-20-1	LOC(4), EFF(4)	EFF(5)	50-13-11	0.78133	0.62643	5.038	0.94758	0.43199	0.95111
DESHARN AIS	4-9-9-9-1	FP(2), EFF(2)	EFF(3)	53-5-21	0.35458	0.93454	0.29805	0.24746	0.96996	0.48074
DESHARN AIS	6-9-9-9-1	FP(3), EFF(3)	EFF(4)	52-5-21	0.041726	0.99919	0.031155	0.032424	0.99951	0.051066

Συγκριτικά, παρατηρούμε ότι από τα αποτελέσματα με το σύνολο δεδομένων του COCOMO εμφανίζουν ικανοποιητικές τιμές κατά την εκπαίδευση του δικτύου, αλλά αυτό που εμφανίζει μεγαλύτερο ενδιαφέρον είναι κατά την δοκιμή του δικτύου, όπου λαμβάνουμε αρκετά ικανοποιητικές τιμές στα λάθη. Ειδικά το τρέξιμο στο οποίο χρησιμοποιήθηκε η αρχιτεκτονική του δικτύου 2-9-9-9-1 και κατά το οποίο είχαμε σαν είσοδο στο δίκτυο τιμές των γραμμών κώδικα από δύο έργα και προβλέπουμε την προσπάθεια που χρειάζεται να καταβληθεί για να διεκπεραιωθεί το δεύτερο έργο, έχουμε πολύ χαμηλές τιμές στα σφάλματα. Όπως μπορούμε να διακρίνουμε και από την γραφική παράσταση (Γραφική Παράσταση 4.1) της πραγματικής τιμής της προσπάθειας που διακρίνεται με κόκκινο χρώμα και της προβλεπόμενης προσπάθειας που διακρίνεται με μπλε χρώμα, οι 7 τιμές από τις 11 προβλέπονται με μεγάλη ακρίβεια και οι 2 τιμές προσεγγίζονται αρκετά. Ενώ υπάρχουν 2 τιμές που εμφανίζονται σαν εξαίρεση στα συνηθισμένα μεγέθη που ακολουθούν οι τιμές τις προσπάθειας, αφού ιδιαίτερα η μία εμφανίζεται να έχει πολύ μεγάλη τιμή σε σχέση με τις άλλες και όπως φαίνεται στην γραφική παράσταση σχηματίζει μια αιχμή προς τα πάνω την οποία όπως είναι φυσικό είναι πολύ δύσκολο να προβλεφτεί. Μία ακόμα πολύ ικανοποιητική πρόβλεψη αυτή στην οποία χρησιμοποιήθηκε η αρχιτεκτονική 3-15-15-15-1 και κατά την οποία εισαγάγαμε στο δίκτυο τιμές των γραμμών κώδικα από δύο έργα και μια τιμή προσπάθειας του ενός έργου, με σκοπό να προβλεφτεί η τιμή τη προσπάθειας που χρειάζεται να καταβληθεί για το δεύτερο έργο. Όπως φαίνεται και από την Γραφική Παράσταση 4.2, οι 7 τιμές από τις 10 προβλέπονται με μεγάλη ακρίβεια και 2 τιμές προσεγγίζονται αρκετά, ενώ και πάλι υπάρχει 1 τιμή που εμφανίζεται σαν εξαίρεση στα

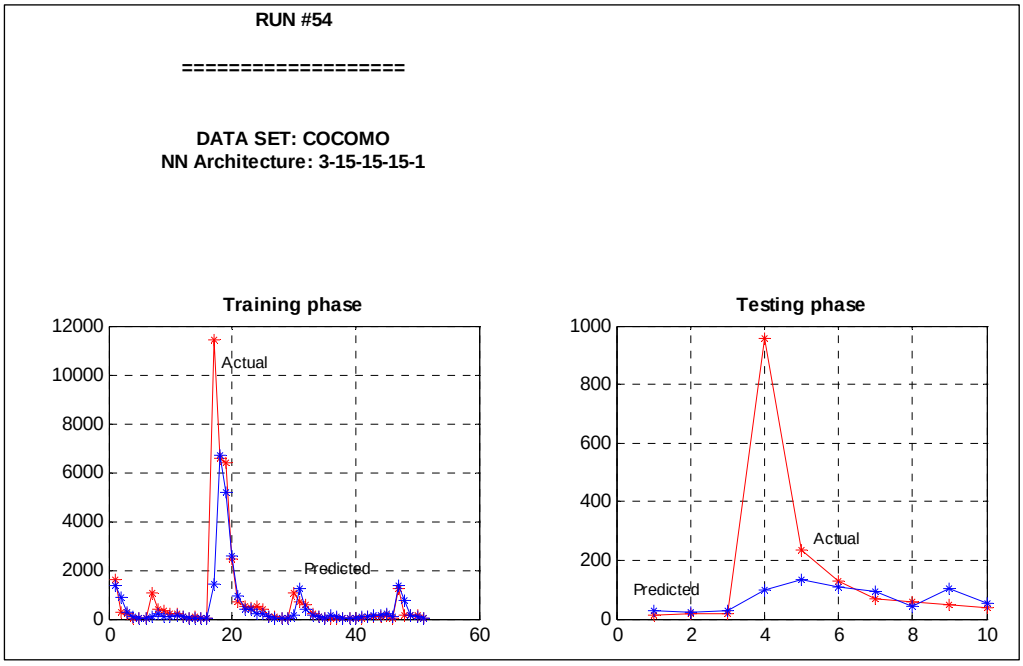
συνηθισμένα μεγέθη που ακολουθούν οι τιμές τις προσπάθειας και δεν προσεγγίζεται καθόλου.

Γραφική Παράσταση 4.1



Γενικά το σύνολο δεδομένων COCOMO παρουσιάζει αξιόλογη πρόβλεψη να πετυχαίνει ακρίβεια ποσοστιαία: με σχετικά πολύ μικρή ακρίβεια Normalized RMSE = 0,9627, με μέση σχετικά ακρίβεια τάσης CC = 0,44737 και μέτριο MRE = 52,54% απόκλιση ή αλλιώς 47,46% σύγκλιση στις τιμές.

Γραφική Παράσταση 4.2



Με την χρήση του συνόλου δεδομένων KEMERER εμφανίζονται περισσότερο ενθαρρυντικά αποτελέσματα και αριθμητικά αλλά και ποιοτικά πάρα πολύ ικανοποιητικά. Το γεγονός αυτό αντικρούει την παραδοχή που αναφέραμε προηγουμένως ότι ένα Τεχνητό Νευρωνικό Δίκτυο χρειάζεται ένα μεγάλο όγκο δεδομένων για εκπαίδευση του δικτύου για να εξαγάγει καλά αποτελέσματα αφού το σύνολο δεδομένων KEMERER είναι σχετικά πολύ μικρό, αλλά η ποιότητα των αποτελεσμάτων της δοκιμής επισκιάζονται από το γεγονός ότι χρησιμοποιούνται λίγα έργα, μονάχα τρία. Εντούτοις, ένα άλλο μικρό σε μέγεθος δηλαδή σε πλήθος έργων, που χρησιμοποιήθηκε στα πειράματα (το σύνολο δεδομένων Albrecht) δεν εμφανίζεται καθόλου στον συνοπτικό πίνακα 4.7 των βέλτιστων αποτελεσμάτων. Αυτό ίσως να οφείλεται στην ποιότητα και όχι στην ποσότητα των συνόλων δεδομένων που χρησιμοποιήθηκαν. Τα αποτελέσματα από τα πειράματα με το σύνολο δεδομένων KEMERER, αν και χρησιμοποιούνται μόνο 3 έργα για δοκιμή, είναι πάρα πολύ εντυπωσιακά, ιδιαίτερα αυτά που λήφθηκαν με αρχιτεκτονικές δικτύων 1-9-9-9-1 και 1-15-15-15-1 και χρησιμοποιήθηκε η τιμή των γραμμών κώδικα που αναπτύχθηκε για ένα έργο για να προβλεφτεί η τιμή της προσπάθειας που θα χρειαστεί να καταβληθεί για το ίδιο έργο, με το σφάλμα MRE να μειώνεται πάρα πολύ. Ακόμα πιο εντυπωσιακά είναι τα αποτελέσματα των τρεξιμάτων που έγιναν με αρχιτεκτονικές δικτύων 5-15-15-15-1 και 5-20-20-20-1 και κατά τα οποία χρησιμοποιήθηκαν οι τιμές των γραμμών κώδικα από τρία έργα και η προσπάθεια που καταβλήθηκε για τα δύο από αυτά, με σκοπό να προβλεφτεί η τιμή της προσπάθειας που θα χρειαστεί για το τρίτο έργο. Από τον πίνακα 4.7 φαίνεται ότι κατά την δοκιμή του δικτύου, τα σφάλματα παρουσιάζουν τις πιο βέλτιστες ενδείξεις. Το σφάλμα MRE και Normalized RMSE είναι ιδιαίτερα μειωμένα και αυτό ερμηνεύεται ότι όσο πιο κοντά στο μηδέν εμφανίζονται τόσο μεγαλύτερη είναι η επιτυχία στην πρόβλεψη και τόσο μικρότερη είναι η απόκλιση των πραγματικών τιμών από τις προβλεπόμενες, ενώ παρατηρούμε ότι ο συντελεστής Correlation Coefficient είναι πολύ κοντά στην μονάδα και άρα οι προβλέψεις μιμούνται την κίνηση, την τάση των πραγματικών τιμών σε κλίση.

Γενικά, το σύνολο δεδομένων KEMERER παρουσιάζει πολύ αξιόλογη πρόβλεψη, να πετυχαίνει ακρίβεια ποσοστιαία: με ικανοποιητική ακρίβεια Normalized RMSE = 0,5027, πολύ ικανοποιητικό CC = 0,98801 και πολύ χαμηλό MRE = 23,20%

η απόκλιση των τιμών, αλλιώς 76,80% σύγκλιση των πραγματικών τιμών με τις προβλεπόμενες, που είναι μια ικανοποιητική επιτυχία.

Ακόμα, από τα αποτελέσματα φαίνεται ότι στα πειράματα που έγιναν με την χρήση του συνδυασμού δύο αμιγών συνόλων δεδομένων COCOMO και KEMERER εμφανίστηκε μόνο ένα ικανοποιητικό αποτέλεσμα στον πίνακα 4.7. Το γεγονός αυτό υποστηρίζει την άποψη ότι ο συνδυασμός δεδομένων διαφορετικών για εκπαίδευση και διαφορετικών για δοκιμή του δικτύου, δίνει αναξιόπιστα αποτελέσματα αφού τα έργα που συγκρίνονται έγιναν σε διαφορετική χρονική περίοδο και σε διαφορετικά περιβάλλοντα εργασίας. Γενικά, το σύνολο δεδομένων COCOMO σε συνδυασμό με το KEMERER παρουσιάζει μέτρια πρόβλεψη να πετυχαίνει ακρίβεια ποσοστιαία τα εξής: με πολύ μικρή ακρίβεια Normalized RMSE = 0,94758, μέτριο CC = 0,43199 και μικρή ακρίβεια η απόκλιση MRE = 95,11%, ή αλλιώς 4,89% σύγκλιση, πολύ χαμηλής επιτυχίας αποτελέσματα και καθόλου ικανοποιητικά.

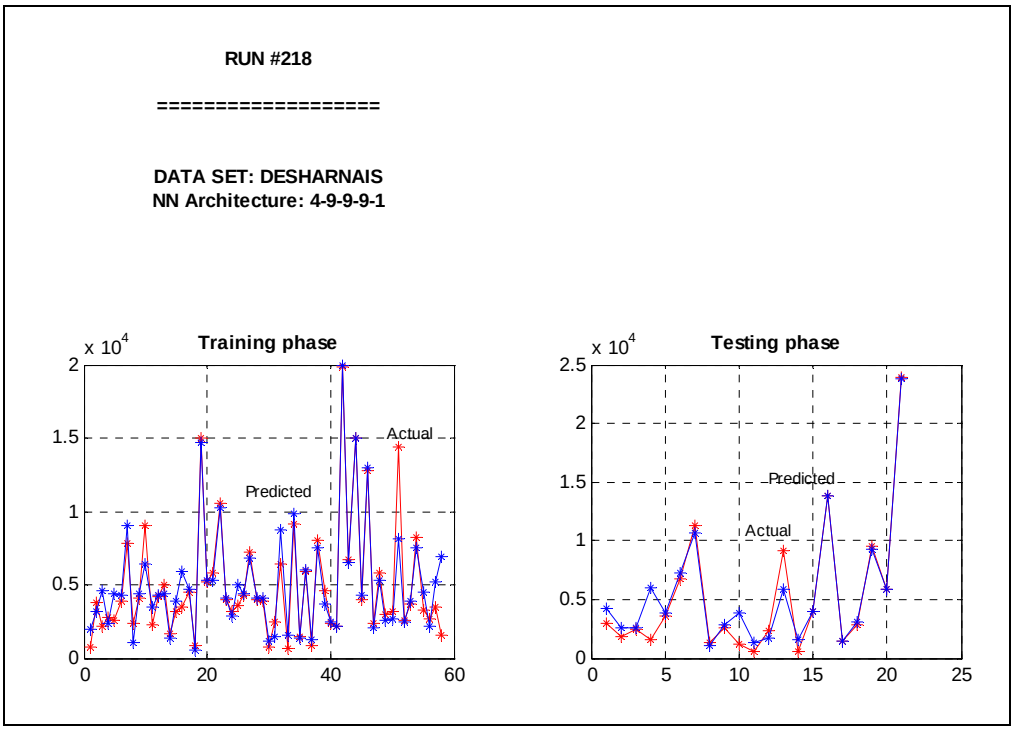
Γενικά τα αποτελέσματα από το σύνολο δεδομένων ALBRECHT δεν υπόσχονται πολλά, παρουσιάζει πρόβλεψη που πετυχαίνει πολύ μικρή ακρίβεια ποσοστιαία τα εξής: με σχετικά μέτρια ακρίβεια Normalized RMSE = 0,4149, πολύ ικανοποιητικό CC = 0,96814 που όμως απορρίπτεται λόγω του ότι το MRE εμφανίζεται πολύ ψηλό και έτσι δεν συμπεριλαμβάνεται καθόλου στα ενδεικτικά αποτελέσματα.

Τα αποτελέσματα από το σύνολο δεδομένων DESHARNAIS που παρουσιάζονται στον πίνακα 4.7, αν και είναι λίγα σε πληθυσμό είναι πάρα πολύ πετυχημένα σε ποιότητα. Χρησιμοποιήθηκε η 4-9-9-9-1 αρχιτεκτονική δικτύου και σαν είσοδο στο δίκτυο δόθηκαν οι τιμές από τα λειτουργικά σημεία από δύο έργα και οι τιμές τις προσπάθειας που καταβλήθηκε για αυτά, με σκοπό να προβλεφτεί η τιμή της προσπάθειας ενός τρίτου έργου. Το σφάλμα MRE και Normalized RMSE είναι πολύ μειωμένα και αυτό σημαίνει επιτυχία στην πρόβλεψη καθώς και το Correlation Coefficient είναι πολύ κοντά στην μονάδα και άρα οι προβλέψεις μιμούνται την κίνηση, την τάση των πραγματικών τιμών σε κλίση. Μεγαλύτερη ακόμα επιτυχία εμφανίζει το τελευταίο πείραμα κατά το οποίο οι τιμές των λαθών μειώνονται δραματικά με το Normalized RMSE λάθος να έχει τιμή 0.032424 και το MRE 0.051066 και ο Συντελεστής Συσχέτισης να ακουμπά σχεδόν την μονάδα. Όπως φαίνεται και από την Γραφική Παράσταση 4.3 οι τιμές της προβλεπόμενης προσπάθειας ακολουθούν τις τιμές των πραγματικών προσπαθειών που καταβάλλονται για την

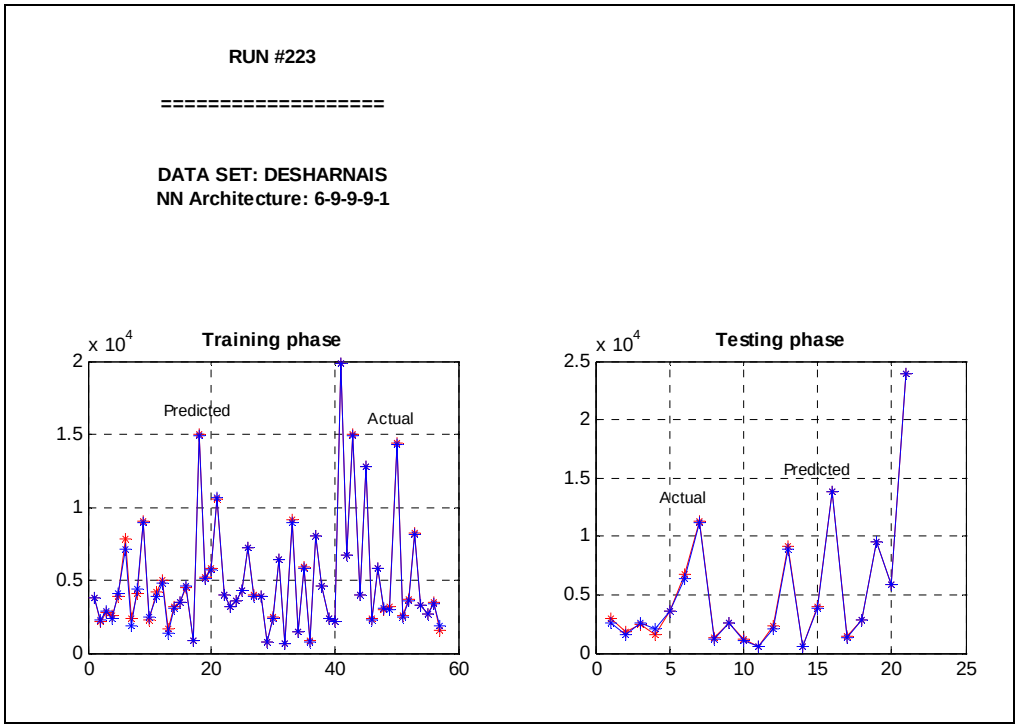
διεκπεραίωση ενός έργου ενώ στην Γραφική Παράσταση 4.4 φαίνεται καθαρά πως η πρόβλεψη ταυτίζεται απόλυτα με την πραγματική τιμή.

Γενικά το σύνολο δεδομένων DESHARNAIS παρουσιάζει πολύ αξιόλογη πρόβλεψη, με μεγάλη ακρίβεια. Πετυχαίνει ακρίβεια ποσοστιαία: με σχετικά πολύ μεγάλη σχετικά ακρίβεια Normalized RMSE = 0,032424, πολύ ικανοποιητικό CC = 0,99951 και πάρα πολύ καλό MRE = 0,5% την απόκλιση των τιμών, αλλιώς 99,5% σύγκλιση στις τιμές μια πολύ αξιόλογη και εντυπωσιακή επιτυχία. Η επιτυχία της ακρίβειας στην πρόβλεψη των τιμών μπορεί να παρατηρηθεί από τις πιο κάτω Γραφικές Παραστάσεις. Με κόκκινο χρώμα εμφανίζονται οι πραγματικές τιμές της προσπάθειας που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο ενώ με μπλε χρώμα εμφανίζεται η πρόβλεψη της προσπάθειας ανάπτυξης. Η επιτυχία της πρόβλεψης φαίνεται με την μεγάλη ταύτιση των τιμών στις γραφικές παραστάσεις.

Γραφική Παράσταση 4.3



Γραφική Παράσταση 4.4



Γενικά, τα συμπεράσματα που διεξάγονται από τα πειραματικά τρεξίματα στο εργαλείο NeuroShell 2, είναι ότι με την βοήθεια Τεχνητών Νευρωνικών Δικτύων μπορούμε να επιφέρουμε πολύ ακριβείς μετρήσεις προβλέψεων για ένα έργο για το οποίο δεν γνωρίζουμε το κόστος ανάπτυξης του. Αυτή η ακρίβεια όμως, εξαρτάται από την εκπαίδευση του δικτύου που θα προηγηθεί, η οποία εκτός του ότι συνιστάται να περιλαμβάνει στο σύνολο δεδομένων της εκπαίδευσης ένα μεγάλο πλήθος έργων, είναι σημαντική και αναγκαία η εγκυρότητα, η συνέπεια και η ποιότητα των δεδομένων.

Επίσης, τα αποτελέσματα είναι απόλυτα συνυφασμένα με την αρχιτεκτονική του δικτύου και από τα πειράματα που διεξήχθησαν φάνηκε ότι αλλαγές σε αυτήν επηρεάζουν τις προβλέψεις. Ακόμα αφού οι αλλαγές στις παραμέτρους των συναρτήσεων εκπαίδευσης, μάθησης και απόδοσης φαινομενικά παρατηρήθηκε ότι άλλαζαν σημαντικά τα αποτελέσματα, έγινε μια διαδικασία προσπάθειας βελτίωσης των αποτελεσμάτων με διάφορους συνδυασμούς των παραμέτρων αυτών. Τα αποτελέσματα όμως δεν αποφέρανε τις επιθυμητές βελτιώσεις, αφού ενώ παρατηρήθηκαν έστω μικρές μειώσεις στα σφάλματα, ακολουθούνταν από αυξήσεις σε άλλα σφάλματα μαζί με την εμφάνιση του μειονεκτήματος της αποστήθισης (over fit).

Συμπερασματικά, αποτελέσματα από τα σύνολα δεδομένων Cocomo, Kemerer, Desharnais εμφανίζονται σαν ποιοτικά καλύτερα από τα υπόλοιπα, άρα μπορούν να χαρακτηριστούν σαν ποιοτικά καλύτερα σύνολα δεδομένων. Επίσης, παρατήρησα ότι όσο αυξάνονται τα νευρώνια στα κρυμμένα επίπεδα του δικτύου τόσο πιο πολύ μειωνόντουσαν τα σφάλματα στις μετρήσεις, ενώ η επιλογή από 10 000 επαναλήψεις εκπαίδευσης του Τεχνητού Νευρωνικού Δικτύου είναι μια λογική και ικανοποιητική τιμή για να επιφέρει τα επιθυμητά αποτελέσματα. Στην συνέχεια, αφού παρατηρήθηκαν ωφελήματα από την χρήση Τεχνητών Νευρωνικών Δικτύων υλοποιήθηκε ένα εύχρηστο εργαλείο διεξαγωγής πειραμάτων με δυναμικές τις επιλογές του χρήστη για τον σχεδιασμό και την χρήση της ίδιας τεχνικής. Στην τεχνική έγινε μια προσπάθεια βελτιστοποίησης των αποτελεσμάτων, με την επιλογή μιας βέλτιστης τοπολογίας που θα χρησιμοποιηθεί στα Τεχνητά Νευρωνικά Δίκτυα, μέσω Γενετικού Προγραμματισμού. Η περιγραφή της προσπάθειας αυτής, η υλοποίηση, η σχεδίαση, αλλά και τα πειράματα και τα συμπεράσματα παραθέτονται στην συνέχεια.

4.5 Υλοποίηση Εργαλείου Πειραμάτων

4.5.1 Σκοπός

Σκοπός του προτεινόμενου εργαλείου διεξαγωγής πειραμάτων που υλοποιήθηκε, είναι η παρουσίαση των λειτουργιών και των δυνατοτήτων της MATLAB® στον τομέα των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγορίθμων, χωρίς πλέον να μας απασχολούν οι περιορισμοί και οι δυσκολίες που υπήρχαν στην σύγκριση της απόδοσης τους, αφού έχουμε σαν γνώμονα τα πειραματικά τρεξίματα του προηγούμενου τομέα της διπλωματικής εργασίας.

Μετά την διεξαγωγή των σειρών των πειραματικών τρεξιμάτων, έχουμε λάβει σημαντικά αποτελέσματα, με τα οποία μπορούμε να συγκρίνουμε τα αποτελέσματα του Τεχνητού Νευρωνικού Δικτύου, το οποίο μπορεί να δημιουργήσει ο χρήστης με την βοήθεια του εύχρηστου γραφικού περιβάλλοντος και το οποίο μπορεί στην συνέχεια προσπαθήσει να βελτιώσει με την χρήση Γενετικών Αλγορίθμων.

Η MATLAB® είναι μια υψηλής απόδοσης γλώσσα που χρησιμοποιείται για Technical Computing και η οποία μπορεί να εκφράζει λύσεις στα προβλήματα με

μαθηματικό τρόπο. Προσφέρει δυνατότητες ορισμού ενός προβλήματος πρόβλεψης, περιγραφής του Τεχνητού Νευρωνικού Δικτύου, το οποίο θα χρησιμοποιηθεί για την επίλυση του προβλήματος, προσφέρει δυνατότητες γραφικής παρουσίασης των αποτελεσμάτων, καθώς και ανάπτυξη βοηθητικών μεθόδων για την βελτίωση των αποτελεσμάτων, όπως είναι οι Γενετικοί Αλγόριθμοι.

4.5.2 Περιγραφή του Προβλήματος

Χρησιμοποιώντας διάφορα διαθέσιμα σύνολα δεδομένων, που περιέχουν τον αριθμό των γραμμών που περιέχονται σε έναν κώδικα προγραμματισμού ή τα λειτουργικά σημεία ενός λογισμικού και τις τιμές της προσπάθειας που χρειάζεται να καταβληθεί για ένα έργο ανάπτυξης λογισμικού, προσπαθούμε να προβλέψουμε την προσπάθεια ή το κόστος ανάπτυξης λογισμικού κάποιων έργων που δεν γνωρίζει το Τεχνητό Νευρωνικό Δίκτυο. Μπορούμε να μεταβάλουμε τις παραμέτρους, όπως είναι ο αριθμός των εισόδων, των νευρώνων στο εσωτερικό επίπεδο, τον αριθμός των επαναλήψεων της εκπαίδευσης, την συνάρτηση εκπαίδευσης, την συνάρτηση μάθησης και απόδοσης με σκοπό να βελτιωθούν τα αποτελέσματα της πρόβλεψης. Ακόμα, με την χρήση Γενετικού Αλγόριθμου μπορούμε να δημιουργήσουμε διάφορες γενεές ατόμων που έχουν διαφορετική αρχιτεκτονική Τεχνητού Νευρωνικού Δικτύου και από αυτά να επιλεχτούν τα πιο αποδοτικά, έτσι ώστε σε κάθε περίπτωση να γνωρίζει ο χρήστης ποια είναι η πιο αποδοτική τοπολογία δικτύου που μπορεί να χρησιμοποιήσει.

4.5.3 Σχεδιασμός Τεχνητού Νευρωνικού Δικτύου

Όπως αναφέραμε και πιο πριν, τα Τεχνητά Νευρωνικά Δίκτυα έχουν την ικανότητα να υιοθετούν και να προσαρμόζονται σε προβλήματα πρόβλεψης. Γενικά τα Τεχνητά Νευρωνικά Δίκτυα έχουν εφαρμοστεί και έχουν μελετηθεί κάτω από μοντέλα μετρικής του κόστους ανάπτυξης λογισμικού, σε ένα ευρύ σύνολο από μελέτες. Μεταξύ των οποίων, αναφέρω μερικές Karunanithi et al.1992, Kumar et al.1994, Srinivasan & Fisher 1995 και Witting and Finnie 1994, οι οποίες ανέδειξαν την τεχνική αυτή σαν την βέλτιστη ως προς τα αποτελέσματα, σε σχέση με άλλες τεχνικές.

Η τεχνική των Τεχνητών Νευρωνικών Δικτύων παρουσιάζει διάφορα πλεονεκτήματα όσον αφορά την ικανότητα να επιλύει προβλήματα υπολογισμού

πρόβλεψης. Η ικανότητα της να αντιμετωπίζει την πολυπλοκότητα των πεδίων τιμών, την ικανότητα γενίκευσης, την ευελιξία που παρέχει αλλά και την χρήση παραλληλισμού, είναι μερικά από τα σημαντικά πλεονεκτήματα που οδήγησαν στην επιλογή της τεχνική των Τεχνητών Νευρωνικών Δικτύων για την έρευνα αυτή.

Στο σημείο αυτό της μελέτης, θα περιγραφεί η διαδικασία και το μοντέλο που αποφάσισα να ακολουθήσω και το οποίο υλοποιήθηκε στα πλαίσια της διπλωματικής αυτής εργασίας, σαν το προτεινόμενο εργαλείο διεξαγωγής πειραμάτων με σκοπό τον υπολογισμό του κόστους ή την πρόβλεψη της προσπάθειας ανάπτυξης λογισμικού.

Ένα Τεχνητό Νευρωνικό Δίκτυο μπορεί να θεωρηθεί σαν ένας κατευθυνόμενος γράφος, ο οποίος αποτελείται από κόμβους (nodes) και συνδέσμους (weights) μεταξύ των κόμβων. Οι κόμβοι αποτελούν ένα σύνολο από νευρώνες, οι οποίοι οργανώνονται σε επίπεδα, κρυμμένα και εξωτερικά, και στους οποίους εισάγονται οι πληροφορίες. Οι πληροφορίες που αναφέρονται εδώ και οι οποίες θα οργανωθούν τελικά σε τρία διαφορετικά σύνολα εισόδων (εκπαίδευσης, επικύρωσης και δοκιμής) για το Τεχνητό Νευρωνικό Δίκτυο, προέρχονται από τα διαθέσιμα σύνολα δεδομένων που αναφέραμε σε προηγούμενο τμήμα της μελέτης.

Με βάση τις πληροφορίες και τα συμπεράσματα μετά από την ανάλυση και την έρευνα που έγινε, συγκεντρώνονται από τα σύνολα δεδομένων βασικά δύο σύνολα δεδομένων τα οποία περιγράφονται στην συνέχεια. Τα διαθέσιμα σύνολα δεδομένων και τα οποία μπορεί να επιλέξει σε κάθε πείραμα ο χρήστης είναι:

- [COCOMO ’81]
- [Kemerer 1987]
- [Albrecht, Gaffney 1983]
- [Desharnais 1989]

Δεδομένης μιας χρονολογικής σειράς $x = \{x(t) : 1 \le t \le N\}$, όπου N είναι ο συνολικός αριθμός των στοιχείων, στην περίπτωσή μας, ο συνολικός αριθμός των έργων που θα ληφθούν υπόψιν και που περιέχονται στα σύνολα δεδομένων, εξάγονται τα σύνολα εισόδου στο Τεχνητό Νευρωνικό Δίκτυο.

Το σύνολο εκπαίδευσης ή εκμάθησης (training set) ορίζεται ως εξής: $x_{train} = \{x(t) : 1 \le t \le T\}$, ενώ το σύνολο ελέγχου (testing set) ως: $x_{test} = \{x(t) : T \le t \le N\}$. Προτείνεται όπως τα μεγέθη των συνόλων που θα

χρησιμοποιούνται, να είναι περίπου το 70% από το συνολικό αριθμό έργων για εκπαίδευση και το υπόλοιπο ποσοστό για δοκιμή. Υπάρχει επίσης και η δυνατότητα χρήσης συνόλου επικύρωσης του οποίο τα δεδομένα βρίσκονται στο ενδιάμεσο του συνόλου εκπαίδευσης και δοκιμής.

Η λογική που ακολουθήθηκε για την διεξαγωγή των πειραμάτων χωρίζεται σε τρεις κατηγορίες όπως αναφέρθηκαν και προηγουμένως και όπως εξηγούνται στην συνέχεια ανάλογα με τον τρόπο επιλογής των εισόδων για την πρόβλεψη των έργων.

Τα σύνολα δεδομένων συλλέγονται και κατά την διάρκεια των σειρών των επαναλήψεων ή των πειραμάτων, δημιουργείται το Τεχνητό Νευρωνικό Δίκτυο, εκπαιδεύεται και δοκιμάζεται, με την χρήση του συνόλου εκπαίδευσης και δοκιμής, αντίστοιχα προβλέποντας στο τέλος μία μόνο τιμή, την επόμενη χρονικά τιμή στην σειρά μετρική της προσπάθειας. Η διαδικασία που ακολουθείται είναι η ακριβώς η ίδια με αυτήν που περιγράφηκε στον προηγούμενο τομέα, όταν περιγράφηκαν με λεπτομέρεια τα πειράματα που διεξήχθησαν στο εργαλείο NeuroShell 2.

Σε κάθε επανάληψη του πειράματος εισάγονται στο σύστημα ένα σύνολο από επιλεγμένα δεδομένα, δηλαδή μετρικές του αριθμού των γραμμών κώδικα για κάποιο έργο λογισμικού ή τα λειτουργικά σημεία ενός συστήματος, αφού πρώτα κανονικοποιούνται (normalized or calibrated). Λαμβάνεται το αποτέλεσμα της πρόβλεψης, το οποίο συγκρίνεται με την πραγματική πρόβλεψη και στη συνέχεια, τα βάρη και οι συνδέσεις των νευρώνων (weights and biases) αναπροσαρμόζονται, έτσι ώστε να εξαλειφθεί η απόκλιση της πραγματικής τιμής της προσπάθειας από την προβλεπόμενη προσπάθεια που παράγει το Τεχνητό Νευρωνικό Δίκτυο στην έξοδο του. Αφού προσαρμοστούν όλα τα πρότυπα στο δίκτυο, η διαδικασία επαναλαμβάνεται. Το δίκτυο σταματάει την εκτέλεση είτε με το πρώτο κριτήριο: μετά από την πάροδο ενός αριθμού επαναλήψεων που μπορεί να καθορίσει ο χρήστης, είτε με το δεύτερο κριτήριο: αν κατά επιλογή εκτελείται ταυτόχρονα με την δοκιμή, η διαδικασία επικύρωσης του δικτύου, το δίκτυο σταματάει την εκπαίδευση όταν το λάθος αρχίσει να αυξάνεται (φαινόμενο early stopping). Η διαδικασία επικύρωσης προσφέρει έναν σίγουρο τρόπο βελτίωσης της γενίκευσης και της αποστήθισης του δικτύου. Για την διαδικασία της επικύρωσης (validation) χρησιμοποιείται ένα πολύ μικρό υποσύνολο των δεδομένων και κατά την διάρκεια της διαδικασίας αυτής το λάθος που παράγεται από το Τεχνητό Νευρωνικό Δίκτυο παρακολουθείται συνεχώς. Όταν το Τεχνητό Νευρωνικό Δίκτυο ξεκινήσει την φάση της εκπαίδευσης του, όπως είναι λογικό ,το

λάθος που θα υπολογίζεται από το σύνολο επικύρωσης θα μειώνεται, όπως και το λάθος που υπολογίζεται από το σύνολο εκπαίδευσης. Αν όμως το δίκτυο ξεκινήσει να αποστηθίζει, να κάνει over fit τα δεδομένα τότε το λάθος που υπολογίζεται από το σύνολο της επικύρωσης θα ξεκινήσει να αυξάνεται. Όταν αυτό το φαινόμενο συνεχιστεί για έναν αριθμό από επαναλήψεις, τότε σταματά η εκπαίδευση και επιστρέφονται οι τιμές των βαρών και των συνδέσεων στους νευρώνες (weights and biases), που εμφανίστηκαν στο χαμηλότερο λάθος.

