Ατομική Διπλωματική Εργασία

Πρόβλεψη της Δευτεροταγούς Δομής των Πρωτεϊνών με την βοήθεια Echo State Network και δημιουργία διαδικτυακής εφαρμογής για Αναπαράσταση του προβλήματος

Ειρήνη Παπακώστα

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πρόβλεψη της Δευτεροταγούς Δομής των Πρωτεϊνών με την βοήθεια Echo State Network και δημιουργία διαδικτυακής εφαρμογής για Αναπαράσταση του προβλήματος

Παπακώστα Ειρήνη

Επιβλέπων Καθηγητής:

Δρ. Χρίστος Χριστοδούλου

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

Μάιος 2016

Ευχαριστίες

Για την επίτευξη αυτής της διπλωματικής θα ήθελα να ευχαριστήσω κάποια άτομά τα οποία με βοήθησαν ως προς την ολοκλήρωσή της. Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή Δρ. Χρίστος Χριστοδούλου ο οποίος μέσω του μαθήματός του και της επιλογής μου για την συγκεκριμένη διπλωματική, μου έδωσε την ευκαιρία να ασχοληθώ με τον κλάδο των Τεχνητών Νευρωνικών Δικτύων. Επίσης Θα ήθελα να τον ευχαριστήσω και για την άρτια συνεργασία που είχαμε κατά την διάρκεια της διπλωματικής αυτής.

Επιπλέων θα ήθελα να ευχαριστήσω τον διδακτορικό φοιτητή Μιχάλη Αγαθοκλαίους για την καθοδήγηση και την βοήθειά του καθ΄ όλη την διάρκεια της διπλωματικής μου εργασίας αλλά και τους φίλους και συμφοιτητές μου που μου στάθηκαν σε όλη αυτή την διαδικασία και μετέτρεψαν την εμπειρία αυτή για μένα αξέχαστη.

Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την αμέριστη συμπαράσταση και υπομονή τους που μου ήταν απαραίτητη.

Περίληψη

Η έρευνα αυτή ασχολείται με το πρόβλημα της πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών και με την δημιουργία μιας διαδικτυακής εφαρμογής για την αναπαράσταση του προβλήματος. Το δεύτερο κομμάτι της έρευνας, που αφορά την δημιουργία μιας διαδικτυακής εφαρμογής πραγματοποιήθηκε μέσω της συνεργασίας μου με τον Παναγιώτη Παυλίδη.

Το πρόβλημα της πρόβλεψης του σχήματος των πρωτεϊνών θεωρείται φλέγων ζήτημα για τον κλάδο της ιατρικής, της φαρμακευτικής και της βιοπληροφορικής και αυτό γιατί οι πρωτεΐνες αποτελούν ένα από τα βασικότερα και ζωτικότερα συστατικά του ανθρώπινου οργανισμού. Για να μπορούμε να εξάγουμε την λειτουργικότητα μιας πρωτεΐνης θα πρέπει να γνωρίζουμε το σχήμα της, διαδικασία που θεωρείται πολύπλοκη και χρονοβόρα. Αντίθετα θεωρείται εύκολη η διαδικασία καταγραφής της πρωτεϊνικής ακολουθίας της κάθε πρωτεΐνης δηλαδή της ακολουθίας των αμινοξέων που την αποτελούν. Αυτή η πληροφορία θεωρείται αρκετά χρήσιμη αλλά όχι αρκετή για να μας οδηγήσει προς την εύρεση της τριτοταγούς δομής των πρωτεϊνών. Μια προσέγγιση που χρησιμοποιείται για την λύση αυτού του προβλήματος είναι η πρόβλεψη της δομής της πρωτεΐνης από την πρωτεϊνική ακολουθία με την βοήθεια Τεχνιτών Νευρωνικών Δικτύων.

Με βάση την παραπάνω ιδέα σε αυτή την διπλωματική δημιουργήθηκε ένα Echo State δίκτυο το οποίο χρησιμοποιήθηκε για την πρόβλεψη της δευτεροταγούς δομής των πρωτεϊνών με βάση την πρωτεϊνική τους ακολουθία. Το δίκτυο που υλοποιήθηκε έχει ως αρχικό αποτέλεσμα 68% ακρίβεια. Τα αποτελέσματα αυτά θεωρούνται ικανοποιητικά σε αρχική φάση εφόσον η καλύτερη επίδοση για το συγκεκριμένο πρόβλημα έχει ποσοστό ακρίβειας 76%. Για το συγκεκριμένο δίκτυο υπάρχουν περιθώρια βελτίωσης που μπορούν να επιτευχθούν με μεθόδους όπως το Filtering και Assembles.

Περιεχόμενα

Κεφάλαιο 1: Εισαγωγή

1.1.Ο Ρόλος και ο στόχος της Έρευνας

1.1.Ο Ρόλος και ο στόχος της Έρευνας

Οι πρωτεΐνες περιλαμβάνουν ένα από τα ζωτικότερα συστατικά του ανθρώπινου οργανισμού. Η λειτουργίες και τα οφέλη που προσφέρουν στον άνθρωπο και σε κάθε ζωντανό οργανισμό είναι αναρίθμητα. Παρόλα αυτά η γνώση μας για τις πρωτεΐνες φέρει βελτίωσης και αυτό γιατί ο χειρισμός της καθίσταται ιδιαίτερα δύσκολος. Πιο συγκεκριμένα για να καταφέρουμε να καθορίσουμε τον ρόλο κάποιας πρωτεΐνης θα πρέπει να γνωρίζουμε την τρισδιάστατή της μορφή, διαδικασία που είναι δύσκολο να επιτευχθεί. Αντίθετα η εύρεση της πρωτοταγούς δομής μιας πρωτεΐνης, δηλαδή της πρωτεϊνικής ακολουθίας των αμινοξέων θεωρείται μια εύκολη διαδικασία. Με βάση αυτή την ιδέα δημιουργήθηκαν πολλές προσεγγίσεις για την επίλυση αυτού του προβλήματος οι οποίες αποσκοπούν στην μετατροπή της πρωτεϊνικής ακολουθίας (πρωτοταγής δομή) στην τριτοταγή δομή της πρωτεΐνης. Παρόλο που η πρωτεϊνική ακολουθία αποθηκεύει πληροφορίες όπως για παράδειγμα τα αμινοξέα που την αποτελούν και της σειρά τους στην ακολουθία, εντούτοις αυτές δεν καθίσταται αρκετές για την εύρεση της τριτοταγούς δομής μιας πρωτεΐνης.

Μια προσέγγιση προς την επίλυση του προβλήματος εύρεσης της δομής μιας πρωτεΐνης είναι η δημιουργία Τεχνητών Νευρωνικών Δικτύων. Με βάση την προσέγγιση αυτή δημιουργείτε ένα δίκτυο που σκοπό του έχει την πρόβλεψη της δευτεροταγούς δομής μιας πρωτεΐνης από την πρωτεϊνική της ακολουθία. Η διαίρεση της πρωτεΐνης στις διάφορες δομές της και η διάσπαση του προβλήματος στην πρόβλεψη της κάθε δομής από την πρωτοταγή είναι μια συνήθης μέθοδος.

Η δική μου προσέγγιση προς την επίλυση αυτού του προβλήματος είναι με την δημιουργία ενός Echo State δικτύου.

Κεφάλαιο 2: Web Application

2.1 Απαιτήσεις και Σχεδιασμός Διαδικτυακής εφαρμογής

2.1.1 Σκοπός

Τα τελευταία χρόνια έχουν μελετηθεί αρκετοί αλγόριθμοι, όπως και τεχνικές βελτίωσής τους, γεγονός που δημιούργησε την ανάγκη για δημοσιοποίησης τους. Επομένως, σκοπός της εφαρμογής είναι η δημοσιοποίηση της μέχρι στιγμής έρευνας του Π.Κ. στο πρόβλημα πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών ενώ παράλληλα να δημιουργηθεί ένα διαδικτυακό εργαλείο όπου ο χρήστης να έχει τη δυνατότητα να το χρησιμοποιεί για εξαγωγή των δικών του αποτελεσμάτων, χωρίς αυτός να πρέπει να δημιουργήσει το δικό του δίκτυο. Το σύστημα μας απευθύνεται σε όλους τους ερευνητές που ασχολούνται με το PSSP πρόβλημα, ενώ σε μελλοντικό επίπεδο, τα ενδιαφερόμενα μέρη ενδέχεται να αυξηθούν.

Ακρωνύμια

ΒΔ: Βάση δεδομένων
Χρήστης: πιθανό μέλος του συστήματος
Σύστημα: Η προς ανάπτυξη εφαρμογή
PSSP: (Predict Secondary Structure of Protein) πρόβλεψη της δευτεροταγής δομής των πρωτεϊνών

2.1.2 Product Perspective

2.1.2.1 User Interfaces – Απαιτήσεις

Διασύνδεση εισόδου

Μέσω της διασύνδεσης εισόδου ο χρήστης επιχειρεί την εισαγωγή του, στο σύστημα μας. Θα παρουσιάζεται στο χρήστη μια φόρμα με δυο βασικές επιλογές. Η πρώτη επιλογή θα είναι το "Sign Up", το οποίο παραπέμπει το χρήστη στη διασύνδεση εγγραφής, όπου και θα κάνει εγγραφή για πρώτη φορά. Η δεύτερη επιλογή θα είναι το "Log In", μέσω του οποίου θα συνδέεται, ο χρήστης, στο σύστημα αφού δώσει τα στοιχεία του για επιβεβαίωση μέσω της δικής μας ΒΔ.

Διασύνδεση εγγραφής

Ο χρήστης θα κάνει την εγγραφή του στο σύστημα μέσω αυτής της διασύνδεσης. Θα παραπέμπεται να συμπληρώσει τη φόρμα που παρουσιάζεται στην οθόνη για να γίνει καταγραφή των προσωπικών του δεδομένων στην ΒΔ.

Διασύνδεση εκτέλεσης εκπαιδευμένου δικτύου

Με τη διασύνδεση αυτή, δίνεται στο χρήστη η ευκαιρία να τρέξει ένα ήδη εκπαιδευμένο δίκτυο, κάνοντας χρήση του συνόλου δεδομένων που επιθυμεί. Αρχικά, ο χρήστης θα καλείται να επιλέξει το δίκτυο με το οποίο θέλει να τρέξει τα δεδομένα του και στη συνέχεια θα πρέπει να δώσει το ηλεκτρονικό του ταχυδρομείο ούτως ώστε να του σταλούν τα αποτελέσματα. Σε περίπτωση που δεν επιθυμεί να καταχωρήσει το ηλεκτρονικό του ταχυδρομείο, το σύστημα θα παράγει έναν μοναδικό κωδικό, βάσει του οποίου θα μπορέσει να κατεβάσει, μόνο μια φορά, τα αποτελέσματα του δικτύου στον προσωπικό του λογαριασμό, όταν αυτά είναι έτοιμα.

Διασύνδεση εκπαίδευσης δικτύου

Πέρα από την εκτέλεση ενός ήδη εκπαιδευμένου δικτύου, επιθυμούμε την παροχή της υπηρεσίας εκπαίδευσης δικτύου από το χρήστη, δικαίωμα που θα μπορούν να έχουν μόνο οι εγγεγραμμένοι χρήστες. Πιο συγκεκριμένα, θα πρέπει να παρέχεται στο χρήση η επιλογή του δικτύου που επιθυμεί να δημιουργήσει (π.χ. BRNN, Echo-State, CNN) και βάσει της επιλογής αυτής, να εμφανίζεται σε αυτόν μια φόρμα για να δηλώσει την αρχιτεκτονική των προσαρμόσιμων παραμέτρων (αριθμό κρυφών δικτύων, αριθμό νευρώνων, learning rate κ.τ.λ).

2.1.2.2 Hardware Interfaces

Απαραίτητη προϋπόθεση για τη λειτουργία του συστήματός μας, είναι η σύνδεση του χρήστη με το διαδίκτυο. Σε πρωταρχικό επίπεδο η διαδικτυακή εφαρμογή που θα σχεδιάζουμε πρέπει να τρέχει σε όλους τους φυλλομετρητές. Δεν έχει συμφωνηθεί λειτουργία της εφαρμογή σε κινητές πλατφόρμες, αλλά οι σχεδιαστικές αρχές της εφαρμογής θα πρέπει να επιτρέπουν μια τέτοια προσαρμογή.

2.1.2.3 Software Interfaces

Εφαρμογή για δημιουργία ιστοσελίδων:

Όνομα: IntelliJ IDEA 14

Documentation: https://www.jetbrains.com/idea/

Θεωρήσαμε το IntelliJ IDEA, ως το ιδανικότερο εργαλείο για την ανάπτυξη της διαδικτυακής μας εφαρμογής, λόγω της υποστήριξης όλων των προγραμματιστικών γλωσσών που χρειάζονται για την ανάπτυξη μιας διαδικτυακής εφαρμογής (HTML5, Javascript, jQuery, CSS, PHP – διερμηνέα).

Σύστημα διαχείρισης ΒΔ

Όνομα: Microsoft SQL Server

Documentation: http://msdn.microsoft.com/en-us/library/ms130214.aspx

Ως εργασία πανεπιστημίου, το σύστημα που θα αναπτύξουμε, δεν υποστηρίζεται από οικονομικούς πόρους. Επομένως οδηγούμαστε στην επιλογή του Microsoft SQL Server αφού παρέχεται στην ομάδα μας δωρεάν από το πανεπιστήμιο. Στην περίπτωση που ο πελάτης θελήσει να χρηματοδοτήσει το έργο μας, τότε θα πρέπει να γίνει υλοποίηση της βάση μας σε cloud.

2.1.2.4 Memory Constraints

Η μέγιστη μνήμη RAM που θα απαιτείται για τη λειτουργία της εφαρμογής σε ένα φυλλομετρητή είναι 128 Mb, αφού η ανάπτυξη βασίζεται σε lightweight – client [1].

Η ελάχιστη μνήμη RAM που χρειάζεται ο server είναι 16 Gb, καθώς θα επιτρέπεται να εκτελούνται πέντε δίκτυα παράλληλα.

2.1.2.5 Product Functions

Διαχείριση Δεδομένων

Η εφαρμογή μας, επιτρέπει στο χρήστη να δημιουργήσει τον δικό του λογαριασμό. Όταν ο χρήστης επιλέξει την δημιουργία λογαριασμού τότε θα του δίνεται η δυνατότητα να εισάγει όλα τα προσωπικά του στοιχεία προκειμένου να κάνει έναν αυτόνομο λογαριασμό, τα οποία αποθηκεύονται στην βάση δεδομένων.

Λειτουργία Εκτέλεσης Δικτύων

Η λειτουργία αυτή, τρέχει από τη στιγμή που ο χρήστης επιλέξει την εκτέλεση κάποιου δικτύου, μέχρι να παραδοθεί σε αυτόν το αποτέλεσμα. Λαμβάνει ως είσοδο την επιλογή του χρήστη για το δίκτυο που επιθυμεί να εκτελέσει καθώς επίσης το ηλεκτρονικό του ταχυδρομείο. Αναλαμβάνει την εκτέλεση του εκπαιδευμένου δικτύου στον εξυπηρετητή και ακολούθως αποστολή των αποτελεσμάτων. Τυγχάνει διακλάδωσης όταν ο χρήστης δεν δώσει το ηλεκτρονικό του ταχυδρομείο και δημιουργεί έναν μοναδικό MD5 κωδικό [2] μέσω του οποίο θα γίνει ανάκτηση των δεδομένων.

Λειτουργία Ανάκτησης Αποτελεσμάτων

Χρησιμοποιώντας τη λειτουργία ανάκτησης αποτελεσμάτων, ο χρήστης παρέχει στο σύστημα τον κωδικό που του δόθηκε, εξετάζεται ακολούθως εάν το αρχείο αποτελεσμάτων έχει δημιουργηθεί. Εάν δεν υπάρχει, ζητείται από το χρήστη να επιστρέψει αργότερα ενώ σε αντίθετη περίπτωση το αρχείο κατεβαίνει στον προσωπικό υπολογιστή.

Λειτουργία Εκπαίδευσης Δικτύων

Μια λειτουργία που ο μεγάλος αριθμός παραμέτρων, που μπορεί να επηρεάσουν τη λειτουργία της, την θέτει ταυτόχρονα ως το πιο ευάλωτο χαρακτηριστικό του συστήματος. Θα δέχεται ως είσοδο μια σειρά από παραμέτρους που προσδιορίζουν την αρχιτεκτονική ενός δικτύου, το σύνολο δεδομένων του χρήστη και ακολούθως θα είναι υπεύθυνη για την ανάθεση της δημιουργίας του δικτύου από τον εξυπηρετητή.

2.1.3 Constraints

Το σύστημα σχεδιάστηκε χρησιμοποιώντας το αυξητικό μοντέλο [3], αφού γνωρίζουμε εξ' αρχής όλες τις απαιτήσεις, αλλά λόγο του περιορισμένου χρόνου, προσπαθήσαμε να καλύψουμε όσο το δυνατό περισσότερες από τις απαιτήσεις που προαναφέραμε.

2.1.4 Performance Requirements

Θεωρείται απαραίτητο να είναι δυνατή η ταυτόχρονη πρόσβαση των χρηστών σε όλες τις λειτουργιές του συστήματος. Αν οι αριθμός των διεργασιών που εμπλέκουν επικοινωνία με τη ΒΔ, την ίδια χρονική στιγμή, ενδέχεται να παρουσιαστεί μείωση στην απόδοση του συστήματος.

Το λογισμικό θα ανταποκρίνεται στις εντολές του χρήστη σε χρόνο λιγότερο των 5 δευτερολέπτων με μέγιστη καθυστέρηση 7 δευτερόλεπτα [1].

Τέλος, για τη ορθή διαχείριση των υπολογιστικών πόρων, δεν θα επιτρέπεται η εκτέλεση περισσότερων από πέντε διεργασιών, την ίδια χρονική στιγμή, στον διακομιστή.

2.1.5 Maintainability

Οι φάσεις που προηγούνται αποσκοπούν στην αποδοτική λειτουργία του συστήματος ούτως ώστε να μεγιστοποιήσουν τον κύκλο ζωής του. Παρόλα αυτά, η φάση της συντήρησης μπορεί να προκληθεί από πολλαπλούς παράγοντες, όπως:

Διόρθωση των bugs

Το μέγεθος του συστήματος δεν πρέπει να υποτιμηθεί. Για την αντιμετώπιση του κινδύνου αυτού, οι απαιτήσεις του συστήματος υλοποιούνται με σειρά προτεραιότητας της λειτουργικότητας τους.

Εμφάνιση νέων απαιτήσεων

Για να διασφαλίσουμε τη δια βίου εξέλιξη του συστήματος, θα πρέπει να είναι αρκετά ευπροσάρμοστο στην εμφάνιση νέων απαιτήσεων. Η εμφάνιση νέας απαίτησης μπορεί να επηρεάσει τόσο τις διεπαφές του χρήστη, όσο και τον λογικό σχεδιασμό του συστήματος.

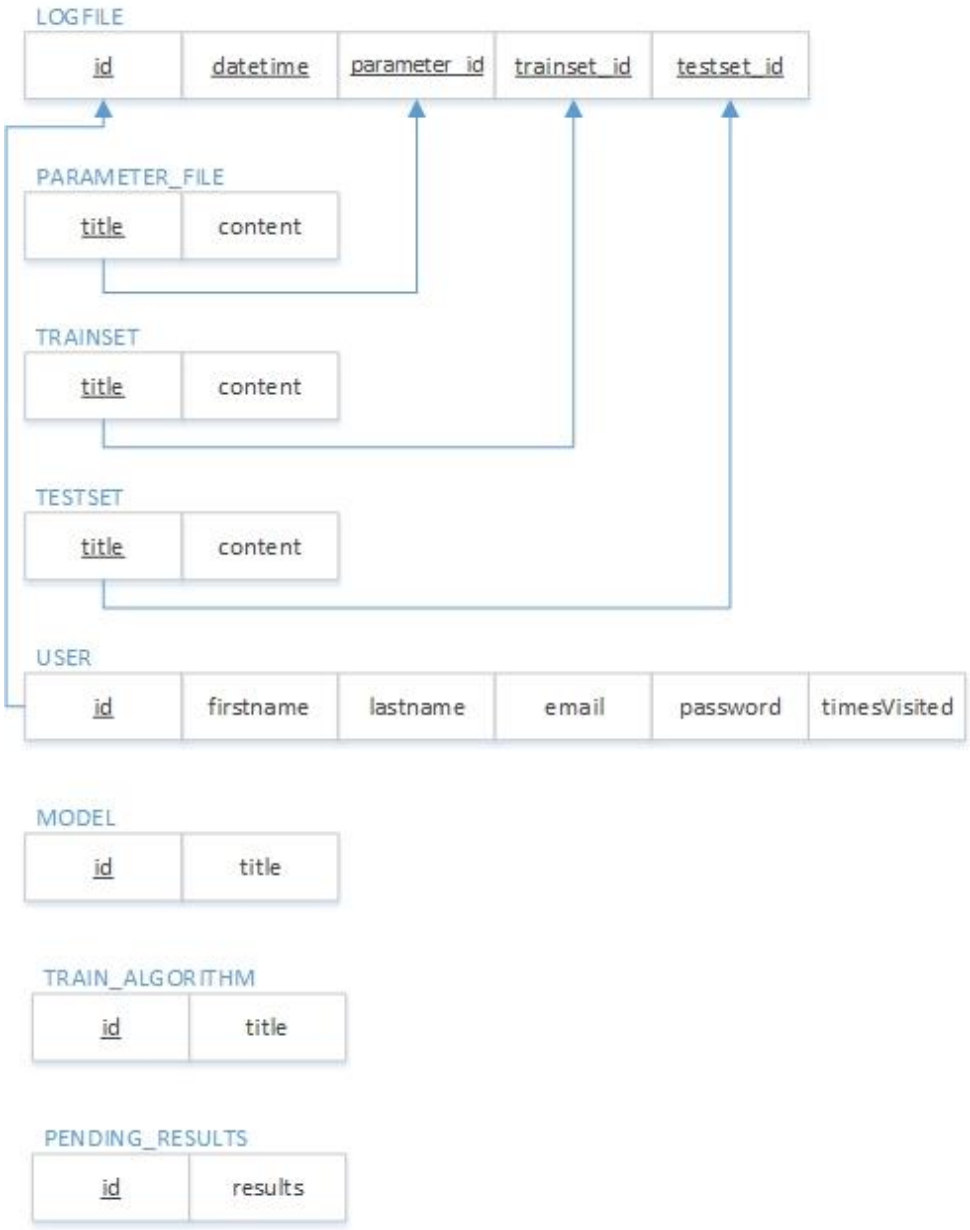
Απόδοση ΒΔ

Η εξελικτική πορεία της τεχνολογίας μπορεί να οδηγήσει στην ανάγκη για αλλαγή της ΒΔ, λόγω της ανικανότητας της τρέχουσας να εξυπηρετήσει τις αναδυόμενες ανάγκες.

Ασφάλεια

Η λειτουργία εκπαίδευσης δικτύων εγείρει νέα ζητήματα ασφάλειας και αξιοπιστίας του συστήματος, τομέας που χρίζει ιδιαίτερης προσοχής.

2.1.6 Logical Database Requirements



Σχήμα 1: Σχεσιακή Αναπαράσταση Βάσης Δεδομένων της διαδικτυακής Εφαρμογής

2.2 Υλοποίηση

2.2.1 Χρήση πακέτου ανοικτού κώδικα XAMMP

Η διαδικτυακή εφαρμογή που κληθήκαμε να υλοποιήσουμε προϋποθέτει την δημιουργία ΒΔ για την αποθήκευση δεδομένων απαραίτητων για την φόρτωση της ιστοσελίδας και δεδομένων που ο χρήστης καταχωρεί στην εφαρμογή (στοιχεία εγγραφής του χρήστη, αποτελέσματα των δικτύων που τρέχει).

Χρησιμοποιήσαμε το πακέτο ανοικτού κώδικα XAMMP [4] το οποίο μας προφέρει την δυνατότητα δημιουργίας Apache server ως localhost στον προσωπικό μας υπολογιστή.

2.2.2 Χρήση γλώσσας προγραμματισμού PHP

Το πακέτο ανοικτού κώδικα XAMPP μας παρέχει τη δυνατότητα γραφής στην γλώσσας PHP. Η PHP είναι γλώσσα προγραμματισμού που χρησιμοποιείται για την δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών με δυναμικό περιεχόμενο. Ουσιαστικά μας βοήθησε στην διαλειτουργικότητα μεταξύ της διεπαφής και της βάσης δεδομένων. Οι ακόλουθες λειτουργίες που υλοποιήθηκαν στην διαδικτυακή εφαρμογή εκτελούνται με κώδικα γραμμένο σε αυτή την γλώσσα:

2.2.2.1 Φόρτωση της διαδικτυακής εφαρμογής στο πρόγραμμα περιήγησης του χρήστη:

Η διαδικτυακή εφαρμογή είναι αποθηκεμένη σε ένα διακομιστή (Server). Όταν ο χρήστης, επισκέπτης αιτηθεί την ιστοσελίδα δηλαδή την καλέσει με την απαραίτητη διεύθυνση-URL αυτή περνά από κάποια επεξεργασία. Αρχικά από τον PHP κώδικα παράγεται το περιεχόμενό της σε κείμενο HTML και ακολούθως στέλνεται σε πραγματικό χρόνο στο πρόγραμμα περιήγησης του χρήστη (user's browser).

2.2.2.2 Εγγραφή του χρήστη στην διαδικτυακή εφαρμογή:

Ό χρήστης είναι απαραίτητο να καταχωρίσει κάποια στοιχεία του στο σύστημα για να μπορεί να εγγραφεί. Όταν η διαδικασία αυτή τελειώσει στα στοιχεία που καταχωρήθηκαν ανατίθενται κάποιες μεταβλητές οι οποίες στέλνονται με την μέθοδο Post στον διακομιστή. Ακολούθως με τον κώδικα PHP δημιουργούνται ερωτήματα (query) σε μορφή MySQL για να καταχωρηθούν τα δεδομένα στην Βάση Δεδομένων.

2.2.2.3 Σύνδεση χρήστη στο σύστημα (Log In)

Ο χρήστης για να ενωθεί στην διαδικτυακή εφαρμογή θα πρέπει να καταχωρήσει το χαρακτηριστικό όνομα χρήστη(Username) και το κωδικό πρόσβασης(Password). Αρχικά όπως και για τα στοιχεία εγγραφής ο κωδικός πρόσβασης και το χαρακτηριστικό όνομα του χρήστη μετατρέπονται σε μεταβλητές και με την μέθοδο Post μεταφέρονται στον διακομιστή. Ακολούθως δημιουργείται με την γλώσσα PHP ένα ερώτημα σε μορφή MySQL το οποίο αναζητεί τον χαρακτηριστικό όνομα χρήστη και σε περίπτωση που περιέχεται στην βάση, ελέγχει τον κωδικό. Η απάντηση αποστέλνεται επίσης με κώδικα PHP.

2.2.2.4 Δοκιμή εκτέλεσης Δικτύου

Όταν ο χρήστης επιλέξει να τρέξει ένα δίκτυο με κάποια συγκεκριμένη είσοδο-πρωτεϊνική ακολουθία που ο ίδιος καταχωρεί, τότε όπως και στις προηγούμενες δύο περιπτώσεις επικοινωνούμε με την βάση μέσω κώδικα PHP. Έτσι θα αναγνωριστεί πιο δίκτυο επιθυμεί ο χρήστης να τρέξει και θα σταλεί η πρωτεϊνική ακολουθία στην είσοδο του δικτύου αυτού.

2.2.3 Χρήση PhpMyAdmin- MySQL

Το πακέτου ανοικτού κώδικα XAMPP [4] επίσης μας παρέχει ένα περιβάλλον το οποίο ονομάζεται phpMyAdmin [5]. Με την βοήθεια του περιβάλλοντος αυτού μπορούμε να διαχειριστούμε με ευκολία βάσεις δεδομένων τύπου MySQL. Εμείς χρησιμοποιήσαμε αυτό το περιβάλλον για να δημιουργήσουμε την Βάση Δεδομένων για την διαδικτυακή εφαρμογή.

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων και την χρησιμοποιήσαμε για την εύρεση, την εισαγωγή και την διαγραφή δεδομένων από και προς την βάση. Η κάθε μια λειτουργία μεταφράστηκε σε ερωτήματα (queries) γραμμένα σε MySQL τα οποία εκτελούνται με την βοήθεια της PHP .

2.2.3.1 Σύνδεση στην Βάση δεδομένων:

Για να συνδεθούμε με την βάση δεδομένων θα πρέπει να ορίσουμε κάποιες μεταβλητές που αντιπροσωπεύουν αυτή την βάση. Οι μεταβλητές που πρέπει να οριστούν είναι το servername στην δική μας περίπτωση είναι localhost εφόσον μετατρέψαμε τον προσωπικό μας υπολογιστή σε διακομιστή, το username και password για περιορισμένη πρόσβαση στην βάση και το dbname (data Base Name) το όνομα της βάση Δεδομένων. (Σχήμα 2)

```php
<?php

$servername = "xxxx";
$username = "xxxx";
$password = "xxxx";
$dbname = "xxxx";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}
catch(PDOException $e)
{
    echo "Connection failed: " . $e->getMessage();
}

```

Σχήμα 2: Κώδικας γραμμένος στην γλώσσα php που επιτρέπει την σύνδεση στην Βάση Δεδομένων. Βρίσκεται στο MySqlConnect.php.

2.2.3.2 Εισαγωγή δεδομένων στην βάση:

Για να εισάξουμε τα δεδομένα στην βάση προϋποθέτεται να συνδεθούμε στην Βάση Δεδομένων. Για την σύνδεση στην βάση χρησιμοποιούμε τον πιο πάνω κώδικα που βρίσκεται στο αρχείο MySqlConnect.php. Στο Σχήμα 3 φαίνεται ο κώδικας που εισάγει μια γραμμή στον πίνακα model όπου στα πεδία 'id'και 'title' καταχωρούνται αντίστοιχα το '3' και το 'Bidirectional Recurrent Neural Network'.

```
8   require "MySqlConnect.php";
9   $Model = $conn->prepare("INSERT INTO `model` (`id`, `title`)
10                      VALUES ('3', 'Bidirectional Recurrent Neural Network')");
11  $Model->execute();
12
13  $ResultModel = $Model->fetchAll(PDO::FETCH_ASSOC);
14
```

Σχήμα 3: Εισαγωγή μιας γραμμής στον πίνακα model στην Βάση Δεδομένων.

2.2.3.3 Εύρεση δεδομένων από την βάση:

Για την εύρεση δεδμένων επό την βάση θα πρέπει επίσης αρχικάωνα συνδεθούμε σε αυτή χρησιμοποιόντας τον κώδικα στην σελίδα MySqlConnect.php. Στο σχήμα 4 φαίνεται ο κώδικας που αποθηκεύει στην μεταβλητή $ResultModel τα αποτελέσματα από την εύρεση όλων των δεδομένων από το πεδίο title από τον πίνακα model.

```
 8  require "MySqlConnect.php";
 9  $Model = $conn->prepare("SELECT title FROM model");
10  $Model->execute();
11
12  $ResultModel = $Model->fetchAll(PDO::FETCH_ASSOC);
13
```

Σχήμα 4: Εύρεση όλων των δεδομένων από το πεδίου title του πίνακα model.

2.2.4 Δημιουργία Διεπαφών Διαδικτυακής Εφαρμογής

2.2.4.1 Εργαλεία και γλώσσες που χρησιμοποιήθηκαν για την διαμόρφωση των διεπαφών

Για την δημιουργία των διεπαφών χρησιμοποιήσαμε ένα αποκριτικό πλαίσιο διεπαφής από την ιστοσελίδα http://html5up.net/. Επίσης για την διαμόρφωση των διεπαφών στιλιστικά, λειτουργικά και για την δημιουργία αντικείμενα που θα περιέχουν χρησιμοποιήσαμε τις ακόλουθες γλώσσες και εργαλεία:

Χρήση γλώσσας HTML

Για την δημιουργία των διεπαφών χρησιμοποιήσαμε την γλώσσα HTML (HyperText Markup Language) η οποία είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες. Τα αντικείμενα και τα στοιχεία τα οποία ονομάζονται ετικέτες (tags) είναι τα βασικά δομικά στοιχεία των ιστοσελίδων όπως για παράδειγμα κουμπί, παράγραφος, κεφαλίδα, σύνδεσμος κ.α.

