Αριστοτελειο Πανεπιστημιο Θεσσαλονικης

Τμημα Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Ηλεκτρονικης και Υπολογιστων

Αυτόματη Συγγραφή Κώδικα με Αναδραστικά Νευρωνικά Δίκτυα (Source Code Generation with Recurrent Neural Networks)

Διπλωματικη Εργασια

του

ΒΑΣΙΛΗ ΜΠΟΥΝΤΡΗ

Επιβλέποντες: Ανδρέας Συμεωνίδης, Επίκουρος Καθηγητής Α.Π.Θ.
Κυριάκος Χατζηδημητρίου, Μεταδιδάκτορας Α.Π.Θ.

Εργαστηριο Ευφυων Συστηματων και Τεχνολογιας Λογισμικου (ISSEL)

Θεσσαλονίκη, Ιούνιος 2017

Ευχαριστίες

Θα ήθελα αρχικά να ευχαριστήσω τον καθηγητή κ. Ανδρέα Συμεωνίδη για την εμπιστοσύνη και την καθοδήγηση και τον κ. Κυριάκο Χατζηδημητρίου για την καθοδήγηση και τη συνεργασία στα πλαίσια αυτής της διπλωματικής εργασίας. Επίσης θέλω να ευχαριστήσω την οικογένειά μου για την αδιάληπτη στήριξη όλα αυτά τα χρόνια.

Περίληψη

Η περίληψη θα συμπληρωθεί αργότερα. Αυτή είναι μια περίληψη άλλης εργασίας:

Ένα σύστημα ομότιμων κόμβων αποτελείται από ένα σύνολο αυτόνομων υπολογιστικών κόμβων στο Διαδίκτυο, οι οποίοι συνεργάζονται με σκοπό την ανταλλαγή δεδομένων. Στα συστήματα ομότιμων κόμβων που χρησιμοποιούνται ευρέως σήμερα, η αναζήτηση πληροφορίας γίνεται με χρήση λέξεων κλειδιών. Η ανάγκη για πιο εκφραστικές λειτουργίες, σε συνδυασμό με την ανάπτυξη του Σημασιολογικού Ιστού, οδήγησε στα συστήματα ομότιμων κόμβων βασισμένα σε σχήματα. Στα συστήματα αυτά κάθε κόμβος χρησιμοποιεί ένα σχήμα με βάση το οποίο οργανώνει τα τοπικά διαθέσιμα δεδομένα. Για να είναι δυνατή η αναζήτηση δεδομένων στα συστήματα αυτά υπάρχουν δύο τρόποι. Ο πρώτος είναι όλοι οι κόμβοι να χρησιμοποιούν το ίδιο σχήμα κάτι το οποίο δεν είναι ευέλικτο. Ο δεύτερος τρόπος δίνει την αυτονομία σε κάθε κόμβο να επιλέγει όποιο σχήμα θέλει και απαιτεί την ύπαρξη κανόνων αντιστοίχισης μεταξύ των σχημάτων για να μπορούν να αποτιμώνται οι ερωτήσεις. Αυτός ο τρόπος προσφέρει ευελιξία όμως δεν υποστηρίζει την αυτόματη δημιουργία και τη δυναμική ανανέωση των κανόνων, που είναι απαραίτητες για ένα σύστημα ομότιμων κόμβων.

Στόχος της διπλωματικής εργασίας είναι η ανάπτυξη ενός συστήματος ομότιμων κόμβων βασισμένο σε σχήματα το οποίο (α) θα επιτρέπει μια σχετική ευελιξία στην χρήση των σχημάτων και (β) θα δίνει την δυνατότητα μετασχηματισμού ερωτήσεων χωρίς την ανάγκη διατύπωσης κανόνων αντιστοίχισης μεταξύ σχημάτων, ξρησιμοποιώντας κόμβους με σχήματα RDF που αποτελούν υποσύνολα-όψεις ενός βασικού σχήματος (καθολικό σχήμα).

Λέξεις Κλειδιά

Σύστημα ομότιμων κόμβων, Σύστημα ομότιμων κόμβων βασισμένο σε σχήματα, Σημασιολογικός Ιστός, RDF/S, RQL, Jxta

Abstract

This is a placeholder for the abstract of my thesis. The actual abstract is going to be written after I finish all chapters. The Compact LInear Collider (CLIC) will use a novel acceleration scheme in which energy extracted from a very intense beam of relatively low-energy electrons (the Drive Beam) is used to accelerate a lower intensity Main Beam to very high energy. The high intensity of the Drive Beam, with pulses of more than 10^{15} electrons, poses a challenge for conventional profile measurements such as wire scanners. Thus, new non-invasive profile measurements are being investigated.

One candidate is the Electron Beam Scanner. A probe beam of low-energy electrons crosses the accelerator beam perpendicularly. The probe beam is deflected by the space-charge fields of the accelerator beam. By scanning the probe beam and measuring its deflection with respect to its initial position, the transverse profile of the accelerator beam can be reconstructed.

Analytical expressions for the deflection exist in the case of long bunches, where the charge distribution can be considered constant during the measurement. In this paper we consider the performance of an electron beam scanner in an accelerator where the bunch length is much smaller than the probe-beam scanning time. In particular, the case in which the bunch length is shorter than the time taken for a particle of the probe beam to cross the main beam is difficult to model analytically. We have developed a simulation framework allowing this situation to be modelled.

Keywords

Fill in

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Η πράξη του προγραμματισμού, δηλαδή η ανάπτυξη μίας διαδικασίας με στόχο την επίτευξη ενός έργου, είναι μια εντυπωσιακή επίδειξη των δυνατοτήτων συλλογιστικής του ανθρώπινου εγκεφάλου. Η αυτοματοποίηση της συγγραφής κώδικα και προγραμμάτων (Αυτόματος Προγραμματισμός) είναι ένας στόχος με μακρόχρονη ιστορία, τόσο για τους μηχανικούς λογισμικού, όσο και για τον κλάδο της τεχνητής νοημοσύνης. Ο ακριβής ορισμός του "Αυτόματου Προγραμματισμού" παραμένει ένα θέμα στο οποίο υπάρχει ασυμφωνία μεταξύ των ειδικών, γεγονός που ενισχύεται από την συνεχή αλλαγή του όρου χάρη στις εξελίξεις της τεχνολογίας. Ο David Parnas, αναζητώντας την ιστορία του όρου, καταλήγει: "Ο αυτόματος προγραμματισμός ήταν πάντα ένας ευφημισμός για προγραμματισμό σε μια υψηλότερου επιπέδου γλώσσα από αυτή που είναι διαθέσιμη στον προγραμματιστή." [;]

Δεδομένης της εγγενούς δυσκολίας και πολυπλοκότητας του στόχου υπάρχει πληθώρα προκλήσεων αλλά και προσεγγίσεων στη λύση του. Δύο σημαντικές ομάδες προσεγγίσεων είναι [;], [;]:

1. Επαγωγικός Προγραμματισμός (Induction Programming)

   Χρησιμοποιώντας τεχνικές τόσο από τον προγραμματισμό όσο και από την τεχνητή νοημοσύνη στοχεύει στη μάθηση προγραμμάτων, τυπικά δηλωτικών και συχνά αναδρομικών. Για την εκμάθηση χρησιμοποιούνται μη αυστηρές προδιαγραφές, όπως παραδείγματα εισόδου - εξόδου ή περιορισμοί.

2. Παραγωγή Κώδικα Βάσει Μοντέλων (Model-Driven Code Generation)

   Στην προσπάθεια των ερευνητών λογισμικού να απλοποιήσουν την "διαδρομή" ανάμεσα στη σχεδίαση και την υλοποίηση χρησιμοποίουνται αφαιρέσεις. Ένα σύνολο τεχνικών με ευρεία και αυξανόμενη χρήση, το οποίο βασίζεται σε τέτοιες αφαιρέσεις, είναι το Model Driven Engineering. Γλώσσες μοντελοποίησης που αφορούν συγκεκριμένους τομείς Domain-specific modeling languages) χρησιμοποιούνται σε συνδυασμό με συστήματα μετατροπών και παραγωγής (transormation engines and generators) για να φτιάξουν αντικείμενα όπως κώδικα, προσομοιώσεις εισόδου ή και άλλα μοντέλα.

Με ένα λειτουργίκο σύστημα αυτόματης παραγωγής κώδικα ο χρόνος ανάπτυξης και ο

αριθμός των λαθών μειώνεται δραματικά. Αντίστροφα, εκτινάσσεται η παραγωγικότητα των χρηστών και απλουστεύεται η αντιμετώπιση σύνθετων προβλημάτων.

1.1 Κίνητρο

Αφενός, η πρόοδος της τεχνητής νοημοσύνης, και ειδικότερα του κλάδου της υπολογιστικής εκμάθησης (Machine Learning), είναι ραγδαία τα τελευταία χρόνια. Αφετέρου, η εξέλιξη και η ευρεία χρήση του λογισμικού δημιουργεί ανάγκες για αυτοματοποίηση στην παραγωγή του αλλά και γιγαντιαία ποσότητα υλικού το οποίο μπορούμε να χρησιμοποιήσουμε. Έχουμε στη διάθεση μας πληθώρα υλοποιημένων προγραμμάτων, σε πολλές διαφορετικές γλώσσες και μορφές, κάθε δυσκολίας και σκοπού. Οι σχετικές τεχνολογικές και θεωρητικές ανακαλύψεις ανοίγουν νέα μονοπάτια πειραματισμού, καινούρια εργαλεία αναπτύσσονται και δημιουργούνται κίνητρα επανεξέτασης κάποιων προβλημάτων.

Σύμφωνα με τα παραπάνω, και ιδιαίτερα χάρη στις πρόσφατες προόδους της υπολογιστικής εκμάθησης γύρω από την ταξινόμηση και παραγωγή κειμένου [;], [;], καλούμαστε να εξετάσουμε πως και σε τι βαθμό μπορούμε να τις εκμεταλλευτούμε ως μηχανικοί λογισμικού. Τι εφαρμογές μπορούν να προκύψουν για την πρόβλεψη και τη διόρθωση κώδικα; Μέχρι ποιο σημείο μπορούμε να αυτοματοποιήσουμε την παραγωγή του;

1.2 Περιγραφή του προβλήματος

Το πρόβλημα που τίθεται προς λύση είναι η αυτοματοποίηση της παραγωγής κώδικα. Δεδομένων των σύγχρονων μεθόδων και τεχνολογιών, αυτό είναι ζήτημα στο οποίο είναι από εξαιρετικά δύσκολο έως αδύνατο να δωθεί μια γενική λύση, τουλάχιστον για το εγγύς μέλλον. Αντί για μία γενική λύση, μπορούμε να επικεντρωθούμε στις διεργασίες οι οποίες είναι μεν απλές, αλλά επαναλαμβάνονται συχνά και είναι χρονοβόρες. Ιδανικά, θα θέλαμε να αποφύγουμε να καταβάλουμε κόπο για να δημιουργήσουμε κάτι το οποίο ήδη υπάρχει.

1.3 Στόχοι της διπλωματικής

Στόχος της διπλωματικής εργασίας αυτής είναι η δημιουργία ενός τεχνητού νευρωνικού δικτύου με αναδράσεις (artificial recurrent neural network) το οποίο αφού εκπαιδευτεί στην συγγραφή κώδικα σε μία γλώσσα προγραμματισμού της επιλογής μας – διαβάζοντας εκατομμύρια γραμμές κώδικα – θα προσπαθήσει να παράξει κώδικα. Δεδομένου του εκπαιδευτικού χαρακτήρα της διπλωματικής εργασίας, θα εξετάσουμε το πρόβλημα αυτόματης παραγωγής κώδικα από μία πληροφοριακά οδηγούμενη (data-driven) σκοπιά, η οποία επιχειρεί να εκμεταλλευτεί τις εξελίξεις στην επιστήμη της πληροφορίας.

Ο κώδικας αυτός γενικά μπορεί να φτάσει σε ένα από τα παρακάτω επίπεδα:

1. Να "μοιάζει" με κώδικα
2. Να μην έχει συντακτικά λάθη

3. Να μπορεί να μεταγλωτιστεί

4. Να "κάνει κάτι χρήσιμο"

Σε επίπεδο διπλωματικής εργασίας επιζητούμε κώδικα στα επίπεδα τουλάχιστον 1 ή και 2.

