Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης Και Υπολογιστων

Προανάκτηση Δεδομένων στην Κρυφή Μνήμη με Χρήση Μοντέλων Μηχανικής Μάθησης

Διπλωματικη Εργασια

ΚΑΡΑΛΗ Γ. ΑΙΚΑΤΕΡΙΝΗ

Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2021

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής Και Υπολογιστών

Προανάκτηση Δεδομένων στην Κρυφή Μνήμη με Χρήση Μοντέλων Μηχανικής Μάθησης

Διπλωματικη Εργασια

ΚΑΡΑΛΗ Γ. ΑΙΚΑΤΕΡΙΝΗ

Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 1η Ιουλίου 2021.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Νεκτάριος Κοζύρης	Γεώργιος Γκούμας	Διονύσιος Πνευματικάτος
Καθηγητής Ε.Μ.Π	Αναπληρωτής Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2021

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής Και Υπολογιστών

(Υπογραφή)

.....
Αικατερίνη Γ. Καραλή
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Πολλοί είναι οι μηχανισμοί που χρησιμοποιούνται για τη βελτίωση της απόδοσης των εφαρμογών. Η προανάκτηση (prefetching) αφορά τη μετακίνηση δεδομένων σε ταχύτερες, ως προς το χρόνο πρόσβασης, μνήμες και έχει προταθεί ως λύση για την μείωση του άεργου χρόνου του επεξεργαστή, εν αναμονή δεδομένων από την κύρια μνήμη. Οι πλέον γνωστοί prefetchers χρησιμοποιούν πρόσφατες προσβάσεις στη μνήμη για να προβλέψουν τις μελλοντικές, ακολουθώντας απλούς ευριστικούς κανόνες ή πολύπλοκους αλγορίθμους. Στις προσεγγίσεις αυτές, δεδομένων των πόρων που έχουν δοθεί, καθορίζεται ένα όριο που αφορά την πολυπλοκότητα των μοντέλων πρόσβασης στη μνήμη, ενώ πιο εξειδικευμένοι prefetchers δύναται να συλλάβουν πιο πολύπλοκα μοτίβα. Τα νευρωνικά δίκτυα, και γενικότερα η μηχανική μάθηση, φαίνεται να μπορούν να συνδράμουν σε αυτό το σημείο και να αποτελέσουν μία καθολική μέθοδο που να ανταποκρίνεται σε όλο το εύρος των εφαρμογών. Μέσω της εκπαίδευσης και της προσαρμογής των βαρών τους, τα νευρωνικά δίκτυα μπορούν να προσεγγίσουν διαφορετικά μοντέλα πρόσβασης, εστιάζοντας στην τρέχουσα εφαρμογή. Σε αυτή την διπλωματική εργασία υλοποιούμε και αξιολογούμε δύο μοντέλα επαναλαμβανόμενων νευρωνικών δικτύων (recurrent neural networks), το πρώτο βασισμένο στα κύτταρα μακράς βραχέας μνήμης (long short-term memory) και το δεύτερο στο μηχανισμό προσοχής Transformer. Αντιμετωπίζουμε το πρόβλημα της προανάκτησης ως πρόβλημα ταξινόμησης και διαχωρίζουμε τη ροή δεδομένων για κάθε δείκτη προγράμματος. Εφαρμόζουμε τον μηχανισμό στην κρυφή μνήμη τρίτου (τελευταίου) επιπέδου. Τα πειραματικά αποτελέσματα μας ενισχύουν την ιδέα πως τα νευρωνικά δίκτυα αποτελούν πρόσφορο έδαφος για την υλοποίηση ενός καθολικού μηχανισμού προανάκτησης, ενώ μεταξύ των δύο υλοποιούμενων τύπων δεν διαφαίνεται υπεροχή του ενός έναντι του άλλου. Τέλος, ιδιαίτερη σημασία έχει η αξιολόγηση των παραμέτρων καθώς και των σχεδιαστικών επιλογών που απαιτήθηκαν για την υλοποίηση.

