Ατομική Διπλωματική Εργασία

ΠΡΌΓΝΩΣΗ ΣΥΝΟΛΙΚΟΥ ΑΡΙΘΜΟΥ ΕΠΙΤΕΥΞΗΣ ΤΕΡΜΑΤΩΝ ΠΟΔΟΣΦΑΙΡΙΚΩΝ ΑΓΩΝΩΝ ΜΕΣΩ DEEP NEURAL NETWORKS

Αντρέας Φράγκου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2016

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πρόγνωση συνολικού αριθμού επίτευξης τερμάτων ποδοσφαιρικών αγώνων μέσω Deep Neural Networks

Αντρέας Φράγκου

Επιβλέπων Καθηγητής

Δρ. Χριστός Χριστοδούλου

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

Μάιος 2016

Ευχαριστίες

Θα ήθελα να ευχαριστήσω ιδιαιτέρως τους συμφοιτητές μου Άντρια Κκουσιή, Ειρήνη Παπακώστα και Παναγιώτη Παυλίδη που χάριν της πολύτιμης βοήθειας τους ολοκληρώθηκε η διπλωματική μου εργασία.

Επίσης ευχαριστώ την οικογένεια μου που μου έδωσε πάσης φύσεως βοήθεια όλα αυτά τα χρόνια.

Τέλος στον καθηγητή μου Δρ. Χρίστο Χριστοδούλου αλλά και τον βοηθό καθηγητή Μιχάλη Αγαθοκλέους για τις συμβουλές και την επίβλεψη της πορείας της Διπλωματικής μου Εργασίας.

Περίληψη

Η Διπλωματική μου εργασία έχει ως σκοπό την υλοποίηση ενός τεχνητού νευρωνικού δικτύου στο οποίο θα προβλέπεται ο αριθμός των τερμάτων που θα επιτευχθούν σε ένα ποδοσφαιρικό αγώνα, με επακόλουθο το κέρδος έπειτα από στοιχηματική δραστηριότητα.

Δηλαδή, θα λαμβάνονται, ως είσοδος σε αυτό, δεδομένα και στατιστικά από προηγούμενους αγώνες για τις δύο διαγωνιζόμενες ομάδες, όπως παραδείγματος χάριν ο μέσος όρος τερμάτων υπέρ και κατά, το ποσοστό αγώνων διατήρησης ανέπαφης εστίας, το ποσοστό αγώνων που σημειώθηκαν άνω των δυόμιση τερμάτων κ.α. Τα στατιστικά αυτά θα συνυπολογίζονται και θα προβλέπεται το κατά πόσον ο αριθμός των τερμάτων που θα επιτευχθούν σε έκαστο, μελλοντικό, ποδοσφαιρικό αγώνα θα υπερβαίνουν ή όχι τον αριθμό δυόμιση. Η τιμή αυτή αποτελεί σημείο στοιχήματος στα πρακτορεία, το λεγόμενο under-over.

Διπλωματικές εργασίες παρόμοιες με την δική μου έγιναν και από άλλους προπτυχιακούς φοιτητές με τον καθένα να προσθέτει, μέσω των πορισμάτων της εργασίας του, την δική του γνώση η οποία με βοήθησε να αναπτύξω την δική μου διπλωματική.

Για την υλοποίηση και τον τρόπο επίλυσης του ανωτέρω προβλήματος αποφασίσαμε, από κοινού με τον επιβλέποντα καθηγητή μου, να χρησιμοποιηθεί μια Deep Learning τεχνική αφού τα υλοποιημένα δίκτυα βασισμένα σε αυτές τις τεχνικές, προσφάτως έχουν να επιδείξουν σπουδαία επιτεύγματα σε πεδία της τεχνίτης νοημοσύνης όπως αναγνώριση φωνής, εικόνων, παραγωγή λόγου αλλά και σε άλλα διαφορετικά προβλήματα στο πεδίο αυτό.

Μια από τις προαναφερθείσες τεχνικές είναι και τα Recurrent Neural Networks την οποία αποφάσισα να χρησιμοποιήσω για το σύστημα μου. Τα RNN μπορούν να εκπαιδευτούν μέσω της «επίβλεψης», δηλαδή κατά την εκπαίδευση να εισάγεται στο σύστημα η επιθυμητή έξοδος την οποία το σύστημα προσπαθεί να προσεγγίσει ούτως ώστε να αυξηθεί η αποτελεσματικότητα του. Έτσι για κάθε αγώνα εισαγόταν μαζί με

τα στατιστικά των ομάδων και η επιθυμητή έξοδος, more ή less, εξαρτώμενη από το αποτέλεσμα του κάθε αγώνα.

Το πρώτο στάδιο της εργασίας αποτέλεσε η άντληση των δεδομένων από την ιστοσελίδα www.betexplorer.com η οποία παρέχει αποτελέσματα αγώνων για κάθε χρονιά. Η διοργάνωση στην οποία έγινε η έρευνα είναι η Serie A, δηλαδή το πρωτάθλημα της πρώτης κατηγορίας του ιταλικού πρωταθλήματος. Μέσω λοιπόν ενός Web Crawler γραμμένου στην php, παίρνονται τα αποτελέσματα της κάθε χρονιάς και γράφονται σε ένα αρχείο κειμένου. Από εκεί και μέσω της γλώσσας Java διαβάζονται, κατηγοριοποιούνται ανά ομάδα και έπειτα για κάθε μια από αυτές δημιουργείται ένας πίνακας με τα στατιστικά για κάθε αγωνιστική της υποχρέωση. Τέλος για κάθε αγώνα δημιουργείται μια σειρά από στατιστικά, και αυτά γράφονται σε ένα δεύτερο αρχείο. Αυτό διαβάζεται μέσω της γλώσσας Python, στην οποία είναι γραμμένο το δίκτυο μου, και μπαίνουν στο RNN δίκτυο, το οποίο εκπαιδεύεται ούτως ώστε για κάθε επερχόμενη είσοδο να μπορεί να βγάλει όσο το δυνατόν συχνότερα μια σωστή πρόβλεψη.

Τα αποτελέσματα της εργασίας μου είναι αρκετά ικανοποιητικά, παρόλο το παράξενο του εγχειρήματος, αφού τα RNN δέχονται κατά κύριο λόγο χρονοσειρές, αφού φαίνεται πως το σύστημα «μαθαίνει» και σταδιακά αυξάνει το ποσοστό εύρεσης σωστών προβλέψεων για την τύχη των αγώνων, σε περιπτώσεις όπου δεν έχει δει ξανά κατά την εκπαίδευση του.

Για τις ανάγκες της διπλωματικής έχουν δημιουργηθεί δύο ειδών δίκτυα RNN το καθένα με τη δική του επιτυχία. Το κάθε δίκτυο εκτελέστηκε με δεδομένα εισόδου τις χρονικές περιόδους από το 2003 μέχρι το 2014 και είχε ως δεδομένα ελέγχου τη χρονιά 2014-2015. Το πρώτο δίκτυο παρουσίασε αποτελέσματα της τάξης του 57% και για δεδομένα εισόδου και για τα δεδομένα εξόδου. Όσον αφορά το δεύτερο δίκτυο έχει να επιδείξει μεγαλύτερη ικανότητα εκμάθησης αφού φτάνει το ποσοστό επιτυχίας 70% για τα δεδομένα εκπαίδευσης και το 60% για τα δεδομένα ελέγχου.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Στόχος

Σκοπός της εργασίας αυτής ήταν η υλοποίηση ενός νευρωνικού δικτύου, στη βάση των RNN το οποίο έπειτα από την εκπαίδευση του θα μπορεί για κάθε αγώνα να προβλέπει την κατάληξη του στο πεδίο του More-Less. Η τεχνική αυτή δεν είχε χρησιμοποιηθεί στο παρελθόν για το συγκεκριμένο πρόβλημα και έτσι, αναλογιζόμενος της ευημερία της στην επίλυση πολυδιάστατων προβλημάτων, επιχείρησα την υλοποίηση του με σκοπό να λάβω καλύτερα αποτελέσματα από ότι οι προγενέστεροι. Ειδικότερα ο σκοπός της διπλωματικής μου είναι η αύξηση του κέρδους των παικτών που αποφασίζουν να ακολουθήσουν το συγκεκριμένο σημείο στο στοίχημα μέσω της ένδειξης εμπιστοσύνης στα αποτελέσματα που το σύστημα μου εξάγει.

1.2 Προηγούμενες προσπάθειες

Δεν θα μπορούσα να ξεκινήσω την υλοποίηση μου χωρίς να δω τι έχουν κάνει οι προηγούμενοι συμφοιτητές μου, με σκοπό να πάρω τα θετικά τους στοιχεία και να αφήσω στην άκρη τα όχι τόσο επιτυχημένα. Αυτό με βοήθησε να μην χρονοτριβήσω σε σημεία όπως την λογική για το ποια στατιστικά θα χρησιμοποιήσω για να αναπαραστήσω έναν αγώνα ως είσοδο του δικτύου. Πιο κάτω θα αναφερθούν σε γενικές γραμμές στον τρόπο με τον οποίο εργάστηκε ο κάθε φοιτητής και ποια ήταν τα αποτέλεσμα του.

Το 2001 ο Matt Jackson του πανεπιστημίου Birkbeck του Λονδίνου επιχείρησε να εντοπίσει το αποτέλεσμα του κάθε αγώνα ως προς τον νικητή. Έτσι με την χρήση τεχνιτών νευρωνικών δικτύων Kohonen, για ομαδοποίηση των δεδομένων, και Back Propagation επιχείρησε να υπολογίσει την κατάληξη μελλοντικών αγώνων. Αναλόγως της δυσκολίας του προβλήματος και των τεχνικών που χρησιμοποίησε ήταν και τα αποτελέσματα του αφού δεν παρουσίασαν αρκετό ενδιαφέρον. Τα αποτελέσματα του ήταν 47% με το Back Propagation και 57% με το Kohonen, αλλά αυτά τα αποτελέσματα σύμφωνα με τον ίδιο, έχουν επιτευχθεί κατά κύριο λόγο, λόγω της εύρεσης της εντός έδρας νίκης η οποία δεν αποφέρει μεγάλο στοιχηματικό κέρδος[1].

Μια άλλη προσπάθεια έχει γίνει από τον Kevin Davies, φοιτητή του ίδιου πανεπιστημίου το 2004. Η εργασία του είχε τις ίδιες προδιαγραφές με την προηγούμενη δηλαδή να βρεθεί ο νικητής του αγώνα όμως επιθυμούσε να βρει αγώνες όπου το στοιχηματικό κέρδος ήταν αυξημένο. Υλοποιώντας τις ίδιες τεχνικές βασίστηκε πιο πολύ στο πως θα προ-επεξεργαστεί τα δεδομένα ούτως ώστε να πάρουν μια μορφή που θα βοηθούσαν το σύστημα να βγάλει καλύτερα αποτελέσματα. Παρόλα αυτά όμως τα ποσοστά επιτυχίας του ήταν ανάλογα με του συμφοιτητή του και έτσι η αύξηση του κέρδους δεν είχε επιτευχθεί σε ικανοποιητικό βαθμό[2].

Μετά τις δυο όχι και τόσο επιτυχημένες προσπάθειες από τους πάρα πάνω φοιτητές το 2005 η Sian Turner αποφάσισε να αλλάξει λίγο την μορφή της εργασίας της και έτσι θέλησε να ακολουθήσει το More-Less σημείο, ομοίως με την παρούσα διπλωματική εργασία. Χρησιμοποίησε και αυτή Back Propagation και αφού επεξεργάστηκε αρκετά

τα δεδομένα εισόδου, με σκοπό την απόκτησης της βέλτιστης τους μορφής, μπόρεσε να έχει αποτέλεσμα ικανοποιητικό με ποσοστό σωστών προβλέψεων 57% με μικρό κέρδος[3].

Η επόμενες δυο προσπάθειες έγιναν από δικούς μας φοιτητές και ειδικότερα η πρώτη από τον Θεοφάνη Νεοκλέους το 2007. Ο ίδιος είχε αρκετά καλά αποτελέσματα και αφού χρησιμοποίησε επίσης το Back Propagation για More-Less, ανέβηκε στο 62%, ποσοστό που του επέφερε σε περιπτώσεις 23% αύξηση στα χρήματα που είχε ποντάρει. Αυτό επιτεύχθηκε μετά από πολλές δοκιμές και μετατροπές των δεδομένων εισόδου καθώς και του συστήματος του[4].

Τέλος ο φοιτητής με το μεγαλύτερο ποσοστό επιτυχίας ήταν ο Αντρέας Ιακώβου όπου έφτασε στο 70%. Αυτή η προσπάθεια έγινε με την χρήση RBF μιας τεχνικής που κατά κύριο λόγο ειδικεύεται στην κατηγοριοποίηση των δεδομένων και έτσι πήρε αποτέλεσμα 68%, καθώς επίσης χρησιμοποίησε και SVM με το ποσοστό να φτάνει στο 70%. Ένα ποσοστό πολύ μεγάλο το οποίο αν μπορέσει να εντοπιστεί θα επιφέρει σημαντικό στοιχηματικό κέρδος σε όποιον ακολουθούσε το σύστημα του[5].

Βλέποντας και μελετώντας τις πάρα πάνω εργασίες προφανώς και αποφάσισα να ασχοληθώ με το More-Less αφού φαίνεται να είναι ένα είδος στοιχήματος που μπορούν τα νευρωνικά δίκτυα να μάθουν. Είναι φανερό πως τα αποτελέσματα των πιο πάνω φοιτητών, όσο προόδευε η μελέτη, αυξάνονταν και έφτασαν σε σημείο που ίσος να μην μπορούν να ξεπεραστούν. Η μεγάλη πρόκληση λοιπόν είναι να φτάσω τα πάρα πάνω αποτελέσματα αλλά κυρίως να δω αν το συγκεκριμένο πρόβλημα μπορεί να βρει εφαρμογή στα RNN και αν αυτά μπορούν να βγάλουν ανάλογα αποτελέσματα με τις πάρα πάνω διπλωματικές εργασίες.

Κεφάλαιο 2

Υπόβαθρο

2.1 Νευρωνικά δίκτυα – Ιστορική Αναδρομή

Τα νευρωνικά δίκτυα εισήχθηκαν ως έννοια για πρώτη φορά από τους McCullock και Pitts το 1943. Οι επινοητές λοιπόν των νευρωνικών δικτύων παρουσίασαν την άποψη ότι ένα νευρωνικό δίκτυο, για παράδειγμα ο ανθρώπινος εγκέφαλος, αποτελείται από μια συλλογή πολλών απλών νευρώνων και πως οι νευρώνες αυτοί διασυνδεμένοι θα μπορούσαν να λειτουργήσουν με σκοπό την αναπαράσταση ανθρωπίνων λειτουργιών, σε ένα πρότυπο ανάλογο του ηλεκτρικού κυκλώματος[6].

Έτσι οι δυο τους, McCullock νευροφυσιολόγος και Pitts μαθηματικός, άνοιξαν τον δρόμο και χάραξαν την πορεία για την ανάπτυξη των τεχνιτών νευρωνικών δικτύων στο επίπεδο που είναι σήμερα. Μετά από τέσσερα χρόνια συνεχούς μελέτης έδωσαν στη δημοσιότητα ένα εξελιγμένο μοντέλο για αναγνώριση προτύπων, όπου ο κάθε νευρώνας θα μπορούσε να έχει μια έξοδο, δηλαδή δυο καταστάσεις ενεργός ή ανενεργός, αλλά πολλές εισόδους που επηρεάζουν την έξοδο αυτή. Η έξοδος ενός νευρώνα στη συνέχεια με τις εξόδους άλλων νευρώνων, αποτελούν είσοδο για έναν επόμενο νευρώνα και έτσι σιγά σιγά κτίζεται ένα νευρωνικό δίκτυο. Όταν ένας νευρώνας είναι ενεργός τότε πυροδοτεί, δηλαδή στέλνει ένα παλμό προς τα εμπρός και

έτσι η αλληλουχία των παλμών εντός ενός δικτύου χαράσσει την διαδρομή προς την έξοδο όλοκληρου του δικτύου[7][8].

Όταν το δίκτυο φτάσει στην τελική έξοδο τότε μια ίνα που ενώνει την έξοδο ενός κυττάρου με τις εισόδους του δημιουργεί μια ανάδραση δηλαδή μεταβάλει την δύναμη που μια είσοδος κατέχει, το βαθμό δηλαδή που επηρεάζει την έξοδο του. Αυτή η δύναμη αποτελεί και την μνήμη του δικτύου.

Πιθανώς όμως αυτή η σπουδαία ανακάλυψη να έμενε μόνο στα χαρτιά αν ο J. Von Neumann δεν οραματιζόταν ότι αυτή η βιολογική ανακάλυψη μπορούσε να εφαρμοστεί στον αναπτυσσόμενο, την δεκαετία του πενήντα, ηλεκτρονικό υπολογιστή. Τα πρώτα τεχνητά δίκτυα δημιουργήθηκαν αυτή την εποχή έπειτα από προσπάθειες για άντληση πληροφοριών από τα βιολογικά δίκτυα και αναπαράσταση τους μέσω των μηχανών[9].

Αφού άρχισε να διαδίδεται η νέα αυτή ιδέα, δεν άργησαν να εμφανιστούν και τα πρώτα βιβλία-έρευνες που μέχρι και σήμερα στηρίζουν όλα τα νεότερα συστήματα νευρωνικών δικτύων. Έτσι ο D.Hebb μέσω του βιβλίου του ‘‘The organisation of behavior’’ το 1949 δημιούργησε την βασική ιδέα για το πως οι συνδέσεις μεταξύ των νευρώνων επιτυγχάνονται και αναθεωρούνται κατά την εκπαίδευση του δικτύου. Έπειτα από πολλά νευροφυσιολογικά πειράματα κατέληξε στο γενικό συμπέρασμα πως οι συνδέσεις-βάρη μεταξύ των νευρώνων ενισχύονται ή αποδυναμώνονται μετά από κάθε είσοδο με σκοπό την δημιουργία της μνήμης στο δίκτυο. Έτσι όταν μια ενεργοποιημένη είσοδος συμβάλλει συνεχώς στην διέγερση του νευρώνα τότε αυτή η σύνδεση, μεταξύ εισόδου και εξόδου, δυναμώνει και επηρεάζει περισσότερο τον νευρώνα με τον οποίο ασχολούμαστε[10].

Μετά από τα πρώτα βασικά βήματα, άρχισαν να δημιουργούνται τα πρώτα μοντέλα. Το 1957 ο F.Rosenblatt έφτιαξε πάνω σε hardware, το πρώτο δίκτυο που μπορούσε να λύσει πολλά προβλήματα και δημιούργησε άκρατο ενθουσιασμό μεταξύ των επιστημόνων οι οποίοι πίστεψαν πως έγινε η αρχή για την λύση όλων των άλυτων, μέχρι εκείνη τη στιγμή προβλημάτων. Η μεγαλοποίηση των δυνατοτήτων του δικτύου του Rosenblatt όμως, μετά από πιο προσεκτικές έρευνες έγινε αντιληπτό πως ήταν βεβιασμένη, αφού παρουσιάστηκαν στην πορεία οι ισχυροί περιορισμοί του[11].

Πιο ολοκληρωμένη έρευνα πάνω σε αυτό το δίκτυο έκαναν οι Minsky και Papert στο βιβλίο τους ‘‘Perceptrons’’. Έκαναν γνωστές λοιπόν τις δυνατότητες και τους περιορισμούς του δικτύου, αποδεικνύοντας τους μαθηματικά, και έθεσαν το πεδίο στο οποίο το δίκτυο αυτό μπορεί να κινηθεί. Έτσι η επιστημονική κοινότητα εγκατέλειψε μερικώς τις τεχνικές αυτές και ακολούθησε μια νέα, παρεμφερή τεχνική, την Τεχνητή Νοημοσύνη[12].

Πέρασαν αρκετά χρόνια για να γίνει μια μεγάλη αλλαγή στα δεδομένα των τεχνιτών νευρωνικών δικτύων και αυτό έγινε το 1982 οπότε και παρουσίασε ο φυσικός John Hopfield το έργο του, δίνοντας μια νέα, γερή ώθηση προς την ανάπτυξη των δικτύων αυτών. Σε μια εργασία του μόλις πέντε σελίδων απέδειξε πως ένα νευρωνικό δίκτυο μπορεί να χρησιμοποιηθεί ως αποθηκευτικός χώρος αλλά και να ανακτήσει πληροφορίες για ένα σύστημα έστω και αν του δοθούν μόνο μερικά κομμάτια του. Η συγκεκριμένη εργασία αναθέρμανε το ενδιαφέρων πολλών και η σημαντικότητα της έγινε έμπνευση για τη ανάπτυξη πολλών νέων ιδεών[13].

Πέρα από τα πάρα πάνω το 1986 έγινε μια άλλη σημαντική ανακάλυψη, αυτή τη φορά από τους McClelland και Rumelhart με το δημοσίευμα τους ‘‘Parallel Distributed Processing’’. Η ιδέα τους ήταν πως ένα νευρωνικό δίκτυο μπορεί να λειτουργήσει ως ένας παράλληλος επεξεργαστής. Μέσω του επιτεύγματος τους αυτού αποκαλύφτηκε πως επιτρέπετε η παρουσία πολλαπλών επιπέδων νευρώνων στο ενδιάμεσο επίπεδο καθώς επίσης θεσπίστηκε η διαδικασία αλλαγής βαρών με την μέθοδο back-propagation, την πιο χρήσιμη τεχνική εκπαίδευσης δικτύων μέχρι σήμερα[14].

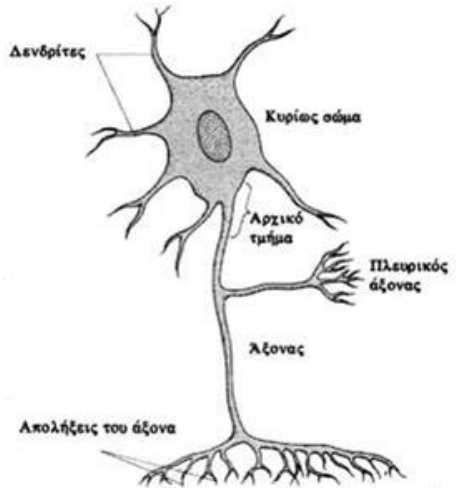
Μετά από τις ανακαλύψεις της δεκαετίας του ογδόντα τα νευρωνικά δίκτυα άρχισαν να δημιουργούν το δικό τους πεδίο στην επιστήμη, με δικά τους ξεχωριστά γνωρίσματα και χαρακτήρα. Έτσι επιστήμονες έχουν αφοσιωθεί καθολικά στη νέα αυτή περιοχή, αναπτύσσοντας την ραγδαία αλλάζοντας της παράλληλα επίπεδο. Άρχισαν να γίνονται τακτικά συνέδρια για αυτά και εκδώσεις περιοδικών αποκλείστηκα για την μελέτη των νευρωνικών δικτύων. Τελευταία, στη διάρκεια κάθε μήνα εμφανίζεται μια πλειάδα από καινούριες μελέτες και απόψεις για τα νευρωνικά δίκτυα γεγονός που δείχνει πως η ενασχόληση των επιστημών στο θέμα αν μη τη άλλο βρίσκεται σε ψηλό επίπεδο.