Χρήση της γλώσσας CSS

Η CSS (Cascading Style Sheets) γλώσσα χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός αρχείου HTML. Διαμορφώνει στιλιστικά την ιστοσελίδα δηλαδή διαμορφώνει χαρακτηριστικά αντικειμένων, χρώματα, στοίχιση κ.α.

Χρήση της γλώσσας Java Script και JQuery

Για την λειτουργικότητα των διεπαφών χρησιμοποιήσαμε την διερμηνευμένη γλώσσα προγραμματισμού JavaScript. Ουσιαστικά η γλώσσα αυτή βοήθησε στην επικοινωνία μεταξύ των αντικειμένων της ιστοσελίδας και του χρήστη. Δηλαδή για να εναλλάσσουν δεδομένα ασύγχρονα και να συνεπώς να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται. Για παράδειγμα η υλοποίηση της λειτουργικότητας ενός κουμπιού.

JQuery είναι βιβλιοθήκη της Java Script και χρησιμοποιείται για τον ίδιο σκοπό όπως και η Java Script.

Χρήση Ajax

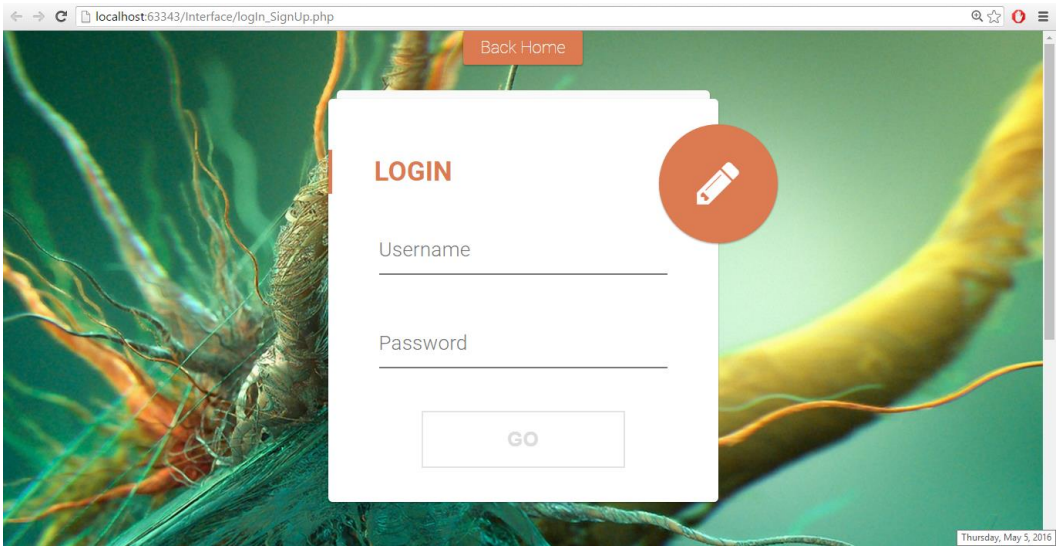
Χρησιμοποιήσαμε Ajax για να μπορούμε να ανταλλάζουμε δεδομένα με τον διακομιστή χωρίς να ξαναφορτώνεται η σελίδα. Ajax χρησιμοποιήσαμε στην λειτουργία όπου έπρεπε να σταλεί ένα email και να παραχθεί ένας κωδικός.

Χρήση Bootstrap framework

Το Bootstrap framework είναι μια βιβλιοθήκη από την οποία μπορείς να επιλέξεις αντικείμενα για την ιστοσελίδα σου ούτως ώστε αυτή να είναι αποκριτική (responsive). Χρησιμοποιήσαμε το Bootstrap framework κυρίως όταν θέλαμε να δημιουργήσουμε modals.(Contact us modal, Email-Generate Code modal)

2.2.4.2 Διεπαφή Σύνδεσης και Εγγραφής του Χρήστη

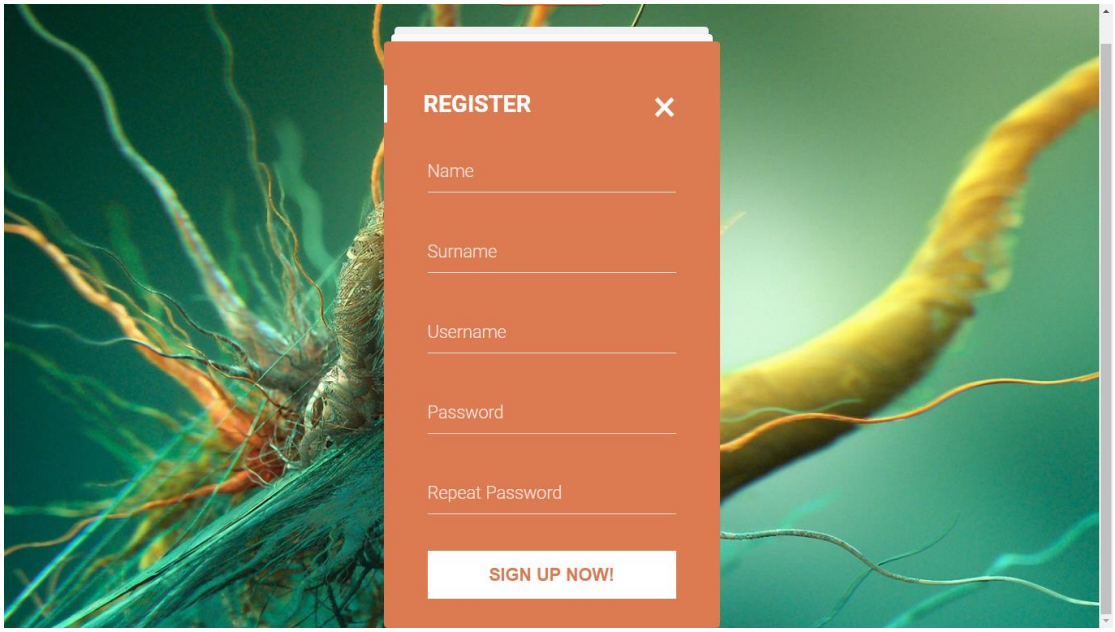
Η Διεπαφή Σύνδεσης Χρήτη επιτρέπει στον χρήστη να συνδεθεί στην διαδικτυακή εφαρμογή χρησιμοποιόντας το χαρακτιρικό όνομα χρήστση (Username) και τον κωδικό πρόσβασής(Password) του.



Σχήμα 5: Διεπαφή Σύνδεσης Χρήστη

Πατώντας το πορτοκαλί κουμπί με το μολύβι ο χρήστης έχει την δυνατότητα εάν δεν είναι εγγεγραμμένος στο σύστημα να εγγραφεί. Όπως φαίνεται και στο Σχήμα 6 ο χρήστης δα πρέπει να διαθέσεις κάποια από τα προσωπικά του στοιχεία για να μπορεί να πραγματοποιηθεί η εγγραφή του. Με το πάτημα του κουμπιού Sign Up Now! Θα γίνεται έλεγχος στην φόρμα αν όλα τα παιδία είναι συμπληρωμένα και με κατάλληλα στοιχεία και

ο χρήστης θα μεταφέρεται στην διεπαφή σύνδεσης χρήστη. Σε αντίθετη περίπτωση θα εμφανίζεται μήνυμα λάθους.



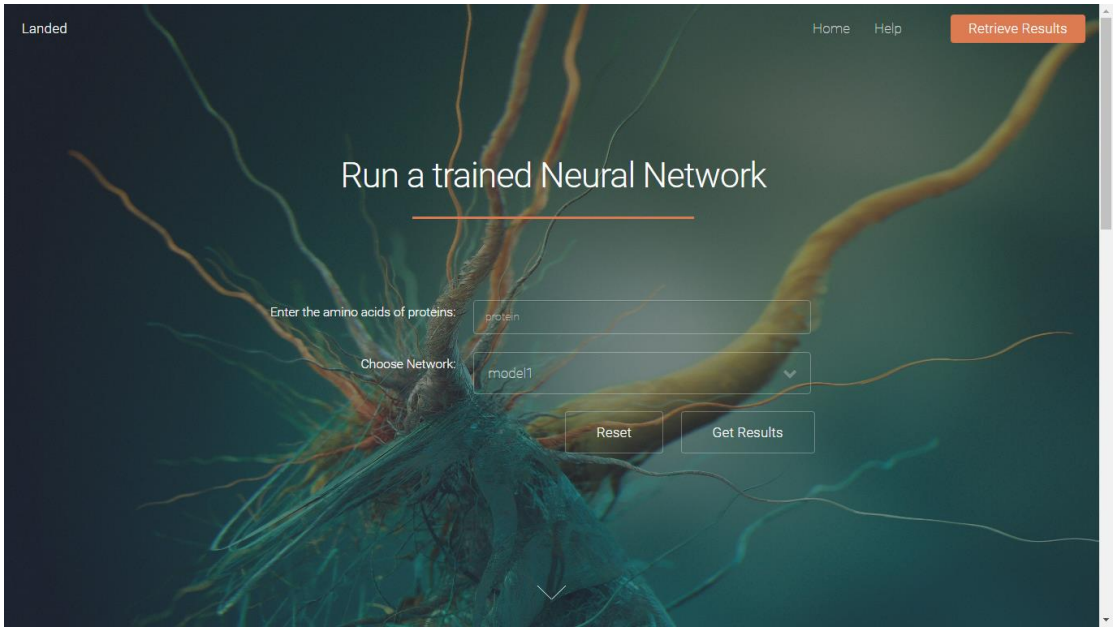
Σχήμα 6: Διεπαφή Εγγραφής Χρήστη

2.2.4.3 Διεπαφή Κεντρικής Σελίδας

Η κεντρική σελίδα χωρίζεται σε 3 κύρια παράθυρα στα οποία ο χρήστης μεταβαίνει μετακινόντας τον καίρσορά του προς τα κάτω ή πατώντας στο τοξο που εμφανίζεται στο κάτω μέρος κάθε παραθύρου.

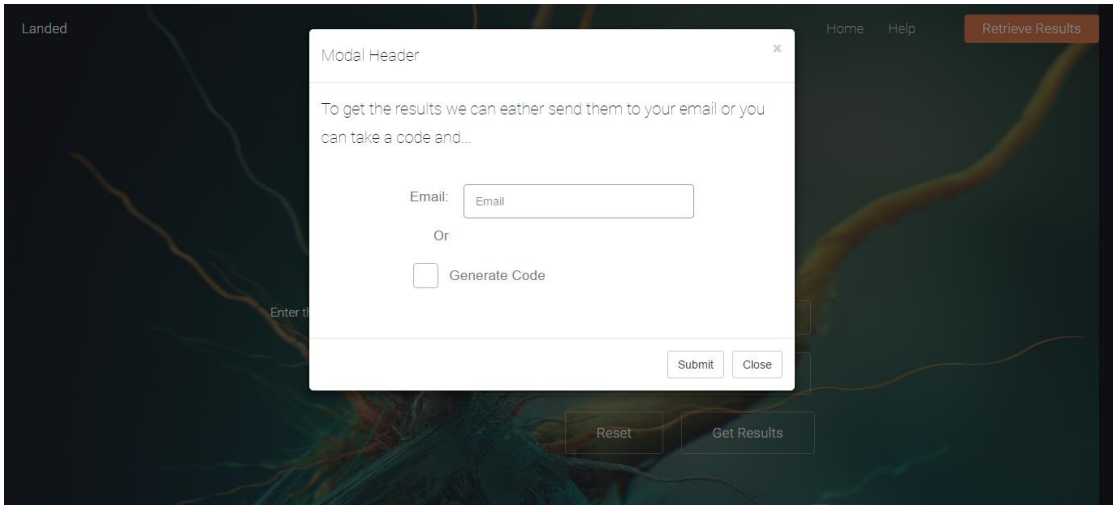
Πρώτο Παράθυρο

Στο πρώτο παράθυρο ο χρήστης θα μπορεί να τρέξει ένα είδη εκπεδευμένο Τεχνητό Νευρωνικό δίκτυο εισάγοντας την πρωτεϊνική ακολουθία με την οποία θέλει να τρέξει το δίκτυο και το επιλέγοντας ένα μοντέλο δικτύου.(Σχήμα 7)



Σχήμα 7: Κύρια σελίδα - Πρώτο Παράθυρο

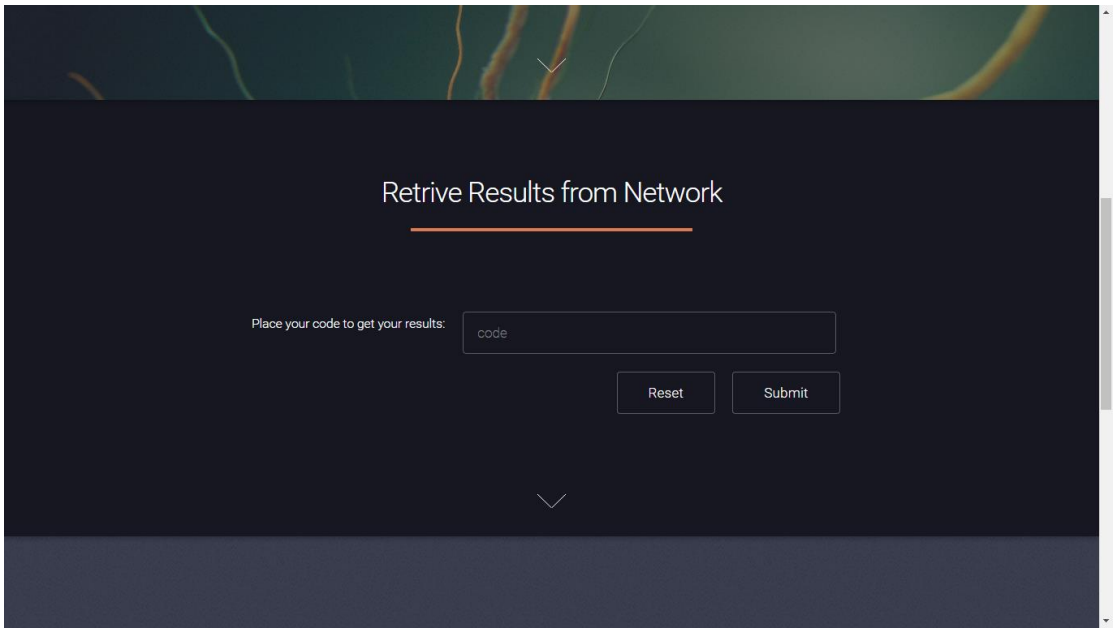
Ακολούθως αν τα δύο πεδία δεν είναι άδια πατώντας το κουμπί Get Results μεταφέρεται στο modal όπως φαίνεται στο Σχήμα 8. Σε αντίθετη περίπτωση εμφανίζεται μήνυμα λάθους. Ο χρήστης σε αυτή τη φόρμα καλείτε αν θέλει να δώσει ένα email του στο οποίο θα του σταλούν τα αποτελέσματα του δικτύου που έτρεξε. Αν δεν επιθημεί να δώσει κάποιο email τότε με το πάτημα του κουμπιού Submit παράγεται ένας κωδικός για να έχει πρόσβαση μέσω αυτού στα αποτελέσματα του δικτύου που έτρεξε.



Σχήμα 8: Καταχώρηση email ή παραγωγή κωδικού

Δεύτερο Παράθυρο

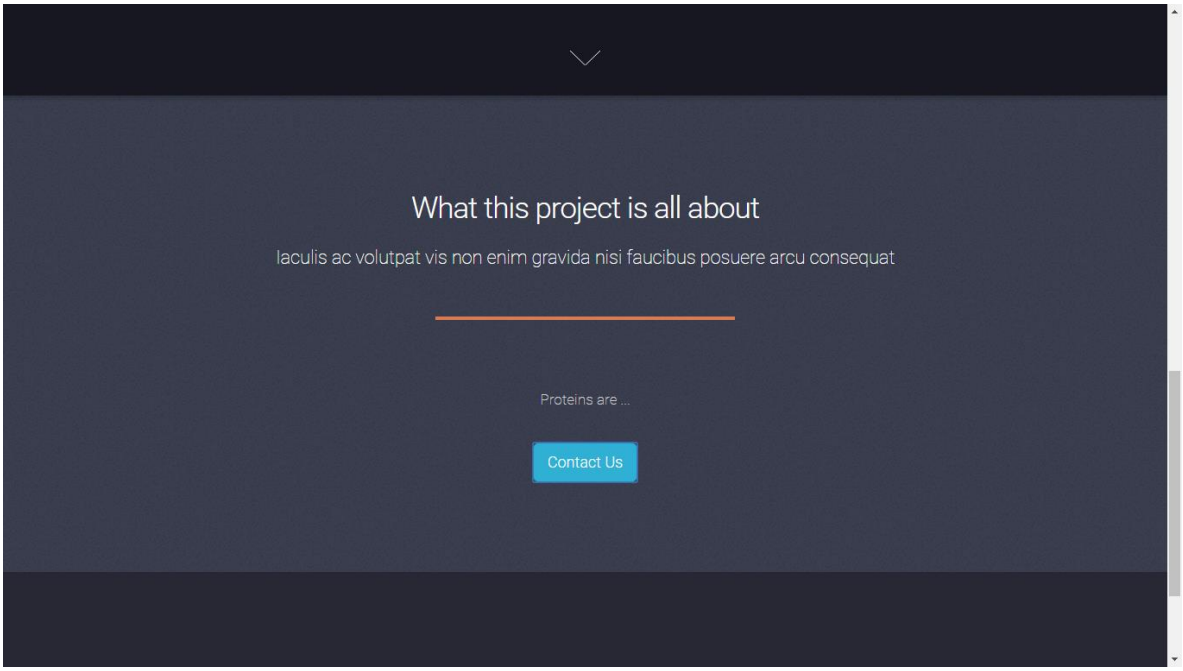
Όπως έχει ααφερθεί πιο πάνω αν ο χρήστης δεν επέλεξε να του σταλούν τα αποτελέσματα μέσω email θα του παραχωρείτε ένας κωδικός για να μπορεί να τα παραλάβει. Σε αυτό το παράθυρο θα καταχωρεί τον κωδικό και θα παίρνει τα αποτελέσματα. Σε περίπτωση που τα αποτελέσματα δεν είναι έτημα θα εμφανίζεται μήνυμα που να τον ενημερώνει. (Σχήμα 9)



Σχήμα 9: Παραλαβή Αποτελεσμάτων - Παράθυρο δεύτερο

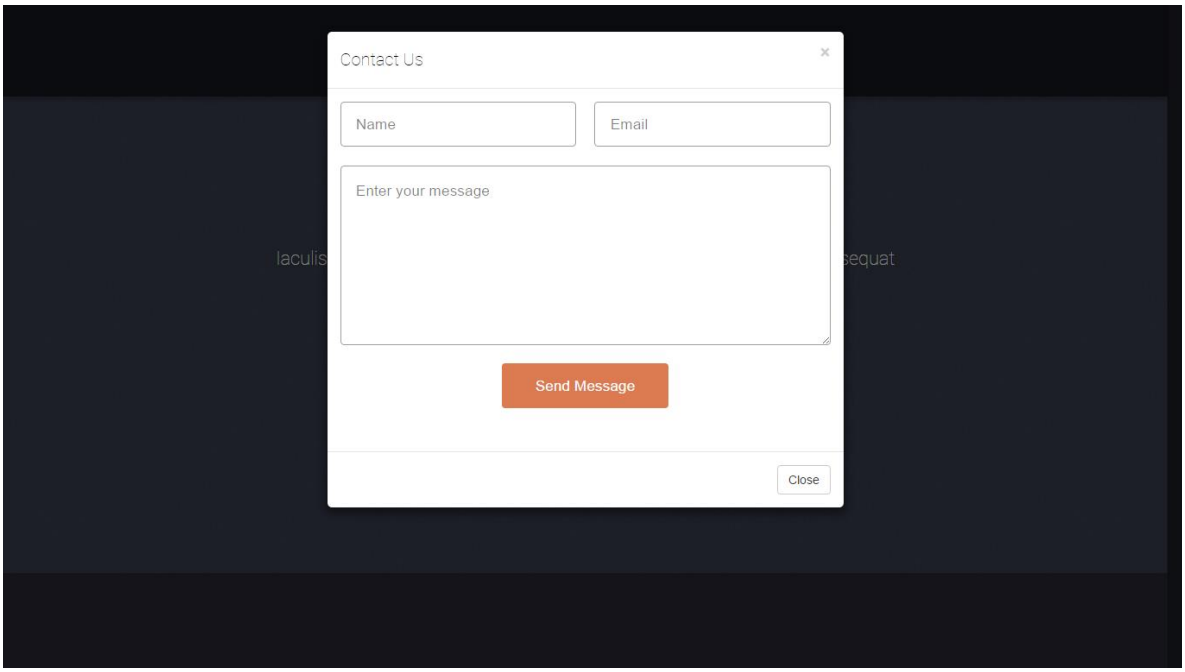
Τρίτο Παράθυρο

Στο τρίτο παράθυρο ο χρήστης θα έχει την δυνατότητα να μάθει περισσότερες πληροφορίες για την διαδικτυακή εφαρμογή και για το PSSP πρόβλημα.



Σχήμα 10: Πληροφορίες για το PSSP πρόβλημα - Τρίτο Παράθυρο

Επίσης με το πάτημα του κουμπιού Contact Us θα εμφανίζεται ένα modal στο οποίο ο χρήστης συμπληρώνοντας τα παιδία name,email και γράφοντας ένα κείμενο θα μπορεί να επικοινωνήσει μαζί μας.



Σχήμα 11: Contact Us

2.2.5 Back End

2.2.5.2 Αποστολή email

Εκτός από την αλληλεπίδραση του χρήστη με τις διεπαφές της διαδικτυακής εφαρμογής, οι οποίες σχολιάστηκαν πιο πάνω, η επικοινωνία του χρήστη και διαδικτυακής εφαρμογής μπορεί να επιτευχθεί και μέσω της αποστολής email.

Από την πλευρά του χρήστη η διαδικασία αυτή προστέθηκε για να μπορεί ο ίδιος ανά πάσα χρονική στιγμή να επικοινωνήσει με τους συντηρητές της διαδικτυακής εφαρμογής για την επίλυση οποιονδήποτε απορίων ή για την αποστολή σχολίων του. Με βάση το δεύτερο παράθυρο της κεντρικής σελίδας ο χρήστης έχει την δυνατότητα να μεταφερθεί σε ένα model το οποίο περιέχει μια φόρμα. Με την συμπλήρωση της φόρμας (Εικόνα 2.10) ο χρήστης θα μπορεί να στείλει email με το κείμενο που επιθυμεί.

Από την πλευρά της διαδικτυακής εφαρμογής η λειτουργία αυτή αποτελεί μια από τις δύο μεθόδους αποστολής των αποτελεσμάτων της εκτέλεσης των εκπαιδευμένων Τεχνητών Νευρωνικών Δικτύων που είναι διαθέσιμα στην εφαρμογή. Ο χρόνος που απαιτείται για να αποσταλεί το email με τα αποτελέσματα εξαρτάται απόλυτα από τον χρόνο που χρειάζεται το επιλεγμένο από τον χρήστη τεχνητό νευρωνικό δίκτυο για να τρέξει την πρωτεϊνική ακολουθία που εισήγαγε ο χρήστης. Ακολούθως αυτόματα θα στέλνεται ένα email στο χρήστη με τα αποτελέσματα.

Για την υλοποίηση της διαδικασίας αποστολής του email χρησιμοποιήθηκε το PHPMailer, πακέτο το οποίο αποτελείται από κλάσεις που διαθέτουν τα πλήρη χαρακτηριστικά για την δημιουργία και αποστολή email με την βοήθεια της γλώσσας προγραμματισμού PHP [39]. Η επιλογή του πακέτου αυτού έγινε επειδή η γλώσσα προγραμματισμού PHP έχει χρησιμοποιηθεί κατά κόρον για την επικοινωνία των διεπαφών της διαδικτυακής εφαρμογής και της Βάσης Δεδομένων. Δηλαδή την επικοινωνία μεταξύ Frond-End και Back-End.

Με βάση το πακέτο αυτό δημιουργείται ένα αυτοματοποιημένο μήνυμα το οποίο περιλαμβάνει τα αποτελέσματα του ΤΝΔ που εκτελέστηκε. Επίσης να σημειωθεί ότι για την επιτυχή αποστολή του email έπρεπε πρώτα να αποφευχθούν ορισμένα προβλήματα που

είχαμε με τις ρυθμίσεις του localhost. Προβλήματα που έχουν σχέση με την προσβασιμότητα μας σε πλατφόρμες αποστολής email όπως για παράδειγμα Gmail.

2.2.5.2 Εκτέλεση δικτύου με δημιουργία MD5 κωδικού

Η εξήγηση των back-end λειτουργιών θα περιγραφούν με ψευδοκώδικα για σαφέστερη κατανόηση των βημάτων στο παρόν αλλά και σε μεταγενέστερο στάδιο.

Στην περίπτωση που χρήστης δεν δώσει το email του, για αποστολή αποτελεσμάτων γίνονται τα ακόλουθα:

1. Γίνεται έλεγχος στη βάση δεδομένων για το κατά πόσο η αίτηση θα ξεπεράσει το μέγιστο αριθμό παράλληλων διεργασιών στο διακομιστή. Εάν η απάντηση είναι αρνητική, τότε ζητείται από το χρήστη να δοκιμάσει αργότερα.
2. Ασύγχρονο ajax call στη PHP για λήψη του μοναδικού MD5 κωδικού. Στο σημείο αυτό παράγεται ο κωδικός. Παράλληλα, δημιουργείται εγγραφή στη ΒΔ με μορφή (κωδικός, αποτέλεσμα), όπου στο πεδίο κωδικός εισάγεται ο MD5 κωδικός ενώ το πεδίο των αποτελεσμάτων είναι κενό.

Όταν τα αποτελέσματα είναι έτοιμα, γίνεται κλήση στη ΒΔ, για συμπλήρωση της εγγραφής (κωδικός, αποτέλεσμα).

Κατά την προσπάθεια του χρήστη να λάβει τα αποτελέσματα, γίνονται τα παρακάτω:

1. Έλεγχος εάν η εγγραφή στη ΒΔ έχει ολοκληρωθεί. Σε περίπτωση που το πεδίο "αποτελέσματα" είναι κενό, ζητείται από το χρήστη να δοκιμάσει αργότερα.
2. Εάν το πεδίο είναι συμπληρωμένο, τότε δημιουργείται αρχείο που να περιέχει τα αποτελέσματα και ο φυλλομετρητής ζητά από το χρήστη να κατεβάσει το αρχείο.
3. Αμέσως, η εγγραφή του αποτελέσματος από τη ΒΔ μας, διαγράφεται.

Κεφάλαιο 3: Μελέτη του προβλήματος των πρωτεϊνών

3.1 Βιολογικό Υπόβαθρο

3.1.1Βιολογικός ρόλος των Πρωτεϊνών

Οι πρωτεΐνες είναι ουσίες σημαντικές για κάθε ζωντανό οργανισμό. Άλλωστε το όνομά τους υποδεικνύει αυτό τους το ρόλο. Ο αριθμός των διαφόρων πρωτεϊνών που υπάρχουν στον άνθρωπο υπερβαίνει τις 30.000 [44]. Ο αριθμός αυτός φαινομενικά είναι μεγάλος αλλά θα πρέπει κανείς να αναλογιστεί ότι κάθε μια πρωτεΐνη επιτελεί και μια συγκεκριμένη λειτουργία καθώς επίσης ότι ο αριθμός των λειτουργιών τις οποίες οι πρωτεΐνες επιτελούν είναι τεράστιος. Μερικές από τις λειτουργίες του ανθρώπινου οργανισμού στις οποίες συμμετέχουν οι πρωτεΐνες είναι οι ακόλουθες:

Συσταλτικές Πρωτεΐνες:

Το κύριο συστατικό του μυϊκού ιστού είναι οι πρωτεΐνες. Τα μυοϊνίδια, τα πραγματικά συσταλτά στοιχεία του μυός , αποτελούντα κατ' εξοχή από χοντρές ίνες πρωτείων ακτίνες της πρωτεΐνης μυοσύνης και λεπτές ίνες των πρωτείων ακτίνης και τροπομυοσίνης. Επίσης η πρωτεΐνη τοπαμίνη βοηθά στην ρύθμιση της μυϊκής συστολής.

Ορμονική δράση Πρωτεϊνών:

Κάποιες άλλες πρωτεΐνες έχουν ορμονική δράση. Για παράδειγμα η ινσουλίνη και η γλυκαγόνη είναι ορμόνες πεπτιδικής φύσεως που εκκρίνονται από το πάγκρεας και ρυθμίζουν την συγκέντρωση του σακχάρου στο αίμα.

Μεταφορικός ρόλος Πρωτεϊνών:

Άλλες πρωτεΐνες έχουν μεταφορικό ρόλο, όπως για παράδειγμα η αιμοσφαιρίνη η οποία είναι υπεύθυνη για την μεταφορά οξυγόνου στο αίμα και η μυοσφαιρίνη η οποία είναι υπεύθυνη για την πρόσληψη οξυγόνου στους μύες.

Πρωτεΐνες - Ένζυμα:

Τα διάφορα ένζυμα είναι σώματα με πρωτεϊνική δομή που βοηθούν στις βιοχημικές αντιδράσεις του οργανισμού. Αναφέρονται συχνά ως καταλύτες επειδή επιταχύνουν τις χημικές αντιδράσεις. Παραδείγματα ενζύμων είναι η λακτάση και η πεψίνη.

Πρωτεΐνες - Αντισώματα:

Τα αντισώματα αποτελούν μια μορφή πρωτεΐνης και παράγονται από τον ίδιο τον οργανισμό. Έχουν δομή τέτοια, που τις καθιστά ειδικές να δεσμεύουν και να εξουδετερώνουν αντιγόνα, ξένους εισβολείς, που εισχωρούν στον οργανισμό. Είναι ο τρόπος με τον οποίο ο οργανισμός του ανθρώπου ή των ζώων αμύνεται από ιούς.

Αποθηκευτικός ρόλος πρωτεϊνών:

Βοηθούν στην αποθήκευση αμινοξέων η άλλων ουσιών όπως ασβέστιο. Παραδείγματα πρωτεϊνών με αποθηκευτικό ρόλο είναι η καζεΐνη που είναι πρωτεΐνη του γάλακτος και έχει ως ρόλο την αποθήκευση ασβεστίου, και η ωαλβουμίνη, που είναι το κύριο συστατικό στο ασπράδι του αβγού και αποτελεί πηγή αμινοξέων για το αναπτυσσόμενο έμβρυο.

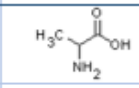
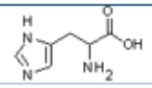
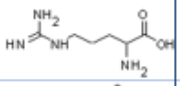
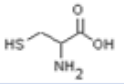
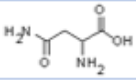
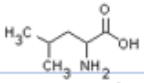
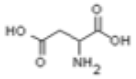
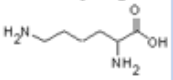
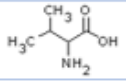
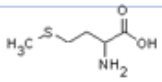
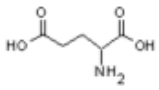
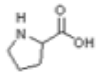
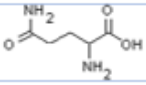
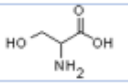
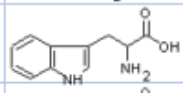
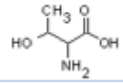
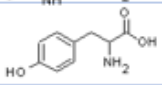
Πρωτεΐνες υποδοχείς:

Οι πρωτεΐνες αυτές βρίσκονται στην μεμβράνη του κυττάρου και ρόλος τους είναι να αναγνωρίζουν και να συνδέονται με ουσίες που είναι σημαντικές για τον μεταβολισμό των κυττάρων. Τέτοιες πρωτεΐνες είναι οι γλυκοπρωτεϊνες.