Λέξεις Κλειδιά

Προανάκτηση, Κρυφή Μνήμη Τελευταίου Επιπέδου, Δέλτα, Νευρωνικά Δίκτυα, Κύτταρα Μακράς Βραχέας Μνήμης, Μηχανισμός Προσοχής, Transformer

Abstract

Many are the mechanisms used to boost execution performance. Prefetching, i.e. fetching data or instructions to a faster in terms of access time local memory before it is actually needed, have been proposed as a solution to reduce processor's idle time, waiting for data from main memory. Best known prefetching techniques are based on recent memory accesses to predict future ones, using simple heuristic rules and algorithms. In these approaches the given resources set a limit on the complexity of memory access patterns that can be captured, while more sophisticated prefetchers are highly capable to respond. Neural networks, in which models are trained independently for each application, may be a promising solution to address this problem. In this thesis the models implemented and evaluated belong to recurrent neural networks. The first one is based on Long Short-term Memory cells while the second uses only attention mechanisms and more specifically a technique called Transformer. The 'prefetching problem', while at first seems to be a regression problem, is treated as a classification one. The prefetching mechanism is PC-based and applied in the Last Level Cache. The experimental results reinforce the idea that neural networks are breeding ground for the implementation of a universal prefetching mechanism. Regarding the two models studied in this thesis, there is no apparent superiority of one over the other. The evaluation of the parameters as well as the design options of the implementation is of particular importance.

Keywords

Prefetching, Last Level Cache, Delta, PC-based prefetching, Neural Networks, Long Short-Term Memory Cells, Attention Mechanism, Transformer

στους γονείς μου

Ευχαριστίες

Θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ. Νεκτάριο Κοζύρη για την επίβλεψη αυτής της διπλωματικής εργασίας και για την ευκαιρία που μου έδωσε να την εκπονήσω στο εργαστήριο Υπολογιστικών Συστημάτων(CSLab). Επίσης ευχαριστώ ιδιαίτερα τον Δρ. Βασίλειο Καρακώστα για την καθοδήγησή του και την εξαιρετική συνεργασία που είχαμε. Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου για την καθοδήγηση και την ηθική συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Αικατερίνη Καραλή