Συνοπτικά, θέλοντας να εντοπίσουμε τους κύριους λόγους που οι άνθρωποι ανέπτυξαν τα τεχνητά νευρωνικά δίκτυα θα λέγαμε πως αυτό έχει πραχθεί πρωτίστως μέσω του οράματος των ανθρώπων να γνωρίσουν τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου αλλά και για να κυνηγήσουν την πιθανότητα δημιουργίας μηχανών οι οποίες να μπορούν να λειτουργούν περίπλοκα, ομοίως με τον ανθρώπινο εγκέφαλο. Τα προαναφερθέντα ιστορικά επιτεύγματα έχουν συντείνει προς αυτή την κατεύθυνση όμως παραμένουν αρκετά μαύρα σύννεφα πάνω από τους δύο αυτούς στόχους, τα οποία προς το παρών δεν έχει γίνει κατορθωτό να απομακρυνθούν. Συνεπακόλουθο της πιο πάνω ενέργειας θα ήταν η εξ ολοκλήρου αντίληψη για το πώς είναι δομημένος ο εγκέφαλος άρα και για το πώς θα μπορέσουμε να δημιουργήσουμε μια μηχανή αντάξια του.

2.2 Νευρώνες του εγκεφάλου

Οι νευρώνες αποτελούν τη βασική μονάδα του νευρικού μας συστήματος και λειτουργούν παράγοντας ηλεκτρικά σήματα, τα οποία μεταδίδονται από ένα μέρος του νευρώνα σε άλλο. Ο ασταμάτητος χορός των εκατομμυρίων αυτών νευρώνων έχει προσελκύσει τα τελευταία περίπου εκατόν χρόνια, το ενδιαφέρον των επιστημών χωρίς όμως ευτυχή κατάληξη, να γίνει δηλαδή αντιληπτός ο συνολικός τρόπος λειτουργίας του ανθρώπινου εγκεφάλου[15].

Το κάθε εγκεφαλικό κύτταρο μπορεί να χωριστεί σε τέσσερα βασικά μέρη, όπως φαίνεται και στην εικόνα 2.1.



Εικόνα 2.1 Νευρώνας εγκεφάλου 1

- Οι δενδρίτες διακλαδώνονται, προς τα έξω του κυρίως σώματος του κυττάρου, και λαμβάνουν πληροφορίες-ερεθύσματα από το περιβάλλον, τα οποία αποθηκεύουν και μεταβιβάζουν στο κυρίως σώμα του.

- Το κυρίως σώμα τώρα περιέχει τον πυρήνα του κυττάρου όπως και τον μηχανισμό παραγωγής πρωτεϊνών.

- Ο άξονας ή νευρική ίνα είναι η προέκταση του κυτταρικού σώματος όπου μεταφέρει τα ερεθύσματα μέσω ηλεκτρικών σημάτων από τον νευρώνα στις απολήξεις με μεγάλες ταχύτητες, σχεδόν ταυτόχρονα.

- Τέλος τα σήματα αυτά καταλήγουν στις νευρικές απολήξεις που περιέχουν συναπτικά κυστίδια τα οποία απελευθερώνουν μια χημική ουσία τον νευροδιαβιβαστή, σε περίπτωση που το κύτταρο υποστεί διέγερση. Ο νευροδιαβιβαστής λειτουργεί ως αγγελιαφόρος προς τους νευρώνες που είναι ενωμένοι μαζί με τον εν λόγω νευρώνα και μεταφέρει το σήμα του προς αυτούς. Έτσι η μεταφορά των ερεθισμάτων από νευρώνα σε νευρώνα καθιστά δυνατή την ένωση τους και όλοι μαζί καθιστούν ένα νευρωνικό δίκτυο, τον εγκέφαλο[16].

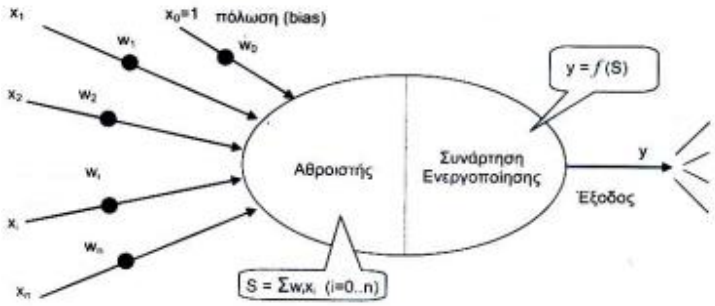
Το σύστημα αυτό των νευρώνων ενωμένων με συνάψεις, αποτελεί την υλική βάση της μάθησης, της μνήμης και των αισθημάτων στη ζωή του ανθρώπου. Όσα περισσότερα ερεθίσματα δέχεται ο εγκέφαλος τόσο περισσότερες είναι και οι συνάψεις που δημιουργούνται άρα και υψηλότερη νοημοσύνη, γνωσιακές λειτουργίες και συναισθηματική ωρίμανση.

2.3 Τεχνητά νευρωνικά δίκτυα

Τα τεχνητά νευρωνικά δίκτυα αναπτύχθηκαν αφού οι συμβατικοί υπολογιστές δεν μπορούσαν να ακολουθήσουν τον τρόπο που ο εγκέφαλος λειτουργεί. Όπως είναι λογικό, προσπαθούν να μεταβιβάσουν τη γνώση που έχουμε για τα νευρωνικά δίκτυα και να την προσομοιώσουν στους ηλεκτρονικούς υπολογιστές σε μια προσπάθεια προσέγγισης της λειτουργίας του ανθρώπινου εγκεφάλου.

Έτσι δημιουργείται αριθμός κόμβων, παίρνουν τον ρόλο των κυττάρων, και πάνω σε κάθε έναν από αυτούς τοποθετούνται ενώσεις προς άλλους κόμβους-κύτταρα, που αντιπροσωπεύουν τις συνάψεις. Οι τιμές των ενώσεων αυτών μεταβάλλονται κατά την εκπαίδευση του δικτύου ούτως ώστε να λάβουν την τελική τους κατάσταση και επομένως να αποτελέσουν την μνήμη του. Έτσι σε τυχούσες επόμενες εισόδους να μπορεί να αντεπεξέλθει σύμφωνα με αυτές τις τιμές, μια λογική που βρίσκεται κοντά στο τι ο ανθρώπινος εγκέφαλος εκτελεί.

Οι νευρώνες των τεχνητών νευρωνικών δικτύων στην απλή τους μορφή παρουσιάζονται πιο κάτω.



Εικόνα 2.2 Τεχνητός Νευρώνας

Από αριστερά προς τα δεξιά βλέπουμε τις εισόδους του δικτύου με τα σύμβολα x_1, x_2, …, x_n. Τα σήματα αυτά εισόδου μεταβάλλονται ανάλογα με τα βάρη w_i που ενώνουν την κάθε είσοδο με τον νευρώνα. Οι τιμές αυτές δηλαδή αντιπροσωπεύουν το πώς και πόσο μια είσοδος επηρεάζει την έξοδο του νευρώνα.

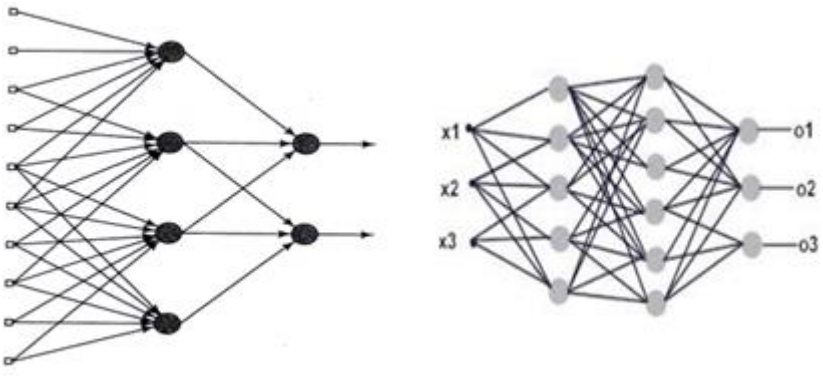
Ο αθροιστής λαμβάνει τις εισόδους, πολλαπλασιάζοντας τις με τα αντίστοιχα βάρη, τις προσθέτει και το αποτέλεσμα του το εισάγει στην συνάρτηση ενεργοποίησης όπου και αποφασίζεται αν ο νευρώνας αυτός θα αποτελέσει ένα ενεργό ή ένα ανενεργό νευρώνα του δικτύου για τις συγκεκριμένες εισόδους[17].

2.3 Αρχιτεκτονικές Νευρωνικών Δικτύων

Σύμφωνα με τον τρόπο σύνδεσης των νευρώνων ενός δικτύου, αναγνωρίζεται και η αρχιτεκτονική του. Το ποια θα είναι η αρχιτεκτονική του δικτύου όμως παίζει σημαντικό ρόλο για την έκβαση της εκπαίδευσης και την μετέπειτα λήψη αποτελεσμάτων. Έτσι λοιπόν, αντιλαμβανόμενος ο καθένας για το ποιο πρόβλημα θέλει να επιλύσει, πριν να αρχίσει η υλοποίηση του θα πρέπει να αναλογιστεί και να αποφασίσει εκ των προτέρων ποια αρχιτεκτονική ταιριάζει καλύτερα στο πρόβλημα, άρα ποια μετά την εκπαίδευση θα παρέχει την πιο καλή λύση για αυτό. Θα πρέπει να είναι γνωστοί επίσης και ποιοι αλγόριθμοι εκπαίδευσης ταιριάζουν στην αρχιτεκτονική αυτή και σε ποιοι όχι[18].

Οι αρχιτεκτονικές αυτές μπορούν να διαχωριστούν σε τρείς βασικές κατηγορίες.

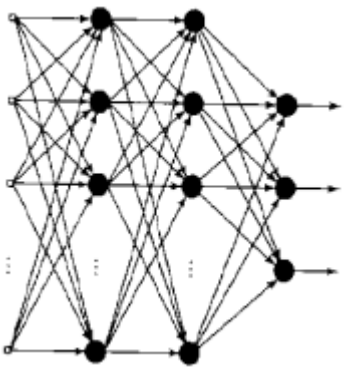
Η πρώτη κατηγορία είναι τα δίκτυα εμπρόσθιας τροφοδότησης με ένα κρυφό επίπεδο. Με τον όρο εμπρόσθια τροφοδότηση εννοούμε πως οι διασυνδέσεις μεταξύ των νευρώνων πηγαίνουν ομαλά προς μια κατεύθυνση σε νευρώνες άλλων, επόμενων, επιπέδων και δεν επιστρέφουν πίσω ούτε πηγαίνουν σε νευρώνα του ίδιου επιπέδου. Στο μοναδικό κρυφό επίπεδο καταχωρούνται οι είσοδοι του νευρώνα και στη συνέχεια γίνονται οι υπολογισμοί του νευρώνα τα αποτελέσματα των οποίων πηγαίνουν προς τους νευρώνες εξόδου.



Εικόνα 2.3 Αριστερά ένα μερικώς και δεξιά ένα πλήρως συνδεδεμένο δίκτυο

Η δεύτερη κατηγορία αρχιτεκτονικής δικτύων είναι η πρόσθιας τροφοδότησης, αυτή την φορά όμως με πολλαπλά κρυφά επίπεδα. Η διαφορά επομένως της πρώτης κατηγορίας με αυτήν είναι η ύπαρξη αριθμού πάνω του ενός κρυφών επιπέδων. Προσθέτοντας περισσότερους νευρώνες, το δίκτυο μας γίνεται ικανότερο στο να αφομοιώνει περισσότερες πληροφορίες για τα δεδομένα εισόδου λόγο των πιο πολύπλοκων διασυνδέσεων του. Αυτό μπορεί να γίνει αντιληπτό στην περίπτωση που το πρόβλημα μας δεν είναι γραμμικώς διαχωρίσιμο, άρα μη επιλύσημο με ένα κρυφό επίπεδο, αλλά επιλύσημο με περισσότερους από έναν. Η συνεχής πρόσθεση όμως όλο και περισσοτέρων νευρώνων και επίπεδων στο δίκτυο, αυξάνει την πολυπλοκότητα του, γεγονός που πιθανώς σε περιπτώσεις να μην είναι επιθυμητό.

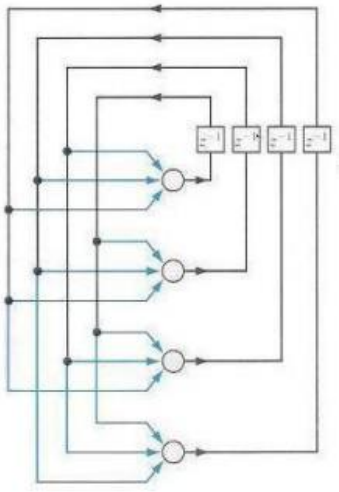
Ο τρόπος λειτουργίας αυτής της κατηγορίας αρχιτεκτονικής είναι παρόμοιος με την πρώτη. Οι νευρώνες ενός επιπέδου μπορεί να συνδέονται ή όχι με όλους τους νευρώνες του επόμενου επιπέδου ενώ οι νευρώνες κάθε επιπέδου κάνουν τους υπολογισμούς αναλόγως των εισόδων τους, με το αποτέλεσμα τους να μεταφέρεται στο επόμενο συνδεδεμένο επίπεδο. Η διαδικασία αυτή εκτελείται επαναληπτικά, μέχρις ότου φτάσουμε στην έξοδο. Η τοπολογία σύνδεσης του δικτύου για κάθε πρόβλημα, πιθανών να πρέπει να είναι διαφορετική ούτως ώστε να αποτελεί τον βέλτιστο αρχιτεκτονικό σχεδιασμό και να αποφέρει τα καλύτερα δυνατά αποτελέσματα.



Εικόνα 2.4 Αρχιτεκτονική εμπρόσθιας τροφοδότησης

με δυο κρυφά επίπεδα νευρώνων

Τελευταία αρχιτεκτονική, και η πιο περίπλοκη είναι η αναδρομική-επαναληπτική σύνδεση. Όταν αναφερόμαστε σε αναδρομική σύνδεση εννοούμε πως τουλάχιστον μία σύνδεση, με τη μορφή βρόγχου επιστρέφει προς τα πίσω, δηλαδή εισάγεται και αυτή ως είσοδος του δικτύου, συνυπολογιζόμενη με τις εισόδους που θα προκύψουν την επόμενη χρονική περίοδο. Τα δίκτυα αυτά πιθανός να έχουν ή όχι κρυφό επίπεδο και έτσι διαχωρίζονται δύο κατηγορίες. Σε αναδρομικά όπου η επιστροφή γίνεται από το κρυφό επίπεδο, επομένως η έξοδος των νευρώνων του κρυφού επιπέδου γίνονται είσοδοι, και η δεύτερη, αυτή όπου οι νευρώνες εξόδου προσφέρουν την είσοδο στην επόμενη περίοδο. Αυτές οι αρχιτεκτονικές προσφέρονται ιδιαιτέρως σε συστήματα τα οποία θέλουμε να θυμόμαστε προηγούμενες καταστάσεις, και τα εξαχθέντα αποτελέσματα προηγούμενων εισόδων, αφού μέσω της ανάδρασης αυτής πετυχαίνεται η πιο πάνω ιδέα. Η ύπαρξη λοιπόν των βρόγχων στο δίκτυο μας επηρεάζει τόσο την εκπαίδευση του συστήματος μας όσο και την επίδοση του.



Εικόνα 2.5 Δίκτυο αναδρομικής αρχιτεκτονικής.

2.5 Deep networks

Τα Deep Networks μέσω και των μεθόδων του Deep Learning έχουν δημιουργήσει μια νέα τάξη πραγμάτων στις μέρες μας. Τα Deep Networks βασίζονται στις ικανότητες τους να μοντελοποιούν μεγάλου όγκου δεδομένων, υψηλού επιπέδου, μέσω της χρήσης αποδοτικών αλγορίθμων καθώς επίσης και μέσω των πολλαπλών κρυφών επιπέδων τα οποία εντοπίζονται μεταξύ των νευρώνων εισόδου και εξόδου. Χάρις αυτού έχουν βρει μέχρι τώρα, τις καλύτερες λύσεις για προβλήματα όπως αναγνώριση εικόνας, αναγνώριση ομιλίας καθώς και στην επεξεργασία της φυσικής γλώσσας.

Οι νευρώνες αυτοί, που βρίσκονται στο ενδιάμεσο-κρυφό επίπεδο, δημιουργούν μια ιεραρχική δομή, χωρίζοντας τα χαρακτηριστικά των δεδομένων σε ομάδες, ούτως ώστε να μπορούν να παρθούν τα επιθυμητά αποτελέσματα στην έξοδο, για διαφορετικές εισόδους. Η δημιουργία της δομής-αναπαράστασης των δεδομένων, όπως και ο τρόπος επεξεργασίας αυτών, αποτελεί μια πολύπλοκη διαδικασία η οποία βασίζεται σε εμπνευσμένες τεχνικές ανεπτυγμένες μέσα από την μελέτη, και την πρόοδο αυτής, στις νευροεπιστήμες καθώς και στον εντοπισμό της σχέσης ερεθισμάτων του περιβάλλοντος και της ανταπόκρισης νευρώνων του ανθρώπινου εγκεφάλου στα συγκεκριμένα ερεθίσματα[19].

Η μέθοδος των Deep Networks παρουσιάζεται ως μια ταχέως εξελίξιμη, αφού επιλέγεται από πολλούς ειδικούς για επίλυση σύγχρονων προβλημάτων. Ευημερεί επομένως στις μέρες μας, όμως το μέλλον της προδιαγράφεται λαμπρότερο, αποτελώντας μια ελπιδοφόρα τεχνική που ενδέχεται να λύνει με καλύτερο τρόπο προβλήματα που στο παρελθόν έμοιαζαν αρκετά δύσκολα.

Πολλές είναι οι υπηρεσίες στις οποίες έχουν βρει εφαρμογή τα Deep Networks. Έχουν χρησιμοποιηθεί λοιπόν σε μεγάλη κλίμακα, ενώ παράλληλα έχουν κατακτήσει πολλά διεθνή βραβεία σε μεγάλες διεθνείς διοργανώσεις[20]. Οι σημαντικότερες επιτυχίες των Deep Networks, και οι συχνότερες περιπτώσεις χρήσης τους, εντοπίζονται στην ανάλυση εικόνων, εντοπισμό αντικειμένων σε αυτές και κατηγοριοποίηση τους. Οι εικόνες οι οποίες έχουν χρησιμοποιηθεί για εκμάθηση των δικτύων αυτών

αναπαριστούν είτε χειρόγραφους χαρακτήρες (π.χ. λατινικά, αραβικά και κινέζικα) είτε εικόνες αντικειμένων. Έτσι επήλθε και η ανάπτυξη εφαρμογών προς αναγνώριση αντικειμένων και προσώπων σε εικόνες, λύνοντας παράλληλα αρκετά σύγχρονα προβλήματα.

Εντωμεταξύ οι μέθοδοι αυτές έχουν υπεισέρθει επίσης και στην ιατρική, μέσω της αναγνώρισης ιατρικών εικόνων ανιχνεύοντας περιπτώσεις ασθενών με καρκίνου του μαστού. Ακόμη τα Deep Networks βρήκαν εφαρμογή και στην Βιοχημεία και στην παραγωγή φαρμάκων. Στον εν λόγω τομέα η χρησιμότητα εντοπίζεται στην εύρεση των επιδράσεων των διαφόρων χημικών στοιχείων σε θρεπτικές ουσίες του ανθρώπινου σώματος. Το συγκεκριμένο γεγονός μπορεί να βοηθήσει καταλυτικά στην επίτευξη της δημιουργίας φαρμάκων για καταπολέμηση ιών, αφού υπάρχει η δυνατότητα, μέσω αυτών, να εντοπίσουν οι επιρροές που αναφέρθηκαν πάρα πάνω και έτσι τα φάρμακα που θα δημιουργούνται να περνούν τα στάδια ελέγχου που απαιτούνται και εν τέλει να βγαίνουν στην αγορά γρηγορότερα[20].

Τα συγκεκριμένα δίκτυα επομένως παρουσιάζουν μεγάλο ενδιαφέρον, αφού η χρήση τους για οποιασδήποτε μορφής πρόβλημα αποτελεί πρόκληση για τους επιστήμονες και ένα μικρό ερωτηματικό κατά πόσον μπορούν να ανταπεξέρχονται σε προβλήματα δύσκολα για τους «συμβατούς» αλγόριθμους.

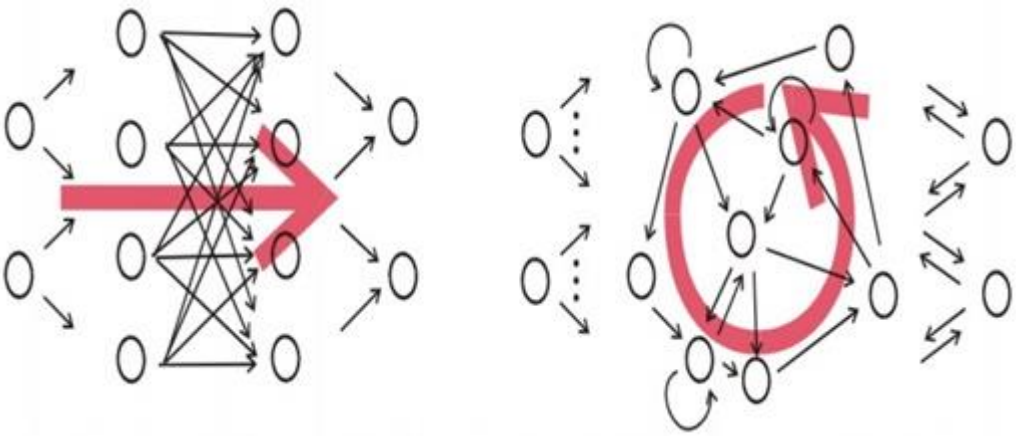
2.6 Recurrent Neural Networks

Τα Recurrent Neural Networks τώρα, αποτελούν μια μέθοδο υλοποίησης των Deep Networks η οποία για πολλά προβλήματα έχει οδηγήσει στα καλύτερα αποτελέσματα, σε σχέση με άλλα δίκτυα που βρίσκονται κάτω από την ομπρέλα αυτή. Άρα στα δίκτυα αυτά κληρονομούνται, όλα τα γνωρίσματα των Deep Networks, επομένως και τα προτερήματα τους[19]. Η γενική τους λειτουργία λοιπόν είναι η μοντελοποίηση πολύπλοκων δεδομένων, με μη γραμμικές σχέσεις, μέσω των κρυφών επιπέδων τους αλλά και των αλγορίθμων που διέπονται της συγκεκριμένης μεθόδου.