1.4 Μεθοδολογία

Θα αντιμετωπίσουμε την παραγωγή κώδικα ως ένα πρόβλημα εκμάθησης ακολουθιών (Sequence Learning), προσέγγιση η οποία βρίσκεται ανάμεσα στον επαγωγικό προγραμματισμό και την παραγωγή κώδικα βάσει μοντέλων. Χρησιμοποιούμε μοντέλα βασισμένα σε αναδραστικά νευρωνικά δίκτυα και ένα σύνολο δεδομένων. Το τελευταίο αποτελείται από έναν μεγάλο αριθμό προγραμμάτων σε μια γλώσσα της επιλογής μας. Σε αυτή την περίπτωση θα χρησιμοποιήσουμε τη γλώσσα θαασςριπτ, αλλά το μοντέλο μας είναι αγνωστικό στο ποια γλώσσα μαθαίνει. Η μεθοδολογία μπορεί να χωριστεί, αφαιρετικά, σε 3 μέρη:

1. Προ-επεξεργασία (Pre-processing)

   Δεδομένου ενός μεγάλου όγκου πληροφοριών σε μορφή κώδικα, καλούμαστε να τις επεξεργαστούμε με στόχο την καλύτερη εκμετάλλευση τους από το μοντέλο μας και τελικώς την επίτευξη βέλτιστων αποτελεσμάτων. Αφαιρούμε την πληροφορία που φαίνεται είτε να δυσκολεύει την εκμάθηση του μοντέλου, είτε είναι αδύνατο να ερμηνευτεί από αυτό. Σε μία από τις προτεινόμενες προσεγγίσεις προσθέτουμε πληροφορία για τον κώδικα με σκοπό την αποσαφήνιση των δεδομένων. Η πληροφορία του κώδικα εκφράζεται ως σειρά από στοιχειώδεις χαρακτήρες.

2. Εκπαίδευση (Training)

   Τα προτεινόμενα μοντέλα, τα οποία είναι σύνθετες δομές αναδραστικών νευρωνικών δικτύων, εκπαιδεύονται βάσει της παραπάνω επεξεργασμένης πληροφορίας. Μετά από το 'διάβασμα' μιας σειράς χαρακτήρων καλούνται να προβλέψουν τον επόμενο χαρακτήρα. Οι επιδόσεις εκφράζονται μέσω μιας μετρικής λάθους, την οποία η εκπαιδευτική διαδικασία προσπαθεί να ελαχιστοποιήσει χρησιμοποιώντας γενικευμένες μεθόδους βελτιστοποίησης.

3. Παραγωγή Κώδικα (Source Code Generation)

   Τα εκπαιδευμένα, πλέον, μοντέλα μπορούν να χρησιμοποιηθούν για την παραγωγή κώδικα. Αρχικοποιούνται με κώδικα της επιλογής μας, ο οποίος επεξεργάζεται όπως και τα δεδομένα στα οποία εκπαιδεύεται. Το μοντέλο παράγει ένα χαρακτήρα σε κάθε πρόβλεψη και χρησιμοποιεί την πρόβλεψη του ως αληθή για να παράξει τον επόμενο χαρακτήρα. Με αυτό τον τρόπο μπορεί να συγγράφει απεριόριστη ποσότητα κώδικα.

1.5 Διάρθρωση

Η εργασία αυτή είναι οργανωμένη σε πέντε κεφάλαια: Στο Κεφάλαιο 2 δίνεται το θεωρητικό υπόβαθρο των βασικών τεχνολογιών που σχετίζονται με τη διπλωματική αυτή. Αρχικά περιγράφονται ..., στη συνέχεια το ... και τέλος ... . Στο κεφάλαιο 3 παρουσιά Στο Κεφάλαιο 4 αρχικά παρουσιάζεται ανάλυση και η σχεδίαση του συστήματος ... .. Τέλος στο Κεφάλαιο 5 δίνονται τα συμπεράσματα, η συνεισφορά αυτής της διπλωματικής εργασίας, καθώς και μελλοντικές επεκτάσεις.

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

Στο κεφάλαιο αυτό παρουσιάζονται αναλυτικά οι ...

2.1 Deep Learning[;]

Η υπολογιστική εκμάθηση (Machine Learning) είναι η κινητήριος δύναμη για διάφορες εκφάνσεις της σύγχρονης κοινωνίας: από αναζητήσεις στο διαδίκτυο μέχρι και φιλτράρισμα περιεχομένου σε κοινωνικά δίκτυα και προτάσεις αγορών σε ηλεκτρονικά καταστήματα. Ολοένα συχνότερη και συνηθέστερη γίνεται η εμφάνιση του σε προϊόντα ευρείας κατανάλωσης όπως κάμερες και κινητά τηλέφωνα. Τα συστήματα υπολογιστικής εκμάθησης χρησιμοποιούνται για την αναγνώριση αντικειμένων σε εικόνες, την αυτόματη καταγραφή προφορικού λόγου, την αντιστοίχηση προϊόντων, νέων, δημοσιεύσεων με τις προτιμήσεις χρηστών. Σε όλες αυτές τις εφαρμογές, είναι αυξανόμενη η χρήση ενός σετ τεχνικών που φέρει το όνομα Deep Learning.

Οι συμβατικές τεχνικές υπολογιστικής εκμάθησης είχαν περιορισμένη δυνατότητα χρήσης της ανεπεξέργαστης πληροφορίας. Για δεκαετίες, η σχεδίαση και η υλοποίηση ενός συστήματος αναγνώρισης προτύπων ή υπολογιστικής εκμάθησης, απαιτούσε προσεκτική προσέγγιση και σημαντική εξειδίκευση στον εκάστοτε τομέα. Αυτό επειδή χρειαζόταν η μετατροπή της ανεπεξέργαστης πληροφορίας σε μία κατάλληλη εσωτερική αναπαράσταση, την οποία το υποσύστημα εκμάθησης – συχνότερα ένας ταξινομητής – θα χρησιμοποιούσε για αναγνωρίσει πρότυπα στις διάφορες εισόδους.

Η εκμάθηση αναπαραστάσεων είναι ένα σύνολο μεθόδων που επιτρέπουν σε ένα σύστημα να ανακαλύψει αυτόματα ποιες ακριβώς αναπαραστάσεις της ανεπεξέργαστης πληροφορίας χρειάζεται για να επιτελέσει την αναγνώριση προτύπων ή την ταξινόμηση. Οι μέθοδοι Deep Learning είναι μέθοδοι εκμάθησης αναπαραστάσεων με πολλαπλά επίπεδα αναπαράστασης, που αποτελούνται από την σύνθεση απλών, μη γραμμικών υποσυστημάτων, το καθένα από τα οποία – ξεκινώντας από την ανεπεξέργαστη είσοδο – μετατρέπει την αναπαράσταση της πληροφορίας σε μια λίγο πιο υψηλά αφαιρετική μορφή σε κάθε επίπεδο. Με την χρήση αρκετών τέτοιων μετατροπών το σύστημα μπορεί να μάθει εξαιρετικά σύνθετες λειτουργίες. Για διαδικασίες ταξινόμησης, τα υψηλότερα επίπεδα αναπαράστασης ενισχύουν πτυχές τις εισόδου που είναι

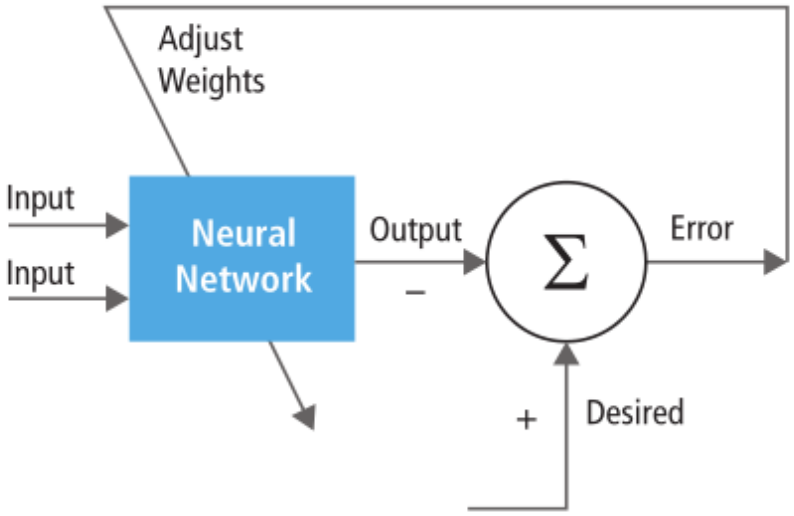
πιο σημαντικές για τον τελικό σκοπό. Σε μία εικόνα, για παράδειγμα, η οποία αναπαριστάται ως διάνυσμα τιμών εικονοκυττάρων, τα χαρακτηριστικά που μαθαίνονται στο πρώτο επίπεδο είναι συνήθως πληροφορία για την παρουσία ή την απουσία ακμών σε συγκεκριμένες θέσεις και προσανατολισμούς. Στο δεύτερο επίπεδο, συνήθως εντοπίζονται μοτίβα μέσω των διαφόρων διατάξεων των ακμών, χωρίς να χρειάζεται τα μοτίβα να επαναλαμβάνονται επακριβώς. Στο τρίτο επίπεδο μπορούν να αναγνωριστούν σύνολα μοτίβων σε μεγάλους συνδυασμούς που αντιστοιχούν σε γνωστά αντικείμενα ή μέρη τους. Τα επόμενα επίπεδα, παρόμοια, εντοπίζουν πιο σύνθετα αντικείμενα ως συνδυασμούς απλούστερων μερών. Το βασικότερο στοιχείο του Deep Learning είναι πως τα επίπεδα που εντοπίζουν χαρακτηριστικά και δομές δεν είναι σχεδιασμένα από τους ανθρώπους: μαθαίνονται από τα δεδομένα χρησιμοποιώντας γενικευμένες διαδικασίες εκμάθησης.

Η χρήση του Deep Learning έχει βοηθήσει στην αντιμετώπιση προβλημάτων που δυσκόλευαν την κοινότητα της τεχνητής νοημοσύνης εδώ και χρόνια. Αποδεικνύεται να έχει επιδόσεις χωρίς προηγούμενο στον εντοπισμό πολύπλοκων δομών σε δεδομένα πολλών διαστάσεων και για αυτό είναι εφαρμόσιμο σε πολλούς διαφορετικούς τομείς, επιστημονικούς, επιχειρησιακούς και κοινωνικοπολιτικούς. Πέρα από επαναστατικές επιδόσεις στην αναγνώριση φωνής και εικόνας, έχει ξεπεράσει άλλες τεχνικές υπολογιστικής εκμάθησης στην πρόβλεψη συμπεριφοράς μορίων φαρμάκων, στην ανάλυση δεδομένων από επιταχυντές σωματιδίων, στην ανακατασκευή εγκεφαλικών κυκλωμάτων και στην πρόβλεψη των επιπτώσεων μεταλλάξεων μη κωδικοποιητικού DNA στις γονιδιακές εκφράσεις και ασθένειες. Ίσως, οι πιο αναπάντεχα υποσχόμενες επιδόσεις έγιναν στον κλάδο της επεξεργασίας φυσικής γλώσσας, συγκεκριμένα στην εντοπισμό θεμάτων, την ανάλυση συναισθήματος, τις ερωτήσεις - απαντήσεις και την μετάφραση.

2.2 Supervised Learning[;]

Η πιο συνήθης μορφή υπολογιστικής εκμάθησης, είτε Deep Learning είτε όχι, είναι αυτή της επιτηρούμενης εκμάθησης. Ας θεωρήσουμε πως θέλουμε να φτιάξουμε ένα σύστημα που αποφασίζει τι περιέχει μια εικόνα, όπως ένα σπίτι, ένα αυτοκίνητο, έναν άνθρωπο ή μία γάτα. Αρχικά, συλλέγουμε ένα αρκετά μεγάλο σύνολο δεδομένων με εικόνες στα οποία σημειώνεται τι αντικείμενο από τα παραπάνω περιέχει κάθε εικόνα. Κατά τη διάρκεια της εκπαίδευσης, δείχνουμε στο σύστημα μια εικόνα και αυτό παράγει μία πρόβλεψη, στη μορφή ενός διανύσματος με σκορ για κάθε κατηγορία. Θέλουμε η επιθυμητή κατηγορία να έχει το μεγαλύτερο σκορ πρόβλεψης, αλλά αυτό είναι πολύ δύσκολο πριν την εκπαίδευση. Υπολογίζουμε μία αντικειμενική συνάρτηση με την οποία μετράμε το λάθος (ή την απόσταση) μεταξύ των αποτελεσμάτων του συστήματος και τον επιθυμητών αποτελεσμάτων. Το σύστημα, ύστερα, προσαρμόζει τις εσωτερικές του παραμέτρους ώστε να μειώσει το λάθος. Οι εσωτερικές παράμετροι, που συχνότερα στη βιβλιογραφία απαντώνται ως βάρη, είναι πραγματικοί αριθμοί που ορίζουν την λειτουργικότητα εισόδου-εξόδου του συστήματος. Σε ένα τυπικό Deep Learning σύστημα, οι εσωτερικές παράμετροι και τα παραδείγματα που χρησιμοποιούμε για την εκμάθηση του συστήματος μπορεί να είναι εκατοντάδες εκατομμύρια σε αριθμό.