Το λάθος της πρόβλεψης που μας ενδιαφέρει είναι αυτό που θα παρατηρηθεί από την δοκιμή του δικτύου και όχι κατά την εκπαίδευση του δικτύου. Έτσι, μετά την εκπαίδευση, και αφού έχει "μάθει" ικανοποιητικά (και δεν έχει αποστηθίσει), διεξάγεται μια διαδικασία δοκιμής ή φάση αποτίμησης (evaluation) του Τεχνητού Νευρωνικού Δικτύου. Στην φάση αυτή, παρουσιάζεται στο δίκτυο το σύνολο δοκιμής ή ελέγχου (testing set), το οποίο δεν έλαβε μέρος πριν στην διαδικασία. Έτσι τα αποτελέσματα που λαμβάνονται θα είναι από δεδομένα τα οποία δεν γνωρίζει το δίκτυο και έτσι σίγουρα δεν είναι αποτελέσματα απομνημόνευσης.

Από την διαδικασία αυτή μπορούμε να σχηματίσουμε δύο πολύ χρήσιμες γραφικές παραστάσεις της πραγματικής προσπάθειας σε σχέση με την προβλεπόμενη προσπάθεια, αρχικά της φάσης της εκπαίδευσης του δικτύου και στην συνέχεια της δοκιμής του δικτύου. Από την διαδικασία της εκπαίδευσης και της δοκιμής του δικτύου, επιστρέφουμε τις γραφικές παραστάσεις της απόδοσης του Τεχνητού Νευρωνικού Δικτύου κάθε φορά, καθώς και το σύνολο των αριθμητικών σφαλμάτων που σημειώνονται, κατά την εκπαίδευση και κατά την δοκιμή του Τεχνητού Νευρωνικού Δικτύου, τα οποία συγκεντρώνονται σε αρχεία excel. Με βάση τα σφάλματα και τις γραφικές παραστάσεις μπορούμε να διακρίνουμε αν υπάρχει επιτυχία ή όχι στην πρόβλεψη του κόστους ανάπτυξης λογισμικού.

4.5.4 Εφαρμογή Γενετικού Αλγορίθμου

Πριν παρουσιαστούν τα αποτελέσματα της εφαρμογής της τεχνικής των Τεχνητών Νευρωνικών Δικτύων στην διαδικασία της προσπάθειας υπολογισμού του κόστους της ανάπτυξης λογισμικού, θα αναφερθούμε στην δυνατότητα που παρέχεται από το εργαλείο εφαρμογής Γενετικού Αλγορίθμου. Η δυνατότητα αυτή θα υποβοηθήσει στην προσπάθεια διεξαγωγής πιο βέλτιστων αποτελεσμάτων, αφού θα

παρέχει στον χρήστη την πιο αποδοτική τοπολογία δικτύου για τον υπολογισμό του κόστους ανάπτυξης λογισμικού για το κάθε σύνολο δεδομένων.

Οι Γενετικοί Αλγόριθμοι αναφέρονται ανάμεσα σε ένα σύνολο από μεθόδων που ονομάζονται σαν Evolutionary Computation Techniques και χαρακτηρίζονται από το γεγονός ότι η λύση αναδεικνύεται μέσα από ένα σύνολο από επαναλήψεις γενεών υποψήφιων λύσεων και από τις οποίες επιλέγεται και επιβιώνει η καλύτερη.

Τα πλεονεκτήματα των Γενετικών Αλγορίθμων, σε συνδυασμό με τις παρατηρήσεις ότι υπάρχουν μεγάλες διαφοροποιήσεις στην απόδοση των Τεχνητών Νευρωνικών Δικτύων, όσο μεταβάλλονται διάφοροι εσωτερικοί παράμετροι των δικτύων, όπως είναι η αρχιτεκτονική, ο αριθμός των εισόδων κ.α., οδήγησαν στην εφαρμογή τους στο προτεινόμενο σύστημα διεξαγωγής πειραμάτων.

Η διαδικασία εφαρμογής Γενετικού Αλγόριθμου αφορά αρχικά την κωδικοποίηση των ατόμων λύσεων, με μια σειρά από bits που αντιπροσωπεύουν την αρχιτεκτονική, την τοπολογία του Τεχνητού Νευρωνικού Δικτύου (βασικά τον αριθμό νευρώνων της εισόδου και τον αριθμό των νευρώνων σε κάθε ένα εσωτερικό slab του κρυμμένου επιπέδου). Στην συνέχεια αρχικοποιείται τυχαία ένα σύνολο από άτομα, ο πληθυσμός. Για κάθε ένα άτομο αποτιμάται η επιλογή του και τι αποτελέσματα θα λαμβάναμε με την χρήση του στο Τεχνητό Νευρωνικό Δίκτυο και επιλέγεται το καταλληλότερο άτομο με βάση την απόδοση του. Στην συνέχεια γίνεται διασταύρωση και μετάλλαξη των ατόμων του πληθυσμού δημιουργώντας τον νέο πληθυσμό με νέα άτομα προς αποτίμηση. Με αυτό τον τρόπο, με πολλές επαναλήψεις και τυχαίες επιλογές ατόμων μπορούμε να πούμε ότι ο Γενετικός Αλγόριθμος ελέγχει έναν ικανοποιητικό αριθμό ατόμων με διαφορετική αρχιτεκτονική στο Νευρωνικό Δίκτυο και επιλέγεται ο καλύτερος. Έτσι λαμβάνουμε σαν αποτέλεσμα την καλύτερη πιθανή και πιο βέλτιστη με τα καλύτερα αποτελέσματα τοπολογία δικτύου, η οποία μπορεί να διεξάγει πρόβλεψη που να πλησιάζει όσο το δυνατόν περισσότερο τις πραγματικές τιμές. Σαν αποτέλεσμα παρουσιάζεται η βέλτιστη τοπολογία και σε αρχεία excel τα σφάλματα των καλύτερων ατόμων από κάθε γενεά που εμφανίζονται κατά την εκπαίδευση και κατά την δοκιμή του Τεχνητού Νευρωνικού Δικτύου. Ακόμα παρουσιάζονται στον χρήστη οι γραφικές παραστάσεις παρακολούθησης του συνολικού fitness, της συνάρτησης καταλληλότητας των ατόμων και αυτή της καταλληλότητας των βέλτιστων ατόμων όπως εμφανίζονται από γενεά σε γενεά.

4.5.5 Προτεινόμενο Εργαλείο Πειραμάτων

Στόχος αυτού του τμήματος της διπλωματικής εργασίας, είναι η περιγραφή και παρουσίαση των δυνατοτήτων του προτεινόμενου εργαλείου διεξαγωγής πειραμάτων με την χρήση Τεχνητών Νευρωνικών Δικτύων και της προσπάθειας βελτιστοποίησης της επιλογής της αρχιτεκτονικής με την χρήση Γενετικού Προγραμματισμού.

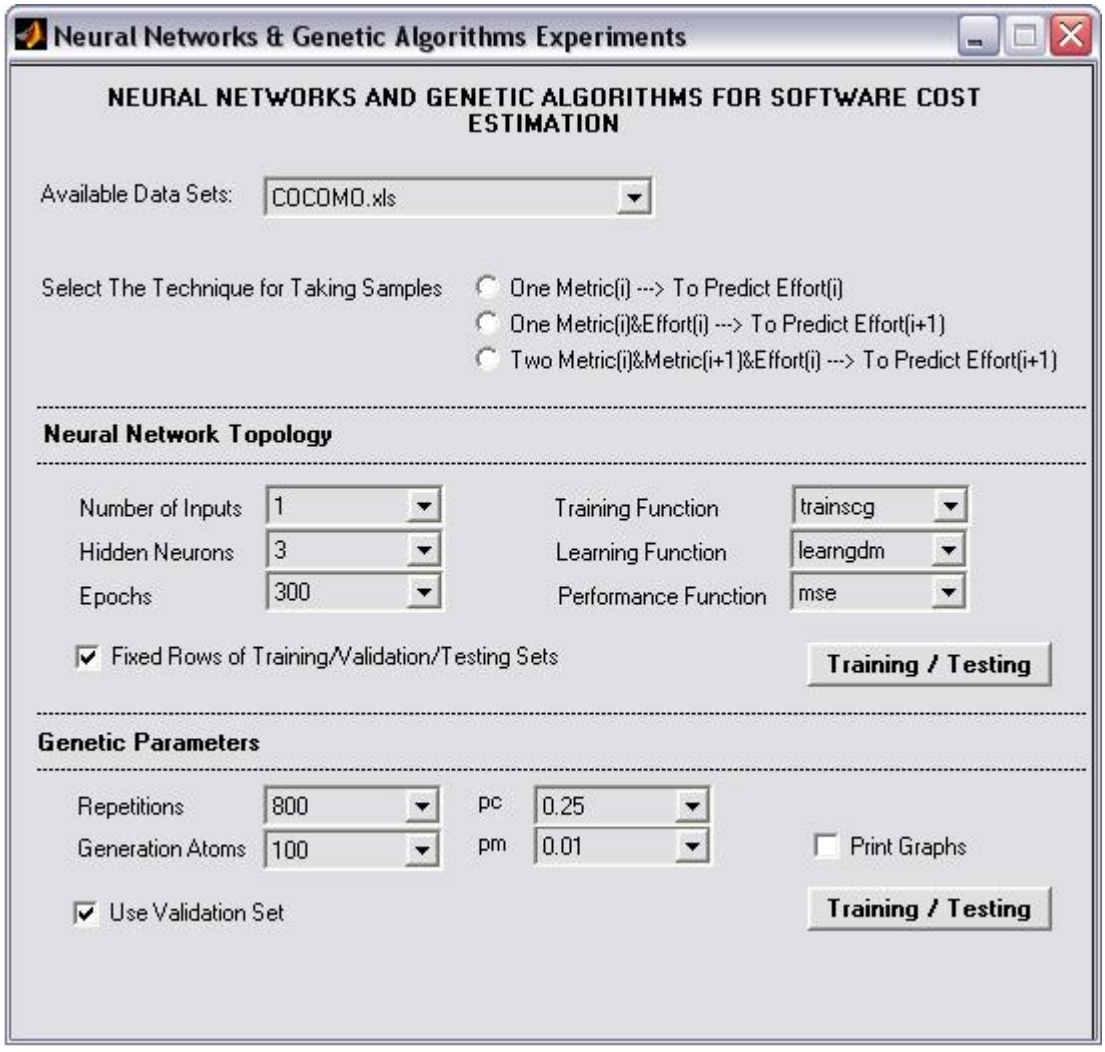
Στην εικόνα 4.2 φαίνεται η κεντρική οθόνη του υλοποιημένου εργαλείου, με το οποίο μπορεί ο χρήστης να διεξάγει πειράματα πρόβλεψης ή υπολογισμού του κόστους ανάπτυξης λογισμικού. Στην αρχή υπάρχει ένα combo box “Available Data Sets”, στο οποίο περιγράφονται τα ονόματα των διαθέσιμων συνόλων δεδομένων. Οι επιλογές των συνόλων δεδομένων φαίνονται στην εικόνα 4.3, στην οποία παρατηρούμε ότι ο χρήστης έχει επιλέξει και τεχνική με την οποία θα ληφθούν τα δείγματα των συνόλων δεδομένων. Τα αρχεία έχουν την επέκταση “.xls” και πρέπει οπωσδήποτε να περιέχουν στην πρώτη γραμμή τα ονόματα των παραγόντων που επιθυμεί ο χρήστης να συμπεριληφθούν στην έρευνα, ενώ στην πρώτη στήλη πρέπει να βρίσκονται οι τιμές του παράγοντα που επιθυμεί να προβλέψει, όπως είναι για παράδειγμα το κόστος ή η προσπάθεια που χρειάζεται να καταβληθεί για να αναπτυχθεί ένα έργο λογισμικού.

Η τεχνική επιλογής των δειγμάτων που θα εισαχθούν στο Νευρωνικό Δίκτυο, μπορεί να γίνει μέσω επιλογής ενός από τα τρία radio buttons “Select The Technique for Taking Samples” που επεξηγούνται στη συνέχεια:

1. Με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i.
2. Με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού του επόμενου στην σειρά έργου i+1
3. Με την χρήση δύο τιμών μιας μετρικής (όπως για παράδειγμα LOC, γραμμές κώδικα του λογισμικού ή FP, λειτουργικά σημεία του λογισμικού) του λογισμικού i, αλλά και του επόμενου στην σειρά έργου i+1, και της τιμής της

προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας ανάπτυξης λογισμικού έργου i+1.

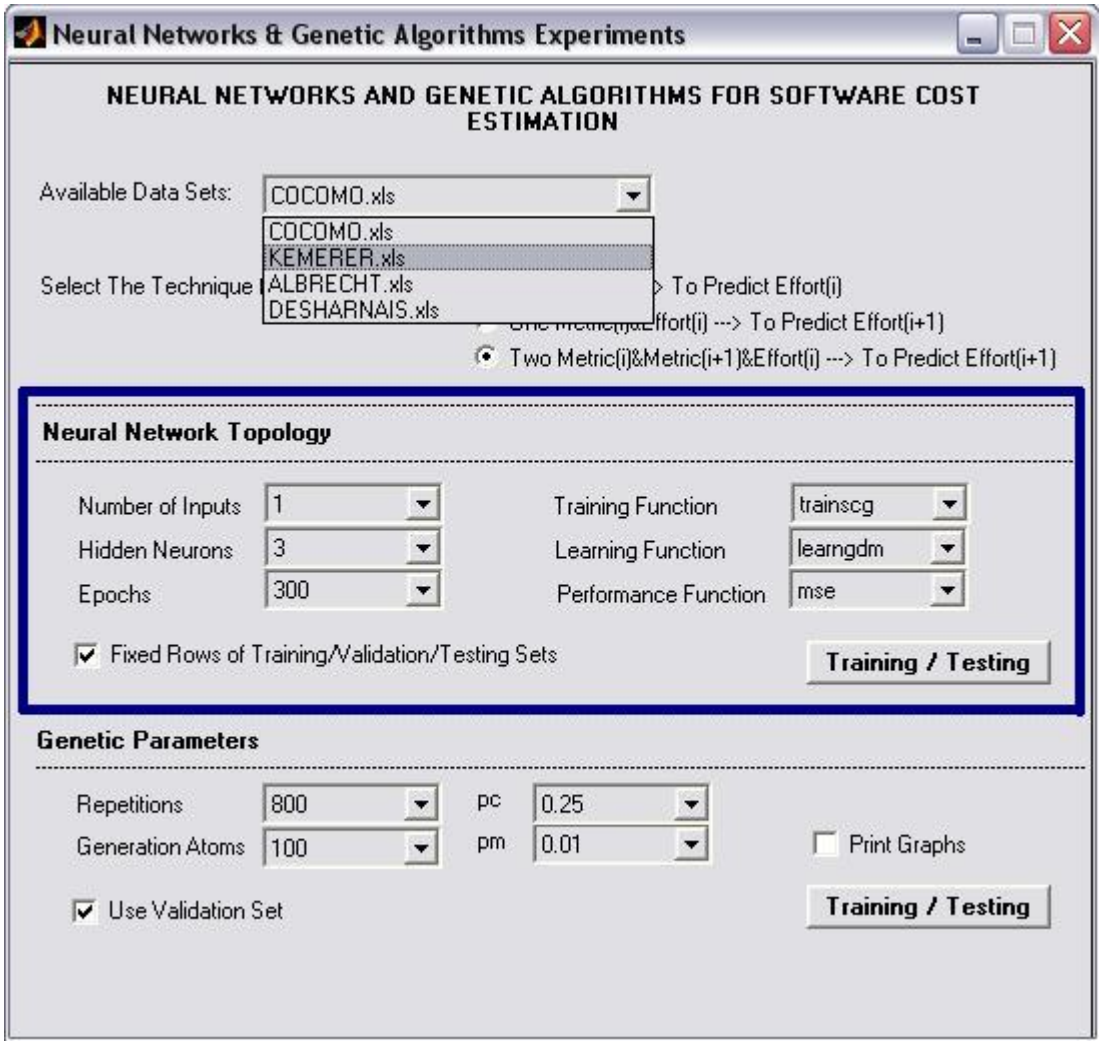
Η τιμή του i, δηλαδή ο αριθμός των έργων που θα ληφθούν υπόψιν καθορίζονται από την τιμή στο combo box "Number of Inputs". Ανάλογα με τον αριθμό αυτό, αλλά και σε συνδυασμό με την τεχνική που έχει επιλεχτεί για την εξαγωγή των δειγμάτων, υπολογίζονται οι νευρώνες που χρειάζονται στο εξωτερικό επίπεδο των εισόδων του Τεχνητού Νευρωνικού Δικτύου.



Εικόνα 4.2: Αρχική Οθόνη Εργαλείου Διεξαγωγής Πειραμάτων

Για να μπορέσει ο χρήστης να προχωρήσει στην επόμενη φάση επιλογών που αφορά είτε την εκπαίδευση και διεξαγωγή πρόβλεψης με την χρήση Τεχνητών Νευρωνικών Δικτύων (Εικόνα 4.3) είτε την εκπαίδευση δικτύων και εύρεση της βέλτιστης αρχιτεκτονικής ή τοπολογία δικτύου με την χρήση Γενετικών Αλγορίθμων

(Εικόνα 4.4), είναι απαραίτητο να επιλέξει τεχνική εξαγωγής των συνόλων δεδομένων που θα χρησιμοποιηθούν, αλλιώς εμφανίζονται τα ανάλογα μηνύματα λάθους.



Εικόνα 4.3: Αρχικές Επιλογές και Χρήση Νευρωνικών Δικτύων

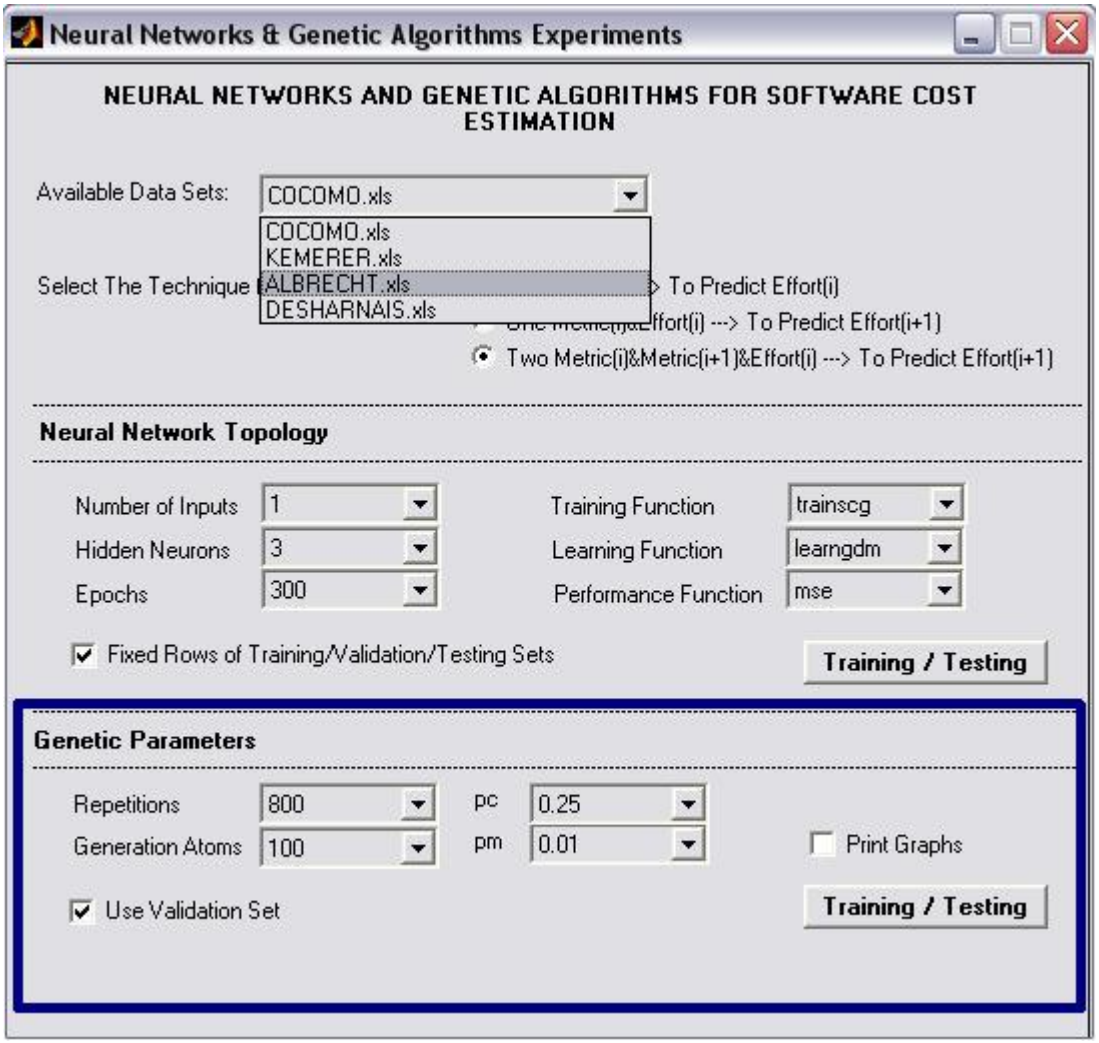
Η πρώτη μεθοδολογία διεξαγωγής πειραμάτων είναι η χρήση Τεχνητών Νευρωνικών Δικτύων. Οι επιλογές στα combo boxes που φαίνονται στην εικόνα 4.3 είναι ικανοποιητικές για την διεξαγωγή των αρχικών πειραμάτων. Το "Number of Inputs" αντιπροσωπεύει τον αριθμό των εισόδων που επιθυμούμε να έχει το Νευρωνικό Δίκτυο και στην ουσία είναι ο αριθμός των έργων που θα ληφθούν υπόψιν στην πρόβλεψη. Ανάλογα με τον αριθμό αυτό αλλά και την μέθοδο που έχει επιλεχτεί για την εξαγωγή των δεδομένων, υπολογίζονται οι νευρώνες που χρειάζονται στο εξωτερικό επίπεδο των εισόδων. Το "Hidden Neurons" αντιπροσωπεύει τον αριθμό των νευρώνων υπολογισμού που θα χρησιμοποιηθούν στα τρία εσωτερικά κρυμμένα επίπεδα του

δικτύου. Το “Epochs” αντιπροσωπεύει τον αριθμό των επαναλήψεων που θα κάνει κατά την εκπαίδευση του το Τεχνητό Νευρωνικό Δίκτυο και είναι το δεύτερο κριτήριο τερματισμού του αλγόριθμου εκπαίδευσης.

Όπως φαίνεται και στην εικόνα 4.3 όταν ξεκινάει το εργαλείο υπάρχει επιλεγμένο το check box “Fixed Rows of Training/Validation/Testing Sets”, το οποίο σημαίνει ότι αυτόματα το εργαλείο χρησιμοποιεί το 70% του συνόλου των έργων για Εκπαίδευση, το 10% του συνόλου των έργων για Επικύρωση και το 20% του συνόλου των έργων για Δοκιμή του Νευρωνικού Δικτύου. Αυτή είναι μια γενικά ικανοποιητική μέθοδος διαχωρισμού και προετοιμασίας των συνόλων, αλλά ο χρήστης πρέπει να προσέξει σε περιπτώσεις που τα διαθέσιμα έργα είναι λίγα (όπως στην περίπτωση του συνόλου δεδομένων “Kemerer.xls” και “Albrecht.xls”), να μεταβάλει ιδιοχείρως τον αριθμό των έργων που θα χρησιμοποιήσει για εκπαίδευση, επικύρωση και δοκιμή του δικτύου. Μία λύση είναι η μείωση των έργων για επικύρωση στο ελάχιστο και η αφαίρεση έργων για εκπαίδευση και η πρόσθεση αυτών στο σύνολο των έργων που θα χρησιμοποιηθούν για δοκιμή. Μπορεί να γίνει, αν ο χρήστης αφαιρέσει την επιλογή στο check box “Fixed Rows of Training/Validation/Testing Sets”. Εάν δεν γίνει κάτι τέτοιο, τότε πολύ πιθανόν τα στοιχεία που έχουν καθοριστεί αυτόματα, ποσοστιαία από το εργαλείο, τα έργα που δίνονται για δοκιμή, να μην είναι αρκετά για όλες τις εκπαιδεύσεις και να τερματιστεί η διαδικασία. Έτσι, σε μικρά σύνολα δεδομένων και με την αναγκαστική αφαίρεση της επιλογής στο check box “Fixed Rows of Training/Validation/Testing Sets”, όταν θα πιεστεί το κουμπί “Training / Testing” στην περιοχή των Τεχνητών Νευρωνικών Δικτύων θα εμφανιστεί το παράθυρο “Change Fixed Sizes of the Sets” που φαίνεται στην εικόνα 4.5. Ο χρήστης μπορεί να μεταβάλει τον αριθμό των έργων που θα χρησιμοποιηθούν για εκπαίδευση, για επικύρωση και για δοκιμή έτσι ώστε να επιτύχει τα αποτελέσματα που επιθυμεί. Οι αλλαγές θα ελεγχτούν αν είναι έγκυρες, αν είναι σωστός ο αριθμός που έγραψε ο χρήστης σε σχέση με το συγκεκριμένο data set και αν δεν είναι, εμφανίζονται τα κατάλληλα μηνύματα λάθους.

Στην συνέχεια, αναφέρουμε το combo box “Training Function”, το οποίο αντιπροσωπεύει την συνάρτηση με την οποία θα εκπαιδευτεί το δίκτυο. Υπάρχει μια μεγάλη επιλογή συναρτήσεων εκπαίδευσης, με την επιλογή μιας θα εισαχθούν στο δίκτυο ένα σύνολο από δεδομένα εισόδου που έχουν έξοδο ένα άλλο σύνολο δεδομένων τον στόχο, τα οποία δίνονται επίσης στο δίκτυο και με βάση αυτά θα προσαρμοστούν τα συναπτικά βάρη και τα biases, με σκοπό να μειωθεί το αποτέλεσμα της συνάρτησης

απόδοσης. Η default συνάρτηση απόδοσης καθορίζεται από το combo box "Performance Function" και αντιπροσωπεύει την συνάρτηση με βάση της οποίας θα επιτευχθεί τελικά η μάθηση του Τεχνητού Νευρωνικού Δικτύου. Το σύνολο των παραμέτρων για την μέθοδο διεξαγωγής πειραμάτων με Νευρωνικά Δίκτυα ολοκληρώνεται με το combo box "Learning Function", η οποία αντιπροσωπεύει την συνάρτηση μάθησης που εφαρμόζεται σε επανάληψη στο δίκτυο μέχρι την εμφάνιση ενός κριτηρίου τερματισμού. Εδώ αξίζει να αναφέρουμε ότι το πρώτο κριτήριο τερματισμού που χρησιμοποιείται για την διαδικασία εκπαίδευσης είναι το Early Stopping και το δεύτερο κριτήριο είναι ο μέγιστος αριθμός των Epochs. Το πρώτο κριτήριο μπορεί να μην ληφθεί υπόψιν αν αφαιρεθεί η επιλογή από το check box "Use Validation Set" και έτσι να μην χρησιμοποιηθούν τα έργα που καθορίστηκαν για επικύρωση. Το δεύτερο κριτήριο μπορεί να αποφασιστεί από τον χρήστη, ενώ μια ικανοποιητική τιμή είναι 800 Epochs για επανάληψη της εκπαίδευσης.

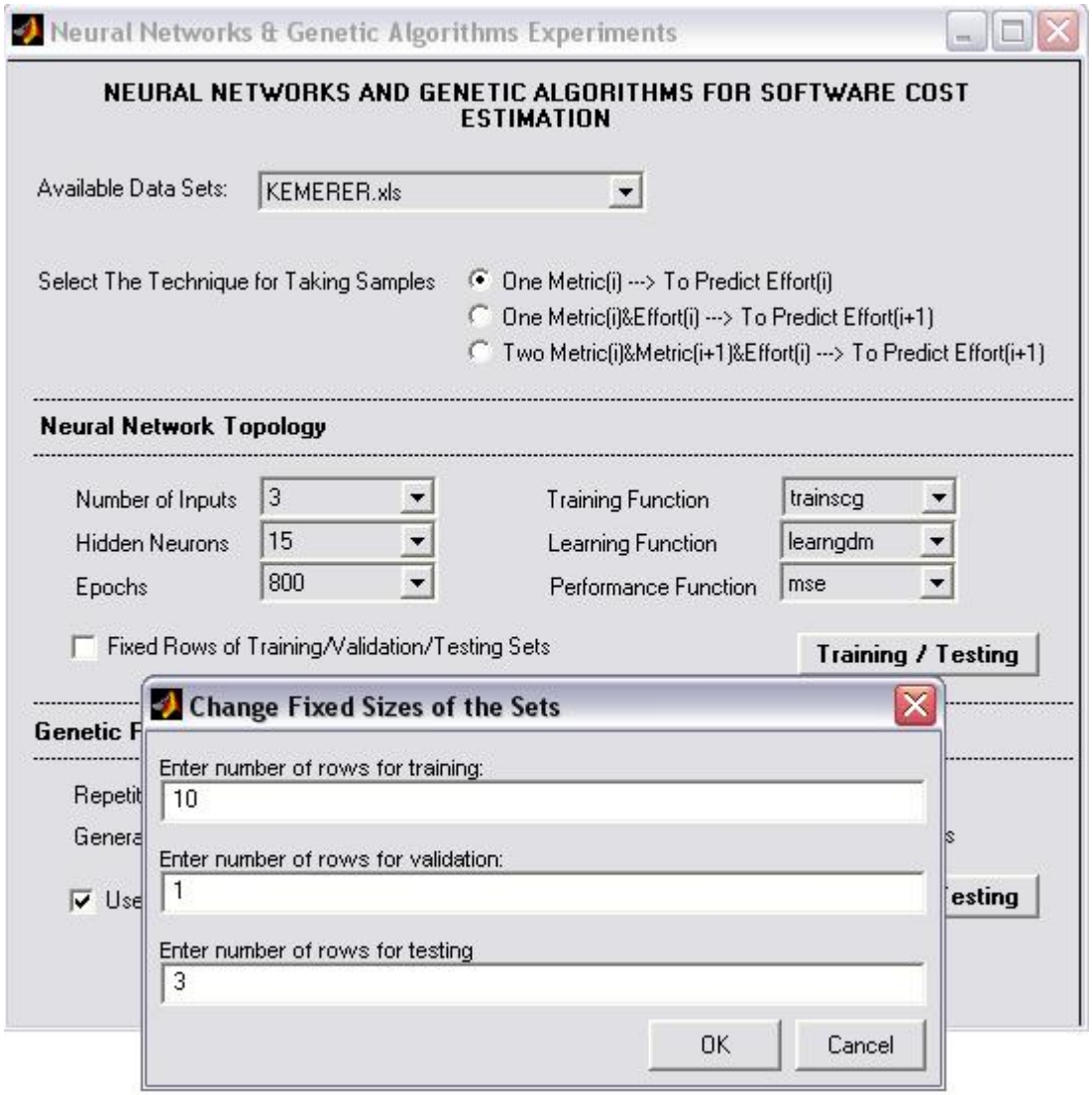


Εικόνα 4.4: Αρχικές Επιλογές και Χρήση Γενετικού Αλγόριθμου

Η δεύτερη μεθοδολογία διεξαγωγής πειραμάτων είναι η χρήση Νευρωνικού Δικτύου (και για αυτό οι προηγούμενες επιλογές στον τομέα των Νευρωνικών Δικτύων εξακολουθούν να ισχύουν) και η χρήση Γενετικού Αλγορίθμου για την επιλογή βέλτιστης τοπολογίας Νευρωνικού Δικτύου. Όπως φαίνεται στην εικόνα 4.4 είναι αναγκαίο να καθοριστούν μερικοί επιπλέον παράμετροι που θα ληφθούν υπόψιν στον Γενετικό Αλγόριθμο. Είναι σημαντικό να αναφέρουμε ότι ο Γενετικός Αλγόριθμος είναι ένας πολύπλοκος αλγόριθμος, ο οποίος χρειάζεται μεγάλο υπολογιστικό κόστος και επομένως οι τιμές θα πρέπει να είναι μικρές και να επιλεχτούν προσεχτικά. Στην συνέχεια αναφέρω τις τιμές στις παραμέτρους που προτείνονται για μια λογική εκτέλεση του αλγόριθμου, η οποία διαρκεί η ολοκλήρωσή της αρκετές ώρες.