Τα Recurrent Neural Networks, όπως το όνομα τους προδίδει, χρησιμοποιούν επαναληπτική δομή για να βρουν το αποτέλεσμα εξόδου. Η επανάληψη αυτή μπορεί να χαρακτηριστεί ως μνήμη του συστήματος, αφού το δίκτυο στηριζόμενο σε αυτήν αποφασίζει για το ποια θα είναι η έξοδος κάθε διαφορετικής εισόδου. Η ιδιότητα τους αυτή, τα ωθεί να μοιάζουν με τα βιολογικά νευρωνικά συστήματα, τα οποία και λειτουργούν με αναδρομή, θυμούμενα δηλαδή παλιές «εμπειρίες». Βρίσκουν χρήση επομένως και επιλύουν με μεγάλη ακρίβεια προβλήματα όπως αυτά της αναγνώρισης χειρόγραφου κειμένου, ομιλίας κτλ, αφού χρησιμοποιούν τη μνήμη που προαναφέρθηκε, επιτρέποντας τους να θυμούνται προηγούμενες καταστάσεις. Έτσι εξάγουν τα αποτελέσματα τους βασιζόμενα στην παρούσα είσοδο αλλά και στα αποτελέσματα που έχουν εξαχθεί από τις προηγηθείσες. Έτσι με το πέρασμα μιας αλληλουχίας εισόδων από το δίκτυο, δημιουργούνται σε αυτό πρότυπα τα οποία ανακτώνται σε περιπτώσεις νέας εμφάνισης παρεμφερών εισόδων με αυτές ούτως ώστε να εξαχθεί το σωστό αποτέλεσμα.

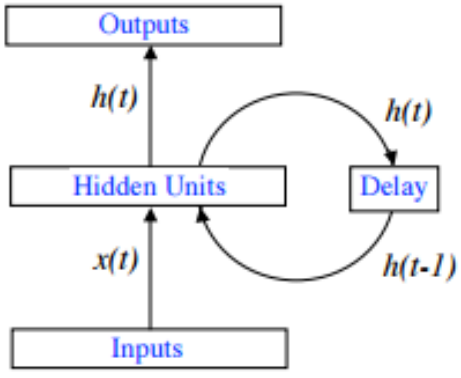
Μιας άλλης μορφής προβλήματα τα οποία λύνουν είναι και αυτά της πρόβλεψης μέσω της χρονικής συσχέτισης. Όπως αναφέρθηκε και στις αρχιτεκτονικές των νευρωνικών δικτύων, τα συγκεκριμένα δίκτυα αποτελούν επαναλαμβανόμενα δίκτυα, των οποίων οι διασυνδέσεις δημιουργούν κατευθυνόμενους κύκλους. Αυτό δημιουργεί μια εσωτερική κατάσταση στο δίκτυο, το οποίο εμφανίζει δυναμική χρονική συμπεριφορά, δηλαδή έχει κατά μίαν έννοια «αίσθηση» του χρόνου συνδέοντας «ενέργειες» που συμβαίνουν συχνά η μία μετά την άλλη.

Το γεγονός αυτό τους δίνει τη δυνατότητα να ανταποκρίνεται διαφορετικά σε κάθε χρονική στιγμή με βάση τις εισόδους που προηγήθηκαν αυτής αλλά και της είσοδο που μόλις έχει έρθει. Σε αντίθεση λοιπόν με τα πρόσθιας τροφοδότησης δίκτυα, τα RNN χρησιμοποιούν την εσωτερική τους μνήμη για την επεξεργασία και «ταίριασμα» αυθαίρετων ακολουθιών που λαμβάνουν ως είσοδο. Συμπερασματικά αυτός είναι και ο τρόπος που μπορούν να μάθουν χρονικές ακολουθίες. Αυτό τους δίνει τη δυνατότητα να αναγνωρίζουν, να αναπαράγουν και να προβλέπουν ακολουθίες μέσω αυτής της συσχέτισης που έχουν δημιουργήσει, βασιζόμενα στην εκπαίδευση τους.



Εικόνα 2.6 Δίκτυο εμπρόσθιας και επαναλαμβανόμενης τροφοδότησης(αριστερά προς δεξιά)

Το απλούστερο και συχνότερα χρησιμοποιημένο RNN δίκτυο είναι το πλήρες συνδεδεμένο επαναλαμβανόμενο νευρωνικό που φαίνεται και στο σχήμα.



Εικόνα 2.7 Ενδεικτικός τρόπος λειτουργίας RNN

Πιο πάνω βλέπουμε σχηματικά πως η έξοδος από τους νευρώνες που βρίσκονται στο κρυφό επίπεδο εισάγονται ξανά ως είσοδοι κατά το επόμενο βήμα εκπαίδευσης. Η μονάδα καθυστέρησης(delay) που βλέπουμε δεικνύει πως κατά την εκπαίδευση στο χρόνο t η κατάσταση του δικτύου, και δη των κρυφών νευρώνων του, στο χρόνο t-1 συνυπολογίζεται για να υπολογιστεί η έξοδος σε αυτό το χρονικό σημείο. Μπορεί όμως, αν αυτό είναι επιθυμητό, να προστεθεί στο δίκτυο όποια καθυστέρηση θεωρείται αναγκαία έτσι ώστε να συσχετίζονται η τωρινή είσοδος με πιο παλιές[21].

Άρα το τι κάνει το δίκτυο αυτό βασικά είναι να συνοψίζει τις πληροφορίες του παρελθόντος κρατώντας τις σημαντικότερες, παρέχοντας μια γενική περιγραφή τους σε κάθε βήμα εκπαίδευσης. Αυτή είναι και η δουλειά των νευρώνων που βρίσκονται στο κρυφό επίπεδο. Ο αριθμός των νευρώνων που λαμβάνει χώρα εκεί, αναπαριστά και τη διάσταση του συστήματος στο χώρο. Έτσι όσο πιο πολύπλοκες παρουσιάζονται οι είσοδοι τόσο πιο μεγάλος θα είναι και ο αριθμός των νευρώνων που βρίσκεται εκεί. Αυτό είναι και ένα σημείο που χρίζει δοκιμών, αφού δεν υπάρχει κάποια φόρμουλα ή τύπος που να καθορίζει τον κατάλληλο αριθμό νευρώνων που θα πρέπει να βρίσκονται στο σημείο αυτό. Άρα ο αριθμός τους, για κάθε ξεχωριστό πρόβλημα, καθορίζεται μετά από προσπάθειες και αλλαγές στις παραμέτρους αυτές σε συνδυασμό με τις άλλες παραμέτρους του συστήματος.

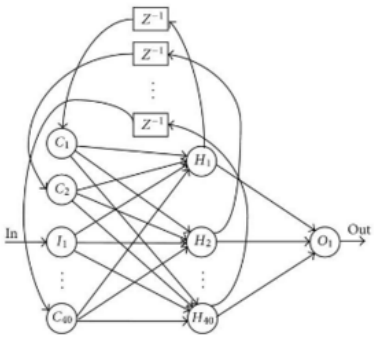
Μια γρήγορη ανασκόπηση της λειτουργίας τους λοιπόν είναι η εξής. Όταν μια νέα είσοδος μπει στο σύστημα τότε ενεργοποιούνται οι νευρώνες εισόδου, οι νευρώνες στο κρυφό επίπεδο υπολογίζουν τις τιμές εξόδου τους, ως μη γραμμικές συναρτήσεις, σύμφωνα με τις εισόδους που έρχονται σε κάθε νευρώνα, και ενεργοποιούνται ή όχι, στέλνοντας ή μη το ρεύμα εισόδου στους επόμενους μέχρι το σήμα να φτάσει στους νευρώνες εξόδου. Από εκεί και πέρα η εκπαίδευση των δικτύων αυτών γίνεται με δύο τρόπους. Με επιβλεπόμενη και ενισχυτική μάθηση. Στην περίπτωση που η μάθηση είναι επιβλεπόμενη, δηλαδή ορίζεται από πριν η επιθυμητή τελική κατάσταση του δικτύου, τότε ο «δάσκαλος» θέτει κατά κάποιο τρόπο στο δίκτυο ποιες διαδρομές νευρώνων θα πρέπει ή θα έπρεπε να ενεργοποιηθούν για κάθε είσοδο που μπαίνει στο σύστημα. Στόχος αυτής της ενέργειας είναι δίκτυο να εξοικειωθεί με την είσοδο αυτή και όταν έχει να αντιμετωπίσει εισόδους με κάποιας μορφής συσχέτισης με την συγκεκριμένη, να «αναπολεί» και να εξάγει το σωστό αποτέλεσμα. Στις περιπτώσεις,

από την άλλη, ενισχυτικής μάθησης το δίκτυο δέχεται ή ένα ενισχυτικό σήμα ή χρησιμοποιεί μια λειτουργία ανταμοιβής για την αξιολόγηση και εκπαίδευση του δικτύου.

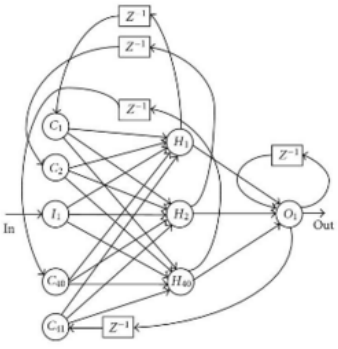
Αν και τα δίκτυα RNN φαίνονται περίπλοκα και ο τρόπος υπολογισμός της εξόδου δυσνόητος, παρουσιάζουν τα δυο εξής κύρια χαρακτηρίστηκα-προτερήματα που τα καθιστούν αξιόπιστα. Το πρώτο είναι η σταθερότητα μετά την εκπαίδευση. Έτσι δεν εντοπίζονται σε αυτά τάσεις ανομοιομορφίας αφού για παρόμοιες εισόδους αποφασίζουν για την κατάληξη τους με τον ίδιο τρόπο. Το δεύτερο είναι η τιθάσευση, αφού αν και στην αρχή πριν την εκπαίδευση το δίκτυο μπορεί να βγάζει οποιαδήποτε αποτελέσματα, μέσω αυτής, γίνετε ελέγξιμο, και μετά από ένα αριθμό επαναλήψεων καταλήγει στην επιθυμητή κατάσταση[21].

Τέλος ποια είναι η «δύναμη» των Recurrent Neural Networks φρόντισαν να αποκαλύψουν οι Siegelmann & Sontag το 1991[21]. Οι δύο τους λοιπόν απέδειξαν πως όλες οι μηχανές Turing μπορούν να προσομοιωθούν και να αναπαρασταθούν με ένα πλήρες συνδεδεμένο επαναλαμβανόμενο RNN με σιγμοειδή συνάρτηση. Η απόδειξη αυτή όμως δεν μας εξηγεί πως μπορούν σε κάθε περίπτωση να το επιτύχουν, αλλά ότι δύνανται να το κάνουν.

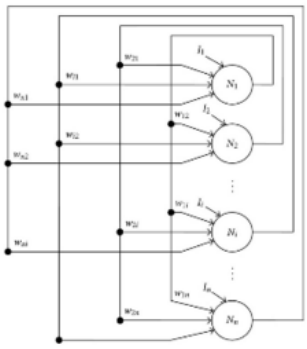
Φτάνοντας στην αρχιτεκτονική των RNN παρουσιάζονται πιο κάτω οι τρείς βασικές αρχιτεκτονικές των RNN σχηματικά. Φαίνεται λοιπόν πως υφίστανται οι διασυνδέσεις σε κάθε περίπτωση μεταξύ των νευρώνων εισόδου, του κρυφού επιπέδου και των εξόδων για τις τρείς βασικές αρχιτεκτονικές των Recurrent Neural Networks.



Elman Neural Network



Jordan Neural Network


Hopfield Neural Network

Ξεκινώντας από το δίκτυο Hopfield πρέπει να αναφέρουμε πως αποτελεί ένα ιστορικού ενδιαφέροντος δίκτυο, αφού ήταν το πρώτο που παρουσιάστηκε και ανήκει στην κατηγορία αυτή. Αρχιτεκτονικά, αποτελείται από ένα μόνο επίπεδο νευρώνων, των οποίων τα αποτελέσματα εισέρχονται ξανά σε αυτούς ως είσοδος. Σε αντίθεση με τα άλλα δύο δίκτυα στα οποία θα αναφερθούμε πιο κάτω, σε αυτό το δίκτυο δεν εφαρμόζεται κάποιος κανόνας εκπαίδευσης των βαρών, αφού τα βάρη που υπάρχουν καθορίζονται εξ αρχής από σταθερές τιμές. Η λειτουργία αυτή μπορεί να γίνει αντιληπτή αν σκεφτούμε τα σταθερά αυτά βάρη του δικτύου ως τα πρότυπα του, και για την εύρεση της εξόδου να χρησιμοποιούνται από αυτό ως μνήμη. Σκοπός της μνήμης αυτής του είναι να εντοπίζει, μέσω της ανάκλησης στα πρότυπα, και να αποφασίζει για το ποια θα είναι η έξοδος του δικτύου δηλαδή ποιο από τα πρότυπα αυτά θα επιλεγεί. Για να ολοκληρωθεί η εκπαίδευση του δικτύου Hopfield πρέπει αυτό, να φτάσει σε μια κατάσταση ισορροπίας. Τέτοια κατάσταση παρουσιάζει ένα σύστημα όταν οι τιμές εξόδου των νευρώνων από το προηγούμενο μέχρι το παρόν βήμα εκπαίδευσης έχουν μείνει σταθερές. Αυτό σημαίνει πως βρέθηκαν οι τιμές εξόδου των νευρώνων για τις συγκεκριμένες τιμές εισόδου, άρα και το ταιριαστό πρότυπο της. Λειτουργεί επομένως σαν μια συσχετιστική μνήμη η οποία για κάποια είσοδο ψάχνει στη μνήμη για να βρει πρότυπο όσο το δυνατόν πιο ταιριαστό με την είσοδο.

Καταλήγοντας στα δίκτυα Elman και Jordan πρέπει να αναφέρουμε πως είναι παρεμφερή μεταξύ τους, γι΄ αυτό συχνά εκφέρονται μαζί. Και στα δύο δίκτυα παρουσιάζεται ένα διάνυσμα εισόδου, και μέσω εμπρόσθιας εκπαίδευσης αποφασίζεται το αποτέλεσμα. Η μεγάλη τους διαφορά όμως έγκειται στο γεγονός ότι σε αυτό του Elman ως επιστροφή προς τους νευρώνες εισόδου παρουσιάζεται το αποτέλεσμα των κρυφών νευρώνων ενώ του Jordan αυτό των νευρώνων εξόδου. Αυτή η θεμελιώδης διαφορά όμως, οδηγεί το κάθε δίκτυο να εμφανίζεται ικανό να λύνει διαφορετικά προβλήματα. Το μεν λοιπόν, του Elman, μπορεί να λύσει καλύτερα προβλήματα αναγνώρισης προτύπων, όπως για παράδειγμα γραφής, ομιλίας κτλ, σε αντίθεση με του Jordan το οποίο χρησιμοποιείται κυρίως για την παραγωγή κινήσεων για ρομποτικά συστήματα. Γι αυτό και το δικό μου δίκτυο είναι προσαρμοσμένο στην αρχιτεκτονική του Elman αφού το πρόβλημα μου μπορεί να χαρακτηρισθεί ως πρόβλημα αναγνώρισης προτύπων κατά κάποιο τρόπο.

Φτάνοντας στα ποιο πρακτικά λοιπόν, αν οι είσοδοι ενός δικτύου παρασταθούν με το διάνυσμα x(t), οι νευρώνες στο κρυφό επίπεδο με h(t) και οι έξοδοι με y(t), καθώς επίσης και τα βάρη των εισόδων προς τους κρυφούς νευρώνες με W_{IH}, τα βάρη που ενώνουν τους κρυφούς νευρώνες με τους άλλους κρυφούς νευρώνες, δηλαδή τα βάρη που θα μπουν μαζί με την επόμενη είσοδο στο ανάλογο βήμα εκπαίδευσης με W_{HH} και τα βάρη από τους κρυφούς νευρώνες προς την έξοδο με W_{HO} τότε θα έχουμε τις γενικές εξισώσεις[21][22]:

$$h(t) = f_H(W_{IH} * x(t) + W_{HH} * h(t-1) + W_{HHbias})$$

$$y(t) = f_O\,(W_{HO} * h(t) + W_{HObias})f_H$$

Τα W_{HHbias} και W_{HObias} είναι οι τιμές των bias, για κάθε νευρώνα ξεχωριστά, η οποία προσθέτεται και αυτή ούτως ώστε να βρεθεί η έξοδος για κάθε νευρώνα.

Οι f_H και f_O είναι οι συναρτήσεις ενεργοποίησης για τους νευρώνες στο κρυφό επίπεδο και σε αυτό της εξόδου.

Η συχνότερα χρησιμοποιημένη συνάρτηση που καθορίζει το πότε ένας νευρώνας θα ενεργοποιηθεί είναι οι σιγμοειδής που περιγράφεται με τον τύπο.

$$\sigma(x)= 1/(1+e^{-x})$$

Το σφάλμα υπολογίζεται μέσω του τύπου του Mean Squared Error(MSE)

$$E=(d-y)^2$$

Όπου d είναι η επιθυμητή έξοδος ενώ y η πραγματική

Στη συνέχεια χρησιμοποιείται η μέθοδος κατάβασης κλήσης όπως δηλαδή και στον back-propagation αλγόριθμο. Ο σκοπός της λειτουργίας της μεθόδου αυτής είναι η ελαχιστοποίηση του συνολικού σφάλματος μέσω της αλλαγής των βαρών. Αυτό επιτυγχάνεται μέσω της αναλογίας της παραγώγου του σφάλματος με αυτήν των βαρών. Μετά από κάθε κλήση της μεθόδου αυτής επομένως υπολογίζεται η μεταβολή

των βαρών ούτως ώστε αυτά να αποκτήσουν τιμές οι οποίες να παρουσιάζουν το μικρότερο σφάλμα. Ο υπολογισμός αυτός πολλαπλασιάζεται με ένα μικρό αριθμό, τον ονομαζόμενο ρυθμό μάθησης, ο οποίος συμβολίζεται με το γράμμα μ. Η τιμή αυτή του ρυθμού μάθησης δηλαδή είναι ανάλογη με το μέγεθος μεταβολής των βαρών.

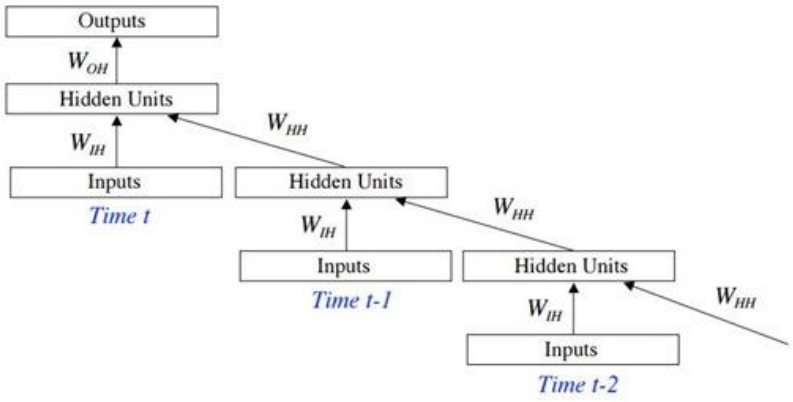
$$\Delta W_{ij} = -\mu * \frac{\partial E(t)}{\partial W_{ij}}$$

Για να καθίσταται εφικτός ο υπολογισμός της παραγώγου αναγκαία συνθήκη αποτελεί η γνώση της παραγώγου της συνάρτησης ενεργοποίησης του κάθε νευρώνα. Άρα αφού η σιγμοειδής συνάρτηση καλύπτει αυτή την προδιαγραφή, ούσα μια συνεχής συνάρτηση, άρα μπορεί να υπολογιστεί και το Δ.

Μόλις υπολογιστεί επομένως το Δ αυτό για κάθε βάρος, προσθέτετε στη μέχρι τώρα τιμή του ούτως ώστε να πάρει τη νέα του, «διορθωμένη» του μορφή.

$$\text{νέο } w_{ij} = w_{ij} + \Delta w_{ij}$$

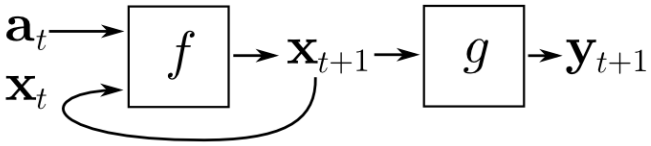
Για το πότε αλλαγή πότε πρέπει να γίνετε η αλλαγή των βαρών υπάρχουν δύο γενικευμένες ιδέες όπου η κάθε μία ταιριάζει σε διαφορετικές περιπτώσεις προβλημάτων. Η πρώτη ονομάζεται unfolding over time, «εποχιακή» εκπαίδευση, όπου τα αποτελέσματα των κρυφών νευρώνων πηγαίνουν όπως προαναφέραμε ως είσοδοι για το επόμενο διάνυσμα εισόδου ακολουθώντας επομένως η θεωρία της εμπρόσθιας τροφοδότησης για την αλίευση των αποτελεσμάτων. Όμως η διόρθωση των βαρών γίνεται με το πέρας της κάθε εποχής μετά την παροχή εκ του δικτύου του τελικού αποτελέσματος αφού με βάση αυτό γίνεται η εξακρίβωση του λάθους, δηλαδή το πόσο μακριά είμαστε από την ορθή-αναμενόμενη έξοδο. Όταν αναφερόμαστε δηλαδή σε εποχή, εννοούμε την ολοκλήρωση ενός κύκλου εκπαίδευσης άρα όταν όλα τα διανύσματα εισόδου περάσουν από το δίκτυο μια φορά[22][28].

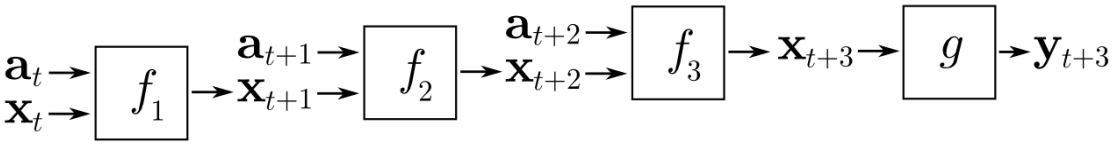


Εικόνα 2.8 διαγραμματικά η εποχιακή εκπαίδευση μέσω του Unfolding over time

Το σφάλμα σε αυτές τις περιπτώσεις υπολογίζεται με το άθροισμα των σφαλμάτων όλων των μεμονωμένων ακολουθιών, δηλαδή η «απόσταση» μεταξύ των ζευγαριών επιθυμητής και πραγματικής εξόδου με τον τύπο[24][27].

Η δεύτερη ιδέα για αλλαγή των βαρών είναι η Backpropagation πραγματικού χρόνου η οποία εκτελείται μετά από κάθε είσοδο, η συνεχής δηλαδή εκπαίδευση. Η τεχνική αυτή χρησιμοποιεί επομένως την μέθοδο κατάβασης κλίσης για την αλλαγή των βαρών, έπειτα από κάθε βήμα εκπαίδευσης. Άρα κατά την εκπαίδευση του δικτύου λοιπόν και όταν ένα διάνυσμα εισόδου παρουσιαστεί τότε γίνεται αρχικά η πρόσθια τροφοδότηση αμέσως ακολουθούμενη από το Backpropagation. Έτσι τα βάρη αλλάζουν συνεχώς ανεξάρτητα από το πόσο μεγάλη είναι η εποχή.





Πιο πάνω φαίνονται κατά σειρά η συνεχής αλλαγή των βαρών όταν και εκτελείται έπειτα από κάθε είσοδο ενώ στο κάτω σχήμα η εποχιακή όπου πρώτα υπολογίζεται το συνολικό αποτέλεσμα και έπειτα γίνεται η αλλαγή των βαρών του δικτύου.