Για την κατάλληλη προσαρμογή των βαρών, ο αλγόριθμος εκμάθησης υπολογίζει ένα διάνυσμα κλίσης, για κάθε βάρος, που δείχνει κατά πόσο και προς ποια κατεύθυνση αλλάζει το λάθος αν αλλάξουμε απειροστά το αντίστοιχο βάρος. Το διάνυσμα των βαρών τελικά ρυθμίζεται έτσι ώστε να έχει αντίθετη φορά με το διάνυσμα κλίσης. Η διαδικασία αυτή είναι μία προσπάθεια ελαχιστοποίησης της συνάρτησης λάθους και μεταγενέστερα της μείωσης, κατά μέσο όρο, των λαθών προβλέψεων του συστήματος.



Σχήμα 2.1: Ένα τυπικό δομικό διάγραμμα επιτηρούμενης εκμάθησης.

Στην πλειοψηφία της σύγχρονης βιβλιογραφίας, και στην παρούσα διπλωματική, ο αλγόριθμος ελαχιστοποίησης που χρησιμοποιείται είναι ο stochastic gradient descent (SGD). Αυτός συνίσταται από την επίδειξη λίγων κάθε φορά, σωστά επισημασμένων, παραδειγμάτων στο σύστημα, τον υπολογισμό των προβλέψεων και του λάθους, τον υπολογισμό του διανύσματος κλίσης και την ρύθμιση των βαρών. Η παραπάνω διαδικασία επαναλαμβάνεται για πολλά μικρά σετ παραδειγμάτων, μέχρι η συνάρτηση στόχου να σταματήσει να μειώνεται. Μετά την εκπαίδευση, οι επιδόσεις του συστήματος μετρώνται σε ένα σύνολο διαφορετικών παραδειγμάτων, έτσι ώστε να εξεταστεί η ικανότητα γενίκευσης του συστήματος σε εισόδους που βλέπει για πρώτη φορά.

2.3 Recurrent Neural Networks

Τα αναδραστικά νευρωνικά δίκτυα (Recurrent Neural Networks - RNNs) είναι μία προσαρμογή των κλασσικών, πλήρως συνδεδεμένων νευρωνικών δικτύων, έτσι ώστε τα πρώτα να μπορούν να διαχειριστούν ακολουθίες. Σε κάθε χρονική στιγμή, τα RNNs δέχονται μια είσοδο, ενημερώνουν την εσωτερική τους κατάσταση και παράγουν μία έξοδο. Η πολυδιάστατη εσωτερική κατάσταση, που συχνά απαντάται στη βιβλιογραφία ως κρυφή κατάσταση (hidden state), και η μη γραμμική εξέλιξη της διαχειριζόμενης πληροφορίας δίνουν στα αναδραστικά νευρωνικά δίκτυα μεγάλη εκφραστική ευελιξία και δυνατότητα ενσωμάτωσης και διατήρησης

της πληροφορίας σε μεγάλα χρονικά διαστήματα. Ακόμα και όταν η μη γραμμική συνάρτηση που χρησιμοποιείται από κάθε στοιχείο του PNN είναι εξαιρετικά απλή, η χρήση της σε πολλά επίπεδα και η επανάληψη της σε κάθε χρονική στιγμή οδηγεί σε ένα εξαιρετικά δυναμικό σύστημα.

Τα αναδραστικά νευρωνικά δίκτυα ορίζονται ως εξής: δεδομένης μιας ακολουθίας διανυσμάτων εισόδου $(x_1, x_2, ..., x_T)$, το σύστημα υπολογίζει μία ακολουθία κρυφών κατάστασεων $(h_1, h_2, ..., h_T)$ και μία παράγει μια ακολουθία εξόδων $(o_1, o_2, ..., o_T)$, σύμφωνα με τον κάτωθι αλγόριθμο:

Algorithm 1 RNN

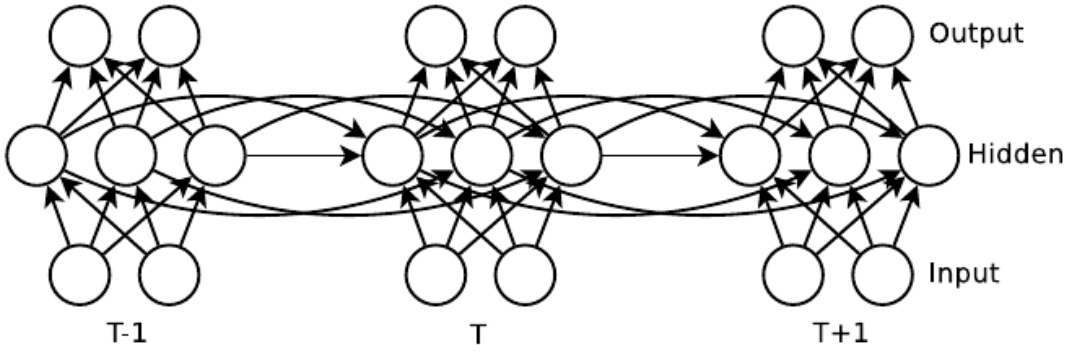
for $t = 1$ to T **do**

$$h_t = \tanh(W_h x_t + W_{hh} h_{t-1} + b_h) \tag{2.1}$$

$$o_t = W_{oh} h_t + b_o \tag{2.2}$$

end for

Σε αυτές τις εξισώσεις, το W_{hx} είναι ο πίνακας βαρών από την είσοδο στην κρυφή κατάσταση, το W_{hh} είναι ο πίνακας απο την κρυφή κατάσταση στην κρυφή κατάσταση, το W_{oh} είναι ο πίνακας βαρών απο την κρυφή κατάσταση στην έξοδο και τα b_h, b_o είναι οι σταθεροί όροι. Η μη ορισμένη σχέση $W_{hh} h_{t-1}$ στη χρονική στιγμή $t = 1$ αντικαθιστάται με ένα διάνυσμα αρχικοποίησης, h_{init}, και η συνάρτηση της υπερβολικής εφαπτομένης, tanh, εφαρμόζεται κατά στοιχείο.



Σχήμα 2.2: Ένα αναδραστικό νευρωνικό δίκτυο είναι ένα πολύ "βαθύ" πλήρως συνδεδεμένο νευρωνικό δίκτυο του οποία τα βάρη επαναχρησιμοποιούνται στις διάφορες χρονικές στιγμές. Η μη γραμμική συνάρτηση ενεργοποίησης που χρησιμοποιεί η κρυφή κατάσταση είναι η πηγή της πλούσιας δυναμικής του συστήματος.

Οι παράγωγοι των στοιχειδών μερών του δικτύου είναι εύκολο να υπολογιστούν, με τη μέθοδο της προς τα πίσω διάδοσης σφάλματος, [;], [;] οπότε ίσως η εκπαίδευση ενός τέτοιου συστήματος φαίνεται εύκολη. Στην πραγματικότητα, η σχέση μεταξύ των παραμέτρων του PNN και της δυναμικής του είναι εξαιρετικά ασταθής, γεγονός που καθιστά τον αλγόριθμο SGD αναποτελεσματικό. Αυτό τεκμηριώνεται απο [;] και [;] που αποδεικνύουν οτι τα διανύσματα κλίσεων τείνουν να μηδενίζονται (ή σπανιότερα να απειρίζονται) εκθετικά με την

διάδοση του σφάλματος στο χρόνο. Στη σχετική βιβλιογραφία αυτό απαντάται ως πρόβλημα εξαφάνισης ή έκρηξης των διανυσμάτων κλίσης ("vanishing or exploding gradients problem") Το παραπάνω χρησιμοποιήθηκε ως επιχείρημα για το οτι τα αναδραστικά νευρωνικά δίκτυα δεν μπορούν να αποτυπώσουν εξαρτήσεις με μεγάλη χρονική απόσταση μεταξύ τους, οταν ο χρησιμοποιείται ο αλγόριθμος SGD. Επιπρόσθετα, ο περιστασιακός απειρισμός των διανυσμάτων κλίσης αυξάνει τη διακύμανση τους και κάνει την εκμάθηση ασταθή. Τα θεωρητικά αποτελέσματα αυτά, δεδομένου πως ο SGD ήταν ο βασικότερος αλγόριθμος εκπαίδευσης νευρωνικών δικτύων, σε συνδυασμό με την εμπειρική δυσκολία εκπαίδευσης των RNNs οδήγησε στη σχεδόν ολοκληρωτική εγκατάλειψη της σχετικής έρευνας.

2.4 Εκπαίδευση των Recurrent Neural Networks

2.4.1 Long Short-Term Memory Units[;]

Ένας τρόπος να αντιμετωπιστεί η αδυναμία που παρουσιάζουν τα PNNs στη δυσκολία εκμάθησης δομών με μακρινές, στο χρόνο, αλληλεξαρτήσεις είναι η τροποποίηση του μοντέλου ώστε να έχει στοιχεία με μνήμη. Η προσέγγιση αυτή ονομάζεται Long Short-Term Memory και γνωρίζει ευρεία χρήση. Οι σχέσεις που ορίζουν κάθε στοιχείο μνήμης είναι:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.3}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2.4}$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}h_{t-1} + W_{cf}c_{t-1} + b_c\right) \tag{2.5}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{2.6}$$

$$h_t = o_t \tanh c_t \tag{2.7}$$

Το σ είναι η σιγμοειδής συνάρτηση, i, f, o και c είναι αντίστοιχα η πύλη εισόδου, η πύλη απώλειας μνήμης, η πύλη εξόδου και η μνήμη. Τα τελευταία είναι διανύσματα με διαστάσεις ίδιες με του διανύσματος h (βλ. εξίσωση 2.1, που ταυτίζεται με τον δέυτερο όρο της εξίσωσης 2.5). Οι δείκτες των πινάκων βαρών W έχουν το προφανές νοήμα, για παράδειγμα ο W_{hi} είναι ο πίνακας βαρών κρυφής κατάστασης – εισόδου, ο W_{xo} είναι πίνακας βαρών εισόδου – εξόδου κ.ο.κ. Οι πίνακες βαρών από την μνήμη στις πύλες είναι διαγώνιοι, έτσι το στοιχείο m σε κάθε πύλη δέχεται είσοδο μόνο από το στοιχείο m του διανύσματος μνήμης.

Μια πιο διαισθητική εξήγηση του συστήματος LSTM είναι η εξής: Η μνήμη c, σε κάθε επανάληψη της λειτουργίας του αναδραστικου νευρωνικού δικτύου, αλλάζει δυναμικά. Μέσω της πύλης απώλειας μνήμης f αρχικά αποφασίζεται πιο κομμάτι της υπάρχουσας πληροφορίας της c θα κρατήσουμε "κοιτώντας" την είσοδο x_t και την προηγούμενη κατάσταση h_{t-1}. Ύστερα η πύλη εισόδου αποφασίζει πιο κομμάτι της εισόδου θα αποθηκευτεί. Αποθηκεύεται η καινούρια μνήμη συνδυάζωντας τις αποφάσεις τον προηγούμενων βημάτων. Τέλος η πύλη εξόδου αποφασίζει πιο κομμάτι της μνήμης θα εξάχθεί.