3.1.2 Δομή των Πρωτεϊνών

3.1.2.1 Αμινοξέα στις Πρωτεΐνες

Οι πρωτεΐνες είναι μακρομοριακές ενώσεις που σχηματίζονται από την συνένωση απλούστερων χημικών ουσιών, τα αμινοξέα. Η διάταξη της σειράς των αμινοξέων για κάθε πρωτεΐνη είναι γενετικά προκαθορισμένη στα χρωμοσώματα του οργανισμού(DNA). Τα αμινοξέα αποτελούν την κύρια πηγή αζώτου στον οργανισμό και παρόλο που υπάρχουν εκατοντάδες αμινοξέα στην φύση εντούτοις μόνο 20 από αυτά εμφανίζονται συνήθως στις πρωτεΐνες. Πιο συγκεκριμένα οι πρωτεΐνες στους περισσότερους οργανισμούς συνθέτονται με τον πολυμερισμό 20 διαφορετικών L-a-αμινοξέων

Αμινοξύ	Συμβολισμός		Τύπος Αμινοξέων	Αμινοξύ	Συμβολισμός		Τύπος Αμινοξέων
Αλανίνη	Ala	A		Ιστιδίνη	His	H	
Αργινίνη	Arg	R		Κυστεΐνη	Cys	C	
Ασπαραγίνη	Asn	N		Λευκίνη*	Leu	L	
Ασπαραγινικό οξύ	Asp	D		Λυσίνη*	Lys	K	
Βαλίνη*	Val	V		Μεθειονίνη*	Met	M	
Γλουταμινικό οξύ	Glu	E		Προλίνη	Pro	P	
Γλουταμίνη	Gln	Q		Σερίνη	Ser	S	
Γλυκίνη	Gly	G		Τρυπτοφάνη*	Trp	W	
Θρεονίνη*	Thr	T		Τυροσίνη	Tyr	Y	

*Πίνακας 12: Τα 20 φυσικά L-α-αμινοξέα που συμμετέχουν στο σχηματισμό της πρωτεΐνης. *Απαραίτητα αμινοξέα*

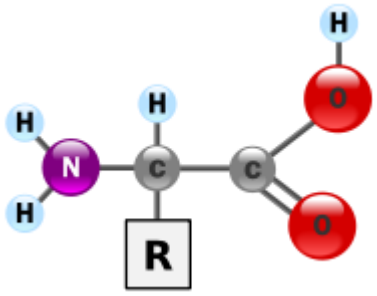
Τα κύρια αμινοξέα χωρίζονται σε δύο ομάδες στα απαραίτητα και τα μη απαραίτητα. Τα αμινοξέα που μπορούν να συντεθούν από τον οργανισμό είναι τα μη απαραίτητα. Τα απαραίτητα αμινοξέα είναι εκείνα που ο ανθρώπινος οργανισμός αδυνατεί να συνθέσει και συνεπώς είναι αναγκασμένος να τα προμηθεύεται αποκλειστικά από πρωτεϊνούχα τρόφιμα.(Σχήμα 12)

Επίσης μια τυπική πρωτεΐνη μπορεί να περιέχει 300 ή και περισσότερα αμινοξέα και κάθε πρωτεΐνη έχει τον δικό της αριθμό και την δική της αλληλουχία αμινοξέων. Για αυτόν τον λόγο οποιαδήποτε αλλαγή στην σειρά ή τον αριθμό των αμινοξέων μιας πρωτεΐνης οδηγεί στην δημιουργία μιας διαφορικής πρωτεΐνης με διαφορετικό βιολογικό ρόλο.

Ορισμός Αμινοξέων

Αμινοξέα εξ ορισμού είναι οι χημικές ενώσεις που περιέχουν τουλάχιστο μια καρβονική ομάδα C από τα καρβονικά οξέα (RCOOH) και μια τουλάχιστον αμινομάδα (-NH2).

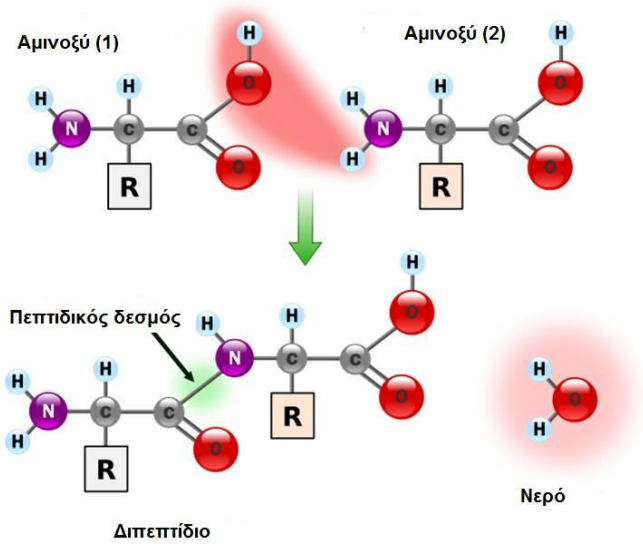
(Σχήμα 13)



Σχήμα 13 : Γενικός τύπος των αμινοξέων όπου R είναι η πλευρική αλυσίδα (ομάδα που ποικίλει ανάλογα με το αμινοξύ)
(https://en.wikipedia.org/wiki/Amino_acid)

Ένωση μεταξύ δύο Αμινοξέων

Ο δεσμός που ενώνει τα αμινοξέα μεταξύ τους ονομάζεται πεπτιδικός δεσμός και σχηματίζεται με αντίδραση συμπύκνωσης μεταξύ της καρβοξυλομάδας (-COOH) του ενός αμινοξέως και της αμινομάδας (-NH2) του επόμενου. Επίσης η διαδικασία αυτή επιτυγχάνεται με την αφαίρεση ενός μορίου νερού.



Σχήμα 14: Ένωση δύο αμινοξέων και δημιουργία διπεπτιδίου. (https://en.wikipedia.org/wiki/Amino_acid)

Η ένωση δύο αμινοξέων δημιουργεί διπετίδιο , τριών αμινοξέων τριπεπτίδιο κ.ο.κ. Πεπτίδια με αριθμό αμινοξέων μεγαλύτερο από 50 καλούνται πολυπεπτίδια. Αν το πολυπεπτίδιο αποτελείται από 100 και πλέον αμινοξέα συνδεδεμένα με πεπτιδικό δεσμό, τότε χαρακτηρίζεται ως πρωτεΐνη. Ένα πολυπεπτίδιο συνήθως αμέσως μετά την σύνθεσή του δεν είναι ικανό να εκδηλώσει το βιολογικό του ρόλο. Η ικανότητα αυτή αποκτάται όταν η πολυπεπτιδική αλυσίδα αποκτήσει την τελική διαμόρφωσή της στον χώρο. Για αυτό τον λόγο στην φύση οι πρωτεΐνες συναντιόνται μόνο στην τελική τρισδιάστατή τους μορφή. Παρόλα αυτά για την διευκόλυνση της μελέτης τους υιοθετήθηκε μια ιεραρχία για την οργάνωση και κατασκευή της πρωτεΐνης.

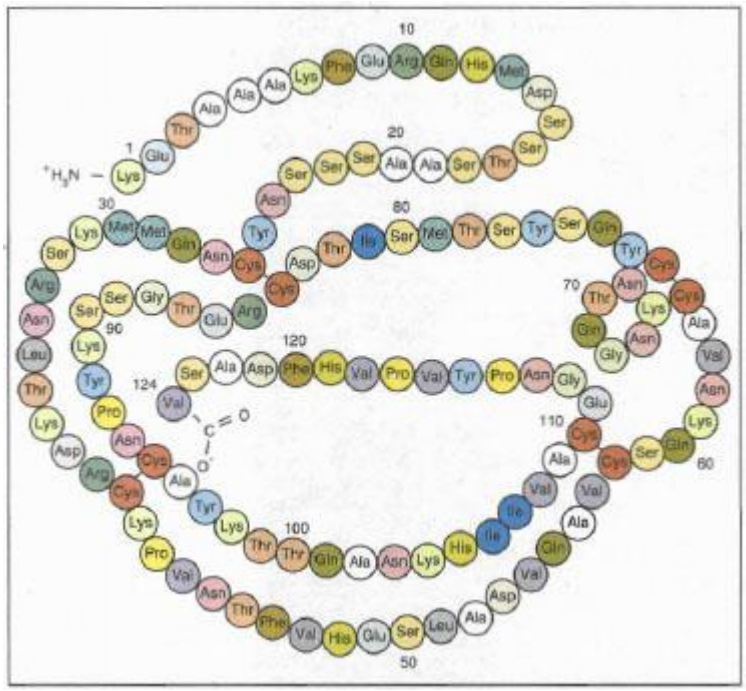
3.1.2.2 Επίπεδα οργάνωσης Δομής Πρωτεϊνών

Η μελέτη των πρωτεϊνών πραγματοποιείται σε τέσσερα επίπεδα ανάλογα με την πολυπλοκότητα που επιδεικνύει ο σχηματισμός των αμινοξέων στις πολυπεπτιδικές αλυσίδες. Για να χαρακτηρίσουμε μια πρωτεΐνη δεν αρκεί μόνο να γνωρίζουμε από ποια και από πόσα κατά περίπτωση αμινοξέα αποτελείτε αλλά πρέπει επιπλέον να προσδιοριστεί και

η σειρά με την οποία βρίσκονται συνδεδεμένα τα αμινοξέα αυτά. Αυτό συμβαίνει επειδή η σειρά, δηλαδή η αλληλουχία των αμινοξέων καθορίζει και τις ιδιότητες της πρωτεΐνης. [45]

Πρωτοταγής Δομή:

Η πρωτοταγής δομή μιας πρωτεΐνης είναι η αλληλουχία των αμινοξέων που την αποτελούν.

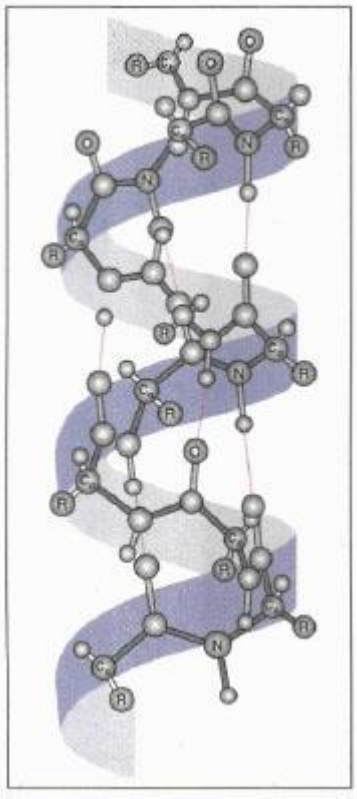


Σχήμα 15: Πρωτοταγής δομή πρωτεΐνης. Το πρωτεϊνικό μόριο μοιάζει με αλυσίδα, οι κρίκοι της οποίας είναι τα αμινοξέα. (http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/)[45]

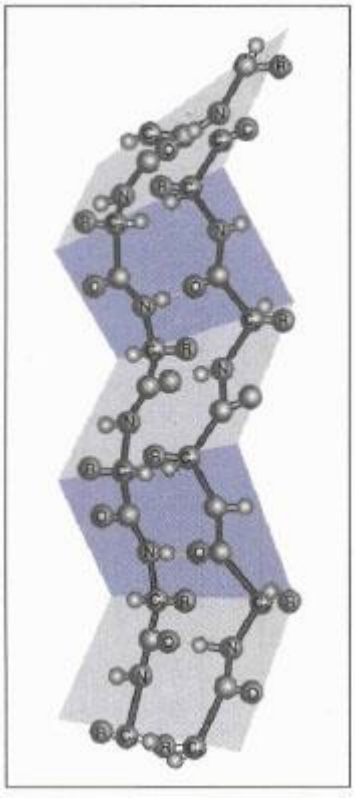
Η εύρεση εντούτοις της πρωτοταγούς δομής της πολυπεπτιδικής αλυσίδας δεν παρέχει τα σύνολο των πληροφοριών για το σχήμα της πρωτεΐνης. Αυτό συμβαίνει επειδή η αλυσίδα των αμινοξέων που αποτελούν την πρωτεΐνη δεν βρίσκονται σε ευθεία στην πραγματικότητα, αλλά πραγματοποιούν ορισμένες αναδιπλώσεις στο χώρο, προσδίδοντας στο μόριο σχήμα. Οι αναδιπλώσει αυτές δεν είναι τυχαίες αλλά εξαρτώνται από την ίδια την αλληλουχία των αμινοξέων και των δυνάμεων που προκύπτουν μεταξύ των τμημάτων της πολυπεπτιδικής αλυσίδας. Εκτός από το σχήμα που προκύπτει από τις αλληλεπιδράσεις των διαφόρων τμημάτων της πρωτεΐνης, οι δυνάμεις και η αλληλουχία των αμινοξέων καθορίζει και τον βιολογικό ρόλο της πρωτεΐνης.

Δευτεροταγής Δομή:

Οι δυνάμεις που συμμετέχουν στις αναδιπλώσεις είναι δεσμοί υδρογόνου και ηλεκτροστατικές αλληλεπιδράσεις. Η δευτεροταγής δομή αναφέρεται στις αναδιπλώσεις που μπορεί να έχουν τα διάφορα τμήματα μιας πολυπεπτιδικής αλυσίδας και μπορεί να καταταχτεί σε δύο κατηγορίες ανάλογα με την μορφή της.[45]



Σχήμα 16 :Δευτεροταγής δομή α-έλικας.



Σχήμα 17: Δευτεροταγής δομή Β-πτυχωτής επιφάνειας

(http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/)[45]

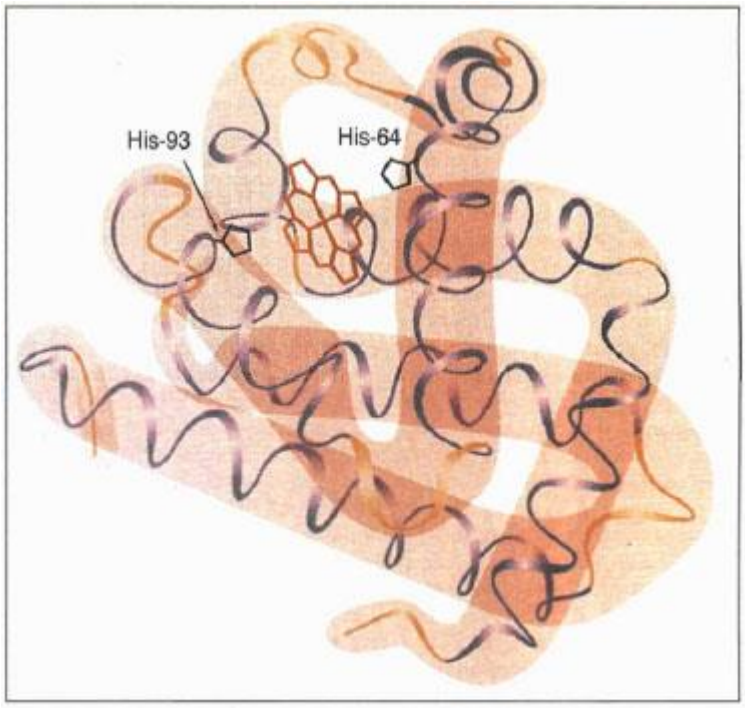
Στην α-έλικα η πρωτεΐνη αναδιπλώνεται με την βοήθεια δεσμών υδρογόνου μεταξύ αμινοξέων της ίδιας πολυπεπτιδικής αλυσίδας τα οποία είναι σε κοντινή μεταξύ τους απόσταση. Έτσι η πρωτεΐνη παίρνει ελικοειδή μορφή.(Σχήμα 16)

Στην β-πτυχωτή επιφάνεια η αναδίπλωση γίνεται με την βοήθεια δεσμών υδρογόνου κυρίως μεταξύ διαφορετικών πολυπεπτιδικών αλυσίδων του ίδιου πρωτεϊνικού μορίου. Έτσι, η πρωτεΐνη αποκτά σχήμα επιφάνειας με πτυχώσεις.(Σχήμα 17)

Επίσης μια πολυπεπτιδική αλυσίδα μπορεί να έχει σε κάποιο τμήμα της τη δομή α-έλικως και σε κάποιο άλλο την δομή β-πτυχωτής επιφάνειας. Μπορεί ακόμη σε άλλο τμήμα της η πρωτεϊνική αλυσίδα να αναδιπλώνεται με τυχαίο τρόπο.

Τριτοταγής Δομή:

Η τριτοταγής δομή προκύπτει από την αναδίπλωση τμημάτων της ήδη αναδιπλωμένης έλικας μιας πρωτεϊνικής αλυσίδας προσδίδοντας στο μόριο συνολικά ένα συγκεκριμένο σχήμα [45]. (Σχήμα 18)

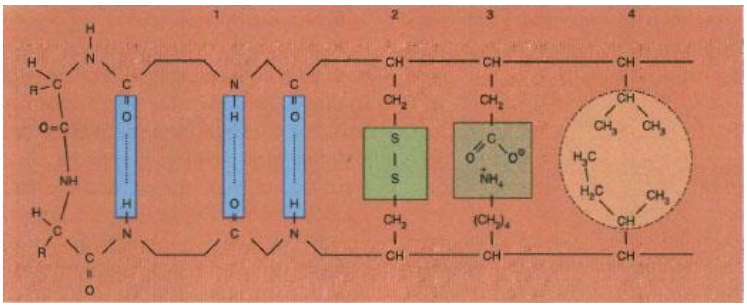


Σχήμα 18: Τριτοταγής δομή η οποία περιέχει τμήματα α-έλικας τα οποία εναλλάσσονται με τμήματα πεπτιδικής αλυσίδας που παρουσιάζουν τυχαία αναδίπλωση.
(http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/)[45]

Στην τριτοταγή δομή συμβάλλουν διάφοροι δεσμοί όπως:

- δεσμοί υδρογόνου
- ηλεκτροστατική έλξη μεταξύ αντίθετα φορτισμένων ομάδων

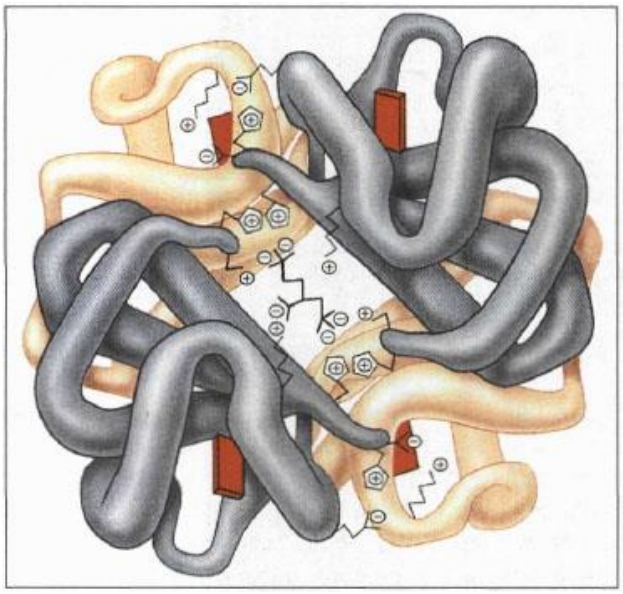
- υδρόφοβοι δεσμοί που δημιουργούνται μεταξύ υδρόφοβων ομάδων, όταν η μια πλησιάζει την άλλη,
- δυνάμεις Van der Waals μεταξύ μη πολικών ομάδων και
- ομοιοπολικοί δισουλφιδικοί δεσμοί που είναι δεσμοί μεταξύ ατόμων S δύο κυστεϊνών



Σχήμα 19: Δεσμοί μεταξύ διαφόρων τμημάτων μίας πολυπεπτιδικής αλυσίδας. Ι. Δεσμοί υδρογόνου. 2 δισουλφιδικοί δεσμοί. 3. Ιοντικοί δεσμοί μεταξύ πλευρικών ομάδων Asp και Lys. 4. Υδρόφοβοι δεσμοί μεταξύ Val και lie. (http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/)[45]

Τεταρτοταγής δομή.

Καθορίζεται από όμοιες ή διαφορετικές πολυπεπτιδικές αλυσίδες που ενώ έχουν αναδιπλωθεί μπορούν συχνά να συνενωθούν μεταξύ τους σχηματίζοντας μεγαλύτερα πρωτεϊνικά σύμπλοκα. Το τελικό σχήμα που αποκτά το πρωτεϊνικό σύμπλοκο στο χώρο αποτελεί την τεταρτοταγή δομή της πρωτεΐνης , ενώ οι ανεξάρτητες πεπτιδικές αλυσίδες που συνθέτουν το πρωτεϊνικό σύμπλοκο αποτελούν τις υπομονάδες [45].



Σχήμα 20: Τεταρτοταγής δομή αιμοσφαιρίνης, όπως αυτή προκύπτει από την αλληλεπίδραση των υπομονάδων της. (http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/)[45]

Τελικό σχήμα πρωτεϊνών

Το τελικό σχήμα που αποκτούν οι πρωτεΐνες μπορεί να είναι είτε σφαιρικό είτε ινώδες. Οι σφαιρικές πρωτεΐνες είναι διαλυτές στο νερό και σε αραιά διαλύματα αλάτων. Παραδείγματα αυτών των πρωτεϊνών είναι οι πρωτεΐνες που βρίσκονται στο αίμα. Οι ινώδεις πρωτεΐνες είναι αδιάλυτες στο νερό και χρησιμεύουν ως στηρικτικές και σκελετικές ουσίες.

3.1.3 Η μελέτη των πρωτεϊνών

3.1.3.1 Αναγκαιότητα της γνώσης των Πρωτεϊνών

Από τα παραπάνω είναι εμφανές ότι οι πρωτεΐνες είναι σημαντικές για κάθε ζωντανό οργανισμό και συνεπώς σημαντική είναι και η ίδια τους η μελέτη. Η εύρεση της τριτοταγούς δομής των πρωτεϊνών συνεπάγεται την εύρεση περισσότερων ρόλων και δράσεων των πρωτεϊνών και την δημιουργία πρωτείων με επιθυμητές ιδιότητες και συγκεκριμένο βιολογικό ρόλο. Δηλαδή θα οδηγήσει στην δημιουργία φαρμακιών και αντιβιοτικών για την ίαση διαφόρων ασθενειών. Συνεπώς η μελέτη των πρωτεϊνών απασχολούσε και απασχολεί τις επιστήμες της βιολογίας, της ιατρικής και της φαρμακευτικής

3.1.3.2 Τα προβλήματα που αντιμετωπίζει αυτή η μελέτη

Η απομόνωση μιας πρωτεΐνης από ένα βιολογικό δείγμα και η απόκτηση της σε καθαρή μορφή ώστε να είναι δυνατή η μελέτη της είναι ένα πρόβλημα το οποίο το αντιμετωπίζουν συχνά οι βιοχημικοί. Οι μέθοδοι που χρησιμοποιούνται για τον διαχωρισμό των πρωτεϊνών στηρίζονται κυρίως σε διαφορές φυσικοχημικών ιδιοτήτων, όπως στο μέγεθος του μορίου και στο ηλεκτρικό φορτίο. Η απομόνωση όμως της πρωτεΐνης από μόνη της ως διαδικασία αντιμετωπίζει δύο δυσκολίες που καθιστούν την πραγματοποίησή της ανέφικτη. Πρώτο οι πρωτεΐνες είναι αρκετά ευαίσθητες οπότε κατά την απομόνωσή τους υπάρχει κίνδυνος να υποστούν αλλοίωση ή διάσπαση και δεύτερο η πρωτεΐνη που επιθυμούμε να απομονώσουμε θα πρέπει πρώτα να διαχωριστεί από ένα πολύπλοκο μείγμα πρωτεϊνών αλλά και άλλων βιολογικών μορίων. Για αυτό τον λόγο δημιουργήθηκε και η ιεραρχική οργάνωση της δομής των πρωτεϊνών.

Ο προσδιορισμός της πρωτοταγούς δομής είναι μία εργασία που παλαιότερα πραγματοποιούνταν πολύ δύσκολα. Τα τελευταία χρόνια όμως γίνεται σχετικά εύκολα με τη βοήθεια μηχανημάτων, τα οποία μπορούν να προσδιορίσουν αυτόματα την αλληλουχία των αμινοξέων. Ο προσδιορισμός αυτός στηρίζεται στη μέθοδο της αποικοδόμησης κατά Edman. Όπως αναφέρθηκε όμως και παραπάνω για να μπορούμε να κατανοήσουμε το βιολογικό ρόλο κάποιας πρωτεΐνης και το ακριβές της σχήμα στο χώρο θα πρέπει να γνωρίζουμε τουλάχιστο την δευτεροταγή δομή της πρωτεΐνης.

Η εξέλιξη των μεθόδων κρυσταλλογραφίας με ακτίνες Χ οδήγησε στην ακριβέστερη μελέτη της δευτεροταγούς δομής και γενικότερα της διάταξης μίας πρωτεΐνης στο χώρο. Έτσι βρέθηκε ότι οι αλυσίδες των πρωτεϊνών μπορεί να έχουν δύο διαφορετικές μορφές και συγκεκριμένα: α) την μορφή α-έλικας, β) την μορφή β-πτυχωτής επιφάνειας. Η μέθοδος αυτή εντούτοις είναι αρκετά δαπανηρή, πολύπλοκη και χρονοβόρα καθιστώντας το πρόβλημα της εύρεσης της δευτεροταγούς και τριτοταγούς δομής των πρωτεϊνών ουσιαστικά άλυτο.

Εφόσον η εύρεση της πρωτοταγούς δομής των πρωτεϊνών πλέον είναι εύκολο να πραγματοποιηθεί το ερώτημα που δημιουργείται είναι αν θα μπορούσαμε με βάση αυτήν να προβλέψουμε την δευτεροταγή δομή εφόσον γνωρίζουμε δηλαδή την αλληλουχία των αμινοξέων που την αποτελούν.

Αυτό το ερώτημα αποτελεί τον στόχο αυτής της διπλωματικής εργασίας.

3.2. Σχετική Έρευνα

Όταν λήφθηκε για πρώτη φορά το τρισδιάστατο σχήμα μιας σφαιρικής πρωτεΐνης με την βοήθεια των ακτίνων Χ κρυσταλλογραφίας, ήταν απογοητευτική η έλλειψη οποιασδήποτε σχέσης μεταξύ της ακολουθίας των αμινοξέων και της τελικής διαμόρφωσης της πρωτεΐνης. Παρόλα αυτά είναι ξεκάθαρο ότι κάποια σχέση πρέπει να υπάρχει εφόσον η ικανότητα μιας σφαιρικής πρωτεΐνης να αναδιπλωθεί χωρίς την βοήθεια οποιουδήποτε άλλου βιολογικού υλικού δηλώνει πως βιολογικά, η γνώση για να διαμορφωθεί στον τελικό σχεδιασμό της συμπεριλαμβάνεται στην ακολουθία των αμινοξέων [7]

Η πιο πάνω ιδέα ήταν ο πρόδρομος για την δημιουργία πολλών αλγόριθμων, δικτύων και μεθόδων όπου η λύση για το πρόβλημα της εύρεσης του σχήματος της πρωτεΐνης, μεταφράζεται ουσιαστικά ως το πρόβλημα της πιο ακριβούς πρόβλεψης της κάθε δομής που έπεται της πρωτοταγής για κάθε πρωτεΐνης. Δηλαδή λαμβάνοντας υπόψη την πρωτοταγή δομή να μπορούμε να προβλέψουμε την δευτεροταγή δομή και ανάλογα με βάση την δευτεροταγή δομή να προβλέψουμε την τριτοταγή .

Τα τελευταία 45 χρόνια δημιουργήθηκαν πολλές μεθόδοι για την επίλυση του προβλήματος της πρόβλεψης της δευτεροταγής δομής των πρωτεϊνών. Αρχικά χρησιμοποιήθηκαν μεθόδοι πρόβλεψης της δευτεροταγής δομή (1960-1970) [8] [9] [10] [11] [12]οι οποίες εστίαζαν στην αναγνώριση της α-έλικως και βασίζονταν κυρίως στο μοντέλο helix-coil transition.[13] Αργότερα αναπτύχθηκαν πιο ακριβής προβλέψεις που συμπεριλάμβαναν και τις β-πτυχωτές επιφάνειες, κατά την δεκαετία του 1970.

Ένας από τους πρώτους αλγόριθμους που χρησιμοποιήθηκαν για την επίλυση του προβλήματος ήταν η μέθοδος Chou-Fasman κατά την οποία η ακρίβεια του αποτελέσματος καθοριζόταν από την συχνότητα εμφάνισης του κάθε αμινοξέος σε κάθε τύπο της δευτεροταγούς δομής.[14] Αυτή η μέθοδος δημιουργήθηκε στα μέσα του 1970 και σημείωσε 50-60% ακρίβεια.

Στα τέλη του 1970 επινοήθηκε το πρόγραμμα GOR το οποίο χρησιμοποιεί μια πιο επιτυχημένη τεχνική από την μέθοδο Chou-Fasman[14]. Με βάση την Bagesian Inference

μέθοδο το πρόγραμμα GOR λαμβάνει υπόψη του πέρα από την συχνότητα εμφάνισης του κάθε αμινοξέος στον κάθε τύπο δευτεροταγούς δομής, την πιθανότητα του κάθε αμινοξέος να ανήκει σε κάποιο τύπο δευτεροταγούς δομής με βάση τα άμεσα γειτονικά του αμινοξέα.[15] Δηλαδή συμπεριλαμβάνει και την πιθανότητα ένα αμινοξύ να ανήκει στον ίδιο τύπο δευτεροταγής δομής όπως και τα άμεσα γειτονικά του αμινοξέα. Η μέθοδος αυτή σημείωσε 65% ακρίβεια.

Σημαντική συνεισφορά για την λύση του προβλήματος της πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών προσέφεραν τα Τεχνητά Νευρωνικά Δίκτυα(ΤΝΔ) τα οποία είναι συστήματα μάθησης που βρίσκουν λύσεις σε προβλήματα με πολύπλοκα, μη γραμμικά δεδομένα. Η γενικότερη ιδέα των ΤΝΔ στο PSSP προβλήματος είναι ότι τα ΤΝΔ εκπαιδεύονται ούτως ώστε να αναγνωρίζουν την δευτεροταγή δομή ακολουθιών αμινοξέων κάποιων γνωστών πρωτεϊνών, και αργότερα να χρησιμοποιούν αυτή την γνώση για να μπορούν να διακρίνουν την δευτεροταγή δομή άγνωστων πρωτεϊνών.

Η τοπολογία των ΤΝΔ είναι οργανωμένη σε επίπεδα και κάθε επίπεδο αποτελείται από μονάδες που αλλιώς ονομάζονται νευρώνες.(Εκτενής περιγραφή τοπολογίας ΤΝΔ στο Κεφάλαιο 4)Τα επίπεδα διακρίνονται σε επίπεδο εισόδου, κρυφά επίπεδα και επίπεδο εξόδου. Οι μονάδες του κάθε επιπέδου συνδέονται με τις μονάδες κάποιου άλλου επιπέδου ανάλογα με την τοπολογία του δικτύου. Κάθε μονάδα σε ένα επίπεδο λαμβάνει πληροφορία από μία ή και περισσότερες μονάδες του ίδιου ή και διαφορετικού επιπέδου και καθορίζει την έξοδό της με βάση κάποια βάρη που δέχεται από κάθε είσοδος που λαμβάνει. Για να μπορούν τα ΤΝΔ να αναγνωρίσουν ένα μοτίβο θα πρέπει πρώτα να περάσουν από μια διαδικασία της εκπαίδευση.(Υποκεφάλαιο 4.2 Εκπαίδευση Τεχνητών Νευρωνικών Δικτύων) Ουσιαστικά τα Τεχνητά Νευρωνικά Δίκτυα λειτουργούν ως ένα μαύρο κουτί εφόσον είναι δύσκολο να κατανοηθεί η εσωτερική λειτουργικότητα του δικτύου.