Τελειώνοντας οφείλω να επισημάνω ότι ενώ στη θεωρία αναφέρεται πως τα RNN δίκτυα μπορούν να κάνουν χρήση αυθαίρετα προηγηθέντων εισόδων της παρούσης και να τις συνδυάσουν μεταξύ τους. Στην πράξη η οπτική εμβέλεια αυτή, σε εισόδους του παρελθόντος περιορίζεται όμως σε μερικά βήματα πιο πίσω γεγονός που μας αποτρέπει από του να έχουμε γνώση για πολύ προηγούμενες χρονικά εισόδους.

Για την επίλυση του προβλήματος αυτού, πρόσφατα εφευρέθηκε ένα νέο δίκτυο, βασισμένο στα RNN. Το προαναφερθέν δίκτυο ονομάζεται LSTM με το ακρώνυμο να αντιπροσωπεύει τις λέξεις Long Short-term Memory, φτιαγμένο εξ ολοκλήρου για να αποφεύγει το μακροπρόθεσμο πρόβλημα της εξάρτησης εισόδων μεταξύ τους. Τα δίκτυα αυτά παρουσιάζουν μια πιο εξελιγμένη έκδοση των επαναλαμβανόμενων δικτύων με αποτέλεσμα αυτού, την παρουσίαση μιας ικανότερης των άλλων αρχιτεκτονικών. Σε αυτά τα δίκτυα «επιτρέπεται» η πρόσβαση σε στοιχεία που έχουν παρουσιαστεί ως είσοδοι σε απροσδιόριστο παρελθοντικό χρόνο τα οποία ένα απλό RNN δίκτυο θα είχε «ξεχάσει». Ακολουθώντας λοιπόν μια πολύπλοκη διαδικασία συσχέτισης εισόδων με μεγάλη χρονική απόκλιση εμφάνισης, παρουσιάζουν μια πιο ισχυρή συσχετιστική μνήμη που αποδίδει καλύτερα σε προβλήματα όπως αναγνώριση προτύπων, αντικειμένων σε εικόνες, φωνής κτλ[25].

Κεφάλαιο 3

Σχεδιασμός και Υλοποίηση

3.1 Δεδομένα εισόδου

Τα δεδομένα εισόδου για το RNN δίκτυο, είναι τα αποτελέσματα των ομάδων που αγωνίζονται στην πρώτη κατηγορία του Ιταλικού πρωταθλήματος. Οι χρονιές στις οποίες έγινε η εκπαίδευση είναι από το 2003 μέχρι το 2014 και τα δεδομένα ελέγχου οι αγώνες που έγιναν το 2014-2015.

Ο μετρικές που θα χρησιμοποιήσω είναι οι ίδιες με αυτές που ο Θεοφάνης Νεοκλέους έχει προτείνει για τη δική του διπλωματική εργασία. Αυτό γιατί πριν αρχίσω και εγώ τη διπλωματική μου εργασία πίστευα πως θα μπορούσα να προσθέσω πολλά χαρακτηρίστηκα στις μετρικές αυτές, όπως ποδοσφαιριστές που είναι παροπλισμένοι, κίνητρο που έχουν οι ομάδες, βαθμολογική θέση και άλλα. Όμως όπως οι φοιτητές που ασχολήθηκαν πιο πριν από μένα κατάλαβαν, μετά από δοκιμές, είναι ότι όλα αυτά αντί να βοηθούν στην περεταίρω εκπαίδευση του δικτύου, μάλλον την εμποδίζουν. Έτσι και εγώ ακολούθησα αυτές τις μετρικές οι οποίες με το μόνο πράγματα που σχετίζονται είναι με τον αριθμό τερμάτων που δέχτηκαν και σκόραραν οι δύο διαγωνιζόμενες ομάδες.

Θα παραθέσω εδώ τις 26 μετρικές που λήφθηκαν υπόψη και αυτές με τις οποίες το δίκτυο με έχει εκπαιδευτεί:

1. Μέσος όρος τερμάτων που πέτυχε η γηπεδούχος ομάδα στην έδρα της. Υπολογίζεται ο μέσος όρος τερμάτων που πέτυχε η γηπεδούχος ομάδα στην έδρα της από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

2. Ποσοστό των παιχνιδιών που σκόραρε η γηπεδούχος ομάδα στην έδρα της. Υπολογίζεται το ποσοστό των παιχνιδιών που έχει σκοράρει η γηπεδούχος ομάδα στην έδρα της από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

3. Μέσος όρος τερμάτων που δέχτηκε η γηπεδούχος ομάδα στην έδρα της. Υπολογίζεται ο μέσος όρος τερμάτων που δέχτηκε η γηπεδούχος ομάδα στην έδρα της από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

4. Ποσοστό των παιχνιδιών που δέχτηκε τέρμα η γηπεδούχος ομάδα στην έδρα της. Υπολογίζεται το ποσοστό των παιχνιδιών που δέχτηκε τέρμα η γηπεδούχος ομάδα στην έδρα της από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

5. Μέσος όρος τερμάτων που πέτυχε η φιλοξενούμενη ομάδα εκτός έδρας. Υπολογίζεται ο μέσος όρος τερμάτων που πέτυχε η φιλοξενούμενη ομάδα εκτός έδρας από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

6. Ποσοστό των παιχνιδιών που σκόραρε η φιλοξενούμενη ομάδα εκτός έδρας. Υπολογίζεται το ποσοστό των παιχνιδιών που σκόραρε η φιλοξενούμενη ομάδα εκτός έδρας από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

7. Μέσος όρος τερμάτων που δέχτηκε η φιλοξενούμενη ομάδα εκτός έδρα. Υπολογίζεται ο μέσος όρος τερμάτων που δέχτηκε η φιλοξενούμενη ομάδα εκτός έδρας από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

8. Ποσοστό των παιχνιδιών που δέχτηκε τέρμα η φιλοξενούμενη ομάδα εκτός έδρας. Υπολογίζεται το ποσοστό των παιχνιδιών που δέχτηκε τέρμα η φιλοξενούμενη ομάδα εκτός έδρας από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

9. Ποσοστό των παιχνιδιών της γηπεδούχου ομάδας στην έδρα της όπου έχουν σημειωθεί περισσότερα των δύο τερμάτων.

Υπολογίζεται το ποσοστό των παιχνιδιών που προηγήθηκαν, της γηπεδούχου ομάδας στην έδρα της και τα τέρματα που σημειώθηκαν είναι περισσότερα των δύο.

10. Ποσοστό των παιχνιδιών της φιλοξενούμενης ομάδας εκτός έδρας όπου έχουν σημειωθεί περισσότερα των δύο τερμάτων.

Υπολογίζεται το ποσοστό των παιχνιδιών που προηγήθηκαν, της φιλοξενούμενης ομάδας εκτός έδρας και τα τέρματα που σημειώθηκαν είναι περισσότερα των δύο.

11. Μέσος όρος τερμάτων που πέτυχε η γηπεδούχος ομάδα στην έδρα της στους τελευταίους τρείς εντός έδρας αγώνες της.

Υπολογίζεται ο μέσος όρος τερμάτων που πέτυχε η γηπεδούχος ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εντός έδρας παιχνίδια της.

12. Ποσοστό των παιχνιδιών που σκόραρε η γηπεδούχος ομάδα στην έδρα της στους τελευταίους τρείς εντός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών που έχει σκοράρει η γηπεδούχος ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εντός έδρας παιχνίδια της.

13. Μέσος όρος τερμάτων που δέχτηκε η γηπεδούχος ομάδα στην έδρα της στους τελευταίους τρείς εντός έδρας αγώνες της.

Υπολογίζεται ο μέσος όρος τερμάτων που δέχτηκε η γηπεδούχος ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εντός έδρας παιχνίδια της.

14. Ποσοστό των παιχνιδιών που δέχτηκε τέρμα η γηπεδούχος ομάδα στους τελευταίους τρείς εντός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών που δέχτηκε τέρμα η γηπεδούχος ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εντός έδρας παιχνίδια της.

15. Μέσος όρος τερμάτων που πέτυχε η φιλοξενούμενη ομάδα εκτός έδρας στους τελευταίους τρείς εκτός έδρας αγώνες της.

Υπολογίζεται ο μέσος όρος τερμάτων που πέτυχε η φιλοξενούμενη ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εκτός έδρας παιχνίδια της.

16. Ποσοστό των παιχνιδιών που σκόραρε η φιλοξενούμενη ομάδα εκτός έδρας στους τελευταίους τρείς εκτός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών που σκόραρε η φιλοξενούμενη ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εκτός έδρας παιχνίδια της.

17. Μέσος όρος τερμάτων που δέχτηκε η φιλοξενούμενη ομάδα εκτός έδρας στους τελευταίους τρείς εκτός έδρας αγώνες της.

Υπολογίζεται ο μέσος όρος τερμάτων που δέχτηκε η φιλοξενούμενη ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εκτός έδρας παιχνίδια της.

18. Ποσοστό των παιχνιδιών που δέχτηκε τέρμα η φιλοξενούμενη ομάδα εκτός έδρας στους τελευταίους τρείς εκτός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών που δέχτηκε τέρμα η φιλοξενούμενη ομάδα στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εκτός έδρας παιχνίδια της.

19. Ποσοστό των παιχνιδιών της γηπεδούχου ομάδας στην έδρα της όπου έχουν σημειωθεί περισσότερα των δύο τερμάτων στους τελευταίους τρείς εντός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών της γηπεδούχου ομάδας στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εντός έδρας παιχνίδια της και τα τέρματα που σημειώθηκαν είναι περισσότερα των δύο.

20. Ποσοστό των παιχνιδιών της φιλοξενούμενης ομάδας εκτός έδρας όπου έχουν σημειωθεί περισσότερα των δύο τερμάτων στους τελευταίους τρείς εκτός έδρας αγώνες της.

Υπολογίζεται το ποσοστό των παιχνιδιών της φιλοξενούμενης ομάδας της στα τρία άμεσα προηγηθέντα του συγκεκριμένου, εκτός έδρας παιχνίδια της και τα τέρματα που σημειώθηκαν είναι περισσότερα των δύο.

21. Μέσος όρος τερμάτων που επιτεύχθηκαν στα παιχνίδια που αγωνίστηκε η γηπεδούχος ομάδα.

Υπολογίζεται ο μέσος όρος τερμάτων που επιτεύχθηκαν σε παιχνίδια στα οποία συμμετείχε η γηπεδούχος ομάδα από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

22. Μέσος όρος τερμάτων που επιτεύχθηκαν στα παιχνίδια που αγωνίστηκε η φιλοξενούμενη ομάδα.

Υπολογίζεται ο μέσος όρος τερμάτων που επιτεύχθηκαν σε παιχνίδια στα οποία συμμετείχε η φιλοξενούμενη ομάδα από την αρχή του πρωταθλήματος μέχρι τη συγκεκριμένη στιγμή.

23. Ποσοστό των παιχνιδιών στα οποία αγωνίστηκε η γηπεδούχος ομάδα και στα οποία σημειώθηκαν πέραν των δύο τερμάτων.

Υπολογίζεται το ποσοστό των παιχνιδιών στα οποία η γηπεδούχος ομάδα συμμετείχε και ο αριθμός των τερμάτων που σημειώθηκαν υπερβαίνουν τα δύο.

24. Ποσοστό των παιχνιδιών στα οποία αγωνίστηκε η φιλοξενούμενη ομάδα και στα οποία σημειώθηκαν πέραν των δύο τερμάτων.

Υπολογίζεται το ποσοστό των παιχνιδιών στα οποία η φιλοξενούμενη ομάδα συμμετείχε και ο αριθμός των τερμάτων που σημειώθηκαν υπερβαίνουν τα δύο.

25. Ποσοστό των παιχνιδιών στα οποία αγωνίστηκε η γηπεδούχος ομάδα και στα οποία σημειώθηκαν πέραν των δύο τερμάτων στους τελευταίους έξι αγώνες της.

Υπολογίζεται το ποσοστό των τελευταίων έξι παιχνιδιών της γηπεδούχου ομάδας στα οποία ο αριθμός των τερμάτων που σημειώθηκαν υπερβαίνουν τα δύο.

26. Ποσοστό των παιχνιδιών στα οποία αγωνίστηκε η φιλοξενούμενη ομάδα και στα οποία σημειώθηκαν πέραν των δύο τερμάτων στους τελευταίους έξι αγώνες της.

Υπολογίζεται το ποσοστό των τελευταίων έξι παιχνιδιών της φιλοξενούμενης ομάδας στα οποία ο αριθμός των τερμάτων που σημειώθηκαν υπερβαίνουν τα δύο.

Από αυτά τα στατιστικά λαμβάνονται τα μισά για την μία ομάδα και τα άλλα μισά για την άλλη. Δηλαδή αν σε ένα αγώνα λαμβάνουν μέρος οι ομάδα Α αγωνιζόμενη στην έδρα της, εναντίον της ομάδας Β η οποία είναι η εκτός έδρας, τα στατιστικά που αναφέρονται για την γηπεδούχο ομάδα θα τα πάρουμε από τον πίνακα στατιστικών της Α ενώ αυτά που αντιστοιχούν στην φιλοξενούμενη από αυτόν της Β.

3.2 Το πρωτάθλημα της Serie A

Το πρωτάθλημα της Ιταλίας αποτελεί ένα ανταγωνιστικότατο πρωτάθλημα με αρκετές μεγάλες ομάδες της Ευρώπης να αποτελούν μέρος του. Πολλοί είναι αυτοί που το παρακολουθούν φανατικά αφού συνδυάζει την υψηλή τεχνική, τη δύναμη αλλά κυρίως την τακτική μέσα στο παιχνίδι. Αναλογικά επομένως είναι αρκετοί και αυτοί που ποντάρουν σε αγώνες του συγκεκριμένου πρωταθλήματος και επιθυμούν να εισπράξουν μέσω αυτού χρήματα. Έτσι, σκεπτόμενος τα πιο πάνω αποφάσισα να ασχοληθώ με το πρωτάθλημα της πρώτης κατηγορίας της Ιταλίας ελπίζοντας σε καλά αποτελέσματα.

Στη διοργάνωση αυτή συμμετέχουν κάθε χρόνο 20 ομάδες. Δηλαδή σε κάθε αγωνιστική διεξάγονται 10 παιχνίδια και η κάθε ομάδα αγωνίζεται με κάθε άλλη μία φορά εντός και μια εκτός έδρας. Άρα συνολικά για κάθε περίοδο έχουμε 380 παιχνίδια. Για να τις μετρικές μου όμως χρειάζομαι να γνωρίζω τι έγινε στα έξι προηγούμενα παιχνίδια των ομάδων έτσι οι πρώτες έξι αγωνιστικές της κάθε χρονιάς δεν μπαίνουν ως δεδομένα εκπαίδευσης. Τελικώς λοιπόν για κάθε ποδοσφαιρική περίοδο λαμβάνονται ως είσοδοι 320 παιχνίδια.

Η εικόνα της ιστοσελίδας www.betexplorer.com στα αποτελέσματα προηγούμενων αγώνων μοιάζει κάπως έτσι:

Serie A 2015/2016

Summary | **Results** | Fixtures | League stats | Head-to-head | Archive

36. Round		1	X	2	
Genoa - AS Roma	2:3	5.27	3.97	1.63	02.05.2016
Napoli - Atalanta	2:1	1.14	8.25	17.81	02.05.2016
AC Milan - Frosinone	3:3	1.25	6.09	11.10	01.05.2016
Empoli - Bologna	0:0	2.37	3.23	3.08	01.05.2016
Juventus - Carpi	2:0	1.29	5.22	11.41	01.05.2016
Lazio - Inter	2:0	2.22	3.37	3.25	01.05.2016
Palermo - Sampdoria	2:0	1.89	3.62	3.96	01.05.2016
Sassuolo - Verona	1:0	1.50	4.31	6.39	01.05.2016
Chievo - Fiorentina	0:0	4.30	3.50	1.86	30.04.2016
Udinese - Torino	1:5	1.95	3.43	3.97	30.04.2016

3.3 Κανονικοποίηση τιμών

Τα νευρωνικά δίκτυα μπορούν να έχουν ως είσοδο οποιανδήποτε τιμή. Όμως για την λήψη καλύτερων αποτελεσμάτων επιλέγονται συνήθως τιμές κανονικοποιημένες από το 0 ως και το 1. Όπως γίνετε αντιληπτό στο δικό μου πρόβλημα υπάρχουν τιμές μεγαλύτερες του 1 αφού πολλά από τα στατιστικά αποτελούν μέσο όρο τερμάτων που πιθανώς να υπερβαίνουν το 1. Έτσι χρησιμοποιήθηκε για την κανονικοποίηση των τιμών η πιο κάτω εξίσωση:

$$y = \frac{X - X_{min}}{X_{max} - X_{min}}$$

X: αρχική τιμή πριν την κανονικοποίηση.

X_{max}: η μεγαλύτερη πιθανή τιμή του συγκεκριμένου στατιστικού πριν την κανονικοποίηση.

X_{max}: η μικρότερη πιθανή τιμή του συγκεκριμένου στατιστικού πριν την κανονικοποίηση.

y: η νέα τιμή του στατιστικού.

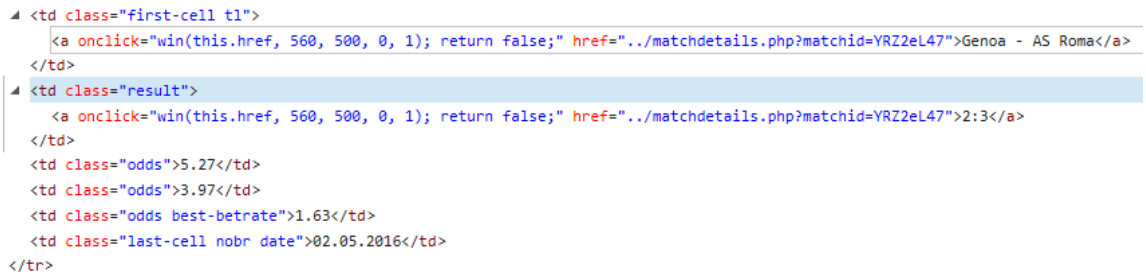
Αφού λοιπόν κανονικοποιηθούν οι τιμές αυτές η προ-επεξεργασία έχει τελειώσει και τα στατιστικά αυτά γράφονται σε ένα δεύτερο αρχείο με το όνομα inputs.txt για να το δεχτεί το RNN ούτως ώστε να εκπαιδευτεί.

3.4 Λήψη δεδομένων

Για να ληφθούν τα δεδομένα εισόδου χρησιμοποιείται ένας Web Crawler γραμμένος στην php ο οποίος λαμβάνει ως ορίσματα το όνομα του πρωταθλήματος καθώς και τις χρονολογίες που μας ενδιαφέρουν. Το πρώτο πράγμα που γίνετε για τη συλλογή των δεδομένων για τα συγκεκριμένα ορίσματα είναι η λήψη του document tree της ιστοσελίδας που δημιουργήθηκε σύμφωνα με τα ορίσματα που έχει δεχθεί το πρόγραμμα.

```
▲ <html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml" xmlns:fb="http://ogp.me/ns/fb#"
  ▷ <head>…</head>
  ▲ <body>
    ▷ <div class=" fb_reset" id="fb-root">…</div>
    ▷ <script>…</script>
    ▷ <div class="adsenvelope adstextvpad banx-top" id="lsadvert-zid-150" style="width: 7
      <div class="adsclear"></div>
    ▲ <div id="be">
      ▲ <div id="screen">
        ▲ <div id="bbscreen">
          ▲ <div class="dcl" id="content">
            ▷ <div id="panel">…</div>
            ▷ <div id="slink">…</div>
            ▲ <div id="main">
              ▲ <div id="ml">
                ▲ <div class="mcontent">
                  ▲ <div class="inner">
                    ▷ <h1>…</h1>
                    ▷ <ul class="localmenu localmenu-nobb">…</ul>
                    ▲ <div id="leagueresults_div">
                      ▲ <table class="result-table league-results" cellspacing="0">
                        ▷ <tbody id="leagueresults_tbody">…</tbody>
                        </table>
                      </div>
                    ▷ <table class="blank" border="0">…</table>
                    ▷ <div id="fadvert">…</div>
                    </div>
                  </div>
                </div>
              ▷ <script type="text/javascript">…</script>
              </div>
            ▷ <div id="menu">…</div>
              <div id="footer"></div>
            ▷ <div id="social">…</div>
            ▷ <script type="text/javascript">…</script>
            ▷ <div id="myselections">…</div>
            ▷ <script type="text/javascript">…</script>
            </div>
          </div>
        ▷ <div id="ufooter">…</div>
        </div>
      </div>
      <div class="scrolling-banner-bottom-stop"></div>
```

Μετά ψάχνει και βρίσκει στα στοιχεία του δέντρου αυτού τα δεδομένα που μας ενδιαφέρουν, δηλαδή τις διαγωνιζόμενες ομάδες και το τελικό σκορ για τον κάθε αγώνα.



Αμέσως μετά την εύρεση τους, τα δεδομένα αυτά γράφονται σε ένα αρχείο κειμένου ανά ποδοσφαιρική περίοδο με την πιο κάτω μορφή:

Ομάδα Α – Ομάδα Β – 0:0

Τελικώς λοιπόν για κάθε χρονιά έχουμε ένα αρχείο δεδομένων με όλα τα παιχνίδια και όλα τα σκορ.

Επίσης αν επιλεγεί η τελευταία χρονιά ως όρισμα στον Crawler τότε τα δεδομένα που συλλέγονται είναι όλες οι προηγούμενες αγωνιστικές, για τις οποίες έχουμε και τα αποτελέσματα, καθώς και η ερχόμενη αγωνιστική για την οποία τα αποτελέσματα προφανώς λείπουν. Για αυτά επομένως τα αποτελέσματα θα προσπαθήσει το πρόγραμμα να βρει την τελική τους κατάληξη, το αν δηλαδή θα είναι under ή over.

3.5 Προ-επεξεργασία δεδομένων

Αφού υπάρχουν σε αρχεία οι αγώνες με τα αποτελέσματα της κάθε χρονιάς για το πρωτάθλημα που έχει επιλεγεί, η επόμενη κίνηση που πρέπει να γίνει είναι να δημιουργηθούν για κάθε ομάδα τα στατιστικά της. Τα στατιστικά αυτά θα ανανεώνονται με την πάροδο των παιχνιδιών και αυτό γιατί θα αλλάζουν τα δεδομένα που θα λαμβάνονται υπόψιν για την οριστικοποίηση της τιμής τους.