2.4.2 Truncated Backpropagaion Through Time

Ένα απο τα βασικά προβλήματα του αλγορίθμου της προς τα πίσω διάδοσης του σφάλματος, είναι το υψηλό κόστος για την ενημέρωση μιας μεμονωμένης παραμέτρου, γεγονός που την καθιστά απαγορευτική για πολλές επαναλήψεις. Για παράδειγμα, ο υπολογισμός του διανύσματος κλίσεων ενός RNN ακολουθιών μήκους 1000 στοιχειών, στοιχίζει όσο και το εμπρόσθιο και προς τα πίσω πέρασμα ενός πλήρως συνδεδεμένου νευρωνικού δικτύου 1000 επιπέδων. Το υπολογιστικό κόστος μπορεί να μειωθεί με μία μέθοδο που χωρίζει την ακολουθία 1000 στοιχείων σε, για παράδειγμα, 50 ακολουθίες μήκους 20 στοιχείων η καθεμία και τις αντιμετωπίζει ως ξεχωριστά παραδείγματα εκπαίδευσης. Αυτή η απλή προσέγγιση μπορεί να εκπαιδεύσει το νευρωνικό δίκτυο ικανοποιητικά, αλλά αδυνατεί να αποτυπώσει σχέσεις που εκτείνονται παραπάνω απο 20 χρονικές στιμές. Ο αλγόριθμος Τρυνςατεδ Βασπροπαγατιον Τηρουγη Τιμε είναι μία συναφής μέθοδος. Έχει το ίδιο κόστος με την απλή μέθοδο που περιγράψαμε παραπάνω αλλά είναι πιο ικανός στο να αποτυπώνει χρονικές εξαρτήσεις μεγάλου μήκους. Επεξεργάζεται την ακολουθία ένα στοιχείο τη φορά, και κάθε k_1 στοιχεία, καλεί τον αλγόριθμο BPTT για k_2 στοιχεία, έτσι η ενημέρωση των παραμέτρων είναι πιο φθηνη επεξεργαστικά αν το k_2 είναι αρκούντως μικρό. Συνεπώς, η κρυφή κατάσταση εκτίθεται σε πολλά στοιχεία και μπορεί να περιέχει χρήσιμη πληροφορία για το παρελθόν της ακολουθίας γεγονός το οποίο μπορούμε να εκμεταλευτούμε. Ο αλγόριθμος Truncated Backpropagation Through Time:

Algorithm 2 Truncated Backpropagation Through Time

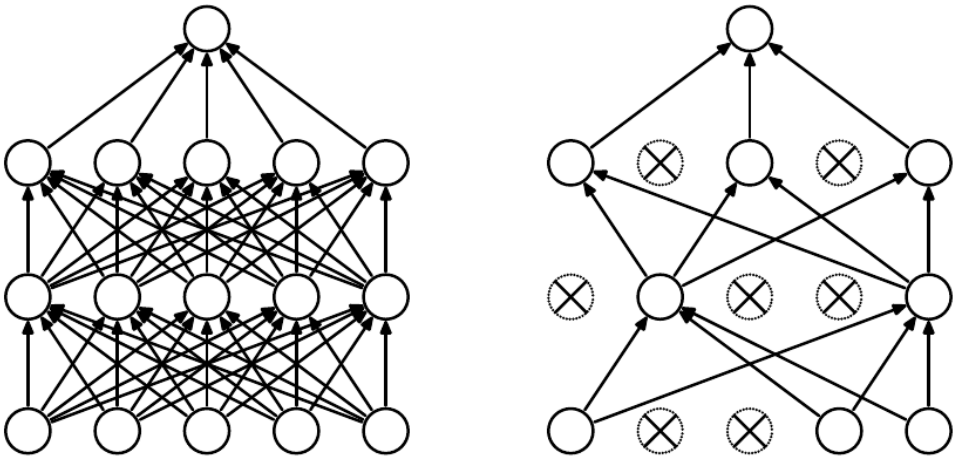
for $t = 1$ to T **do**
  RNN iteration
  if t divides k_1 **then**
    BPPT from t to $t - k_2$
  end if
end for

2.4.3 Dropout[;]

Τα βαθιά νευρωνικά δίκτυα περιέχουν πολλαπλά μη γραμικά επίπεδα και, όπως είδαμε, αυτό τα κάνει εξαιρετικά εκφραστικά μοντέλα που μπορούν να μάθουν περίπλοκες σχέσεις μεταξύ εισόδου και εξόδου. Με περιορισμένα, όμως, δεδομένα εκπαίδευσης πολλές απο τις σχέσεις που αποτυπώνονται μπορεί να είναι αποτέλεσμα θορύβου δειγματοληψίας και έτσι θα υπάρχουν στα δεδομένα εκπαίδευσης και όχι στα πραγματικά δεδομένα ελέγχου επιδόσεων του μοντέλου, ακόμα και αν είναι βασισμένα στην ίδια κατανομή. Αυτό, όπως γνωρίζουμε, οδηγεί στο overfitting και διάφοροι μέθοδοι έχουν αναπτυχθεί για την αντιμετώπισή του, όπως οι L1 και L2 κανονικοποίηση.

Μία τέτοια τεχνική κανονικοποίησης είναι και το Δροπουτ. Συνοπτικά, μας δίνει τη δυνατότητα να συνδυάσουμε προσεγγιστικά εκθετικά πολλές διαφορετικές αρχιτεκτονικές νευρωνικών δικτύων, αποτελεσματικά. Ο όρος δροπουτ, του οποιού η ελληνική μετάφραση είναι

"αυτός που αποσύρεται", αναφέρεται στην παράλειψη στοιχείων του νευρωνικού δικτύου. Παραλείποντας ένα στοιχείο εννοούμε την προσωρινή του αφαίρεση από το δίκτυο, μαζί με τις συνδέσεις απο και προς αυτό. Η επιλογή των στοιχείων που παραλείπονται είναι τυχαία. Στην πιο απλή εφαρμογή, κάθε στοιχείο κρατείται με μία πιθανότητα p που είναι σταθερή και ανεξάρτητη των υπολοίπων στοιχείων.



Σχήμα 2.3: Δροπουτ ςαπτιον.

Η χρήση της τεχνικής dropout αναλογεί στη χρήση διαφόρων "αραιωμένων" δικτύων που βασίζονται στο αρχικό. Το αραιωμένο δίκτυο αποτελείται απο τα στοιχεία που "επέζησαν" της χρήσης dropout (βλ. 2.3) και για κάθε παράδειγμα απο το σετ εκπαίδευσης επιλέγεται τυχαία ένα αραιωμένο δίκτυο. Έτσι, νευρωνικό δίκτυο με n στοιχεία μπορεί να θεωρηθεί μια συλλογή από 2^n πιθανά αραιωμένα νευρωνικά δίκτυα τα οποία μοιράζονται βάρη με το αρχικό, και η εκπαίδευση του αρχικού ανάγεται στην εκπαίδευση της συλλογής αραιωμένων δικτύων. Για την εκτίμηση των επιδόσεων του συστήματος χρησιμοποιούμε όλα τα στοιχεία του νευρωνικού δικτύου, αλλα τα εξερχόμενα βάρη τους είναι πολλαπλασιασμένα με την αρχική πιθανότητα p να κρατηθούν στη διαδικασία της εκπαίδευσης.

Κεφάλαιο 3

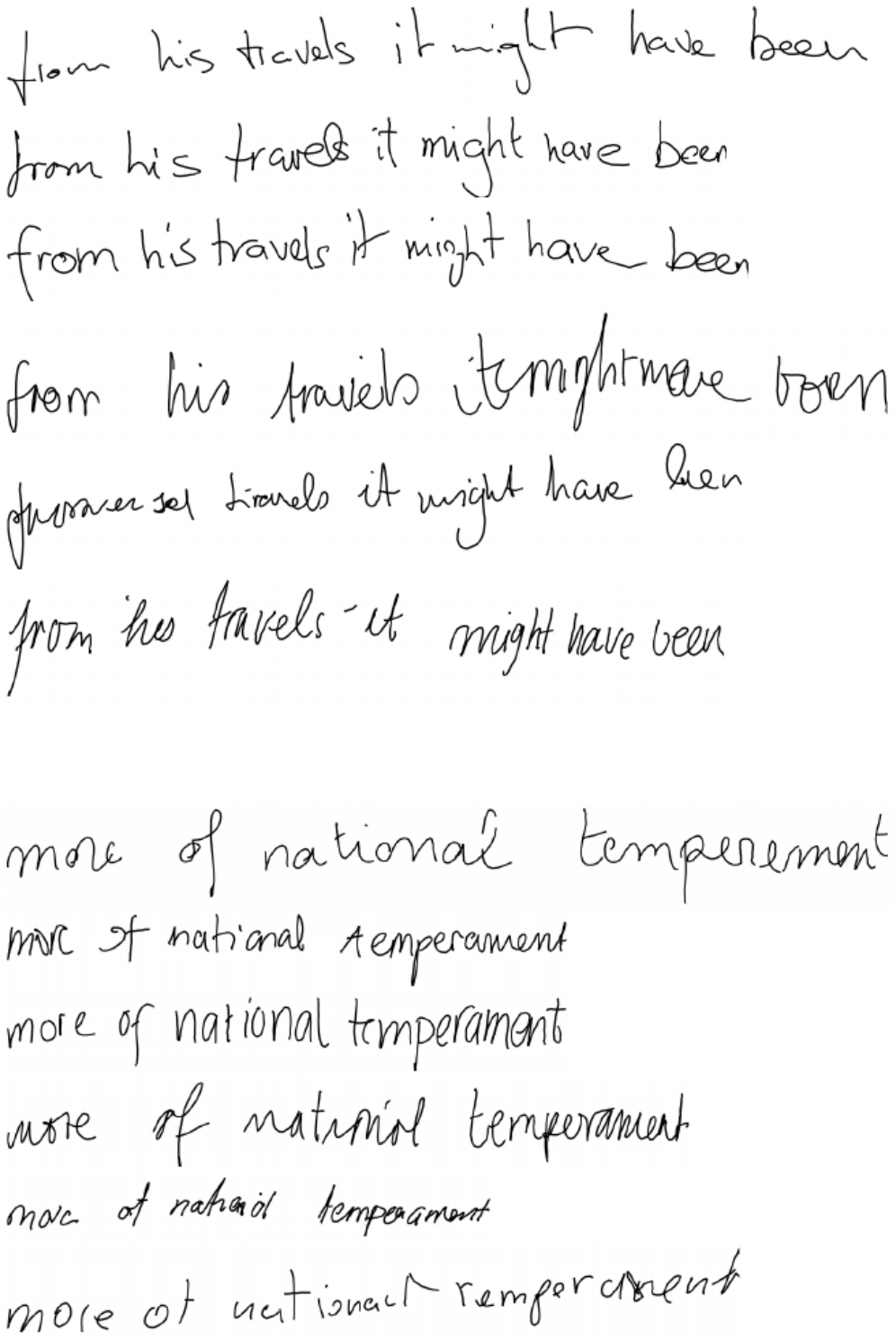
Σχετική βιβλιογραφία

Στο κεφάλαιο αυτό παρουσιάζουμε σχετικές προσεγγίσεις και υλοποιήσεις στο πρόβλημα του αυτόματου προγραμματισμού και της παραγωγής κώδικα. Επειδή είναι πρακτικά αναρίθμητες, θα επικεντρωθούμε σε αυτές που είναι σχετικές με τα αναδραστικά νευρωνικά δίκτυα και σε κάποιες που παρουσιάζουν ιδιαίτερο ενδιαφέρον.

3.1 Generating Sequence with Recurrent Neural Networks

Στην εργασία των Graves et al. παρουσιάζεται πως απλές δομές αναδραστικών νευρωνικών δικτύων με στοιχεία μνήμης LSTM μπορούν να χρησιμοποιοηθούν για να παράξουν σύνθετες ακολουθίες, απλά προβλέποντας ένα στοιχείο της ακολουθίας τη φορά. Θεωρώντας τις προβλέψεις στοχαστικές, καινούριες ακολουθίες μπορούν να προκύψουν απο ένα εκπαιδευμένο δίκτυο, δειγματοληπτώντας επαναληπτικά από την έξοδο του δικτύου και ύστερα ξαναδίνοντας ως είσοδο στο δίκτου την δειγματοληπτημένη πρόβλεψη. Με μία διαφορετική διατύπωση, αφήνουμε το δίκτυο να αντιμετωπίσει τις επινοήσεις του ως αληθινές, περίπου σαν έναν άνθρωπο ο οποίος ονειρεύεται. Αν και το σύστημα είναι ντερεμινιστικό, η στοχαστικότητα που εισάγεται δειγματοληπτώντας δημιουργεί μία κατανομή σε σχέση με τις ακολουθίες. Αυτή η κατανομή είναι δεσμευμένη, αφού η εσωτερική αναπαράσταση του δικτύου, άρα και κατανομή προβλέψεων του, εξαρτάται από τις προηγούμενες εισόδους.