Το ποσοστό ακρίβειας του κάθε δικτύου για το πρόβλημα πρόβλεψης της δευτεροταγής δομής των πρωτεϊνών ονομάζεται Q3 και καθορίζεται από την επιτυχία αναγνώρισης των αμινοξέων σε μια άγνωστη προς το δίκτυο πρωτεϊνική ακολουθία. [18] Υπολογίζεται από την πιο κάτω εξίσωση: (Εξίσωση 1)

$$Q_3 = 100 \frac{1}{N_{res}} \sum_{i=0}^{3} M_{ii}$$

Εξίσωση1: Καθορίζει το ποσοστό επιτυχίας αναγνώρισης αμινοξέων, όπου N_{res} η ποσότητα των αμινοξέων της πρωτεϊνικής ακολουθίας εισόδου και M_{ij} καθορίζεται διαφορικά παίρνοντας την τιμή 1 αν η έξοδος του i ταυτίζεται με την έξοδο j και 0 σε αντίθετη περίπτωση.

3.2.1 Παραδείγματα προσεγγίσεων για το PSSP πρόβλημα

Μερικά από τα δίκτυα που η συνεισφορά τους προς την λύση του προβλήματος της πρόβλεψης της δευτεροταγής δομής των πρωτεϊνών είναι τα ακόλουθα:

Πλήρως Συνδεδεμένο Δίκτυο Εμπρόσθιου Περάσματος (Fully connected Feed Forward Neural)

Το 1988 οι Qian και Sejnowski [34] χρησιμοποίησαν το πλήρως συνδεδεμένο δίκτυο εμπρόσθιου περάσματος(Fully connected Feed Forward Neural Network) για να λύσουν το πρόβλημα της πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών. Η τοπολογία του δικτύου ήταν περιείχε ένα κρυφό επίπεδο και τα οι πρωτεϊνικές ακολουθίες εισέρχονταν στο δίκτυο με ένα παράθυρο 13 αμινοξέων. Για την κανονικοπίηση των δεδομένων εισόδου χρησιμοποίησαν ορθογώνια κωδικοποίηση. Αυτή η προσέγγιση πέτυχε ποσοστό ακρίβειας Q3 63.3% και παρατηρήθηκε το πρόβλημα της υπερεκπαίδευσης στο δίκτυο (over fitting problem).

Μέθοδος PHD

Προσπαθώντας να αντιμετωπίσουν το πρόβλημα της υπερεκπαίδευσης και τα βελτιώσουν την πιο πάνω προσέγγιση οι Rost και Sander[35][36] δημιούργησαν την μέθοδο PHD. Αυτή η μέθοδος χρησιμοποίησε τεχνικές όπως το γρήγορο σταμάτημα (early stopping) και την τεχνική ensemble average με την οποία εκπαίδευαν ένα σύνολο από Νευρωνικά Δίκτυα βρίσκοντας το Μέσο όρο τους. Πιο συγκεκριμένα εκπαίδευσαν τρία διαδοχικά νευρωνικά δίκτυα όπου στα πρώτα δύο γίνεται η πρόβλεψη της δευτεροταγούς δομής των πρωτεϊνών και το αποτέλεσμα αυτών τροφοδοτείται στο τρίτο δίκτυο που αλλιώς ονομάζεται και δίκτυο δικαστής. Η πρόβλεψη που γίνεται από το τρίτο δίκτυο περνά από την διαδικασία filtering

με βάση την οποία αποτρέπονται τυχών λάθη που προέκυψαν κατά την πρόβλεψη[46]. Ακόμα μια σημαντική αλλαγή ως προς το Fully connected Feed Forward Neural είναι ότι στο PHD για την κωδικοποίηση των δεδομένων εισόδου χρησιμοποιήθηκαν πολλαπλές ευθυγραμμισμένες ακολουθίες οι λεγόμενες MSA (Κεφάλαιο 123). Με αυτή την αλλαγή τα δεδομένα εμπλουτίστηκαν εφόσον περιείχαν περισσότερη πληροφορία για το κάθε αμινοξύ και συνεπώς για την αλληλεπίδραση των αμινοξέων. Αυτή η μέθοδος πέτυχε ποσοστό ακρίβειας Q3 71.4%.

NNSSP (Nearest –Neighbor method)

Μια παρόμοια προσέγγιση με το PHD παρουσιάστηκε το 1997 από τους Salamov και Soloveyen. Το NNSSP δίκτυο χρησιμοποιεί την μέθοδο του κοντινότερου γείτονα (k nearest neighbor algorithm) και κωδικοποιεί τα δεδομένα εισόδου με βάση τα MSA. Αυτή η προσέγγιση σημείωσε ποσοστό επιτυχίας Q3 68.41%.

BRNN (Bidirectional Recurrent Neural Network)

Μια από τις πιο σημαντικές προσεγγίσεις ως προς την λύση του προβλήματος της πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών είναι η δημιουργία και εφαρμογή του Bidirectional Recurrent Neural Network (BRRN) από τον Baldi [19][20]. Έμπνευση για την δημιουργία αυτού του δικτύου υπήρξε η αδυναμία των νευρωνικών δικτύων εμπρόσθιου περάσματος (feed forward Neural Networks) με σταθερού μεγέθους παράθυρο να αναπαραστήσει μεγάλου εύρους εξαρτήσεις όπως οι πρωτεϊνικές ακολουθίες. Το BRNN δέχεται ως είσοδο ένα καθορισμένου μεγέθους παράθυρο το οποίο έχει ως κέντρο του κάθε φορά ένα αμινοξύ. Με αυτό τον τρόπο προσπαθεί κάθε φορά να προβλέψει την δευτεροταγή δομή για το αμινοξύ που βρίσκεται στο κέντρο του παραθύρου λαμβάνοντας υπόψη τα αμινοξέα που βρίσκονται αριστερά αλλά και δεξιά του. Σύμφωνα με την τριτοταγή δομή των πρωτεϊνών η πρωτεΐνη αναδιπλώνεται στο χώρο και αυτό το νευρωνικό δίκτυο μπορεί να προβλέψει σε μεγάλο βαθμό τις δυνάμεις που θα λαμβάνει το κάθε αμινοξύ σε μια πρωτεϊνική ακολουθία, από τα γειτονικά του αμινοξέα. Για αυτό τον λόγο το BRNN σημείωσε επιτυχία Q3 76%.

Κεφάλαιο 4: Νευρωνικά δίκτυα:

4.1.Τί είναι τα νευρωνικά δίκτυα

Τα Τεχνητά Νευρωνικά Δίκτυα(ΤΝΔ) αποτελούν ένα σύγχρονο και γοργά αναπτυσσόμενο κλάδο της τεχνητής Νοημοσύνης. Τα ΤΝΔ είναι απλοποιημένα μοντέλα του κεντρικού νευρικού συστήματος του ανθρώπου. Μιμούνται τη λειτουργία των βιολογικών νευρώνων του εγκεφάλου και την αρχιτεκτονική-δομή των βιολογικών Νευρωνικών Δικτύων. Αποτελούνται από διασυνδεδεμένα υπολογιστικά στοιχεία που έχουν την ικανότητα να ανταποκρίνονται σε ερεθίσματα που δέχονται ως είσοδό τους και να μαθαίνουν να προσαρμόζονται στο περιβάλλον τους.

Ουσιαστικά τα Τεχνητά Νευρωνικά Δίκτυα προσπαθούν να μιμηθούν την λειτουργίες του ανθρώπινου εγκεφάλου όπως επεξεργασία πολύπλοκων δεδομένων σε μικρό χρονικό διάστημα, ή την λύση και προσαρμογή του σε μια πλειάδα προβλημάτων. Αυτό που διαφοροποιεί τα Τεχνητά νευρωνικά δίκτυα από οποιοδήποτε άλλο υπολογιστικό πρόγραμμα είναι η ικανότητά τους να μπορούν να προβλέψουν το αποτέλεσμα μιας κατάστασης με βάση κάποια είσοδο και όχι με τυχαίο τρόπο. Ακόμα μια ικανότητά που έχουν τα ΤΝΔ είναι ότι σε αντίθεση με άλλα υπολογιστικά συστήματα ή προγράμματα αυτά δεν είναι επιρρεπής σε λάθη ή θορυβώδη δεδομένα και αυτό γιατί τα ΤΝΔ περνούν από μια συγκεκριμένη διαδικασία που ονομάζεται εκπαίδευση. Επίσης για την ικανοποιητική τους λειτουργία δεν έχουν μεγάλες υπολογιστικές απαιτήσεις και μπορούν οι διάφορες μονάδες επεξεργασίας του ή επίπεδα να λειτουργούν παράλληλα ή και αναδρομικά.

Τα ΤΝΔ είναι ιδανικά συστήματα για να δίνουν λύση σε προβλήματα τα οποία δεν είναι καλά ορισμένα ή η λύση τους δεν διέπεται από συγκεκριμένους κανόνες ή οι κανόνες αυτοί αλλάζουν δυναμικά και κάπως απρόβλεπτα ή ακόμα και προβλήματα που έχουν μεγάλο όγκο δεδομένων και χρειάζονται μεγάλη υπολογιστική ισχύ και ταχύτητα.

4.2. Εκπαίδευση Τεχνητών Νευρωνικών Δικτύων

Όπως αναφέρθηκε και πιο πάνω για να μπορεί ένα νευρωνικό δίκτυο να λύσει οποιοδήποτε πρόβλημα θα πρέπει πρώτα να περάσει από μια διαδικασία που λέγεται εκπαίδευση, αντιγράφοντας δηλαδή την ανθρώπινη διαδικασία μάθησης η οποία περνά από κάποια εκπαίδευση. Η εκπαίδευση ενός ΤΝΔ καθορίζεται από ένα σύνολο δεδομένων που δέχεται ως είσοδο ένα Νευρωνικό δίκτυο, ένα σύνολο αλγορίθμων-τεχνικών και κάποιες μεταβλητές που διατηρούν, αποθηκεύουν την μάθηση. Οι αλγόριθμοι μάθησης των ΤΝΔ χωρίζονται στις ακόλουθες τρεις κατηγορίες.

4.2.1 Επιβλεπόμενη Μάθηση

Η επιβλεπόμενη μάθηση μιμείται την μάθηση υπό διδασκαλία δηλαδή το δίκτυο μαθαίνει με βάση την βοήθεια κάποιου δασκάλου. Σε αυτήν την περίπτωση εκτός από τα σύνολα εισόδου, αλγορίθμων και μεταβλητών για την εκπαίδευση επιβάλλεται και η χρήση ενός συνόλου δεδομένων το οποίο περιέχει το σωστό-επιθυμητό αποτέλεσμα για κάθε μια από τις εισόδους και είναι αυτό που παίρνει τον ρόλο του δασκάλου.

4.2.2 Μη επιβλεπόμενη Μάθηση

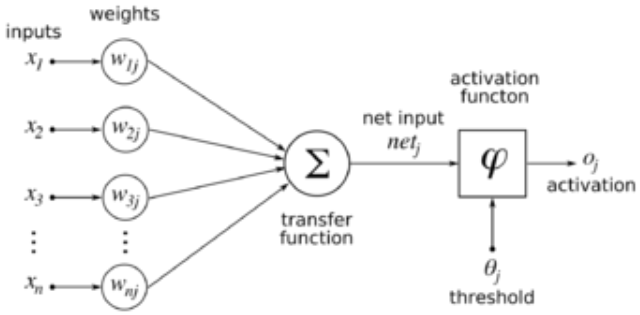
Η μη επιβλεπόμενη μάθηση μιμείται την ικανότητα του ανθρώπινου εγκέφαλου να κατηγοριοποιεί δεδομένα όταν τα λαμβάνει ως ερεθίσματα. Έτσι και τα ΤΝΔ στην μη επιβλεπόμενη μάθηση ενώ δέχονται τα δεδομένα εισόδου προσπαθούν να τα κατηγοριοποιήσουν ανάλογα με κάποια χαρακτηριστικά που ξεχωρίζουν από αυτά

4.2.3 Ενισχυτική Μάθηση

Η ενισχυτική μάθηση μιμείται την ικανότητα του ανθρώπινου εγκεφάλου ανάλογα με το επακόλουθο που θα έχει κάποια πράξη του. Τα ΤΝΔ στην ενισχυτική Μάθηση ανάλογα με το αποτέλεσμα επακόλουθο της επεξεργασίας κάποιας εισόδου επιβραβεύει ή τιμωρεί αυτή την εξέλιξη και με αυτό τον τρόπο το δίκτυο μαθαίνει να επαναλαμβάνει ή όχι κάποια ενέργεια.

4.3. McCulloch και Pitts μοντέλο

Το 1943 επινοήθηκε ένα αρκετά σημαντικό και απλοϊκό μοντέλο των νευρώνων του εγκεφάλου το οποίο ονομάστηκε McCulloch και Pitts από τους δημιουργούς του.[16] Η δημιουργία αυτού του μοντέλου προέκυψε από τον προβληματισμό των Warren McCulloch και Walter Pitts για το πώς ο ανθρώπινος εγκέφαλος έχει την ικανότητα να εκτελεί πολύπλοκα μοτίβα και λειτουργείες χρησιμοποιώντας πολλά απλά εγκεφαλικά κύτταρα συνδεδεμένα μεταξύ τους. Το βασικό μοντέλο McCulloch και Pitts είναι ένας τεχνητός νευρώνας που δέχεται ένα σύνολο από εισόδους x1,x2,x3…,xn και έχει μια έξοδο oj. Τα βάρη του νευρώνα w1j,w2j,w3j…,wnj κανονικοποιούνται και για αυτό οι τιμές που παίρνουν κυμαίνονται μεταξύ των αριθμητικών συνόλων (0,1) ή (-1,1). Επίσης τα βάρη συσχετίζονται με κάθε γραμμή εισόδου όπως φαίνεται και στο σχήμα (Σχήμα 21). Στο μοντέλο McCulloch και Pitts τα βάρη χρησιμοποιούνται για να ενδυναμώσουν και να αποδυναμώσουν τις τιμές εισόδου και σε συσχέτιση με ένα βιολογικό νευρώνα, νευρώνα εγκεφάλου, τα βάρη προσομοιώνουν τις συνάψεις που βρίσκονται στους δεντρίτες του νευρώνα και τον συνδέουν με κάποιους άλλους. Το θj είναι η τιμή κατωφλιού. Η δημιουργία και απομνημόνευση γνώσης στο μοντέλο McCulloch και Pitts προκύπτει κατά την διάρκεια της εκπαίδευσης και όταν συνενώσουμε πολλούς νευρώνες αυτού του μοντέλου. Οι μεταβλητές τιμές για την παραπάνω διαδικασία είναι τα βάρη και η τιμή κατωφλίου.[Ref10]



Σχήμα 21:McCalloch και Pitts νευρώνας

http://www.tau.ac.il/~tsirel/dump/Static/knowino.org/wiki/Artificial_neural_network.html

Το net input είναι το άθροισμα του γινομένου των εισόδων επί των βαρών που τα συσχετίζουν και υπολογίζεται από την ακόλουθη εξίσωση η οποία ονομάζεται εξίσωση εισόδου.

$$net = \sum_i (w_{y,x_i} x_i) + w_{y,x_b} \cdot 1 = \sum_i (w_{y,x_i} x_i) + w_{y,x_b}$$

Εξίσωση 2: Εξίσωση εισόδου όπου $w_{y,xi}$ τα βάρη που συνδέουν κάθε είσοδο x_i και $w_{y,xb}$ ττο βάρος που ενώνει την τιμή κατωφλίου με την έξοδο του νευρώνα. Η τιμή κατωφλίου είναι 1.

Στο μοντέλο McCulloch και Pitts[16] η εξίσωση εισόδου περνά από την βηματική συνάρτηση για να δοθεί μια τιμή στην έξοδο του νευρώνα. Η βηματική συνάρτηση για αυτό το μοντέλο είναι η συνάρτηση κατωφλίου κατά την οποία αν στην συνάρτηση εισόδου δοθεί για αποτέλεσα τιμή μεγαλύτερη από την τιμή κατωφλίου, τότε η έξοδος του νευρώνα παίρνει την τιμή ένα και σε αντίθετη περίπτωση την τιμή 0.

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

Εξίσωση 3: Συνάρτηση κατωφλίου όπου εξίσωση εισόδου και τιμή κατωφλίου ίση με 0.

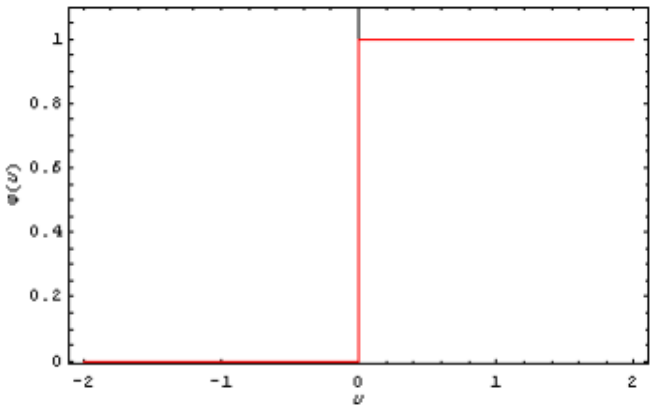
Σε μεταγενέστερο στάδιο για αυτό το μοντέλο χρησιμοποιήθηκε ως βηματική συνάρτηση η σιγμοειδής συνάρτηση.(Υποκεφάλαιο 4.4.Perceptron)

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

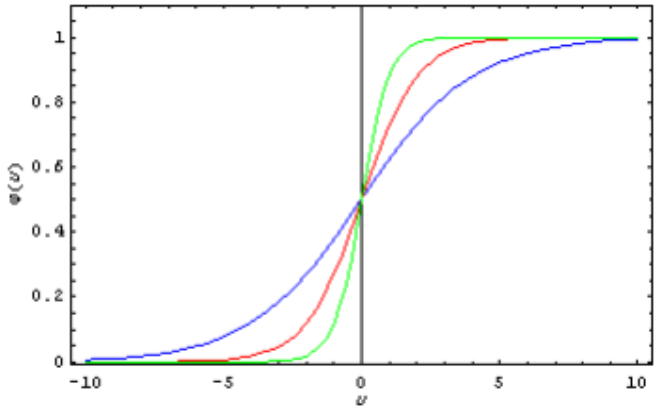
Εξίσωση4: Εξίσωση Σιγμοειδούς συνάρτησης, όπου α είναι η κλήση της συνάρτησης

Η σιγμοειδής συνάρτηση χρησιμοποιήθηκε για να αποφευχθούν δύο μειονεκτήματα που προέκυψαν από την συνάρτηση κατωφλίου.

1. Στην συνάρτηση κατωφλίου η μέθοδος κατάβασης κλήσης δεν θα μπορούσε να χρησιμοποιηθεί και
2. Με την συνάρτηση κατωφλίου δεν μπορεί να προσδιοριστεί το λάθος όταν η έξοδος του νευρώνα δεν παίρνει σωστή τιμή.



Σχήμα 22:Βηματική συνάρτηση για τιμή κατωφλίου ίση προς μηδέν (http://users.auth.gr/~voyatzis/SeniorThesis/mTsouxnika.pdf)



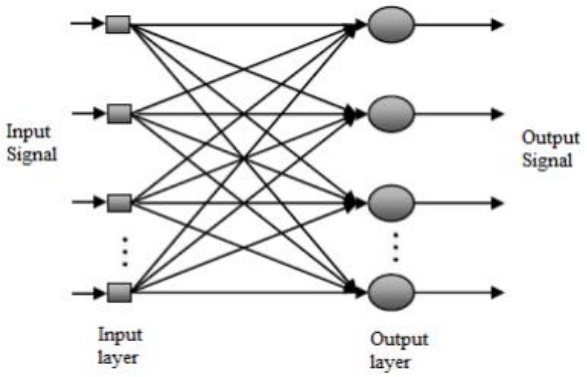
Σχήμα 23: Η λογιστική συνάρτηση, για α = 0.5 (μπλε), α = 1 (κόκκινο) και α = 2 (πράσινο) (http://users.auth.gr/~voyatzis/SeniorThesis/mTsouxnika.pdf)

4.3.1. Σημαντικότητα της έρευνας McCulloch και Pitts

Η έρευνα των Warren McCulloch και Walter Pitts[16] πυροδότησε την απαρχή για την ανάπτυξη πολλών και εντυπωσιακών τεχνητών νευρωνικών δικτύων που χρησιμοποιούνται μέχρι σήμερα. Ακόμα ένας λόγος που καθιστά την έρευνα αυτή σημαντική είναι η σταδιακή αποστασιοποίηση από την κλασσική ιδέα των δικτυακών υπολογιστών και την στροφή προς την ιδέα υπολογιστών που για την επεξεργασία δεδομένων χρησιμοποιούν τεχνητά νευρωνικά δίκτυα και λέγονται μη-κλασσικοί υπολογιστές.[17]

4.4.Perceptron

Το 1957 εφευρέθηκε ο νευρώνας Perceptron ή αλλιώς μονοεπίπεδο δίκτυο Perceptron από τον Frank Rosenblatt [31]που ήταν ουσιαστικά το πρώτο τεχνητό νευρωνικό δίκτυο το οποίο αποτελείται από δύο επίπεδα. Το πρώτο επίπεδο απαρτίζεται από τις εισόδους του δικτύου ενώ το δεύτερο επίπεδο το οποίο ονομάζεται επίπεδο εξόδου, αποτελείται από νευρώνες τύπου McCulloch και Pitts. Λύνει προβλήματα ταξινόμησης. Προσπαθεί δηλαδή να αντιστοιχήσει κάθε σύνολο εισόδων με την σωστή κλάση.

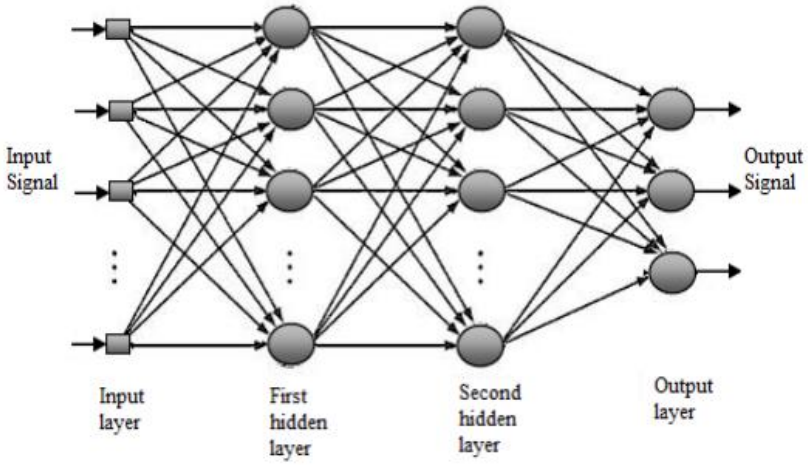


Σχήμα 24: Νευρώνας Perceptron - Μονοεπίπεδο δίκτυο Perceptron

Ο νευρώνας Perceptron φάνηκε αρκετά υποσχόμενος για την λύση προβλημάτων που χρησιμοποιούσαν επιβλεπόμενη μάθηση. Παρόλα αυτά το 1969 οι Marvin Minsky και Seymour Papert στο βιβλίο τους με τίτλο Perceptron[32] απέδειξαν ότι προβλήματα όπως το XOR δεν μπορούν να επιλυθούν με αυτόν τον νευρώνα και συνεπώς δεν θα μπορούν να επιλυθούν μη γραμμικώς διαχωρίσιμα προβλήματα.

4.4.1. Πολυστρωματικά δίκτυα Perceptron

Για να λυθεί το πιο πάνω πρόβλημα δημιουργήθηκαν τα πολυστρωματικά δίκτυα Perceptron. Τα πολυστρωματικά δίκτυα Perceptron μοιάζουν αρκετά Perceptron με την διαφορά ότι μεταξύ των επιπέδων εισόδου και εξόδου μεσολαβούν ένα ή και περισσότερα επίπεδα τα οποία ονομάζονται κρυφά (Hidden Layer).



Σχήμα 25: Πολυστρωματικό Δίκτυο Perceptron

Το στρώμα-επίπεδο εισόδου δεν είναι ενεργό επίπεδο. Σκοπός του είναι να εισάγει τα δεδομένα στο δίκτυο. Τα κρυφά επίπεδα και το επίπεδο εξόδου είναι ενεργά. Σκοπός των κρυφών επιπέδων είναι να αποθηκεύουν γνώση και πιο συγκεκριμένα αποθηκεύουν πρότυπα των εισόδων που λαμβάνουν. Δέχονται δηλαδή τα δεδομένα εισόδου και γενικεύουν αυτήν την γνώση για να τα αντιστοιχήσουν με κάποια κλάση του προβλήματος-πρότυπο.

Με βάση το πρόβλημα καθορίζεται ο αριθμός των κρυφών επιπέδων που θα χρησιμοποιηθούν καθώς και ο αριθμός των νευρώνων για κάθε επίπεδο. Επίσης με βάση το πρόβλημα καθορίζεται ο αριθμός νευρώνων στο επίπεδο εισόδου και ο αριθμός των νευρώνων στο επίπεδο εξόδου. Παρόλα αυτά με βάση το θεώρημα Kolmogorov (Kolmogorov 1957)[33] δύο κρυφά επίπεδα είναι αρκετά θεωρητικά για να λύσουν οποιοδήποτε πρόβλημα. Αυτό συμβαίνει γιατί μπορούν να δημιουργηθούν τόσα όρια αποφάσεων όσες και οι κλάσεις του προβλήματος. Ωστόσο για κάθε πρόβλημα μπορεί να καθοριστεί ο ιδανικός αριθμός κρυφών επιπέδων και των νευρώνων σε κάθε κρυφό επίπεδο με την βοήθεια των πειραμάτων.

Δίκτυο Εμπρόσθιου Περάσματος

Τα πολυστρωματικά δίκτυα Perceptron είναι δίκτυα εμπρόσθιου περάσματος (). Δηλαδή η ροή της πληροφορίας σε τέτοιο δίκτυο γίνεται πάντα από τα αριστερά προς τα δεξιά και δεν υπάρχει κανένας βρόγχος ανάδρασης. Αυτό σημαίνει ότι δεν ανατροφοδοτήται πληροφορία στο δίκτυο από κανένα κρυφό επίπεδο και το επίπεδο εξόδου. Επίσης οι νευρώνες σε κάθε επίπεδο συνδέονται μεταφέρουν δηλαδή πληροφορία μόνο στους νευρώνες του αμέσως επόμενου προς τα δεξιά επιπέδου.

Αλγόριθμος Οπισθοδρόμησης σφάλματος

Υπάρχουν πολλοί αλγόριθμοι εκπαίδευσης για τα Τεχνητά Νευρωνικά Δίκτυα αλλά ο πιο συνήθης για το πολυστρωματικό δίκτυο Perceptron είναι ο αλγόριθμος οπισθοδρόμησης σφάλματος. Σκοπός αυτού του αλγόριθμου είναι να βρει τα κατάλληλα βάρη έτσι ώστε η επιθυμητή κλάση για κάθε γραμμή εισόδου να συμπίπτει με το αποτέλεσμα που λαμβάνουμε τρέχοντας το δίκτυο με αυτή την είσοδο. Με βάση αυτό τον αλγόριθμο σε κάθε επανάληψη γίνεται η μετατροπή των βαρών έτσι ώστε η συνάρτηση κόστους να ελαχιστοποιείται. Ουσιαστικά τα βάρη προσαρμόζονται σύμφωνα με τα σφάλματα που υπολογίζονται σε κάθε

βήμα της διαδικασίας, δηλαδή για κάθε γραμμή εισόδου. Όταν φτάσουμε σε ένα επιθυμητό ελάχιστο σφάλμα ή όταν ξεπεραστεί ένας συγκεκριμένος αριθμός επαναλήψεων τότε θεωρούμε πως το δίκτυο έχει εκπαιδευτεί. Επομένως ο μέσος όρος της μεταβολής των βαρών από όλο το σύνολο των δεδομένων εισόδου, είναι μια εκτίμηση της μεταβολής που θα προ έκυπτε, αν ελαχιστοποιούσαμε την συνάρτηση κόστους με βάση το μέτρο εκπαίδευσης στο οποίο κατέληξε το δίκτυο.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Εξίσωση 5: Μέση τιμή της Συνάρτησης κόστους όπου n ο αριθμός των δεδομένων εισόδου, $\hat{Y}$ i η επιθυμητή έξοδος για την είσοδο i και Y i η πραγματική έξοδος του δικτύου για την είσοδο i

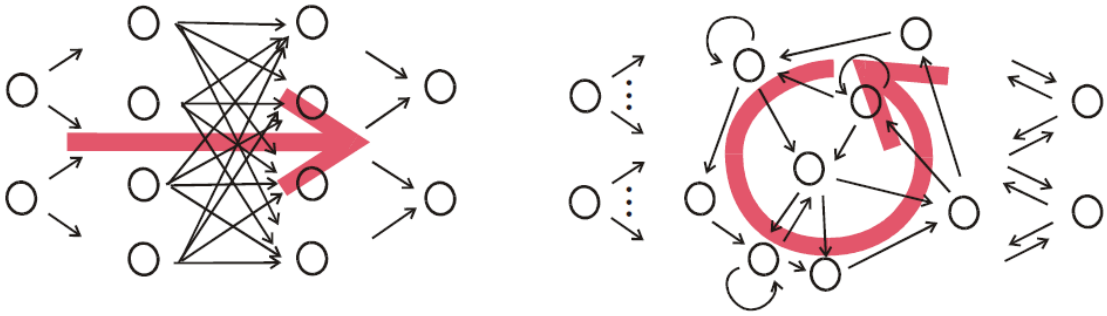
4.5. Τεχνητό Δίκτυο με Ανάδραση

4.5.1 Περιγραφή Δικτύου με Ανάδραση

Υπάρχουν δύο κύριοι τύποι Τεχνητών Νευρωνικών Δικτύων. Τα δίκτυα εμπρόσθιου περάσματος και τα δίκτυα με ανάδραση. Στα τεχνητά δίκτυα εμπρόσθιου περάσματος η ενεργοποίηση, δηλαδή η γνώση, τροφοδοτείται από την είσοδο στην έξοδο διαμέσου των κρυφών επιπέδων. Όσο αφορά την μαθηματική τους προσέγγιση τα τεχνητά δίκτυα εμπρόσθιου περάσματος μπορούν με ακρίβεια να αντιστοιχίσουν εισόδους και εξόδους, να αναπαραστήσουν δηλαδή συναρτήσεις. Παρόλα αυτά για αυτά τα δίκτυα δεν μπορούν να αναπαραστήσουν δυναμικά δεδομένα. Όπως για παράδειγμα προβλήματα των οποίων τα δεδομένα εισόδου δεν είναι ανεξάρτητα μεταξύ τους αλλά αλληλοεπιδρούν και καθιστούν έτσι απαραίτητη την διατήρηση ιστορικού. Τέτοιου είδους προβλήματα μπορούν να λυθούν με την βοήθεια δικτύων με ανάδραση.