Αρχικά, όσον αφορά τα δεδομένα, όπως αναφέραμε και πριν, ο χρήστης πρέπει να προσέξει σε περιπτώσεις που τα διαθέσιμα έργα είναι λίγα (όπως στην περίπτωση του συνόλου δεδομένων "Kemerer.xls" και "Albrecht.xls"), να μεταβάλει ιδιοχείρως τον αριθμό των έργων που θα χρησιμοποιήσει για εκπαίδευση, επικύρωση και δοκιμή του δικτύου (Εικόνα 4.5). Το combo box "Repetitions" αντιπροσωπεύει τον αριθμό τον επαναλήψεων του Γενετικού Αλγόριθμου και βασικά αποτελεί το σύνολο των γενεών ατόμων που θα παραχθούν, προτεινόμενη τιμή είναι περίπου 200 γενεές. Το combo box "Generation Atoms" αντιπροσωπεύει τον αριθμό των ατόμων, των διαφορετικών τοπολογιών δικτύων που θα παραχθούν από τον Γενετικό Αλγόριθμο, προτεινόμενη τιμή είναι 30 άτομα. Για το κάθε άτομο ή Νευρωνικό Δίκτυο που θα δημιουργηθεί θα χρησιμοποιηθεί ο αριθμός των Epochs, οι συναρτήσεις Training, Performance και Learning Functions που ορίστηκαν στον προηγούμενο τομέα. Επίσης, καθορίζονται οι δύο παράμετροι στα combo boxes "pc" και "pm" που αντιπροσωπεύουν την Πιθανότητα Διασταύρωσης και την Πιθανότητα Μετάλλαξης αντίστοιχα. Συνιστάται το check box "Print Graphs" να μην επιλεχτεί στην περίπτωση των Γενετικών Αλγορίθμων, αν ο χρήστης δεν επιθυμεί να μελετήσει την απόδοση του κάθε ενός ατόμου ξεχωριστά, και έτσι δεν θα τυπωθούν οι γραφικές παραστάσεις της πραγματικής προσπάθειας και της προβλεπόμενης προσπάθειας κατά την εκπαίδευση και κατά την δοκιμή του δικτύου (για παράδειγμα βλ. Εικόνα 4.6) που δίνονται σαν αποτέλεσμα στην μέθοδο των Τεχνητών Νευρωνικών Δικτύων. Επίσης συνίσταται, για την καλύτερη απόδοση του Γενετικού Αλγόριθμου, να αφαιρεθεί η επιλογή από το check box "Use

Validation Set" και έτσι να μην χρησιμοποιηθούν τα έργα που καθορίστηκαν για επικύρωση, δηλαδή να αφαιρεθεί το κριτήριο παύσης της εκπαίδευσης Early Stopping.



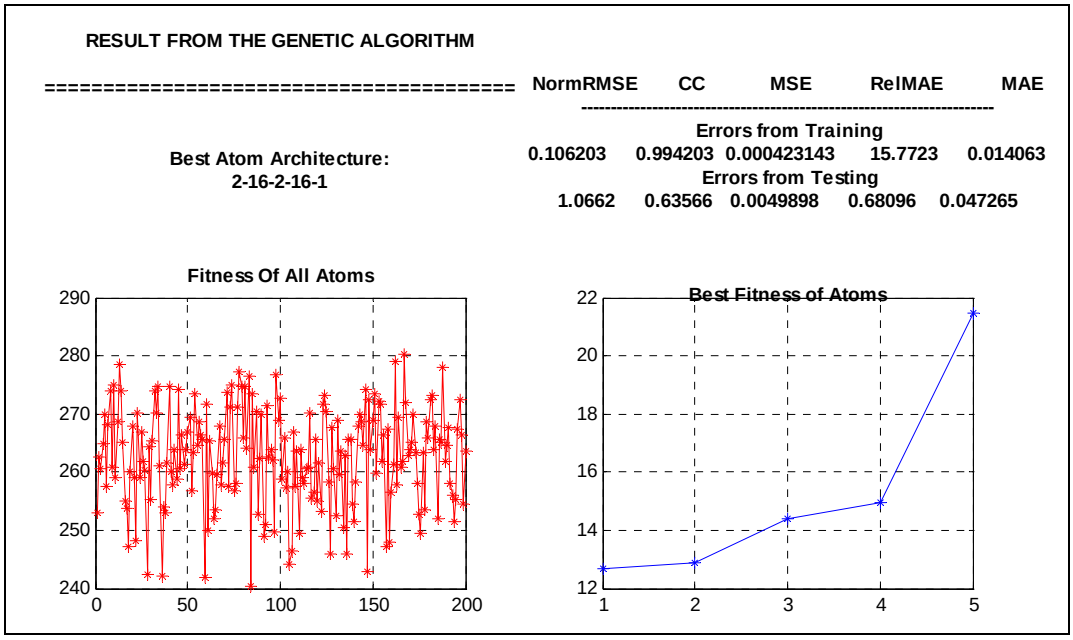
Εικόνα 4.5: Αλλαγή των Μεγεθών των Συνόλων Εκπαίδευσης/ Επικύρωσης/ Δοκιμής

4.5.6 Αποτελέσματα Πειραμάτων Γενετικού Αλγόριθμου

Σκοπός αυτού του μέρους είναι η περιγραφή της σειράς των πειραμάτων που έγιναν με την χρήση Γενετικού Αλγόριθμου και η παρουσίαση των αποτελεσμάτων. Η διαδικασία εφαρμογής Γενετικού Αλγόριθμου αφορά το έλεγχο ενός ικανοποιητικού αριθμού ατόμων με διαφορετική αρχιτεκτονική στο Νευρωνικό Δίκτυο και με την διαδικασία εκπαίδευσης και δοκιμής, επιλέγεται και κυριαρχεί η καλύτερη τοπολογία.

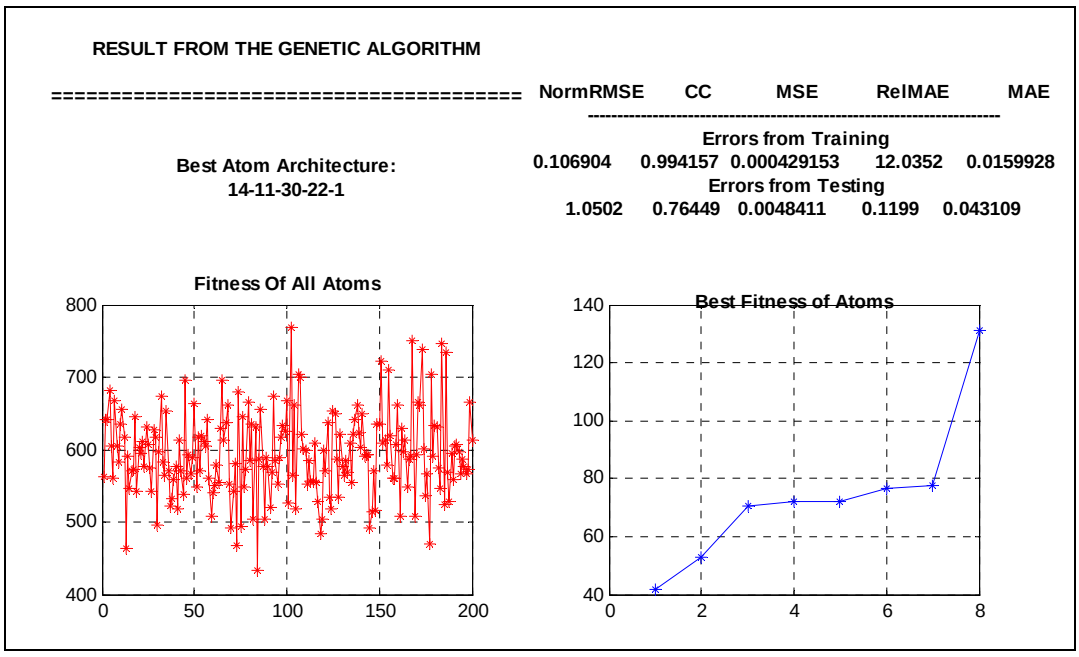
Για τα πειράματα που διεξήχθησαν χρειάστηκαν κάθε φορά περίπου πολλές ώρες λειτουργίας του Γενετικού Αλγορίθμου, ανάλογα με τις παραμέτρους και το μέγεθος του συνόλου δεδομένων που επιλέγεται. Έγιναν συνολικά δώδεκα πειράματα, τρία για κάθε σύνολο δεδομένων, αφού χρησιμοποίησα κάθε φορά διαφορετική τεχνική διεξαγωγής των συνόλων εκπαίδευσης, επικύρωσης και δοκιμής.

Στο πρώτο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων COCOMO και με την χρήση μιας τιμής μιας μετρικής (τις LOC, γραμμές κώδικα του λογισμικού ή τα FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 44 αριθμό έργων για εκπαίδευση, με 6 έργα για επικύρωση, με 12 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή σε αρχείο όπως για παράδειγμα "Cocomo 1 BEST ATOMS - 24-May-2004 - Time 4.35.xls" , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



Γραφική Παράσταση 4.5: Καλύτερη Αρχιτεκτονική COCOMO με την πρώτη τεχνική

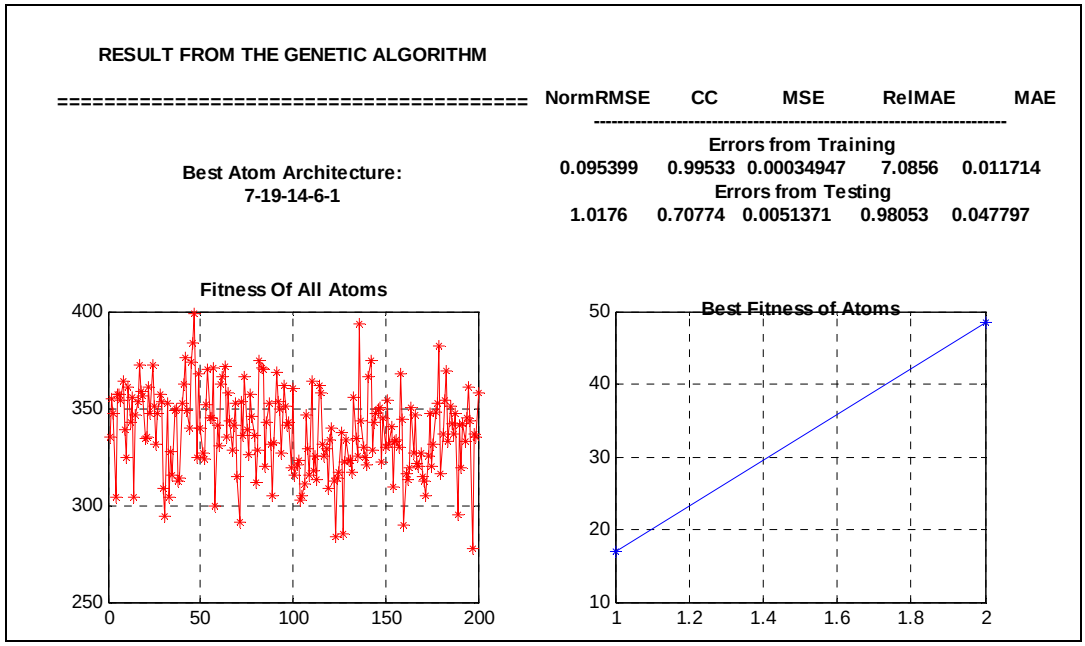
Στο δεύτερο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων COCOMO, και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού του επόμενου στην σειρά έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 44 αριθμό έργων για εκπαίδευση, με 6 έργα για επικύρωση, με 12 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο "Cocomo 2 BEST ATOMS - 20-May-2004 - Time 4.35.xls" , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



Γραφική Παράσταση 4.6: Καλύτερη Αρχιτεκτονική COCOMO με την δεύτερη τεχνική

Στο τρίτο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων COCOMO και με την χρήση δύο τιμών μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, αλλά και του επόμενου στην σειρά έργου i+1 και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της

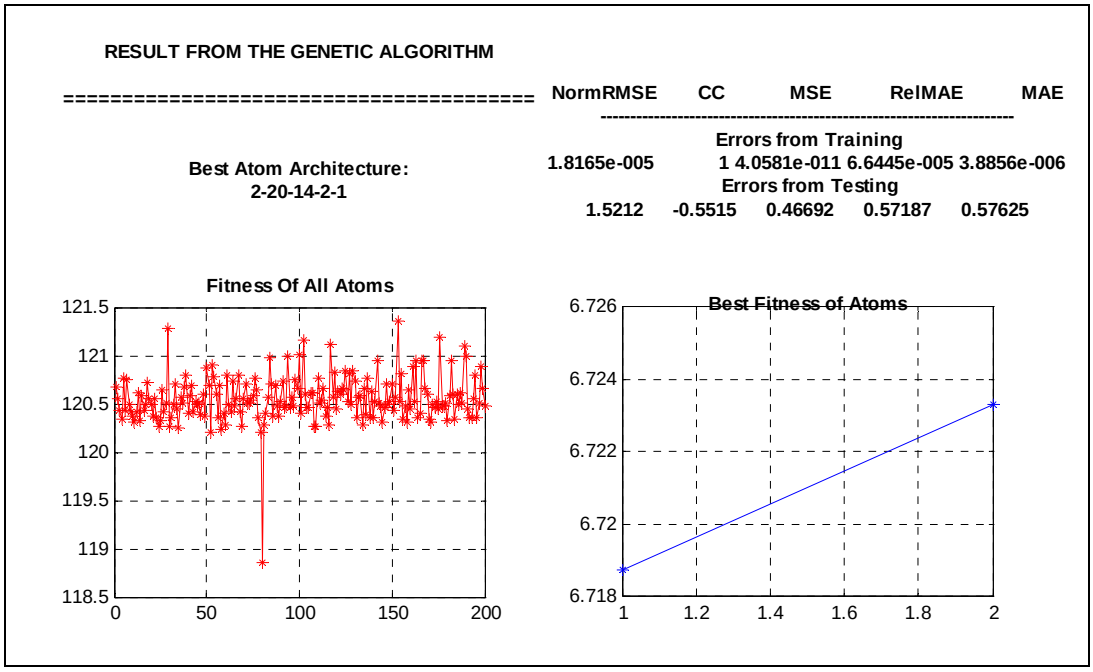
προσπάθειας (effort) ανάπτυξης λογισμικού έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 44 αριθμό έργων για εκπαίδευση, με 6 έργα για επικύρωση, με 12 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο "Cocomo 3 BEST ATOMS - 20-May-2004 - Time 4.31.xls" , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



Γραφική Παράσταση 4.7: Καλύτερη Αρχιτεκτονική COCOMO με την τρίτη τεχνική

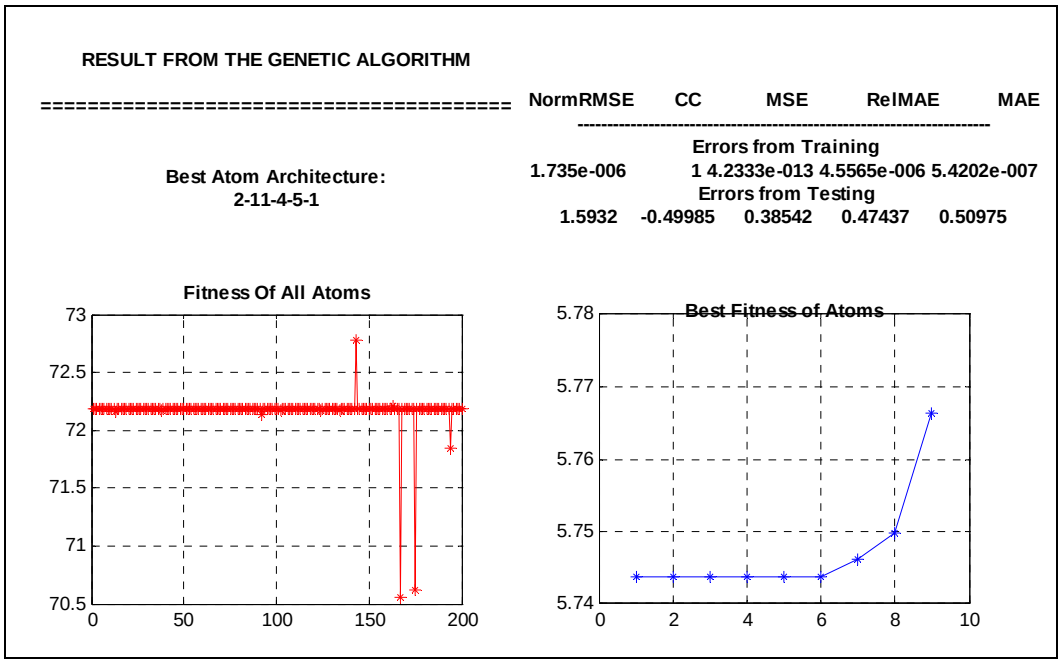
Στο τέταρτο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Kemerer και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 8 αριθμό έργων για εκπαίδευση, με 1 έργα για επικύρωση, με 5 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική

μορφή στο αρχείο “ Kemerer 1 BEST ATOMS - 19-May-2004 - Time 19.44.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



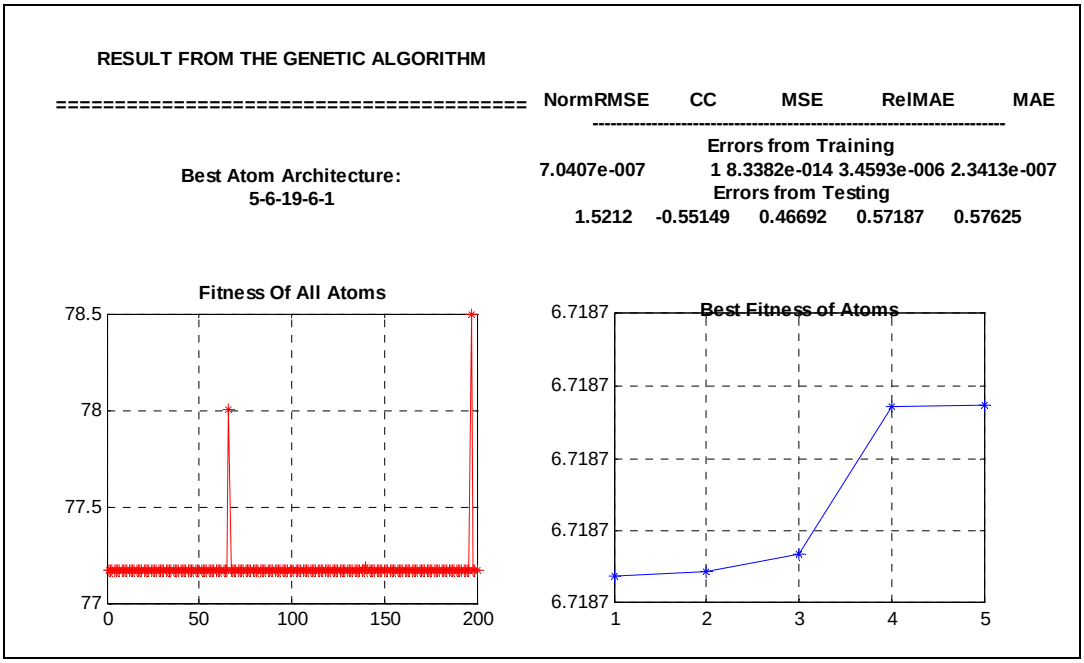
Γραφική Παράσταση 4.8: Καλύτερη Αρχιτεκτονική Kemerer με την πρώτη τεχνική

Στο πέμπτο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Kemerer, και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού του επόμενου στην σειρά έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 8 αριθμό έργων για εκπαίδευση, με 1 έργα για επικύρωση, με 5 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “Kemerer 2 BEST ATOMS - 19-May-2004 - Time 21.6.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



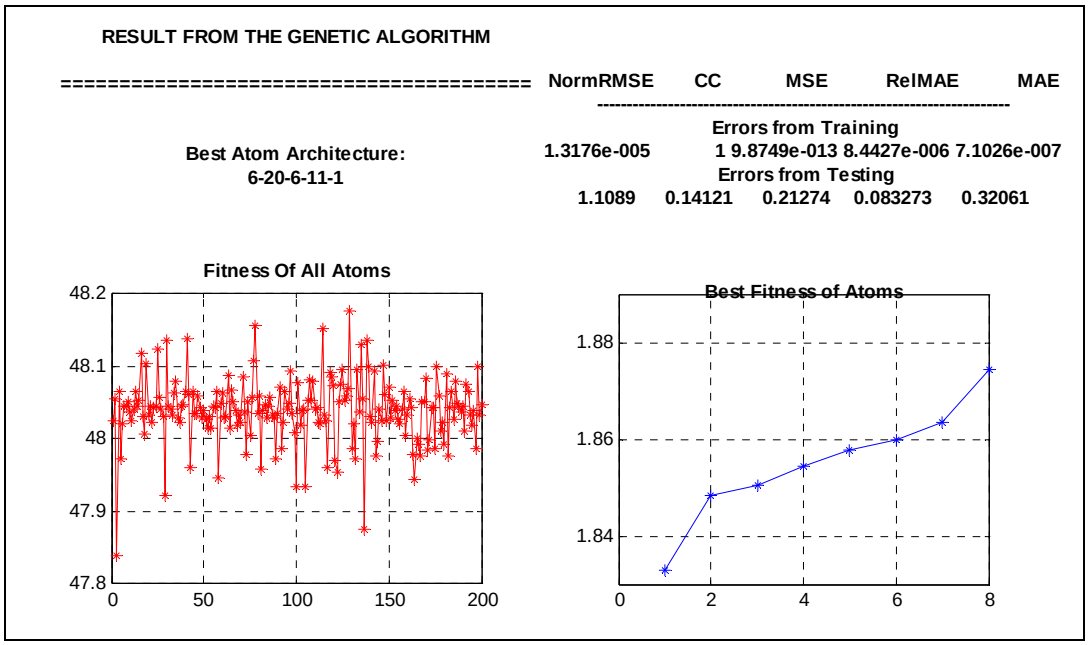
Γραφική Παράσταση 4.9: Καλύτερη Αρχιτεκτονική Kemerer με την δεύτερη τεχνική

Στο έκτο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Kemerer και με την χρήση δύο τιμών μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, αλλά και του επόμενου στην σειρά έργου i+1 και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 7 αριθμό έργων για εκπαίδευση, με 1 έργα για επικύρωση, με 6 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “Kemerer 3 BEST ATOMS - 19-May-2004 - Time 19.35.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



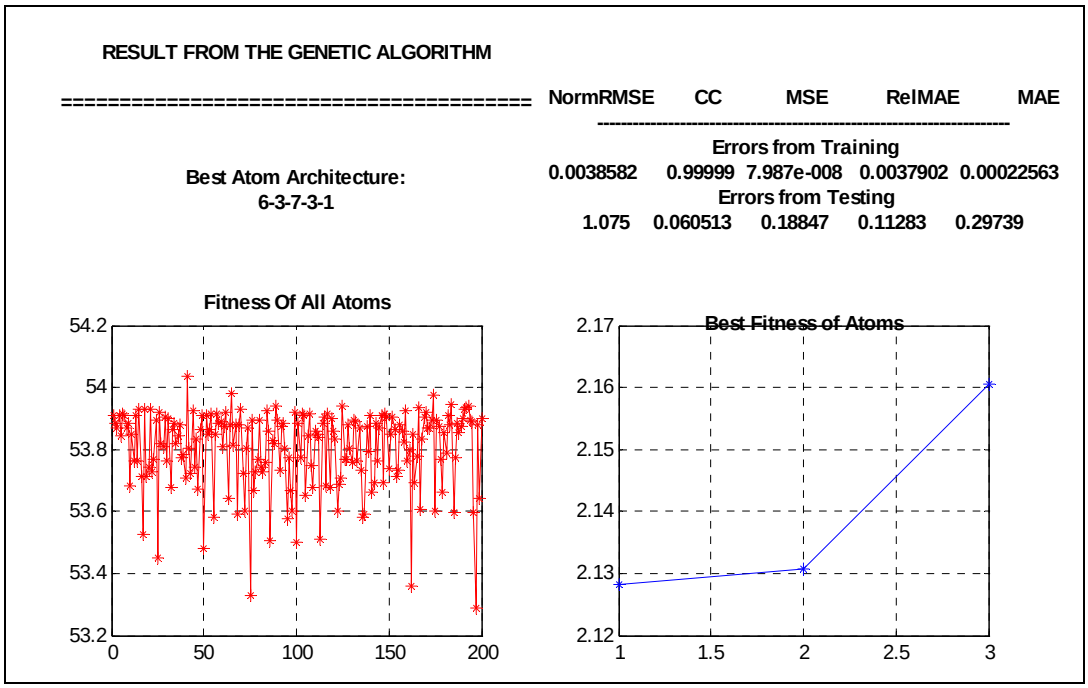
Γραφική Παράσταση 4.10: Καλύτερη Αρχιτεκτονική Kemerer με την τρίτη τεχνική

Στο έβδομο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Albrecht και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 16 αριθμό έργων για εκπαίδευση, με 2 έργα για επικύρωση, με 4 έργα για δοκιμή, με 200 γενεές και 100 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “Albrecht 1 BEST ATOMS - 20-May-2004 - Time 9.29.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



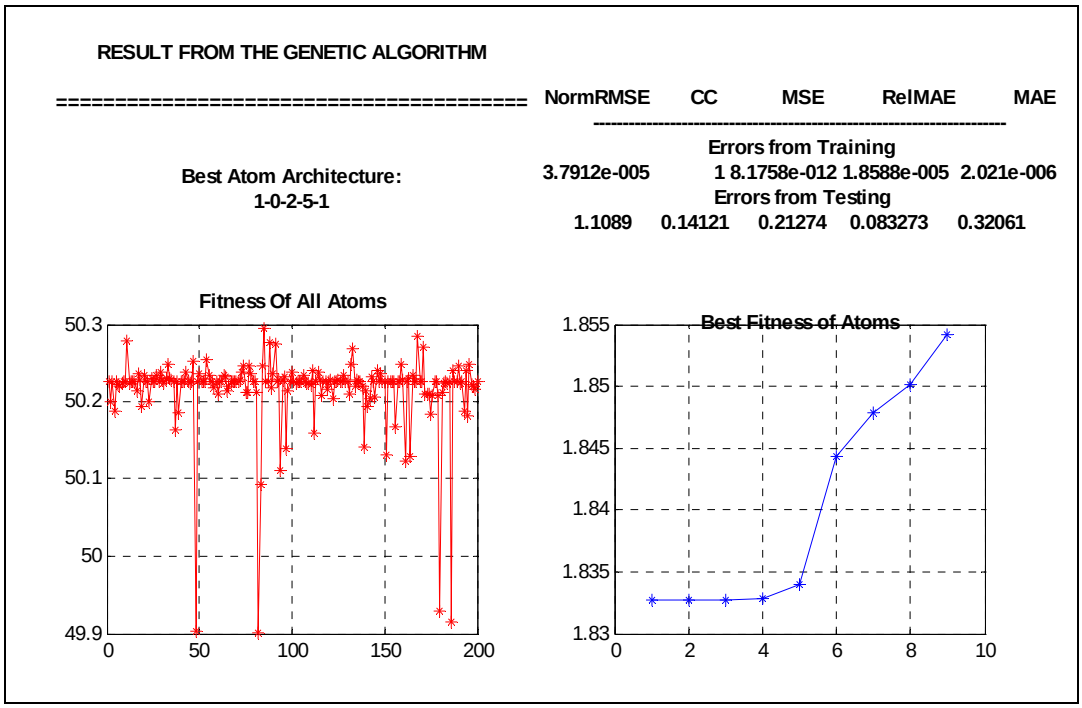
Γραφική Παράσταση 4.11: Καλύτερη Αρχιτεκτονική Albrecht με την πρώτη τεχνική

Στο όγδοο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων του Albrecht, και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού του επόμενου στην σειρά έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 12 αριθμό έργων για εκπαίδευση, με 2 έργα για επικύρωση, με 8 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο "Albrecht 2 BEST ATOMS - 20-May-2004 - Time 3.1.xls", ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



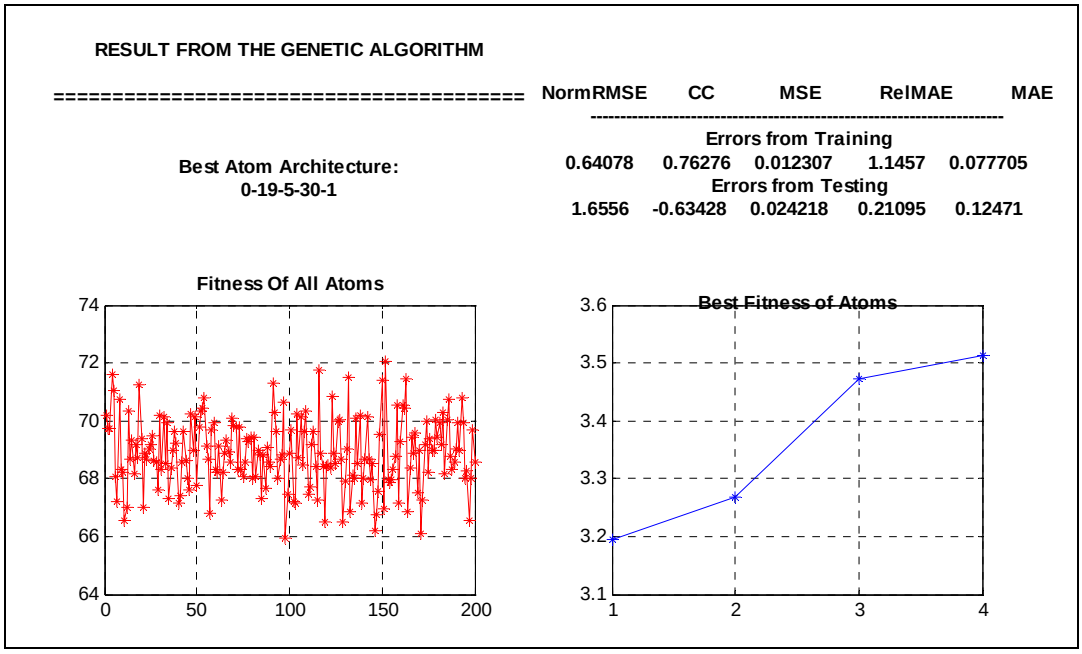
Γραφική Παράσταση 4.12: Καλύτερη Αρχιτεκτονική Albrecht με την δεύτερη τεχνική

Στο ένατο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Albrecht και με την χρήση δύο τιμών μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, αλλά και του επόμενου στην σειρά έργου i+1, , και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 12 αριθμό έργων για εκπαίδευση, με 2 έργα για επικύρωση, με 8 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “Albrecht 3 BEST ATOMS - 20-May-2004 - Time 1.29.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



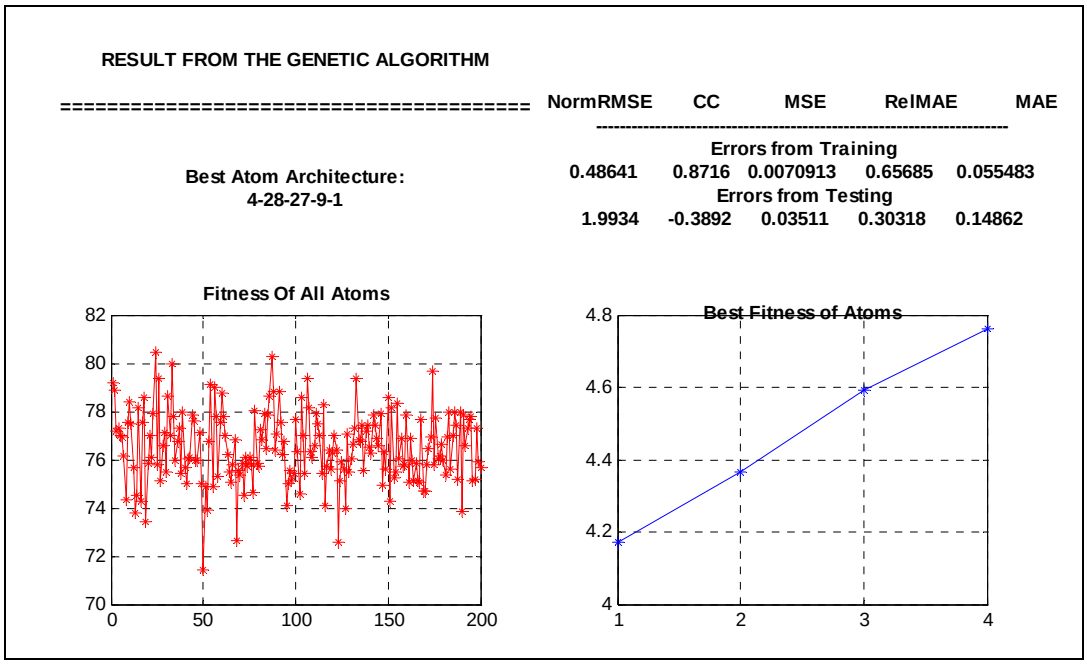
Γραφική Παράσταση 4.13: Καλύτερη Αρχιτεκτονική Albrecht με την τρίτη τεχνική

Στο δέκατο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Desharnais και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης "trainscg", με συνάρτηση εκμάθησης "learngdm", με συνάρτηση απόδοσης (performance) "mse", με 56 αριθμό έργων για εκπαίδευση, με 8 έργα για επικύρωση, με 16 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο "Desharnais 1 BEST ATOMS - 25-May-2004 - Time 4.35.xls" , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



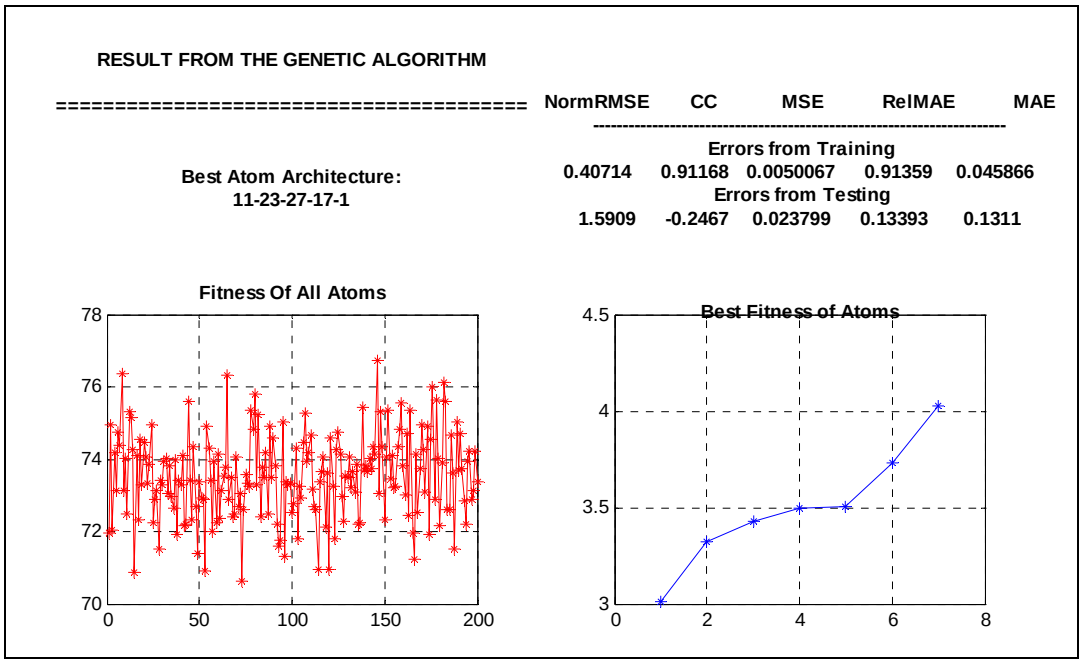
Γραφική Παράσταση 4.14: Καλύτερη Αρχιτεκτονική Desharanais με την πρώτη τεχνική

Στο ενδέκατο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων Desharnais, και με την χρήση μιας τιμής μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού του επόμενου στην σειρά έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 56 αριθμό έργων για εκπαίδευση, με 8 έργα για επικύρωση, με 16 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “Desharnais 2 BEST ATOMS - 20-May-2004 - Time 11.23.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



Γραφική Παράσταση 4.15:Καλύτερη Αρχιτεκτονική Desharnais με την δεύτερη τεχνική

Στο δωδέκατο πείραμα χρησιμοποιήθηκε το σύνολο δεδομένων του Desharnais και με την χρήση δύο τιμών μιας μετρικής (όπως για παράδειγμα το LOC, γραμμές κώδικα του λογισμικού ή το FP, λειτουργικά σημεία του λογισμικού) του λογισμικού έργου i, αλλά και του επόμενου στην σειρά έργου i+1, , και της τιμής της προσπάθειας που χρειάστηκε για να διεξαχθεί το έργο i, διεξάγεται πρόβλεψη για την τιμή του κόστους ή της προσπάθειας (effort) ανάπτυξης λογισμικού έργου i+1, όπου i ο αριθμός των έργων. Έγινε με αριθμό επαναλήψεων (epochs) της εκπαίδευσης του Νευρωνικού Δικτύου ίσο με 500, με συνάρτηση εκπαίδευσης “trainscg”, με συνάρτηση εκμάθησης “learngdm”, με συνάρτηση απόδοσης (performance) “mse”, με 56 αριθμό έργων για εκπαίδευση, με 8 έργα για επικύρωση, με 16 έργα για δοκιμή, με 200 γενεές και 30 διαφορετικές τοπολογίες ατόμων. Τα αποτελέσματα των 200 καλύτερων ατόμων βρίσκονται σε ηλεκτρονική μορφή στο αρχείο “ Desharnais 3 BEST ATOMS - 20-May-2004 - Time 6.8.xls” , ενώ η γραφική παράσταση της καλύτερης αρχιτεκτονικής φαίνεται πιο κάτω.