Η προσέγγιση αυτή επιδεικνύεται για κείμενο (όπου οι τιμές είναι διακριτές) και για "online" χειρόγραφο κείμενο (όπου οι τιμές είναι πραγματικές). Με τον όρο "online" εννοούμε οτι η γραφή αποτυπώνεται ως ακολουθία διάνυσματων θέσης ενός μολυβιού – σε αντίθεση με το "offline" στο οποίο έχουμε διαθέσιμη ολόκληρη την εικόνα του χειρόγραφου. Το σύστημα που χρησιμοποιείται είναι μια συστάδα που αποτελείται απο 7 επίπεδα αναδραστικών νευρωνικών δικτυών με 700 στοχεία μνήμης LSTM. Για την παραγωγή ακολουθιών κειμένου χρησιμοποιούνται τρία διαφορετικά σετ δεδομένων. Το Πενν Τρεεβανκ και το Ωικιπεδια Ηυττερ Πριζε για το γραπτό κείμενο και το ¡¡IAM online handwriting database¿¿. Το μοντέλο καταφέρνει να παράξει ακολουθιές τόσο ρεαλιστικές ώστε να είναι συχνά δύσκολο να τις ξεχωρίσει κανείς από πραγματικές, τουλάχιστον σε πρώτη όψη. Στα αποτελέσματα είναι ορατή μια μεγάλης εμβέλειας δομή και συνοχή.



Σχήμα 3.1: Ένα τυπικό δομικό διάγραμμα επιτηρούμενης εκμάθησης.

Επιπρόσθετα, βασισμένοι στην προηγούμενη δομη, οι Graves et al. σχεδιάζουν ένα σύστημα παραγωγής χειρόγραφου κειμένου το οποίο μπορεί να γράψει αυτό που του ζητάμε. Αυτό γίνεται με την προσθήκη ενός διανύσματος της πρότασης που θέλουμε να γράψουμε, το οποίο δίνεται στο σύστημα πρόβλεψης την ώρα της παραγωγής, αφού προφανώς εκπαιδευτεί σε σχετικά προβλήματα. Η απόφαση για το πότε και πως θα γραφεί κάθε χαρακτήρας αφήνεται στο νευρωνικό δίκτυο και τα αποτελέσματα είναι αρκετά ικανοποιητικά ώστε να είναι και πάλι δύσκολο να διακριθεί αν τα "χειρόγραφα" ανήκουν σε κάποιον άνθρωπο ή στο σύστημα.

3.2 Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets

Οι Joulin et al. στην έρευνα τους εξετάζουν τα όρια των "state of the art" Deep Learning προσεγγίσεων. Πιο συγκεκριμένα, εξετάζονται τα απλούστερα προβλήματα πρόβλεψης ακολουθιών που είναι πέρα από τις δυνατότητες εκμάθησης των τυπικών αναδραστικών δικτύων: αλγοριθμικά παραγμένες ακολουθίες που μπορούν να μαθευτούν μόνο απο συστήματα με δυνατότητα μνήμης και απαρίθμησης. Για παράδειγμα η σχέση $a^n b^n, n > 0$ μπορεί να παράξει την ακολουθία aab**ba**aab**bba**b**a**aaaab**bbbb**, όπου με έντονη γραμματοσειρά σημειώνονται τα στοιχεία της ακολουθίας που μπορούν να προβλεφθούν ντετερμενιστικά.

Εξετάζονται 4 διαφορετικά μοντέλα: ένα απλό αναδραστικό νευρωνικό δίκτυο, ένα RNN με στοιχεία μνήμης LSTM και 2 RNN με εξωτερική μνήμη. Η εξωτερική μνήμη είναι για το ένα μοντέλο μια διπλά συνδεδεμένη λίστα και για το άλλο μοντέλο μια στοίβα. Όλα τα μοντέλα εκπαιδεύονται με τον αλγόριθμο SGD. Τα μοντέλα με την εξωτερική μνήμη μαθαίνουν να χρησιμοποιούν τις θεωρητικά απείρου μήκους εξωτερικές μνήμες τους, με στοιχειώδεις εντολές (push, pop, insert, no-op).

Τελικώς δείχνουν πως μερικοί βασικοί αλγόριθμοι μπορούν να μαθευτούν απο ακολουθιακά δεδομένα χρησιμοποιώντας RNNs με μνήμη. Τα μοντέλα με εξωτερική μνήμη ξεπερνούν σε επιδόσεις τα υπόλοιπα μοντέλα. Οι συγγραφείς υποσημειώνουν πως είναι σημαντικό να μεγαλώσουμε την πολυπλοκότητα του μοντέλου με δομημένο τρόπο και πως η δομή των νευρωνικών δικτύων θα πρέπει να μαθαίνεται απο τα δεδομένα και να μην προαποφασίζεται.

3.3 A Synthetic Neural Model for General Purpose Code Generation

Στην έρευνα τους, οι Yin et al. ασχολούνται με την αυτόματη μετατροπή εντολών φυσικής γλώσσας σε πηγαίο κώδικα γλωσσών γενικής χρήσης. Σε αντίθεση με την πλειοψηφία των μεθόδων που απαντώνται στην βιβλιογραφία, που αντιμετωπίζουν το πρόβλημα χωρίς να λαμβάνουν υπ' όψιν την γραμματική της τελικής γλώσσας, οι ερευνητές προτείνουν ένα μοντέλο στο οποίο η γραμματική είναι γνωστή a priori.

Το συντακτικά-οδηγούμενο νευρωνικό μοντέλο παραγωγής κώδικα που προτείνεται βασίζεται σε ένα γραμματικό μοντέλο που ορίζει την παραγωγή ενός Αβστραχτ Σψνταξ Τρεε σε

ακολουθίες στοιχειωδών δράσεων. Οι δράσεις αυτές χωρίζονται κανόνες παραγωγής κώδικα και σε εντολές. Με αυτό τον τρόπο το μοντέλο δε χρειάζεται να μάθει την γραμματική απο τα περιορισμένα σε ποσότητα δεδομένα εκμάθησης. Το αναδραστικό νευρωνικό δίκτυο που χρησιμοποιείται βασίζεται σε στοιχεία LSTM με τροποποίηση, ώστε να λαμβάνεται υπ΄ όψιν η αναδρομική φύση των γλωσσών προγραμματισμού. Η δομή του συστήματος γίνεται σύμφωνα με αρχιτεκτονική encoder-decoder RNN with attention [;], τεχνική η οποία γνωρίζει μεγάλη χρήση και επιτυχία τα λίγα χρόνια ύπαρξης της. Για την εκπαίδευση "δείχνουμε" στο νευρωνικό κομμάτια κώδικα, μετατρέπονται σε ΑΣΤς και από εκεί σε κώδικα σύμφωνα την γραμματική που υποδεικνύεται.

Το μοντέλο ξεπερα τις state of the art προσεγγίσεις νευρωνικών δικτύων των Λινγ και των Δονγ[] ανδ Λαπατα[] στο Ηεαρτηστονε δαταση με παραγώμενη γλώσσα την Πψτηον. Συμπεραίνεται έτσι, η σημαντικότητα της γραμματικής της γλώσσας σε σχέση με τις επιδόσεις.

3.4 End-to-End Memory Networks

Οι Συκηβααταρ ετ αλ., παρουσιάζουν ένα ευέλικτο νευρωνικό μοντέλο με μεγάλη εξωτερική μνήμη. Το μοντέλο σε αντίθεση με αντίστοιχες εργασίες δικτύων με μνήμη εκπαιδεύεται "end-to-end, που, στα πλαίσια της εκπαίδευσης μοντέλων νευρωνικών δικτύων, σημαίνει πως το μοντέλο εκπαιδεύεται σε μια ενιαία διαδικασία και απλα του δίνονται οι είσοδοι και οι σωστές έξοδοι, χωρίς επιπρόσθετη εργασία για δημιουργία και ρύθμιση χαρακτηριστικών. Οι επιδόσεις του συστήματος εξετάζονται σε προβλήματα συνθετικών ερωταπαντήσεων και σε προβλήματα μοντελοποίησης φυσικής γλώσσας.

Το σύστημα δέχεται ένα σετ εισόδων, μια ερώτηση και εξάγει μία απάντηση. Το σετ εισόδων αποθηκεύεται στη μνήμη σε μορφή εσωτερικών αναπαραστάσεων. Για κάθε ερώτηση υπολογίζεται ένας δείκτης που εκφράζει κατά πόσο αντιστοιχεί η ερώτηση με τα στοιχεία της μνήμης. Από τις εισόδους, επιπρόσθετα, υπολογίζεται και μία αναπαράσταση της αναμενόμενης εξόδου. Η τελευταία σε συνδυασμό με τον δείκτη συσχέτισης ερώτησης-μνήμης χρησιμοποιείται για την εξαγωγή της τελικής απάντησης. Ολόκληρο το σύστημα είναι παραγωγίσιμο, οπότε μπορούμε να χρησιμοποιήσουμε τις τυπικές μεθόδους για την εκμάθηση του.

Για να εξετάσουμε τις επιδόσεις στην μοντελοποίηση φυσικής γλώσσας (με την οποία ασχολούμαστε επειδή βρίσκεται πιο κοντά στο πρόβλημα του αυτόματου προγραμματισμού) χρησιμοποιούμε τα Πενν Τρεεβανκ Δαταση και Τεξτ8 δαταση. Το δίκτυο μνήμης το οποίο εξετάσαμε ξεπερνά σε επιδόσεις διατάξεις RNN και LSTM. Αξιοσημειώτο είναι πως το νευρωνικό μοντέλο μνήμης έχει σημαντικά λιγότερες παραμέτρους από το αντίστοιχο LSTM. Σε ακόμα ένα πείραμα, έτσι, υποδεικνύεται η σημαντικότητα ύπαρξης εξωτερικής μνήμης στις διατάξεις εκμάθησης.

3.5 Neuro Symbolic Program Synthesis

Στην έρευνα τους οι Παρισοττο ετ αλ. ασχολούνται με ένα νευρωνικό μοντέλο σύνθεσης προγραμμάτων με σκοπό την επεκτασιμότητα και την έυκολη εξέταση της ορθότητας του παραγώμενου μοντελου. Σε αντίθεση με την πλειοψηφία των προσεγγίσεων στη σύγχρονη βιβλιογραφία, όπου ο χώρος αναζήτησης είναι σύμβολα της γλώσσας την οποία παράγουμε, εδώ, ο χώρος αναζήτησης είναι υποπρογράμματα που μαθαίνει το σύστημα κατά τη διάρκεια της μαθησης. Το όνομα που δίνεται στο υποσύστημα παραγωγής είναι Recursive-Reverse-Recursive Neural Network (R3NN).

Το υποσύστημα παραγωγής εξάγει αναπαραστάσεις υποπρογραμμάτων σε μορφές δέντρων, στις οποίες κάθε στοιχείο είναι είτε κανόνας παραγωγής είτε σύμβολο, διαδικασία η οποια χωρίζεται σε 3 μέρη. Αρχικά δεδομένου ενός τέτοιου δέντρου, δίνεται ένα διάνυσμα αναπαράστασης σε κάθε φύλλο του. Ύστερα, το δέντρο διαβάζεται προς τα πάνω, ώστε να δωθεί μία αναπαράσταση ολόκληρου του δέντρου στη ρίζα του. Τέλος επαναλαμβάνεται το προς τα κατω πέρασμα ώστε να δωθεί σε κάθε φύλλο μια αναπαράσταση ολόκληρου του δέντρου. Με αυτό τον τρόπο κάθε φύλλο έχει πληροφορία για τα υπόλοιπα φύλλα και για την συνολική λειτουργικότητα του δέντρου. Τα προγράμματα στο σετ δεδομένων χωρίζονται σε στοιχειώδη βήματα για να επεξεργαστούν με τον τρόπο που περιγράψαμε παραπάνω. Τα δεδομένα εκπαίδευσης σε αυτή την περίπτωση αποτελούνται από αναπαραστάσεις εισόδων και εξόδων που δίνονται στο σύστημα παραγωγής με σε κάθε φύλλο του δέντρου.