Σε ένα δίκτυο με ανάδραση(RNN-Recurrent Neural Network) αποτελείται και αυτό από επίπεδα αποτελούμενα από μονάδες, νευρώνες με την διαφορά όμως ότι υπάρχει τουλάχιστο ένα αναδρομικό μονοπάτι μεταξύ των συναπτικών ενώσεων (βαρών). Το αναδρομικό μονοπάτι μπορεί να υλοποιείται από οποιοδήποτε επίπεδο σε ένα άλλο ανατρέποντας την κατεύθυνση του δικτύου. Με αυτή τους την ιδιότητα ανατροφοδοτείται στο δικτυού πληροφορία που είδη έχει επεξεργαστεί για να ξαναλυφθεί υπόψη. Με αυτό τον τρόπο δημιουργείται μια εσωτερική κατάσταση του δικτύου που του επιτρέπει να εμφανίσει μια

δυναμική χρονική συμπεριφορά. Εμφανίζει δηλαδή το χαρακτηριστικό της διατήρησης μνήμης. Τα δίκτυα με ανάδραση αντιπροσωπεύουν καλύτερα τα βιολογικά Νευρωνικά δίκτυα τα οποία είναι όλα αναδρομικά. Τα δίκτυα με ανάδραση βρίσκουν ανταπόκριση σε προβλήματα τα οποία απαιτούν την διατήρηση ιστορικού. (Σχήμα 123)



Σχήμα 26: Τυπική δομή ενός δικτύου εμπρόσθιου περάσματος (Αριστερά) και τυπική δομή ενός δικτύου με ανάδραση (Δεξιά) .
(Herbert Jaeger (2007) - "The "echo state")

4.5.2 Εκπαίδευση Δικτύου με Ανάδραση

Κατά την διάρκεια της τελευταίων χρόνων χρησιμοποιήθηκαν αρκετές μεθόδοι για την εκπαίδευση των RNN με επιβλεπόμενη μάθηση. Μερικές από αυτές είναι η μέθοδος ανάστροφη μετάδοση λάθους στο χρόνο (BPTT - backpropagation through time), η μέθοδος πραγματικού χρόνου μάθηση με ανάδραση (RTRL - real-time recurrent learning) και οι τεχνικές εκτεταμένου φιλτραρίσματος Kalman (EKF- extended Kalman filtering based techniques). Η μέθοδος BPTT είναι η πιο διαδεδομένη, η μέθοδος RTRL είναι η πιο μαθηματικός αποδεκτή ενώ για την EKF οι απόψεις είναι διχασμένες ως προς την αποτελεσματικότητά της αν και θεωρητικά δίνει τα καλύτερα αποτελέσματα.

Κεφάλαιο 5: Δεδομένα Εισόδου-Εξόδου

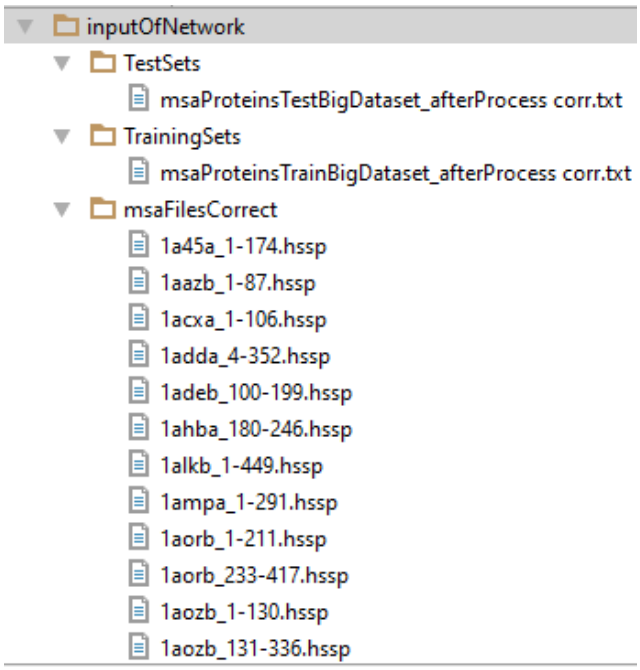
5.1 Περιγραφή Δεδομένων εισόδου

Για την επιτυχή εκπαίδευση των Τεχνητών Νευρωνικών δικτύων απαιτείται η όσο το δυνατό καλύτερη αναπαράσταση των δεδομένων εισόδου του δικτύου από τα πραγματικά δεδομένα. Με βάση αυτήν την ιδέα χρησιμοποίησα για την διπλωματική μου την τεχνική της πολλαπλής ευθυγραμμισμένης ακολουθίας πρωτεϊνών. (MSA-Multiple Sequence Alignment).

Η MSA τεχνική είναι μια από τις πιο διαδεδομένες τεχνικές στην βιοπληροφορική για την απεικόνιση των σχέσεων μεταξύ των αμινοξέων σε σύνολο πρωτεϊνών. Βάση αυτής της μεθόδους συγκρίνονται αλληλουχίες αμινοξέων ενός συνόλου πρωτεϊνών και το αποτέλεσμα είναι μια ευθυγράμμιση που εμφανίζει τα κατάλοιπα αμινοξέος για κάθε πρωτεΐνη σε μια ευθεία. Η ευθεία αυτή διαχωρίζει τα κατάλοιπα με κενά έτσι ώστε τα ισοδύναμα κατάλοιπα από κάθε αμινοξύ να εμφανίζονται στην ίδια στήλη. Με τον όρο ισοδύναμα κατάλοιπα εννοούμε τα κατάλοιπα που έχουν κοινή εξελικτική καταγωγή για την δομική βιολογία, τα κατάλοιπα που αντιστοιχούν σε ανάλογες θέσεις σε ομοιογενείς πτυχώσεις (homologous folds) σε ένα σύνολο πρωτεϊνών για την μοριακή βιολογία και γενικά τα κατάλοιπα που διαδραματίζουν τον ίδιο ρόλο αντίστοιχα στις πρωτεΐνες που ανήκουν. Ουσιαστικά η ευθυγράμμιση των πρωτείων μας παρέχει μια γενική αναπαράσταση για τα αμινοξέα που ανήκουν σε μια πρωτεΐνη. [37]

5.3 Δομή Δεδομένων

Τα δεδομένα που χρησιμοποίησα ως είσοδο στο δίκτυο μου δημιουργήθηκαν με βάση την MSA τεχνική. Η διάταξή τους είναι χωρισμένη σε φακέλους με την ακόλουθη ιεραρχία: Σχήμα 27



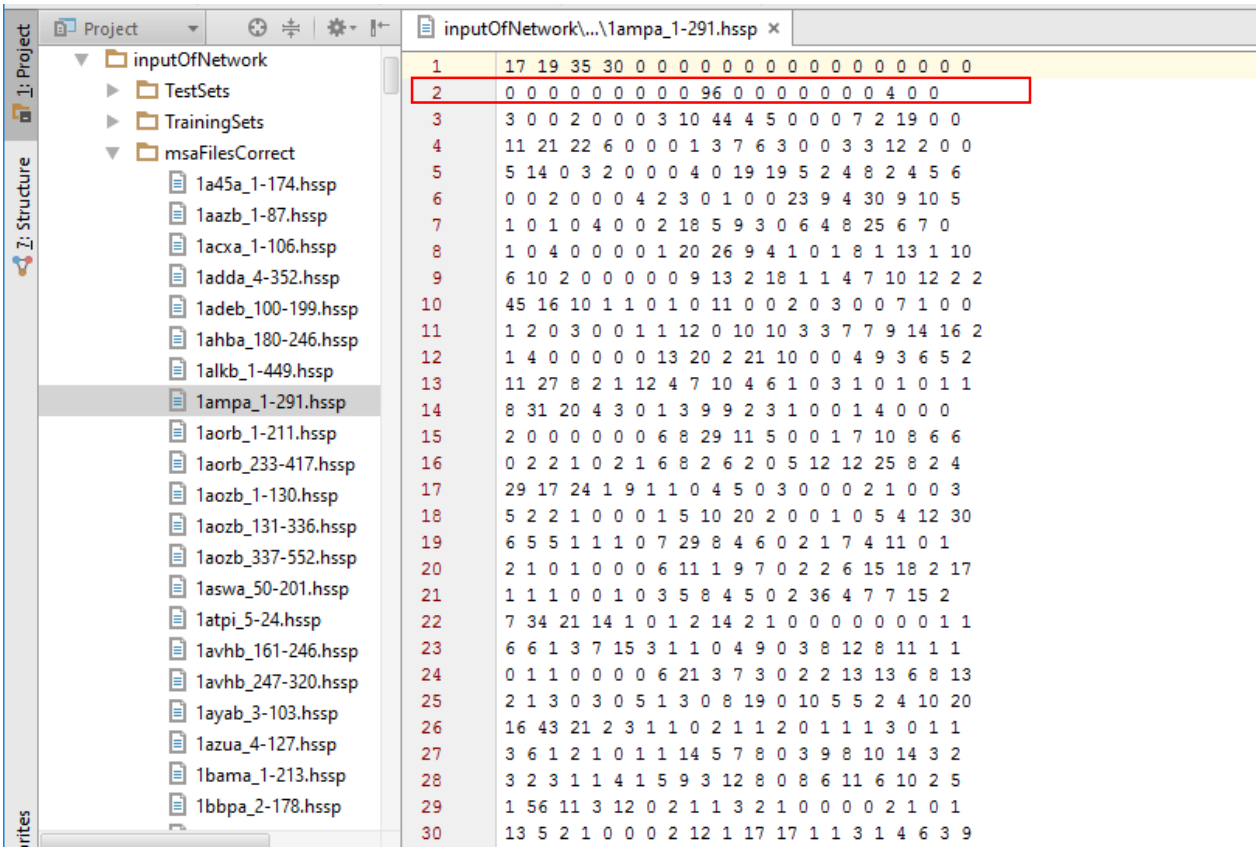
Σχήμα 27: Διάταξη αρχείων που περιέχουν τα δεδομένα εισόδου του δικτύου, όπου TestSets περιέχει τα δεδομένα ελέγχου του δικτύου, TrainingSets περιέχει τα δεδομένα εκπαίδευσης, msaFilesCorrect περιέχει τα MSA αρχεία.

Στο Σχήμα 27 παρουσιάζονται τρεις φάκελοι που περιέχουν τα δεδομένα εισόδου του δικτύου. Το TestSets το οποίο περιέχει ένα σύνολο πρωτεϊνών με το όνομα msaProteinsTestBigDataset_afterProcess corr.txt. Το σύνολο αυτό περιέχει τα δεδομένα με τα οποία το δίκτυο θα ελέγξει αν έχει γίνει σωστά η εκπαίδευση. Το TrainSets περιέχει επίσης ένα αρχείο με το όνομα msaProteinsTrainBigDataset_afterProcess corr.txt που περιέχει ένα σύνολο από πρωτεΐνες. Αυτές οι πρωτεΐνες θα είναι τα δεδομένα εισόδου στο δίκτυο και χρησιμοποιούνται για την εκπαίδευσή του. Η δομή στα δύο πιο πάνω αρχεία φαίνεται στο Σχήμα 28.

Σχήμα 28: Σύνολο πρωτεϊνών στα αρχεία msaProteinsTestBigDataset_afterProcess corr.txt και msaProteinsTrainBigDataset_afterProcess corr.txt

Κάθε τρεις γραμμές του αρχείου αναπαριστούν μια πρωτεΐνη.(π.χ. οι γραμμές στο κόκκινο κουτί) Η πρώτη γραμμή για κάθε πρωτεΐνη αντιπροσωπεύει το όνομα της, η δεύτερη γραμμή αντιπροσωπεύει την Πρωτοταγή δομή της και η τρίτη γραμμή την Δευτεροταγή της δομή. Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια για το πρόβλημα της πρόβλεψης της δευτεροταγούς δομής επιθυμούμε με βάση την πρωτοταγή δομή της πρωτεΐνης να βρούμε την δευτεροταγή. Με βάση τα δεδομένα στο Σχήμα 28 η πρωτοταγής δομή αποτελεί την είσοδο στο δίκτυο (δεύτερη γραμμή για κάθε πρωτεΐνη) και η δευτεροταγής δομή αποτελεί την επιθυμητή έξοδο για το δίκτυο (τρίτη γραμμή για κάθε πρωτεΐνη). Με βάση την MSA τεχνική γίνεται αντιστοίχηση των αμινοξέων της πρωτοταγούς δομής και το αποτέλεσμα είναι μια ευθεία για κάθε αμινοξύ όπου εμφανίζονται τα κατάλοιπα του από την σύγκρισή του με τα 20 α-αμινοξέα. Αυτή η αναπαράσταση των αμινοξέων της κάθε πρωτεΐνης βρίσκεται στον φάκελο με το όνομα msaFilesCorrect. Τα αμινοξέα για κάθε πρωτεΐνη περιέχονται στο αρχείο με όνομα το όνομα της πρωτεΐνης αυτής. Για παράδειγμα για να μπορέσω να έχω πρόσβαση στην MSA αναπαράσταση των αμινοξέων μιας πρωτεΐνης την οποία διαβάζω από το αρχείο εκπαίδευσης ή από το αρχείο ελέγχου, θα πρέπει να διαβάσω το όνομά της και με βάση αυτό να βρω το αρχείο από το φάκελο msaFilesCorrect που του

αντιστοιχεί. Η δομή των αρχείων φαίνεται στο Σχήμα 27. Ακολούθως για κάθε αμινοξύ μιας πρωτεΐνης διαβάζω από το αρχείο με το όνομα της πρωτεΐνης (π.χ. nameOfprotein.hssp), την MSA αναπαράστασή του ανάλογα με την θέση του στην πρωτεϊνική ακολουθία. Για την πρωτεΐνη στο Σχήμα 27 που βρίσκεται στο κόκκινο κουτί, το αρχείο με την MSA αναπαράσταση των αμινοξέων της φαίνεται στο Σχήμα 28.



Σχήμα 28: MSA αναπαράσταση της πρωτοταγούς δομής των αμινοξέων της πρωτεΐνης 1bamA_1-213 από το φάκελο msaFilesCorrect.

Όπως αναφέρθηκε πιο πάνω η MSA αναπαράσταση για κάθε αμινοξύ είναι μια ευθεία που αποτελεί τα κατάλοιπά του από την σύγκριση του με τα 20 α-αμινοξέα. Άρα από το Σχήμα 28 κάθε γραμμή αναπαριστά το αμινοξύ στην θέση που βρίσκεται στην πρωτοταγή δομή της πρωτεΐνης και κάθε στήλη την σύγκριση του κάθε αμινοξέως με την κάθε κατηγορία των 20

α-αμινοξέων. Το άθροισμα των στοιχείων κάθε γραμμής είναι 100 επειδή το κάθε στοιχείο αντιπροσωπεύει ποσοστικά την σύγκριση.

5.4 Προεπεξεργασία Δεδομένων

54.1 Κλάση StoreProtein.py

Για την προεπεξεργασία των δεδομένων ακολούθησα την διαδικασία που αναφέρθηκε πιο πάνω για να διατρέξω τις πρωτεΐνες και να θέσω ως είσοδο στο δίκτυο που υλοποίησα τις MSA αναπαραστάσεις της κάθε πρωτεΐνης . Πιο συγκεκριμένα δημιούργησα μια κλάση StorePython.py στην γλώσσα προγραμματισμού python στην οποία διαβάζω όλες τις πρωτεΐνες από τα αρχεία εκπαίδευσης και ελέγχου και για κάθε μια από αυτές ανέτρεξα στο φάκελο για να βρω την MSA αναπαράσταση των αμινοξέων της . Ακολούθως αποθηκεύω κάθε γραμμή στο MSA αρχείο η οποία αποτελεί την είσοδο του δικτύου. Για το δίκτυο αποθήκευσα τις πιο κάτω μεταβλητές:

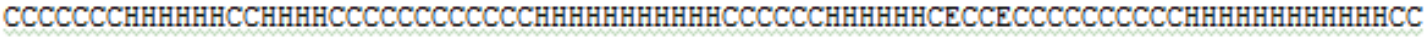
Μεταβλητή Data:

Αυτή η μεταβλητή ουσιαστικά αποτελείται από μια λίστα η οποία αποθηκεύει τα δεδομένα εισόδου του δικτύου. Τα δεδομένα εισόδου είναι τα αμινοξικά κατάλοιπα της κάθε πρωτεΐνης τα οποία παίρνω από το αρχείο εκπαίδευσης και το αρχείο ελέγχου. Όπως φαίνεται και στο Σχήμα 28 το άθροισμα των στοιχείων της κάθε γραμμής είναι 100. Η γραμμή 2 η οποία αντιπροσωπεύει ένα αμινοξύ, συγκριτικά μοιάζει 96% με το αμινοξύ στην στήλη 10 και 4% με την στήλη 18. Διαβάζοντας κάθε στοιχείο μιας γραμμής το διαιρώ με 100 έτσι ώστε να το φέρω στο διάστημα [0,1].

Μεταβλητή Yt:

Στην μεταβλητή Yt αποθηκεύεται το επιθυμητό αποτέλεσμα. Το επιθυμητό αποτέλεσμα ορίζεται από την δευτεροταγή δομή κάθε πρωτεΐνης που βρίσκεται στο αρχείο msaProteinsTrainBigDataset_afterProcess και στο αρχείο msaProteinsTestBig Dataset_ after Process. Η Δευτεροταγής δομή είναι ουσιαστικά μια ακολουθία από γράμματα κάθε ένα από τα οποία αντιστοιχεί με ένα τύπο δευτεροταγούς δομής. Για να τα αναπαραστήσω στο δίκτυο χρειάστηκε να ταυτίσω τον κάθε τύπο δευτεροταγούς δομής με ένα από τους τρεις νευρώνες στο επίπεδο εξόδου. Με αυτό τον τρόπο αν κατά την έξοδο του δικτύου ενεργοποιηθεί ο

πρώτος νευρώνας τότε θεωρούμε ότι το αμινοξύ εισόδου ανήκει στην κατηγορία α-έλικας (C) , αν ενεργοποιηθεί ο δεύτερος νευρώνας τότε το αμινοξύ εισόδου ανήκει στην κατηγορία β –πτυχωτής επιφάνειας (H) και αντίστοιχα αν ενεργοποιηθεί ο τρίτος νευρώνας τότε ανήκει στην 3η κατηγορία (E).



Σχήμα 29: Παράδειγμα Δευτεροταγούς Δομής

Άρα με βάση την πιο πάνω ιδέα, το y^{target} στο δίκτυο είναι ένας πίνακας ο οποίος περιέχει τις αναπαραστάσεις των αμινοξέων του συνόλου πρωτεϊνών με τον εξής τρόπο:

Νευρώνας εξόδου που Ενεργοποιούνται	**Τύποι Δευτεροταγούς Δομής**	**Αναπαράσταση στον y^{target} πίνακα**
1	Αμινοξέα με τύπο α-έλικας	[1, 0, 0]
2	Αμινοξέα με τύπο β-πτυχωτής επιφάνειας	[0, 1, 0]
3	Αμινοξέα με τύπο διαφορετικό από τα πιο πάνω	[0, 0, 1]

Σχήμα 30: Πίνακας που αναπαριστά το επιθυμητό αποτέλεσμα y^{target} για κάθε αμινοξύ εισόδου.

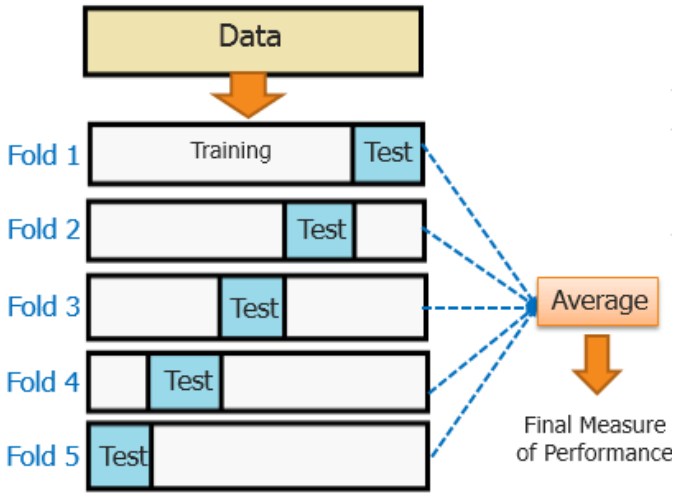
Μεταβλητή SizeOfAminoacids

Η μεταβλητή αυτή είναι ένας μετρητής ο οποίος κρατά τον αριθμό των αμινοξέων που περιέχονται στο σύνολο δεδομένων εισόδου Data

5.5 Cross Validation

Το cross Validation είναι μια τεχνική επικύρωσης η οποία χρησιμοποιείται για την αξιολόγηση της γενίκευσης που πέτυχε ένα δίκτυο. Αξιολογεί πως το δίκτυο αντιδρά σε ανεξάρτητα σύνολα δεδομένων όπως τα σύνολα ελέγχου. Χρησιμοποιείται κυρίως σε προβλήματα τα οποία αποσκοπούν στην πρόβλεψη και στα οποία επιθυμούμε να γνωρίζουμε πόσο ακριβής είναι η πρόβλεψη του αποτελέσματος του δικτύου. Ο σκοπός

του Cross Validation είναι να καθορίσει ένα σύνολο δεδομένων για τον έλεγχο του δικτύου κατά την εκπαίδευσης, έτσι ώστε να περιορίσει προβλήματα όπως υπερεκπαίδευση του δικτύου και να μας δώσει μια εικόνα για το πώς το δίκτυο θα γενικεύσει δεδομένου ενός ανεξάρτητου συνόλου δεδομένων. Για αυτό το λόγο χρησιμοποίησα cross validation. Αρχικά δημιούργησα ένα αρχείο στο οποίο αποθήκευσα το σύνολο των δεδομένων εισόδου και το σύνολο των δεδομένων ελέγχου. Ακολούθως διαβάζοντας τα δεδομένα εισόδου κατά την εκκίνηση του δικτύου τα χώρισα σε 10 κομμάτια. Τρέχω το δίκτυο μου όσες φορές το έσπασα και κάθε φορά χρησιμοποιώ τα 9/10 ως σύνολο εκπαίδευσης και το άλλο 1/10 ως σύνολο ελέγχου. Όπως φαίνεται στο Σχήμα 31 τα δεδομένα εναλλάσσονται μεταξύ του συνόλου εκπαίδευσης και ελέγχου έτσι ώστε να περάσουν και από τις δύο διαδικασίες. Τέλος το σφάλμα αλλά και η ακρίβεια του αποτελέσματος του δικτύου καθορίζεται από τον μέσο όρο όλων των επαναλήψεων.



Σχήμα 31: Η διαδικασία Cross validation με διαχωρισμό των δεδομένων σε 5 περιόδους.
(http://www.edureka.co/blog/implementation-of-decision-tree/)

Κεφάλαιο 6: Echo State Network

6.1 Εισαγωγή

Όπως έχει αναφερθεί και σε πιο πάνω κεφάλαια η δυσκολία του προβλήματος της πρόβλεψης της δευτεροταγής δομής των πρωτείων βασίζεται στο γεγονός ότι το κάθε αμινοξύ ασκεί δυνάμεις έλκτικές ή αποστικές σε πολλά από τα αμινοξέα μιας πρωτεϊνικής ακολουθίας. Παρόλο που γνωρίζουμε ποιες δυνάμεις ασκούνται ανάλογα με τον τύπο του αμινοξέως εντούτοις δεν γνωρίζουμε ποια αμινοξέα αλληλοεπιδρούν μεταξύ τους για να μπορούμε να βρούμε την δευτεροταγή δομή των πρωτεϊνών. Ο καλύτερος αλγόριθμος για τον καθορισμό των αμινοξέων που αλληλοεπιδρούν μεταξύ τους μέχρι στιγμής, είναι αυτός του Baldi που υλοποιήθηκε στο BRNN δίκτυο[19][20] με ακρίβεια Q3 76%. Με βάση αυτό το δίκτυο κάθε αμινοξύ θεωρήθηκε ότι αλληλοεπιδρά περισσότερο με τα κοντινότερα γειτονικά του αμινοξέα και από δεξιά και αριστερά ενώ η αλληλεπίδραση αυτή εξασθενεί με την αύξηση της απόσταση του από τα γειτονικά αμινοξέα, σε μια πρωτεϊνικής ακολουθία.

Με βάση το πιο πάνω καταλαβαίνουμε ότι η διατήρηση ιστορικού είναι απαραίτητη για την λύση αυτού του προβλήματος. Λαμβάνοντας υπόψη μου την ιδέα αυτή θεώρησα ότι μια καλή προσέγγιση για την επίλυση του προβλήματος της πρόβλεψης των πρωτεϊνών θα ήταν η υλοποίηση ενός Echo state Network[21][22].(Υποκεφάλαιο 6.6 Επιλογή Echo State Δικτύων)

6.2 Περιγραφή Echo state Network

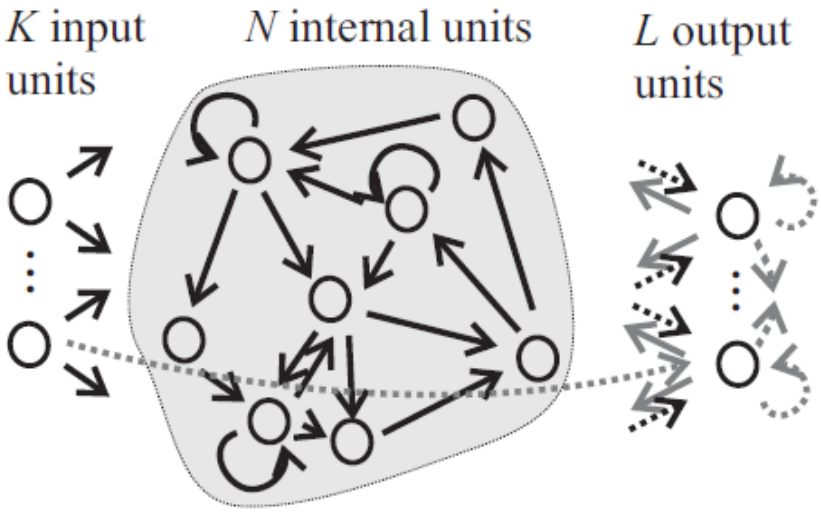
Το Echo state Network(ESN) επινοήθηκαν από τον Herbert Jaeger το 2002 [21][22] και συγκαταλέγεται στην κατηγορία των τεχνητών νευρωνικών δικτύων με ανάδραση διακριτού χρόνου.(Discrete Time Recurrent Neural Network). Αυτό το δίκτυο εκπαιδεύεται με επιβλεπόμενη μάθηση και η αρχιτεκτονική του είναι πολύ παρόμοια με την αρχιτεκτονική των δικτύων με ανάδραση με την διαφορά ότι το Echo State network χρησιμοποιεί Reservoir Computing για την επίλυση προβλημάτων. Το Reservoir Computing είναι μια προσέγγιση για σχεδιασμό, εκπαίδευση και ανάλυση νευρωνικών δικτύων με ανάδραση (Υποκεφάλαιο 5.4.1 Δημιουργία Dynamic Reservoir(DR)) .

6.3 Αρχιτεκτονική του δικτύου

Σκοπός της εκπαίδευσης των ESN είναι να εμφανίσουν την ιδιότητα echo states η οποία εν μέρει καθορίζεται από την αρχιτεκτονική τους. Όπως και στα RNN το ESN χωρίζεται σε

επίπεδα με την διαφορά ότι τα διακριτά επίπεδα για το Echo State Network είναι μόνο αυτά της εισόδου και της εξόδου. Οι νευρώνες του επιπέδου εισόδου u(n) και οι νευρώνες του επιπέδου εξόδου y(n) καθορίζονται από το πρόβλημα . Όπως φαίνεται και στο Σχήμα 28 τα κρυφά επίπεδα είναι ομαδοποιημένα σε μια περιοχή που ονομάζεται δυναμικό reservoir (DR - Dynamical Reservoir) και ο αριθμός τους δεν διακρίνεται. Οι νευρώνες στο DR, x(n), είναι συνδεδεμένοι μεταξύ τους με κάποιο ποσοστό. Το ποσοστό αυτό καθορίζει το πόσο αραιό (sparsity) θα είναι το DR και για κάθε πρόβλημα καθορίζεται μέσω της πειραματικής μεθόδου.

Οι συναπτικές ενώσεις που ενώνουν τα επίπεδα μεταξύ τους και το DR χαρακτηρίζονται από μια τιμή που προσδιορίζει τα βάρη. Στα ESN ο κάθε νευρώνας εισόδου συνδέεται μέσω των βαρών W^{in}_{ij} (i-νευρώνας εισόδου, j-νευρώνας στο DR) με κάθε νευρώνα από το DR. Αυτά τα βάρη καθορίζονται τυχαία αλλά κανονικοποιημένα πριν από την εκπαίδευση και οι τιμές τους είναι οι τελικές, δεν αλλάζουν κατά την διάρκεια της εκπαίδευσης. Επίσης ο κάθε νευρώνας από το DR συνδέεται μέσω βαρών W_{jk} (j-νευρώνας στο DR, κ-νευρώνας DR και $j \neq k$) με κάθε άλλο νευρώνα στο DR. Και αυτοί οι νευρώνες καθορίζονται τυχαία και κανονικοποιημένα πριν από την εκπαίδευση και οι τιμές τους δεν αλλάζουν. Τέλος ο κάθε νευρώνας από το DR συνδέεται μέσω βαρών W^{out}_{jm} (j-νευρώνας στο DR, m-νευρώνας εισόδου) με τους νευρώνες του επιπέδου εξόδου. Αυτά τα βάρη εκπαιδεύονται για να πάρουν τις τελικές τους τιμές . Στο Σχήμα 28 επίσης φαίνονται επιπλέον, συναπτικές ενώσεις μεταξύ του επιπέδου εισόδου και του επιπέδου εξόδου, συναπτικές ενώσεις μεταξύ των νευρώνων του επιπέδου εξόδου και συναπτικές ενώσεις μεταξύ του επιπέδου εξόδου και του DR. Οι συνδέσεις αυτές είναι προαιρετικές και δεν έχουν χρησιμοποιηθεί σε αυτή την διπλωματική [21][22].



Σχήμα 28: Βασική Τοπολογία ενός Echo State δικτύου όπου K ο αριθμός των νευρώνων στο επίπεδο εισόδου, N ο αριθμός των νευρώνων στο DR, L ο αριθμός των νευρώνων στο επίπεδο εξόδου. Οι διακεκομμένες μαύρες γραμμές δηλώνουν τα βάρη τα οποία εκπαιδεύονται, Οι μαύρες συμπαγής γραμμές δηλώνουν βάρη τα οποία είναι υποχρεωτικά για την λειτουργία του δικτύου ενώ οι γαλάζιες διακεκομμένες δεν είναι υποχρεωτικά συναπτικά βάρη.
(Herbert Jaeger (2007) - "The "echo state")

6.4 Δημιουργία Δικτύου

Αρχικά καθορίζουμε με βάση τις απαιτήσεις του προβλήματός, πόσοι νευρώνες θα χρησιμοποιηθούν στο επίπεδο εισόδου και στο επίπεδο εξόδου. Για το PSSP και με βάση τα δεδομένα εισόδου που σχολιάστηκαν στο Κεφάλαιο 5 οι νευρώνες που χρησιμοποιήθηκαν στο επίπεδο εισόδου είναι 20 και αντιπροσωπεύουν τα κατάλοιπα ενός αμινοξέως. Οι αναγκαίοι νευρώνες της εξόδου για το PSSP πρόβλημα είναι 3 και αντιπροσωπεύουν τους πιθανούς τύπους δευτεροταγούς δομής στους οποίους είναι πιθανό να ανήκει το κάθε αμινοξύ.

Ακολούθως θα πρέπει να καθοριστεί ο αριθμός νευρώνων στο DR. Το μέγεθος του reservoir εξαρτάται από την πολυπλοκότητα του προβλήματος. Ο ιδανικός πληθικός αριθμός νευρώνων στο DR μπορεί να καθοριστεί μέσω της πειραματικής μεθόδου. Για την υλοποίηση του ESN δικτύου για την επίλυση του PSSP προβλήματος χρησιμοποίησα αρχικά 500 νευρώνες.

Ο αριθμός των βαρών του δικτύου καθορίστηκε αντίστοιχα από τα επίπεδα που το συνέδεαν ως εξής:

Όπως αναφέρθηκε πιο πάνω τα βάρη που συνδέουν το επίπεδο εισόδου και το DR παίρνουν τυχαίες κανονικοποιημένες τιμές και ο πληθικός αριθμός τους με βάση το Σχήμα 28 είναι (K+1)×N όπου (K+1) ο αριθμός νευρώνων στο επίπεδο εισόδου μαζί με το κατώφλι .