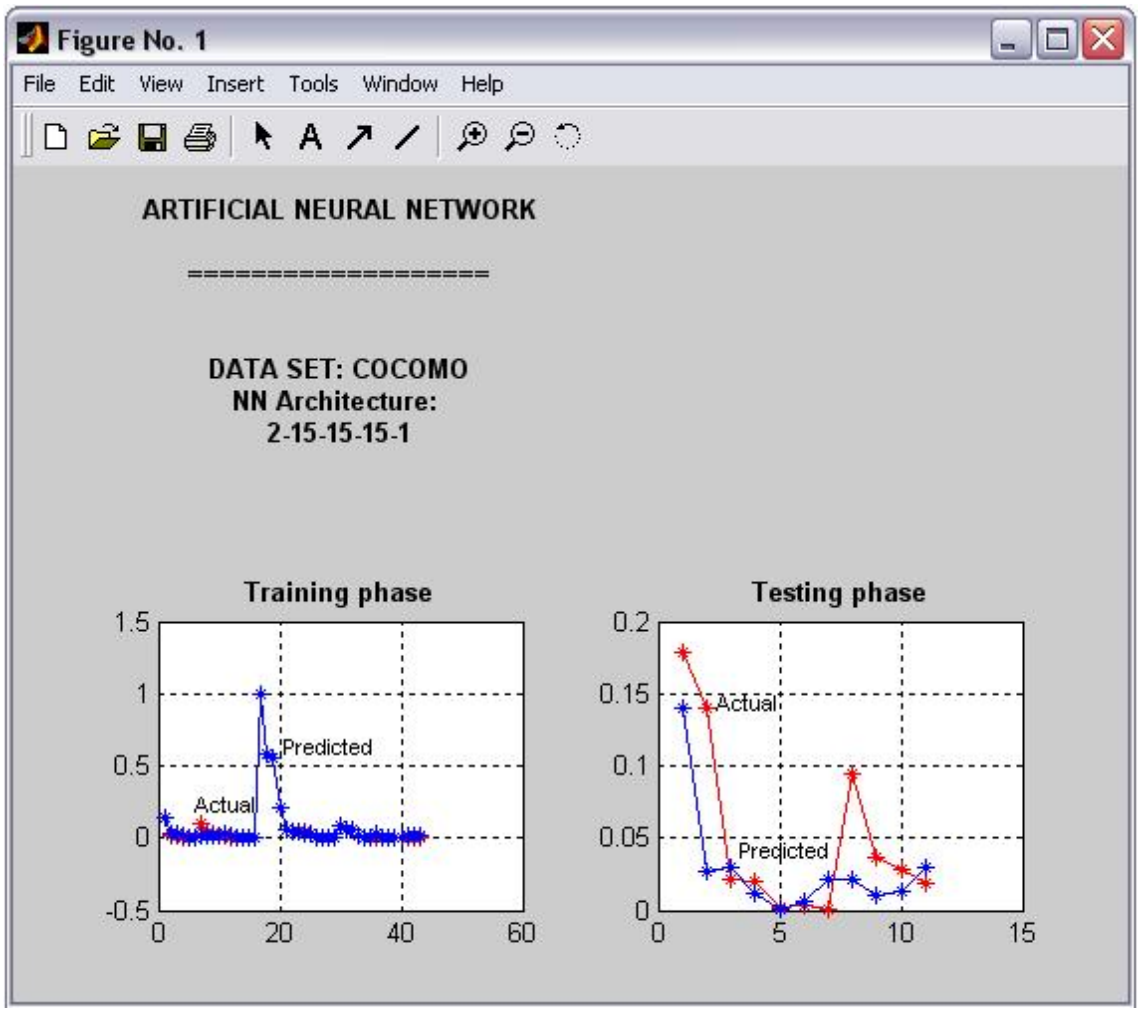
Γραφική Παράσταση 4.16: Καλύτερη Αρχιτεκτονική Desharnais με την τρίτη τεχνική

4.5.7 Ενδεικτικά Αποτελέσματα

Σε αυτόν τον τομέα παρουσιάζεται ένα ενδεικτικό αποτέλεσμα χρήσης του εργαλείου και της μεθόδου των Τεχνητών Νευρωνικών Δικτύων και τα συμπεράσματα και οι παρατηρήσεις μετά από πολλά πειράματα των οποίων οι γραφικές παραστάσεις και τα αποτελέσματα συμπεριλαμβάνονται σε ηλεκτρονική μορφή. Επίσης, θα παρουσιαστούν τα αποτελέσματα των πειραμάτων με την δεύτερη μέθοδο, του Γενετικού Αλγόριθμου.

Τα αποτελέσματα της μεθόδου των Τεχνητών Νευρωνικών Δικτύων γράφονται σε αρχεία τύπου excel, και περιλαμβάνουν τις τιμές των λαθών: η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (CC, Correlation Coefficient), το Μέσο Τετραγωνικό Σφάλμα (MSE, Mean Squared Error), MRE, το Σχετικό Μέσο Σφάλμα (MRE, Mean Relative Error) και το Μέσο Απόλυτο Σφάλμα (MAE, Mean Absolute Error), κατά την εκπαίδευση και κατά την δοκιμή του Τεχνητού Νευρωνικού Δικτύου και με βάση τα οποία μπορούμε να διακρίνουμε αν υπάρχει επιτυχία ή όχι στην πρόβλεψη. Επίσης, τυπώνονται οι σχετικές γραφικές παραστάσεις, όπου μπορεί εύκολα ο χρήστης να δει αν η πρόβλεψη ακολουθεί ή όχιτην πραγματική τιμή. Ένα παράδειγμα γραφικών παραστάσεων της πραγματικής

και της προβλεπόμενης τιμής της προσπάθειας ανάπτυξης λογισμικού στην εκπαίδευση και στην δοκιμή του Τεχνητού Νευρωνικού Δικτύου φαίνεται στην συνέχεια.



Εικόνα 4.6: Παράδειγμα Αποτελέσματος Τεχνητών Νευρωνικών Δικτύων

Όπως φαίνεται και από το παράδειγμα του αποτελέσματος της χρήσης Τεχνητών Νευρωνικών Δικτύων για την πρόβλεψη του κόστους ανάπτυξης λογισμικού, κατά την φάση της εκπαίδευσης, το δίκτυο μπορεί να ταυτίζει την πρόβλεψη με την πραγματική τιμή του κόστους και άρα εκπαιδεύεται επαρκώς. Κατά την δοκιμή του δικτύου με άγνωστα δεδομένα, φαίνεται ότι οι πραγματικές τιμές μπορούν να προσεγγιστούν σχετικά σε ικανοποιητική ακρίβεια. Μετά από την διεξαγωγή πολλών πειραματικών τρεξιμάτων μπορούμε να εξάγουμε μερικές παρατηρήσεις και συμπεράσματα τα οποία παραθέτονται σε αυτό το σημείο.

Αρχικά, συμπεράναμε πως το δίκτυο έχει περισσότερες πιθανότητες να εκπαιδευτεί πιο καλά, εάν χρησιμοποιηθούν πολλές επαναλήψεις στο εργαλείο, καθώς και αν χρησιμοποιηθούν πολλοί νευρώνες στο εσωτερικό επίπεδο. Ακόμα, εκπαιδεύεται καλύτερα εάν ληφθεί υπόψιν μόνο το κριτήριο του αριθμού των Epochs της εκπαίδευσης και εάν δεν χρησιμοποιηθεί το κριτήριο Early Stopping, αφού το κριτήριο αυτό σταματά πολύ σύντομα την εκπαίδευση του δικτύου, πριν προλάβει να μάθει το δίκτυο. Επίσης, μια ακόμα παρατήρηση είναι ότι αν το σύνολο δοκιμής φτάσει σε ελάχιστο σε σημαντικά διαφορετική επανάληψη από το λάθος επικύρωσης, τότε αυτό μπορεί να σημαίνει λανθασμένο διαχωρισμό των συνόλων. Ακόμα, έγινε μια προσπάθεια αλλαγής διαφόρων εσωτερικών παραμέτρων του δικτύου, όπως είναι οι συναρτήσεις εκπαίδευσης, μάθησης και απόδοσης. Έγιναν διάφοροι δυνατοί συνδυασμοί χωρίς όμως να καταφέρουμε να βελτιώσουμε σημαντικά την πρόβλεψη. Παρατηρήσαμε όμως, ότι στην περίπτωση που άλλαζε η τοπολογία του δικτύου είχαμε διαφορετική πρόβλεψη και έτσι κρίθηκε αναγκαία η προσπάθεια εύρεσης της βέλτιστης αρχιτεκτονικής δικτύου που θα δίνει και πιο επιτυχημένη πρόβλεψη. Αυτό μπορεί, όπως εξηγήσαμε και προηγουμένως να γίνει με την χρήση Γενετικού Αλγόριθμου. Τα αποτελέσματα και συμπεράσματα παραθέτονται στην συνέχεια.

Τα αποτελέσματα της μεθόδου των Γενετικών Αλγορίθμων εμφανίζονται μετά από επιλογή του χρήστη, αν επιθυμεί να μελετήσει την απόδοση του κάθε ενός ατόμου ξεχωριστά, θα τυπωθούν όλες οι γραφικές παραστάσεις της πραγματικής προσπάθειας και της προβλεπόμενης προσπάθειας κατά την εκπαίδευση και κατά την δοκιμή του δικτύου. Ακόμα, τυπώνονται σε αρχεία excel τα αριθμητικά σφάλματα του κάθε ενός βέλτιστου ατόμου σε κάθε γενεά: η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (CC, Correlation Coefficient), το Μέσο Τετραγωνικό Σφάλμα (MSE, Mean Squared Error), το Σχετικό Μέσο Σφάλμα (MRE, Mean Relative Error) και το Μέσο Απόλυτο Σφάλμα (MAE, Mean Absolute Error) κατά την εκπαίδευση και κατά την δοκιμή του Τεχνητού Νευρωνικού Δικτύου. Με βάση αυτά τα καλύτερα άτομα, τυπώνεται η βέλτιστη αρχιτεκτονική και τα σφάλματα που έχει το καλύτερο άτομο από αυτά που δοκιμάστηκαν, μαζί με μια γραφική παράσταση της συνολικής πορείας του fitness των καλύτερων ατόμων, καθώς και πώς κινείται εξελικτικά η γραφική παράσταση των βέλτιστων ατόμων μετά το πρώτο πιο βέλτιστο, καταλήγοντας στην κορυφή με το

βέλτιστο άτομο. Έτσι μπορούμε για κάθε σύνολο δεδομένων να βρούμε την βέλτιστη αρχιτεκτονική δικτύου, που εξαγάγει την καλύτερη δυνατή πρόβλεψη και η οποία μπορεί να είναι πολύ χρήσιμη στην συνέχεια.

Τα αποτελέσματα συγκεντρώθηκαν στον πίνακα 4.8, όπου για κάθε μια τεχνική εξαγωγής των συνόλων δεδομένων και για κάθε ένα σύνολο δεδομένων έγινε ένα τρέξιμο, έτσι ώστε να βρεθεί η βέλτιστη αρχιτεκτονική. Στον πίνακα αυτό φαίνονται τα σύνολα δεδομένων που χρησιμοποιήθηκαν, στην στήλη IN περιγράφεται η τεχνική με την οποία λαμβάνονται τα σύνολα εισόδου, αν χρησιμοποιήθηκαν γραμμές κώδικα (LOC) ή λειτουργικά σημεία (FP) σε συνδυασμό με τη προσπάθεια ή όχι, όπου το i καθορίζει τον αριθμό των έργων που χρησιμοποιήθηκαν. Στην στήλη OUT περιγράφεται το αποτέλεσμα της τεχνικής που είναι η προσπάθεια (EFF) που χρειάζεται να καταβληθεί για την ανάπτυξη των έργων. Στην στήλη Inputs αναγράφεται ο αριθμός των νευρώνων που χρησιμοποιούνται στο εξωτερικό επίπεδο εισόδου, ενώ στις στήλες Hidden Layer 1, Hidden Layer 2 και Hidden Layer 3, ο αριθμός των νευρώνων σε κάθε εσωτερικό κρυμμένο επίπεδο. Ακόμα, στον πίνακα εμφανίζονται τα ενδεικτικά αποτελέσματα των σχετικών πειραμάτων που έγιναν. Στην συνέχεια, φαίνονται οι λεπτομέρειες των πειραμάτων, καθώς και μερικά ενδεικτικά αποτελέσματα Γραφικών Παραστάσεων.

Πίνακας 4.8: Αποτελέσματα Καλύτερων Αρχιτεκτονικών Γενετικού Αλγόριθμου

							TRAINING					TESTING				
DATASET	**Inputs**	**Hidden Layer 1**	**Hidden Layer 2**	**Hidden Layer 3**	**IN**	**OUT**	**Normalized RMSE**	**CC**	**MSE**	**MRE**	**MAE**	**Normalized RMSE**	**CC**	**MSE**	**MRE**	**MAE**
COCOMO	2	16	2	16	LOC(i)	EFF(i)	0.106203	0.994203	0.000423143	15.7723	0.014063	1.0662	0.63566	0.0049898	0.68096	0.047265
COCOMO	**14**	**11**	**30**	**22**	**LOC(i), EFF(i)**	**EFF (i+1)**	**0.106904**	**0.994157**	**0.000429153**	**12.0352**	**0.0159928**	**1.0502**	**0.76449**	**0.0048411**	**0.1199**	**0.043109**
COCOMO	7	19	14	6	LOC(i+1) EFF(i)	EFF (i+1)	0.095399	0.99533	0.00034947	7.0856	0.011714	1.0176	0.70774	0.0051371	0.98053	0.047797
Kemerer	2	20	14	2	LOC(i)	EFF(i)	1.8165E-005	1	4.0581E-011	6.6445E-005	3.8856E-006	1.5212	-0.5515	0.46692	0.57187	0.57625
Kemerer	2	11	4	5	LOC(i), EFF(i)	EFF (i+1)	1.735E-006	1	4.2333E-013	4.5565E-006	5.4202E-007	1.5932	-0.49985	0.38542	0.47437	0.50975
Kemerer	5	6	19	6	LOC(i+1) EFF(i)	EFF (i+1)	7.0407E-007	1	8.3382E-014	3.4593E-006	2.3413E-007	1.5212	-0.55149	0.46692	0.57187	0.57625
Albrecht	**6**	**20**	**6**	**11**	**FP(i)**	**EFF(i)**	**1.3176E-005**	**1**	**9.8749E-013**	**8.4427E-006**	**7.1026E-007**	**1.1089**	**0.14121**	**0.21274**	**0.083273**	**0.32061**
Albrecht	6	3	7	3	FP(i), EFF(i)	EFF (i+1)	0.0038582	0.99999	7.987E-008	0.0037902	0.00022563	1.075	0.060513	0.18847	0.11283	0.29739
Albrecht	1	0	2	5	FP(i+1), EFF(i)	EFF (i+1)	3.7912E-005	1	8.1758E-012	1.8588E-005	2.021E-006	1.1089	0.14121	0.21274	0.083273	0.32061
Desharnais	1	19	5	30	FP(i)	EFF(i)	0.64078	0.76276	0.012307	1.1457	0.077705	1.6556	-0.63428	0.024218	0.21095	0.12471
Desharnais	4	28	27	9	FP(i), EFF(i)	EFF (i+1)	0.48641	0.8716	0.0070913	0.65685	0.055483	1.9934	-0.3892	0.03511	0.30318	0.14862
Desharnais	11	23	27	17	FP(i+1), EFF(i)	EFF (i+1)	0.40714	0.91168	0.0050067	0.91359	0.045866	1.5909	-0.2467	0.023799	0.13393	0.1311

Τα αποτελέσματα που πήραμε από τον Γενετικό Αλγόριθμο, δεν είναι όπως περιμέναμε. Εντούτοις, οι βέλτιστες αρχιτεκτονικές των Τεχνητών Νευρωνικών Δικτύων παρουσιάζουν μεγάλη βελτίωση στις προβλέψεις, κατά την εκπαίδευση ενώ τα αποτελέσματα δεν φαίνονται ιδιαίτερα βελτιωμένα στην δοκιμή του δικτύου, από όταν απλά χρησιμοποιήθηκαν συνδυασμοί τοπολογιών με το εργαλείο πειραμάτων NeuroShell και το απλό εργαλείο δημιουργίας Νευρωνικών Δικτύων. Στον πίνακα 4.8 εμφανίζονται με έντονο χαρακτήρα τα καλύτερα αποτελέσματα, που σε σχέση τα προηγούμενα αποτελέσματα με την χρήση Τεχνητών Νευρωνικών Δικτύων παρουσιάζουν κάποια βελτίωση.

Όπως φάνηκε και από τον πίνακα 4.8, αλλά και από τις γραφικές παραστάσεις των αποτελεσμάτων του Γενετικού Αλγόριθμου, έχουν γίνει κάποιες μεμονωμένες βελτιώσεις στα σύνολα δεδομένων COCOMO και Albrecht. Υπάρχει βελτίωση στο COCOMO στο σφάλμα του Συντελεστή Συσχέτισης, ο οποίος αυξήθηκε από 0,44737 που είχαμε στο καλύτερο πείραμα και έγινε 0.76449, δηλαδή πλησίασε την μονάδα δείχνοντας μεγαλύτερη ικανότητα πιστής ακολουθίας της τάσης των αποτελεσμάτων. Επίσης, στο COCOMO παρατηρείται και μια γενική βελτίωση στο σφάλμα του MRE αφού από 52,54% απόκλιση στις τιμές παρουσιάζεται τώρα 11,99% μια σημαντική βελτίωση στις τιμές αφού μεταφράζεται σε 88% σύγκλιση ή επιτυχία στην πρόβλεψη των τιμών. Η βελτίωση όμως δεν εμφανίζεται σε όλα τα σφάλματα, αλλά το Normalized RMSE σφάλμα εμφανίζεται πολύ υψηλό, σε σημείο που δεν είναι αποδεκτό.

Ακόμα, μια σημαντική βελτίωση της μεθόδου των Γενετικών Αλγορίθμων φαίνεται και από το γεγονός ότι τώρα τα αποτελέσματα από το σύνολο δεδομένων Albrecht εμφανίζονται σαν πολύ επιτυχημένα αποτελέσματα, ενώ πριν δεν συμπεριλαμβάνονταν καν σαν ενδεικτικά βέλτιστα αποτελέσματα στον πίνακα 4.7. Γενικά παρατηρείται μια βελτίωση σε όλα τα σφάλματα αφού πέφτουν σε πολύ χαμηλές τιμές και να δίνουν ικανοποιητικές μετρήσεις, με εξαίρεση το Normalized RMSE το οποίο και πάλι εμφανίζεται μεγαλύτερο από την μονάδα

Γενικά με την εφαρμογή Γενετικού Αλγόριθμου στην τεχνική, παρατηρήσαμε ότι τα αποτελέσματα δεν αποφέρανε τις επιθυμητές αλλαγές, αν και επέφεραν κάποιες μεμονωμένες βελτιώσεις στα σφάλματα. Η μέθοδος μπορεί να θεωρηθεί σαν μια προσπάθεια βελτιστοποίησης, η οποία ίσως με περισσότερη επεξεργασία των εισόδων και των παραμέτρων να έδινε ακόμα καλύτερα αποτελέσματα.

Κεφάλαιο 5

Παρουσίαση Αποτελεσμάτων

5.1 Σκοπός

Σκοπός αυτού του κεφαλαίου είναι η περιγραφή μιας περιληπτικής ανασκόπησης των βημάτων της έρευνας που προηγήθηκε, η παρουσίαση γενικών αποτελεσμάτων της μεθόδου που ακολουθήθηκε με το εργαλείο NeuroShell 2 και της μεθόδου με το εργαλείο διεξαγωγής πειραμάτων που υλοποιήθηκε και το οποίο χρησιμοποιεί Τεχνητά Νευρωνικά Δίκτυα και εφαρμόζει Γενετικό Προγραμματισμό σε αυτά για να πετύχει βελτίωση των αποτελεσμάτων. Στην συνέχεια, θα γίνει παρουσίαση και σύγκριση των καλύτερων αποτελεσμάτων με τις τεχνικές που έγιναν και με άλλες μεθόδους και τέλος θα παραθέσω μερικά συμπεράσματα, καθώς και σχετικά προβλήματα που παρουσιάστηκαν.

5.2 Γενική Ανασκόπηση

Η έρευνα ξεκίνησε με την προσπάθεια μελέτης και κατανόησης της πολύπλοκης λειτουργίας μεθόδων Υπολογιστικής Νοημοσύνης, με την βοήθεια της οποίας θα ήταν δυνατή η μοντελοποίηση και τελικά η διεξαγωγή πρόβλεψης του κόστους ή της προσπάθειας ανάπτυξης έργων λογισμικού. Το ενδιαφέρον εστιάστηκε

στα Τεχνητά Νευρωνικά Δίκτυα, τα οποία είναι μια αρκετά διαδεδομένη μεθοδολογία εκμάθησης προτύπων δεδομένων και εξαγωγής υπολογισμών και συμπερασμάτων. Έχουν την γενική ικανότητα που εκφράζεται μέσα από τις διαδικασίες υπολογισμών, συλλογισμών, λογικής, διακρίβωσης, μάθησης, χρήσης γλώσσας, αντίληψης του περιβάλλοντος σε διαφορετικούς βαθμούς λεπτομέρειας, εξοικείωσης σε νέο περιβάλλον, αυτοδιόρθωσης και επινόησης. Μπορούν να προσεγγίζουν οποιανδήποτε συνάρτηση, και με την ικανότητά τους να γενικεύουν και να λύνουν προβλήματα μεγάλης πολυπλοκότητας, εμφανίζονται σαν μια από τις πιο βέλτιστες μεθόδους επίλυσης του προβλήματος του υπολογισμού του κόστους ανάπτυξης λογισμικού

Αρχικά, χρησιμοποιήθηκε το εργαλείο της Ward Systems Group, NeuroShell 2, όπου δημιουργήσαμε Νευρωνικά Δίκτυα και τα οποία εκπαιδεύσαμε να δίνουν βελτιωμένες προβλέψεις του κόστους ανάπτυξης λογισμικού, έτσι ώστε στην συνέχεια να καθοριστεί ένα πλαίσιο αποτελεσμάτων με βάση το οποίο θα μπορέσουμε να βασίσουμε την έρευνα και να αξιολογήσουμε πιο εύκολα τα αποτελέσματα. Στη συνέχεια, υλοποιήθηκε ένα εργαλείο διεξαγωγής πειραμάτων, στην MATLAB®, η οποία είναι μια υψηλής απόδοσης γλώσσα που χρησιμοποιείται για Technical Computing και η οποία μπορεί να εκφράζει λύσεις στα προβλήματα με μαθηματικό τρόπο. Υποστηρίζει την δημιουργία Τεχνητών Νευρωνικών Δικτύων, με την βοήθεια των οποίων μπορούμε να λύνουμε προβλήματα πρόβλεψης. Η μέθοδος βελτιστοποιήθηκε με την προσαρμογή στο εργαλείο Γενετικού Αλγόριθμου, ο οποίος βοήθησε στην εύρεση της κατάλληλης αρχιτεκτονικής του δικτύου με σκοπό εξαγωγής καλύτερων αποτελεσμάτων από πριν.

Ο Γενετικός Προγραμματισμός είναι μια σχετικά νέα τεχνική, όσον αφορά την εφαρμογή της στο συγκεκριμένο θέμα της επίλυσης του προβλήματος του υπολογισμού του κόστους ανάπτυξης λογισμικού. Έχει αποδειχτεί αποτελεσματική στην μοντελοποίηση δεδομένων και στην παραγωγή εξισώσεων που περιγράφουν με επιτυχία τις διαδικασίες, και που τελικά είναι πολύ χρήσιμος στην αναζήτηση λύσεων σε πρακτικά προβλήματα βελτιστοποίησης και εκμάθησης. Είναι ένας τύπος εξελικτικής τεχνικής υπολογισμού, ο οποίος χρησιμοποιήθηκε σε ένα ευρύ σύνολο από τομείς με επιτυχία.

Τα αποτελέσματα της μοντελοποίησης των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγόριθμων, καθώς και τα συμπεράσματα και η εμπειρία της χρήσης αυτών των μεθόδων αναφέρονται στην συνέχεια.

5.3 Παρουσίαση Αποτελεσμάτων

Σκοπός είναι η παρουσίαση των αποτελεσμάτων και η αναγνώριση των βέλτιστων αποτελεσμάτων που σημειώθηκαν με την μέθοδο της χρήσης του εργαλείου NeuroShell 2, αλλά και με το προτεινόμενο εργαλείο διεξαγωγής πειραμάτων που υλοποιήθηκες στη MATLAB®.

Αρχικά, στον πίνακα 5.1 παρουσιάζονται συγκεντρωμένα τα αποτελέσματα για το κάθε ένα σύνολο δεδομένων (DATASET), μαζί με την περιγραφή της τοπολογίας των Τεχνητών Νευρωνικών Δικτύων (NN), τον τρόπο εξαγωγής των δεδομένων εισόδων από τα σύνολα δεδομένων (IN) και ο στόχος (OUT) για τον οποίο γίνεται η πρόβλεψη. Ακόμα, φαίνεται ο αριθμός των έργων (Data) που χρησιμοποιείται για εκπαίδευση, για επικύρωση και για δοκιμή.

Τα κριτήρια αξιολόγησης που λήφθηκαν υπόψιν είναι η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος (Normalized RMSE), ο Συντελεστής Συσχέτισης (CC, Correlation Coefficient) και το Σχετικό Μέσο Σφάλμα (MRE, Mean Relative Error). Ο συνδυασμός αυτών των κριτηρίων αξιολόγησης διασφαλίζει ότι η αξιολόγηση του μοντέλου θα είναι έγκυρη και ορθή. Αρχικά η Κανονικοποιημένη Ρίζα Μέσου Τετραγωνικού Σφάλματος αξιολογεί και ανιχνεύει την ποιότητα των προβλέψεων, η οποία για να είμαστε εκ του ασφαλούς συνδυάζεται με άλλα μέτρα σφάλματος, όπως είναι ο Συντελεστής Συσχέτισης, ο οποίος, εκτός του ότι μας επιτρέπει να παρακολουθήσουμε την εξέλιξη της τροχιάς των σημείων των πραγματικών τιμών και των προβλεπόμενων, όσον αφορά την συσχέτισή τους, αποτρέπει την πιθανόν λανθασμένη μας αξιολόγηση να εκλάβουμε τις προβλέψεις που κάνει το μοντέλο και είναι κοντά στο μέσο όρο σαν καλές προβλέψεις. Αν για παράδειγμα παραχθούν προβλέψεις με μικρή απόκλιση από τις πραγματικές τιμές, αλλά στην ουσία οι τιμές της πρόβλεψης σχηματίζουν μια ευθεία γραμμή κοντά στη μέση τιμή των αυθεντικών, πραγματικών δειγμάτων, ο Συντελεστής Συσχέτισης θα δείξει φτωχά αποτελέσματα σηματοδοτώντας ότι η μέθοδος πρόβλεψης κάνει μια προσέγγιση στην μέση τιμή, άρα και τίποτα αξιόλογο. Τέλος, για την αξιολόγηση των αποτελεσμάτων του μοντέλου χρησιμοποιήθηκε και ένα τρίτο κριτήριο αξιολόγησης το Σχετικό Μέσο Σφάλμα (Mean Relative Error), το οποίο δίνει το σφάλμα των

προβλέψεων εκφρασμένο ως την απόλυτη τιμή της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές.

Πίνακας 5.1: Βέλτιστα Αποτελέσματα με το εργαλείο NeuroShell 2.