Περιεχόμενα

Περίληψη **1**

Abstract **3**

Ευχαριστίες **7**

1 Εισαγωγή **17**

1.1 Αντικείμενο της διπλωματικής 17

1.2 Κίνητρο και συναφείς προσεγγίσεις 17

1.2.1 Μηχανική μάθηση για τη βελτίωση της αρχιτεκτονικής 18

1.2.2 Μηχανική μάθηση για προανάκτηση δεδομένων 18

1.3 Προσέγγιση και Συνεισφορές 19

1.4 Οργάνωση της Διπλωματικής 20

2 Προανάκτηση **21**

2.1 Εισαγωγή 21

2.2 Μετρικές και ορολογία 22

2.3 Χαρακτηριστικά Προανάκτησης 23

2.3.1 Δεδομένα Προανάκτησης 23

2.3.2 Χρονική τοποθέτηση πρόανάκτησης 24

2.3.3 Προέλευση και προορισμός προανάκτησης 24

2.3.4 Εκκίνηση της προανάκτησης 25

2.3.5 Ταξινόμηση Μοντέλων Πρόσβασης στη μνήμη 25

2.4 Μηχανισμοί Προανάκτησης 27

2.4.1 Υλοποιήσεις Υλικού 27

2.4.2 Υλοποιήσεις Λογισμικού 30
2.4.3 Συνδυαστικές Υλοποιήσεις 31
2.5 Προκλήσεις και δυσκολίες στην προανάκτηση 32
2.5.1 Επιβάρυνση υλοποίησης 32
2.5.2 Συμβιβασμοί απόδοσης 32
2.5.3 Μόλυνση της κρυφής μνήμης και διαμοιρασμός πόρων 32
2.5.4 Προκλήσεις αξιοπιστίας 33
3 Μηχανική Μάθηση και Τεχνητά Νευρωνικά Δίκτυα 35
3.1 Τύποι προβλημάτων και εργασιών 35
3.1.1 Μάθηση με επίβλεψη 35
3.1.2 Μάθηση χωρίς επίβλεψη 36
3.1.3 Ενισχυμένη Μάθηση 36
3.2 Τεχνητά Νευρωνικά δίκτυα 36
3.2.1 Ιστορική αναδρομή 36
3.2.2 Εκπαίδευση Νευρωνικών Δικτύων 37
3.2.3 Επαναλαμβανόμενα Νευρωνικά δίκτυα - Recurrent Neural Networks (RNN) 44
3.2.4 Σύνοψη 49
4 Σχετική βιβλιογραφία 51
4.1 Βασικές ιδέες 51
4.2 LSTM embedding and clustering 52
4.3 Παράμετροι Νευρωνικού Δικτύου 53
5 Ανάλυση και σχεδίαση 57
5.1 Μέθοδος προανάκτησης 57
5.2 Υλοποίηση μοντέλου πρόβλεψης 58
5.2.1 Keras 58
5.2.2 Επιλογή και υλοποίηση μοντέλων 58

5.2.3 Εκπαίδευση 60

5.2.4 Αξιολόγηση 60

5.3 ChampSim simulator 61

5.4 Μετροπρογράμματα 63

5.5 Τελική αξιολόγηση μηχανισμών προανάκτησης 66

6 Αξιολόγηση **67**

6.1 Στατιστικά Μετροπρογραμμάτων 67

6.1.1 Αναπαράσταση δεικτών προγράμματος 67

6.1.2 Αντίκτυπος μεγέθους σελίδας 68

6.2 Σύγκριση μοντέλων πρόβλεψης 69

6.2.1 Μοντέλα χωρίς παράλληλη χρήση άλλης μεθόδου προανάκτησης 69

6.2.2 Μοντέλα με παράλληλη χρήση προανάκτησης επόμενης γραμμής στις κρυφές μνήμες πρώτου και δευτέρου επιπέδου 73

6.3 Συγκριτική αξιολόγηση μεθόδων προανάκτησης στο πλαίσιο του προσομοιωτή Champsim 76

7 Συμπεράσματα και Μελλοντικές Επεκτάσεις **81**

7.1 Συμπεράσματα 81

7.2 Μελλοντικές Επεκτάσεις 82

Παραρτήματα **85**

Α΄ Μοντέλο Transformer **87**

Βιβλιογραφία **96**

Κατάλογος Σχημάτων

2.1 βαθμός και απόσταση προανάκτησης 23

2.2 pre-execution 29

3.1 Σιγμοειδής συναρτηση 38

3.2 Υπερβολική εφαπτομένη 39

3.3 ReLU(α) και Leaky ReLU(β) 40

3.4 Σύγκλιση: (α)gradient descent, (β)stochastic gradient descent, (γ)mini-batch gradient descent 42

3.5 Ρυθμός μάθησης (α)Μεγάλο βήμα (β) Μικρό βήμα 42

3.6 Συναρτήσεις με μηδενικό κόστος στα επιλεγμένα σημεία, με τη μία εξ αυτών (μπλε) να είναι πιο πολύπλοκη, προσαρμοσμένη με μεγαλύτερη ακρίβεια στα δεδομένα εισόδου και συνεπως με μικρότερη δυνατότητα γενίκευσης (overfitted) 43

3.7 Ανάλυση αναδρομικού νευρωνικού δικτύου 44

3.8 Κύτταρο (LSTM) 45

3.9 Κύτταρο (GRU) 46

3.10 Αμφίδρομο αναδρομικό νευρωνικό δίκτυο 46

3.11 Encoder-decoder sequence to sequence model 47

3.12 γραφική απεικόνιση μοντέλου για την δημιουργία εισόδου προορισμού y_t δεδομένης εισόδου $(x_1, x_2, .., x_T)$ 48