Με την ίδια λογική, τα βάρη που συνδέουν τους νευρώνες στο DR μεταξύ τους έχουν πληθικό αριθμό N×N και παίρνουν τιμές τυχαίες και κανονικοποιημένες.

Αντίστοιχα τα βάρη που ενώνουν τον κάθε νευρώνα από το DR με κάθε νευρώνα από το επίπεδο εξόδου έχουν πληθικό αριθμό N×L. Αυτά τα βάρη δεν παίρνουν τυχαίες τιμές εφόσον οι τιμές τους καθορίζονται από την εκπαίδευση του δικτύου, με βάση την Εξίσωση 10.

6.4.1 Δημιουργία Dynamic Reservoir(DR)

Η ιδέα πίσω από το reservoir είναι ότι επιτρέπει προηγούμενες καταστάσεις του δικτύου να ηχούν ακόμα και μετά από το πέρασμά τους. Έτσι αν το δίκτυο παραλάβει μια γραμμή εισόδου παρόμοια με δεδομένα στα οποία έχει εκπαιδευτεί θα ακολουθείται η κατάλληλη τροχιά ενεργοποίησης στο reservoir ούτως ώστε να παράγεται το ταιριαστό σήμα εξόδου και σε περίπτωση που το δίκτυο έχει εκπαιδευτεί ικανοποιητικά θα καταφέρει να γίνει γενίκευση από τα δεδομένα με τα οποία έχει εκπαιδευτεί. Για να δημιουργηθεί το ιδανικό reservoir είναι σημαντικό να κατανοηθεί η λειτουργικότητα που επιτελεί. Στην πράξη είναι σημαντικό να έχουμε υπόψη μας ότι το reservoir δρα ως μη γραμμική επέκταση των δεδομένων εισόδου(α) και ως μνήμη (β).

(α) Το reservoir είναι μια μεγαλύτερης διάστασης μη γραμμική αναπαράσταση x(n) των δεδομένων εισόδου u(n). Για παράδειγμα σε προβλήματα κατηγοριοποίησης όπου τα δεδομένα εισόδου είναι μη γραμμικώς διαχωρίσιμα στο χώρο R^{Nu}, όταν μετατεθούν στον χώρο R^{Nx} των x(n) γίνονται διαχωρίσιμα από το W^{out}. Μπορείς να καθορίσεις το reservoir χρησιμοποιώντας το "τέχνασμα πυρήνα" ("kernel trick") με βάση το οποίο δοκιμάζουμε όλα τα πιθανά reservoir, στο πλαίσιο του Reservoir Computing. Αυτή η τεχνική ωστόσο δεν είναι ιδιαίτερα πρακτική [25].

(β) Παράλληλα το reservoir χρησιμοποιείται για να αποθηκεύσει δεδομένα ως εσωτερική μνήμη, παρέχοντας χρονικό περιεχόμενο (temporal context). Αυτός είναι και ο λόγος που χρησιμοποιείται αρχιτεκτονική παρόμοια με το RNN για να μπορεί να διασφαλιστεί η διατήρηση ιστορικού.

Συνδυάζοντας και τους δύο πιο πάνω παράγοντες το DR θα πρέπει να μπορεί να παρέχει ένα σχετικό χώρο x(n), έτσι ώστε να παράγεται από αυτό η επιθυμητή έξοδος y. Ωστόσο, στις περισσότερες περιπτώσεις δεν μπορεί να δημιουργηθεί το ιδανικό reservoir και για αυτό θα πρέπει να γίνει κάποιος συμβιβασμός μεταξύ των ιδιοτήτων του (α) και (β) και συνεπώς των παραμέτρων του DR [24].

Με βάση την αρχιτεκτονική δομή του ESN, το DR ορίζεται από τρεις βασικές παραμέτρους (W^{in} ,W, α) .(Εξίσωση 7,8). Τα βάρη W^{in} και W αρχικοποιούνται τυχαία με βάση κάποιες παραμέτρους και το leaking rate α επιλέγεται ως ανεξάρτητη παράμετρος. Οι καθολικές παράμετροι για το reservoir είναι :

Το μέγεθος του reservoir:

Το μέγεθος του reservoir είναι μια από τις πιο βασικές παραμέτρους. Θωρείται γενικώς αποδεκτή η άποψη ότι όσο πιο μεγάλο είναι το reservoir τόσο καλύτερη θα είναι και η απόδοση που θα εξασφαλιστεί . Λόγω του ότι τα ESN δεν έχουν πολύ μεγάλο υπολογιστικό κόστος σε πολλές περιπτώσεις το μέγεθος του reservoir φτάνει την τιμή μέχρι και 10^4[23]. Μεγαλύτερο μέγεθος του reservoir εξυπακούει πιο εύκολη εύρεση ενός γραμμικού συνδυασμού που να μπορεί να παράξει το επιθυμητό αποτέλεσμα. Το κατώτερο όριο για το μέγεθος του reservoir μπορεί να υπολογιστεί προσεγγιστικά με βάση το επιθυμητό αριθμό από τιμές τις οποίες θα πρέπει το δίκτυο να θυμάται. Άρα ο μεγαλύτερος αριθμός από τιμές που θα αποθηκευτούν δεν πρέπει να ξεπερνά το N^x δηλαδή το μέγεθος του reservoir [21][22].

Sparsity του reservoir:

Η μεταβλητή Sparsity του reservoir δηλώνει πόσο αραιές θα είναι οι συνδέσεις μεταξύ των νευρώνων του DR και είναι μια παράμετρος που καθορίζεται κατά την δημιουργία του δικτύου. Σε πολλές προσεγγίσεις ενθαρρύνεται η χρησιμοποίηση αραιού reservoir γιατί δίνει

ελαφρώς καλύτερα αποτελέσματα. Παρόλα αυτά σε σχέση με άλλες παραμέτρους το Sparsity δεν έχει υψηλή προτεραιότητα για καλυτέρευση με την έννοια ότι δεν επηρεάζει σε μεγάλο βαθμό την λειτουργικότητα του δικτύου.

Κλιμάκωση του W:

Όπως αναφέρθηκε και πιο πάνω οι τιμές των βαρών στο W δίνονται τυχαία και κανονικοποιημένα και η κλιμάκωση που χρησιμοποιείται σε αυτά τα βάρη συνήθως είναι η ίδια με αυτή που χρησιμοποιείται στα βάρη W^{in}.

Το leaking rate α:

Το leaking rate α των νευρώνων του reservoir μεταφράζεται και ως η ταχύτητα με την οποία το δίκτυο θα αναβαθμίσει το reservoir σε σχέση με το χρόνο. Δηλαδή το πόσο γρήγορα θα πάρουν οι νευρώνες του reservoir την ιδανική τιμή. Η τιμή της μεταβλητής αυτής μπορεί να προκύψει από τον χρόνο που χρειάζεται η είσοδος του δικτύου να μετατραπεί στην επιθυμητή έξοδο. Με βάση την πιο πάνω ιδέα καταλαβαίνουμε ότι είναι δύσκολο να καθοριστεί μια συγκεκριμένη τιμή για το leaking rate α και συνεπώς η ιδανική τιμή μπορεί να βρεθεί μέσω της πειραματικής μεθόδου.

Spectral radius των βαρών του DR:

Μια από τις πιο σημαντικές καθολικές παραμέτρους του reservoir είναι η φασματική ακτίνα (Spectral radius) των βαρών W του DR. Αυτή η παράμετρος εκφράζει την μέγιστη ιδιοτιμή του reservoir και θέτει μια κλίμακα στα βάρη W. Ορίζει ουσιαστικά την μέγιστη τιμή που μπορούν να πάρουν οι μη μηδενικές ενώσεις του reservoir. Έχει εξαιρετική σημασία για την διατήρηση της ιδιότητας Echo state με βάση την οποία το ιστορικό που διατηρείται θα πρέπει ξεθωριάζει σε μεγάλο χρονικό διάστημα και να μην εξαρτάται στις αρχικές συνθήκες του δικτύου. Σε περιπτώσεις που η παράμετρος αυτή οριστεί πολύ μεγάλες τιμές δημιουργείτε στο δίκτυο μια χαοτική κατάσταση στην οποία τα βάρη του reservoir αλλάζουν ανεξέλεγκτα και το δίκτυο δεν εκπαιδεύεται.

6.5 Εκπαίδευση Echo state Network

6.5.1 Σκοπός

Το Echo state Network εκπαιδεύεται με επιβλεπόμενη μάθηση όπου για μια είσοδο u(n) ∈ R^{Nu} γνωρίζουμε το επιθυμητό αποτέλεσμα $y^{target}(n) \in R^{Ny}$. Η μεταβλητή n αντιπροσωπεύει το διακριτό χρόνο και κυμαίνεται μεταξύ του διαστήματος n = 1,...,T όπου T ο αριθμός των δεδομένων εισόδου στο σύνολο εκπαίδευσης (training set). Μια είσοδος αντιπροσωπεύει ένα αμινοξύ μιας πρωτεΐνης και την δεδομένη στιγμή n είναι ένα διάνυσμα με 20 τιμές(διαστάσεις), όσες και τα 20 γνωστά α-αμινοξέα. Η επιθυμητή έξοδος ytarget(n) και η πραγματική έξοδος y(n) του δικτύου είναι διανύσματα 3 τιμών που αντιπροσωπεύουν τους 3 πιθανούς τύπους της δευτεροταγούς δομής.

Σκοπός της εκπαίδευσης του δικτύου είναι να μάθει ένα μοντέλο με έξοδο $y(n) \in R^{Ny}$, όπου το y(n) να ταυτίζεται με όσο το δυνατό μεγαλύτερη ακρίβεια με το ytarget(n), μειώνοντας το σφάλμα E(y, ytarget) και επίσης να αποκτήσει την δυνατότητα να μπορεί να γενικεύσει και να προβλέψει δεδομένα τα οποία δεν έχει ξαναδεί.

Η εξίσωση που καθορίζει το σφάλμα E είναι συνήθως το Mean-Square Error(MSE) ή το Root-Mean-Square Error(RMSE). Σε περιπτώσεις που η έξοδος του δικτύου είναι πολυδιάστατη χρησιμοποιείται η συνάρτηση Normalized-Root-Mean-Square Error (NRMS). Η συνάρτηση NRMS, Εξίσωση 6, έχει απόλυτη ερμηνεία δηλαδή το αποτέλεσμά της δεν εξαρτάται από την αυθαίρετη κλιμάκωση του $y^{target}(n)$. Λόγω της πολυδιάστατης εξόδου του δικτύου για το PSSP πρόβλημα, θα χρησιμοποιηθεί για την εύρεση του σφάλματος τη συνάρτηση NRMS.

$$E(\mathbf{y}, \mathbf{y}^{\text{target}}) = \frac{1}{N_y} \sum_{i=1}^{N_y} \sqrt{\frac{1}{T} \sum_{n=1}^{T} \left(y_i(n) - y_i^{\text{target}}(n)\right)^2}$$

Εξίσωση 6: Η εξίσωση υπολογίζει το Normalized-Root-Mean-Square Error (NRMS), όπου N_y πληθικός αριθμός εξόδου δικτύου, T αριθμός training set, y(n) output δικτύου και $y^{target}(n)$ επιθυμητή έξοδος δικτύου (Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks"-2012)

6.5.2 Συναρτήσεις Αναβάθμισης

Οι νευρώνες του επιπέδου εισόδου είναι ουσιαστικά ανενεργοί εφόσον δεν πραγματοποιούν οποιαδήποτε υπολογισμό. Σκοπός τους είναι να μεταβιβάσουν την είσοδο του δικτύου προς το DR.

Αναβάθμιση Νευρώνων Dynamic Reservoir(DR)

Οι νευρώνες στο DR αναβαθμίζουν τις τιμή τους μέσω της Εξίσωσης 7 και της Εξίσωσης 8.

$$\tilde{\mathbf{x}}(n) = \tanh\left(\mathbf{W}^{\text{in}}[1; \mathbf{u}(n)] + \mathbf{W}\mathbf{x}(n-1)\right)$$

Εξίσωση 7: Το $\tilde{x}(n) \in R^{Nx}$ καθορίζει τις τιμές αναβάθμισης για κάθε νευρώνα του DR όπου tanh συνάρτηση αναβάθμισης , u(n) είσοδος την χρονική στιγμή n και 1 δηλώνει την τιμή κατωφλίου(bias)
(Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks"-2012)

$$\mathbf{x}(n) = (1-\alpha)\mathbf{x}(n-1) + \alpha\tilde{\mathbf{x}}(n)$$

Εξίσωση 8: Το $x(n) \in R^{Nx}$ είναι ένα διάνυσμα που περιέχει τους νευρώνες του DR, όπου α leaky integration και $\tilde{x}(n) \in R^{Nx}$ η τιμή της αναβάθμισης που προκύπτει από την Εξίσωση 7.
(Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks"-2012)

Στην Εξίσωση 7 δίνονται τιμές στο διάνυσμα $\tilde{x}(n) \in R^{Nx}$ με βάση την συνάρτηση αναβάθμισης υπερβολικής εφαπτομένης (tanh). Η τελική τιμή που θα πάρουν οι νευρώνες στο DR είναι από την Εξίσωση 8. Στην χρονική στιγμή n εφαρμόζουμε την συνάρτηση $\tanh(\cdot)$ για κάθε τιμή σε ένα διάνυσμα. Μπορούν να χρησιμοποιηθούν και άλλες συναρτήσεις αναβάθμισης εκτός από την $\tanh(\cdot)$, παρόλα αυτά για την επίλυση του PSSP προβλήματος χρησιμοποιήθηκε η πιο πάνω συνάρτηση. Τα βάρη $W^{in} \in R^{Nx\times(1+Nu)}$ και $W \in R^{Nx\times Nx}$ είναι τα βάρη εισόδου και τα βάρη του reservoir αντίστοιχα. Η μεταβλητή $\alpha \in (0, 1]$ είναι το leaking Rate. Επίσης μπορούμε αναθέτοντας στο α την τιμή 1 ($\alpha=1$) να αποφύγουμε να έχουμε leaky integration στην αναβάθμιση των νευρώνων και επομένως $\tilde{x}(n) \equiv x(n)$.

Initial Transient (Αρχική Πάροδος)

Συνήθως τα αρχικά αποτελέσματα των νευρώνων x(n) απορρίπτονται δηλαδή δεν χρησιμοποιούνται για την εκπαίδευση των βαρών W^{out}. Αυτή η τεχνική χρησιμοποιείται για

να διαφυλάξουμε τα δεδομένα μας από τον θόρυβο που μπορεί να προκύψει κατά τα αρχικά στάδια, όπου τα βάρη W^{in} και W, τα οποία συμβάλλουν για τις τιμές των x(n), αρχικοποιούνται τυχαία. Έτσι αποφεύγουμε να ρυθμίσουμε αυθαίρετα τις τιμές x(n) για την εκπαίδευση και συγκεκριμένα τις τιμές x(n)=0, οι οποίες οδηγούν σε μια αφύσικη κατάσταση εκκίνησης του δικτύου. Η ανάθεση του χρόνου που θα χρειαστεί για να αποφύγουμε την πιο πάνω κατάσταση εξαρτάται από την μνήμη του δικτύου και τυπικά ορίζεται σε δεκάδες ή εκατοντάδες.

Αναβάθμιση νευρώνων στο επίπεδο εξόδου

Η αναβάθμιση των Νευρώνων στο επίπεδο εξόδου προκύπτει από την Εξίσωση 9 με βάση την οποία η οι νευρώνες $y(n) \in R\ N^y$ καθορίζονται από το εσωτερικό γινόμενο των βαρών εξόδου $W^{out} \in R\ N^{y\times(1+Nu+Nx)}$ και του διανύσματος που δημιουργείτε με την συνένωση της τιμής του κατωφλίου, των διανυσμάτων $u(n) \in R^{Nu\ y(n)}$ και $x(n) \in R\ N^x$.

$$\mathbf{y}(n) = \mathbf{W}^{\text{out}}[1; \mathbf{u}(n); \mathbf{x}(n)]$$

Εξίσωση 9: Αναβάθμιση νευρώνων $y(n) \in R\ N^y$ στο επίπεδο εξόδου.
(Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks"-2012)

Αλλαγή των Βαρών Εξόδου W^{out}

Η αλλαγή των βαρών προκύπτει από την Εξίσωση 10 κατά την οποία προσπαθούμε να τα μετατρέψουμε έτσι ώστε το αποτέλεσμα του δικτύου y(n) να μπορεί να προσεγγίσει το επιθυμητό αποτέλεσμα $y^{target}(n)$.

$$\mathbf{W}^{\text{out}} = \mathbf{Y}^{\text{target}}\mathbf{X}^{\text{T}}\left(\mathbf{X}\mathbf{X}^{\text{T}} + \beta\mathbf{I}\right)^{-1}$$

Εξίσωση 10: Αναβάθμιση των βαρών εξόδου, όπου β καθορίζει το regression.
(Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks"-2012)

Regression (μεταβλητή β)

Η μεταβλητή β (Εξίσωση 10) ονομάζεται regression και χρησιμοποιείται για να αποφευχθεί η υπερεκπαίδευση στο δίκτυο. Μπορούμε να ανιχνεύσουμε αν υπάρχει υπερεκπαίδευση στο

δίκτυο όταν τα δεδομένα εκπαίδευσης μαθαίνονται σε πολύ μεγαλύτερο βαθμό από τα δεδομένα ελέγχου και όταν τα βάρη W^{out} παίρνουν πολύ μεγάλες τιμές. Το regression χρησιμοποιείται για να προσαρμόζει τα βάρη W^{out} έτσι ώστε το δίκτυο να δίνει καλά αποτελέσματα και κατά την εκπαίδευση και κατά τον έλεγχο.

Ο παραπάνω αλγόριθμος για εκπαίδευση είναι ο πιο βασικός αλγόριθμος για ένα ESN και είναι σε batch mode. Ο Batch mode αλγόριθμος είναι όταν τα βάρη αλλάζουν μετά από την καταχώρηση όλων των δεδομένων εκπαίδευσης. Μετά από δοκιμές χρησιμοποιήθηκε για την υλοποίηση του ESN, online αλγόριθμός ο οποίος έδινε καλύτερα αποτελέσματα. Με βάση τον online αλγόριθμο τα βάρη του δικτύου μεταβάλλονταν σε κάθε χρονική στιγμή δηλαδή σε κάθε μια γραμμή εισόδου των δεδομένων εκπαίδευσης n. Στο Κεφάλαιο 7 περιγράφεται πως προέκυψε αυτή η αλλαγή.

6.5.3 Online Αλγόριθμος

Για τα Echo state Network ένας από τους πιο αποτελεσματικούς online αλγόριθμους είναι ο Recursive-Least-Square(RLS)[30], ο οποίος ενσωματώνει το ιστορικό του σφάλματος του δικτύου στους υπολογισμούς αναβάθμισης του δικτύου. Ο αλγόριθμος χρησιμοποιήθηκε για να προσαρμόσει τα βάρη W^{out} στον πραγματικό χρόνο του δικτύου.

Forgetting factor

Ο αλγόριθμος περιέχει την παράμετρο λ που ονομάζεται forgetting factor. Αυτή η παράμετρος καθορίζει πόσο λιγότερο σημαντικό είναι το ιστορικό του σφάλματος εκθετικά. Για παράδειγμα αν η παράμετρος forgetting factor έχει την τιμή 1 (λ=1), το ιστορικό του σφάλματος έχει την ίδια βαρύτητα με το σφάλμα του δικτύου στην παρούσα στιγμή. Ανάλογα με την τιμή του λ , αν αυτή είναι μικρότερη από 1(λ<1), το ιστορικό του σφάλματος επηρεάζει το δίκτυο χρονικά. Δηλαδή το σφάλμα την χρονική στιγμή n έχει μεγαλύτερη βαρύτητα από το σφάλμα την χρονική στιγμή n-1. Σε πολλές προσεγγίσεις η παράμετρος αυτή έχει τιμή πολύ κοντά στο 1, αλλά όχι 1. [26][27]Η συνάρτηση εύρεσης του σφάλματος στον αλγόριθμο RLS περιγράφεται από την Εξίσωση 11[28]:

$$E(k) = \sum_{i=1}^{k} \lambda^{k-i} e(k)^2$$

Εξίσωση 11: Η συνάρτηση εύρεσης του σφάλματος στον αλγόριθμο RLS, όπου e είναι η διαφορά της επιθυμητής εξόδου y^{target} με την πραγματική έξοδο του y (y^{target}-y)
(Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation")

Η συνάρτηση σφάλματος περιέχει την μεταβλητή e(k) η οποία δηλώνει την διαφορά μεταξύ της επιθυμητής εξόδου y^{target} του δικτύου και της πραγματικής εξόδου y για την χρονική στιγμή k (Εξίσωση 12).

$$e(k) = y_{target}(k) - y(k)$$

Εξίσωση 12:Το σφάλμα τη χρονικής στιγμή k, όπου y^{target} επιθυμητή έξοδος και y πραγματική έξοδος. (Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation")

Οι συναρτήσεις που αλλάζουν σε σχέση με τον χρόνο για τον αλγόριθμο RLS είναι οι ακόλουθες:

$$\mathbf{w}_{out}(k+1) = \mathbf{w}_{out}(k) + e(k)\mathbf{g}(k)$$

Εξίσωση 13:Συνάρτηση μεταβολής των βαρών σε κάθε χρονική στιγμή κ, όπου e(k) καθορίζεται από την Εξίσωση 12 και g(k) καθορίζεται από την Εξίσωση 14. (Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation")

$$\mathbf{g}(k) = \frac{\mathbf{P}(k-1)\mathbf{x}(k)}{\lambda + \mathbf{x}(k)^T\mathbf{P}(k-1)\mathbf{x}(k)}$$

Εξίσωση 14: Συνάρτηση που καθορίζει την σημαντικότητα του ιστορικού του σφάλματος στην εναλλαγή των βαρών, x νευρώνες στο DR, όπου P(k-1) καθορίζεται από Εξίσωση 15. (Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation")

$$\mathbf{P}(k) = \lambda^{-1}\mathbf{P}(k-1) - \mathbf{g}(k)\mathbf{x}^T(k)\lambda^{-1}\mathbf{P}(k-1)$$

Εξίσωση 15:Αναδρομική συνάρτηση που επιτρέπει στο ιστορικό του σφάλματος να λαμβάνεται υπόψη κατά την αλλαγή των βαρών W^{out}, λ είναι ο forgetting factor, x νευρώνες στο DR. (Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation")

6.6 Επιλογή Echo Sate Δικτύων

Όπως αναφέρθηκε και πιο πάνω το Echo State δίκτυο ανταποκρίνεται στις απαιτήσεις του προβλήματος της πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών που αφορούν τους πιο κάτω παράγοντες:

1. Θα πρέπει να επιλύνει προβλήματα πρόβλεψης και κατηγοριοποίησης: Λόγω της αρχιτεκτονικής του δικτύου δηλαδή λόγω του reservoir, το Echo State δίκτυο έχει την δυνατότητα να διατηρεί να χαρακτηριστικά (Features) των δεδομένων εισόδου ως ηχώ στο δίκτυο. Δεδομένου μιας άγνωστης προς το δίκτυο εισόδου έχει την δυνατότητα λοιπόν να απομονώσει τα χαρακτηριστικά (feature extraction) της και με βάση την υποθηκευμένη πληροφορία του δικτύου να προβλέψει την κατηγορία αυτής της γραμμής εισόδου.

2. Θα πρέπει να γίνεται διατήρηση ιστορικού: Λόγων της αλληλεξάρτησης των αμινοξέων και της κατανομής τους στον χώρο θα πρέπει το δίκτυο να μπορεί να διατηρήσει το ιστορικό των δεδομένων εισόδου και πιο συγκεκριμένα τα χαρακτηριστικά τους ούτως ώστε να λαμβάνεται υπόψη η κατηγοριοποίηση γειτονικών αμινοξέων για ένα συγκεκριμένο αμινοξύ. Λόγω των ιδιοτήτων του Reservoir (Υποκεφάλαιο 6.4.1 Δημιουργία Dynamic Reservoir(DR))επιτυγχάνεται ο πιο πάνω στόχος. Δηλαδή γίνεται διαχείριση των δεδομένων με χρονική διαφορά (temporal difference) όπως είναι τα αμινοξέα σε μια πρωτεϊνική ακολουθία.

Ένας ακόμα παράγοντας ο οποίος επίσης επηρέασε την απόφαση για την επιλογή του συγκεκριμένου δικτύου ως προς την λύση του PSSP προβλήματος ήταν και το υπολογιστικό κόστος την υλοποίησης και της εκπαίδευσης του Echo Sate δικτύου. Το Echo State δίκτυο ως προς την εκπαίδευσή του σημειώνει ένα αρκετά μεγάλο πλεονέκτημα από τον συνηθισμένους αλγορίθμους εκπαίδευσης των δικτύων με ανάδραση RNN.

6.6.1 Δίκτυο με ανάδραση (RNN) VS Echo state δίκτυο(ESN)

Οι αλγόριθμοι που χρησιμοποιούνται στην εκπαίδευση των δικτύων με ανάδραση αντιμετωπίζουν προβλήματα αργής σύγκλισης δηλαδή απαιτείται τεράστιος χρόνος και υπολογιστικό κόστος για την εκπαίδευσή τους [39]. Συνήθης αλγόριθμος που

χρησιμοποιείται για την εκπαίδευση των δικτύων με ανάδραση είναι ο αλγόριθμος ανάστροφης μετάδοσης λάθους μέσα στο χρόνο (BPTT- Back Propagation Through Time).

Τα Echo state δίκτυο ωστόσο χρησιμοποιεί ένα αρκετά απλό αλγόριθμο μάθησης με βάση τον οποίο τα μόνα βάρη που εκπαιδεύονται είναι αυτά που ενώνουν τους νευρώνες του reservoir και του επιπέδου εξόδου, ενώ στα υπόλοιπα βάρη ανατίθενται τυχαίες τιμές. Η λειτουργικότητα του αλγορίθμου εκπαίδευσης έγκειται στην αρχιτεκτονική του δικτύου και πιο συγκεκριμένα στο reservoir το οποίο του επιτρέπει να συλλαμβάνει σύνθετα μοτίβα με την πάροδο του χρόνου και να διαφυλάξει χαρακτηριστικά από δυναμικά δεδομένα. Για αυτό το λόγω το υπολογιστικό κόστος που χρειάζεται για την εκπαίδευση του Echo State δικτύου σε σύγκριση με τα δίκτυα με ανάδραση είναι ελάχιστος, όπως και ο χρόνος εκπαίδευσης. Αυτό δίνει και την δυνατότητα εύρεσης των ιδανικών παραμέτρων για το δίκτυο εφόσον επιτρέπει την εκτέλεση πολλών πειραμάτων. Στο άρθρο του ο Herbert Jaeger [21] [22] σημειώνει ότι παρόλο που το ESN είναι εύκολο να εκπαιδευτούν και να κατανοηθούν εντούτοις χρειάζεται αρκετός χρόνος για την εύρεση των ιδανικών παραμέτρων για το reservoir οι οποίες θα προσδίδουν την echo state ιδιότητα στο δίκτυο.

6.6.2 Χρήση ESN σε άλλα προβλήματα

Το Echo State δίκτυο έχουν χρησιμοποιηθεί και στις πιο κάτω εφαρμογές οι οποίες επίσης περιλαμβάνουν δεδομένα εισόδου τα οποία κατά κάποιο τρόπο αλληλεξαρτιόνται μεταξύ τους:

- Αναγνώριση ομιλίας (speech recognition) [40]
- Αναγνώριση κίνησης (Identification of motion)[41]
- Πρόβλεψη χαοτικών χρονοσειρών (Chaotic Time Series Prediction)[42]
- Πρόβλεψη της εισροής νερού(Water Inflow Forecasting)[43]

Κεφάλαιο 7: Υλοποίηση

Για τον σχεδιασμό και την Υλοποίηση του δικτύου Echo State χρησιμοποίησα τις προδιαγραφές που αναφέρονται στο κεφάλαιο 6. Δημιούργησα κλάσεις στην γλώσσα προγραμματισμού python για την κατανομή των διαφόρων διεργασιών που επιτελούνται σε ένα δίκτυο Echo State.

7.1 Επιλογή γλώσσας προγραμματισμού

Επιλέχτηκε η γλώσσα προγραμματισμού (python) λόγω των build-in βιβλιοθηκών και συναρτήσεων που παρέχει η γλώσσα αυτή σε σχέση με τα Τεχνητά Νευρωνικά Δίκτυα. Πιο συγκεκριμένα βιβλιοθήκες που χρησιμοποιήθηκαν για την δημιουργία του Echo State δικτύου είναι οι ακόλουθες:

Numpy: Η βιβλιοθήκη Numpy χρησιμοποιείται για την διαχείριση πινάκων (array, matrix). Παράδειγμα συνάρτησης που χρησιμοποίησα είναι η dot(a,b) η οποία δέχεται δύο πίνακες τύπου (ndarray) και επιστρέφει το εσωτερικό τους γινόμενο.

Matplotlib: Η βιβλιοθήκη Matplotlib χρησιμοποιήθηκε για την αναπαράσταση των αποτελεσμάτων. Πιο συγκεκριμένα με βάση τις συναρτήσεις αυτής της βιβλιοθήκης μπορείς να δημιουργήσεις γραφικές παραστάσεις και χρησιμοποιήθηκε στο δίκτυο για την αναπαράσταση του σφάλματος και της ακρίβειας των αποτελεσμάτων. Παραδείγματα συναρτήσεων είναι: Plot(), ylabel(), xlabel(), show()

Math: Η βιβλιοθήκη math χρησιμοποιήθηκε για την αυτοματοποίηση κάποιων βασικών μαθηματικών εξισώσεων όπως για παράδειγμα το άθροισμα αριθμών ή πινάκων με την συνάρτηση sum() ή η εύρεση της θέσης του μεγαλύτερου στοιχείου σε ένα πίνακα με την συνάρτηση argmax().

7.2 Κλάσεις υλοποίησης του Δικτύου

7.2.1 Κλάση StoreProtein.py - Προεπεξεργασία δεδομένων εισόδου

Ο κώδικας της κλάσης αυτής βρίσκεται στο Παράρτημα 1

Αρχικά δημιουργήθηκε η κλάση storeProteins.py με βάση την οποία έγινε η προεπεξεργασία των δεδομένων εισόδου και η έχει αναλυθεί στο Υποκεφάλαιο 5.4.1 Κλάση Store Protein.py

7.2.2 Κλάση Main.py – Παράμετροι

Ο κώδικας της κλάσης αυτής βρίσκεται στο Παράρτημα 2

Η κλάση Main.py ορίζει τις παραμέτρους του δικτύου με βάση τις οποίες πραγματοποιείται την διαδικασία εκπαίδευσης και η διαδικασία ελέγχου του δικτύου. Στα προηγούμενα κεφάλαια αναφέρθηκε σε πολλά σημεία ότι για την επιλογή κάποιων παραμέτρων χρησιμοποιήθηκε η πειραματική μέθοδος για τον καθορισμό της ιδανικής τους της τιμής. Στο άρθρο “A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach ” του H. Jaeger [21] αναφέρεται ότι η δυσκολία που συνήθως εμφανίζεται κατά τη υλοποίηση του δικτύου, δεν προκύπτει ούτε από την αρχιτεκτονική, ούτε κατά την εκπαίδευση αλλά στον καθορισμό των παραμέτρων. Με βάση τις παραμέτρους θα καθοριστεί κατά πόσο θα εμφανιστεί στο δίκτυο η echo state ιδιότητα που επιθυμούμε για την εκπαίδευση του δικτύου.