					TRAINING			**TESTING**		
DATASET	**NN**	**IN**	**OUT**	**Data**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**	**Normalized RMSE error**	**Correlation coefficient**	**MRE error**
COCOMO	**2-9-9-9-1**	**LOC (2)**	**EFF**	**45-5-12**	**0.71759**	**0.69648**	**0.87066**	**0.9627**	**0.44737**	**0.5254**
COCOMO	5-6-6-6-1	LOC (5)	EFF	45-5-9	0.57749	0.82066	0.83338	0.95258	0.4972	0.74621
COCOMO	5-20-20-20-1	LOC (5)	EFF	45-5-9	0.58427	0.83076	1.2695	0.99617	0.63815	0.80458
COCOMO	3-15-15-15-1	LOC(2), EFF(1)	EFF (2)	46-5-10	0.71569	0.70877	0.92923	0.9522	0.40724	0.55149
COCOMO	3-20-20-20-1	LOC(2), EFF(1)	EFF (2)	46-5-10	0.72121	0.7053	0.70128	0.98873	0.4354	0.5229
COCOMO	5-15-15-15-1	LOC(3), EFF(2)	EFF (3)	45-5-10	0.75723	0.65835	1.1601	0.97982	0.19506	0.56702
KEMERER	1-3-3-3-1	LOC (1)	EFF (1)	9-3-3	0.76079	0.75026	0.9635	0.63212	0.87793	0.48763
KEMERER	1-6-6-6-1	LOC (1)	EFF (1)	9-3-3	0.82501	0.77251	0.91535	0.70659	0.87961	0.4585
KEMERER	1-9-9-9-1	LOC (1)	EFF (1)	9-3-3	0.69446	0.76166	0.53244	0.77448	0.87848	0.24051
KEMERER	1-15-15-15-1	LOC (1)	EFF (1)	9-3-3	0.77384	0.75904	0.49384	0.83044	0.87824	0.25624
KEMERER	4-20-20-20-1	LOC(2), EFF(2)	EFF (3)	7-3-3	0.86013	0.5222	1.4207	0.74941	0.66681	0.5694
KEMERER	5-3-3-3-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.58434	0.89168	1.2118	0.632	0.9991	0.50663
KEMERER	5-6-6-6-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.73439	0.75875	1.3746	0.73792	0.98486	0.5988
KEMERER	5-9-9-9-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.67789	0.8142	1.0555	0.58642	0.93271	0.47352
KEMERER	5-15-15-15-1	LOC(3), EFF(2)	EFF(3)	7-3-3	0.48461	0.90991	0.88118	0.47411	0.987	0.27859
KEMERER	**5-20-20-20-1**	**LOC(3), EFF(2)**	**EFF(3)**	**7-3-3**	**0.38442**	**0.93928**	**0.75894**	**0.5027**	**0.98801**	**0.23201**
COCOMO &KEMERER	8-20-20-20-1	LOC(4), EFF(4)	EFF(5)	50-13-11	0.78133	0.62643	5.038	0.94758	0.43199	0.95111
DESHARNAIS	4-9-9-9-1	FP(2), EFF(2)	EFF(3)	53-5-21	0.35458	0.93454	0.29805	0.24746	0.96996	0.48074
DESHARNAIS	**6-9-9-9-1**	**FP(3), EFF(3)**	**EFF(4)**	**52-5-21**	**0.041726**	**0.99919**	**0.031155**	**0.032424**	**0.99951**	**0.051066**

Οι προβλέψεις της μεθόδου είναι αρκετά ικανοποιητικές από ότι φαίνεται στις τιμές που εξάγονται στα σφάλματα. Στα πιο βέλτιστα αποτελέσματα που επιλέχτηκαν και παρουσιάστηκαν στον πίνακα 5.1, μπορούμε να διακρίνουμε τις ελάχιστες τιμές σφαλμάτων που εμφανίστηκαν για το κάθε ένα σύνολο δεδομένων και οι οποίες παρουσιάσουν την μεγαλύτερη επιτυχία πρόβλεψης.

Το ελάχιστο σφάλμα στο σύνολο των αποτελεσμάτων με το COCOMO που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.9522, στο CC ίσο με 0.63815 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE

παρουσιάζεται (0.5229) σε ποσοστό 52,29% η απόκλιση, δηλαδή 47,71% η σύγκλιση. Παρατηρούμε ότι η μέθοδος δίνει σαν καλύτερα αποτελέσματα πειράματος, αυτά στα οποία χρησιμοποιήθηκε η αρχιτεκτονική του Τεχνητού Νευρωνικού Δικτύου 2-9-9-9-1 αφού παρουσιάστηκε μια σχετικά ικανοποιητική ακρίβεια στην πρόβλεψη, που ξεπερνά την ακρίβεια στις τιμές των προβλέψεων που έγιναν από άλλες μελέτες. Πιο συγκεκριμένα, το καλύτερο αποτέλεσμα για το COCOMO είναι το Normalized RMSE = 0,9627, ο Συντελεστής Συσχέτισης CC = 0,44737 και ένα αρκετά μειωμένο MRE = 0,5254 που σημαίνει 52,54% απόκλιση στις τιμές. Συγκριτικά με άλλες μελέτες που χρησιμοποίησαν το ίδιο σύνολο δεδομένων, όπως με την μελέτη των Ali Idri, Taghi M. Khoshgoftaar, Alain Abran, "Can Neural Networks be easily Interpreted in Software Cost Estimation?", 2002 [5], όπου το MRE εμφανίζεται ίσο με 2,03 και με 0,84 παρατηρούμε ότι έχουμε εξαγάγει καλύτερα αποτελέσματα. Επίσης, σε σχέση με την μελέτη των Javier Dolado και Luis Fernandez, "Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation", 1998 [12] όπου το MRE εμφανίζεται ίσο με 5.85 και βελτιώνεται στην συνέχεια με Γενετικό Προγραμματισμό και εμφανίζεται ίσο με 3.22, παρατηρούμε ότι και πάλι έχουμε εξαγάγει πολύ καλύτερα αποτελέσματα.

Το ελάχιστο σφάλμα στο σύνολο των αποτελεσμάτων με το Kemerer που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.47411, στο CC ίσο με 0.9991 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE παρουσιάζεται (0.23201) σε ποσοστό 23,20% η απόκλιση, δηλαδή 76,80% η σύγκλιση. Παρατηρούμε ότι η μέθοδος δίνει σαν καλύτερα αποτελέσματα πειράματος, αυτά στα οποία χρησιμοποιήθηκε η αρχιτεκτονική του Τεχνητού Νευρωνικού Δικτύου 5-20-20-20-1 αφού παρουσιάστηκε μια σχετικά ικανοποιητική ακρίβεια στην πρόβλεψη, που ξεπερνά την ακρίβεια στις τιμές των προβλέψεων που έγιναν από άλλες μελέτες. Πιο συγκεκριμένα, το καλύτερο αποτέλεσμα για το Kemerer είναι το Normalized RMSE = 0,5027, ο Συντελεστής Συσχέτισης CC = 0,98801 και ένα πολύ μειωμένο MRE = 0,23201 που σημαίνει 23,20% απόκλιση στις τιμές, δηλαδή 76,80% σύγκλιση στις τιμές, μια πραγματικά εντυπωσιακή επιτυχία στην ακρίβεια της πρόβλεψης. Αυτό φαίνεται και από το γεγονός ότι συγκριτικά με την μελέτη μελέτης των Javier Dolado και Luis Fernandez, "Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation", 1998 [12] όπου χρησιμοποιήθηκε το ίδιο σύνολο δεδομένων παρουσιάστηκε MRE=2.123 και βελτιστοποιήθηκε με Γενετικό

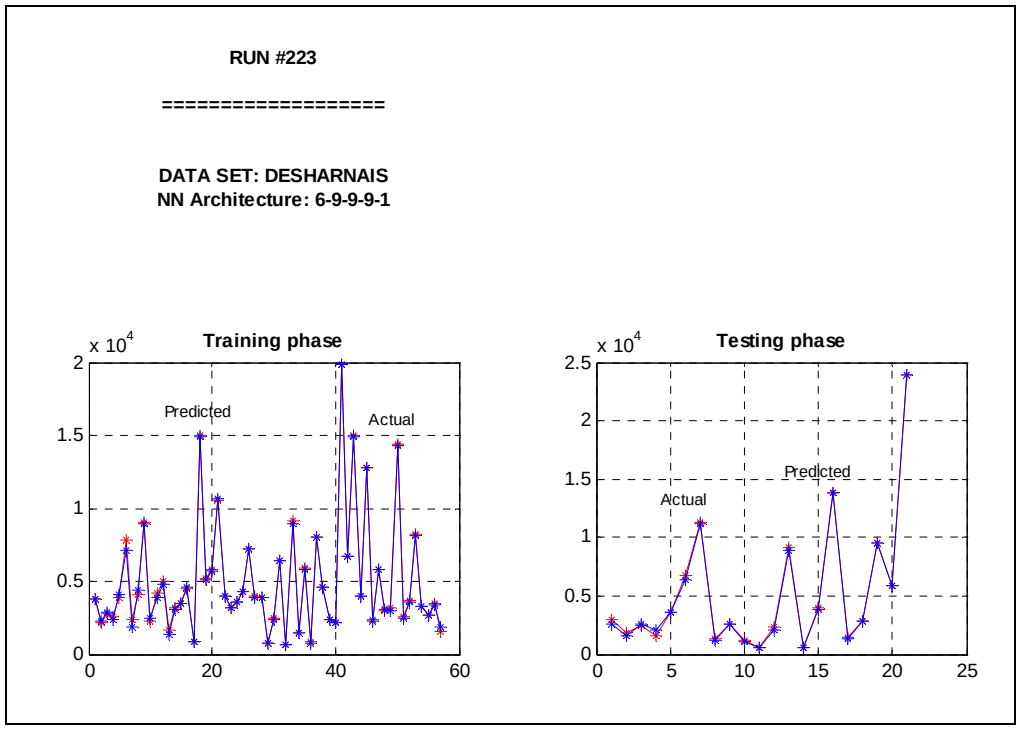
Προγραμματισμό και έφτασε MRE=0.684 τιμές που οπωσδήποτε είναι χειρότερες από τα αποτελέσματα της διπλωματικής αυτής εργασίας. Παρατηρούμε, πως αν και τα αποτελέσματα του συνόλου Kemerer υποσκιάζονται από το γεγονός της ύπαρξης λίγων έργων, παρουσιάζουν εντυπωσιακή ακρίβεια.

Το ελάχιστο σφάλμα στο σύνολο των αποτελεσμάτων με τον συνδυασμό των δεδομένων COCOMO και Kemerer που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.94758, στο CC ίσο με 0.43199 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE παρουσιάζεται (0.95111) σε ποσοστό 95,11% η απόκλιση, δηλαδή η 4,89% σύγκλιση. Οι τιμές των σφαλμάτων εμφανίζονται σε σχέση με τα αποτελέσματα των άλλων συνόλων πολύ ψηλές και έτσι δεν μπορεί να θεωρηθεί αυτό το αποτέλεσμα σαν ικανοποιητικό. Αυτό οφείλεται στην ανομοιογένεια των δεδομένων και αν συγκριθεί με τα ανάλογα αποτελέσματα της χρήσης αυτού του συνδυασμού συνόλων δεδομένων που εμφανίζονται στην μελέτη των Srinivasan και Fisher όπου το MRE εμφανίζεται να είναι ίσο με 0.7, παρατηρούμε ότι έχουμε πετύχει χειρότερα αποτελέσματα.

Στο σύνολο Albrecht δεν εμφανίζονται αποτελέσματα που να θεωρούνται καλά ή έστω ικανοποιητικά. Δεν υπάρχει κανένα πείραμα που να εξαγάγει αξιόλογη τιμή σφαλμάτων και τα αποτελέσματα είναι χειρότερα από τις εμπειρικές μελέτες των Javier Dolado και Luis Fernandez, “Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation”, 1998 [12] όπου χρησιμοποιήθηκε το ίδιο σύνολο δεδομένων και παρουσιάστηκαν MRE=2.123 και MRE=0.684.

Στο σύνολο Desharnais, παρατηρούμε ότι η μέθοδος δίνει σαν καλύτερα αποτελέσματα πειράματος, αυτά στα οποία χρησιμοποιήθηκε η αρχιτεκτονική του Τεχνητού Νευρωνικού Δικτύου 6-9-9-9-1 αφού παρουσιάστηκε μια πολύ εντυπωσιακή ακρίβεια στην πρόβλεψη, που ξεπερνά κατά πολύ την ακρίβεια στις τιμές των προβλέψεων που έγιναν από άλλες μελέτες. Πιο συγκεκριμένα, το καλύτερο αποτέλεσμα για το Desharnais είναι το Normalized RMSE = 0.032424, ο Συντελεστής Συσχέτισης CC = 0.99951 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και ένα πολύ μειωμένο MRE = 0.051066 που σημαίνει 0,5% απόκλιση στις τιμές, δηλαδή 99,5% σύγκλιση στις τιμές, μια πραγματικά εντυπωσιακή επιτυχία στην ακρίβεια της πρόβλεψης. Σε σύγκριση με άλλες μελέτες που χρησιμοποίησαν το ίδιο σύνολο δεδομένων όπως η μελέτη S.G. MacDonell, A.R. Gray, “A Comparison of Modeling Techniques for Software Development Effort Prediction”, 1997 [17] έδωσαν

αποτελέσματα από τρία πειράματα με σφάλμα CC=0.8896, σφάλμα MRE=0,2968 και σφάλμα CC= 0,7745, σφάλμα MRE=0.4586 και σφάλμα CC=0,7379, σφάλμα MRE=0.4350 αποτελέσματα πολύ καλά, αλλά χειρότερα από αυτά της διπλωματικής αυτής εργασίας, που εμφανίζονται με εντυπωσιακή ακρίβεια. Ακόμα, αν και το σφάλμα της μελέτης Wittig και Finnie είναι ίσο με MRE=0,27 και πάλι δεν είναι τόσο καλό όσο το αποτέλεσμα του MRE = 0,051066 που πετύχαμε.



Γραφική Παράσταση 5.1

Από το σύνολο των αποτελεσμάτων στον πίνακα 5.1, για κάθε ένα σύνολο δεδομένων, βρέθηκαν τα βέλτιστα, τα πιο μειωμένα σφάλματα και εμφανίζονται με έντονο χαρακτήρα. Η επιλεγμένη αυτή σειρά παρουσιάζει την βέλτιστη αρχιτεκτονική Τεχνητού Νευρωνικού Δικτύου που παρουσιάζει το μεγαλύτερο ποσοστό ακρίβειας στην πρόβλεψη κατά την δοκιμή. Παρατηρούμε ότι τα αποτελέσματα που λαμβάνονται με την χρήση του συνόλου δεδομένων Kemerer και Desharnais είναι μεγάλης ακρίβειας, ενώ μέτρια είναι τα αποτελέσματα του συνόλου δεδομένων COCOMO. Ενώ τα αποτελέσματα που παρατηρήθηκαν από το Albrecht και τον συνδυασμό των συνόλων COCOMO και Kemerer, δεν είναι καθόλου ικανοποιητικά. Η ακρίβεια της πρόβλεψης διακρίνεται εύκολα και από την Γραφική Παράσταση 5.1, στην οποία με κόκκινο χρώμα εμφανίζεται η πραγματική τιμή του κόστους, ενώ με μπλε χρώμα αντιπροσωπεύεται η πρόβλεψη του κόστους ανάπτυξης λογισμικού. Παρατηρούμε, ότι

η πρόβλεψη προσεγγίζει με μεγάλη ακρίβεια το κόστος, αφού η γραφική παράσταση των πραγματικών τιμών (κόκκινο χρώμα) ταυτίζεται με αυτή των προβλέψεων (μπλε χρώμα) και αυτό αποδεικνύει σαν συμπέρασμα την ικανότητά της μεθόδου να γενικεύει και να λύνει το πρόβλημα του υπολογισμού του κόστους ανάπτυξης λογισμικού και γενικά άλλα προβλήματα πρόβλεψης μεγάλης πολυπλοκότητας.

Ακόμα από το σύνολο των αποτελεσμάτων που εμφανίστηκαν στους πίνακες 4.2.1 μέχρι 4.6.4 συγκεντρώθηκαν στον πίνακα 5.2 τα χαμηλότερα μεμονωμένα σφάλματα που εμφανίστηκαν. Τα αποτελέσματα αφού είναι μεμονωμένες τιμές αν και είναι αρκετά ικανοποιητικά, δεν προέρχονται από την δοκιμή μιας αρχιτεκτονικής δικτύου, αλλά παρουσιάζονται στο σύνολο των αποτελεσμάτων.

Πίνακας 5.2: Μεμονωμένα Καλύτερα Αποτελέσματα με το εργαλείο NeuroShell 2.

Data Set	**Normalized RMSE**	**Correlation Coefficient**	**MRE**
COCOMO	0,83376	0.97461	0.5229
Kemerer	0,47411	0,9991	0,23201
COCOMO & Kemerer	0,96815	0,95173	0,76817
Albrecht	0,4149	0,96814	Κανένα
Desharnais	0,24746	0,96996	0,0510

Στην συνέχεια, έχοντας σαν θεμέλιο λίθο αυτής της έρευνας τα αποτελέσματα του εργαλείου NeuroShell 2, που περιγράφηκαν μπορέσαμε να καθορίσουμε το πλαίσιο στο οποίο ψάχνουμε να πετύχουμε ικανοποιητικά αποτελέσματα με την υλοποίηση του εργαλείου διεξαγωγής πειραμάτων. Το εργαλείο που υλοποιήθηκε και προτείνεται, συνδυάζει την τεχνική των Νευρωνικών Δικτύων και του Γενετικού Προγραμματισμού, αφού όπως φάνηκε και από την έρευνα που προηγήθηκε, ένα δίκτυο για να μπορέσει να επιφέρει ικανοποιητικά αποτελέσματα πρέπει να εκπαιδευτεί πλήρως και χρησιμοποιώντας κατάλληλες παραμέτρους, όπως για παράδειγμα είναι μια πολύ σημαντική παράμετρος, η τοπολογία του δικτύου να βελτιστοποιήσει τα αποτελέσματα. Έτσι, αντί να δοκιμάζονται ιδιοχείρως συνδυασμοί τοπολογιών δικτύων μέχρι να βρεθεί η καλύτερη για κάθε σύνολο δεδομένων, χρησιμοποιείται Γενετικός Αλγόριθμος με τον οποίο δοκιμάζονται συνδυασμοί, διασταυρώσεις και μεταλλάξεις προτεινόμενων αρχιτεκτονικών. Αυτές στην συνέχεια αξιολογούνται, εντοπίζεται η κατάλληλη και παρουσιάζεται τελικά στον χρήστη η καλύτερη δυνατή αρχιτεκτονική, με τις καλύτερες δυνατές προβλέψεις. Αυτό είναι μια πολύ σημαντική πληροφορία

αφού αν ο χρήστης γνωρίζει για κάθε σύνολο δεδομένων την κατάλληλη αρχιτεκτονική Τεχνητού Νευρωνικού Δικτύου, μπορεί να την χρησιμοποιήσει και να λάβει τα καλύτερα δυνατά αποτελέσματα πρόβλεψης του κόστους ανάπτυξης λογισμικού.

Στην συνέχεια, παραθέτω τον πίνακα 5.3 ο οποίος περιλαμβάνει τα αποτελέσματα του Γενετικού Αλγόριθμου. Τα αποτελέσματα που πήραμε από τον Γενετικό Αλγόριθμο, δεν είναι όπως περιμέναμε. Εντούτοις, οι βέλτιστες αρχιτεκτονικές των Τεχνητών Νευρωνικών Δικτύων παρουσιάζουν μεγάλη βελτίωση στις προβλέψεις, κατά την εκπαίδευση ενώ τα αποτελέσματα δεν φαίνονται ιδιαίτερα βελτιωμένα στην δοκιμή του δικτύου, από όταν απλά χρησιμοποιήθηκαν συνδυασμοί τοπολογιών με το εργαλείο πειραμάτων NeuroShell και το απλό εργαλείο δημιουργίας Νευρωνικών Δικτύων.

Πίνακας 5.3: Αποτελέσματα Καλύτερων Αρχιτεκτονικών Γενετικού Αλγόριθμου

							TRAINING			**TESTING**		
DATA SET	**Inputs**	**Hidden Layer 1**	**Hidden Layer 2**	**Hidden Layer 3**	**IN**	**OUT**	**Normalized RMSE**	**CC**	**MRE**	**Norm. RMSE**	**CC**	**MRE**
COCOMO	2	16	2	16	LOC(i)	EFF(i)	0.106203	0.9942	15.7723	1.0662	0.63566	0.68096
COCOMO	14	11	30	22	LOC(i), EFF(i)	EFF (i+1)	0.106904	0.9941	12.0352	1.0502	0.76449	0.1199
COCOMO	7	19	14	6	LOC(i+ 1)EFF(i)	EFF (i+1)	0.095399	0.9953	7.0856	1.0176	0.70774	0.98053
Kemerer	2	20	14	2	LOC(i)	EFF(i)	1.8165E-005	1	6.6445E-005	1.5212	-0.5515	0.57187
Kemerer	2	11	4	5	LOC(i), EFF(i)	EFF (i+1)	1.735E-006	1	4.5565E-006	1.5932	-0.49985	0.47437
Kemerer	5	6	19	6	LOC(i+ 1)EFF(i)	EFF (i+1)	7.0407E-007	1	3.4593E-006	1.5212	-0.55149	0.57187
Albrecht	6	20	6	11	FP(i)	EFF(i)	1.3176E-005	1	8.4427E-006	1.1089	0.14121	0.08327
Albrecht	6	3	7	3	FP(i), EFF(i)	EFF (i+1)	0.0038582	0.9999	0.0037902	1.075	0.060513	0.11283
Albrecht	1	0	2	5	FP(i+1), EFF(i)	EFF (i+1)	3.7912E-005	1	1.8588E-005	1.1089	0.14121	0.08327
Desharnais	1	19	5	30	FP(i)	EFF(i)	0.64078	0.7627	1.1457	1.6556	-0.63428	0.21095
Desharnais	4	28	27	9	FP(i), EFF(i)	EFF (i+1)	0.48641	0.8716	0.65685	1.9934	-0.3892	0.30318
Desharnais	11	23	27	17	FP(i+1), EFF(i)	EFF (i+1)	0.40714	0.9116	0.91359	1.5909	-0.2467	0.13393

Παρατηρούμε ότι έχουν γίνει κάποιες μεμονωμένες βελτιώσεις στα σύνολα δεδομένων COCOMO και Albrecht. Υπάρχει βελτίωση στο COCOMO στο σφάλμα του Συντελεστή Συσχέτισης, ο οποίος αυξήθηκε από 0,44737 που είχαμε στο καλύτερο πείραμα και έγινε 0.76449, δηλαδή πλησίασε την μονάδα δείχνοντας μεγαλύτερη ικανότητα πιστής ακολουθίας της τάσης των αποτελεσμάτων. Επίσης, στο COCOMO παρατηρείται και μια γενική βελτίωση στο σφάλμα του MRE αφού από 52,54%

απόκλιση στις τιμές παρουσιάζεται τώρα 11,99% μια σημαντική βελτίωση στις τιμές αφού μεταφράζεται σε 88% σύγκλιση ή επιτυχία στην πρόβλεψη των τιμών. Η βελτίωση όμως δεν εμφανίζεται σε όλα τα σφάλματα, αλλά το Normalized RMSE σφάλμα εμφανίζεται πολύ υψηλό, σε σημείο που δεν είναι αποδεκτό.

Ακόμα, μια σημαντική βελτίωση της μεθόδου των Γενετικών Αλγορίθμων φαίνεται και από το γεγονός ότι τώρα τα αποτελέσματα από το σύνολο δεδομένων Albrecht εμφανίζονται σαν πολύ επιτυχημένα αποτελέσματα, ενώ πριν δεν συμπεριλαμβάνονταν καν σαν ενδεικτικά βέλτιστα αποτελέσματα στον πίνακα 5.1. Γενικά παρατηρείται μια βελτίωση σε όλα τα σφάλματα αφού πέφτουν σε πολύ χαμηλές τιμές και να δίνουν ικανοποιητικές μετρήσεις, με εξαίρεση το Normalized RMSE το οποίο και πάλι εμφανίζεται μεγαλύτερο από την μονάδα

Γενικά με την εφαρμογή Γενετικού Αλγόριθμου στην τεχνική, παρατηρήσαμε ότι τα αποτελέσματα δεν αποφέρανε τις επιθυμητές αλλαγές. Εντούτοις, έχουν επιφέρει κάποιες μεμονωμένες βελτιώσεις στα περισσότερα σφάλματα, όμως έχουν χειροτερέψει κάποια άλλα. Η μέθοδος μπορεί να θεωρηθεί σαν μια προσπάθεια βελτιστοποίησης, η οποία ίσως με περισσότερη επεξεργασία των εισόδων και των παραμέτρων να έδινε ακόμα καλύτερα αποτελέσματα. Υπάρχει σίγουρα χώρος για βελτιστοποίηση της μεθόδου εφαρμογής Γενετικού Αλγόριθμου στο μοντέλο, έτσι ώστε να προσφέρει σημαντικές βελτιστοποιήσεις ποιοτικά, ποσοτικά αλλά και χρονικά στα αποτελέσματα.

5.4 Αλγόριθμος Αύξηση-Μείωση

Στόχος είναι η παρουσίαση, η ανάλυση και αξιολόγηση των αποτελεσμάτων των προβλέψεων με χρήση τον αλγόριθμο αύξηση - μείωση. Ο αλγόριθμος αυτός λαμβάνει υπόψιν την τάση προς τα πάνω ή προς τα κάτω των αποτελεσμάτων της πρόβλεψης σε σχέση με τις πραγματικές τιμές του κόστους ανάπτυξης λογισμικού.

Τα αποτελέσματα που διεξήχθησαν από τα πειράματα μπορούν να θεωρηθούν ως ιδιαίτερα χρήσιμα όταν αξιολογηθούν με βάση τον αλγόριθμο αυτό. Στην περίπτωση που ο χρήστης έχει στη διάθεσή του τα αριθμητικά αποτελέσματα από έργα που έχει διεκπεραιώσει στο παρελθόν, μπορεί να χρησιμοποιήσει την μέθοδο του αλγόριθμου και πριν την έναρξη ενός έργου ανάπτυξης λογισμικού, να μπορέσει να προβλέψει με σημαντική ακρίβεια εάν η προσπάθεια που θα καταβληθεί για το νέο

έργο, θα μειωθεί ή θα αυξηθεί σε σχέση με τα προηγούμενα. Αποτελεί μια ακόμα μέθοδο αξιολόγησης της πρόβλεψης που παρουσιάστηκε στα αποτελέσματα.

Η προσφορά μιας τέτοιας γνώσης, πριν το έργο αρχίσει είναι ιδιαίτερα σημαντική, καθώς μπορεί να βοηθήσει σημαντικά στο επίπεδο της διαχείρισης ενός οργανισμού ή μιας επιχείρησης, αφού οι λανθασμένοι υπολογισμοί και οι ανακρίβειες στον υπολογισμό του κόστους και του χρόνο-προγραμματισμού είχαν οδηγήσει, επανειλημμένα, έργα στην καταστροφή. Ειδικότερα, η παραγωγή εντός χρόνου και εντός χρηματικού κόστους, είναι κρίσιμη για την βιωσιμότητα οργανισμών που αναλαμβάνουν την διεκπεραίωση ενός έργου και έχουν άμεση επίδραση στην φήμη τους, στην ανταγωνιστικότητά τους και στην απόδοσή τους.

Πιο συγκεκριμένα, ο αλγόριθμος αυτός συγκρίνει τα αποτελέσματα των πειραμάτων του εργαλείου NeuroShell 2, των οποίων οι λεπτομερείς τιμές των αποτελεσμάτων φαίνονται στο Παράρτημα Α, λαμβάνοντας υπόψιν μόνο τις τάσεις που ακολουθούν τα πραγματικά και τα προβλεπόμενα αποτελέσματα του δικτύου. Με αυτό τον τρόπο μπορούμε να διεξάγουμε αποτελέσματα και παρατηρήσεις που συσχετίζονται με το γεγονός αν αυξάνεται ή αν μειώνεται η προσπάθεια ανάπτυξης ενός έργου σε σχέση με το αμέσως προηγούμενο του. Αν συγκρίνουμε αυτά τα αποτελέσματα των πραγματικών έργων και της πρόβλεψης, τότε παίρνουμε τα πρόσημα της αύξησης ή της μείωσης και παρακολουθούμε κατά πόσον τα πρόσημα αυτά συμπίπτουν ή όχι.

Αυτή η διαδικασία πραγματοποιήθηκε για το σύνολο της σειράς των πιο βέλτιστων αποτελεσμάτων που συγκεντρώθηκαν στον πίνακα 5.1 και τα αποτελέσματα του αλγορίθμου αύξησης – μείωσης φαίνονται στον πίνακα 5.4. Τα αποτελέσματα αφορούν το σύνολο δοκιμής, δηλαδή την πρόβλεψη που εξάγει το δίκτυο και με αυτό τον τρόπο μπορούμε να διεξάγουμε αποτελέσματα και παρατηρήσεις που συσχετίζονται με το γεγονός αν αυξάνεται ή αν μειώνεται η προσπάθεια ανάπτυξης ενός έργου σε σχέση με το αμέσως προηγούμενο του στην διαδικασία δοκιμής. Στη στήλη “Projects with the same tendency” εμφανίζονται σε σύνολο ο αριθμός των έργων των οποίων οι τάσεις της πραγματικής προσπάθειας και της προβλεπόμενης προσπάθειας από ένα έργο στο επόμενό του, συμπίπτουν είτε προς τα πάνω, είτε προς τα κάτω σε σχέση με το σύνολο των έργων που λήφθηκαν υπόψιν. Ενώ στη στήλη “Percentage of successful tendency” εμφανίζονται τα ίδια αποτελέσματα σε ποσοστιαία

αναλογία, δηλαδή το σύνολο των έργων επί τοις εκατό, των οποίων η τάση αύξησης ή μείωσης των πραγματικών τιμών με των προβλεπόμενων τιμών συμπίπτουν.

Πίνακας 5.4: Αποτελέσματα Αλγόριθμου Αύξησης Μείωσης

DATASET	NN	IN	OUT	Data	Projects with the same tendency	Percentage of successful tendency
COCOMO	2-9-9-9-1	LOC (2)	EFF	45-5-12	7/12	58,33%
COCOMO	5-6-6-6-1	LOC (5)	EFF	45-5-9	5/9	55,56%
COCOMO	5-20-20-20-1	LOC (5)	EFF	45-5-9	5/9	55,56%
COCOMO	3-15-15-15-1	LOC(2), EFF(1)	EFF (2)	46-5-10	5/10	50%
COCOMO	3-20-20-20-1	LOC(2), EFF(1)	EFF (2)	46-5-10	4/6	66,66%
COCOMO	5-15-15-15-1	LOC(3), EFF(2)	EFF (3)	45-5-10	4/9	44,44%
KEMERER	1-3-3-3-1	LOC (1)	EFF (1)	9-3-3	2/2	100%
KEMERER	1-6-6-6-1	LOC (1)	EFF (1)	9-3-3	2/2	100%
KEMERER	1-9-9-9-1	LOC (1)	EFF	9-3-3	2/3	66,67%
KEMERER	1-15-15-15-1	LOC (1)	EFF	9-3-3	2/3	66,67%
KEMERER	4-15-15-15-1	LOC(2), EFF(2)	EFF (+1)	7-3-3	3/3	100%
KEMERER	4-20-20-20-1	LOC(2), EFF(2)	EFF (+1)	7-3-3	3/3	100%
KEMERER	5-3-3-3-1	LOC(3), EFF(2)	EFF(3)	7-3-3	2/2	100%
KEMERER	5-6-6-6-1	LOC(3), EFF(2)	EFF(3)	7-3-3	2/2	100%
KEMERER	5-9-9-9-1	LOC(3), EFF(2)	EFF(3)	7-3-3	2/2	100%
KEMERER	5-15-15-15-1	LOC(3), EFF(2)	EFF(3)	7-3-3	2/2	100%
KEMERER	5-20-20-20-1	LOC(3), EFF(2)	EFF(3)	7-3-3	2/2	100%
COCOMO&KEMERER	8-20-20-20-1	LOC(4), EFF(4)	EFF(+1)	50-13-11	3/8	37,5%
DESHARNAIS	4-9-9-9-1	FP(2), EFF(2)	EFF(3)	53-5-21	17/20	85%
DESHARNAIS	6-9-9-9-1	FP(3), EFF(3)	EFF(4)	52-5-21	20/20	100%

Τα αποτελέσματα του αλγορίθμου αύξησης δείχνουν ότι στην αριθμητική πρόβλεψη που γίνεται κατά την δοκιμή ενός συνόλου δεδομένων, άγνωστα προς το Νευρωνικό Δίκτυο η επιτυχία της τάσης των τιμών προς τα πάνω είτε προς τα κάτω, είναι πολύ πετυχημένη. Παρατηρούμε ότι μόνο στην περίπτωση του συνδυασμού των δύο συνόλων δεδομένων και σε μερικές περιπτώσεις στο COCOMO το ποσοστό είναι πολύ χαμηλό και άρα δεν γίνονται ικανοποιητικές προβλέψεις, ενώ στο σύνολο δεδομένων Kemerer αλλά και Desharnais παρατηρείται ιδιαίτερα μεγάλη επιτυχία.

Αξιολογώντας την απόδοση του Τεχνητού Νευρωνικού Δικτύου σε κάθε ένα σύνολο δεδομένων ξεχωριστά παρατηρούμε αρχικά ότι στο σύνολο δεδομένων

COCOMO το μέγιστο ποσοστό πρόβλεψης είναι 66,66%, όταν πετυχαίνουμε σωστή πρόβλεψη της τάσης αύξησης ή μείωσης του κόστους σε 4 από τα 6 άγνωστα έργα ανάπτυξης λογισμικού. Συνήθως οι τιμές, όμως επιτυχίας της πρόβλεψης κυμαίνονται στο 55% και άρα έχει ποιότητα ενός μέσου όρου (κάτι που θα μπορούσε να πετύχει ένας naïve αλγόριθμος). Το σύνολο δεδομένων Kemerer παρουσιάζει ιδιαίτερα μεγάλη επιτυχία και ακρίβεια, αφού πολύ συχνά εμφανίζει ποσοστό πρόβλεψης σωστής τάσης αύξησης ή μείωσης 100%. Η επιτυχία αυτή, αν και σημαντική, υποσκιάζεται από το γεγονός ότι τα έργα που δοκιμάζονται είναι πολύ λίγα (2 ή 3) και έτσι δεν μπορούμε να γενικεύσουμε με σιγουριά πως αν χρησιμοποιούσαμε περισσότερα έργα ανάπτυξης λογισμικού θα λαμβάναμε τα ίδια επιτυχημένα αποτελέσματα. Τέλος, η μεγαλύτερη και πιο εντυπωσιακή επιτυχία εμφανίζεται στο σύνολο δεδομένων Desharnais, στην πρώτη περίπτωση όπου, με αρκετά μεγάλο σύνολο έργων για δοκιμή, παρατηρήθηκε επιτυχία πρόβλεψης 85%, με 17 από τα 20 έργα να προβλέπονται με τρόπο επιτυχημένο, η αύξηση ή η μείωση του κόστους ανάλογα, και στην δεύτερη περίπτωση όπου παρατηρήθηκε 100% επιτυχία πρόβλεψης, αφού υπολογίστηκαν με επιτυχία όλα τα έργα που δόθηκαν, δηλαδή και για τα 20 από τα 20 άγνωστα έργα.