Το σύστημα εξετάζεται στη δημιουργία προγραμμάτων διαχείρισης αλφαριθμητικών ακολουθιών. Καταφέρνει σε ένα βαθμό και να επεκτείνει προγράμματα που ήδη έχει "δει" ωστε να συμπεριλαμβάνουν καινούρια ζευγάρια εισόδου εξόδου αλλά και να δημιουργήσει καινούρια προγράμματα για καινούριου είδους ζευγάρια εισοδου εξόδου. Η επεκτασιμότητα που παρουσιάζει το σύστημα αυτό, με την έννοια οτι μπορεί να ξεκινήσει από κάποια προγράμματα και ύστερεα να τα εμπλουτίσει είναι ενα σημαντικό και αισιόδοξο στοιχείο στην κατεύθυνση του αυτόματου προγραμματισμού.

Κεφάλαιο 4

Μεθοδολογία

Στο κεφάλαιο αυτό περιγράφεται η προσέγγιση μας στην παραγωγή κώδικα χρησιμοποιώτας αναδραστικά νευρωνικά δίκτυα. Εμπνεόμαστε απο το βλογ ποστ του Andrej Karpathy, στο οποίο χρησιμοποιείται μια σχετικά απλή δομή PNN με ΛΣΤΜ στοιχεία η οποία εκπαιδεύεται στα έργα του Σηακεσπεαρε, κατά χαρακτήρα, και παράγει παρόμοιο κείμενο. Χρησιμοποιούμε το ίδιο μοντέλο, εκπαιδευμενο σε κώδικα javascript. Με σκοπό να βελτιώσουμε τις επιδόσεις πρόβλεψης του μοντέλου και να εξετάσουμε τη διαίσθηση οτι με περισσότερη χρήσιμη πληροφορία ο παραγώμενος κώδικας θα είναι ποιοτικότερος, προτείνουμε μία επέκταση του προηγούμενου μοντέλου που χρησιμοποιεί a priori γνώση για τον κώδικα. Εξετάζουμε τα μοντέλα σε 2 διαφορετικά σετ δεδομένων. Παρακάτω ακολουθεί αναλυτική παρουσίαση της μεθόδου, την οποία χωρίζουμε σε προ-επεξεργασία (pre-processing), εκπαίδευση (training) και παραγωγή κώδικα (Source Code Generation).

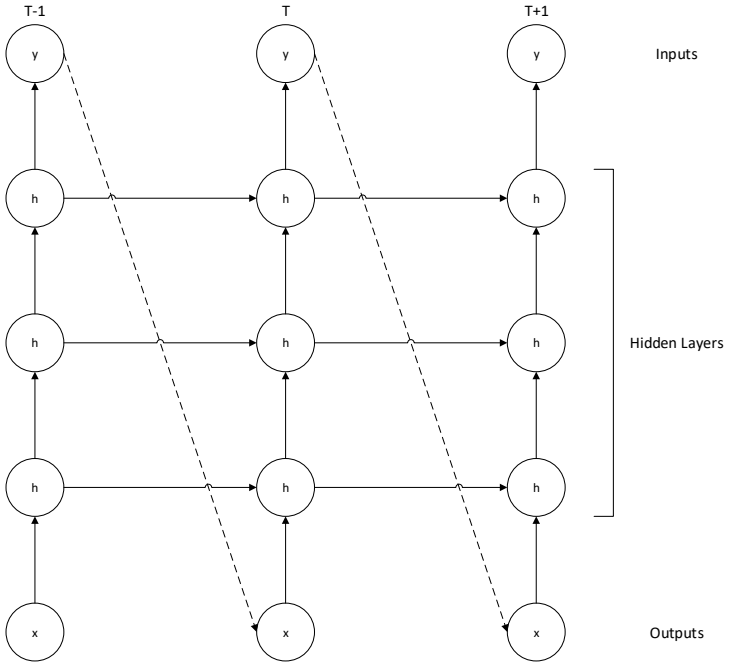
4.1 Τα μοντέλα

4.1.1 Recurrent Neural Networks as Generative Models

Ο στόχος της μοντελοποίησης γλώσσας κατα χαρακτήρα (χωρίς να αναφερόμαστε απαραίτητα στην προγραμματιστική γλώσσα) είναι να προβλέψει τον επόμενο χαρακτήρα σε μία ακολουθία. Δεδομένης μιας εκπαιδευτικής ακολουθίας $(x_1, x_2, ..., x_T)$, τα αναδραστικά νευρωνικά δίκτυα χρησιμοποιούν τις εξόδους τους $(o_1, o_2, ..., o_T)$ για να πάρουν κατανομές πρόβλέψεων της μορφής $P(x_{t+1}|x_{\leq t}) = P(softmax(o_t))$, όπου η κατανομή "softmax" ορίζεται: $P(softmax(o_t) = j) = exp(o_t^{(j)} / \sum_k exp(o_t^{(k)})$. Ο στόχος που χρησιμοποιείται για την μοντελοποίηση της γλώσσας είναι η μεγιστοποίηση της λογαριθμικής πιθανότητας της εκπαιδευτικής ακολουθίας $\sum_{t=0}^{T-1} logP(x_{t+1}|x_{\leq t})$. Όπως και στην εργασία των Γραες ετ αλ. [;], εισάγουμε στοχαστικότητα δειγματοληπτώντας απο την έξοδο του νευρωνικού δικτύου και δίνοντας την τυχαία επιλογή μας ως είσοδο, την επόμενη χρονική στιγμή.

4.1.2 Model char-rnn

Το πρώτο μοντέλο είναι ένα αναδραστικό νευρωνικό δίκτυο με 3 κρυμμένα επίπεδα στοιχείων LSTM. Δέχεται ακολουθιές χαρακτήρων κώδικα και εξάγει προβλέψεις για τα επόμενα στοιχεία τους. Η προβλέψεις του ςηαρ-ρνν έχουν μία διάσταση ίση με τον αριθμό διαφορετικών χαρακτήρων που υπάρχουν στο εκάστοτε σετ δεδομένων.



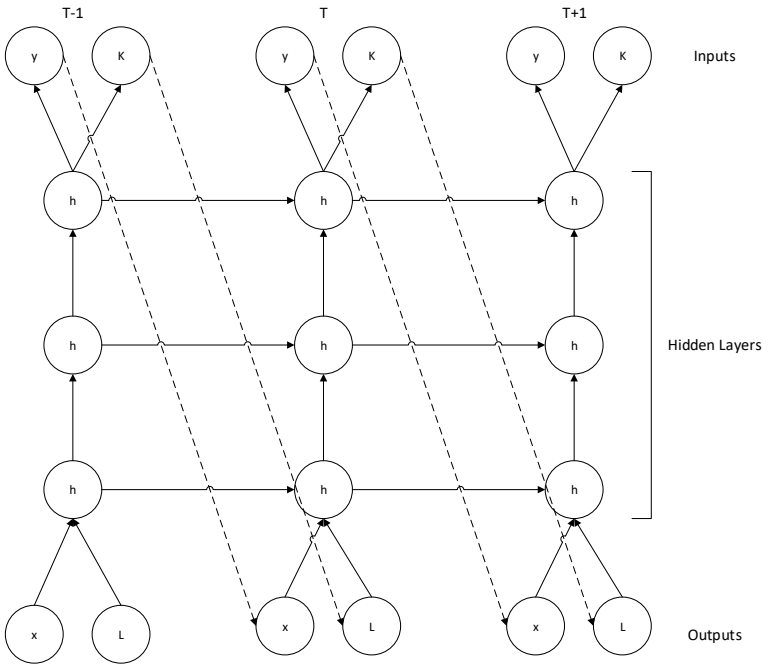
Σχήμα 4.1: Το μοντέλο char-rnn.

4.1.3 Model labeled-char-rnn

Το δεύτερο μοντέλο είναι επίσης ένα αναδραστικό νευρωνικό δίκτυο με 3 κρυμμένα επίπεδα στοιχείων LSTM. Εκτός απο ακολουθίες χαρακτήρων, το μοντέλο αυτό δέχεται και πληροφορία για το είδος του χαρακτήρα. Αντίστοιχα οι έξοδοί του, εκτός απο προβλέψεις για τον χαρακτήρα, περιέχουν και προβλέψεις για το είδος του χαρακτήρα. Με τον τρόπο αυτό θα εξετάσουμε κατά πόσο τα RNN μπορούν να εκμεταλλευτούν a priori γνώσεις για τον κώδικα. Σημειώνεται πως η συνάρτηση επιδόσεων αυτού το μοντέλου είνα γραμμικός συνδυασμός των επιμέρους επιδόσεων πρόβλεψης χαρακτήρα και είδους χαρακτήρα.

4.2 Pre-processing

Ο κορμός της διαδικασίας της προ-επεξεργασίας είναι ίδιος και για τα δύο σετ δεδομένων. Αρχικά αναζητούμε τα αρχεία με κατάληξη “.js” διασχίζοντας σειριακά όλους τους φακέλους



Σχήμα 4.2: Το μοντέλο labeled-char-rnn.

των προθεστς, εκτός απο αυτούς που αφορούν testing και localization. Ο έλεγχος για το τελευταίο γίνεται απλοικά, ελέγχουμε δηλαδή αν οι φάκελοι φέρουν τα συνήθη ονόματα που χρησιμοποιούνται για τέτοιου είδους φακέλους. Ο μη ενδελεχής έλεγχος καταφέρνει να αφαιρέσει την πλειοψηφία των επαναλαμβανόμενων αρχείων αφήνοντας ένα μικρό ποσοστό να περάσει. Αυτό έχει ως αποτέλεσμα να εμπλουτιστεί η εκπαίδευση του νευρωνικού, χωρίς όμως να μονοπωλείται το ενδιαφέρον από αρχεία που περιέχουν τετριμμένο κώδικα. Στη συνέχεια, με τη βοήθεια ενός εργαλείου ανάλυσης της σύνταξης και γραμματικής προγραμματιστικών γλωσσών (ονόματι linguist) προχωράμε στο περαιτέρω φιλτράρισμα αρχείων. Συγκεκριμένα, εξαιρούμε αρχεία που έχουν την κατάληξη “.js” αλλα δεν είναι αρχεία κειμένου και αρχεία που είναι αυτόματα παραγώμενα και αποτελούν παραπροϊόν της διαδικασίας ανάπτυξης λογισμικού σε θαασςριπτ.

Αφού επιλέξουμε τα αρχεία τα οποία θα αποτελούν το σετ δεδομένων μας προχωράμε στη διαδικασία της ελαχιστοποίησης του κώδικα (minification, minimisation). Η ελαχιστοποίηση κώδικα, είναι η διαδικασία αφαίρεσης περιττών χαρακτήρων από των πηγαίο κώδικα, χωρίς να αλλάζει η λειτουργικότητά του. Τέτοιοι χαρακτήρες είναι τα κενά, τα σύμβολα αλλαγής παραγράφου, τα σχόλια και άλλα. Εδώ χρησιμοποίηθηκε το εργαλείο jsmin. Με την επιλογή αυτή προσπαθούμε να αφαιρέσουμε την περιττή πληροφορία απο τα δεδομένα μας, ώστε να είναι πιο εύκολο για το μοντέλο να αποτυπώσει τις σημαντικές σχέσεις ανάμεσα στους δίαφορους χαρακτήρες. Μετά το minification προσθέτουμε 2 ειδικούς χαρακτήρες για την αρχή και το τέλος κάθε αρχείου. Σημειώνεται πως θεωρούμε πως τα αρχεία ειναι extended ASCII

Πίνακας 4.1: Παράδειγμα αντιστοιχείας χαρακτήρων με το είδος τους σε μια ακολουθία

String 1	v	a	r		a	=	1	;	f	u	n	c	t	i	o	n		f	(	A	)
Label 1	K	K	K	P	I	O	N	P	K	K	K	K	K	K	K	K	P	I	P	I	P
String 2	{	r	e	t	u	r	n		ˋ	o	k	ˊ	;	}	c	=	f	(	1	0	)
Label 2	P	K	K	K	K	K	K	P	S	S	S	S	P	P	I	O	I	P	N	N	P

κωδικοποιημένα και στην ουσία διαβάζουμε bytes.