7.2.2.1 Παράμετροι:

Οι παράμετροι του Echo state δικτύου που δημιουργήθηκε στα πλαίσια της παρούσας διπλωματικής είναι:

maxIterations:

Καθορίζει τον μέγιστο αριθμό επαναλήψεων που θα χρειαστεί το δίκτυο μέχρι να εκπαιδευτεί, δηλαδή πόσες φορές δα διατρέξουμε τα δεδομένα εισόδου από το δίκτυο.

initLen - Initial Length:

Καθορίζει το αρχικό αριθμό εισόδων του δικτύου τις οποίες θελημένα θα αγνοήσουμε ούτως ώστε να αποφύγουμε το θόρυβο που προκύπτει στα αρχικά στάδια της εκπαίδευσης (6.5.2 Συναρτήσεις Αναβάθμισης - Initial Transient (Αρχική Πάροδος)).

inSize – Input Size:

Καθορίζει τον αριθμό των νευρώνων στο επίπεδο εισόδου. Για το PSSP πρόβλημα οι νευρώνες στο επίπεδο εισόδου είναι 20 και ο αριθμός αυτός παραμένει σταθερός (Κεφάλαιο 5: Δεδομένα Εισόδου-Εξόδου).

outsize - Size of Output:

Καθορίζει τον αριθμό των νευρώνων στο επίπεδο εξόδου. Για το PSSP πρόβλημα οι νευρώνες στο επίπεδο εξόδου είναι 3 και ο αριθμός τους παραμένει σταθερός (Κεφάλαιο 5: Δεδομένα Εισόδου-Εξόδου).

resize – Reservoir Size:

Καθορίζει τους Νευρώνες στο DR, μέσω του αριθμού των δεδομένων που επιθυμούμε να μάθει το δίκτυό μας και μετά από πειράματα καθορίστηκε ο αριθμός τους (Υποκεφάλαιο 6.4.1 Δημιουργία Dynamic Reservoir (DR) - Το μέγεθος του reservoir).

a – Leaking Rate:

Η παράμετρος αυτή καθορίστηκε επίσης μέσω της πειραματικής μεθόδου και είναι το leaking Rate του DR (Υποκεφάλαιο 6.4.1 Δημιουργία Dynamic Reservoir(DR) - Το leaking rate α).

window - Window Size:

Για να αυξήσω την έννοια της αλληλεξάρτησης των δεδομένων εισόδου στο δίκτυο χρησιμοποιήθηκε ένα χρονικό κινητό παράθυρο το οποίο ορίζει το χρονικό διάστημα με βάση το οποίο θα μεταβάλλονται τα βάρη του δικτύου. Το χρονικό κινητό παράθυρο είναι μια τεχνική που χρησιμοποιείται για την πρόβλεψη χρονοσειρών και συνηθίζεται σε αρχιτεκτονικές όπως νευρωνικά δίκτυα με ανάδραση RNN. Με βάση το χρονικό κινητό παράθυρο τα δεδομένα εισόδου συσχετίζονται με άλλα δεδομένα εισόδου (δηλαδή αμινοξέα με αμινοξέα) τα οποία βρίσκονται μεταξύ του χρονικού περιθωρίου που ορίζει η παράμετρος window.

sparsity – Sparsity of Reservoir:

Αυτή είναι μια παράμετρος του reservoir η οποία καθορίζει πόσο αραιό θα είναι. Πιο συγκεκριμένα καθορίζει τον αριθμό των συναπτικών ενώσεων που θα υπάρχουν στο DR (Υποκεφάλαιο 6.4.1 Δημιουργία Dynamic Reservoir(DR) - Sparsity του reservoir).

spectral_radius – Spectral Radius:

Είναι μια από τις πιο βασικές παραμέτρους του reservoir με βάση την οποία θέτεται μια μέγιστη τιμή για τις μη μηδενικές του τιμές, δηλαδή τα βάρη που ενώνουν τους νευρώνες στο reservoir μεταξύ τους (Υποκεφάλαιο 6.4.1 Δημιουργία Dynamic Reservoir(DR) – Sectral radius των βαρών του DR).

forgettingFactor – Forgetting Factor:

Αυτή η παράμετρος χρησιμοποιείται στον online αλγόριθμο εκπαίδευσης RLS και καθορίζει πόσο λιγότερο σημαντικό είναι το ιστορικό του σφάλματος εκθετικά (6.5.3 Online Αλγόριθμος –forgetting factor).

regression - Regression :

Η μεταβλητή αυτή χρησιμοποιείται για να επιτευχθεί η υπερεκπαίδευσης στο δίκτυο και εφαρμόζεται στην εξίσωση αναβάθμισης των Βαρών (Εξίσωση 10)(6.5.2 Συναρτήσεις Αναβάθμισης – Regression).

num_folds – Number of Fords of Data:

Καθορίζει σε πόσα κομμάτια θα χωριστούν τα δεδομένα εισόδου κατά την διαδικασία cross Validation (Υποκεφάλαιο 5.5 Cross Validation).

7.2.3 Κλάση ESN.py

Ο κώδικας βρίσκεται στο Παράρτημα 3

Αυτή είναι η κύρια κλάση του προγράμματός μου η οποία δημιουργεί το δίκτυο, το εκπαιδεύει και το ελέγχει. Αυτή η κλάση καλείται από την Main.php και συγκεκριμένα καλείτε η συνάρτηση run() από αυτή την κλάση.

Η πρώτη διαδικασία που εκτελείται είναι η αρχικοποίηση των βασικών παραμέτρων . Επίσης αρχικοποιεί τις και τις πιο κάτω μεταβλητές

Αρχικοποίηση των βαρών W^{in}, W:

Τα βάρη εισόδου και τα βάρη στο Reservoir του δικτύου αρχικοποιούνται τυχαία και κανονικοποιημένα μεταξύ του συνόλου τιμών [-1,1] (Γραμμή 1,2). Ακολούθως αρχικοποιείται η παράμετρος regression η οποία σχολιάστηκε πιο πάνω (6.5.2 Συναρτήσεις Αναβάθμισης – Regression) και διαιρείται με τα βάρη στο reservoir (Γραμμή 3,4). Με βάση το sparsity καθορίζονται και οι μη μηδενικές συναπτικές ενώσεις στο reservoir (Γραμμή 5).

```
1. self.W_in = (random.rand (self.resSize, 1+self.inSize)-0.5) * 1
2. self.W = random.rand(self.resSize, self.resSize)-0.5
3. rhoW = max(abs(linalg.eig(self.W)[0]))
4. self.W *= self.spectral_radius / rhoW
5. self.W[random.rand(*self.W.shape) < self.sparsity] = 0
```

Αρχικοποίηση των νευρώνων του δικτύου:

Αρχικά δημιουργούνται οι πίνακες οι οποίοι αποθηκεύουν τις τιμές των νευρώνων του reservoir για την εκπαίδευση (Γραμμή 6) και τον έλεγχο (Γραμμή 7).

```
6. self.x = zeros((self.resSize, 1))
7. self.xTest = zeros((self.resSize, 1))
8. self.Yt = self.trainData.yt[self.initLen:]
9. self.X = zeros((self.resSize,self.trainLen- self.initLen))
10. self.Y = zeros((self.outSize,self.trainLen-self.initLen))
```

Αρχικοποίηση των μεταβλητών Σφάλματος και Ακρίβειας:

Αρχικοποιούνται οι μεταβλητές σφάλματος και ακρίβειας και για την εκπαίδευση αλλά και για τον έλεγχο του δικτύου. Στις γραμμές 15 και 16 αρχικοποιούνται οι πίνακες που θα αποθηκεύουν το σφάλμα για τις γραφικές παραστάσεις. Στις γραμμές 17 και 18 αρχικοποιούνται οι πίνακες για την δημιουργία γραφικών παραστάσεων για την ακρίβεια του αποτελέσματος του δικτύου.

```
11. self.TrainError=0
12. self.TestError=0
13. self.TrainAccuracy=0
14. self.TestAccuracy = 0
15. self.TestErrorplot=[]
16. self.TrainErrorplot=[]
17. self.TrainAccuracyplot=[]
18. self.TestAccuracyplot=[]
```

Το επόμενο βήμα στον αλγόριθμο είναι η διαδικασία του Cross Validation (Υποκεφάλαιο 5.5 Cross Validation) με βάση την οποία χωρίζουμε τα δεδομένα μας σε αριθμό κομματιών ανάλογο με την τιμή της παραμέτρου num_folds. Οι επαναλήψεις του δικτύου είναι επίσης όσο και η num_folds παράμετρος. Στις γραμμές 19 και 20 φαίνεται ο διαχωρισμός των κομματιών των δεδομένων εισόδου. Στις γραμμές 21 και 22 γίνεται κλήση συναρτήσεων της κλάσης StoreProtein.py για να διαβαστούν τα δεδομένα από το σύνολο ελέγχου και από το σύνολο εκπαίδευσης.

```
19. testing_this_round = lines[j*subset_size*3:][:subset_size*3]
20. training_this_round = lines[:j*subset_size*3] + lines[(j+1)*subset_size*3:]
21. pTrain.readProteins(training_this_round)
22. pTest.readProteins(testing_this_round)
```

Ακολούθως για κάθε μια επανάληψη από το num_folds τρέχουμε για όσες επαναλήψεις καθορίζει το maxIteration τις συναρτήσεις training() και testing().

7.3.2.1 Συνάρτηση training ()

Ολοκληρωμένος ο κώδικας βρίσκεται στο Παράρτημα 3.

Με βάση αυτή την συνάρτηση εκπαιδεύεται το δίκτυο. Για κάθε ένα από τα δεδομένα εισόδου(Γραμμή 23) και με βάση το μέγεθος του παραθύρου καλούνται οι εξισώσεις αναβάθμισης του δικτύου. Επίσης σε αυτή την συνάρτηση υλοποιείται και η τεχνική Initial Transient με βάση την οποία οι νευρώνες στο reservoir αρχίζουν να μεταβάλλονται μετά από κάποιο αριθμό επαναλήψεων ο οποίος καθορίζεται από την παράμετρο initLen. Μετά από αυτό το στάδιο αναβαθμίζονται οι νευρώνες στο reservoir (Γραμμή 24,25) και στο επίπεδο εξόδου (Γραμμή 26). Με βάση την τιμή στο επίπεδο εξόδου μπορούμε να υπολογίσουμε το προσωρινό σφάλμα(το σφάλμα για την χρονική στιγμή n) (Γραμμή 33) και να μεταβάλουμε τα βάρη. Επίσης σε αυτό το σημείο υπολογίζεται η ακρίβεια (accuracy) (Γραμμή 32) των αποτελεσμάτων. Όπως έχει προαναφερθεί το αποτέλεσμα του δικτύου είναι ένα διάνυσμα τριών τιμών όσοι και οι νευρώνες στο κρυφό επίπεδο. Για να υπολογίσουμε αν μια έξοδος είναι η σωστή θα πρέπει να μετατρέψουμε την μεγαλύτερη από τις τρεις τιμές σε 1 και τις υπόλοιπες σε 0 (Γραμμή 27-31). Με αυτό τον τρόπο δηλώνεται ότι η έξοδος με την μεγαλύτερη τιμή αντιπροσωπεύει τον τύπο δευτεροταγούς δομής ο οποίος ταυτίζεται με τον κάθε νευρώνα εξόδου. Γίνεται ο υπολογισμός των βαρών εξόδου w^{out} (Γραμμή 37) του δικτύου και η αναβάθμισή τους (Γραμμή 38). Για την αναβάθμιση των βαρών w^{out} λόγω του ότι χρησιμοποιείται online αλγόριθμος ορίζουμε τις μεταβλητές g (Γραμμή 34) και p (Γραμμή 35).

```
23. u=zip(*[iter(self.trainData.data[(w+iterW)*self.inSize:(w+iterW)*self.inSize+self.in
    Size])]*1)
24. self.x = array((1-self.a)*self.x + self.a*tanh( dot( self.W_in, vstack((u,[1])) ) +
    dot(self.W, self.x ) ))                                              (Εξίσωση 7)
25. self.X[:,iteration-self.initLen] = self.x[:,0]                       (Εξίσωση 8)
26. y =tanh(dot( self.W_out.T,self.x ))                                  (Εξίσωση 9)
27. y = reshape(y,(1,self.outSize))[0]
28. maxPosition=argmax(y)
29. yout=zeros(3)
30. yout[maxPosition]=1
31. self.Y[:,iteration-self.initLen]=yout
```

```
32. self.TrainAccuracy=self.TrainAccuracy+1-sum(abs(self.Yt[iteration-self.initLen],
    self.Y[:,iteration-self.initLen]))/2
33. error = self.Yt[iteration-self.initLen]-y                          (Εξίσωση 12)
34. g=dot(self.p ,self.x)/dieretisg                                   (Εξίσωση 14)
35. self.p=pow(self.forgettingFactor,-1)*self.p - g*array(self.x).T
    *pow(self.forgettingFactor,-1)*self.p                             (Εξίσωση 15)
36. self.TrainError=self.TrainError+sum(pow(er,2))/self.outSize
37. self.W_out = (dot( dot(array(self.Yt).T,self.X.T), linalg.inv( dot(self.X,self.X.T) + \
                        self.regression*eye(self.resSize) ) )).T      (Εξίσωση 10)
38. self.W_out=self.W_out+(sum(error)/self.outSize*g)*10              (Εξίσωση 13)
```

7.3.2.2 Συνάρτηση testing ()

Ο κώδικας βρίσκεται στο Παράρτημα 3.

Η συνάρτηση αυτή κάνει σχεδόν τις ίδιες διαδικασίες όπως και η συνάρτηση testing με την διαφορά ότι τα δεδομένα εισόδου είναι τα δεδομένα ελέγχου και δεν γίνεται αναβάθμιση των βαρών W^{out} στο στάδιο αυτό.

Κεφάλαιο 8: Περιγραφή Αποτελεσμάτων και Μελλοντικές Έρευνες

8.1. Μετρικές αποτελεσμάτων

Η μετρική που χρησιμοποίησα για την εξαγωγή των αποτελεσμάτων είναι η ακρίβεια του αποτελέσματος αμινοξύ προς αμινοξύ.

8.1.1 Ακρίβεια Αποτελέσματος

Για τον καθορισμό της ακρίβειας του αποτελέσματος χρησιμοποίησα την Εξίσωση 1 από το Κεφάλαιο 3 η οποία ανάλογα με το αποτέλεσμα του δικτύου σε κάθε επανάληψη n καθορίζει με βάση το επιθυμητό αποτέλεσμα αν αυτή η έξοδος ήταν σωστή ή λάθος (Κεφάλαιο 7-Συνάρτηση training() Γραμμή 32). Όταν με βάση την εξίσωση 9 βρεθεί το αποτέλεσμα του δικτύου y(n), για να αποφασιστεί αν το αποτέλεσμα αυτό είναι σωστό, δηλαδή αν ταυτίζεται με το επιθυμητό αποτέλεσμα, θέτω την μεγαλύτερη τιμή του διανύσματος σε 1 και τις άλλες δύο σε 0 (Κεφάλαιο 7-Συνάρτηση training() Γραμμή 28-31).

Να σημειωθεί ότι λόγω του Cross Validation τα αποτελέσματα είναι διανύσματα των 10 τιμών(διαστάσεων) και για τον καθορισμό του της ακρίβειας των αποτελεσμάτων υπολογίζεται ο μέσος όρος αυτών των τιμών.

8.2 Προσεγγίσεις λύσης του PSSP προβλήματος

8.2.1 Προσέγγιση 1

Σε αρχική φάση , χρησιμοποίησα για την υλοποίηση του Echo State δικτύου ένα batch mode αλγόριθμο μάθησης. Ο αλγόριθμος αυτός ακολουθούσε την πιο κάτω διαδικασία για την αναβάθμιση των βαρών και την εκπαίδευση του δικτύου:

Για κάθε ένα από τα δεδομένα εισόδου και εφόσον έχουν περάσει οι αρχικές επαναλήψεις που καθορίζονται από το Initial Transient, αναβαθμίζονται οι νευρώνες στο DR(Εξίσωση 7 και 8) . Διατρέχοντας όλα τα δεδομένα εισόδου από το δίκτυο θεωρούμε ότι πετύχαμε ένα καλό αποτέλεσμα στις τιμές εξόδου του DR και έτσι θέτουμε τιμές σε όλα τα εξωτερικά βάρη (Εξίσωση 10) με βάση τις τελευταίες τιμές που πήραν οι νευρώνες στο DR. Δεν γίνεται αναβάθμιση των βαρών σε αυτή την προσέγγιση. Ακολούθως περνούν τα δεδομένα ελέγχου από το δίκτυο και για κάθε μια είσοδο υπολογίζονται οι τιμές στους νευρώνες του επιπέδου εξόδου (Εξίσωση 9). Αυτή η έξοδος του δικτύου ανατροφοδοτείται πίσω στο δίκτυο και η

διαδικασία αυτή επαναλαμβάνεται. Στο τέλος υπολογίζεται το σφάλμα με βάση την Εξίσωση 17 η οποία για ορίσματα δέχεται τους πίνακες y^{target} και y.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Εξίσωση 17: Mean Square Error. Υπολογισμός σφάλματος.

Αυτός ο αλγόριθμος είναι η πιο απλοϊκή προσέγγιση για την δημιουργία ενός Echo State δικτύου αλλά δεν μπορεί να εφαρμοστεί για το πρόβλημα της πρόβλεψης της δευτεροταγής δομής των πρωτεϊνών για τους πιο κάτω λόγους:

- Μπορεί να δώσει καλά αποτελέσματα σε προβλήματα στα οποία η είσοδος κατά την χρονική στιγμή n είναι το επιθυμητό αποτέλεσμα για την είσοδο στην χρονική στιγμή n-1. Για αυτό το λόγο γίνεται ανατροφοδότηση του αποτελέσματος y πίσω στο δίκτυο. Στο PSSP πρόβλημα τα αμινοξέα αλληλεξαρτώνται μεταξύ τους αλλά η είσοδος και η έξοδος του δικτύου περιέχουν εντελώς διαφορετικά δεδομένα.
- Λόγω της απλότητας της υλοποίησης αυτής τα δεδομένα εισόδου θα πρέπει να περιέχουν περιορισμένο όγκο και μικρή πολυπλοκότητα εκτός από τα βασικά χαρακτηριστικά που ορίζει το δίκτυο. Δηλαδή το να είναι μη διαχωρίσιμα και να απαιτούν την διατήρηση ιστορικού .
- Δεν γίνεται αναβάθμιση των βαρών αντίθετα στα βάρη ανατίθενται τιμές μια φορά οι οποίες παραμένουν αμετάβλητες. Για μεγάλο όγκο δεδομένων τα σφάλμα που προκύπτει είναι πολύ μεγάλο.
- Το πρόβλημα πρόβλεψης της δευτεροταγούς δομής των πρωτεϊνών είναι αρκετά περίπλοκο ως προς την λύση αλλά και τα δεδομένα του.

Προσπάθησα να προσαρμόσω τον batch mode αλγόριθμο ούτως ώστε να μπορεί να εφαρμοστεί στο PSSP πρόβλημα και να αποφύγω τα πιο πάνω προβλήματα.

8.2.2 Προσέγγιση 2

Διατηρώντας την ίδια αρχιτεκτονική με την πρώτη προσέγγιση προσπάθησα να αλλάξω τον αλγόριθμο ούτως ώστε να μπορεί να εφαρμοστεί στο PSSP πρόβλημα. Αρχικά διαχώρισα το κώδικα σε δύο συναρτήσεις από τις οποίες η πρώτη θα είναι η συνάρτηση εκπαίδευσης training() και η δεύτερη η συνάρτηση ελέγχου testing(). Η συνάρτηση εκπαίδευσης είναι υπεύθυνη όπως και στην προηγούμενη προσέγγιση για την αναβάθμιση των νευρώνων στο DR x(n). Επίσης στην συνάρτηση training γίνεται και η αναβάθμιση των νευρώνων στο επίπεδο εξόδου y(n). Για να μπορούν όμως να αναβαθμιστούν οι νευρώνες στο επίπεδο εξόδου y(n) θα πρέπει να έχουν οριστεί οι τιμές στα βάρη εξόδου W^{out}. Ο batch mode αλγόριθμος ωστόσο αναβαθμίζει τα βάρη μετά από το πέρασμα των δεδομένων εισόδου από το δίκτυο και μόνο μια φορά. Για να επιλυθεί αυτό το πρόβλημα και να καταφέρω να υλοποιήσω τον αλγόριθμο αρχικοποίησα τους νευρώνες στο DR με την τιμή 0. Με αυτό τον τρόπο πριν από την συνάρτηση training() ανατίθενται τιμές στα βάρη εξόδου και μετά εκπαιδεύεται το δίκτυο. Ο τρόπος αυτός είναι λίγο ανορθόδοξος γιατί ουσιαστικά θέτω τιμές στα βάρη εξόδου χωρίς πρώτα να βρω τις τιμές στους νευρώνες του DR. Η διαδικασία όμως αυτή εκτελείται τουλάχιστο δύο φορές και λόγω του ότι η συνάρτηση που δίνει τιμές στα βάρη εξόδου δεν είναι αναδρομική (Εξίσωση 10), δηλαδή τα βάρη στην στιγμή n+1 δεν εξαρτιόνται από τα βάρη την στιγμή n, την δεύτερη φορά που θα εκτελεστεί η συνάρτηση των βαρών εξόδου αυτά θα πάρουν τις κατάλληλες τιμές. Μετά από την συνάρτηση εκπαίδευσης εκτελείται η συνάρτηση ελέγχου. Η συνάρτηση αυτή διατρέχει τα δεδομένα από το σύνολο ελέγχου και ανάλογα τα εφαρμόζει στο δίκτυο για να ελέγξει τα αποτελέσματα με βάση τα βάρη που έχουν εκπαιδευτεί, δηλαδή στην δεύτερη επανάληψη του αλγορίθμου. Και στις δύο συναρτήσεις η ακρίβεια των αποτελεσμάτων.

Καλύτερο Πείραμα

maxIterations	2
initLen	770
inSize	20
outSize	3
resSize	100
a	0.6
window	X
Sparsity	1
spectral_radius	1.25
forgettingFactor	X
regression	1e-8
num_folds	10

Οι παράμετροι (Σχήμα 30) τέθηκαν στο δίκτυο με βάση τους περιορισμούς που σχολιάστηκαν στο Υποκεφάλαιο 7.2.2.1. Για αυτή την προσέγγιση τα καλύτερα αποτελέσματα λήφθηκαν με βάση τις παραμέτρους στο Σχήμα 30. Εδώ να σημειωθεί ότι για το δίκτυο δεν εφαρμόστηκε κινητό χρονικό παράθυρο (Προσέγγιση 3) και επίσης δεν υπάρχει forgetting factor εφόσον αυτή η παράμετρος προορίζεται για τον RLS αλγόριθμο.

Σχήμα 30: Παράμετροι δικτύου Προσέγγισης 1.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	66.3976522377
1	67.5477239354
2	67.282127031
3	67.8545454545
4	67.6274944568
5	67.282127031
6	67.6274944568
7	67.4349442379
8	67.6274944568
9	66.6922486569
Μέσος Όρος	67.3373851955

Πίνακας 1: Αποτελέσματα Πειράματος Training

Num_folds	Ακρίβεια - Accuracy
0	62.5698324022
1	48.8888888889
2	62.4338624339
3	58.6826347305
4	66.3157894737
5	60.1063829787
6	64.2105263158
7	61.421319797
8	58.7301587302
9	56.9037656904
Μέσος Όρος	60.0263161441

Πίνακας 2: Αποτελέσματα Πειράματος Testing

Παρόλο που τα αποτελέσματα της εκπαίδευσης του δικτύου (training) είναι αρκετά ικανοποιητικά με μέσο όρο ακρίβειας αποτελέσματος 67.33 % ,δεν μπορούμε να πούμε το ίδιο και για τα αποτελέσματα ελέγχου του δικτύου όπου η ακρίβεια των αποτελεσμάτων ανέρχεται μόλις στο 60.02%. Παρόλα αυτά η εύρεση των ιδανικών παραμέτρων για αυτή

την προσέγγιση βοήθησαν αρκετά στο καθορισμό των παραμέτρων σε επόμενες προσεγγίσεις. Επίσης ενθαρρυντικό είναι το γεγονός ότι αντιμετωπίστηκαν τα προβλήματα τα οποία εμφανίστηκαν στην Προσέγγιση 1.
Ακολούθως αναλογιζόμενη τα αποτελέσματα στην δεύτερη προσέγγιση θεώρησα πως θα ήταν καλή ιδέα η εισαγωγή κινητού χρονικού παραθύρου.

8.2.3 Προσέγγιση 3-Κινητό Χρονικό παράθυρο

Για να ενισχύσω την ιδέα της αλληλεξάρτησης των δεδομένων εισόδου, όπως αυτή προκύπτει από τις ελκτικές και αποστικές δυνάμεις μεταξύ αμινοξέων σε μια πρωτεϊνική ακολουθία, χρησιμοποίησα ένα κινητό χρονικό παράθυρο (Υποκεφάλαιο 7.2.2.1 Παράμετροι). Διατηρώντας και την αρχιτεκτονική και τον αλγόριθμο από την προηγούμενη προσέγγιση όρισα κάποιο όριο με βάση το οποίο θα εκτελούνται οι εξισώσεις αναβάθμισης. Για παράδειγμα αν το μέγεθος του παραθύρου είναι 15, αυτό σημαίνει ότι για το αμινοξύ στην θέση 0 μέχρι το αμινοξύ στην θέση 14 θεωρώ ότι υπάρχει σχέση μεταξύ τους. Για την υλοποίηση του κινητού παραθύρου εισήγαγα τα δεδομένα στο δίκτυο με αριθμό όσο και το μέγεθος του παραθύρου και παρόλο που η Εξίσωση 7 εκτελείται σε κάθε επανάληψη οι Εξισώσεις 8 και 9 εκτελούνται μόνο μετά το πέρασμα όλων των δεδομένων του παραθύρου.

Πείραμα 1

maxIterations	2
initLen	770
inSize	20
outSize	3
resSize	100
a	0.6
window	10
Sparsity	1
spectral_radius	1.25
forgettingFactor	X
regression	1e-8
num_folds	10

Σχήμα 31: Προσέγγιση 3-Παράμετροι

Όπως φαίνεται στο Σχήμα 31 οι παράμετροι του δικτύου είναι ίδιες με της προηγούμενη προσέγγισης με την διαφορά ότι χρησιμοποιήθηκε κινητό χρονικό παράθυρο μεγέθους 10. Η παράμετρος forgettingFactor δεν χρησιμοποιείται επίσης σε αυτή την προσέγγιση.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	68.8644688645
1	69.2307692308
2	67.8966789668
3	68.7272727273
4	67.8966789668
5	67.8966789668
6	67.8966789668
7	68.4014869888
8	67.8966789668
9	67.0498084291
Μέσος Όρος	68.1757201074

Πίνακας 3: Προσέγγιση 3-Πείραμα 1- Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	63.1578947368
3	64.7058823529
4	71.0526315789
5	60.5263157895
6	73.6842105263
7	60.0
8	57.8947368421
9	56.25
Μέσος Όρος	61.560500516

Πίνακας 4: Προσέγγιση 3-Πείραμα 1- Testing

Με βάση τα πιο πάνω αποτελέσματα (Πίνακας 3
και 4), συμπεραίνουμε ότι η προσθήκη του κινητού παραθύρου μεγέθους 10 αύξησε την ακρίβεια των αποτελεσμάτων. Για την ακρίβεια των αποτελεσμάτων στην εκπαίδευση υπάρχει αύξηση περίπου 0.5% ενώ για την ακρίβεια των αποτελεσμάτων στον έλεγχο δικτύου έχουμε αύξηση γύρο στο 1.5%. Με βάση αυτά τα αποτελέσματα δοκίμασα να

αυξήσω το μέγεθος του παραθύρου με σκοπό την αύξηση της ακρίβειας των αποτελεσμάτων του δικτύου.

Πείραμα 2- Καλύτερα Αποτελέσματα

window	15

Οι παράμετροι του δικτύου είναι ίδιες με της προηγούμενη προσέγγισης με την διαφορά ότι χρησιμοποιήθηκε κινητό χρονικό παράθυρο μεγέθους 15.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	69.2307692308
1	67.032967033
2	66.8508287293
3	66.847826087
4	66.8508287293
5	66.8508287293
6	66.8508287293
7	66.6666666667
8	66.8508287293
9	67.816091954
Μέσος Όρος	67.1848464618

Πίνακας 5: Προσέγγιση 3 - Πείραμα 2 - Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	57.6923076923
3	78.2608695652
4	73.0769230769
5	73.0769230769
6	69.2307692308
7	51.8518518519
8	69.2307692308
9	68.75
Μέσος Όρος	64.9503747058

Πίνακας 6: Προσέγγιση 3 - Πείραμα 2 - Testing

Στο πείραμα αυτό παρόλο που υπάρχει μείωση στην ακρίβεια του αποτελέσματος για την εκπαίδευση (Πίνακας 5) εντούτοις παρατηρούμε από τον Πίνακα 6 μια σημαντική αύξηση της ακρίβειας του αποτελέσματος για τον έλεγχο του δικτύου. Με ακρίβεια Q3 64.95% θεωρούμε ότι το δίκτυο μπορεί να προβλέψει σε καλό βαθμό την δευτεροταγή δομή των δεδομένων εισόδου. Λόγω της επιτυχίας αυτού του πειράματος συνέχισα να αυξάνω το μεγέθους του παραθύρου.

Πείραμα 3

window	20

Οι παράμετροι του δικτύου είναι ίδιες με της προηγούμενη προσέγγισης με την διαφορά ότι χρησιμοποιήθηκε κινητό χρονικό παράθυρο μεγέθους 20.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	68.6131386861
1	67.1532846715
2	64.7058823529
3	66.6666666667
4	64.7058823529
5	64.7058823529
6	64.7058823529
7	65.1851851852
8	64.7058823529
9	66.4122137405
Μέσος Όρος	65.7559900715

Πίνακας 6: Προσέγγιση 3 - Πείραμα 3 - Training

Num_folds	Ακρίβεια - Accuracy
0	72.2222222222
1	50.0
2	52.6315789474
3	70.5882352941
4	73.6842105263
5	52.6315789474
6	73.6842105263
7	55.0
8	52.6315789474
9	58.3333333333
Μέσος Όρος	61.1406948744

Πίνακας 7: Προσέγγιση 3 - Πείραμα 3 - Testing

Τα αποτελέσματα του πειράματος 3 δεν σημείωσαν τόσο καλή ακρίβεια αποτελεσμάτων όσο το προηγούμενο πείραμα οπότε το πείραμα αυτό απορρίπτεται.

Με βάση τα αποτελέσματα από αυτή την προσέγγιση φαίνεται ότι η προσθήκη του κινητού χρονικού παραθύρου βοήθησε το δίκτυο να βρει καλύτερες συσχετίσεις και χαρακτηριστικά μεταξύ των αμινοξέων και συνεπώς να αυξηθεί ο αριθμός των σωστών προβλέψεων των αμινοξέων. Τα 3 πιο πάνω πειράματα είναι αντιπροσωπευτικά εφόσον για την συγκεκριμένη προσέγγιση έχουν γίνει πολύ περισσότερα πειράματα προσαρμογής του κινητού παραθύρου. Το καλύτερο αποτέλεσμα επιτεύχθηκε με το πείραμα 2.

8.2.4 Προσέγγιση 4 -Online αλγόριθμος

Για αυτή την προσέγγιση προσπάθησα να αλλάξω τον αλγόριθμο μάθησης από batch mode σε online. Ο online αλγόριθμος βοηθά στην αλλαγή των βαρών με βάση το προσωρινό σφάλμα (Εξίσωση 12) που προκύπτει σε κάθε επανάληψη της συνάρτησης εκπαίδευσης. Για αυτό τον λόγο χρησιμοποίησα τον αλγόριθμο RLS (Υποκεφάλαιο 6.5.3 Online Αλγόριθμος) ο οποίος συνηθίζεται να χρησιμοποιείται σε αρχιτεκτονικές όπως το Echo State δίκτυο. Οι αλλαγές που προέκυψαν σε κάθε επανάληψη της συνάρτησης εκπαίδευσης ήταν η πρόσθεση της εξίσωσης αναβάθμισης των βαρών (Εξίσωση 13)και των εξισώσεων που συμβάλλουν σε αυτή την αναβάθμιση (Εξίσωση 14 και 15).