Τα αποτελέσματα αυτά σε γενικές γραμμές, δείχνουν ότι η μεθοδολογία που επιλέχτηκε μπορεί να προβλέψει με εξαιρετικά μεγάλη ακρίβεια το κόστος ανάπτυξης λογισμικού σε σχέση πάντα με άλλα έργα ανάπτυξης λογισμικού. Ενώ η ακρίβεια των αποτελεσμάτων είναι απόλυτα συνυφασμένη με την ποιότητα των συνόλων δεδομένων.

5.5 Αλγόριθμος Αφελούς Πρόβλεψης

Στόχος είναι η παρουσίαση και ανάλυση των αποτελεσμάτων με χρήση τον αλγόριθμο αφελούς πρόβλεψης (naive algorithm). Τον αλγόριθμο αυτό τον ακολουθεί ένα δίκτυο το οποίο, αντί να προβλέπει την προσπάθεια με τον τρόπο που προτείνεται στην μελέτη αυτή, με την χρήση Υπολογιστικής Νοημοσύνης συνδυάζοντας Τεχνητά Νευρωνικά Δίκτυα και άλλες μεθόδους, διεξάγει αποτελέσματα με μια μεθοδολογία αρκετά αφελή. Μια τέτοια ανάλυση θα μπορέσει να συγκριθεί με τα αποτελέσματα μας και θα είναι χρήσιμο να φανεί κατά πόσο είναι καλύτερη η μεθοδολογία που προτείναμε από μια αφελή πρόβλεψη. Η μεθοδολογία αυτή παίρνει σαν είσοδο την προσπάθεια που χρειάστηκε για την ανάπτυξη μερικών έργων στο παρελθόν και

υπολογίζει μια μελλοντική προσπάθεια ανάπτυξης ενός άγνωστου έργου ανάπτυξης λογισμικού, με τον τρόπο του υπολογισμού του μέσου όρου.

Στην περίπτωση που ο χρήστης έχει στη διάθεσή του τα αριθμητικά αποτελέσματα από έργα που έχει διεκπεραιώσει στο παρελθόν, μπορεί να υπολογίσει αριθμητικά μια πρόβλεψη για την προσπάθεια που θα χρειάζεται ένα έργο πριν την έναρξη ενός έργου ανάπτυξης λογισμικού. Η διαδικασία αυτή έχει ακολουθηθεί για να εξαγάγουμε τα αποτελέσματα ενός αφελούς αλγόριθμου και στη συνέχεια να τα συγκρίνουμε με τα αποτελέσματα που διεξήχθησαν κατά τα προτεινόμενα πειραματικά τρεξίματα της μελέτης αυτής.

Πιο συγκεκριμένα, στον αλγόριθμο αυτό έχουν χρησιμοποιηθεί οι πραγματικές τιμές των προσπαθειών που υπάρχουν στα σύνολα δεδομένων COCOMO, KEMERER, ALBRECHT και DESHARNAIS. Το αποτέλεσμα του μέσου όρου των τιμών για ένα έργο, γίνεται με χρήση των δύο προηγούμενων τιμών. Για παράδειγμα στη χρονική στιγμή i υπολογίζεται με την χρήση των πραγματικών τιμών της προσπάθειας των έργων στις χρονικές στιγμές i-2 και i-1, ενώ στη συνέχεια υπολογίζουμε τον μέσο όρο χρησιμοποιώντας τις τρεις προηγούμενες τιμές της προσπάθειας.

Τα αποτελέσματα της μεθόδου αφελούς αλγόριθμου και οι τιμές των λαθών, φαίνονται στον πίνακα 5.5. Στον πίνακα αυτό περιγράφονται στις τρεις πρώτες στήλες τα χαρακτηριστικά του κάθε πειράματος που αντιπροσωπεύουν, την στήλη DATASET όπου εμφανίζεται το όνομα του σύνολου δεδομένων που χρησιμοποιήθηκε και στις στήλες IN και OUT εμφανίζονται οι είσοδοι και οι έξοδοι του.

Πίνακας 5.5: Αποτελέσματα Αφελούς Αλγόριθμου

			NAIVE ALGORITHM				
DATASET	**IN**	**OUT**	**Normalized RMSE error**	**Correlation coefficient**	**MSE error**	**Relative MAE error**	**MAE error**
COCOMO	**EFF(i),EFF(i+1)**	**EFF(i+2)**	**0.46716**	**0.8822**	**729332.9309**	**0.67407**	**273.2355**
COCOMO	EFF(i),EFF(i+1),EFF(i+2)	EFF(i+3)	0.61815	0.78275	1292540.0966	1.2746	397.0372
KEMERER	EFF(i),EFF(i+1)	EFF(i+2)	0.75709	0.6194	42497.8924	1.0205	129.1043
KEMERER	EFF(i),EFF(i+1),EFF(i+2)	EFF(i+3)	0.75274	0.62997	44626.3807	0.95892	139.2959
ALBRECHT	EFF(i),EFF(i+1)	EFF(i+2)	0.54905	0.84437	161.7986	1.3617	7.4848
ALBRECHT	EFF(i),EFF(i+1),EFF(i+2)	EFF(i+3)	1.3026	0.30965	317.2692	1.6379	11.5167
DESHARNAIS	**EFF(i),EFF(i+1)**	**EFF(i+2)**	**0.68901**	**0.7268**	**9386596.5625**	**0.78323**	**2236.85**
DESHARNAIS	EFF(i),EFF(i+1),EFF(i+2)	EFF(i+3)	0.74156	0.68384	11010184.2532	0.78638	2380.5063

Τα αποτελέσματα που μπορούμε να κρατήσουμε σαν καλύτερα, για να συγκρίνουμε στη συνέχεια, είναι αυτά που εμφανίζονται με έντονους χαρακτήρες στον πίνακα και περιγράφονται στην συνέχεια. Το ελάχιστο σφάλμα στο COCOMO που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.46716, στο CC ίσο με 0.8822 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE παρουσιάζεται (0.67404) σε ποσοστό 67,40% η απόκλιση, δηλαδή 32,6% η σύγκλιση, που είναι ικανοποιητική πρόβλεψη. Το ελάχιστο σφάλμα στο KEMERER που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.75274, στο CC ίσο με 0.62997 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE παρουσιάζεται (0.95892) σε ποσοστό 95,89% η απόκλιση, δηλαδή 4,11% η σύγκλιση, πρόβλεψη που δεν είναι αξιόλογη. Το σύνολο δεδομένων ALBRECHT δεν παρουσιάζει κάποιο αξιόλογο αποτέλεσμα. Ενώ το ελάχιστο σφάλμα στο DESHARNAIS που εμφανίζεται στο Normalized RMSE είναι ίσο με 0.68901, στο CC ίσο με 0.7268 η πιστότητα των τιμών προς τα πάνω ή προς τα κάτω και στο σφάλμα MRE παρουσιάζεται (0.78323) σε ποσοστό 78,32% η απόκλιση, δηλαδή 21,8% η σύγκλιση.

Τα αποτελέσματα γενικά ενός αφελούς αλγόριθμου γενικά δεν είναι ικανοποιητικά, με εξαίρεση το πρώτο αποτέλεσμα του COCOMO, και παρουσιάζονται χειρότερα σε σχέση με τα βέλτιστα αποτελέσματα της μεθοδολογίας που προτάθηκε, αφού μπορεί να προβλέψει με μεγαλύτερη ακρίβεια το κόστος ανάπτυξης λογισμικού.

5.6 Συμπεράσματα

Τα συμπεράσματα αλλά και η εμπειρία της ανάπτυξης μεθοδολογίας και της μελέτης του θέματος της μοντελοποίησης και πρόβλεψης του κόστους ανάπτυξης λογισμικού με την χρήση Υπολογιστικής Νοημοσύνης, μέσω Τεχνητών Νευρωνικών Δικτύων και Γενετικών Αλγόριθμων, αποδείχτηκε ικανοποιητική και πολύ ενδιαφέρουσα. Μπορέσαμε με σχετικά μεγάλη επιτυχία να προβλέψουμε το κόστος ανάπτυξης λογισμικού με την χρήση του εργαλείου NeuroShell 2. Πιο συγκεκριμένα, αν συγκρίνουμε τα αποτελέσματα που εμφανίστηκαν με αυτά των εμπειρικών μελετών που εμφανίζονται στον πίνακα 3.2 και που χρησιμοποίησαν τα ίδια σύνολα δεδομένων και ίδιες τεχνικές μπορούμε να πούμε, πως συχνά εμφανίζουμε καλύτερα αποτελέσματα και άλλες φορές έχουμε κατά πολύ μεγαλύτερη επιτυχία στις

προβλέψεις. Για το σύνολο δεδομένων COCOMO, η μέθοδος της διπλωματικής εργασία παρουσιάζεται πολύ καλύτερη από τα αποτελέσματα της μελέτης [5] που έγινε το 2002 και της [11] που έγινε το 1998. Για το σύνολο Kemerer τα αποτελέσματα, αν και υποσκιάζονται από τον μικρό αριθμό έργων, είναι πολύ επιτυχημένα με πολύ μικρότερο σφάλμα MRE σε σχέση με τις μελέτες [11] και Srinivasan & Fisher. Ακόμα, τα αποτελέσματα για το σύνολο δεδομένων Albrecht και για τον συνδυασμό COCOMO και Kemerer σε ένα σύνολο δεδομένων δεν εμφανίζονται καθόλου ικανοποιητικά, χωρίς καμιά αξιόλογη τιμή σφαλμάτων και με αποτελέσματα χειρότερα από τις εμπειρικές μελέτες. Τέλος, τα αποτελέσματα του συνόλου δεδομένων Desharnais, σε σχέση και με τις μελέτες [17] και Wittig & Finnie, εμφανίζονται να είναι αρκετά καλύτερα και πετυχαίνουν πολύ μικρά σφάλματα πρόβλεψης και συνεπώς ικανοποιητική επιτυχία.

Η μέθοδος των Τεχνητών Νευρωνικών Δικτύων, ενώ εμφανίζει πολύ ικανοποιητικά αποτελέσματα προσπαθήσαμε να τα βελτιώσουμε με την εφαρμογή Γενετικού Αλγορίθμου και μεταβάλλοντας την τοπολογία του δικτύου. Η μέθοδος έχει επιτύχει σημαντικές βελτιστοποιήσεις σε πολλά από τα σφάλματα ενώ έχει χειροτερέψει κάποια άλλα και έχει δώσει τις καλύτερες αρχιτεκτονικές σε κάθε περίπτωση. Υπάρχει σίγουρα βελτιστοποίησης της μεθόδου έτσι ώστε να προσφέρει ακόμα καλύτερα αποτελέσματα.

Στην συνέχεια, αξιολογήσαμε τα αποτελέσματα των Τεχνητών Νευρωνικών Δικτύων χρησιμοποιώντας τον αλγόριθμο αύξηση - μείωση. Ο αλγόριθμος αυτός λαμβάνει υπόψιν την τάση προς τα πάνω ή προς τα κάτω των αποτελεσμάτων της πρόβλεψης σε σχέση με τις πραγματικές τιμές του κόστους ανάπτυξης λογισμικού και τα αποτελέσματα κρίθηκαν ιδιαίτερα αξιόλογα αφού το σύνολο δεδομένων COCOMO παρουσίασε επιτυχία 66%, το σύνολο Kemerer 100% και το σύνολο Desharnais 100%.

Τέλος, τα αποτελέσματα της μελέτης, συγκρίθηκαν με τα αποτελέσματα ενός αλγόριθμου που διεξάγει πρόβλεψη με αφελή τρόπο. Τα αποτελέσματα με την χρήση Τεχνητά Νευρωνικά Δίκτυα και Γενετικούς Αλγόριθμους παρουσίασε καλύτερα αποτελέσματα και φάνηκε πως έχουν την ικανότητα να λύνουν πολύπλοκα προβλήματα πρόβλεψης σε ικανοποιητικό βαθμό.

Τα αποτελέσματα που λήφθηκαν με την χρήση αυτής της μεθοδολογίας είναι περισσότερο από ενθαρρυντικά αφού παρουσίασαν ιδιαίτερη επιτυχία στις προβλέψεις του κόστους ανάπτυξης λογισμικού σε σχέση με άλλες τεχνικές αλλά και με άλλες

εμπειρικές μελέτες που έγιναν στο παρελθόν. Υπάρχει σίγουρα ανάγκη βελτιστοποίησης, ιδιαίτερα στο θέμα του Γενετικού Αλγόριθμου, έτσι ώστε να εξαχθούν ακόμα πιο εντυπωσιακά και ακριβή αποτελέσματα.

5.7 Προβλήματα

Η έρευνα και η μελέτη των συγκεκριμένων μεθοδολογιών που χρησιμοποιήθηκαν κατά την προσπάθεια της μοντελοποίησης και της πρόβλεψης του κόστους ανάπτυξης λογισμικού, η χρήση δηλαδή των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγόριθμων έχει οδηγήσει συχνά στην ανάγκη αντιμετώπισης διαφόρων προβλημάτων και δυσκολιών, κάποτε ανυπέρβλητων και κάποτε όχι, που παρουσιάστηκαν. Σε αυτόν τον τομέα παραθέτονται τα τυχόν προβλήματα και οι δυσκολίες που παρουσιάστηκαν για τα Τεχνητά Νευρωνικά Δίκτυα και για τους Γενετικούς Αλγόριθμους.

Αρχικά, αξίζει να αναφέρω μερικές ανησυχίες στην δυσκολία εύρεσης του ιδανικού μοντέλου που να προσφέρει ακριβή πρόβλεψη. Το πρόβλημα αυτό οφείλεται στην ύπαρξη πολλών δυσκολιών στην διαδικασία υπολογισμού του κόστους ανάπτυξης λογισμικού, τα οποία αναφέρθηκαν στο δεύτερο κεφάλαιο, κυρίως όμως εντείνεται από τον κύριο παράγοντα δυσκολίας, την έλλειψη και την δυσκολία εύρεσης ποιοτικών δεδομένων. Είναι σημαντικό να αναφερθώ σε ακόμα μερικές επιπλέον ανησυχίες, που είναι η δυσκολία που συναντήθηκε να συμπεριληφθεί στο μοντέλο η έμπειρη γνώση, με σκοπό να μειωθούν οι ελεύθεροι παράμετροι, αλλά και να υπάρξουν σταθερές τιμές σε ανεξάρτητες μεταβλητές που θα ήταν καλό να χρησιμοποιηθούν, έτσι ώστε να εμφανίζει πιο σταθερή απόδοση το μοντέλο. Αξίζει να σημειώσουμε, πως το ρίσκο της επιλογής σταθερών τιμών για τις μεταβλητές αυτές θα μπορούσε να μειωθεί εάν εκφράζονταν οι μεταβλητές αυτές με ασαφή τρόπο, δίνοντας τους πεδία τιμών βασιζόμενοι σε ασαφή λογισμό.

Χρησιμοποιώντας την προσέγγιση υπολογισμού του κόστους ανάπτυξης λογισμικού βασισμένοι σε Τεχνητά Νευρωνικά Δίκτυα και δημιουργώντας μοντέλο υπολογισμού της προσπάθειας, η τεχνική αυτή μεταφέρει μαζί της την αδυναμία της μη ύπαρξης κάποιου μηχανισμού διερμηνείας της αρχιτεκτονικής του δικτύου, καθώς συχνά θεωρούνται σαν «μαύρα κουτιά». Αυτό συμβαίνει γιατί τα Νευρωνικά Δίκτυα δεν είναι εύκολο να κατανοηθούν και επεξηγηθούν, ιδιαίτερα σε ένα πιθανό χρήστη,

και επιπλέον γιατί δεν υπάρχουν συγκεκριμένες οδηγίες ή μέθοδοι για την κατασκευή της τοπολογίας των δικτύων. Ένα σημαντικό μειονέκτημα που εμφανίζουν τα Τεχνητά Νευρωνικά Δίκτυα είναι το γεγονός ότι ο αναλυτής δεν έχει την δυνατότητα να μεταβάλει και να χρησιμοποιήσει αλλιώς τις παραμέτρους του δικτύου, αφότου έχει τελειώσει η διαδικασία της μάθησης.

Τέλος, το πιο σημαντικό μειονέκτημα και πρόβλημα που εμφάνισαν οι Γενετικοί Αλγόριθμοι είναι το γεγονός η επιτυχία βελτιστοποιήσεων περιλαμβάνει πολλούς πειραματισμούς στις παραμέτρους του αλγόριθμου και το γεγονός ότι η εκτέλεση των πειραμάτων και η χρήση τους είναι πολύ χρονοβόρα, αφού χρειάζονται πολλές ώρες για να πάρουμε αποτελέσματα.

Κεφάλαιο 6

Συμπεράσματα – Εισηγήσεις

6.1 Γενικά Συμπεράσματα

Ο υπολογισμός του κόστους ανάπτυξης ενός έργου λογισμικού στον τομέα της Τεχνολογίας Λογισμικού, είναι μια δραστηριότητα δύσκολη, προκλητική αλλά και πολύ σημαντική και ενδιαφέρουσα στο επίπεδο της διαχείρισης ενός οργανισμού ή μιας επιχείρησης. Ειδικότερα, ο υπολογισμός εντός χρόνου και εντός χρηματικού κόστους είναι κρίσιμη για την βιωσιμότητα οργανισμών που αναλαμβάνουν την διεκπεραίωση ενός έργου και έχουν άμεση επίδραση στην φήμη τους, στην ανταγωνιστικότητά τους και στην απόδοσή τους. Ιδιαίτερα οι διαχειριστές (managers) των έργων τονίζουν την σημασία της βελτίωσης του υπολογισμού αυτού, έτσι ώστε να είναι πιο ακριβής, αφού είναι αναγκαίο στοιχείο για την επίτευξη καλύτερης κατανομής των πόρων, καλύτερης παρακολούθησης, διεξαγωγής και επιτυχημένης τελικά παράδοσης των έργων λογισμικού.

Το θέμα της εύρεσης μεθοδολογίας ακριβής πρόβλεψης του κόστους ανάπτυξης λογισμικού είναι ένα ερευνητικό θέμα το οποίο παραμένει ενδιαφέρον και είναι ακόμα ανοικτό. Έχει απασχολήσει ένα μεγάλο κοινό ερευνητών και μέχρι σήμερα κανένα μοντέλο δεν μπόρεσε να προβλέψει το κόστος ανάπτυξης λογισμικού με υψηλή ακρίβεια.

Η μεθοδολογία που μελετήθηκε Τεχνητής Νοημοσύνης, τα Τεχνητά Νευρωνικά Δίκτυα αποδείχτηκε πως έχουν την ικανότητα να αντιμετωπίζουν προβλήματα με μεγάλη πολυπλοκότητα, μπορούν να γενικεύουν, να υιοθετούν λύσεις και με ευελιξία να προβλέπουν με σχετικά ικανοποιητική ακρίβεια. Αυτή η προσέγγιση έχει καλές δυνατότητες και προοπτικές, στο μέλλον να μπορέσει είτε με ενδεχόμενες βελτιώσεις,

είτε σε συνδυασμό και με άλλες μεθόδους να προσφέρει ακόμα μεγαλύτερης ακρίβειας αποτελέσματα. Είναι ξεκάθαρο πως υπάρχει ανάγκη και προοπτική για περισσότερη αναζήτηση, μελέτη και βελτιστοποίηση, ιδιαίτερα στο θέμα της μοντελοποίησης θέμα ανοικτό για περισσότερη διερεύνηση και ανάλυση.

6.2 Εισηγήσεις και Μελλοντική Εργασία

Οι εισηγήσεις που προσφέρονται, αφορούν κυρίως την επίλυση των ήδη αναφερόμενων δυσκολιών που εμφανίστηκαν στην διαδικασία του υπολογισμού του κόστους ανάπτυξης λογισμικού. Υπάρχει ανάγκη της συστηματικής έρευνας και προσέγγισης του υπολογισμού, η διεύρυνση της περιορισμένης γνώση για την πολύπλοκη και δυναμική διαδικασία της ανάπτυξης λογισμικού, καθώς και για τις συνεχώς μεταβαλλόμενες συσχετίσεις των παραγόντων που επηρεάζουν και προκαλούν τις διαφοροποιήσεις στην παραγωγικότητα.

Ένα από τα μεγαλύτερα προβλήματα που αντιμετωπίζουμε είναι η έλλειψη αξιόπιστων συνόλων δεδομένων που να περιλαμβάνουν τιμές του κόστους και των πόρων που ξοδεύονται για να αναπτυχθούν έργα λογισμικού. Οι πληροφορίες αυτές, θα μπορούσαν να συλλεχτούν από περασμένα έργα που χρειάστηκαν τους πόρους αυτούς και το κόστος καταβολής προσπάθειας για την διεξαγωγή των έργων και ακόμα, θα βοηθούσε σημαντικά η εύρεση εκπαιδευμένου προσωπικού που να κάνει τους σωστούς υπολογισμούς και να έχει εμπειρία στη διαδικασία αυτή.

Μια εισήγηση για μελλοντική εργασία είναι η ανάπτυξη και η υποστήριξη της διαδικασίας των Τεχνητών Νευρωνικών Δικτύων και των Γενετικών Αλγόριθμων, με πιθανές βελτιώσεις, με την χρήση περισσότερων μετρικών στην διαδικασία των πειραμάτων από αυτές που χρησιμοποιήθηκαν και με συνδυασμό με ειδικές τεχνικές, μοντέλα ή /και εργαλεία.

Βιβλιογραφία

[1] Ian Sommerville, “Software Engineering”, Addison-Wesley, 2001.

[2] Kartalopoulos, S. V., “Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications”, IEEE Press, 1996.

[3] Russ Eberhart, Pat Simpson, Roy Dobbins, “Computational Intelligence PC Tools”, AP PROFESSIONAL,1996.

[4] Simon Haykin, “Neural Networks: A Comprehensive Foundation”, Prentice Hall, 1999.

[5] Ali Idri, Taghi M. Khoshgoftaar, Alain Abran, “Can Neural Networks be easily Interpreted in Software Cost Estimation?”, 2002.

[6] Carolyn Mair, Gada Kadoda, Martin Lefley, Keith Phalp, Chris Schofield, “Non-Algorithmic Effort Estimation Techniques”,1998.

[7] Desharnais, J.M., “Analyse Statistique de la Productivite des Projects de Development en Informatique a Partir de la Technique de Points de Fonction”, 1988.

[8] Gary D. Boetticher, “An Assessment of Metric Contribution in the Construction of a Neural Network-Based Effort Estimator”.

[9] Gary D. Boetticher, “Using Machine Learning to Predict Project Effort: Empirical Case Studies in Data-Starved Domains”.

[10] Hareton Leung, Zhang Fan, “Software Cost Estimation”.

[11] J. Javier Dolado, “Limits to the Methods in Software Cost Estimation / Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation”.

[12] Javier Dolado and Luis Fernandez, “Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation”, 1998.

[13] K.K. Shukla, “Neuro-genetic prediction of software development effort”, 2000.

[14] Lionel C. Briand, Isabella Wieczorek, “Resource Estimation in Software Engineering”.

[15] Lionel C. Briand, Khaled El Emam, Dagmar Surmann, Isabella Wieczorek, Katrina Maxwell, “An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques”, 1998.

[16] Martin Shepperd, Steve Webster, “An Investigation of Machine Learning Based Prediction Systems”, 1999.

[17] S.G. MacDonell, A.R. Gray, “A Comparison of Modeling Techniques for Software Development Effort Prediction”, 1997.

[18] Stensrud, Tron Foss, Barbara Kitchenham, Ingunn Myrtveit, “An Empirical Validation of the Relationship Between the Magnitude of Relative Error and Project Size”.

[19] Symons R., C., “Function Points Analysis, Difficulties and Improvements”, IEEE Trans. on Soft. Eng., Vol-Se 14, No 16, Jan 1988, pp.2-11.

[20] The Standish Group, CHAOS Chronicles, Standish Group Internal Report, 1995.

[21] http://www.vuse.vanderbilt.edu/~dfisher/tech-reports/raw-TSE-95.

[22] Kemerer, C.F. “An Empirical Validation of Software Cost Estimation Models”, CACM, 30(5), pp416-429, 1987.

[23] Albrecht, A.J. and J.R. Gaffney, 'Software Function Source Lines of Code, and Development Effort Prediction: A Software Science Validation', IEEE trans. on Softw. Eng., 9(6), pp639-648,1983.

Παράρτημα Α

Λεπτομερής Παρουσίαση Ενδεικτικών Αποτελεσμάτων και Γραφικών Παραστάσεων των Πειραμάτων

Α.1. Εισαγωγή

Η αναλυτική και λεπτομερής παρουσίαση των γραφικών παραστάσεων και των τιμών της πραγματικής προσπάθειας και της προβλεπόμενης προσπάθειας που λήφθηκαν κατά τα πειραματικά τρεξίματα με το εργαλείο NeuroShell εμφανίζονται στο παράρτημα αυτό.

Α.2. Λεπτομερής Παρουσίαση Αποτελεσμάτων Πειραμάτων

Σε αυτό το σημείο παραβάλλονται οι πίνακες που εξήγαγε το εργαλείο NeuroShell κατά τα πειραματικά τρεξίματα. Οι στήλες Actual και Predicted αντιπροσωπεύουν τις τιμές των πραγματικών τιμών και των προβλεπόμενων τιμών της προσπάθειας ανάπτυξης λογισμικού, ενώ με την σκιασμένη χρωματική ένδειξη διακρίνονται ποιες τιμές χρησιμοποιήθηκαν σαν σύνολο εκπαίδευσης, επικύρωσης - δοκιμής και δοκιμής - παραγωγής του δικτύου. Ακόμα στον πίνακα παραβάλλονται και άλλες πληροφορίες του συγκεκριμένου πειράματος, όπως ο μοναδικός αριθμός του τρεξίματος, το σύνολο των δεδομένων, η αρχιτεκτονική του δικτύου, από πόσα έργα (αναγράφονται στην παρένθεση) χρησιμοποιήθηκαν οι γραμμές του κώδικα ή τα λειτουργικά σημεία ή / και η προσπάθεια ανάπτυξης των έργων για να προβλεφτεί η τιμή της προσπάθειας που χρειάζεται να καταβληθεί για το επόμενο στη σειρά έργο. Ακόμα παρουσιάζονται οι τιμές των σφαλμάτων κατά την εκπαίδευση και κατά την δοκιμή του δικτύου, αλλά και οι γραφικές παραστάσεις των τιμών της προβλεπόμενης

και της πραγματικής προσπάθειας. Στη συνέχεια παρουσιάζονται τα πειραματικά τρεξίματα με αύξων αριθμό 8, 22, 25, 54, 55, 59, 76, 77, 78, 79, 95, 101, 102, 103, 104, 105, 150, 218 και 223.

Actual	Predicted	Actual	Predicted
1600	1432.856	230	120.4219
243	1070.114	82	61.45673
240	370.8275	55	45.02002
33	107.7467	47	121.7028
43	21.90938	12	86.14023
8	14.93873	8	27.97549
1075	71.08706	8	5.9
423	122.2376	6	7.483142
321	90.89135	45	60.00528
218	80.25208	83	111.1732
201	144.9817	87	131.3917
79	123.1489	106	141.859
60	29.80195	126	240.977
61	5.9	36	150.4404
40	5.9	1272	1590.266
9	5.9	156	992.1109
11400	1590.013	176	209.0458
6600	8116.677	122	53.89842
6400	5103.064	41	27.73843
2455	2004.494	14	15.39121
724	944.3794	20	8.338754
539	455.0402	18	13.47402
453	457.4251	958	87.92457
523	262.699	237	75.79139
387	215.4245	130	93.63604
88	87.06322	70	94.17744
98	43.3314	57	38.19561
7.3	11.43681	50	92.08621
5.9	5.9	38	56.55747
1063	165.738	15	32.60946
702	1333.968		
605	602.7516		

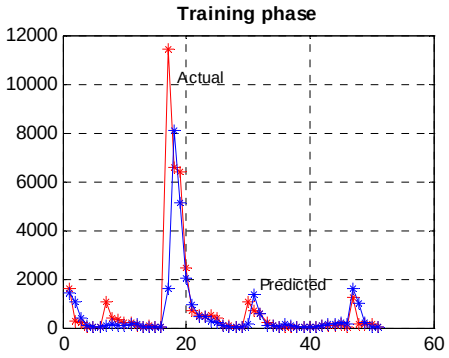
Run #	8
Data Set	COCOMO
NN Architecture	2-9-9-9-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (2)
OUTPUTS	EFF(2)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	45
Num of Test Set	5
Num of Production Set	12

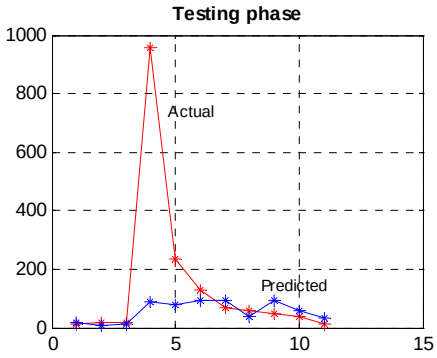
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.71759	0.69648	2091156.8185	0.87066	408.998
TESTING	0.9627	0.44737	65669.9967	0.5254	101.6436

RUN #8

==========

DATA SET: COCOMO
NN Architecture: 2-9-9-9-1





Actual	Predicted	Actual	Predicted
33	93.58966	47	148.1992
43	37.10049	12	57.46402
8	31.87079	8	28.88002
1075	92.96465	8	13.14156
423	128.9244	6	23.89873
321	74.31118	45	84.01208
218	85.94879	83	120.6581
201	161.6538	87	129.953
79	97.01009	106	138.4462
60	13.67978	126	265.2441
61	20.67058	36	88.96185
40	16.46139	1272	3573.971
9	10.13179	156	490.5515
11400	3898.506	176	167.2603
6600	6672.231	122	80.59928
6400	6043.295	41	53.07623
2455	2310.541	14	22.26847
724	775.9415	20	20.11857
539	582.993	18	28.21552
453	437.3615	958	118.7266
523	154.7074	237	59.81353
387	215.5594	130	110.3056
88	32.57274	70	86.37347
98	59.09863	57	27.43769
7.3	8.694773	50	123.3567
5.9	11.61657	38	32.74773
1063	234.1905	15	45.2295
702	2647.361		
605	179.1317		
230	142.8035		
82	77.0842		
55	57.73058		

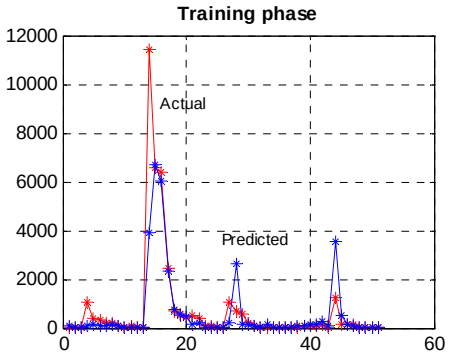
Run #	22
Data Set	COCOMO
NN Architecture	5-6-6-6-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (5)
OUTPUTS	EFF(5)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	45
Num of Test Set	5
Num of Production Set	9

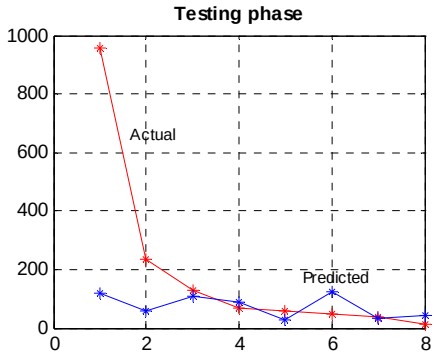
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.57749	0.82066	1357087.7339	0.83338	345.5678
TESTING	0.95258	0.4972	82636.863	0.74621	133.4605

RUN #22

==========

DATA SET: COCOMO
NN Architecture: 5-6-6-6-1





Actual	Predicted	Actual	Predicted
33	69.03414	47	77.38832
43	23.74762	12	50.77076
8	33.48704	8	41.52266
1075	65.18267	8	51.90079
423	71.9067	6	48.77557
321	55.17892	45	68.34927
218	56.98469	83	73.16534
201	80.07082	87	68.53593
79	54.77865	106	68.30354
60	39.14927	126	101.8166
61	49.08086	36	56.03771
40	49.53791	1272	2675.32
9	45.87394	156	539.6918
11400	3452.793	176	199.8219
6600	6252.448	122	130.9917
6400	5804.158	41	30.84995
2455	2495.416	14	33.66245
724	770.037	20	47.32911
539	365.2372	18	54.15182
453	149.0404	958	78.69673
523	36.97963	237	56.77851
387	48.56822	130	62.12494
88	40.74366	70	63.11776
98	28.02908	57	40.20173
7.3	47.79039	50	72.99916
5.9	37.86485	38	48.36214
1063	128.164	15	43.60009
702	1716.89		
605	451.1199		
230	93.72114		
82	118.1062		
55	19.14156		

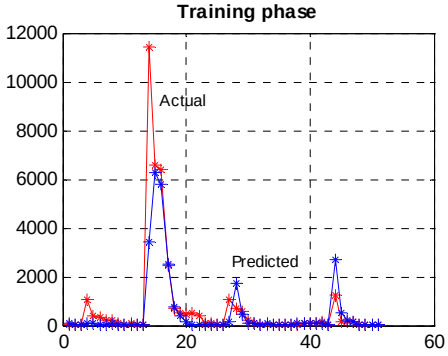
Run #	25
Data Set	COCOMO
NN Architecture	5-20-20-20-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (5)
OUTPUTS	EFF(5)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	45
Num of Test Set	5
Num of Production Set	9