Για την εκπαίδευση του μοντέλου labeled-char-rnn χρειάζεται να προετοιμάσουμε με ανάλογο τρόπο την πληροφορία για το είδος των χαρακτήρων. Για το σκοπό αυτό χρησιμοποιούμε ένα άλλο εργαλείο ανάλυσης σύνταξης και γραμματικής προγραμματιστικών γλωσσών που φέρει το όνομα pygments. Η επιλογή για τον διαχωρισμό των ειδών βασίζεται στα αυθαίρετα συντακτικά δέντρα abstract syntax trees της θαασςριπτ, είναι όμως απλουστευμένη και δε χρησιμοποιεί δομές δέντρων, αλλά απλών διανυσμάτων. Ο διαχωρισμός των χαρακτήρων γίνεται ανάμεσα στις ακόλουθες κλάσεις: (**K**eyword, **N**umber, **R**egex, **S**tring, **O**perator, **P**unctuator, **I**dentifier).

Οι χαρακτήρες και τα είδη τους αποθηκεύονται ως λίστες απο αλφαριθμητικά στοιχεία ώστε να είναι διαθέσιμα ανά πάσα στιγμή στην εκπαιδευτική διαδικασία. Προφανώς υπάρχει χρονική αντιστοιχία μεταξύ των αρχείων που περιέχουν της ακολουθίες χαρακτήρων με τα αρχεία που περιέχουν το είδος κάθε χαρακτήρα, όπως στα παραδείγματα των πινάκων **;;**, **;;**. Συνηθίζεται σε τέτοιου είδους προβλήματα να "ανακατεύονται" οι ακολουθίες αλφαριθμητικών χαρακτήρων με σκοπό την γρηγορότερη/καλύτερη εκπαίδευση των μοντέλων. Επειδή το ζητούμενο μας στη διπλωματική αυτή είναι η παραγωγή κώδικα και η σειρά των ακολουθιών είναι άρρικτα συνδεδεμένη με τη λειτουργικότητα και την ουσία των προγραμμάτων δεν προχωράμε σε αυτή την επιλογή.

4.3 Training

Η εκπαίδευση γίνεται στο training set του καθενός απο τα δύο σετ δεδομένων. Οι χαρακτήρες δίνονται ως one-hot vectors με διαστάσεις όσες και οι διαφορετικοί χαρακτήρες του σετ δεδομένων. Χρησιμοποιείται η τεχνική του dropout και αλγόριθμος που χρησιμοποιείται για την ελαχιστοποίηση του λάθους είναι ο TBPTT. Η συνάρτηση λάθους είναι η cross-entropy loss function: $\sum_x p(x) \log q(x)$, όπου $p(x)$ είναι η πραγματική κατανομή των χαρακτήρων και $q(x)$ η προβλεπόμενη κατανομή χαρακτήρων του μοντέλου. Η συνάρτηση αυτή χρησιμοποιείται στην πλειοψηφία της σύγχρονης βιβλιογραφίας και εμπειρικά έχει καλά αποτελέσματα στην εκπαίδευση των αναδραστικών νευρωνικών δικτύων. Εξίσου ευρεία χρήση συναντά και η συνάρτηση rmsprop που χρησιμοποιούμε για τη βελτιστοποίηση του gradient descent.

Η εκπαίδευση του αναδραστικού νευρωνικού δικτύου γίνεται, πιο περιγραφικά ως εξής: δείχνουμε στο νευρωνικό δίκτυο ακολουθίες σταθερού μηκους, το οποίο προαποφασίζεται της εκπαίδευσης. Ως αληθείς απαντήσεις δίνουμε ένα διάνυσμα ίσου μήκους με το προηγούμενο που περιέχει τους χαρακτήρες της επόμενης χρονικής στιγμής (κύλιση του διανύσματος κα-

τά μία θέση). Στην περίπτωσω του μοντέλου labeled-char-rnn με όμοιο τρόπο δίνται και οι πληροφορίες σχετικά με το είδος των χαρακτήρων, μάζι με τους αντίστοιχους χαρακτήρες. Με σκοπό την παραλληλοποίηση του προγράμματος, δίνουμε πολλά τέτοια παραδείγματα ταυτόχρονα.

Συνολικά εκπαιδεύουμε 4 διαφορετικά μοντέλα. Για κάθε δαταση το αντίστοιχο char-rnn και labeled-char-rnn μοντέλο. Για την εκπαίδευση των μοντέλων, πρέπει να αποφασιστεί ένα σύνολο παραμέτρων, που φέρουν σημαντική αξία για τις τελικές επιδόσεις του μοντέλου και την διάρκεια της εκπαίδευσης. Αυτές είναι:

- Μήκος ακολουθίας (Sequence length): Ο αριθμός χαρακτήρων που περιέχει μία ακολουθία.
- Μέγεθος παρτίδας (Batch size): Ο αριθμός των εκπαιδευτικών ακολουθιών που δίνονται παράλληλα στο μοντέλο.
- Μέγεθους κρυμμένων επιπέδων (Hidden state size): Ο αριθμός των στοιχείων LSTM που απαρτίζουν κάθε κρυφό επίπεδο.
- Πιθανότητα dropout: Η πιθανότητα να κρατηθεί ένα στοιχείο στη διάρκεια τη εκπαίδευσης.
- Αριθμός εποχών (Epoch number): Ο αριθμός "περασμάτων" του τεστ δεδομένων.
- Ρυθμός εκμάθησης (Learning Rate): Πόσο γρήγορα μαθαίνει το σύστημα από τα λάθη του.

Για την στρατηγική επιλογής και την ακριβή τιμή των υπερ-παραμέτρων θα μιλήσουμε στο κεφάλαιο 5.

4.4 Inferring

Το μοντέλο που επιλέγουμε για καθένα απο τα πειράματα αποφασίζεται σύμφωνα με τις επιδόσεις του στην μετρική λάθους της εκπαίδευσης. Για να είναι ευκολότερα ερμηνεύσιμα τα αποτελέσματα της εκπαίδευσης, θα χρησιμοποιούμε και την μετρική της "ἐυστοχίας". Η ευστοχία είναι το ποσοστό επιτυχημένων προβλέψεων επόμενου χαρακτήρα σε μία παρτίδα.

Η διαδικασία παραγωγης κώδικα που περιγράψαμε γενικεύεται και για τα μοντέλα που περιέχουν πληροφορία για το είδος των χαρακτήρων, δηλαδή δειγματοληπτούμε από την προβλεπόμενη κατανομή και χρησιμοποιούμε το αποτέλεσμα ως επόμενη είσοδο. Μπορούμε να οδηγήσουμε, εν μέρει το σύστημα, αρχικοποιώντας το με κώδικα της επιλογής μας. Αυτό αλλάζει την εσωτερική κατάσταση του μοντέλου και το "προϊδεάζει" για το τι κώδικας μπορεί να ακολουθεί. Επιπρόσθετα, κατά τη διάρκεια της δειγματοληψίας έχουμε τη δυνατότητα να επηρεάσουμε την κατανομή που προτείνει το μοντέλο. Αυτό ελέγχει το μοντέλο ως προς την "σιγουριά" του για τις προβλέψεις του και έχει τη δυνατότητα να κάνει τον παραγώμενο κώδικα είτε πιο ντετερμινιστικό είτε πιο ποικίλο. Σημαντική ιδιότητα αυτής της προσθήκης είναι οτι

δίνει στο μοντέλο τη δυνατότητα να ξεφύγει απο φαύλους κύκλους ντετερμινιστικών λαθών χάρη στην επιπλέον τυχαιότητας που εισάγεται. Η συνάρτηση Softmax Temperature:

$$P = \frac{e^{y/T}}{\sum_{k=1}^{n} e^{y_k/T}} \tag{4.1}$$

Όπου P ειναι η νέα κατανομή, y είναι η εξαγόμενη του νευρωνικού δικτύου πιθανοτικη κατανομή και n ο αριθμός των διαφορετικών στοιχείων προς πρόβλεψη. T είναι η τιμή της θερμοκρασίας που επηρεάζει την κατανομή. Για τιμές μεγαλύτερες του 1 ο κώδικας γίνεται πιο ποικίλος αλλα στο κόστος περισσότερων λαθών. Τιμές μικρότερες του 1 έχουν ως αποτέλεσμα το σύστημα να είναι πιο σίγουρο για τις προβλέψεις του.

Κεφάλαιο 5

Πειράματα και Αποτελέσματα

Στο κεφάλαιο αυτό θα αναλύσουμε τα πειράματα που έγιναν για την εκπαίδευση του μοντέλου παραγωγής κώδικα και θα εξετάσουμε την ποιότητα του παραγώμενου κώδικα. Θα εξετάσουμε ξεχωριστά κάθε σετ δεδομένων και θα συγκρίνουμε τις επιλογές και τις επιδόσεις των 2 προσεγγίσεων σε καθ΄ ένα απο αυτά. Τέλος θα ελέγξουμε τις επιδόσεις του μοντέλου σε ένα υπο-προϊόν της λειτουργίας του, στην αυτόματη συμπλήρωση κώδικα.

5.1 Πειράματα εκπαίδευσης

Ένα πολύ σημαντικό κομμάτι της εκπαίδευσης ενός τέτοιου συστήματος είναι η κατάλληλη επιλογή των υπερπαραμέτρων. Αποδεικνύεται πως η αποδοτικότερη μέθοδος για την επιλογή τους είναι η τυχαία μέθοδος [;]. Η υπολογιστική πολυπλοκότητα που εισάγουν τα αναδραστικά νευρωνικά δίκτυα και οι περιορισμένοι υπολογιστικοί πόροι που έχουμε στη διάθεση μας κάνουν αυτή την επιλογή αδύνατη. Αντ΄ αυτού επιλέγουμε εμπειρικά τις υπερπαραμέτρους (με δοκιμές) και με οδηγό της επιλογές στη σύγχρονη σχετική βιβλιογραφία.

5.1.1 Top 100 Github Javascript Projects Πειράματα

Το σετ δεδομένων αυτό αποτελείται από τα 100 πιο δημοφιλή projects σε γλώσσα javascript στον ιστότοπο αποθετηρίων λογισμικού github. Μετά το πρεποςεσσινγ παίρνουμε ακολουθίες συνολικού μήκους περίπου 79 εκατομμυρίων χαρακτήρων. Υπάρχουν 212 διαφορετικοί χαρακτήρες, συμπεριλαμβανομένων των ειδικών χαρακτήρων αρχής και τέλους αρχείων. Χρησιμοποιούμε το 95% των δεδομένων για την εκπαίδευση του συστήματος και το υπόλοιπο 5% για την επικύρωση της μάθησης. Η έλλειψη ξεχωριστού τεστ σετ μπορεί να σημαίνει οτι τα αποτελέσματα μας κάνουν οερφιτ στα δεδομένα επικύρωσης, αλλά αυτό είναι δευτερευούσης σημασίας αφού στόχος μας είναι να παράξουμε κώδικα και δεν υπάρχει αντικειμενική μαθησιακή μετρική για τον σκοπό αυτό.

Η στρατηγική επιλογής των παραμέτρων έχει ως εξής: Για να είναι οι δύο προσεγγίσεις συγκρίσιμες κρατάμε ίδιο το μέγεθος των κρυφών επιπέδων. Από αυτή την επιλογή εξαρτάται κυρίως ο αριθμός συνολικών παραμέτρων προς εκπαίδευση. Για το πρώτο σετ δεδομένων

	char-rnn	labeled-char-rnn
# Παραμέτρων	23M	23M
# Χαρακτήρων	212	212, 8
# Εποχών	40	60
Μέγεθος LSTM	1024	1024
Μήκος Ακολουθίας	100	100
Ρυθμός Εκμάθησης	0.002	0.002
% Dropout	20	20
Μέγεθος Παρτίδας	200	200

Πίνακας 5.1: Υπερπαράμετοι για τα top 100 Github js projects

αποφασίζουμε τον αριθμό αυτό σε 1024, αριθμός αρκετά μεγάλος ωστε να ειναι αντιμετωπίσιμο από το σύστημα το ογκώδες σετ δεδομένων.