Έτσι ένα πρόγραμμα γραμμένο στην Java, διαβάζει τα δεδομένα κάθε χρονιάς από τα αρχεία αυτά με σκοπό να δημιουργήσει τα στατιστικά για κάθε αγώνα ξεχωριστά. Πρώτη κίνηση είναι η δημιουργία μιας λίστας(λίστα omades, παράρτημα Β) στην οποία περιέχονται όλες οι διαγωνιζόμενες στο πρωτάθλημα ομάδες. Έπειτα δημιουργούνται οι δύο κλάσεις agwnes.java και apotelesmata.java όπου σε αυτές αντίστοιχα υπάρχει ένας πίνακας(πίνακας table_m και table_s, Παράρτημα Β). Ο πρώτος αποθηκεύει για μια ομάδα τη σειρά των παιχνιδιών που αγωνίστηκε στο πρωτάθλημα και ο δεύτερος τη σειρά των αποτελεσμάτων που έφερε μια ομάδα. Οι δύο αυτοί πίνακες είναι παράλληλοι πίνακες, δηλαδή στην ίδια θέση των πινάκων αποθηκεύονται αντίστοιχα δεδομένα. Για παράδειγμα αν στην πρώτη θέση του πίνακα των αγώνων υπάρχει το παιχνίδι μεταξύ της ομάδας Α και της ομάδας Β τότε στην πρώτη θέση του πίνακα των αποτελεσμάτων αποθηκεύεται το τελικό αποτέλεσμα της συγκεκριμένης αναμέτρησης.

Μετά δημιουργούνται δύο πίνακες(πίνακες team_matches και team_scores, Παράρτημα Β) στους οποίους περιέχονται στον ένα αντικείμενα της κλάσης agwnes.java και στον άλλο αντικείμενα της apotelesmata.java. Αυτοί οι δύο πίνακες είναι παράλληλοι μαζί με τον πίνακα omades, στον οποίο υπάρχουν οι ομάδες του πρωταθλήματος ούτως ώστε να μπορεί να γίνει η αντιστοίχιση και να είναι γνωστά τα παιχνίδια της κάθε ομάδας αλλά και τα αποτελέσματα της.

Από εκεί και πέρα η γενική τακτική που χρησιμοποιείται είναι μέσω της κλάσης pososta.java να δημιουργείται για κάθε ομάδα ένας δυσδιάστατος πίνακας(πίνακας table_p, Παράρτημα Β) με γραμμές όσες οι αγωνιστικές του πρωταθλήματος και στήλες όσες και τα στατιστικά. Εν προκειμένω 32Χ26. Οι γραμμές του πίνακα είναι κατά έξι μειωμένες αφού σύμφωνα με τον τύπο των μετρικών, χρειάζεται να περάσουν

τουλάχιστον έξι αγωνιστικές για να μπορούν να δημιουργηθούν. Έτσι ξεκινά ο υπολογισμός για την κάθε μια τιμή των στατιστικών από την έβδομη αγωνιστική και τελειώνει την τελευταία. Για να αποθηκευτούν στατιστικά αυτά για την κάθε αγωνιστική της κάθε ομάδας, δημιουργείται ένας άλλος πίνακας(πίνακας pososta, Παράρτημα Β) ο οποίος περιέχει αντικείμενα της κλάσης pososta.java. Δηλαδή έχουμε για κάθε ομάδα, για μία αγωνιστική περίοδο τις 32 σειρές με τα στατιστικά της. Άρα για όλες τις διαγωνιζόμενες ομάδες 320.

Άρα τελικώς έχουμε την αντιστοιχία μεταξύ της κάθε ομάδας και των στατιστικών της σε παράλληλους πίνακες. Το τελευταίο πράγμα που μένει λοιπόν, είναι για κάθε ένα από παιχνίδια του πρωταθλήματος να δημιουργηθεί η αλυσίδα στατιστικών. Αυτά λαμβάνονται συνδυάζοντας τα στατιστικά που έχουν οι δύο διαγωνιζόμενες ομάδες για τη συγκεκριμένη αγωνιστική. Στο τέλος της κάθε γραμμής αυτής προθέτεται επίσης ο χαρακτήρας + ή – για more ή less αντίστοιχα για να λειτουργήσει ως η επιθυμητή έξοδος του συστήματος.

Η κάθε μια γραμμή καταγράφεται στο αρχείο inputs.txt το οποίο το αρχείο μπορεί να χρησιμοποιηθεί αυτούσιο με σκοπό την εκπαίδευση του δικτύου είτε να χωριστεί σε δεδομένα εισόδου και δεδομένα ελέγχου.

Αν τώρα επιλεγεί η τρέχουσα περίοδος για λήψη στατιστικών, αυτόματα δημιουργείται και ένα άλλο, νέο αρχείο με το όνομα next_fixtures.txt το οποίο περιέχει τις μετρικές για τα παιχνίδια τα οποία περιέχονται στην ερχόμενη αγωνιστική για το πρωτάθλημα της Ιταλίας, 10 στον αριθμό, καθώς και το αρχείο matches.txt το οποίο περιέχει με την ίδια σειρά, τα ονόματα των διαγωνιζόμενων ομάδων για τα στατιστικά που περιέχονται στο next_fixtures.txt ούτως ώστε να μπορεί να γίνει η αντιστοιχία με τα αποτελέσματα του RNN δικτύου.

Για δεδομένα εισόδου και δεδομένα ελέγχου όμως μπορούν να χρησιμοποιηθούν οποιεσδήποτε σειρές στατιστικών είναι αρεστές. Για τον έλεγχο του προγράμματος μου χρησιμοποίησα μια ολόκληρη χρονιά με σκοπό να βρεθεί ένα ποσοστό που θα είναι αξιόπιστο. Όμως ως δεδομένα ελέγχου τοποθετήθηκαν επίσης και τα παιχνίδια της

επόμενης αγωνιστικής για να βρεθεί και το πιθανό κέρδος μετά από τοποθέτηση χρημάτων στο στοίχημα.

3.6 Δίκτυα RNN

Αφού τώρα υπάρχουν τα στατιστικά για διάφορα παιχνίδια γραμμένα σε αρχείο το μόνο που μένει είναι η εκπαίδευση του δικτύου RNN. Το πρόγραμμα αυτό είναι υλοποιημένο στη γλώσσα Python. Η γλώσσα Python αποτελεί μια πολύ σημαντική γλώσσα προγραμματισμού με πολλούς προγραμματιστές να την προτιμούν διακρίνοντας την έναντι των άλλων για το συμπαγή, περιεκτικό και εύχρηστο κώδικα της αλλά και για τις πολλές βιβλιοθήκες που τη συνοδεύουν και κάνουν το έργο του προγραμματιστή ευκολότερο.

Η γενικότερη ιδέα πίσω από την υλοποίηση είναι αφού υπάρχουν σειρές τιμών για τα παιχνίδια ποδοσφαίρου, αυτές να εισάγονται στο δίκτυο και αυτό να εκπαιδεύεται μέσω των βαρών των κρυφών νευρώνων που σιγά σιγά θα δημιουργήσουν μνήμη για το δίκτυο, η οποία θα συνυπολογίζεται με την είσοδο ούτως ώστε να βγαίνει το αποτέλεσμα της κάθε εισόδου. Η αναπόληση όμως έχει να κάνει με την όσο πιο καλή ταύτιση προηγούμενων εισόδων και της παρούσης, άρα για να γνωρίζει τι θα απαντήσει σε κάθε είσοδο το δίκτυο θα πρέπει να έχει δει μια παρόμοια είσοδο.

3.6.1 Υλοποίηση δικτύων RNN(Through Time)

Η υλοποίηση αυτή έγινε με σκοπό τη δημιουργία και αναγνώριση προτύπων. Δηλαδή για κάθε είσοδο το δίκτυο θα ψάχνει να βρει συνδέσεις μεταξύ των δεδομένων εισόδου, θα δημιουργεί πρότυπα και τέλος θα ψάχνει να βρει τέτοια κατά την είσοδο δεδομένων ελέγχου.

Το πρώτο-πρώτο πράγμα που γίνεται είναι η αρχικοποίηση του δικτύου σύμφωνα με τις εισόδους-παραμέτρους που μπαίνουν σε αυτό. Υπάρχουν πολλές παράμετροι στο σύστημα οι οποίες μπορούν να το τροποποιήσουν όπως ο αριθμός νευρώνων που υπάρχουν σε κάθε επίπεδο, ρυθμός μάθησης και οι επαναλήψεις εκπαίδευσης του δικτύου.

Αρχικά δημιουργούνται οι πίνακες με τα βάρη των συνδέσεων μεταξύ νευρώνων. Κατά σειρά δημιουργούνται(συνάρτηση init της κλάσης RNN, Παράρτημα Γ)

-Ο πίνακας των βαρών των νευρώνων εισόδου, 26 στον αριθμό όσες και οι τιμές των στατιστικών, και των νευρώνων του κρυφού επιπέδου(πίνακας W_uh, Παράρτημα Γ) έχουν μέγεθος 26X20, αφού ο αριθμός των νευρώνων του κρυφού επιπέδου είναι 20.

-Ο πίνακας που περιέχει τις συνδέσεις μεταξύ των κρυφών νευρώνων που επιστρέφουν σαν είσοδος στους ίδιους(πίνακας W_hh, Παράρτημα Γ) μεγέθους 20X20

-Ο πίνακας για την ένωση των νευρώνων εισόδου με τους νευρώνες εξόδου(πίνακας W_hy, Παράρτημα Γ) μεγέθους 20X1, γιατί ένας είναι ο νευρώνας εισόδου.

-Ο πίνακας με το κατώφλι για τον κάθε νευρώνα στο κρυφό επίπεδο μεγέθους 20X1

-Ο πίνακας με το κατώφλι για τον νευρώνα στο επίπεδο εξόδου με μια τιμή

Αφού γίνουν οι αρχικοποιήσεις αυτές ξεκινά να διαβάζεται το αρχείο εκπαίδευσης γραμμή-γραμμή. Επιθυμητό θα ήταν να παρουσιάζετε ως αρχείο εκπαίδευσης το input.txt και ως αρχείο ελέγχου το next_fixture.txt αρχείο το οποίο παράγεται από το πρόγραμμα της Java όπως προαναφέρθηκε. Σε αυτό το σημείο έχω τοποθετήσει

αρχικώς για αρχείο ελέγχου μια ολόκληρη ποδοσφαιρική χρονιά(2014-2015) ούτως ώστε να έχω ένα μεγαλύτερο σύνολο μέσω του οποίου θα εξάγω πιο ασφαλή συμπεράσματα για την επίδοση του δικτύου.

Τώρα η κάθε γραμμή εισόδου αποστέλνεται μαζί με την επιθυμητή έξοδο στο RNN δίκτυο. Το πρώτο πράγμα που γίνετε είναι η τοποθέτηση της εισόδου στους 26 νευρώνες εισόδου. Έπειτα γίνεται ο υπολογισμός της τιμής εξόδου των κρυφών νευρώνων (συνάρτηση recurrent_fn, Παράρτημα Γ) με τη πράξη

$$h = sigmoid(hidden_{out}[-1].W\_hh \ + \ u(t).W\_uh \ + \ b\_hh)$$

Όπου η συνάρτηση ενεργοποίησης είναι η σιγμοειδής και το $hidden_{out}$[-1] η προηγούμενη τιμή των εξόδων του κρυφού νευρώνα. Αυτό γιατί θέλουμε οι νευρώνες που βρίσκονται στο κρυφό επίπεδο να συνυπολογίζουν τις εξόδους των κρυφών νευρώνων κατά την προηγούμενη είσοδο επί τα βάρη που συνδέουν τον κάθε ένα με τον κάθε άλλο.

Έπειτα υπολογίζεται η τιμή του νευρώνα εξόδου με την εξίσωση

$$y = hidden_{out}.W\_h + \text{b\_hy}$$

Από αυτή την τιμή υπολογίζεται έπειτα και το κόστος με το Mean Square Error και την εξίσωση

$$cost = (t - y)^2$$

Το t είναι η επιθυμητή έξοδος ενώ το y η πραγματική.

Βρίσκοντας επομένως την τιμή του κόστους, μπορούμε να βασιστούμε σε αυτή για να βρεθεί η αλλαγή που πρέπει να γίνει στα βάρη. Ο τρόπος που εντοπίζεται η τιμή της αλλαγής των νέων βαρών είναι με την μέθοδο κατάβασης κλίσης, η οποία ως βάση της έχει τον υπολογισμό της παραγώγου του σφάλματος(κόστους) ως προς το κάθε βάρος. Στη python υπάρχει υλοποιημένη η συνάρτηση grad η οποία υπολογίζει την παράγωγο αυτή και ως ορίσματα δέχεται το κόστος της εν λόγω εισόδου προς το δίκτυο, και τα παρόντα βάρη. Αυτά είναι τα βάρη από τους νευρώνες εισόδου προς τους κρυφούς νευρώνες (W_uh), τα βάρη που ενώνουν τους κρυφούς νευρώνες με τους εαυτούς

τους(W_hh), τα βάρη από τους κρυφούς νευρώνες προς τους νευρώνες εξόδου(W_hy) και τέλος τα κατώφλια των κρυφών νευρώνων(b_hh) και της εξόδου(b_hy). Για να βρεθεί η νέα τιμή των κάθε νευρώνων, μέσω της κατάβασης κλίσης πρέπει να εντυπωθεί το πόσο ρόλο έπαιξε ο κάθε νευρώνας και το κάθε βάρος του δικτύου για την έξοδο.

Για να βρούμε το ποσό της μεταβολής του κάθε βάρους ξεκινάμε από τους νευρώνες εξόδου και ερχόμαστε πίσω-πίσω μέχρις ότου να φτάσουμε στους νευρώνες εισόδου δηλαδή χρησιμοποιώντας τον Back Propagation. Το σφάλμα των νευρώνων εξόδου υπολογίζεται βάση της διαφοράς της επιθυμητής εξόδου με την πραγματική.

Από ΄κεί και πέρα το σφάλμα κάθε σύνδεσης συνυπολογίζεται βάση της εξόδου που έχει βγάλει ο νευρώνας από όπου ξεκινά η σύνδεση, αφού μας ενδιαφέρει αν συνείσφερε ή όχι στο τελικό αποτέλεσμα, το σφάλμα που έρχεται από τον νευρώνα στον οποίο καταλήγει το βάρος καθώς τη τιμή που το βάρος κατείχε μέχρι εκείνη τη στιγμή. Η τιμή που βρέθηκε από την εξίσωση αυτή αφαιρείται από το υπάρχον βάρος ούτως ώστε αυτό να πάρει τη νέα του τιμή.

Με τον τρόπο αυτό τα βάρη προσπαθούν να βρουν τις κατάλληλες τιμές σύμφωνα με τις οποίες τα δεδομένα θα παίρνουν μια μορφή ομαδοποίησης και το σύστημα θα μπορεί να γενικεύσει. Με τον όρο γενίκευση εννοούμε ότι θα μπορεί όταν δεχθεί μια νέα, άγνωστη είσοδο να αποφασίσει σωστά για το ποια θα πρέπει είναι η έξοδος.

Η τακτική που ακολουθείται για την αλλαγή των βαρών είναι Back Propagation Through Time κατά την οποία τα βάρη αλλάζουν μόνο στο τέλος κάθε εποχής. Επομένως για κάθε είσοδο του το δίκτυο φυλάει το κόστος της και με το πέρας της κάθε εποχής συσσωρεύει το σφάλμα και εκτελεί την αλλαγή των βαρών του δικτύου βασισμένο σε αυτή την τιμή. Μόλις εκτελεστεί η διαδικασία αυτή ξεκινά ξανά η εισαγωγή στο σύστημα των δεδομένων εκπαίδευσης και αυτό γίνεται για όσες εποχές επιθυμούμε.

Η πιο πάνω περιγραφή του γενικού δικτύου υλοποιήθηκε με τρείς διαφορετικούς τρόπους. Σε κάθε προσπάθεια έμενε σταθερός ο αριθμός νευρώνων εισόδου ο οποίος

ήταν όσος και τα στατιστικά του κάθε αγώνα, δηλαδή 26 ενώ ο αριθμός των κρυφών επιπέδων και των νευρώνων εξόδου άλλαζαν σε κάθε περίσταση.

Ένα πολύ σημαντικό κομμάτι των νευρωνικών δικτύων είναι ο τρόπος αναπαράστασης των δεδομένων στο δίκτυο. Για αυτό και εγώ υλοποίησα αυτά τα τρία δίκτυα ελπίζοντας να εντοπίσω το καταλληλότερο για την δική μου περίσταση.

Ο πρώτος λοιπόν, τρόπος υλοποίησης είναι με ένα επίπεδο νευρώνων και μια έξοδο. Η επιθυμητή έξοδος του δικτύου είναι 0 για more και 1 για less. Έτσι αν το δίκτυο έβγαζε αποτέλεσμα κοντά στο μηδέν σήμαινε ότι το δίκτυο, για τη συγκεκριμένη είσοδο υπολογίζει την κατάληξη του παιχνιδιού ως more, ενώ αν η τιμή ήταν πιο κοντά στο ένα αυτό θα σήμαινε πως το υπολογιζόμενο αποτέλεσμα θα ήταν less.

Η δεύτερη προσπάθεια ήταν με δύο νευρώνες στην έξοδο όπου για τον πρώτο νευρώνα επιθυμητό θα ήταν να βγάζει 0 στα παιχνίδια που η κατάληξη τους ήταν less και ο δεύτερος για παιχνίδια που θα ήταν more. Άρα για να αποφασιστεί ποια θα ήταν η απάντηση του δικτύου για κάθε είσοδο έπαιρνα την μεγαλύτερη τιμή των δύο νευρώνων ως έξοδο του δικτύου.

Η τρίτη προσπάθεια ήταν μια προσπάθεια για υλοποίηση ενός deep network. Έτσι πρόσθεσα ένα επιπλέον επίπεδο νευρώνων στο κρυφό επίπεδο. Άρα οι ήδη υπάρχοντες νευρώνες στο κρυφό επίπεδο αντί να ενώνονται κατευθείαν με τον νευρώνα εξόδου ενώνονται, ο κάθε ένας, με τους νευρώνες του δεύτερου κρυφού επιπέδου και αυτοί είναι που ενώνονται με το νευρώνα εξόδου. Η προσπάθεια αυτή έγινε με στόχο να βρεθούν πιθανές επιπλέον συνδέσεις μεταξύ των δεδομένων εισόδου. Και αν αυτοί οι συνδυασμοί εντοπίζονταν και στα δεδομένα ελέγχου τότε θα λαμβάνονται και καλύτερα αποτελέσματα.

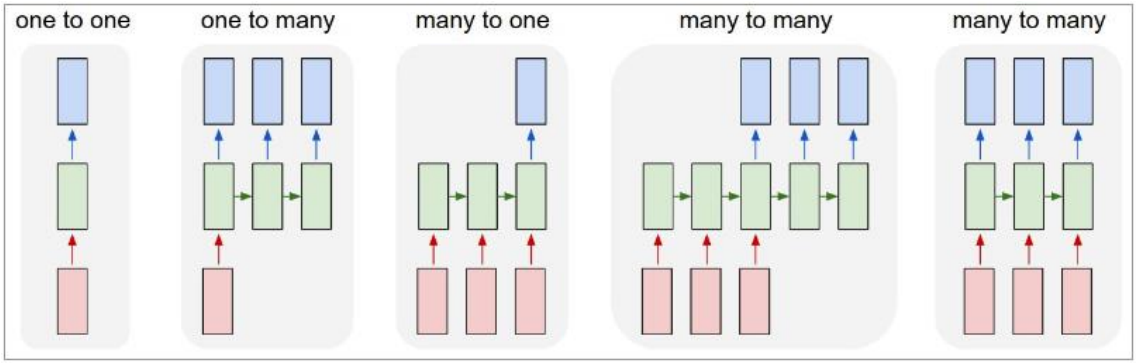
Για το κάθε ένα από τα πιο πάνω δίκτυα έχω αλλάξει πολλές φορές παραμέτρους, όπως ο αριθμός νευρώνων στο κρυφό επίπεδο, το ρυθμό μάθησης και τις αρχικές τιμές των βαρών. Όλα όμως τα δίκτυα έδειξαν να λαμβάνουν μερικώς ικανοποιητικά αποτελέσματα για τα δεδομένα εισόδου και το κάθε ένα τα δικά του αποτελέσματα για τα δεδομένα ελέγχου τα οποία θα συζητηθούν στο επόμενο κεφάλαιο.

3.6.2 Υλοποίηση δικτύων RNN για λέξης-παιχνίδια

Βλέποντας τις αδυναμίες των πιο πάνω δικτύων και σκεπτόμενος ότι η επιδίωξη μου είναι να υπολογίσω αυτόνομα τον κάθε αγώνα, με στατιστικά του κάθε αγώνα που μέσω των δεδομένων εισόδου περιγράφονται, πήγα προς τη μεριά της επεξεργασίας κειμένου. Σκέφτηκα λοιπόν να θεωρήσω τον κάθε αγώνα ως λέξη και να αφήσω το δίκτυο μου να αποφασίσει για την κάθε λέξη-αγώνα το επόμενο γράμμα-σημείο.

Το πρώτο πράγμα που έπρεπε να γίνει είναι να αποφασιστεί η αρχιτεκτονική του δικτύου. Πρώτα από όλα ήθελα μια σύνδεση νευρώνων που θα μπορεί να συνδυάσει όλη την ακολουθία από τα γράμματα-στατιστικά μαζί με την επιθυμητή έξοδο. Για να αποφασίσω επομένως τη δομή του δικτύου μου μελέτησα τις πιθανούς τρόπους που αυτό μπορεί να δομηθεί ούτως ώστε να διαλέξω τον πιο ταιριαστό.

Στο σχήμα 3.1 φαίνονται τα είδη των δικτύων από τα οποία μπορώ να επιλέξω το καταλληλότερο.



Εικόνα 3.1

Ο κάθε τρόπος σύνδεσης ταιριάζει σε διαφορετικά προβλήματα. Ας πούμε το one to many ταιριάζει για να δέχεται μια εικόνα και να αποφασίζει για το ποια αντικείμενα υπάρχουν εντός της ή η many to many η οποία μπορεί να λύσει προβλήματα όπου εισάγεται μια πρόταση σε μια γλώσσα και εξάγεται η μετάφραση της σε μια άλλη.