3.13 Δίκτυο επίλυσης προβλήματος λεζάντας (image captioning problem) 48

3.14 Visual Attention 49

3.15 Παράδειγμα λειτουργίας από το [1]. Οι προτάσεις ένα και τρία συμβάλουν περισσότερο σημασιολογικά, ενώ στο κομμάτι των λέξεων υπογραμμίζονται ως σημαντικότερες οι delicious και amazing 49

3.16 Transformer - Αρχιτεκτονική μοντέλου 50

3.17 Σύγκριση μηχανισμού προσοχής απλού γινομένου με μηχανισμό προσοχής πολλαπλών κεφαλών 50

4.1 Embedding LSTM model - Learning Memory Access Patterns 53

4.2 Clustering, data processing and LSTM models - Learning Memory Access Patterns 54

4.3 Διαγράμματα μεταβολής της ακρίβειας του νευρωνικού δικτύου - Understanding Memory Access Patterns for Prefetching 55

5.1 Layers και αριθμός παραμέτρων για μοντέλοLSTM πλάτους 8 (8 cells) 59

5.2 Layers και αριθμός παραμέτρων για μοντέλοtransformer 59

6.1 Ποσοστό επιτυχημένων προβλέψεων για καθένα από τα σχήματα προανάκτησης σε δεδομένα που αφορούν την εκτέλεση των μετροπρογραμμάτων χωρίς την παράλληλη χρήση άλλου σχήματος προανάκτησης 71

6.2 Γράφημα μεταβολής ακρίβειας σε σχέση με τον αριθμό των LSTM κυττάρων . 72

6.3 Πλήρες διάγραμμα απόδοσης όλων των συγκρινόμενων μεθόδων - offline analysis 72

6.4 Ποσοστό επιτυχημένων προβλέψεων για καθένα απο τα σχήματα προανάκτησης σε δεδομένα που αφορούν την εκτέλεση των μετροπρογραμμάτων με παράλληλη χρήση προανάκτησης επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου 74

6.5 Γράφιμα μεταβολής ακρίβειας σε σχέση με τον αριθμό των LSTM κυττάρων . 75

6.6 Πλήρες διάγραμμα απόδοσης όλων των συγκρινόμενων μεθόδων - offline analysis 75

6.7 Απόδοση μεθόδων προανάκτησης σε εντολές ανά κύκλο 78

6.8 Διάγραμμα επιτάχυνσης μηχανισμών με βάση την εκτέλεση χωρίς τη χρήση προανάκτησης 79

7.1 Γράφημα διαμόρφωσης εισόδου, όπως προτείνεται στην εργασία Learning Memory Access Patterns 83

Κατάλογος Πινάκων

2.1 Συνήθη μοντέλα πρόσβασης 26

2.2 Αντιστοίχιση απλών και σύνθετων λειτουργιών μηχανής σε συνήθη μοντέλα πρόσβασης 27

5.1 Χαρακτηριστικά μεγέθη των caches στον προσομοιωτή ChampSim 63

6.1 Αντιστοιχία πλήθους μοναδικών δεικτών προγράμματος και της αντίστοιχής αναπαράστασης τους 68

6.2 Πίνακας αναπαράστασης των 2 Δέλτα με τις περισσότερες εμφανίσεις στο ιστορικό για κάθε μετροπρόγρμμα και ποσοστό των μηδενικών που αφορά προσβάσεις εκτός σελίδας. 69

6.3 Ακρίβεια μοντέλων μιας πρόβλεψης 70

6.4 Ακρίβεια μοντέλων διπλής πρόβλεψης 70

6.5 Ακρίβεια μοντέλων πρόβλεψης 73

6.6 Ακρίβεια μοντέλων διπλής πρόβλεψης 73

6.7 Επιλεγόμενα μοντέλα για (α) την αξιολόγηση αποκλειστικά του νευρωνικού μοντέλου και (β) την αξιολόγηση του μοντέλου σε συνδυασμό με προανάκτηση επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου 76