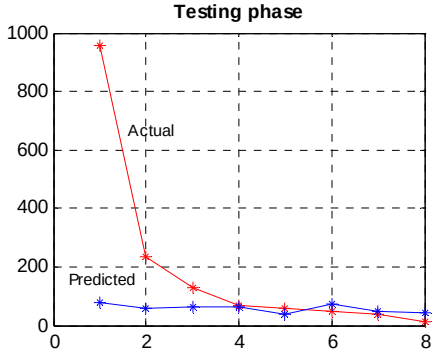
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.58427	0.83076	1389156.1593	1.2695	341.4701
TESTING	0.99617	0.63815	90372.4277	0.80458	138.7993

RUN #25

==========

DATA SET: COCOMO
NN Architecture: 5-20-20-20-1





Actual	Predicted	Actual	Predicted
1600	1366,097	230	139,7787
243	886,1965	82	73,28153
240	255,1812	55	55,84617
33	96,28872	47	125,3928
43	28,84787	12	73,26466
8	33,16848	8	33,92105
1075	80,37421	8	18,21219
423	193,5267	6	25,31039
321	107,2919	45	71,44191
218	100,2947	83	108,3436
201	150,7476	87	121,7782
79	114,8507	106	129,7486
60	32,46408	126	218,3242
61	24,43196	36	117,1565
40	24,26525	1272	1392,348
9	17,26787	156	746,8989
11400	1428,791	176	143,2868
6600	6671,627	122	61,00001
6400	5171,59	41	45,02597
2455	2547,362	14	29,88971
724	913,069	20	24,0978
539	388,2119	18	30,01132
453	403,4297	958	97,08247
523	214,2283	237	136,2349
387	221,6413	130	109,3903
88	86,67548	70	93,84328
98	57,76624	57	42,04061
7,3	25,85423	50	103,1322
5,9	14,80322	38	51,6685
1063	171,859		
702	1242,512		
605	403,413		

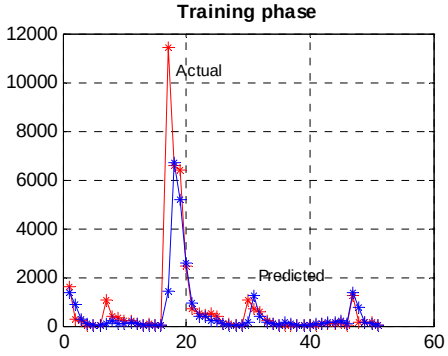
Run #	54
Data Set	COCOMO
NN Architecture	3-15-15-15-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (2),EFF(1)
OUTPUTS	EFF(2)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	46
Num of Test Set	5
Num of Production Set	10

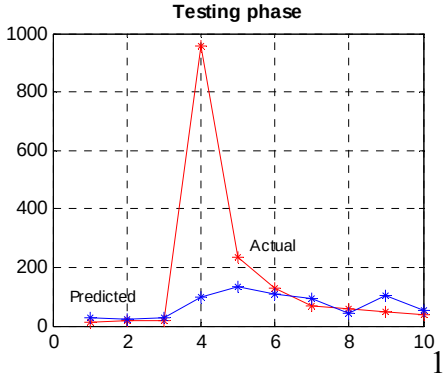
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.71569	0.70877	2044066.0647	0.92923	353.7588
TESTING	0.9522	0.40724	75597.3066	0.55149	111.9895

RUN #54

==========

DATA SET: COCOMO
NN Architecture: 3-15-15-15-1





Actual	Predicted	Actual	Predicted
1600	1260,377	230	102,5576
243	903,855	82	46,61357
240	303,8373	55	31,72201
33	84,49437	47	104,4326
43	8,545362	12	65,08451
8	5,9	8	14,36995
1075	57,4151	8	5,9
423	116,2745	6	5,9
321	74,74105	45	47,27021
218	66,38496	83	92,3654
201	125,7229	87	109,485
79	100,6373	106	118,8113
60	14,58781	126	210,9407
61	5,9	36	120,3208
40	5,9	1272	1426,937
9	5,9	156	821,1965
11400	1443,667	176	166,5479
6600	7041,689	122	38,40328
6400	4742,195	41	16,39899
2455	1973,898	14	5,9
724	835,0433	20	5,9
539	391,7039	18	5,9
453	400,5784	958	73,66621
523	219,5285	237	69,4827
387	190,2146	130	78,85644
88	66,54764	70	75,70904
98	31,07195	57	23,17269
7,3	5,9	50	77,84293
5,9	5,9	38	38,68268
1063	148,5665		
702	1189,639		
605	490,0364		

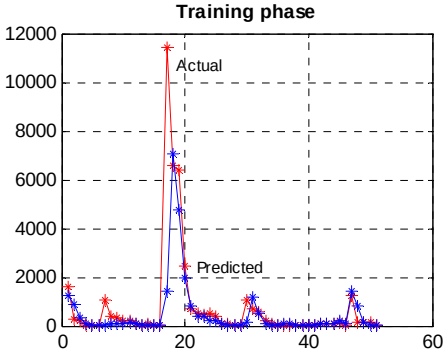
Run #	55
Data Set	COCOMO
NN Architecture	3-20-20-20-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (2),EFF(1)
OUTPUTS	EFF(2)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	46
Num of Test Set	5
Num of Production Set	10

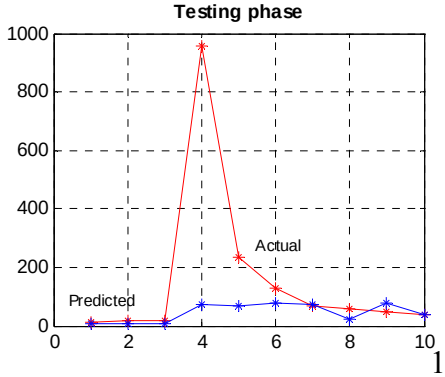
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.72121	0.7053	2075729.8117	0.70128	382.8061
TESTING	0.98873	0.4354	81508.7374	0.5229	120.5357

RUN #55

==========

DATA SET: COCOMO
NN Architecture: 3-20-20-20-1





Actual	Predicted	Actual	Predicted
243	858,3215	82	96,12955
240	308,5135	55	65,14086
33	93,85612	47	101,4693
43	45,42295	12	80,91753
8	39,148	8	41,84217
1075	71,21323	8	29,95667
423	165,8173	6	33,96212
321	150,1392	45	63,88657
218	106,1182	83	93,65705
201	133,6055	87	105,7045
79	116,014	106	111,5567
60	52,67395	126	165,4225
61	33,94599	36	119,6148
40	37,78229	1272	791,4707
9	32,88663	156	665,6038
11400	801,2972	176	192,0356
6600	6801,237	122	61,05975
6400	5501,604	41	55,15621
2455	2364,93	14	44,06986
724	1279,661	20	35,6795
539	436,6803	18	38,43384
453	327,5121	958	80,04652
523	216,084	237	131,7418
387	185,2262	130	132,713
88	118,0977	70	97,2374
98	66,21571	57	56,05978
7,3	43,93223	50	83,21758
5,9	29,57518	38	65,8391
1063	121,5158		
702	786,985		
605	473,8726		
230	135,2452		

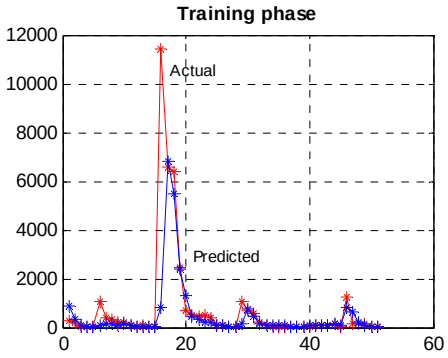
Run #	59
Data Set	COCOMO
NN Architecture	5-15-15-15-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	45
Num of Test Set	5
Num of Production Set	10

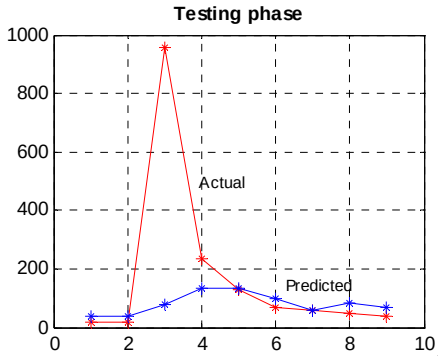
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.75723	0.65835	2286430.0032	1.1601	361.9915
TESTING	0.97982	0.19506	87241.5028	0.56702	123.4747

RUN #59

==========

DATA SET: COCOMO
NN Architecture: 5-15-15-15-1





Run #	76
Data Set	KEMERER
NN Architecture	1-3-3-3-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (1)
OUTPUTS	EFF(1)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	9
Num of Test Set	3
Num of Production Set	3

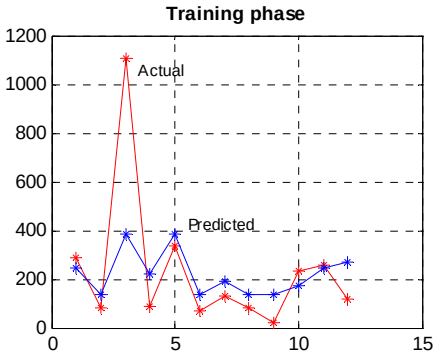
Actual	Predicted
287	244.6353
82.5	134.7337
1107.31	387.6462
86.9	219.394
336.3	387.5742
72	134.1959
130.3	191.779
84	138.2103
23.2	135.6368
230.7	171.917
258.7	245.0369
116	269.0193
157	188.7407
246.9	190.5797
69.9	142.0801

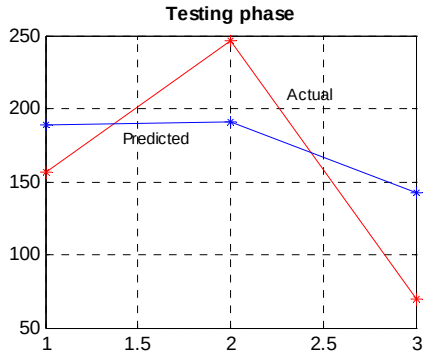
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.76079	0.75026	49409.0639	0.9635	126.1515
TESTING	0.63212	0.87793	3129.8062	0.48763	53.4137

RUN #76

==========

DATA SET: KEMERER
NN Architecture: 1-3-3-3-1





Run #	77
Data Set	KEMERER
NN Architecture	1-6-6-6-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (1)
OUTPUTS	EFF(1)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	9
Num of Test Set	3
Num of Production Set	3

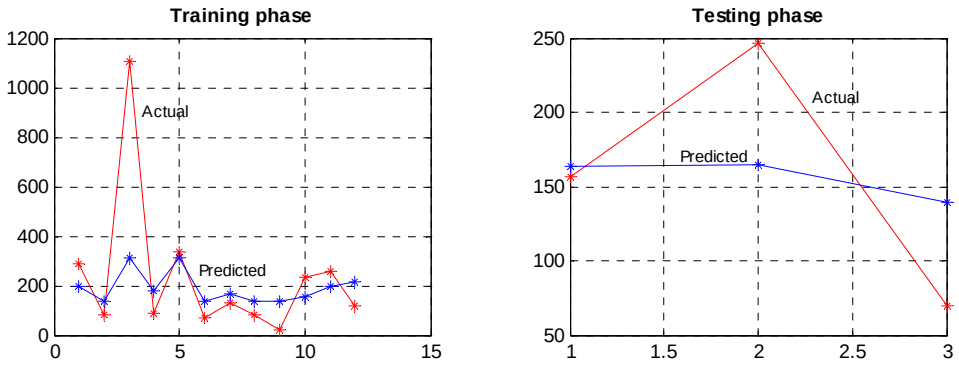
Actual	Predicted
287	197.1302
82.5	136.7674
1107.31	309.6788
86.9	181.3048
336.3	309.6085
72	136.552
130.3	165.1212
84	138.1934
23.2	137.1321
230.7	154.303
258.7	197.3899
116	213.3579
157	163.4178
246.9	164.4469
69.9	139.8459

	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.82501	0.77251	58102.7317	0.91535	130.4524
TESTING	0.70659	0.87961	3910.7112	0.4585	52.9389

RUN #77

==========

DATA SET: KEMERER
NN Architecture: 1-6-6-6-1



Run #	78
Data Set	KEMERER
NN Architecture	1-9-9-9-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (1)
OUTPUTS	EFF(1)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	9
Num of Test Set	3
Num of Production Set	3

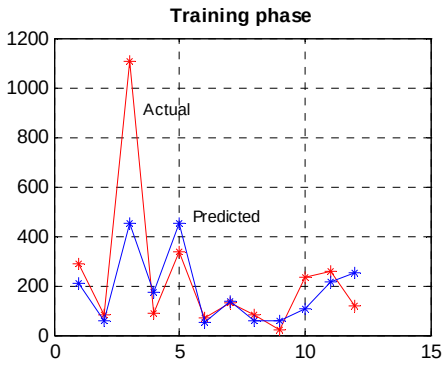
Actual	Predicted
287	211.9455
82.5	54.80942
1107.31	452.4987
86.9	173.7319
336.3	452.367
72	54.10038
130.3	133.3656
84	59.40681
23.2	56.00128
230.7	105.2684
258.7	212.5637
116	250.056
157	129.0166
246.9	131.6466
69.9	64.55392

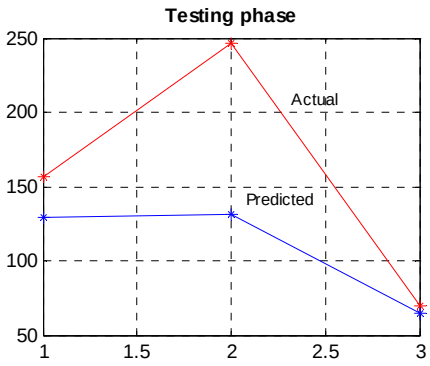
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.69446	0.76166	41169.3677	0.53244	112.0366
TESTING	0.77448	0.87848	4698.333	0.24051	49.5276

RUN #78

==========

DATA SET: KEMERER
NN Architecture: 1-9-9-9-1





Run #	79
Data Set	KEMERER
NN Architecture	1-15-15-15-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (1)
OUTPUTS	EFF(1)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	9
Num of Test Set	3
Num of Production Set	3

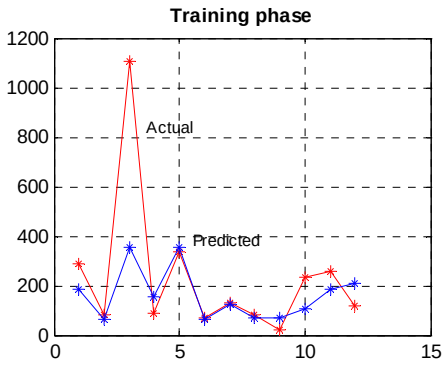
Actual	Predicted
287	183.532
82.5	66.0604
1107.31	355.8853
86.9	155.5946
336.3	355.7912
72	65.51151
130.3	125.7121
84	69.61479
23.2	66.98258
230.7	104.6301
258.7	183.9817
116	211.1351
157	122.4654
246.9	124.4295
69.9	73.58464

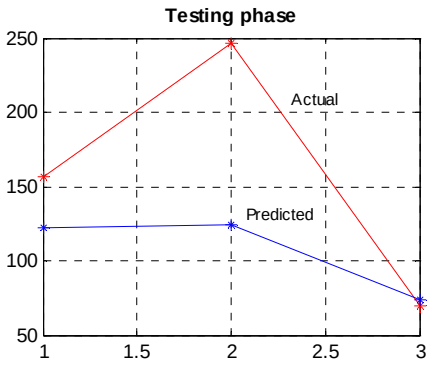
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.77384	0.75904	51119.0082	0.49384	110.3905
TESTING	0.83044	0.87824	5401.7461	0.25624	53.5632

RUN #79

==========

DATA SET: KEMERER
NN Architecture: 1-15-15-15-1





Run #	95
Data Set	KEMERER
NN Architecture	4-20-20-20-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (2),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

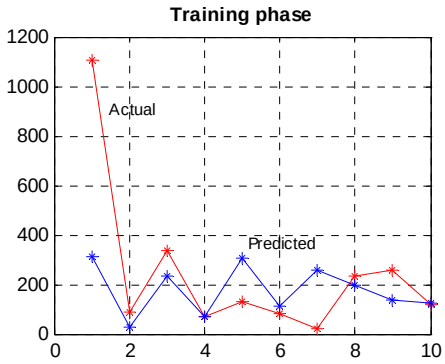
Actual	Predicted
1107.31	312.3629
86.9	29.41367
336.3	231.2148
72	70.7884
130.3	305.5661
84	111.1623
23.2	258.6324
230.7	198.8878
258.7	133.9466
116	123.1846
157	160.162
246.9	232.5703
69.9	183.838

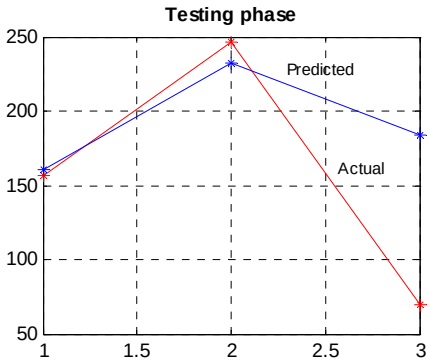
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.86013	0.5222	74980.155	1.4207	156.0341
TESTING	0.74941	0.66681	4399.0665	0.5694	43.8099

RUN #95

==========

DATA SET: KEMERER
NN Architecture: 4-20-20-20-1





Run #	101
Data Set	KEMERER
NN Architecture	5-3-3-3-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

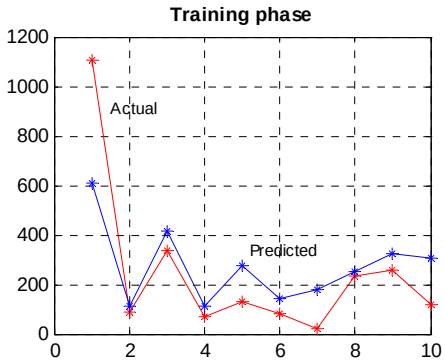
Actual	Predicted
1107,31	610,3717
86,9	112,5952
336,3	413,0841
72	113,6016
130,3	277,9168
84	142,3942
23,2	179,5429
230,7	254,0469
258,7	327,1265
116	305,6232
157	204,5535
246,9	249,3592
69,9	154,2723

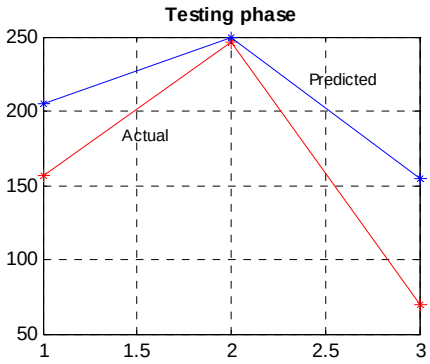
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.58434	0.89168	34606.2374	1.2118	128.477
TESTING	0.632	0.9991	3128.6915	0.50663	44.795

RUN #101

==========

DATA SET: KEMERER
NN Architecture: 5-3-3-3-1





Run #	102
Data Set	KEMERER
NN Architecture	5-6-6-6-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

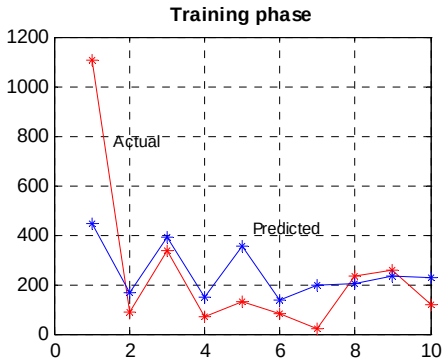
Actual	Predicted
1107,31	447,8079
86,9	169,6286
336,3	392,0263
72	146,5867
130,3	352,5529
84	135,9992
23,2	196,8683
230,7	201,3296
258,7	232,2452
116	227,0173
157	201,0502
246,9	254,0021
69,9	173,8456

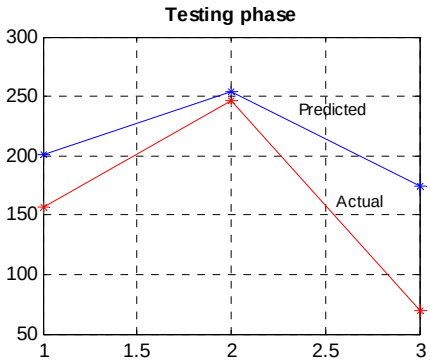
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.73439	0.75875	54660.399	1.3746	148.7307
TESTING	0.73792	0.98486	4265.1859	0.5988	51.6993

RUN #102

==========

DATA SET: KEMERER
NN Architecture: 5-6-6-6-1





Run #	103
Data Set	KEMERER
NN Architecture	5-9-9-9-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

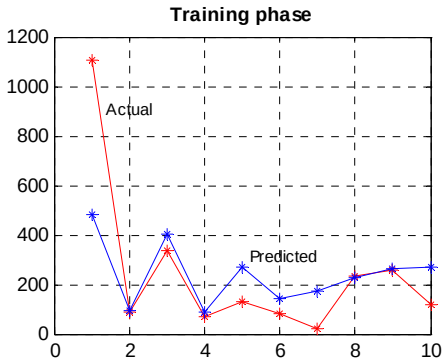
Actual	Predicted
1107,31	479,6989
86,9	95,98276
336,3	404,6939
72	90,57819
130,3	267,4804
84	141,3655
23,2	170,6883
230,7	229,0874
258,7	263,7527
116	267,1547
157	195,8704
246,9	207,5021
69,9	140,737

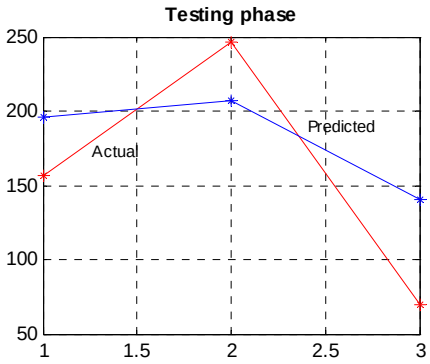
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.67789	0.8142	46573.908	1.0555	122.352
TESTING	0.58642	0.93271	2693.6594	0.47352	49.7018

RUN #103

==========

DATA SET: KEMERER
NN Architecture: 5-9-9-9-1





Run #	104
Data Set	KEMERER
NN Architecture	5-15-15-15-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

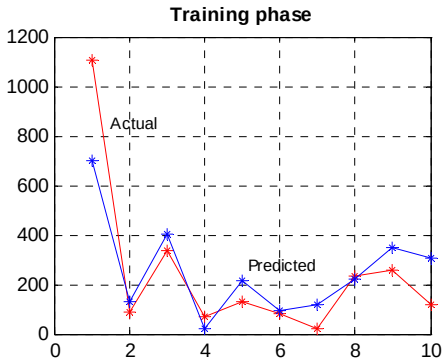
Actual	Predicted
1107,31	698,3923
86,9	132,3661
336,3	402,7727
72	23,2
130,3	212,7295
84	96,91717
23,2	121,0492
230,7	224,2189
258,7	349,0978
116	308,8285
157	129,5223
246,9	186,1873
69,9	98,89799

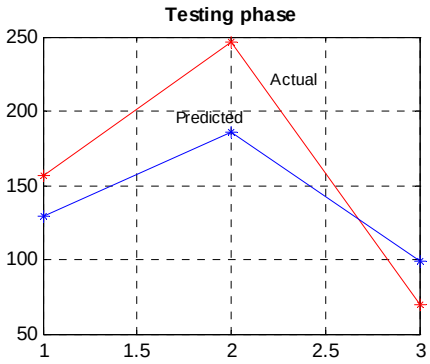
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.48461	0.90991	23801.3446	0.88118	105.256
TESTING	0.47411	0.987	1760.6478	0.27859	39.0628

RUN #104

==========

DATA SET: KEMERER
NN Architecture: 5-15-15-15-1





Run #	105
Data Set	KEMERER
NN Architecture	5-20-20-20-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC (3),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	7
Num of Test Set	3
Num of Production Set	3

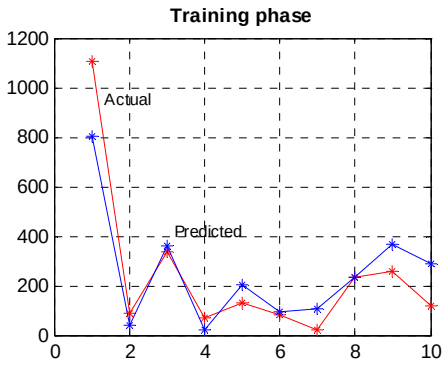
Actual	Predicted
1107,31	804,8362
86,9	40,9173
336,3	362,9564
72	23,2
130,3	205,7175
84	91,52464
23,2	103,572
230,7	232,4196
258,7	367,2479
116	286,5531
157	118,0105
246,9	181,6809
69,9	82,72849

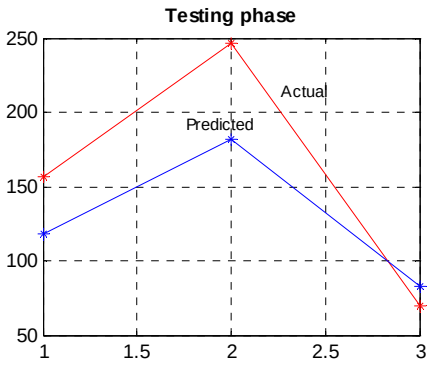
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.38442	0.93928	14977.4907	0.75894	86.8048
TESTING	0.5027	0.98801	1979.4266	0.23201	39.0124

RUN #105

==========

DATA SET: KEMERER
NN Architecture: 5-20-20-20-1





Actual	Predicted	Actual	Predicted	Actual	Predicted
33	71,82648	1063	348,7873	237	346,1693
43	223,5205	702	249,5084	130	358,4468
8	5,9	605	147,7558	70	235,7434
1075	163,7953	230	5,9	57	191,8643
423	277,1146	82	5,9	50	252,0094
321	341,9996	55	5,9	38	197,5027
218	250,804	47	147,8196	15	264,2271
201	134,555	12	215,2818	287	304,3916
79	113,6511	8	279,8081	82,5	246,6934
60	194,402	8	268,0193	1107,31	325,6726
61	204,7426	6	219,6709	86,9	130,0766
40	174,8223	45	333,6637	336,3	73,93882
9	285,4156	83	361,3959	72	5,9
11400	415,0936	87	340,1215	130,3	5,9
6600	6225,775	106	273,6566	84	5,9
6400	6351,034	126	180,7311	23,2	10,7739
2455	2366,463	36	117,0114	230,7	30,47791
724	650,7435	1272	148,7494	258,7	5,9
539	427,7054	156	51,93347	116	267,0526
453	280,8433	176	164,5958	157	365,7334
523	13,10962	122	5,9	246,9	95,81247
387	5,9	41	146,8471	69,9	5,9
88	5,9	14	5,9		
98	5,9	20	259,7923		
7,3	91,32728	18	349,1069		
5,9	165,867	958	369,3101		

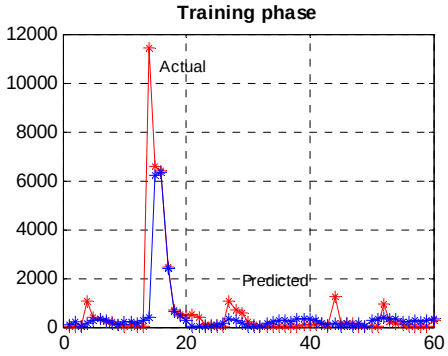
Run #	150
Data Set	COCOMO&KEMERER
NN Architecture	8-20-20-20-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	LOC(4),EFF(4)
OUTPUTS	EFF(5)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	Tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	Logistic
Num of Training Set	50
Num of Test Set	13
Num of Production Set	11

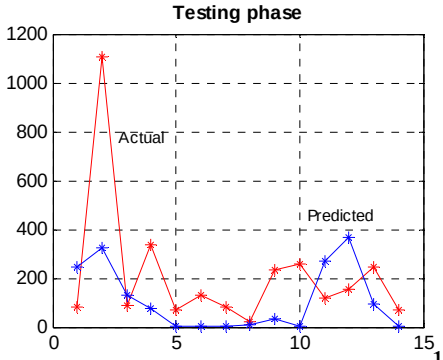
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.78133	0.62643	2097968.6715	5.038	388.7748
TESTING	0.94758	0.43199	66572.816	0.95111	182.8779

RUN #150

==========

DATA SET: COCOMO&KEMERER
NN Architecture: 8-20-20-20-1





Actual	Predicted	Actual	Predicted	Actual	Predicted
805	1977,629	3948	4104,43	3276	4447,707
3829	3181,197	3927	4047,077	2723	2158,424
2149	4590,372	710	1187,628	3472	5171,423
2821	2355,057	2429	1442,898	1575	6892,152
2569	4398,191	6405	8731,681	2926	4261,023
3913	4281,07	651	1581,377	1876	2564,624
7854	9014,612	9135	9895,454	2520	2606,235
2422	1101,758	1435	1371,555	1603	6023,491
4067	4414,416	5922	6019,221	3626	3895,391
9051	6380,88	847	1275,916	6783	7303,205
2282	3468,49	8050	7494,367	11361	10668,5
4172	4268,744	4620	3716,476	1267	1117,318
4977	4388,922	2352	2426,531	2548	2870,09
1617	1365,561	2174	2217,23	1155	3838,03
3192	3860,771	19894	19907,31	546	1268,687
3437	5943,373	6699	6509,631	2275	1750,155
4494	4673,147	14987	14979,88	9100	5928,732
840	546	4004	4312,933	595	1564,197
14973	14685,79	12824	13002,68	3941	4000,102
5180	5293,016	2331	2079,172	13860	13870,6
5775	5308,137	5817	5336,146	1400	1435,853
10577	10249,53	2989	2621,582	2800	3036,612
3983	4138,394	3136	2662,828	9520	9248,257
3164	2925,543	14434	8174,405	5880	5835,782
3542	5025,813	2583	2500,193	23940	23764,5
4277	4419,518	3647	3920,05		
7252	6779,663	8232	7519,417		

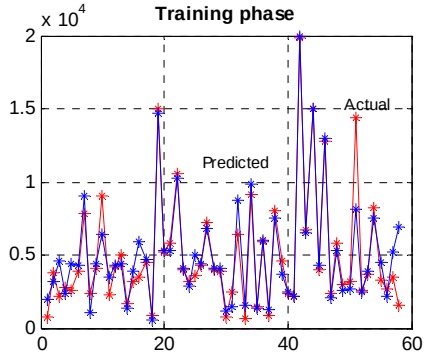
Run #	218
Data Set	DESHARNAIS
NN Architecture	4-9-9-9-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	FP (2),EFF(2)
OUTPUTS	EFF(3)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	53
Num of Test Set	5
Num of Production Set	21

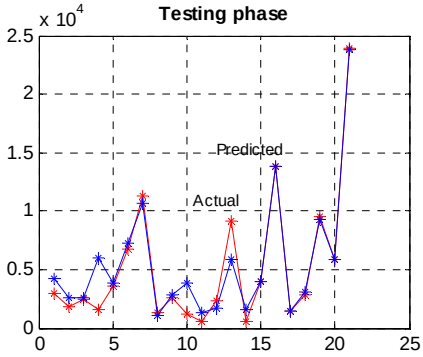
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.35458	0.93454	2004852.7334	0.29805	830.299
TESTING	0.24746	0.96996	1995890.8337	0.48074	828.043

RUN #218

==========

DATA SET: DESHARNAIS
NN Architecture: 4-9-9-9-1





Actual	Predicted	Actual	Predicted	Actual	Predicted
3829	3810,38	3927	3938,539	2723	2654,619
2149	2226,719	710	750,6874	3472	3357,653
2821	2839,955	2429	2395,375	1575	1881,022
2569	2367,534	6405	6394,41	2926	2569,114
3913	4056,276	651	677,7527	1876	1536,395
7854	7074,973	9135	8945,877	2520	2542,896
2422	1851,816	1435	1449,435	1603	2089,645
4067	4386,105	5922	5821,561	3626	3588,844
9051	8948,063	847	776,489	6783	6410,07
2282	2426,307	8050	8031,096	11361	11179,96
4172	3912,698	4620	4554,076	1267	1244,567
4977	4843,779	2352	2332,513	2548	2565,912
1617	1332,754	2174	2163,392	1155	1122,64
3192	3122,845	19894	19874,38	546	605,1394
3437	3444,745	6699	6676,112	2275	2139,423
4494	4586,813	14987	14937,6	9100	8910,798
840	862,7466	4004	4017,948	595	546
14973	14934,5	12824	12764,23	3941	3899,798
5180	5111,623	2331	2301,651	13860	13856,85
5775	5727,415	5817	5834,295	1400	1356,896
10577	10611,64	2989	3100,602	2800	2782,362
3983	3950,726	3136	2964,175	9520	9500,968
3164	3184,272	14434	14316,11	5880	5868,382
3542	3560,93	2583	2517,409	23940	23940
4277	4257,397	3647	3628,276		
7252	7179,576	8232	8169,132		
3948	3885,834	3276	3314,518		