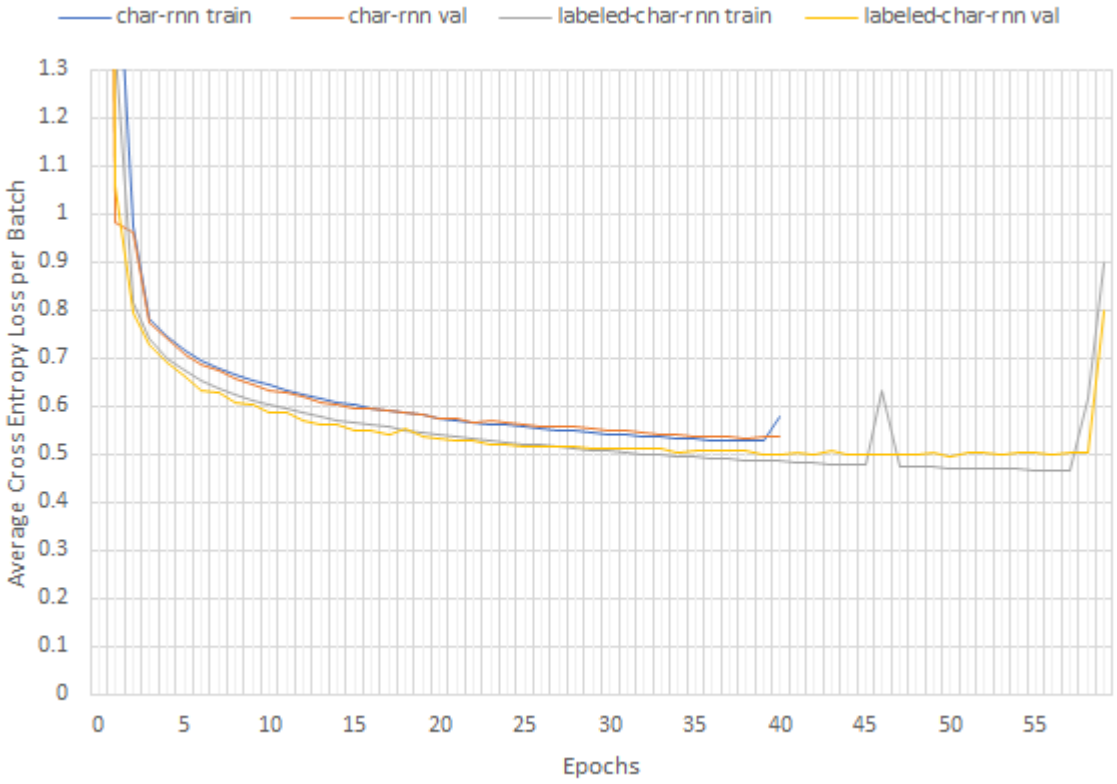
Η επόμενη υπερ-παράμετρος που πρέπει να αποφασιστεί είναι το μήκος της εκπαιδευτικής ακολουθίας, η μεταβλητή k_2 του αλγορίθμου TBPTT. Η υπερ-παράμετρος αυτή έχει μεγάλη σχέση τόσο με την ποιότητα του παραγώμενου κώδικα, αφού ελέγχει πόσους από τους προηγούμενους χαρακτήρες "βλέπει" το σύστημα, αλλά και με τον χρόνο εκτέλεσης μιας εποχής, αφού μεγαλύτερες ακολουθίες εισάγουν υπολογιστική πολυπλοκότητα. Το μέγεθος των εκπαιδευτικων ακολυθιών αποφασίζεται στους 100 χαρακτήρες και για τα δύο μοντέλα.

Ο ρυθμός εκμάθησης είναι άμεσα συνδεδεμένος με το μέγεθος παρτίδας. Όσο περισσότερα παραδείγματα βλέπει ταυτόχρονα το σύστημα τόσο πιο σίγουρο θα πρέπει να είναι για τα συμπεράσματα του. Εξαγωγή δυνατών συμπερασμάτων απο λιγοστά παραδείγματα πρέπει να αποφεύγεται. Επιπρόσθετα υπάρχει και ένας φυσικός περιορισμός στο πόσα παραδείγματα μπορούν να δείχνονται ταυτόχρονα, η μνήμη της επεξεργαστικής μας μονάδας. Τελικώς δείχνουμε 200 ακολουθίες σε κάθε βήμα εκμάθσης και θέτουμε τον ρυθμό εκμάθησης στην τιμή 0.002, ωστέ να γεμίζουμε όσο καλύτερα γίνεται την μνήμη του υπολογιστικού συστήματος αλλά να συνεχίσουμε να μαθαίνουμε αποτελεσματικά. Σημειώνεται πως η προτεινόμενη τιμή για τον ρυθμό εκμάθησης της rmsprop είναι το 0.001.

Τέλος, επειδή το σετ δεδομένων αυτό είναι αρκετά ογκώδες και περίπλοκο, είναι δύσκολο το μοντέλο μας να κάνει οερφιτ. Έτσι, δε χρείαζεται η πιθανότητα δροπουτ να είναι εξαιρετικά μεγάλη. Επιλέγουμε την υπερπαράμετρο αυτή στο 20%, ενώ η γενική προτεινόμενη τιμή είναι 40% με 50%. Ο αριθμός των εποχών αποφασίζεται έτσι ώστε κανένα από τα 2 μοντέλα να μην βελτιώνει τις επιδόσεις του στο σετ δεδομένων επιβεβαίωσης. Ο αριθμός αυτός προκύπτει στις 60 εποχές. Στον πίνακα 5.1 παρουσιάζονται συνοπτικά οι παραπάνω αποφάσεις.

Η εκπαίδευση έγινε σε μία κάρτα γραφικών Nvidia Gtx 960 με 4 gb RAM. Η εκπαίδευση διαρκεί 6 περίπου ημέρες για το πρώτο μοντέλο και 7 περίπου για το δεύτερο. Όπως αναφέραμε, η παρακολούθηση των επιδόσεων και η επιλογή των σετ βαρών για την παραγωγή κώδικα γίνεται σύμφωνα με την μετρική Average cross entropy per minibatch. Σημειώνεται πως η σύγκριση των μοντέλων στην μετρική αυτή γίνεται μόνο στο κομμάτι που αφορά την πρόβλεψη χαρακτήρων. Στην εικόνα 5.2 φαίνεται η εξέλιξη της εκπαίδευσης των 2 μοντέλων στην περίοδο

40 και 60 εποχών στο σετ εκπαίδευσης και το σετ επαλήθευσης. Ως μοντέλα παραγωγής, επιλέγουμε αυτά με τα βάρη τις 38ης εποχής για το μοντέλο char-rnn και της 53 εποχής για το μοντέλο labeled-char-rnn, αφού παρουσιάζουν την ελάχιστη τιμή της μετρικής μας. Οι επιδόσεις των μοντέλων αυτών αντιστοιχούν σε 85.6% και 87.2% ποσοστιαία επιτυχία στην πρόβλεψη του επόμενου χαρακτήρα. Η επιτυχία πρόβλεψης του είδους του χαρακτήρα στο σετ επαλήθευσης βρίσκεται πάνω από το 97%. Τα αποτελέσματα της εκπαιδευτικής διαδικασίας είναι σε πρώτη όψη ικανοποιητικά. Οι καμπύλες εκπαίδευσης και επαλήθευσης μένουν σε κοντινά επίπεδα και για τα δύο μοντέλα, γεγονός που μαρτυρά καλή γενίκευση τών χαρακτηριστικών που μαθαίνονται. Η εκμάθηση, ιδιαίτερα, της ανάθεσης είδους στον επόμενο χαρακτήρα κυμαίνεται σε πολύ υψηλά επίπεδα.



Σχήμα 5.1: Καμπύλες εκμάθησης για τα top 100 github js projects

5.1.2 Top 200 npm Projects Πειράματα

Το δεύτερο σετ δεδομένων αποτελείται από τις 200 πιο δημοφιλείς βιβλιοθήκες javascript του ιστοχώρου ωωω.νπμθς.ςομ. Οι ακολουθίες μετά την προ-επεξεργασία αριθμούν περίπου 49 εκατομμύρια χαρακτήρες με 210 διαφορετικούς χαρακτήρες. Σε αυτό το πείραμα χωρίζουμε το 90% των ακολουθιών στο σετ εκπαίδευσης και το 10% στο σετ επαλήθευσης, επειδή έχουμε λιγότερα δεδομένα και θέλουμε να αποφύγουμε μεγάλη διακύμανση στο σετ επαλήθευσης.

Οι αποφάσεις των υπερπαραμέτρων βασίζονται στις παρατηρήσεις μας από τα προηγούμενα πειράματα. Έτσι, κρατάμε ίδιο το μέγεθος παρτίδας, τον ρυθμό εκμάθησης και το μήκος

	char-rnn	labeled-char-rnn
# Παραμέτρων	10M	10M
# Χαρακτήρων	210	210, 8
# Εποχών	60	80
Μέγεθος LSTM	700	700
Μήκος Ακολουθίας	100	100
Ρυθμός Εκμάθησης	0.002	0.002
% Dropout	30	40
Μέγεθος Παρτίδας	200	200

Πίνακας 5.2: Υπερπαράμετοι για τα top 200 npm js libaries

ακολουθίας. Το σετ εκπαίδευσης έχει μικρότερο μέγεθος από το προηγούμενο πείραμα και από τις πρώτες δοκιμές παρατηρούμε σημαντικό overfitting. Προς την κατεύθυνση καλύτερης γενίκευσης των συμπερασμάτων του συστήματος, αρχικά μικραίνουμε το δίκτυο θέτοντας το μέγεθος LSTM σε 700, κίνηση η οποία μειώνει σημαντικά τις εκπαιδεύσιμες παραμέτρους του συστήματος. Έπειτα αυξάνουμε την πιθανότητα dropout σε 30% και 40% που αποτέλεσμα έχει την αργήτερη εκπαίδευση του αναδραστικού νευρωνικού δικτύου. Για να αποζημιώσουμε την τελευταία μας επιλογή αυξάνουμε τις εκπαιδευτικές εποχές του μοντέλου σε 60 και 80 αντίστοιχα. Οι εκπαιδευτικές επιλογές συνοψίζονται στον πίνακα 5.2.

Η διαδικασία που ακολουθείται είναι η ίδια με του προηγούμενου πειράματος, δηλαδή εκπαιδεύουμε το σύστημα σε μία κάρτα γραφικών Nvidia Gtx 960 με 4 gb RAM και επιλέγουμε το μοντέλο με τις καλύτερες επιδόσεις στη μετρική πρόβλεψης χαρακτήρων. Η εκπαίδευση του char-rnn διαρκεί 3 ημήρες ενώ του labeled-char-rnn διαρκεί περίπου 4. Η εξέλιξη της εκπαίδευσης φαίνεται στην εικόνα **;;**.

α) 72, 78β) 78, 72,3 94,7

Το επιλεγόμενο μοντέλο για το char-rnn είναι αυτό της 72ης εποχής με ποσοστό επιτυχίας πρόβλεψης 78%. Για το μοντέλο labeled-char-rnn το επιλεγόμενο μοντέλο είναι αυτό της 78ης εποχής με ποσοστό επιτυχίας 72.3% την πρόβλεψη χαρακτήρων και 94.7. Είναι εμφανές από το διάγραμμα οτι τα μοντέλα μας δυσκολεύονται περισσότερο να γενικεύσουν τα συμπεράσματα που εξάγουν από αυτό το σετ δεδομένων. Η ικανότητα πρόβλεψης του είδους του επόμενου χαρακτήρα παραμένει σε σχετικά υψηλά επίπεδα αλλα η προσθήκη της δεν βελτιώνει τις επιδόσεις στο σετ επιβεβαιωσης. Θα εξετάσουμε αναλυτικότερα τα αποτελέσματα αυτά στο υποκεφάλαιο των αποτελεσμάτων και στο κεφάλαιο των συμπερασμάτων.

Σχήμα 5.2: Καμπύλες εκμάθησης για τα top 100 github js projects

Κεφάλαιο 6

Συμπεράσματα και Μελλοντική Εργασία

Παράρτημα Α΄

Μεταφράσεις Ξένων όρων

Μετάφραση	Αγγλικός όρος
αδερφός	sibling
αμεταβλητότητα	idempotency
ανάκτηση πληροφορίας	information retrieval
αντιμεταθετικότητα	commutativity
απόγονος	descedant
απορρόφηση	absorption
βάση δεδομένων	database
γνώρισμα	attribute
διαπροσωπεία	interface
διαφορά	difference
δικτυακός κατάλογος	portal catalog
δικτυωτή δομή	lattice
δομικές επερωτήσεις	structural queries
δομικές σχέσεις	structural relationships
δομικό σχήμα	schema
εγκυρότητα	validity
ένωση	union