Ο τρόπος που είναι κτισμένη η αρχιτεκτονική του δικτύου έδωσε να καταλάβω πως ο τρόπος που το δίκτυο αντιλαμβάνεται τον συνδυασμό εισόδου-εξόδου είναι ως το συνδυασμό, και κατά κάποιο τρόπο τη συμβολοσειρά, που κατέρχεται μια είσοδος και η

αντιστοίχηση της ολοκλήρωσης της να σηματοδοτεί την εκάστοτε επιθυμητή έξοδο. Είναι επομένως αυτή η σειρά και αποπεράτωση της που λαμβάνεται υπόψη για να αντιληφθεί το δίκτυο κατά που να κινηθεί, προς τη μία κατεύθυνση(more) ή προς την άλλη(less). Έτσι αν το δίκτυο έχει δει ξανά ένα κομμάτι της εισόδου σε ένα συγκεκριμένο σημείο τότε κατευθύνει την απόκριση του προς τη μία ή την άλλη πλευρά. Ένα τέτοιο παράδειγμα υλοποίησης έγινε για την ταξινόμηση σχολίων που αφορούσαν κριτικές ταινιών, με τα RNN να δείχνουν μεγάλη ικανότητα μάθησης που φτάνει το 83%[26]

Το να θεωρήσει κάποιος μια είσοδο ως χρονική συμβολοσειρά είναι ένα θεωρητικό κομμάτι του πως βλέπει κανείς τον συνδυασμό παρελθόντος και παρόντος χρόνου. Δηλαδή ο τρόπος που σκέφτομαι εγώ μια συμβολοσειρά είναι ο εξής. ‘‘Μετά’’ από κάποιες καταστάσεις (οι οποίες παρουσιάζονται κατά σειρά) είναι πιθανό να συμβεί μια επόμενη. Δηλαδή μετά από το χαμηλό βαρομετρικό είναι πιθανόν να συμβεί βροχόπτωση, μετά από μια καλή προετοιμασία της ομάδας (πολλά λαμβάνονται υπόψη με τον όρο προετοιμασία αλλά όλα αυτά συνοψίζονται στο παρελθόν) πιθανόν να παρουσιαστεί καλή απόδοση, μετά από έναν συνδυασμό ικανοποιητικού ύπνου, αρκετού διαβάσματος και ικανότητας σε ένα μάθημα πιθανός να γράψεις καλά, και φτάνοντας στα καθ΄ ημάς, μετά από ένα συνδυασμό στατιστικών πιθανόν να εμφανιστεί η εξής κατάσταση. Όμως πως ορίζεται το ‘‘πριν’’ και το ‘‘μετά’’; Γιατί να θεωρούνται ως σειρά τα γεγονότα αυτά και όχι ως καταστάσεις που μπορούν να συμβαίνουν μαζί;

Το θέμα-πρόβλημα επομένως είναι η μη συμβατική σειρά που έρχονται τα στατιστικά. Δηλαδή το ότι το κάθε ένα δεν προλαμβάνεται από το επόμενο και το επόμενο δεν έπεται λογικά (χρονικά) του προηγουμένου. Το πρόβλημα όμως είναι φτιαχτό. Δηλαδή ποιος ορίζει τον χρόνο και τη σειρά εμφάνισης των πραγμάτων. Ποιος μας δίνει τον τρόπο συσχέτισης του κάθε τι. Ο χρόνος. Ο χρόνος όμως είναι μια ανθρώπινη επινόηση που φτιάχτηκε από αυτόν, για σκοπούς αυτού και για εξυπηρέτηση αυτού. Πράγμα που πιστεύει ο Αϊνστάιν καθώς και εγώ. Άρα τον χρόνο ο καθένας μπορεί να τον ορίσει σύμφωνα με τον δικό του τρόπο θέασης των πραγμάτων.

Ναι τα στατιστικά, δεν μπορούν να χαρακτηριστούν ως χρονοσειρά όπως για παράδειγμα μια πρόταση στην ελληνική γλώσσα όπου το κάθε γράμμα ακολουθεί

ακριβώς το προηγούμενο και ούτω κάθε εξής για να φτιαχτεί μια πρόταση. Όμως μπορούν εύκολα να χαρακτηριστούν ως στοιχεία-γράμματα μιας πρότασης, άσχετα όσον αφορά τη σειρά εμφάνισης τους στον πραγματικό κόσμο, μιας νέας γλώσσας, με δικά της γνωρίσματα και δικό της τρόπο γραφής. Τώρα ποια η σειρά εμφάνισης τους είναι δική μου επιλογή. Δική μου δημιουργία η γλώσσα δική μου και η σύνδεση. Όμως ο χρόνος, ο δικός μου χρόνος δεν είναι κάτι εντελώς αφηρημένο. Απλά θεωρώ ως χρόνο μηδέν το πρώτο στατιστικό, χρόνο ένα το δεύτερο στατιστικό και πάει λέγοντας. Αυτός ο χρόνος είναι σταθερός και ακλόνητος που βάση αυτού υφίσταται η ύπαρξη μιας γλώσσας.

Μιας γλώσσας ομοιόμορφης, με λέξεις-προτάσεις 26 γραμμάτων-στατιστικών που στο ανθρώπινο μάτι μοιάζουν ακαταλαβίστικες. Όπως όμως, ακαταλαβίστικα θα έμοιαζαν σε κάποιον αρχαίο έλληνα οι αγγλικές προτάσεις, ή όπως ακαταλαβίστικες μοιάζουν σε μένα οι κινέζικες προτάσεις. Όλες αυτές οι φτιαχτές γλώσσες πριν να δημιουργηθούν-ανακαλυφτούν και πριν να βρεθούν οι λέξης που την απαρτίζουν, ο τρόπος που συνδέονται αυτές, όλα τα γνωρίσματα τους (γένος, αριθμός, κλίσεις, ορθογραφία κτλ) ήταν ακαταλαβίστικες όπως και η δική μου ακαταλαβίστικη γλώσσα.

Όπως δηλαδή αυτές οι γλώσσες μπορούν σήμερα να θεωρούνται κατανοητές, και γλώσσες που συμβαδίζουν χρονικά, έτσι και η δική μου, ομολογουμένως πολύ δυσνόητη γλώσσα και άχρηστη για οποιοδήποτε άλλο σκοπό παρά της περιγραφής του αποτελέσματος, more ή less, μπορεί. Μιας γλώσσας που ο μόνος σκοπός της είναι η δυαδική κατηγοριοποίηση. Μιας καινούργιας ακαταλαβίστικης γλώσσας που δεν έχει τα γνωρίσματα κοινών γλωσσών αλλά τα δικά της καινούργια γνωρίσματα που βοηθούνται μέσω του δικτύου να βρεθούν, να βγουν στην επιφάνεια και να ξεχωρίσουν κάθε λέξη-πρόταση της γλώσσας προς τη μία ή την άλλη πλευρά του φράχτη κατηγοριοποίησης. Μιας γλώσσας που το κάθε αντικείμενο-γράμμα-στατιστικό της έχει τη σημασία του να βρίσκεται σε ορισμένο σημείο, κάθε κομμάτι της λέξης το ίδιο καθώς και κάθε λέξη-πρόταση επίσης.

Έτσι αν θεωρήσουμε το κάθε στατιστικό ως γράμμα μιας πρότασης μιας γλώσσας ανύπαρκτης, το οποίο χρονικά (όχι όπως το ορίζει ο άνθρωπος όπου δεν έχει λογική) έχει σκοπό και λόγω ύπαρξης στο κάθε σημείο, τότε μπορεί να γίνει αντιληπτό το πώς μπορούμε να δημιουργήσουμε μια λέξη-πρόταση δικής μας έμπνευσης και μιας τέτοιας έμπνευσης γλώσσα βασισμένη στις νέες αυτές προτάσεις.

Το να γίνει αυτό αντιληπτό είναι θέμα όρασης και οπτικής αφού η προαναφερθείσα κατάσταση αποτελεί μια νοητή γραμμή που δύσκολα μπορεί να εμφανιστεί αλλά αν φανεί παρουσιάζεται εύκολη η υπέρβασή και υπερπήδηση της.

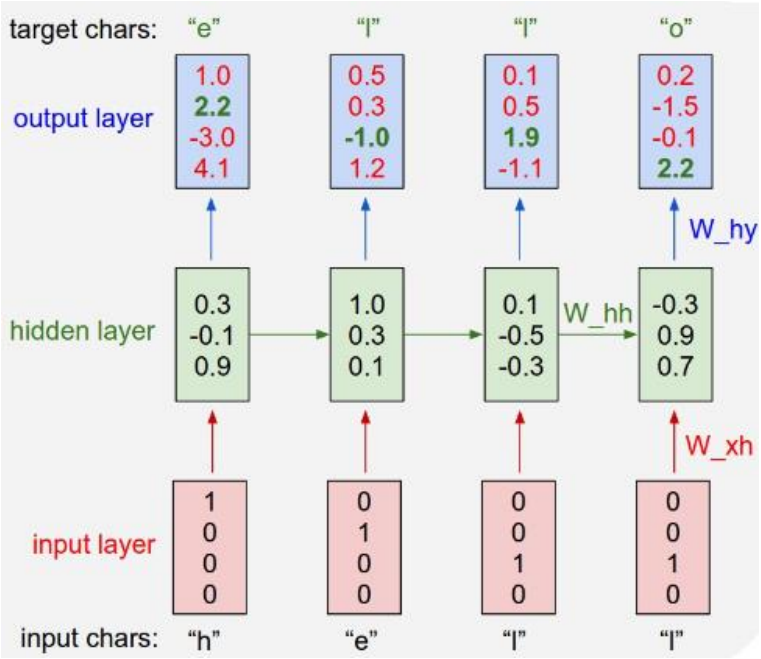
Η αρχιτεκτονική που ταιριάζει προφανώς καλύτερα στο δίκτυο μου και αυτή που επέλεξα είναι η many to one. Έτσι εισάγω τη σειρά με τα στατιστικά στο δίκτυο και επιθυμώ να διαβάσει τη σειρά αυτή και βάση αυτής να αποφασίσει πιο θα είναι το αποτέλεσμα. Άρα έχω 26 νευρώνες εισόδου και ένα νευρώνα για έξοδο στο δίκτυο μου.

Το δεύτερο πράγμα που έπρεπε να γίνει είναι η μετατροπή των δεδομένων. Αυτό έγινε με μιας μορφής κανονικοποίηση. Αυτό που έγινε είναι αφού τα δεδομένα μου κατέχουν τιμές από το μηδέν μέχρι το ένα αυτά να χωριστούν ως γράμματα από το αγγλικό a μέχρι το z. Έτσι η νέα μορφή των δεδομένων είναι η κάτωθι:

aheksockqoepxlakhqidkaldpf+

με το κάθε γράμμα της πιο πάνω λέξης να σημαίνει και ένα από τα στατιστικά του παιχνιδιού.

Πιο κάτω θα αναλύσω απλά το πώς το δίκτυο αυτό εκπαιδεύεται και πως μαθαίνει να επιλέγει το κάθε επόμενο αντικείμενο.



Εικόνα 3.2

Στο σχήμα 3.2 φαίνεται ένα μικρό δίκτυο το οποίο για σκοπούς απλοποίησης έχει ως λεξικό μόνο τέσσερα γράμματα, τα h,e,l και o. Το επιθυμητό δίκτυο σε αυτή την περίπτωση θα ήταν αυτό που μαθαίνει τη λέξη hello. Έτσι εισάγουμε κατά σειρά τα γράμματα h,e,l,l στους νευρώνες εισόδου και επιθυμήσουμε μετά από κάθε γράμμα εισόδου να βρεθεί το αμέσως επόμενο της λέξης.

Ο κάθε νευρώνας εισόδου μεταφράζει το γράμμα εισόδου σε τιμή αναλόγως της θέσης του εν λόγω γράμματος στον πίνακα του λεξιλογίου, δηλαδή βάζει τον αριθμό 1 σε αυτή τη θέση και μηδέν στις υπόλοιπες. Κάτι παρόμοιο γίνετε και με την επιθυμητή έξοδο αφού για την τιμή που θέλουμε να βγάλει το σύστημα σε κάθε περίπτωση, προσπαθούμε να αυξήσουμε την τιμή της θέσης αυτής και να μειώσουμε τις τιμές των άλλων. Αυτό γιατί ως αποτέλεσμα θα πάρουμε την πιο μεγάλη τιμή από όλες τις τιμές και θα την αντιστοιχήσουμε με το γράμμα στην ίδια θέση του λεξιλογίου.

Έτσι για την έξοδο του h επιθυμητό θα ήταν το γράμμα e, χρώματος πρασίνου, για το e το l για το πρώτο l πάλι το l και για το δεύτερο το o. Το δίκτυο επιτυγχάνει να γνωρίζει τι θα έχει ως αποτέλεσμα για κάθε τιμή εισόδου ακόμα και για τα l τα οποία είναι

δίπλα-δίπλα αφού καταλαβαίνει τη σειρά με την οποία η λέξη έρχεται και γνωρίζοντας αυτό αποφασίζει για κάθε γράμμα τι αποτέλεσμα θα ήταν επιθυμητό να βγει. Αυτό που γίνεται λοιπόν είναι το δίκτυο για κάθε γράμμα και θέση στη λέξη στην οποία παρουσιάζεται, να εντοπίζει την πιθανότητα του κάθε επόμενου γράμματος να εμφανιστεί. Και αυτό γιατί μέσω της εκπαίδευσης και δη μέσω του εντοπισμού του σφάλματος σε κάθε περίπτωση ενημερώνεται το δίκτυο για το ποια θα πρέπει να είναι τα βάρη των νευρώνων στο κρυφό επίπεδο .

Για να βρεθεί το σφάλμα το RNN δίκτυο εκτελεί την παραγωγίσιμη συνάρτηση της υπερβολικής εφαπτομένης(tanh). Με βάση αυτή εκτελείται το Back Propagation σε αυτή την αλυσίδα δεδομένων ούτως ώστε τα βάρη να γίνεται η εκπαίδευση από το τέλος μέχρι την αρχή της αλυσίδας. Άρα για κάθε είσοδο προσπαθεί να συνδυάσει το κάθε γράμμα εισόδου και τη σειρά στην οποία αυτό βρίσκεται με σκοπό να βρεθεί η επιθυμητή έξοδος.

Στο δικό μου πρόβλημα όμως έχουμε μόνο μία έξοδο στο τέλος της συμβολοσειράς, άρα δεν πρέπει να υπάρχει έξοδος για κάθε γράμμα λόγω του ότι δεν θέλουμε να μάθει ποιο γράμμα πιθανώς να ακολουθεί το κάθε άλλο αφού αυτό δεν έχει καμία σημασία, αλλά ποιο γράμμα ακολουθεί ολόκληρη τη συμβολοσειρά. Έτσι για το κάθε γράμμα πρέπει να λαμβάνεται υπόψιν η θέση του αλλά και ο συνδυασμός της ακολουθίας για να αποφασιστεί ποιο θα είναι το αποτέλεσμα στο τέλος.

Έτσι δημιούργησα μια μορφή μάσκας για τα δεδομένα εξόδου. Άρα αντί να μπαίνει, ως επιθυμητή έξοδος για κάθε γράμμα-στατιστικό, μια τιμή διαφορετική κάθε φορά, όπως θα ήταν λογικό σε προβλήματα όπως αυτό που περιγράφεται πιο πάνω, δημιουργήθηκε η μάσκα αυτή με μηδέν για τις πρώτες εικοσιπέντε θέσεις του πίνακα με τις επιθυμητές εξόδους και η τιμή του χαρακτήρα για more ή less στην τελευταία μόνο θέση. Έτσι το δίκτυο επικεντρώνεται μόνο στο να βρει τις λύσεις για την τελευταία έξοδο, το συνδυασμό δηλαδή όλης της γραμμής εισόδου και όχι να συγχύζεται, νομιζόμενο πως πρέπει να βρει συνδέσεις μεταξύ κάθε γράμματος με το επόμενο σε όποια θέση και να βρίσκονται, αλλά μόνο τη σύνδεση που υπάρχει μεταξύ όλων των εισόδων και της επιθυμητής εξόδου.

Ειδικότερα λοιπόν όσον αφορά την υλοποίηση το πρώτο που γίνεται είναι να δημιουργηθεί το δίκτυο, όπως ακριβώς και το πρώτο με τα ίδια ονόματα μεταβλητών. Έπειτα η κάθε γραμμή εισόδου στέλνεται στη συνάρτηση όπου γίνεται η πρόσθια και αντίστροφη τροφοδότηση και επιστρέφεται (loss_fun, Παράρτημα Γ). Η εκπαίδευση των βαρών γίνεται με παρόμοια τακτική όπως και στην πρώτη προσπάθεια αφού το σφάλμα που εντοπίζεται από την έξοδο μεταφέρεται προς τα πίσω μέσω του Back Propagation. Η διαφορά όμως είναι πως για την έξοδο του κάθε νευρώνα δε λαμβάνονται υπόψην όλες οι έξοδοι των κρυφών νευρώνων κατά το προηγούμενο βήμα αλλά μόνο το τι έχει βγει από το

Όμως αφού είδα πως η αλλαγή των βαρών κατά το τέλος της κάθε εποχής δημιουργεί μια σύγχυση στο δίκτυο η λειτουργία που γίνεται σε αυτό το δίκτυο είναι να γίνεται αλλαγή των βαρών κάθε φορά που διαβάζεται μια λέξη έτσι ώστε για την κάθε περίπτωση να εντοπίζεται το λάθος και έτσι να υφίσταται σαν αυθαίρετη οντότητα δεδομένου.

Για την αλλαγή τώρα των βαρών εκτελείται η συνάρτηση Adagrad η οποία είναι μια συνάρτηση βασισμένη στην μέθοδο κατάβαση κλίσης. Η μέθοδος αυτή όμως με τη χρήση έξυπνων τεχνικών, επιλέγει πότε θα κάνει μικρότερες και πότε μεγαλύτερες αλλαγές βαρών γεγονός που βοήθησε σε προβλήματα να βρεθεί καλύτερη σύγκλιση. Η συνάρτηση αυτή χρησιμοποιείται σε περιπτώσεις επεξεργασίας γλώσσας και αναγνώρισης εικόνας περιπτώσεις που τα δεδομένα δηλαδή είναι αραιά και έτσι θα αποφασίζει το ποσό της μεταβολής των βαρών.

Άλλη μια αλλαγή που γίνεται σε αυτό το δίκτυο σε σχέση με το προηγούμενο είναι ότι η συνάρτηση ενεργοποίησης είναι η tanh. Η συνάρτηση αυτή μοιάζει αρκετά με τη σιγμοειδή συνάρτηση ενεργοποίησης, που χρησιμοποιήθηκε στο πρώτο δίκτυο, με τη διαφορά πως αντί για αποτελέσματα από μηδέν ως ένα, εξάγει αποτελέσματα σε μεγαλύτερο εύρος αλλά και αρνητικά αποτελέσματα αφού η έξοδος του κυμαίνεται από το πλην ένα μέχρι το ένα. Η διαφορά της δηλαδή με τη σιγμοειδή είναι πως για πολύ μικρές τιμές εισόδου η σιγμοειδής θα έβγαζε μια τιμή κοντά στο μηδέν άρα καθόλου αλλαγή, ενώ η υπερβολική εφαπτομένη(tanh) για τέτοιες τιμές βγάζει αρνητική τιμή. Αυτό τη βοηθά σε περιπτώσεις όπου με τη σιγμοειδή, ως συνάρτησης ενεργοποίησης,

θα κολλούσε το δίκτυο σε τοπικό ελάχιστο, με τη χρήση υπερβολικής εφαπτομένης θα το υπερπηδήσει αφού η μόνη περίπτωση να βγάλει μηδέν η tanh είναι να δεχθεί ως είσοδο με άθροισμα μηδέν. Βάση των πιο πάνω, με τη χρήση της συνάρτησης αυτής πιθανώς να φτάσουμε σε καλύτερα αποτελέσματα και το δίκτυο να μην κολλήσει σε κάποιο σημείο.

Με την πάροδο του χρόνου το δίκτυο φαίνεται να μαθαίνει πολύ καλύτερα από το πρώτο τα δεδομένα εκπαίδευσης και αρκετά καλύτερα τα δεδομένα ελέγχου γεγονός που θα παρουσιαστεί στο αμέσως επόμενο κεφάλαιο με τα αποτελέσματα.

Κεφάλαιο 4

Αποτελέσματα και Συζήτηση

4.1 Αποτελέσματα και Συζήτηση

4.1Αποτελέσματα και Συζήτηση

Σε αυτό το κεφάλαιο θα αναφέρω μερικά από τα πειράματα που έγιναν και μας δίνουν αποτελέσματα τα οποία επιδέχονται συζήτησης και μέσω αυτών μπορούν να εξαχθούν κάποια συμπεράσματα.

Όλα τα δίκτυα μου έγιναν με τον μέγιστο αριθμό δεδομένων εκπαίδευσης που είναι όλα τα παιχνίδια του ιταλικού πρωταθλήματος από την περίοδο 2003-04 μέχρι και αυτή του 2013-14. Τα δεδομένα ελέγχου αν και στην πραγματικότητα θα είναι μόνο 10 παιχνίδια, αυτά δηλαδή της ερχόμενης αγωνιστικής, χρησιμοποίησα ολόκληρη τη ποδοσφαιρική περίοδο του 2014-15 με σκοπό να εξαχθούν πιο ασφαλή συμπεράσματα σε σχέση με τη λειτουργικότητα του δικτύου. Για το τελικό δίκτυο όμως δίκτυο θα δειχθούν τα αποτελέσματα εκπαίδευσης και για τις δύο τελευταίες αγωνίστηκες του φετινού πρωταθλήματος της Ιταλίας και τα ποσοστά επιτυχίας για την κάθε μία.

Πιο κάτω θα παρατεθούν τα ποσοστά επιτυχίας για κάθε μια από τις περιπτώσεις εκτέλεσης των δικτύων RNN. Για το κάθε δίκτυο έγιναν αρκετές δοκιμές αλλά θα παρουσιάσω μόνο τα καλύτερα αποτελέσματα που παρουσιάστηκαν από το κάθε ένα. Έτσι θα παρουσιάσω και για τα τρία μόνο ένα παράδειγμα εκπαίδευσης και τα αποτελέσματα θα συζητηθούν πιο κάτω.

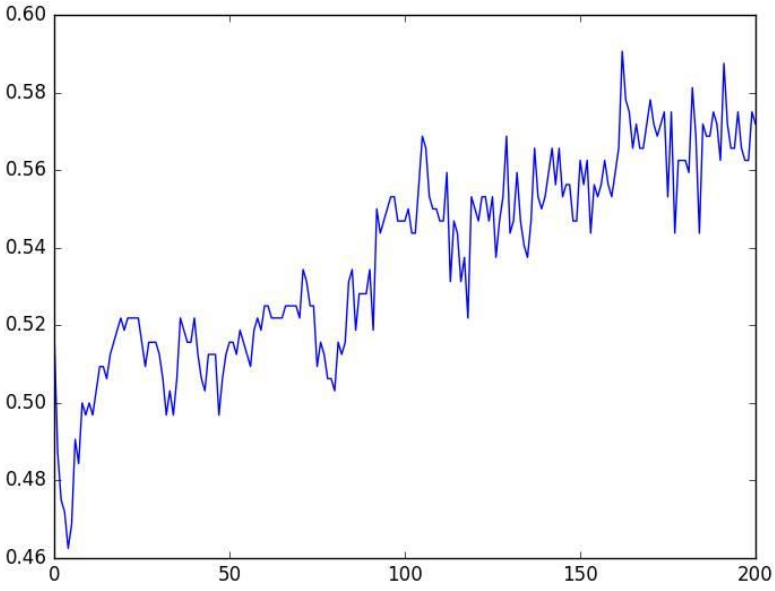
Το πρώτο, απλούστερο, δίκτυο, με ένα επίπεδο νευρώνων και ένα νευρώνα εξόδου, και αριθμό κρυφών νευρώνων 18 και ρυθμό μάθησης 0.01 έβγαλε τα εξής αποτελέσματα:

Ποσοστό επιτυχίας δεδομένων εκπαίδευσης: 1900/3452 (55%)
Ποσοστό επιτυχίας δεδομένων ελέγχου: 173/320 (54%)

Τα αποτελέσματα αυτά δεν παρουσιάζουν κάποιο ενδιαφέρον αφού δεν φαίνεται το δίκτυο να μπορεί να παρουσιάσει αρκετά μεγάλο ποσοστό επιτυχίας άρα και κέρδος.