Πείραμα 1

maxIterations	10
initLen	770
inSize	20
outSize	3
resSize	100
a	0.6
window	15
Sparsity	1
spectral_radius	1.25
forgettingFactor	0.6
regression	1e-8
num_folds	10

Λόγω της αλλαγής του αλγορίθμου μάθησης οι επαναλήψεις του δικτύου έπρεπε να αυξηθούν από τις προηγούμενες προσεγγίσεις. Σε αυτή την προσέγγιση χρησιμοποιείται και η παράμετρος forgettingFactor (Υποκεφάλαιο 6.5.3 Online Αλγόριθμος- Forgetting Factor) η οποία επηρεάζει τις εξισώσεις που αναφέρονται πιο πάνω.

Σχήμα: Προσέγγιση 4-Παράμετροι

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	69.2307692308
1	67.032967033
2	66.8508287293
3	66.3043478261
4	66.8508287293
5	66.8508287293
6	66.8508287293
7	66.1111111111
8	66.8508287293
9	68.3908045977
Μέσος Όρος	68.1757201074

Πίνακας 8: Προσέγγιση 4-Πείραμα 1- Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	57.6923076923
3	78.2608695652
4	80.7692307692
5	73.0769230769
6	69.2307692308
7	51.8518518519
8	69.2307692308
9	68.75
Μέσος Όρος	65.7919605475

Πίνακας 9: Προσέγγιση 4-Πείραμα 1- Testing

Βλέπουμε ότι με αυτή την προσέγγιση επίσης αυξήθηκε ο Μέσος Όρος της ακρίβειας των αποτελεσμάτων για την εκπαίδευση κατά 1% και για τον έλεγχο περίπου 1%.

Πείραμα 2

maxIterations	100

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	69.2307692308
1	67.032967033
2	66.8508287293
3	66.3043478261
4	66.8508287293
5	66.8508287293
6	66.8508287293
7	66.1111111111
8	66.8508287293
9	68.3908045977
Μέσος Όρος	68.1757201074

Πίνακας 10: Προσέγγιση 3-Πείραμα 2- Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	57.6923076923
3	78.2608695652
4	80.7692307692
5	73.0769230769
6	69.2307692308
7	51.8518518519
8	69.2307692308
9	68.75
Μέσος Όρος	65.7919605475

Πίνακας 11: Προσέγγιση 4-Πείραμα 2- Testing

Με την αύξηση των επαναλήψεων πρόσεξα ότι από κάποιο στάδιο και μετά τα βάρη δεν αλλάζουν η αν αλλάζουν τότε η διαφορά είναι ελάχιστη. Έτσι το αποτέλεσμα δεν επηρεάζεται ούτε αρνητικά ούτε θετικά. Τα αποτελέσματα όπως φαίνονται στους πίνακες 8 και 9 και στους πίνακες 10 και 11 είναι τα ίδια.

8.2.5 Προσέγγιση 5-Προσθήκη παραμέτρου ως learning Rate

Παρόλο που προστέθηκε ο online αλγόριθμος πρόσεξα ότι οι μεταβολές των βαρών που προκύπταν από τις Εξισώσεις 14 και 15 δεν ήταν αρκετές. Δεδομένου αυτών των μετρικών πρόσθεσα μια μεταβλητή η οποία να ελέγχει την τιμή της μεταβολής των βαρών. Αυτή η μεταβλητή ονομάστηκε learning Rate και για τον καθορισμό της ιδανικής της τιμής χρησιμοποιήθηκε η πειραματική μέθοδος.

Πείραμα 1

maxIterations	10
initLen	770
inSize	20
outSize	3
resSize	100
a	0.6
window	15
Sparsity	1
spectral_radius	1.25
forgettingFactor	0.6
regression	1e-8
num_folds	10
LearningRate	5

Σχήμα: Προσέγγιση 5-Παράμετροι

Σε αυτή την προσέγγιση προστέθηκε και η παράμετρος learningRate στο πείραμα αυτό πήρε την τιμή 5.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	69.2307692308
1	67.032967033
2	66.8508287293
3	66.847826087
4	66.8508287293
5	66.8508287293
6	66.8508287293
7	66.6666666667
8	66.8508287293
9	67.816091954
Μέσος Όρος	67.1848464618

Πίνακας 12: Προσέγγιση 5-Πείραμα 1- Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	57.6923076923
3	78.2608695652
4	73.0769230769
5	73.0769230769
6	69.2307692308
7	51.8518518519
8	69.2307692308
9	68.75
Μέσος Όρος	64.9503747058

Πίνακας 13: Προσέγγιση 5-Πείραμα 1- Testing

Προσθέτοντας την παράμετρο learningRate=5 παρατηρούμε ότι τα αποτελέσματα και για την εκπαίδευση και για τον έλεγχο του δικτύου κυμαίνονται σε πιο χαμηλά επίπεδα από την προηγούμενη προσέγγιση στην οποία δεν υπήρχε αυτή η παράμετρος.

Πείραμα 2

LearningRate	10

Σε αυτό το πείραμα διπλασιάζω το learningRate.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	69.2307692308
1	67.032967033
2	66.8508287293
3	66.847826087
4	66.8508287293
5	66.8508287293
6	66.8508287293
7	66.6666666667
8	66.8508287293
9	67.816091954
Μέσος Όρος	67.1848464618

Πίνακας 14: Προσέγγιση 5-Πείραμα 2- Training

Num_folds	Ακρίβεια - Accuracy
0	66.6666666667
1	41.6666666667
2	57.6923076923
3	78.2608695652
4	80.7692307692
5	73.0769230769
6	76.92307692
7	51.8518518519
8	69.2307692308
9	68.75
Μέσος Όρος	66.4888362443

Πίνακας 15: Προσέγγιση 5-Πείραμα 2- Testing

8.2.6 Προσέγγιση 5-Sparse

Σε πολλά άρθρα αναφέρεται ότι το reservoir που χρησιμοποιείται είναι αρκετά αραιό. Αρχικά από προηγούμενα πειράματα σε προηγούμενες προσεγγίσεις έθεσα την παράμετρο sparse=1 επειδή σε αντίθετη περίπτωση τα αποτελέσματα ήταν σχεδόν τυχαία γύρω στο 37%. Δοκίμασα ωστόσο να αλλάξω την παράμετρο αυτή και πρόσεξα ότι τα αποτελέσματα δεν ήταν απογοητευτικά. Μετά από κάποια πειράματα κατέληξα στο πιο κάτω.

Πείραμα 1-Καλύτερο Πείραμα

maxIterations	10
initLen	770
inSize	20
outSize	3
resize *	50
a *	0.3
window	15
Sparsity *	0.2
spectral_radius *	1.20
forgettingFactor	0.6
regression	1e-8
num_folds	10
LearningRate	10

Σχήμα: Προσέγγιση 6-Παράμετροι

Σε αυτή την περίπτωση άλλαξα τις παραμέτρους οι οποίες διακρίνονται με *.

Αποτελέσματα

Num_folds	Ακρίβεια - Accuracy
0	67.5824175824
1	64.2857142857
2	63.5359116022
3	64.1304347826
4	63.5359116022
5	63.5359116022
6	63.5359116022
7	63.3333333333
8	63.5359116022
9	66.091954023
Μέσος Όρος	64.3103412018

Πίνακας 17: Προσέγγιση 6-Πείραμα 1- Train

Num_folds	Ακρίβεια - Accuracy
0	62.5
1	58.3333333333
2	61.5384615385
3	69.5652173913
4	88.4615384615
5	57.6923076923
6	73.0769230769
7	62.962962963
8	65.3846153846
9	81.25
Μέσος Όρος	68.0765359841

Πίνακας 18: Προσέγγιση 6-Πείραμα 1- Testing

Όπως φαίνεται από τα αποτελέσματα κατά τον έλεγχο του δικτύου η ακρίβεια των αποτελεσμάτων είναι γύρω στο 68% Q3 αλλά κατά την εκπαίδευση του δικτύου η ακρίβεια του αποτελέσματος είναι πιο χαμηλή, γύρω στο 64% Q3. Το φαινόμενο αυτό δεν συνηθίζεται να έχουμε πιο χαμηλό ποσοστό ακρίβειας κατά την εκπαίδευση παρά κατά τον έλεγχο του δικτύου και μπορεί να έγκειται στο γεγονός ότι στα σύνολα ελέγχου όπου η ακρίβεια αποτελεσμάτων ξεπέρασε το σύνολο εκπαίδευσης, οι πρωτεΐνες που βρίσκονται σε αυτά έχουν μεγάλη ομοιότητα με αυτές στο σύνολο εκπαίδευση. Έτσι το δίκτυο ακολουθώντας παρόμοια τροχιά για το δεδομένο εισόδου με αυτές που έχει εκπαιδευτεί μπορεί να κατηγοριοποιείσαι τα αμινοξέα της πρωτεΐνης με μεγάλη ακρίβεια.

8.3 Συμπεράσματα

Με βάση τις πιο πάνω προσεγγίσεις έλαβα τα ακόλουθα αποτελέσματα με βάση την διαδικασία ελέγχου του δικτύου:

	Μέσος Όρος
Προσέγγιση 2	60.0263161441
Προσέγγιση 3	64.9503747058
Προσέγγιση 4	65.7919605475
Προσέγγιση 5	66.4888362443
Προσέγγιση 6	68.0765359841

Η καλύτερη προσέγγιση είναι η τελευταία στην οποία σημειώθηκε ακρίβεια αποτελέσματος γύρω στο 68% Q3. Τα αποτελέσματα αυτά είναι αρκετά ικανοποιητικά δεδομένου ότι στην εφαρμογή του PSSP προβλήματος σε άλλα δίκτυα το αποτέλεσμα είναι γύρω στο 76% με βάση το Q3, γίνεται όμως η χρήση μεθόδων Filtering και Ensembles.[19][20].

8.4 Μελλοντικές Έρευνες

Έρευνα ως προς το πώς προκύπτει η ακρίβεια των αποτελεσμάτων κατά τον έλεγχο του δικτύου να κυμαίνεται σε μεγαλύτερα ποσοστά από την ακρίβεια των αποτελεσμάτων κατά την εκπαίδευση. Υλοποίηση μετρικής Segment OVerlap - SOV και confusion matrix για καλύτερη κατανόηση των αποτελεσμάτων.

Εκτέλεση περισσότερων πειραμάτων στα οποία θα γίνει αλλαγή της συνάρτησης αναβάθμισης και του πως τα δεδομένα αναπαρίστανται με βάση το κινητό παράθυρο.

Μέχρι στιγμής παρόλο που χρησιμοποιούσαμε δεδομένα πρωτεϊνών το δίκτυο ανταποκρινόταν σε αυτά ως πίνακες με αριθμητικά δεδομένα χωρίς να γνωρίζει την σημασία τους. Για την βελτιστοποίησή των αποτελεσμάτων σε δίκτυα που ασχολούνται με το PSSP πρόβλημα τους συνηθίζεται να χρησιμοποιούνται μεθόδοι όπως το Filtering και Ensembles με βάσει τις οποίες εισάγουμε πληροφορίες στο δίκτυο σχετικές με τις πρωτεΐνες.

Επίσης αναλογιζόμενη την αύξηση της ακρίβειας των αποτελεσμάτων του δικτύου με την προσθήκη ενός κινητού χρονικού παραθύρου , ενθαρρύνεται η δημιουργία δικτύου αμφίδρομης ανάδρασης BRRN (Bidirectional Recurrent Neural Network), παρά όμως να χρησιμοποιηθούν RNN να χρησιμοποιηθεί Echo State Network για τη υλοποίησή του. Με αυτό τον τρόπο θα επιδιώξουμε να ενστερνιστεί το δίκτυο την ιδέα της αλληλεξάρτησης μεταξύ του κάθε αμινοξέως που προηγείται κάποιου άλλου και κάθε αμινοξέως που έπεται αυτού.

Βιβλιογραφία

Γ. Χριστοδούλου, " Διερεύνηση μεθόδων εκπαίδευσης Νευρωνικών Δικτύων Αμφίδρομης Ανάδρασης για Πρόβλεψη Δευτεροταγούς Δομής Πρωτεϊνών", Διπλωματική Εργασία, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου, 2010.

Μ. Αγαθοκλέους, "Πρόβλεψη δευτεροταγούς δομής πρωτεϊνών με Νευρωνικά δίκτυα Αμφίδρομης Ανάδρασης", Διπλωματική Εργασία, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου,2009

Αναφορές

[1] Ian Sommerville, 'Software Engineering', Pearson, 9th edition, 2011, ISBN 0-13- 705346-0.

[2] Hawking, S. (2000). Professor Stephen Hawking's website. Retrieved 9 February, 2009, from http://www.hawking.org.uk/

[3] H. van Vliet, Software Engineering: Principles and Practice, Third edition, John Wiley & Sons, 2008.

[4] XAMPP is a php development tool that use Apache, MariaDB, PHP and Perl, from https://www.apachefriends.org/index.html

[5] phpMyAdmin is a free software tool written in PHP, from https://www.phpmyadmin.net/

[7] B Robson and E Suzuki "Conformational properties of amino acid residues in globular proteins", J.Mol.Bio, Volume 107, Issue 3, 5, (1976): 327-356

[8] AV Guzzo, "Influence of Amino-Acid Sequence on Protein Structure" Biophys. J, Volume 5(6), (1965): 809-822.

[9] JW Prothero "Correlation between Distribution of Amino Acids and Alpha Helices", Biophys. J, Volume 6 (3), (1966): 367-370.

[10] M Schiffer and AB Edmundson "Use of Helical Wheels to Represent Structures of Proteins and to Identify Segments with Helical Potential", Biophys. J, Volume 7 (2), (1967): 121-135

[11] D Kotelchuck and HA Scheraga "The Influence of Short-Range Interactions on Protein Conformation, II. A Model for Predicting the a-Helical Regions of Proteins", Proc Natl Acad Sci USA, Volume 62 (1), (1969): 14-21

[12] PN Lewis and N Gō and M Gō and D Kotelchuck and HA Scheraga "Helix Probability Profiles of Denatured Proteins and Their Correlation with Native Structures", Proc Natl Acad Sci USA, Volume 65 (4), (1970): 810–815

[13] M Froimowitz and GD Fasman"Prediction of the secondary structure of proteins using the helix-coil transition theory". Macromolecules, Volume 7 (5), (1974): 583-589

[14] PY Chou and GD Fasman; Fasman "Prediction of protein conformation",Biochemistry, Volume 13 (2), (1974): 222-245

[15] J Garnier and DJ Osguthorpe and B Robson "Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins", J Mol Biol, Volume 120 (1), (1978): 97-120.

[16] W.S McCulloch and W Pitts "A Logical Calculus of the Ideas Immanent in Nervous

Activity", Bulletin of Mathematical Biophysics, Volume 5, (1943): 115-133.

[17] http://www.mind.ilstu.edu/curriculum/nature_of_computers/computer_types.php

[18] F.M. Richards and C.E Kundrot "Identification of Structural Motifs from Proteins Coordinate Data: Secondary Structure and First-level Supersecondary Structure", Proteins, Volume 3,Issue 2, (1988) : 71-84.

http://users.auth.gr/~voyatzis/SeniorThesis/mTsouxnika.pdf

[19] P. Baldi, S. Brunak, P. Frasconi, G. Soda and G. Pollastri "Exploiting the Past and the Future in Protein Secondary Structure Prediction", Volume 15, Issue 11,Bioinformatics, (1999) : 937-946.

[20] P. Baldi, S. Brunak, P. Frasconi, G. Soda and G. Pollastri "Bidirectional Dynamics for Protein Secondary Structure Prediction", Sequence Learning, Volume 1828, (2000) : 80-104.

[21] H. Jaeger, "A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach " , Fifth revision,(2013)
Μεταφορτώθηκε στις 27/10/2015:
(http://minds.jacobs-university.de/sites/default/files/uploads/papers/ESNTutorialRev.pdf)

[22] H. Jaeger, "The "echo state" approach to analyzing and training recurrent neural networks – with an Erratum note1", Corrected version of the technical report H. Jaeger (2001): The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, German National Research Center for Information Technology, (2010)

[23] F. Triefenbach, A. Jalalvand, B. Schrauwen, and J.P. Martens,"Phoneme recognition with large hierarchical reservoirs",In Advances in Neural Information Processing Systems 23, (2011): 2307-2315,2011.

[24] D. Verstraeten, J. Dambre, X. Dutoit, and B. Schrauwen."Memory versus non-linearity in reservoirs", In Proc. Int Neural Networks (IJCNN) Joint Conf, (2010): 1-8.

[25] M. Hermans and B. Schrauwen,"Recurrent kernel machines: Computing with infinite echo state networks", Neural Computation, Volume 24(1), (2012): 104-133.

[26] M. Lukosevi ˇ cius, "A practical guide to applying echo state networks", Neural Networks: Tricks of the Trade. Springer, (2012): 659–686.

[27] M. H. Hayes,"Statistical Digital Signal Processing and Modeling",John Wiley & Sons, (1996).

[28] S. Haykin,"Adaptive filter theory",Prentice Hall, (1991).

[29] Mantas Lukoˇsevi ˇcius," A Practical Guide to Applying Echo State Networks", Springer Berlin Heidelberg, Volume 7700(2), (2012): 659-686.
Μεταφορτώθηκε στις 10/5/2016:
"http://organic.elis.ugent.be/sites/organic.elis.ugent.be/files/PracticalESN.pdf"

[30] Daniel Dean," Recursive Least Squares for Echo State Network Damage Compensation"
Μεταφορτώθηκε στις 13/5/2016:
"http://doc.gold.ac.uk/aisb50/AISB50-S05/AISB50-S5-Dean-paper.pdf"

[31] F. Rosenblatt, (1957)," The Perceptron--a perceiving and recognizing automaton, Cornell Aeronautical Laboratory, Report 85-460-1.

[32] S. Papert and M. L. Minsky," Perceptrons: an introduction to computational geometry",Cambridge, Mass: MIT Press, (1988).

[33] Øksendal, Bernt,"Stochastic Differential Equations: An Introduction with Applications ", Berlin: Springer, Edition 6, (2003):11.

[34] N Qian, TJ Sejnowski, "Predicting the secondary structure of globular proteins using neural network models", Journal of molecular biology, (1988).

[35] Rost B: "PHD, predicting one-dimensional protein structure by profile-based neural networks", Methods in Enzymology, Volume 266, (1996): 525-539.

[36] B Rost and C Sander, "Prediction of protein secondary structure at better than 70% accuracy". J. Mol. Biol, Volume 232, (1992): 584-599.

[37] C.B. Do and K. Katoh,"Protein Multiple Sequence Alignment",
Μεταφορτώθηκε στις 17/5/2016:
"http://ai.stanford.edu/~chuongdo/papers/alignment_review.pdf"

[38] PHPMailer, A full-featured email creation and transferclass for PHP, GitHub
Μεταφορτώθηκε στις 25/3/2016:
"https://github.com/PHPMailer/PHPMailer "

[39] R. Pascanu and T. Mikolov and Y. Bengio,"On the difficulty of training recurrent neural networks"
Μεταφορτώθηκε στις 25/5/2016:
"http://jmlr.org/proceedings/papers/v28/pascanu13.pdf"

[40] Mark D. Skowronski and John G. Harris, "Automatic speech recognition using a predictive echo state network classifier", Science Direct, Volume 20, Issue 3, (2007): 414-423

[41] K. Ishu, "Identification of motion with echo state network", OCEANS '04. MTTS/IEEE TECHNO-OCEAN '04, Volume 3, (2004): 1205 - 1210

[42] Decai Li, "Chaotic Time Series Prediction Based on a Novel Robust Echo State Network", IEEE Transactions on Neural Networks and Learning Systems, Volume 23, Issue 5, (2012): 787-799

[43] R. Sacchi, "Water Inflow Forecasting using the Echo State Network: a Brazilian Case Study", 2007 International Joint Conference on Neural Networks, (2007): 2403 – 2408

[44] C. Starr and B. McMillan, "Human Biology", (2016)

[45] Γ. Παρασκευάς and Μ. Κατερίνα and Σ. Διαμαντής, "Βιοχημεία", Τεχνολογικής Κατεύθυνσης, Γ' ΤΑΞΗΣ ΓΕΝΙΚΟΥ ΛΥΚΕΙΟΥ, Ψηφιακό Σχολείο, Κεφάλαιο 3-Πρωτείνες.
Μεταφορτώθηκε στις 01/5/2016:

“ http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C120/480/3166,12748/ “

[46] A.K. Konopka and M. James and C. Crabbe, “Compact Handbook of Computational Biology”, (2004):155-156

Παραρτήματα

Παράρτημα 1: stroreProtein.py

```
from numpy import *
import protein


class storeProteins:

    def __init__(self):
        self.sizeOfAminoascids=0
        self.aminoacids=[]
        self.data=[]
        self.yt = []

    def readProteins(self,lines):
        for i in range(0, len(lines), 3):
            name = lines[i].rstrip()
            secondary=lines[i+2].rstrip()
            msa=loadtxt("../input/msaFilesCorrect/"+name+'.hssp')
            self.data=append(self.data,msa/100)
            self.sizeOfAminoascids+=len(msa)
            self.normilizeOutput(secondary)
    def readAminoAcids(self,fileName):
        with
open("../input/msaFilesCorrect/"+fileName+".hssp","r+") as f_in:
            lines = filter(None, (line.rstrip() for line in f_in))
        return lines
    def normilizeOutput(self,secondary):
        se=secondary.split()
        for t in se[0]:

            if(t=="C"):
                self.yt.append([1,0,0])
                self.aminoacids.append(t)
            elif (t=="E"):
                self.yt.append([0,0,1])
                self.aminoacids.append(t)
            elif (t=="H"):
                self.yt.append([0,1,0])
                self.aminoacids.append(t)
            else:
```

```
            continue
        yt=[0,1,0]
        random.shuffle(yt)
        self.yt.append(yt)
```

Παράρτημα 2: Main.py

```
import ESN
from numpy import *
from matplotlib.pyplot import *
import storeProteins
import Oger


network = ESN.TrainESN(
    maxIterations=10,
    initLen=100,
    inSize=20,
    outSize=3,
    resSize=40,
    a=0.3,
    window=15,
    sparsity=0.2,
    spectral_radius=1.25,
    forgettingFactor=0.1,
    regression=1e-8,
    num_folds=10
)
network.run("../input/TrainingSets/msaProteinsTrainBigDataset
_afterProcess corr.txt")
```

Παράρτημα 3: ESN.py

```
# -*- coding: utf-8 -*-
"""
A minimalistic Echo State selfs demo with Mackey-Glass (delay
17) data
in "plain" scientific Python.
by Mantas LukoÅ¡eviÄ?ius 2012
http://minds.jacobs-university.de/mantas
"""
from numpy import *
from matplotlib.pyplot import *
import storeProteins
import Oger
```

```
class TrainESN:
    def
__init__(self,maxIterations,initLen,inSize,outSize,resSize,a,
window,sparsity,spectral_radius,forgettingFactor,regression,n
um_folds):
        self.maxIteration=maxIterations
        self.initLen = initLen
        self.inSize = inSize
        self.outSize = outSize
        self.resSize = resSize
        self.a = a
        self.window=window
        self.sparsity=sparsity
        self.spectral_radius=spectral_radius
        self.forgettingFactor=forgettingFactor
        self.regression=regression
        self.outTrain = open("output/OutputTrain.txt", "w")
        self.outTest = open("output/OutputTest.txt", "w")
        self.num_folds=num_folds

    def init_weights(self):
        random.seed(42)
        self.W_in = (random.rand(self.resSize,1+self.inSize)-
0.5) * 1
        self.W = random.rand(self.resSize,self.resSize)-0.5
        print 'Computing spectral radius...',
        rhoW = max(abs(linalg.eig(self.W)[0]))
        print 'done.'
        print rhoW
        self.W *= self.spectral_radius / rhoW
        #print self.W.tolist()
        self.W[random.rand(*self.W.shape) < self.sparsity] =
0
        #print self.W.tolist()
        #self.W_out = (random.rand(self.resSize,
self.outSize)-0.5)*1
        self.x = zeros((self.resSize, 1))
        self.xTest = zeros((self.resSize, 1))

    def init_error_accuracy(self):
         self.TrainError=0
         self.TestError=0
         self.TrainAccuracy=0
         self.TestAccuracy = 0
    def createplot(self):
```

```
        plot(array(self.TrainErrorplot))
        title("Train Error")
        ylabel('Error')
        xlabel('Iterations')
        show()
        plot(array(self.TestErrorplot))
        title("Test Error")
        ylabel('Error')
        xlabel('Iterations')
        show()
        plot(array(self.TrainAccuracyplot))
        title("Train Accuracy")
        ylabel('Accuracy %')
        xlabel('Iterations')
        show()
        plot(array(self.TestAccuracyplot))
        title("Test Accuracy")
        ylabel('Accuracy %')
        xlabel('Iterations')
        show()

    def init_training(self,trainData,testData):
        self.trainData = trainData
        self.testData = testData
        self.trainLen=self.trainData.sizeOfAminoascids
        self.testLen=self.testData.sizeOfAminoascids

        self.Yt = self.trainData.yt[self.initLen:]
        self.X = zeros((self.resSize,self.trainLen-
self.initLen))
        self.Y = zeros((self.outSize,self.trainLen-
self.initLen))

#self.Secondary=array(self.trainData.aminoacids)[self.initLen
:]


    def run(self,filename):
        open_file = open(filename,"r")
        lines = open_file.readlines()
        subset_size = len(lines)/self.num_folds/3
        self.output = open("output.txt", "w")
        sumTest=0
        sumTrain=0
        for j in range (self.num_folds):
            self.init_weights()
```

```
            pTrain=storeProteins.storeProteins()
            pTest=storeProteins.storeProteins()

            testing_this_round =
lines[j*subset_size*3:][:subset_size*3]
            training_this_round = lines[:j*subset_size*3] +
lines[(j+1)*subset_size*3:]

            pTrain.readProteins(training_this_round)
            pTest.readProteins(testing_this_round)
            self.init_training(pTrain,pTest)
            #self.output.write("--- training "+str(j)+"---
/n")
            self.TestErrorplot=[]
            self.TrainErrorplot=[]
            self.TrainAccuracyplot=[]
            self.TestAccuracyplot=[]
            for i in range(self.maxIteration):
                self.init_error_accuracy()
                if(i<2):
                    self.W_out = (dot(
dot(array(self.Yt).T,self.X.T), linalg.inv(
dot(self.X,self.X.T) + \

self.regression*eye(self.resSize) ) )).T
                #print self.W_out
                self.p= zeros((1,self.resSize))+i
                self.training()

self.TrainErrorplot.append(self.TrainError/self.iterationTrai
n)

self.TrainAccuracyplot.append(self.TrainAccuracy)
                if(i==self.maxIteration-1):
                    self.outTrain.write(str(
self.TrainError)+" "+ str(self.TrainAccuracy)+ "
"+str(self.correctTrain) +" " +str(self.iterationTrain)
+"\n")
                    sumTrain=sumTrain+self.TrainAccuracy
                self.testing()

self.TestErrorplot.append(self.TestError/self.iterationTest)

               # print str(self.TrainErrorplot) +"
"+str(self.TestErrorplot)
                if(i==self.maxIteration-1):
```

```
                    self.outTest.write(str( self.TestError)+"
"+ str(self.TestAccuracy)+" "+str(self.correctTest)+"
"+str(self.iterationTest)+"\n")
                    sumTest=sumTest+self.TestAccuracy

self.TestAccuracyplot.append(self.TestAccuracy)

            #del(self.X)
            print j
            #self.createplot()

        self.outTest.write(str(sumTest/self.num_folds)+"\n")

self.outTrain.write(str(sumTrain/self.num_folds)+"\n")


    def training(self):
        iteration=-1
        lr=10
        self.iterationTrain=0
        for w in range
(0,self.trainLen/self.inSize,self.window):
            for iterW in range (0,self.window,1):

u=zip(*[iter(self.trainData.data[(w+iterW)*self.inSize:(w+ite
rW)*self.inSize+self.inSize])]*1)
                self.x = array((1-self.a)*self.x +
self.a*tanh( dot( self.W_in, vstack((u,[1])) ) + dot(self.W,
self.x ) ))
                iteration+=1
                if iteration>=self.initLen:
                    self.X[:,iteration-self.initLen] =
self.x[:,0]
                    y =tanh(dot( self.W_out.T,self.x ))
                    y = reshape(y,(1,self.outSize))[0]
                    maxPosition=argmax(y)
                    yout=zeros(3)
                    yout[maxPosition]=1
                    self.Y[:,iteration-self.initLen]=yout
                    # self.Y[:,iteration-self.initLen]=y
                    #print y
                    if iterW==self.window-1:
                        self.iterationTrain+=1

self.TrainAccuracy=self.TrainAccuracy+1-
sum(abs(self.Yt[iteration-self.initLen]-self.Y[:,iteration-
```

```
self.initLen]))/2

                        #print str(self.numToA(maxPosition))
+ " " + str(self.Secondary[iteration-self.initLen])+ "
"+str(self.TrainAccuracy)

#self.output.write(str(self.numToA(maxPosition)) + " " +
str(self.Secondary[iteration-self.initLen])+ " "+"\n")

                        error=self.Yt[iteration-
self.initLen]-y
                        #print ((error) ** 2).mean(axis=None)
                        er=(((self.Yt[iteration-
self.initLen]-y)+1)/3 ** 2).mean(axis=None)

self.TrainError=self.TrainError+sum(pow(er,2))/self.outSize
                        dieretisg=array(self.forgettingFactor
+ dot( array(self.x).T*self.p ,self.x ))

#dieretisg[dieretisg==inf]=self.regression
                        g=dot(self.p ,self.x)/dieretisg
                        self.p=pow(self.forgettingFactor,-
1)*self.p - g*array(self.x).T*pow(self.forgettingFactor,-
1)*self.p

                        #print str(sum(self.TrainAccuracy))+
" " +str(sum(error))+ " " + str(g) +" "+str(dieretisg)


self.W_out=self.W_out+(sum(error)/self.outSize*g)*lr

        self.correctTrain=self.TrainAccuracy

self.TrainAccuracy=(self.TrainAccuracy/self.iterationTrain)*1
00


    def testing(self):
        self.Ytest= zeros((self.outSize,self.testLen))
        self.YtTest = self.testData.yt
        self.Secondary=array(self.testData.aminoacids)

        iteration=-1
        self.iterationTest=0
        for w in range
(0,self.testLen/self.inSize,self.window):
```

```
            for iterW in range (0,self.window,1):

u=zip(*[iter(self.testData.data[(w+iterW)*self.inSize:(w+iter
W)*self.inSize+self.inSize])]*1)
                self.xTest = array((1-self.a)*self.xTest +
self.a*tanh( dot( self.W_in, vstack((u,[1])) ) + dot(self.W,
self.xTest ) ))

                iteration+=1
                y = tanh(dot( self.W_out.T,self.xTest ))+1
                y = reshape(y,(1,self.outSize))[0]
                maxPosition=argmax(y)
                yout=zeros(3)
                yout[maxPosition]=1
                self.Ytest[:,iteration]=yout

                if iterW==self.window-1:
                    error=((self.YtTest[iteration]-y)+1)/3
                    # print error
                    # print "-----"

                    er=(((self.YtTest[iteration]-y)+1)/3 **
2).mean(axis=None)

self.TestError=self.TestError+sum(pow(er,2))/self.outSize
                    self.iterationTest+=1
                    self.TestAccuracy=self.TestAccuracy+1-
sum(abs(self.YtTest[iteration]-self.Ytest[:,iteration]))/2
                    #print str(self.numToA(maxPosition)) + "
" + str(self.Secondary[iteration])+ "
"+str(self.TestAccuracy)




        self.correctTest=self.TestAccuracy

self.TestAccuracy=(self.TestAccuracy/self.iterationTest)*100

    def numToA(self,possition):

        if (possition==0):
            return("C")
        elif (possition==1):
            return("E")
```

```
elif (possition==2):
    return("H")
else:
    print "ERROR !!!"
```