6.8 Απόδοση μεθόδων προανάκτησης σε εντολές ανά κύκλο 77

6.9 Απόδοση μεθόδων προανάκτησης, επιτάχυνση ως προς την εκτέλεση χωρίς προανάκτηση 77

Κεφάλαιο 1

Εισαγωγή

Την τελευταία δεκαετία η Μηχανική Μάθηση έχει εξελιχθεί σε παράγοντα καινοτομίας σε κάθε κλάδο, με το εύρος των εφαρμογών να περιλαμβάνει από εμπορικά προϊόντα έως ιατρικές εφαρμογές. Η ευρύτητα αυτή, σε συνδυασμό με το γεγονός πως η απόδοση των μοντέλων είναι καλή ακόμη και σε εφαρμογές όπου η αναλυτική μοντελοποίηση είναι ιδιαίτερα πολύπλοκη, ωθούν την ταχύτατη ανάπτυξή της. Ταυτόχρονα, στον κλάδο της αρχιτεκτονικής υπολογιστών, η εκθετική πρόοδος που προβλέπεται από το νόμο του Moore φαίνεται να βρίσκεται σε επιβράδυνση, πιέζοντας για συνεχή πρόοδο. Οι δύο αυτές τάσεις κινητοποιούν για τη συνδυαστική χρήση και αξιοποίηση των μεθόδων και τεχνικών, έτσι ώστε η μηχανική μάθηση να υποστηρίζεται από την αρχιτεκτονική αλλά και να τη βελτιώνει.

1.1 Αντικείμενο της διπλωματικής

Η διπλωματική εργασία αυτή ασχολείται με τη χρήση τεχνητών νευρωνικών δικτύων για την προανάκτηση δεδομένων (data prefetching).

Είναι γνωστό πως οι σύγχρονες εφαρμογές δαπανούν μεγάλο μέρος του χρόνου εκτέλεσης τους αναμένοντας δεδομένα από την κύρια μνήμη, χρόνος που παραμένει άεργος για τον επεξεργαστή. Το prefetching, η προφόρτωση δηλαδή δεδομένων σε ταχύτερα προσπελάσιμες μνήμες, έχει αποτελέσει από αρκετά νωρίς μία προσέγγιση επίλυσης του ζητήματος αυτού.

1.2 Κίνητρο και συναφείς προσεγγίσεις

Παραδοσιακές μέθοδοι προανάκτησης ξεκινούν από πολύ απλούς ευριστικούς κανόνες, όπως οι αιτήσεις επόμενης γραμμής, ενώ περιλαμβάνουν έως και πολύπλοκους αλγορίθμους. Αξιοσημείωτες τεχνικές είναι η προανάκτηση βήματος [2], οι συσχετιστικές μέθοδοι όπως η προανάκτηση Markov [3], ο GHB [4], η προανάκτηση μέσω βοηθητικού νήματος [5] και άλλες. Από αυτές οι πιο απλές, οπως είναι λογικό, δεν ανταποκρίνονται σε όλο το εύρος των εφαρμογών, ενώ στις πιο σύνθετες είναι υψηλές οι υπολογιστικές απαιτήσεις αλλά και οι α-

παιτήσεις χωρητικότητας. Για το λόγο αυτό, πρόσφατες έρευνες εχουν στραφεί στη χρήση τεχνικών μηχανικής μάθησης για προανάκτηση.