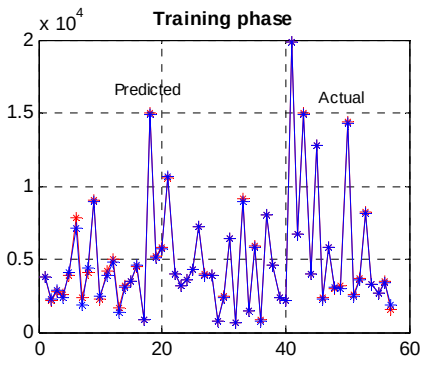
Run #	223
Data Set	DESHARNAIS
NN Architecture	6-9-9-9-1
Method	Backprob w/ 3 Hidden Slabs
INPUTS	FP (3),EFF(3)
OUTPUTS	EFF(4)
Activation Function Slab 1	linear [-1,1]
Activation Function Slab 2	Gaussian
Activation Function Slab 3	tanh
Activation Function Slab 4	Gaussian comp.
Activation Function Slab 5	logistic
Num of Training Set	52
Num of Test Set	5
Num of Production Set	21

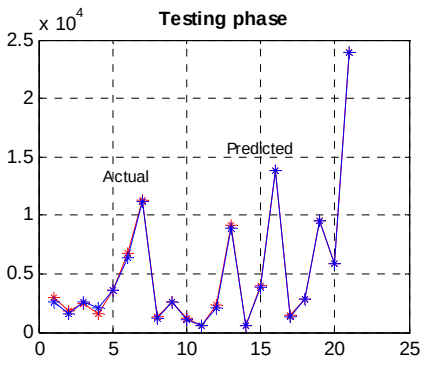
	Normalized RMSE error	Correlation coefficient	MSE error	MRE error	MAE error
TRAINING	0.041726	0.99919	27708.9043	0.031155	97.5483
TESTING	0.032424	0.99951	34266.779	0.051066	116.1206

RUN #223

==========

DATA SET: DESHARNAIS
NN Architecture: 6-9-9-9-1





Παράρτημα Β

Ενδεικτικός Τεκμηριωμένος Κώδικας Εργαλείου

Β.1. Κώδικας Εργαλείου

```
function varargout = NN_GA_SCE(varargin)


%    NN_GA_SCE Application M-file for NN_GA_SCE.fig
%    FIG = NN_GA_SCE launch NN_GA_SCE GUI.
%    The Tool Interface
% Last Modified by GUIDE v2.0 10-May-2004 20:51:29


if nargin == 0  % LAUNCH GUI

        fig = openfig(mfilename,'reuse');

        % Use system color scheme for figure:
        set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));

        % Generate a structure of handles to pass to callbacks, and store it.
        handles = guihandles(fig);
        guidata(fig, handles);


        if nargout > 0
                varargout{1} = fig;
        end

elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK

        try
```

```
            if (nargout)
                [varargout{1:nargout}] = feval(varargin{:}); % FEVAL
        switchyard
            else
                feval(varargin{:}); % FEVAL switchyard
            end
        catch
            disp(lasterr);
        end

end
```

Επιλογή Μεθόδου Διαχωρισμού των Δεδομένων στα απαραίτητα σύνολα

```
% --------------------------------------------------------------------
function varargout = radiobutton1_Callback(h, eventdata, handles, varargin)
off = [handles.radiobutton2,handles.radiobutton3];
mutual_exclude(off)
set(handles.radiobutton1,'Value',1)

% --------------------------------------------------------------------
function varargout = radiobutton2_Callback(h, eventdata, handles, varargin)
off = [handles.radiobutton1,handles.radiobutton3];
mutual_exclude(off)
set(handles.radiobutton2,'Value',1)


% --------------------------------------------------------------------
function varargout = radiobutton3_Callback(h, eventdata, handles, varargin)
off = [handles.radiobutton1,handles.radiobutton2];
mutual_exclude(off)
set(handles.radiobutton3,'Value',1)
```

Λήψη Επιλογών Χρήστη και Επιλογή Τεχνικής Τεχνητών Νευρωνικών Δικτύων

```
% --------------------------------------------------------------------
function varargout = pushbutton1_Callback(h, eventdata, handles, varargin)


datasets = get(handles.popupmenu1,'String');
position_datasets=get(handles.popupmenu1,'Value');
excel_file = datasets{position_datasets,:};

inputs = get(handles.popupmenu2,'String');
position_inputs = get(handles.popupmenu2,'Value');
chosen_input = inputs{position_inputs,:};
number_of_inputs = str2num(chosen_input);

neurons = get(handles.popupmenu3,'String');
```

```
position_neurons = get(handles.popupmenu3,'Value');
chosen_neurons = neurons{position_neurons,:};
number_of_hidden_neurons = str2num(chosen_neurons);

epochs = get(handles.popupmenu4,'String');
position_epochs = get(handles.popupmenu4,'Value');
chosen_epochs = epochs{position_epochs,:};
number_epochs = str2num(chosen_epochs);

tr_funcs = get(handles.popupmenu5,'String');
position_tr_funcs = get(handles.popupmenu5,'Value');
training_func = tr_funcs{position_tr_funcs,:};

lr_funcs = get(handles.popupmenu6,'String');
position_lr_funcs = get(handles.popupmenu6,'Value');
learning_func = lr_funcs{position_lr_funcs,:};

perf_funcs = get(handles.popupmenu7,'String');
position_perf_funcs = get(handles.popupmenu7,'Value');
performance_func = perf_funcs{position_perf_funcs,:};

if get(handles.checkbox2,'Value') == 0
   fixed_rows = 1;
else
   fixed_rows = 0;
end

if get(handles.checkbox4,'Value') == 1
   use_val = 1;
else
   use_val = 0;
end

if get(handles.radiobutton1,'Value') == 1
[G,H]=nn_loc_predicts_eff(excel_file,number_of_inputs,number_of_hidden_neurons,tr
        aining_func,learning_func,performance_func,number_epochs,fixed_rows,use_v
        al);

elseif get(handles.radiobutton2,'Value')== 1
 [G,H] =
        nn_loc_eff_predicts_next_eff(excel_file,number_of_inputs,number_of_hidden_
        neurons,training_func,learning_func,performance_func,number_epochs,fixed_r
        ows,use_val);
elseif get(handles.radiobutton3,'Value') == 1
 [G,H] =
        nn_nextloc_eff_predicts_nexteff(excel_file,number_of_inputs,number_of_hidd
        en_neurons,training_func,learning_func,performance_func,number_epochs,fixe
        d_rows,use_val);
else
```

```
   msgbox('You must select one of the Techniques for Taking Samples','Error','error')
end

dlmwrite('NN RESULTS.xls',[G H],'\t')
```

Λήψη Επιλογών Χρήστη και Επιλογή Τεχνικής Γενετικού Αλγόριθμου

```
% --------------------------------------------------------------------
function varargout = pushbutton2_Callback(h, eventdata, handles, varargin)

datasets = get(handles.popupmenu1,'String');
position_datasets=get(handles.popupmenu1,'Value');
excel_file = datasets{position_datasets,:};

epochs = get(handles.popupmenu4,'String');
position_epochs = get(handles.popupmenu4,'Value');
chosen_epochs = epochs{position_epochs,:};
number_epochs = str2num(chosen_epochs);

tr_funcs = get(handles.popupmenu5,'String');
position_tr_funcs = get(handles.popupmenu5,'Value');
training_func = tr_funcs{position_tr_funcs,:};

lr_funcs = get(handles.popupmenu6,'String');
position_lr_funcs = get(handles.popupmenu6,'Value');
learning_func = lr_funcs{position_lr_funcs,:};

perf_funcs = get(handles.popupmenu7,'String');
position_perf_funcs = get(handles.popupmenu7,'Value');
performance_func = perf_funcs{position_perf_funcs,:};

repetitions = get(handles.popupmenu8,'String');
position_rep = get(handles.popupmenu8,'Value');
chosen_rep = repetitions{position_rep,:};
repetitions_ga = str2num(chosen_rep);

atoma = get(handles.popupmenu11,'String');
position_atoms = get(handles.popupmenu11,'Value');
chosen_atom = atoma{position_atoms,:};
atoms = str2num(chosen_atom);

repetitions = get(handles.popupmenu8,'String');
position_rep = get(handles.popupmenu8,'Value');
chosen_rep = repetitions{position_rep,:};
repetition_ga = str2num(chosen_rep);

pcs = get(handles.popupmenu9,'String');
position_pc = get(handles.popupmenu9,'Value');
chosen_pc = pcs(position_pc,:);
```

```
pc = str2num(chosen_pc);

pms = get(handles.popupmenu10,'String');
position_pm = get(handles.popupmenu10,'Value');
chosen_pm = pms(position_pm,:);
pm = str2num(chosen_pm);

if get(handles.checkbox1,'Value') == 0
   printgraphs = 0;
else
   printgraphs = 1;
end

if get(handles.checkbox2,'Value') == 0
   fixed_rows = 1;
else
   fixed_rows = 0;
end

if get(handles.checkbox4,'Value') == 1
   use_val = 1;
else
   use_val = 0;
end

if get(handles.radiobutton1,'Value') == 1
   [best_of_atoms_architecture,fitness_all,best_atom_train,best_atom_test,fitness_best]
         =
         ga_loc_predicts_eff(excel_file,training_func,learning_func,performance_func,n
         umber_epochs,repetition_ga,atoms,pc,pm,printgraphs,fixed_rows,use_val);
   % choose best atom and plot
   topology =
         choose_best_atom(best_of_atoms_architecture,fitness_all(:),best_atom_train,be
         st_atom_test,fitness_best(:),1,0);
elseif get(handles.radiobutton2,'Value')== 1
   [best_of_atoms_architecture,fitness_all,best_atom_train,best_atom_test,fitness_best]
         =
         ga_loc_eff_predicts_next_eff(excel_file,training_func,learning_func,performan
         ce_func,number_epochs,repetition_ga,atoms,pc,pm,printgraphs,fixed_rows,use
         _val);
   % choose best atom and plot
   topology =
         choose_best_atom(best_of_atoms_architecture,fitness_all(:),best_atom_train,be
         st_atom_test,fitness_best(:),2,0);
elseif get(handles.radiobutton3,'Value') == 1
   [best_of_atoms_architecture,fitness_all,best_atom_train,best_atom_test,fitness_best]
         =
         ga_next_loceff_predicts_nexteff(excel_file,training_func,learning_func,perform
```

```
        ance_func,number_epochs,repetition_ga,atoms,pc,pm,printgraphs,fixed_rows,u
        se_val);
    % choose best atom and plot
    topology =
        choose_best_atom(best_of_atoms_architecture,fitness_all(:),best_atom_train,be
        st_atom_test,fitness_best(:),2,1);

else
    msgbox('You must select one of the Techniques for Taking Samples','Error','error')
end
```

B.2 Κώδικας Λήψης Δεδομένων και Διαχωρισμός σε Σύνολα

```
function [G,TE] = nn_loc_predicts_eff
        (excel_file,number_of_inputs,number_of_hidden_neurons,training_func,learnin
        g_func,performance_func,number_epochs,fixed_rows,use_val)

% Function Name :
% nn_loc_predicts_eff - Neural Network for the Cocomo data set
%
% Description :
% Reads the data set from an excel file and determines the sets used for training, for
        validation and for testing. Trains and Tests the NN
%
% Inputs :
%       excel_file                  : the name of the excel file with the data
%       number_of_inputs            : the number of input neurons for the neural network
%       number_of_hidden_neurons : the number of neurons in the intermediate, hidden
        layer between input and output
%       training_func               : training function for the neural net
        (e.g.'trainlm','trainbfg','trainrp','traingd' etc)
%       learning_func               : learning function for the neural net
        (e.g.'learngdm','learngd' etc)
%       performance_func            : performance function (differentiable) for the
        neural net (e.g.'mse','mae','msereg' etc)
%       number_epochs               : the number of the epochs to train the network
        (iterations)
%       fixed_rows                  : defines whether the number of projects used for
        training, validation, testing will be determined by the user
%       use_val                     : defines whether validation will be used for the
        training
%
% Outputs :
%         training results, testing results, graphs of actual and predicted effort
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Read the data
%   CHARA : columns names  (note: Assume that the first column is the Effort !!!)
%   NUM   : data

[NUM,CHARA] = xlsread(excel_file);

[row,col] = size(NUM);

% number of inputs = how many previous effort values, to predict the next
% e.g. if given inputs = 3 then for the cocomo set (Effort,ALOC,LOC) 3 is used for the
        Effort
%                       and 2*3=6 is used for ALOC and LOC
%                       the targer is the 4th row
%     if given inputs = 5 then for the cocomo set (Effort,ALOC,LOC) 5 are used for the
        Effort (5 rows)
%                       and 2*5=10 is used for ALOC and LOC,
%                       the targer is the 6th row


% effort_neurons       : number of neurons for Effort
% other_input_neurons  : number of neurons for the rest parameters
% NEURONS_NM           : the total number of neurons used for the network input

  NEURONS_NUM = number_of_inputs;

  other_input_neurons = NEURONS_NUM-number_of_inputs;
  effort_neurons=number_of_inputs;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Προετοιμασία Συνόλου Εκπαίδευσης / Επικύρωσης / Δοκιμής

```
% Prepare the data

   % Seperate the table into two arrays
   % INPUTS : the other parameters
   % EFFORT : the one column with the effort
   i=1:row;
   j=2:col;
   INPUTS_temp = NUM(i,j);

   %% NORMALIZATION
   % Input  : Array , Old lower&higher bound , New lower&higher bound
   % Output : New Normalized Array

   if isempty(INPUTS_temp) == 1
   INPUTS = zeros(row,0);
elseif isempty(INPUTS_temp) == 0
```

```
    IN1 =
        rescale(INPUTS_temp(i,1),min(INPUTS_temp(i,1)),max(INPUTS_temp(i,1)),0
        .00001,1);
    %IN2 =
        rescale(INPUTS_temp(i,2),min(INPUTS_temp(i,2)),max(INPUTS_temp(i,2)),0
        .00001,1);
    %INPUTS = [IN1,IN2];
    INPUTS=IN1;
end

    [row_size_INPUTS,col_size_INPUTS] = size(INPUTS);

    i=1:row;
    EFF_temp = NUM(i,1);

    EFFORT =
        rescale(EFF_temp(i,1),min(EFF_temp(i,1)),max(EFF_temp(i,1)),0.00001,1);

    EFFORT_ROW = conv_column2row(EFFORT);
    effort_range = minmax(EFFORT_ROW);


 % Prepare the sets

    % Definition :      70% of the rows will be used for training
    %                   10% of the rows will be used for validation
    %                   20% of the rows will be used for testing
    % note: some rows will be ignored
    num4 = floor((row * 70)/100);
    num5 = floor((row * 10)/100);
    num6 = floor((row * 20)/100);

    num7 = num4+num5+num6;

if fixed_rows == 1
   prompt  = {'Enter number of rows for training:','Enter number of rows for
        validation:','Enter number of rows for testing'};
   title1   = 'Change Fixed Sizes of the Sets';
   lines= 1;
   num1=num2str(num4);
   num2=num2str(num5);
   num3=num2str(num6);
   def     = {num1,num2,num3};
   answer  = inputdlg(prompt,title1,lines,def);

   num4 = str2num(answer{1,:});
   num5 = str2num(answer{2,:});
   num6 = str2num(answer{3,:});
```

```
  num7=num4+num5+num6;
end

  if num7 <= row
  num_rows_for_training = num4;
  num_rows_for_validation = num5;
  num_rows_for_testing = num6;
  %warndlg('The Sets have been changed Successfully!','Done!');
  else
  helpdlg('Choose Numbers that Make a total of:63 for COCOMO, 15 for Kemerer,24
        for Albrecht, 81 for Desharnais','ERROR Creating Sets');
  break
  end

if num_rows_for_training==0 | num_rows_for_validation==0 |
        num_rows_for_testing==0
   errordlg('Neural Network cannot Function with this File because there are too few
        data!!No training!','File Error');
   break
end
   % produce num_rows_for_training of samples for the training set
   %
   i=1:num_rows_for_training;
   j=1:col_size_INPUTS;
   train_set_inputs = INPUTS(i,j);

   i=1:num_rows_for_training;
   j=1;
   train_set_effort = EFFORT(i,j);

   % produce num_rows_for_validation of samples for the validation set
   %
   i=1:num_rows_for_validation;
   j=1:col_size_INPUTS;
   val_set_inputs = INPUTS(i,j);

   i=1:num_rows_for_validation;
   j=1;
   val_set_effort = EFFORT(i,j);

   % produce num_rows_for_testing of samples for the testing set
   %
   i=1:num_rows_for_testing;
   j=1:col_size_INPUTS;
   test_set_inputs = INPUTS(i,j);

   i=1:num_rows_for_testing;
   j=1;
   test_set_effort = EFFORT(i,j);
```

B.3 Κώδικας Τεχνητού Νευρωνικού Δικτύου

Δημιουργία Τεχνητού Νευρωνικού Δικτύου

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%CREATE THE NETWORK

 % Find the Ranges
 % Seperates all input columns into Yx1 columns (one-dimensional)
 % using the function conv_column2row(column_inp) that converts a column into a
        row
% The Neural Net

% NEWFF Create a feed-forward backpropagation network.
% create a minmax(train_set)-layer network
% me ? neurones sto input layer
%     The network will have an input (ranging from the min to max values of the
        train_set),
%     followed by a layer (1st hidden layer) of 20 TANSIG neurons, and followed
%     by a second layer (2nd hidden layer) of 20 TANSIG neurons, and followed
%     by a third layer (3rd hidden layer) of 20 TANSIG neurons, and at the end followed
%         with 1 PURELIN neuron (output).  TRAINLM backpropagation is used.The
        last layer is the network output.

   % The data need to be divided according to the number of the input neurons

 switch effort_neurons
 case 1,
      net =
        newff(range,[number_of_hidden_neurons,number_of_hidden_neurons,number_
        of_hidden_neurons,1],{'tansig','tansig','tansig','purelin'},training_func,learning_f
        unc,performance_func);

 case 2,
      net =
        newff(range,[number_of_hidden_neurons,number_of_hidden_neurons,number_
        of_hidden_neurons,1],{'tansig','tansig','tansig','purelin'},training_func,learning_f
        unc,performance_func);

 case 3,
      net =
        newff(range,[number_of_hidden_neurons,number_of_hidden_neurons,number_
        of_hidden_neurons,1],{'tansig','tansig','tansig','purelin'},training_func,learning_f
        unc,performance_func);

 case 4,
      net =
        newff(range,[number_of_hidden_neurons,number_of_hidden_neurons,number_
```

```
        of_hidden_neurons,1],{'tansig','tansig','tansig','purelin'},training_func,learning_f
        unc,performance_func);

 case 5,
     net =
        newff(range,[number_of_hidden_neurons,number_of_hidden_neurons,number_
        of_hidden_neurons,1],{'tansig','tansig','tansig','purelin'},training_func,learning_f
        unc,performance_func);

 otherwise,

 end

 %% TRAIN PARAMETERS %%
 net.trainParam.show = 5;
% the number of epochs used to train the network
 net.trainParam.epochs = number_epochs;
```

<u>Εκπαίδευση Νευρωνικού Δικτύου</u>

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% TRAINING
% The training will be repeated as many times as the total number of the input rows
        used for training -
% the number of inputs for the effort + 1
% produce samples for training
% note : a moving window of size = number_of_inputs
% train_set = effort rows (as many as number_of_inputs)/rows for all other parameters
        (as many as number_of_inputs)
if number_of_inputs == 1
   times = num_rows_for_training - number_of_inputs;
else

times = num_rows_for_training - number_of_inputs+1;
end

k=1:col-1;

for i=1:times

       j=i:i+number_of_inputs-1;
       train_set = train_set_inputs(j,k);
       train_set=train_set(:);
       %effort_col = train_set_effort(j);

       % vazo se mia stili tis times gia training
       %train_set=cat(1,effort_col,train_set);

       [r,c] = size(train_set);
```

```
        s=1:r;
       %% TRAINING SET ARRAY
        new_train_set(s,i)=train_set(s);


        target=EFFORT(i+number_of_inputs-1);

        %% TARGET SET ARRAY
       new_target(1,i) = target;
  end
```

Επικύρωση Δικτύου

```
%%%%%%%%%%%%
%% VALIDATION %%
%%%%%%%%%%%%
if number_of_inputs == 1
   times2 = num_rows_for_validation - number_of_inputs;
else

times2 = num_rows_for_validation - number_of_inputs+1;
end

k=1:col-1;

for ii=1:times2
     j=ii:ii+number_of_inputs-1;
        val_set = val_set_inputs(j,k);
        val_set=val_set(:);
       % val_effort_col = val_set_effort(j);

        % vazo se mia stili tis times gia validation
       % val_set=cat(1,val_effort_col,val_set);

        [r,c] = size(val_set);
        s=1:r;

       %% VALIDATION SET ARRAY
        val.P(s,ii) = val_set(s);


       if(ii <= times2)
         val_target = val_set_effort(ii);
       %% VALIDATION TARGET SET ARRAY
         val.T(1,ii) = val_target;
      end
  end

%%%%%%%%%%%%%%%%
```

```
%% TRAINING FUNCTION %%
%%%%%%%%%%%%%%%%%%%
if use_val == 0
   [net,tr,outp]=train(net,new_train_set,new_target);
else
   [net,tr,outp]=train(net,new_train_set,new_target,[],[],val);
end
          % Results from training
          array_tr = [new_target(:),outp(:)];
         G = result_calc(array_tr);
         dlmwrite('OUT_TRAINING.xls',G,'\t')

%%%%%%%%%%
%% TESTING %%%
%%%%%%%%%%


 if number_of_inputs == 1
   times3 = num_rows_for_testing - number_of_inputs;
else

times3 = num_rows_for_testing - number_of_inputs+1;
end

kk=1:col-1;

for iii=1:times3
     j=iii:iii+number_of_inputs-1;
        test_set = test_set_inputs(j,kk);
        test_set=test_set(:);
       %test_effort_col = test_set_effort(j);

        % vazo se mia stili tis times gia validation
       %test_set=cat(1,test_effort_col,test_set);

        [r,c] = size(test_set);
        s=1:r;

       %% TESTING SET ARRAY
        verification(s,iii) = test_set(s);


       if(iii <= times3)
         test_target = test_set_effort(iii);
       %% TESTING TARGET SET ARRAY
         verification_target(1,iii) = test_target;
     end
  end
```

```
 simulation = sim(net,verification);



   % Results from validation
   % Results from simulation / testing
array = [verification_target(:),simulation(:)];
TE = result_calc(array);

dlmwrite('OUT_TESTING.xls',TE,'\t')

H=num2str(effort_neurons);
A=num2str(number_of_hidden_neurons);
R=num2str(number_of_hidden_neurons);
I=num2str(number_of_hidden_neurons);
S=num2str(1);

StrArchitecture = [H,'-',A,'-',R,'-',I,'-',S];

%% graphic result
figure,
subplot(2,2,3),plot(new_target(:),'-r*'),title({'ARTIFICIAL NEURAL
        NETWORK','','==================','','DATA SET: COCOMO','NN
        Architecture: ',StrArchitecture,'','','','Training
        phase'},'Fontsize',10,'Fontweight','bold'),hold on,plot(outp(:),'-b*'),grid on,
subplot(2,2,4),plot(verification_target(:),'-r*'),hold on,plot(simulation(:),'-
        b*'),title('Testing phase','Fontsize',10,'Fontweight','bold'),grid on

gtext('Actual','Fontsize',8)
gtext('Predicted','Fontsize',8)

gtext('Actual','Fontsize',8)
gtext('Predicted','Fontsize',8)
```

B.4 Κώδικας Γενετικού Αλγόριθμου

```
function [best_of_atoms_architecture,fitness_all,best_atom1,best_atom2,fitness_best] =
        ga_loc_predicts_eff(excel_file,training_func,learning_func,performance_func,n
        umber_epochs,repetition_ga,atoms,pc,pm,printgraphs,fixed_rows,use_val)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GENETIC ALGORITHM
% Define domains for the input and the hidden layer
 % Convert to binary representation

 input_layer_min = 1;
 input_layer_max = 5;
 input_no_of_bits = 3;
```

```
 hidden1_layer_min = 2;
 hidden1_layer_max = 20;
 hidden1_no_of_bits = 5;

 hidden2_layer_min = 2;
 hidden2_layer_max = 20;
 hidden2_no_of_bits = 5;

 hidden3_layer_min = 2;
 hidden3_layer_max = 20;
 hidden3_no_of_bits = 5;

% We need total 18 bits to represent the neurons for each layer
 % The first 3 is the input layer neurons and the next 15 is the 3 hidden layers neurons

 % bits used for each atom
 bits = input_no_of_bits + hidden1_no_of_bits + hidden2_no_of_bits +
         hidden3_no_of_bits;

 % Create randomly the first generation of atoms

j=1;
for i=1:atoms
       chromo_num_input(i,j)=round(((input_layer_max-
       input_layer_min)*rand(1,1))+input_layer_min);
       chromo_input(i,j)=str2num(dec2bin(chromo_num_input(i,j)));
end

 for i=1:atoms
       chromo_num_hidden1(i,j)=round(((hidden1_layer_max-
       hidden1_layer_min)*rand(1,1))+hidden1_layer_min);
       chromo_hidden1(i,j)=str2num(dec2bin(chromo_num_hidden1(i,j)));
end

 for i=1:atoms
       chromo_num_hidden2(i,j)=round(((hidden2_layer_max-
       hidden2_layer_min)*rand(1,1))+hidden2_layer_min);
       chromo_hidden2(i,j)=str2num(dec2bin(chromo_num_hidden2(i,j)));
end

 for i=1:atoms
       chromo_num_hidden3(i,j)=round(((hidden3_layer_max-
       hidden3_layer_min)*rand(1,1))+hidden3_layer_min);
       chromo_hidden3(i,j)=str2num(dec2bin(chromo_num_hidden3(i,j)));
end

chromo_population_div =
         [chromo_input,chromo_hidden1,chromo_hidden2,chromo_hidden3];
```

```
% Insert zeros ahead
chromo_input = convert2population(chromo_input,input_no_of_bits);
chromo_hidden1 = convert2population(chromo_hidden1,hidden1_no_of_bits);
chromo_hidden2 = convert2population(chromo_hidden2,hidden2_no_of_bits);
chromo_hidden3 = convert2population(chromo_hidden3,hidden3_no_of_bits);

% Print initial population
%fprintf('The initial population is:')
chromo_population =
        [chromo_input,chromo_hidden1,chromo_hidden2,chromo_hidden3];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Repeat procedure of network creation and simulation
jone=1;
for rep=1:repetition_ga + 1

   for atom=1:atoms
```

<u>Δημιουργία Τεχνητού Νευρωνικού Δικτύου</u>

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%       CREATE THE NETWORK
switch effort_neurons
 case 1,
      net =
        newff(range,[hidden1,hidden2,hidden3,1],{'tansig','tansig','tansig','purelin'},train
        ing_func,learning_func,performance_func);

 case 2,
      net =
        newff(range,[hidden1,hidden2,hidden3,1],{'tansig','tansig','tansig','purelin'},train
        ing_func,learning_func,performance_func);

 case 3,
      net =
        newff(range,[hidden1,hidden2,hidden3,1],{'tansig','tansig','tansig','purelin'},train
        ing_func,learning_func,performance_func);

 case 4,
      net =
        newff(range,[hidden1,hidden2,hidden3,1],{'tansig','tansig','tansig','purelin'},train
        ing_func,learning_func,performance_func);

 case 5,
    net =
        newff(range,[hidden1,hidden2,hidden3,1],{'tansig','tansig','tansig','purelin'},train
        ing_func,learning_func,performance_func);

 otherwise,
```

```
 end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%( ...) RUN FOR ALL BEST ATOMS NN %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find the atom with the best evaluation from each repetition
[max_eval,max_index] = max(eval(:));

best_of_atoms_architecture(counterA,:) = chromo_population(max_index,:);
counterA=counterA+1;

% keep the errors of the best atom from each generation
best_atom1(counterB1,:) = write_train(max_index,:);
best_atom2(counterB2,:) = write_test(max_index,:);
counterB1= counterB1+1;
counterB2= counterB2+1;

   if rep==repetition_ga+1
      break;
   end

% Calculate the total fitness of the population
fitness = sum(eval(:));

% Hold the fitness of all the populations
fitness_all(rep)=fitness;
fitness_best(rep) = max(eval(:));

% Calculate the probability of each atom
prob = eval / fitness;

% Calculate the cummulative probability of each atom
cumul = cumsum(prob);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%         CREATE THE NEW POPULATION
%%%%%%%%%%%%%%%%%%%%%%%%%

% First spin of roulette
first_spin = rand(1,atoms);

% The new population is based on probability
for s=1:atoms
   popul = find(first_spin(s) < cumul);
   new_chromo(s,:) = chromo_population(popul(1),:);
end

%%%%%%%%%%
```

```
% CROSSOVER %
%%%%%%%%%%
% Find random numbers to use in crossover to choose chromos
second_spin = rand(1,atoms);
crossover = find(second_spin < pc);

for i=1:floor(length(crossover)/2)
   % position for the crossover to take place

   cross_pos=round(((bits-1-1)*rand(1,1))+1);
   % swap
   temp = new_chromo(crossover(2*i-1),cross_pos+1:bits);
   new_chromo(crossover(2*i-1),cross_pos+1:bits)=
         new_chromo(crossover(2*i),cross_pos+1:bits);
   new_chromo(crossover(2*i),cross_pos+1:bits)=temp;
end

%%%%%%%%%
% MUTATION %
%%%%%%%%%
mutation_bits = bits*atoms;

mut = rand(atoms,bits);

for i=1:atoms
   for j=1:bits
      if mut(i,j) < pm
         new_chromo(i,j)=num2str(~(str2num(chromo_population(i,j))));
      end
   end
end

% Save the new_chromo for the next repetition
chromo_population = new_chromo;

end

% Write the errors to the files
c=clock;
t1=num2str(c(4));
t2=num2str(c(5));

name_file1 = ['OUT_TRAINING - ',date,' - Time ',t1,'.',t2,'.xls'];
name_file2 = ['OUT_TESTING - ',date,' - Time ',t1,'.',t2,'.xls'];
dlmwrite(name_file1,write_train,'\t')
dlmwrite(name_file2,write_test,'\t')

name_file3 = ['BEST ATOMS - ',date,' - Time ',t1,'.',t2,'.xls'];
dlmwrite(name_file3,[best_atom1 best_atom2],'\t')
```

Συνάρτηση Επιλογής Καλύτερου Ατόμου

```
%-------------------------------------------------------------------
function [topology_bin] =
        choose_best_atom(best_of_atoms_architecture,fitness_all,best_atom_train,best_
        atom_test,fitness_best,multiplication,addition)

counter=1;
[row_test,col_test] = size(fitness_best);

best_fit = fitness_best(1);
topology_bin = best_of_atoms_architecture(1,:);
errors_train = best_atom_train(1,:);
errors_test = best_atom_test(1,:);
fitness_best_atom = fitness_best(1);

for i=1:row_test
   num1 = fitness_best(i);
   if num1>best_fit
      best_fit = num1;
      topology_bin = best_of_atoms_architecture(i,:);
      errors_train = best_atom_train(i,:);
      errors_test = best_atom_test(i,:);
      counter=counter+1;
      fitness_best_atom(counter) = fitness_best(i);
   end
end

inputs = bin2dec(topology_bin(1:3));
hidden1 = bin2dec(topology_bin(4:8));
hidden2 = bin2dec(topology_bin(9:13));
hidden3 = bin2dec(topology_bin(14:18));
output = 1;

inputss=(inputs*multiplication)+addition;

H=num2str(inputss);
A=num2str(hidden1);
R=num2str(hidden2);
I=num2str(hidden3);
S=num2str(output);

StrArchitecture = [H,'-',A,'-',R,'-',I,'-',S];

Errors1 = num2str(errors_train);
Errors2 = num2str(errors_test);
```

```
figure,
subplot(2,2,3),plot(fitness_all(:),'-r*'),title({'RESULT FROM THE GENETIC
ALGORITHM','==========================================','','Best
Atom Architecture:',StrArchitecture,"","','Fitness Of All
Atoms'},'Fontsize',10,'Fontweight','bold'),grid on,
subplot(2,2,4),plot(fitness_best_atom(:),'-b*'),title({","","","","","","","","','NormRMSE
CC MSE RelMAE MAE','-------------------------------------------------
-----------------',' Errors from Training',Errors1,' Errors from Testing',Errors2,"","','Best
Fitness of Atoms'},'Fontsize',10,'Fontweight','bold'),grid on
```

Συνάρτηση Κανονικοποίησης

```
%------------------------------------------------------------------
function outmatrix=rescale(inmatrix,olb,oub,nlb,nub)
% a plot function
% inmatrix : the input matrix
% olb, oub : the old lower and upper bounds
% nlb, nub : the new lower and upper bounds
clc;
outmatrix=((inmatrix-olb)*((nub-nlb)/(oub-olb)))+nlb;

[r c]=size(inmatrix);
if c==3
  outmatrix(:,3)=outmatrix(:,1)-outmatrix(:,2);
End
```

Συνάρτηση Υπολογισμού Σφαλμάτων

```
%------------------------------------------------------------------
function [G] = result_calc(s)
[e q]=size(s);
ms=mean (s(:,1));
mk=mean (s(:,2));
sxx=sum((s(1:e,1)-ms).^2);
syy=sum((s(1:e,2)-mk).^2);
sxy=sum((s(1:e,1)-ms).*(s(1:e,2)-mk));
stdev=std(s(:,1));
sd=sqrt (mean ((s(1:e,1)-s(1:e,2)).^2));
nrms=sd/stdev;
mserr=(mean ((s(1:e,1)-s(1:e,2)).^2));
maerr=(mean (abs(s(1:e,1)-s(1:e,2))));
relmaerr=(mean (abs((s(1:e,1)-s(1:e,2))./s(1:e,1))));
corrc=sxy/sqrt(sxx*syy);

G = [nrms,corrc,mserr,relmaerr,maerr];
```