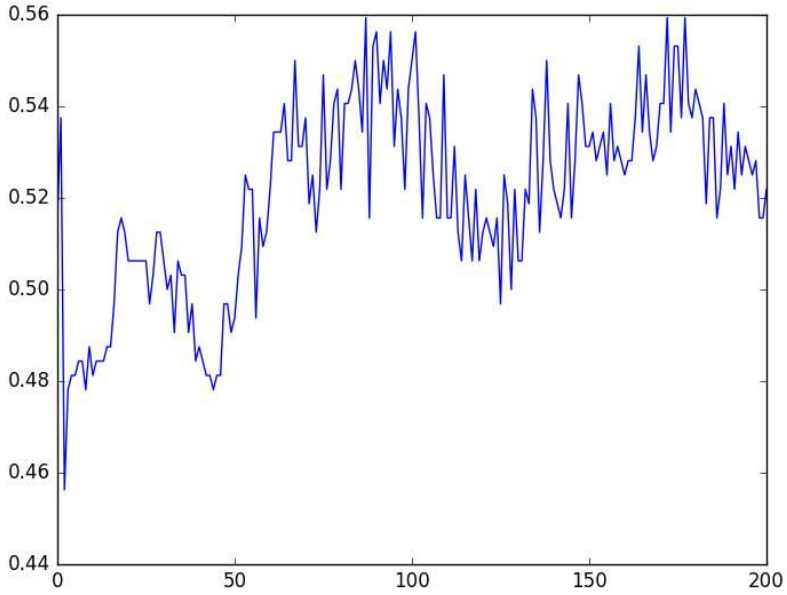
Το δεύτερο δίκτυο, παρόμοιο με το πρώτο αλλά με δύο νευρώνες στην έξοδο παρήγαγε τα πιο κάτω αποτελέσματα:

Ποσοστό επιτυχίας δεδομένων εκπαίδευσης: 2025/3452 (58.5%)



Γραφική Δεδομένων Εκπαίδευσης

Ποσοστό επιτυχίας δεδομένων ελέγχου: 179/320 (56%)



Γραφική Δεδομένων Ελέγχου

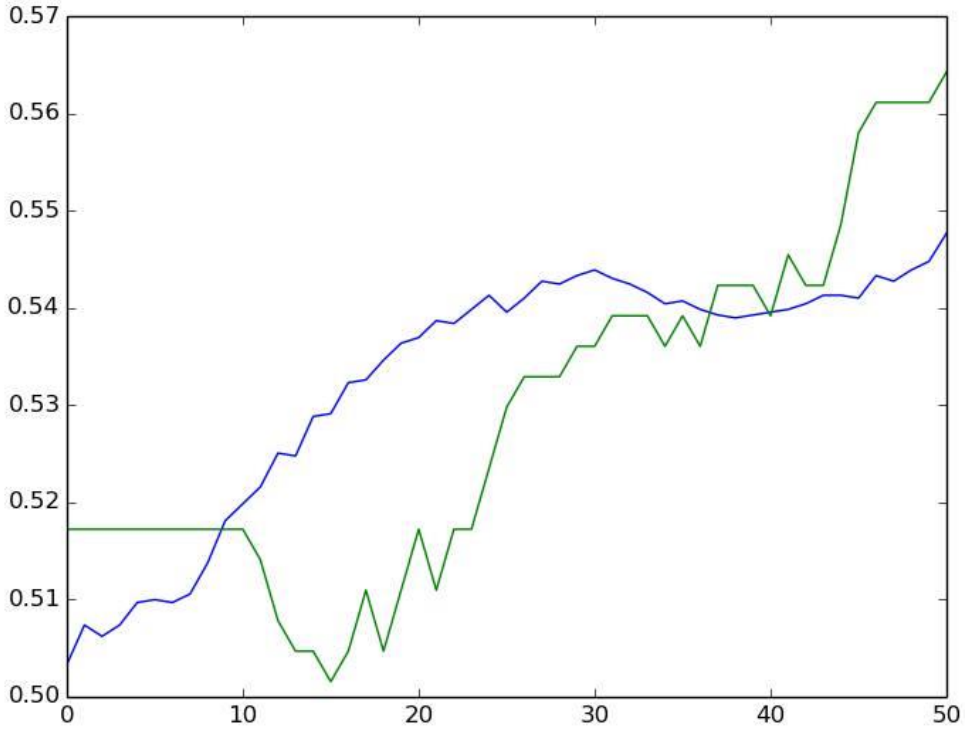
Τα αποτελέσματα αυτού του δικτύου δείχνουν μιας μορφής βελτίωση σε σχέση με το προηγούμενο δίκτυο. Άρα φαίνεται ότι η πρόσθεση ενός ακόμα νευρώνα εξόδου να βοήθησε στην καλύτερη εκμάθηση των δεδομένων. Όμως το πρόβλημα συνεχίζει να υφίσταται αφού το δίκτυο λαμβάνει και εξάγει λίγα συμπεράσματα από την εκπαίδευση του, όχι αρκετά για να μας προσφέρουν ένα ικανοποιητικό αποτέλεσμα.

Για όσες εποχές και εκπαιδευόταν τα πιο πάνω δίκτυα και πάλι δεν παρουσίαζαν κάποια σύγκληση παρά μόνο άλλαζαν αρκετές από τις τιμές που υπέθετε σε σύγκριση με την αμέσως προηγούμενη εποχή εκπαίδευσης. Δηλαδή δεν ακολουθούν μια σταθερή

ανοδική πορεία αφού μετά από μερικά βήματα προόδου, όπου και φαίνεται να μαθαίνουν μερικά δεδομένα ελέγχου, το ποσοστό επιτυχίας κατεβαίνει ξανά προς τα κάτω και αυτή η διαδικασία επαναλαμβάνεται συνεχώς. Μάλλον σε αυτό το σημείο έχει βρεθεί το δίκτυο μεταξύ δυο τοπικών ελαχίστων σφάλματος και πηγαίνει από το ένα στο άλλο συνεχώς. Το γεγονός αυτό θα μπορούσε να λυθεί με αύξηση του ρυθμού μάθησης ή και την αλλαγή του αριθμού των νευρώνων στο κρυφό επίπεδο. Όμως τα αποτελέσματα μετά τις αλλαγές αυτές ήταν παρεμφερή γεγονός που μας εμπνέει την υποψία ότι το πρόβλημα πιθανός να βρίσκεται κάπου αλλού.

Το τρίτο δίκτυο εκπαιδεύτηκε στη λογική των deep networks δηλαδή προστέθηκε στο ήδη υπάρχον δίκτυο ένα ακόμα επίπεδο κρυφών νευρώνων. Αρχική μου προσπάθεια για την υλοποίηση του δικτύου αυτού ήταν να βρεθούν πιο πολλές συνδέσεις μεταξύ των στοιχείων εισόδου και αυτό σε μερικό βαθμό κατέστη δυνατό. Τα δύο αυτά επίπεδα κατείχαν αριθμό νευρώνων 12 και 10 αντίστοιχα ενώ ο τρόπος εκπαίδευσης του δικτύου δεν διαφέρει σε τίποτα από τους προηγούμενους δύο. Τα αποτελέσματα που πάρθηκαν και από αυτό το δίκτυο και παρουσιάζονται πιο κάτω:

Ποσοστό επιτυχίας δεδομένων εκπαίδευσης: 1950/3452 (57%)
Ποσοστό επιτυχίας δεδομένων ελέγχου: 183/320 (57%)



Γραφική Δεδομένων Εκπαίδευσης(μπλέ) Δεδομένα Ελέγχου(πράσινο)

Όπως φαίνεται τα αποτελέσματα είναι μερικώς ικανοποιητικά. Το δίκτυο αυτό αν και δεν παρουσιάζει μεγαλύτερο ποσοστό επιτυχίας στα δεδομένα εκπαίδευσης, από τα προηγούμενα δίκτυα, εν τούτης δείχνει να αποδίδει καλύτερα στα δεδομένα ελέγχου. Αυτό είναι πιθανό να γίνεται, αφού η ποσότητα των δεδομένων εκπαίδευσης είναι πολύ μεγαλύτερη από αυτή των δεδομένων ελέγχου. Αυτό σημαίνει πως πιθανώς το δίκτυο να εντόπισε και να έμαθε κάποιες σχέσεις μεταξύ των δεδομένων και αυτό μεταφράστηκε σε εντοπισμό σωστού σημείου ως έξοδο σε δεδομένα ελέγχου.

Άλλο ένα χαρακτηριστικό που παρουσιάζει αυτό το δίκτυο είναι η σταθερότητα άρα και μεγαλύτερη αξιοπιστία. Αυτό είναι άλλη μια απόδειξη πως το δίκτυο έχει πάρει τη γενική του μορφή και από εκεί και πέρα αλλάζει ελαφρώς.

Τα δεδομένα εκπαίδευσης από την άλλη όσο περνά ο χρόνος σιγά-σιγά παρουσιάζουν καλύτερα αποτελέσματα. Αυτό δεικνύει πως το δίκτυο έχει τη δυνατότητα να μάθει να

συνδέει κάποια δεδομένα μεταξύ τους αλλά επίσης σημαίνει πως για να μαθευτούν τα αποτελέσματα αυτά χρησιμοποιεί τα δεδομένα, ως μορφή σειράς αφίξεως σειρών εισόδου και όχι ως σειρά αφίξεως των στοιχείων της κάθε εισόδου ξεχωριστά.

Αυτό μάλλον είναι και το μεγαλύτερο πρόβλημα του δικτύου αφού το κύριο μέλημα μου είναι να χρησιμοποιηθεί ένα δίκτυο που αναγνωρίζει το κάθε παιχνίδι ξεχωριστά ως είσοδο και όχι να συνδέει την κάθε είσοδο με την προηγούμενη. Το γεγονός αυτό μπορεί να γίνει αντιληπτό αν κάποιος δει τις συνδέσεις μεταξύ των νευρώνων εισόδου.

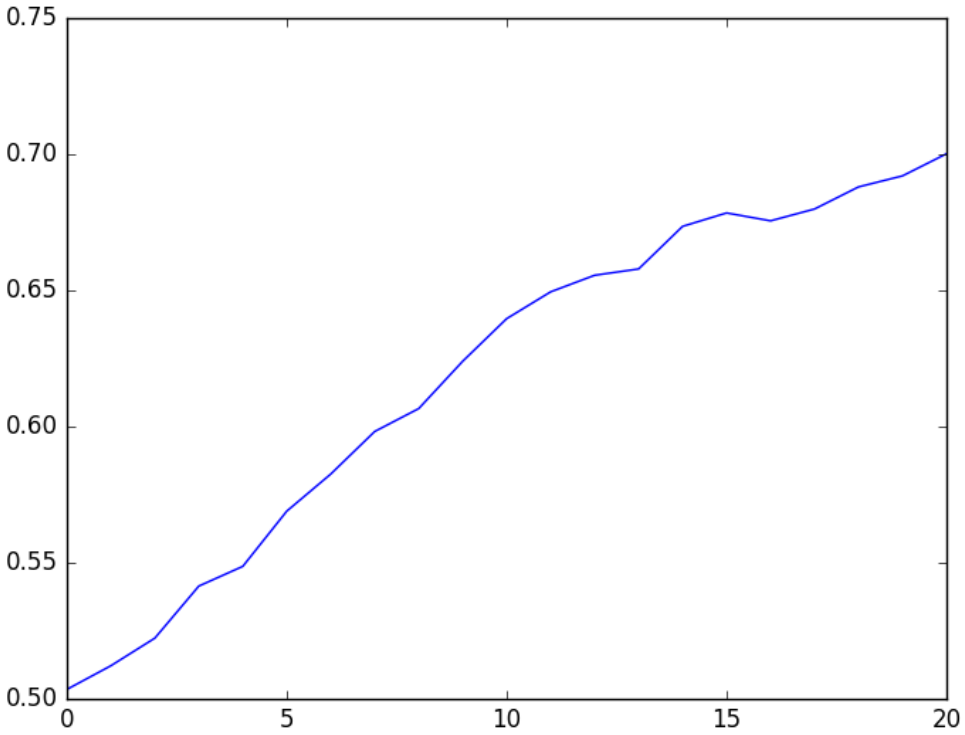
Στις πρώτες αυτές προσπάθειες για υλοποίηση RNN δικτύων, ο κάθε κρυφός νευρώνας είναι ενωμένος με κάθε άλλο νευρώνα στο κρυφό επίπεδο. Έτσι προσπαθεί να αντιληφθεί την κάθε σύνδεση που μπορεί να υπάρχει μεταξύ των στοιχείων και μπροστά του και πίσω του για να αποφασίσει ποιο θα είναι το αποτέλεσμα και όχι να συνδυάσει την κάθε θέση κάθε στοιχείου ως προς την έξοδο.

Ακόμα μια παράμετρος που παίζει σημαντικό ρόλο στη γενίκευση είναι και ο αριθμός των δεδομένων εκπαίδευσης. Ένα δίκτυο RNN συνήθως χρειάζεται πολύ μεγάλο αριθμό εισόδων για να εκπαιδευτεί ούτως ώστε να έχει την ευχέρεια να μπορεί για κάθε δεδομένο που δεν έχει ξαναδεί να εξάγει τη σωστή απάντηση.

Όπως βλέπουμε και από τις τρείς περιπτώσεις τα δίκτυα φαίνονται να μην μπορούν να συνδυάσουν ικανοποιητικά την κάθε είσοδο με την ανάλογη έξοδο. Αυτό γίνεται, όπως προανέφερα, επειδή τα δεδομένα μου είναι σε μια μορφή ανεξαρτησίας μεταξύ τους. Δηλαδή το κάθε ένα παιχνίδι διακρίνεται να ξεχωρίζει από τα υπόλοιπα ως ένα αυτάρκες αντικείμενο που στέκει από μόνο του. Όμως αφού ο τρόπος εκπαίδευσης του δικτύου αυτού γίνεται με το τέλος της κάθε εποχής αυτό δεν δίνει στο δίκτυο να καταλάβει για κάθε μια από τις εισόδους τι θα έπρεπε να βγάζει ως έξοδο ούτως ώστε να ψάξει και να αντιληφθεί τον τρόπο που το κάθε στοιχείο κάθε δεδομένου εισόδου συνδέεται με το κάθε άλλο. Το δίκτυο αυτό επομένως ψάχνει για να βρει συνδέσεις μεταξύ της μιας εισόδου και των επομένων αφού το λάθος συσσωρεύεται και η αλλαγές των βαρών γίνονται στο τέλος κάθε εποχής.

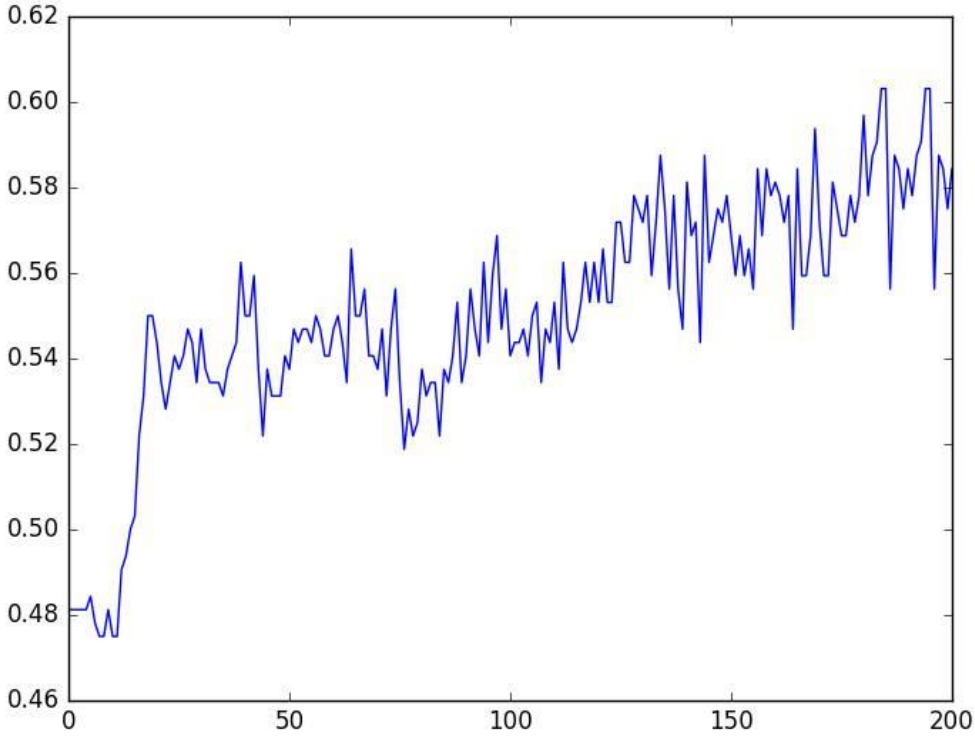
Αυτό φαίνεται να διορθώνεται σε αρκετά μεγάλο βαθμό μέσω της υλοποίησης με τον Back Propagation αληθινού χρόνου όπου βασίζεται στην επεξεργασία κειμένου. Πιο κάτω θα δειχθούν μερικά από τα αποτελέσματα της μεθόδου αυτής και έπειτα θα αναλυθούν.

Ποσοστό επιτυχίας δεδομένων εκπαίδευσης: 2420/3452 (70%)

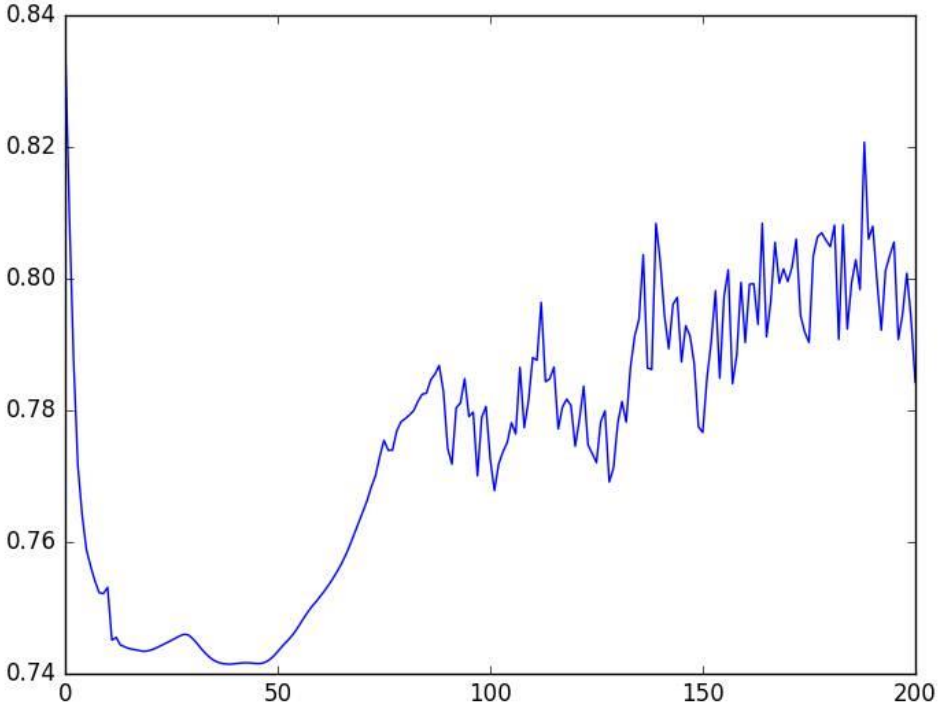


Γραφική Δεδομένων Εκπαίδευσης 1

Ποσοστό επιτυχίας δεδομένων ελέγχου: 193/320 (60%)



Γραφική Δεδομένων Ελέγχου 2



Γραφική ΣφάλματοςΔεδομένων Ελέγχου 3

ΓραφικήError Δεδομένων Ελέγχου 4

Τα ποσοστά αυτά είναι ικανοποιητικά αφού φαίνεται πως το δίκτυο μπορεί να μάθει πολύ καλά την κάθε είσοδο, πολύ καλύτερα από ότι τα πρώτα δίκτυα, αλλά και να γενικεύει καλύτερα αφού το ποσοστό επιτυχίας στα δεδομένα ελέγχου ανεβαίνει και αυτό. Η εκπαίδευση όμως δεν μπορεί να γίνεται επ άπειρο. Όσο μεγαλώνει ο χρόνος εκπαίδευσης, το δίκτυο δείχνει να δίνει έμφαση στις λεπτομέρειες γεγονός που το κάνει όλο και πιο εξειδικευμένο άρα παράλληλα και λιγότερο γενικευμένο.

Όπως προανέφερα έχω ελέγξει το δίκτυο μου και σε προβλήματα αληθινού χρόνου. Έτσι έχω βρει για τις τελευταίες δυο αγωνίστηκες τα σημεία που το δίκτυο μου επισημαίνει ως ορθά για να εξαχθούν τα αποτελέσματα, κατά πόσον δηλαδή μπορεί να βγει κέρδος αν ακολουθηθούν τα σημεία αυτά. Οι τιμές κέρδους που θα παρουσιαστούν πιο κάτω πάρθηκαν από την εταιρία Bwin. Για να ληφθούν τα αποτελέσματα αυτά έχω εκπαιδεύσει το δίκτυο μου με όλα τα παιχνίδια του ιταλικού πρωταθλήματος μέχρι εκείνη τη στιγμή. Άρα τα δεδομένα εκπαίδευσης μου είναι περισσότερα σε σχέση με αυτά του δικτύου που υλοποιήθηκε για να ελέγξει όλα τα παιχνίδια της ποδοσφαιρικής περιόδου του 2014-15.

Στην προτελευταία αγωνιστική το δίκτυο μου έχει εντοπίσει 7 σωστά σημεία, περισσότερα δηλαδή από το μέσο όρο του δικτύου για όλα τα περσινά αποτελέσματα. Πιο συγκεκριμένα λοιπόν σωστά υπολογίστηκαν τα εξής αποτελέσματα.

	Over	**Under**	**Επιτυχίες**
Atalanta – Udinese	1.87	**1.87**	+
Roma – Chievo	**1.45**	2.50	+
Carbi – Lazio	**1.95**	1.80	+
Fiorentina – Palermo	1.65	**2.15**	
Frosinone – Sassuolo	1.65	**2.15**	
Samptoria – Genoa	**1.80**	1.95	+
Torino – Napoli	**1.42**	2.70	
Verona – Juventus	**1.65**	2.15	+
Bologna – Milan	1.80	**1.90**	+
Inter – Empoli	**1.50**	2.40	+

** με bold τα τελικά αποτελέσματα.*

Αν δηλαδή τοποθετήσουμε για κάθε ένα αγώνα από 1 ευρώ το κέρδος θα είναι **12.12** ευρώ(12%). Άρα έχουμε κάποιο κέρδος σε αυτή την περίπτωση. Όμως αφού ο μέσος όρος του δικτύου μου είναι κοντά στο 60% άρα σε άλλες περιπτώσεις το κέρδος μάλλον ή θα είναι μικρότερο ή δεν θα υπάρχει αφού θα η επιστροφή θα είναι μικρότερη του ποσού που πονταρίστηκε.

Υπάρχει και η περίπτωση να τοποθετήσουμε όλους τους συνδυασμούς δύο ομάδων μαζί οι οποίοι είναι 45. Άρα θα βάλουμε 45 ευρώ και το κέρδος θα είναι 60.52(33%).

Στο δεύτερο πείραμα ο αριθμός επιτυχών προβλέψεων ήταν 5 όπως φαίνεται και πιο κάτω.