1.2.1 Μηχανική μάθηση για τη βελτίωση της αρχιτεκτονικής

Η μηχανική μάθηση έχει χρησιμοποιηθεί για την βελτίωση της επίδοσης διαφόρων στοιχείων της αρχιτεκτονικής. Ένας αξιοσημείωτο παράδειγμα εφαρμογής μηχανικής μάθησης αποτελεί η πρόβλεψη διακλάδωσης με επίδοση που ξεπερνά ακόμη και τα πιο καινοτόμα μοντέλα. Η πρόβλεψη με χρήση μοντέλου βασισμένο στα δίκτυα perceptron είναι η πρώτη που προτάθηκε [6] ως εναλλακτική υψηλής επίδοσης στα σχήματα δύο επιπέδων με τη χρήση πινάκων ιστορικού, ενώ ακολούθησαν αρκετές ακόμη προσεγγίσεις βασιζόμενες σε αυτά [7, 8, 9] αξιοποιώντας διαφορετικά τις δυνατότητες τους. Σύγχρονοι προηγμένοι προβλεπτές έχουν σημαντική βελτίωση στον IPC με μόνο λίγες, ΄δυσκολες να προβλεφθούν΄ (hard to predict), εντολές διακλάδωσης [10, 11]. Σε αυτή την κατεύθυνση προτάθηκαν προβλεπτές συνελικτικών νευρωνικών δικτύων δύο επιπέδων [11], με τα αποτελέσματα να δίνουν κίνητρο για περαιτέρω διερεύνηση και μελέτη.

Ένας άλλος τομέας εφαρμογής μηχανικής μάθησης είναι οι ελεγκτές της μνήμης και γενικότερα των αποθηκευτικών μέσων που επηρεάζουν τόσο την απόδοση όσο και την αξιοπιστία ενός συστήματος. Αρχικά προτάθηκε ένα μοντέλο ενισχυμένης μάθησης [12] για την επίλυση θεμάτων ταυτοχρονισμού, καθυστερήσεων και άλλων, ενώ στη συνέχεια η μέθοδος επεκτάθηκε [13] ώστε να βελτιστοποιεί πέραν της απόδοσης και παραμέτρους όπως η ενέργεια. Ακολούθησαν εφαρμογές του αλγορίθμου Q-learning [14] [15][16] για την ελάττωση της καταναλισκόμενης ενέργειας, ενώ σε ότι αφορά την αξιοπιστία του συστήματος προτάθηκαν μοντέλα παλινδρόμησης [17], όπως και η μέθοδος τυχαίου δάσους (random forest)[18].

Τέλος, σημαντικό είναι να αναφερθούν αντίστοιχες προσεγγίσεις εφαρμογής τεχνικών μηχανικής μάθησης που έχουν πραγματοποιηθεί στη δικτύωση των chip και πιο συγκεκριμένα σε θέματα δυναμικής κλιμάκωσης της τάσης και της συχνότητας (DVFS), τον έλεγχο της συνδεσμολογίας και της ροής, της τοπολογίας και της γενικότερης σχεδίασης καθώς και την πρόβλεψη της απόδοσης. Ομοίως σε ότι αφορά τη συνολική απόδοση του συστήματος με τη βελτιστοποίηση της ενεργειακής απόδοσης, τον διαμοιρασμό πόρων και εργασιών και τη διάταξη του Chip, ενώ αξιοσημείωτες είναι και οι νέες δυνατότητες που δημιουργεί η μηχανική μάθηση στο κομματι των κατά προσέγγιση υπολογισμών με μεγαλύτερη εφαρμογή στα δεδομένα.