	Over	**Under**	**Επιτυχίες**
Juventus – Samptoria	**1.55**	2.35	+
Sassuolo – Inter	**1.60**	2.35	+
Napoli – Frosinone	**1.25**	3.70	
Milan – Roma	**1.57**	2.30	
Genoa – Atalanta	**1.53**	2.40	
Chuevi – Bologna	1.75	**2.00**	+
Empoli – Torino	**1.55**	2.35	
Udinese – Carbi	**1.83**	1.91	+
Lazio – Fiorentina	**1.60**	2.25	
Palermo – Verona	**1.42**	2.75	+

** με bold τα τελικά αποτελέσματα.*

Σε αυτή την περίπτωση η επιστροφή είναι 8.40. Άρα σε αυτή την περίπτωση αν ακολουθηθούν οι προτροπές του δικτύου θα βγούμε χαμένοι. Όπως είναι λογικό θα βγούμε χαμένοι και στα πονταρίσματα με δύο ομάδες ανά κουπόνι. Η επιστροφή σε αυτή την περίπτωση θα είναι μόνο 28.11 ευρώ.

Όπως είναι προφανές θα υπάρχουν περιπτώσεις όπου το κέρδος θα είναι αρκετό αλλά και περιπτώσεις όπως η δεύτερη όπου θα βγαίνουμε χαμένοι. Όσο πιο μεγάλο είναι το ποσοστό επιτυχίας για όλα τα παιχνίδια τόσο πιο μεγάλο θα είναι το κέρδος αν παίξουμε και κουπόνια ανά δυο ομάδες.

Κεφάλαιο 5

Συμπεράσματα και Μελλοντική Εργασία

5.1 Συμπεράσματα και Μελλοντική Εργασία

5.1 Συμπεράσματα και μελλοντική εργασία

Σε αυτό το κεφάλαιο θα αναλυθούν τα τελικά αποτελέσματα της εργασίας μου και θα παρατεθούν τα συμπεράσματα για την κάθε μια από τις προσπάθειες που έγιναν προς επίλυση του προβλήματος.

Πρώτα από όλα η υλοποίηση αυτή καθ΄ αυτή παρουσίαζε ενδιαφέρον από την αρχή. Αφού δεν έχει γίνει ξανά προσπάθεια με αυτά τα μέσα να λυθεί ένα τέτοιο πρόβλημα. Μέχρι τώρα η προσπάθεια που γινόταν είναι με μεθόδους ομαδοποίησης δεδομένων όπως δηλαδή με τις υλοποιήσεις δικτύων Kohonen, RBF και SVM. Η δική μου υλοποίηση έγινε στη βάση των δικτύων RNN τα οποία έδειξαν ενδιαφέροντα αποτελέσματα σε προβλήματα αναγνώρισης φωνής και χειρόγραφου κειμένου, σε προβλήματα επεξεργασίας κειμένου και αναγνώριση αντικειμένων σε εικόνες.

Έτσι με την δική μου υλοποίηση η οποία εξ αρχής ήταν περιπετειώδης έχει δειχθεί ότι αν μη τι άλλο μπορεί να γίνει εκμάθηση των δεδομένων και κατηγοριοποίηση τους γεγονός που φέρνει και μικρό κέρδος σε όποιον ακολουθήσει τα αποτελέσματα του.

Η δυσκολία του προβλήματος αυτού είναι εμφανής όταν κοιτάξουμε για να δούμε τις αποδώσεις που οι ιστοσελίδες στοιχημάτων δίνουν για τα σημεία under – over. Ενώ οι τιμές για 1X2 στις περισσότερες περιπτώσεις είναι μικρές για περιπτώσεις όπου παιχνίδια περιέχουν φαβορί ανάμεσα στις διαγωνιζόμενες ομάδες και ειδικότερα αν

αγωνίζονται εντός έδρας, στο under – over οι πιθανότητες που δίνονται σε κάθε σημείο είναι πολύ πιο μοιρασμένες. Έτσι πολλές τιμές σε αυτά τα σημεία είναι πολύ κοντινές, ακόμη και ίδιες, αφού ακόμη και οι στοιχηματικές εταιρίες δυσκολεύονται να βρουν το σημείο με τις περισσότερες πιθανότητες για εμφάνιση.

Τα αποτελέσματα που παράγονται από το τελευταίο δίκτυο παρουσιάζουν ενδιαφέρον αφού φαίνεται πως εντοπίζονται από αυτό συσχετίσεις μεταξύ των δεδομένων εκπαίδευσης και των δεδομένων ελέγχου. Αυτό γίνεται αντιληπτό από το γεγονός ότι τα ποσοστά επιτυχίας και στα δύο σύνολα αυξάνονται και φτάνουν σε ικανοποιητικό σημείο. Το σημαντικό σε αυτό το σημείο είναι να επιτευχθεί το ποσοστό αυτό ούτως ώστε να αποφέρει και στοιχηματικό κέρδος. Όπως φαίνεται και πιο πάνω, στις περιπτώσεις εκπαίδευσης του δικτύου

Όμως το δίκτυο μπορεί να δείχνει ότι μπορεί να μάθει πολλά από τα δεδομένα εισόδου από την άλλη όμως δεν μπορεί να αυξήσει σταθερά το κέρδος μετά από στοιχηματική δραστηριότητα. Η παραδοχή υπάρχει λόγω πολλών παραμέτρων που πιθανώς αν μεταβληθούν να μπορέσουν να βελτιώσουν το εν λόγω δίκτυο.

Ο πρώτος και εμφανέστερος λόγος είναι ότι τα δεδομένα εκπαίδευσης για ένα τέτοιο δίκτυο είναι πολύ λίγα. Τα RNN δίκτυα και ειδικότερα τα Deep RNN θέλουν αρκετά περισσότερα δεδομένα ούτως ώστε να μπορούν να πάρουν μια γενική ιδέα για το ποια θα είναι η ανταπόκρισή τους μετά από κάθε είσοδο.

Αφού όμως περισσότερα δεδομένα δεν μπορούν να ληφθούν αφού δεν υπάρχουν μπορούν να γίνουν μερικές αλλαγές στο δίκτυο που πιθανότατα θα αποφέρουν καλύτερα αποτελέσματα.

Μια εκδοχή είναι να μειώσουμε τις διαστάσεις της εισόδου άρα και του ίδιου του δικτύου. Άρα μπορεί να γίνει μια προεπεξεργασία των δεδομένων εισόδου και να βρεθεί έτσι ποια στοιχεία από τα 26 δεν λαμβάνονται τόσο πολύ υπόψην για την κατάληξη του αγώνα. Έτσι μειώνοντας τα στοιχεία αυτά το δίκτυο θα έχει να δημιουργήσει λιγότερες συνδέσεις μεταξύ των γραμμάτων και έτσι να αυξάνει το ποσοστό επιτυχίας του.

Μια άλλη σκέψη που έχω είναι αφού το δίκτυο είναι ικανό να μάθει λέξης και οι λέξεις στην πραγματικότητα έχουν μέγεθος περίπου μέχρι 10 γράμματα κάτι πρέπει να γίνει και με τη δική μας μεγάλη λέξη για να φτάσει αυτό τον αριθμό. Για να γίνει αντιληπτό αυτό πρέπει να αναφέρω πως μια λέξη 26 γραμμάτων μπορεί να πάρει διαφορετικές μορφές 26^{26}. Ένας πολύ μεγάλος αριθμός που σίγουρα δεν μπορεί να καλυφθεί από τα δικά μας δεδομένα εκπαίδευσης. Επίσης είναι προφανές πως το μεγαλύτερο μερίδιο ευθύνης λαμβάνεται από τα τελευταία γράμματα στα οποία γίνονται και οι μεγαλύτερες αλλαγές κατά την οπισθοδρόμηση. Για να αποφευχθεί αυτό μια πολύ ενδιαφέρουσα λύση θα ήταν να σπάσει η λέξη στα τρία, δηλαδή να γίνουν τρείς λέξης μεγέθους 9,9 και 8 και στην κάθε μια από αυτές να προστεθεί ίδιο σύμβολο ως ένδειξη αποτελέσματος. Έτσι για κάθε μια από τις λέξης θα λαμβάνονταν η έξοδος από το δίκτυο, άρα θα είχαμε 3 απαντήσεις για κάθε λέξη, και όποια απάντηση υπερτερεί θα θεωρείτο0 ως η τελική απάντηση. Το πρόβλημα όμως με αυτή την υλοποίηση είναι ότι δεν παίζει ρόλο σε ποια θέση της μεγάλης λέξης βρίσκεται η υπολέξη αυτή. Έτσι θεμιτό θα ήταν να δημιουργηθούν τρία ίδια δίκτυα ενωμένα στο τέλος με ένα Multi Layer Perceptron, και έτσι το κάθε υποκομμάτι της λέξης θα συγκρινόταν μόνο με τα ίδια κομμάτια της κάθε μιας άλλης λέξης και στο τέλος να γινόταν η σύγκριση των αποτελεσμάτων και η τελική κατάληξη του δικτύου.

Τέλος μια καλή ιδέα για μελλοντική υλοποίηση θα μπορούσε να ήταν το δίκτυο αυτό να γίνει ένα Deep Network ούτως ώστε να λάβει τα προτερήματα που ένα τέτοιο δίκτυο συνοδεύουν, προσθέτοντας ένα επιπλέον επίπεδο νευρώνων στο κρυφό επίπεδο του δικτύου.

Βιβλιογραφία

[1] Matt Jackson, MSc project, School of Computer Science and Information Systems, Birkbeck College, 2001

[2] Kevin Davies, MSc project, School of Computer Science and Information Systems, Birkbeck College, 2004

[3] Sian Turner, MSc project, School of Computer Science and Information Systems, Birkbeck College, 2005

[4] Θεοφάνης Νεοκλέους , Ατομική Διπλωματική Εργασία, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου 2007

[5] Ιακώβου Α. (2010) Πρόβλεψη ποδοσφαιρικών αποτελεσμάτων με τη χρήση τεχνητών νευρωνικών δικτύων, Διπλωματική Εργασία, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου

[6] kelifos.physics.auth.gr/COURSES/neural/K2.pdf

[7] W. S. McCulloch and W. Pitts, A logical calculus of ideas immanent in nervous activity, Bullettin of Mathematical Biophysics, 5, 115(1943).

[8] W. Pitts and W. S. McCulloch, Bullettin of Mathematical Biophysics, 9, 127(1947).

[9] J. von Neumann, The computer and the brain, Yale University Press, New Haven (1958).

[10] D. Hebb, The organisation of behavior, (1949).

[11] F. Rosenblatt, Principles of Neuodynamics, Spartan (N.Y), 1962.

[12] M. Minsky and S. Papert, Perceptrons, MIT Press (1969); expanded edition, 1988.

[13] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy of Sciences, 79,2554(1982).

[14] J. L. McClelland and D. E. Rumelhart, Parallel Distributed Processing, Volumes 1 and 2, MIT Press, Cambridge, Mass. (1986)

[15] http://psi-gr.tripod.com/choc_20_app_neuro.html

[16] http://jkarantzis.blogspot.com.cy/2012/10/blog-post_24.html

[17] el.wikipedia.org/wiki/Νευρωνικό_δίκτυο

[18]http://nemertes.lis.upatras.gr/jspui/bitstream/10889/4419/1/ΔΙΠΛΩΜΑΤΙΚΗ%20ΣΤΑΘΟΠΟΥΛΟΥ%20ΔΗΜΗΤΡΑΣ.pdf

[19] https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks

[20] http://people.idsia.ch/~juergen/deeplearning.html

[21] http://www.cs.bham.ac.uk/~jxb/INC/l12.pdf

[22] http://deeplearning.cs.cmu.edu/notes/shaoweiwang.pdf

[23] https://en.wikipedia.org/wiki/Recurrent_neural_network

[24] http://ir.hit.edu.cn/~jguo/docs/notes/bptt.pdf

[25] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[26] https://cs224d.stanford.edu/reports/TimmarajuAditya.pdf

[27] http://deeplearning.cs.cmu.edu/pdfs/Werbos.backprop.pdf

[28]http://www.cs.colorado.edu/~mozer/Research/Selected%20Publications/reprints/Mozer1989.pdf

Παράρτημα Α

-Κώδικας Crawler για συλλογή δεδομένων των αγώνων

```
<?php
   include_once('simple_html_dom.php');

   for ($offset=2015; $offset<=2015 ; $offset+=1){

      $next = $offset +1;

      $target_url = 'http://www.betexplorer.com/soccer/italy/serie-a-'.$offset.'-
'.$next.'/fixtures/';

      $html = new simple_html_dom();
      $html->load_file($target_url);
      $kr = "";
      $ps="";
      $o=0;
      $p=0;

      foreach($html->find('tr') as $tr){
         if ($tr->children[1]==null)
            break;
         if (($tr->class == "rtitle first-row") || ($tr->class == "rtitle")){
            $o++;
            $link="";
            $th1=$tr->children[0];
            echo $th1->innertext();
            $kr=$ps.$kr;
            $ps="";
         }

         if  (($tr->class == "") && ($tr->children[0]->class=="left") && ($tr-
>children[1]->class=="left")){
            continue;
         }
         if (($tr->class == "first-row") || ($tr->class == "match-line")|| ($tr->class ==
"match-line first-row") || ($tr->class == "match-line strong")){

            if ($tr->children[1]==null)
               break;
            $b=$tr->children[1]->children[0]->innertext();

            $ps=$ps.$b." - 9:9"."\n";

         }
         echo "<br>";
```

```
            $p++;
        }

        $kr=$ps.$kr;

        $target_url = 'http://www.betexplorer.com/soccer/italy/serie-a-'.$offset.'-
'.$next.'/results/';

        $html = new simple_html_dom();
        $html->load_file($target_url);
        $o=0;

        foreach($html->find('tr') as $tr){
            if (($tr->class == "rtitle first-row") || ($tr->class == "rtitle")){
                //echo "<br>".$tr->class;
                $o++;
                $link="";
                $th1=$tr->children[0];
                echo $th1->innertext();
            }
            if  (($tr->class == "") && ($tr->children[0]->class=="left") && ($tr-
>children[1]->class=="left")){
                continue;
            }
            if (($tr->class == "first-row") || ($tr->class == "strong") || $tr->class == ""){
                $a=$tr->children[0]->children[0]->innertext();
                $b=$tr->children[1]->children[0]->innertext();
                $b=substr($b,0,3);
                $kr=$kr.$a." - ".$b."\n";

            }
        }
        $file=$offset.'-'.$next.".txt";
        $pr=fopen($file,"w");
            fwrite($pr, $kr);
            fclose($pr);

    }
?>
```

Παράρτημα Β

-Κώδικας για δημιουργία στατιστικών

```
ArrayList<String> omades=new ArrayList<String>();

int agwnistikes= (omades.size()-1)*2;

agwnes[] team_matches=new agwnes[omades.size()];
apotelesmata[] team_scores=new apotelesmata[omades.size()];
pososta[] team_metrics=new pososta[omades.size()];

for(int k=0; k<omades.size();k++){
        team_matches[k]=new agwnes(agwnistikes);
        team_scores[k]=new apotelesmata(agwnistikes);
        team_metrics[k]=new pososta(agwnistikes);
        for(int i=0; i<agwnistikes;i++)
                for(int j=0; j<23; j++)
                        team_metrics[k].table_p[i][j]=0.0;
}
```

*κομμάτι κώδικα

Παράρτημα Γ

-Κώδικας RNN βαθιού δικτύου

```
def __init__(self, nin, n_hidden_1, n_hidden_2, nout):
        rng = np.random.RandomState(1234)
        W_uh = np.asarray(rng.normal(size=(nin, n_hidden_1), scale= .01, loc = .0),
dtype = theano.config.floatX)
        W_hh_1 = np.asarray(rng.normal(size=(n_hidden_1, n_hidden_1), scale=.01,
loc = .0), dtype = theano.config.floatX)
        W_hh_1_2 = np.asarray(rng.normal(size=(n_hidden_1, n_hidden_2), scale=.01,
loc = .0), dtype = theano.config.floatX)
        W_hh_2 = np.asarray(rng.normal(size=(n_hidden_2, n_hidden_2), scale=.01,
loc = .0), dtype = theano.config.floatX)
        W_hy = np.asarray(rng.normal(size=(n_hidden_2, nout), scale =.01, loc=0.0),
dtype = theano.config.floatX)
        b_hh_1 = np.zeros((n_hidden_1,), dtype=theano.config.floatX)
        b_hh_2 = np.zeros((n_hidden_2,), dtype=theano.config.floatX)
        b_hy = np.zeros((nout,), dtype=theano.config.floatX)
        self.activ = T.nnet.sigmoid
        lr = T.scalar()
        u = T.matrix()
        t = T.scalar()

        W_uh = theano.shared(W_uh, 'W_uh')
        W_hh_1 = theano.shared(W_hh_1, 'W_hh_1')
        W_hh_1_2 = theano.shared(W_hh_1_2, 'W_hh_1_2')
        W_hh_2 = theano.shared(W_hh_2, 'W_hh_2')
        W_hy = theano.shared(W_hy, 'W_hy')
        b_hh_1 = theano.shared(b_hh_1, 'b_hh_1')
        b_hh_2 = theano.shared(b_hh_2, 'b_hh_2')
        b_hy = theano.shared(b_hy, 'b_hy')

        h0_tm1 = theano.shared(np.zeros(n_hidden_1, dtype=theano.config.floatX))
        h1_tm2 = theano.shared(np.zeros(n_hidden_2, dtype=theano.config.floatX))

        h1, _ = theano.scan(self.recurrent_fn, sequences = u,
                                        outputs_info = [h0_tm1],
                                        non_sequences = [W_hh_1, W_uh, W_hy,
b_hh_1])

        h2, _ = theano.scan(self.recurrent_fn_2, sequences = h1,
                                        outputs_info = [h1_tm2],
```

```
                                        non_sequences = [W_hh_2,W_hh_1_2, W_hy,
b_hh_2])

        y = T.dot(h2[-1], W_hy) + b_hy

        cost = (((t - y)**2).mean(axis=0).sum() , y, lr)

        cost_test = y

        gW_hh_1,  gW_hh_1_2, gW_hh_2, gW_uh, gW_hy,\
        gb_hh_1, gb_hh_2, gb_hy = T.grad(
                cost[0], [W_hh_1,  W_hh_1_2,W_hh_2
                        , W_uh, W_hy, b_hh_1, b_hh_2, b_hy])

        self.train_step = theano.function([u, t, lr], cost,
                        on_unused_input='warn',
                        updates=[(W_hh_1, W_hh_1 - lr*gW_hh_1),
                                (W_hh_1_2, W_hh_1_2 - lr*gW_hh_1_2),
                                (W_hh_2, W_hh_2 - lr*gW_hh_2),
                                (W_uh, W_uh - lr*gW_uh),
                                (W_hy, W_hy - lr*gW_hy),
                                (b_hh_1, b_hh_1 - lr*gb_hh_1),
                                (b_hh_2, b_hh_2 - lr*gb_hh_2),
                                (b_hy, b_hy - lr*gb_hy)],
                        allow_input_downcast=False)

        self.test_step = theano.function([u, t, lr], cost_test,
                        on_unused_input='warn',
                        allow_input_downcast=False)

        def recurrent_fn(self, u_t, h_tm1, W_hh_1, W_uh, W_hy, b_hh_1):
                        h_t = self.activ(T.dot(h_tm1, W_hh_1) + T.dot(u_t, W_uh) +
                        b_hh_1)
        return h_t

        def recurrent_fn_2(self, h1_t, h_tm2, W_hh_2, W_hh_1_2, W_hy, b_hh_2):
                        h_t = self.activ(T.dot(h_tm2, W_hh_2) + T.dot(h1_t, W_hh_1_2)
                        + b_hh_2)
        return h_t
```

*κομμάτι κώδικα

Παράρτημα Δ

-Κώδικας RNN για λέξεις παιχνίδια

```
# hyperparameters
hidden_size = 20 # size of hidden layer of neurons
seq_length = 27 # number of steps to unroll the RNN for
learning_rate = 0.1

# model parameters
Wxh = np.random.randn(hidden_size, vocab_size)*0.01 # input to hidden
Whh = np.random.randn(hidden_size, hidden_size)*0.01 # hidden to hidden
Why = np.random.randn(vocab_size, hidden_size)*0.01 # hidden to output
bh = np.zeros((hidden_size, 1)) # hidden bias
by = np.zeros((vocab_size, 1)) # output bias

def lossFun(inputs, targets, hprev):

        xs, hs, ys, ps = {}, {}, {}, {}
        hs[-1] = np.copy(hprev)
        loss = 0
        # forward pass
        for t in xrange(len(inputs)):
                xs[t] = np.zeros((vocab_size,1)) # encode in 1-of-k representation
                xs[t][inputs[t]] = 1
                # hidden state
                hs[t] = np.tanh(np.dot(Wxh, xs[t]) + np.dot(Whh, hs[t-1]) + bh)
                # unnormalized log probabilities for next chars
                ys[t] = np.dot(Why, hs[t]) + by
                # probabilities for next chars
                ps[t] = np.exp(ys[t]) / np.sum(np.exp(ys[t]))

        loss += -np.log(ps[t][targets[t],0]) # softmax (cross-entropy loss)
        # backward pass: compute gradients going backwards
                dWxh, dWhh, dWhy = np.zeros_like(Wxh), np.zeros_like(Whh),
        np.zeros_like(Why)
        dbh, dby = np.zeros_like(bh), np.zeros_like(by)
        dhnext = np.zeros_like(hs[0])
         for t in reversed(xrange(len(inputs))):
                dy = np.copy(ps[t])
                dy[targets[t]] -= 1 # backprop into y
                dWhy += np.dot(dy, hs[t].T)
                dby += dy
                dh = np.dot(Why.T, dy) + dhnext # backprop into h
                dhraw = (1 - hs[t] * hs[t]) * dh # backprop through tanh nonlinearity
                dbh += dhraw
                dWxh += np.dot(dhraw, xs[t].T)
                dWhh += np.dot(dhraw, hs[t-1].T)
                dhnext = np.dot(Whh.T, dhraw)
```

```
        for dparam in [dWxh, dWhh, dWhy, dbh, dby]:
        np.clip(dparam, -5, 5, out=dparam) # clip to mitigate exploding gradients
return loss, dWxh, dWhh, dWhy, dbh, dby, hs[len(inputs)-1] , ps

def lossFun_test(inputs, targets, hprev):
        xs, hs, ys, ps = {}, {}, {}, {}
        hs[-1] = np.copy(hprev)
        loss = 0
        # forward pass
        for t in xrange(len(inputs)):
                xs[t] = np.zeros((vocab_size,1)) # encode in 1-of-k representation
                xs[t][inputs[t]] = 1
                # hidden state
                hs[t] = np.tanh(np.dot(Wxh, xs[t]) + np.dot(Whh, hs[t-1]) + bh)
                # unnormalized log probabilities for next chars
                ys[t] = np.dot(Why, hs[t]) + by
                # probabilities for next chars
                ps[t] = np.exp(ys[t]) / np.sum(np.exp(ys[t]))
                loss += -np.log(ps[t][targets[t],0]) # softmax (cross-entropy loss)

  return loss, ps
```

*κομμάτι κώδικα