1.2.2 Μηχανική μάθηση για προανάκτηση δεδομένων

Εστιάζοντας και πάλι στο πρόβλημα της προανάκτησης που στοχεύει η παρούσα διπλωματική εργασία, η χρήση νευρωνικών δικτύων αφορμάται από την έλλειψη καθολικής μεθόδου αντιμετώπισης όλου του εύρους εφαρμογών. Μια προσέγγιση του ζητήματος αποτέλεσε η συσχέτιση του με τα προβλήματα φυσικής γλώσσας και η αντιμετώπισής του με παρόμοιο τρόπο. Έτσι,

από τις πρώτες κιόλας σχετικές εργασίες, το είδος του νευρωνικού δικτύου που χρησιμοποιήθηκε ήταν το μοντέλο μακράς βραχέας μνήμης [19], ενώ ακολουθήθηκαν διάφορες προσεγγίσεις, με την προανάκτηση να αντιμετωπίζεται είτε ως πρόβλημα παλινδρόμησης(regression) [20], είτε ως πρόβλημα ταξινόμησης(classification) [21][19]. Μία αναλυτική περιγραφή των σχετικών εργασιών που λειτούργησαν ως βάση για τη δεδομένη διπλωματική περιλαμβάνεται στο κεφάλαιο 4.

1.3 Προσέγγιση και Συνεισφορές

Σε αυτή τη διπλωματική, ακολουθούμε την συλλογιστική πορεία των άρθρων Learning Memory Access Pattern[21] και Understanding memory access patterns for prefetching [19].

Αντιμετωπίζουμε την προανάκτηση ως πρόβλημα ταξινόμησης, υλοποιούμε διαχωρισμό των ροών για κάθε δείκτη προγράμματος (program counter) και χρησιμοποιούμε Δέλτα αντί των διευθύνσεων. Με τον όρο Δέλτα αναφερόμαστε στην απόσταση της επόμενης διεύθυνσης από την τρέχουσα που προκαλούν αστοχία (μισς) σε μια κρυφή μνήμη. Ως δεδομένα λαμβάνουμε δείκτες προγράμματος και την ακολουθία από τα δέλτα που τους αντιστοιχεί, από τις αστοχίες στη κρυφή μνήμη τρίτου επιπέδου όπου και χρησιμοποιούμε το μηχανισμό. Τα δεδομένα κωδικοποιούνται ομοιόμορφα με χρήση των έξι λιγοτερων σημαντικών bit, ενώ παράλληλα περιορίζουμε τα αποτελέσματα στο όριο της σελίδας (4KB). Η πρόβλεψη υλοποιείται με τόσο με χρήση αναδρομικών νευρωνικών δικτύων μακράς βραχέας μνήμης LSTM [22], όσο και με χρήση μηχανισμού προσοχής (Transformer) [23]. Πραγματοποιούμε σύγκριση μεταξύ των μοντέλων χρησιμοποιώντας offline ανάλυση, από την οποία επιλέγουμε αυτό με τη μεγαλύτερη απόδοση. Στη συνέχεια, το μοντέλο αυτό που έχει επιλεγεί, ενσωματώνεται στον προσωμοιωτή ChampSim και έχουμε εκ νέου εκτέλεση της εφαρμογής. Από τα αποτελέσματα της προσωμοίωσης χρησιμοποιούμε ως μετρική τον μέσο αριθμό εντολών ανά κύκλο και συγκρίνουμε τα μοντέλα με παραδοσιακές, απλές αλλά και πιο σύνθετες μεθόδους προανάκτησης που έχουν προταθεί στη βιβλιογραφία. Σημειώνεται πως οι χρησιμοποιούμενες εφαρμογές αποτελούν τμήμα της σουίτας SPEC CPU®2017.

Με βάση τα αποτελέσματα των πειραμάτων, φαίνεται να υπάρχει προοπτική ώστε τα νευρωνικά δίκτυα να αποτελέσουν έναν αποτελεσματικό μηχανισμό προανάκτησης. Με βάση όμως και τις διακυμάνσεις που προκύπτουν στην απόδοση τους, σημαντική είναι η περαιτέρω διερεύνηση των μεθόδων, των τεχνικών αλλά και των παραδοχών που έχουν πραγματοποιηθεί σε κάθε περίπτωση.

Ως σημαντικότερες προτάσεις για μία εκ νέου προσέγγιση του ζητήματος θα μπορούσαμε να αναφέρουμε την καταγραφή πλήρους ιστορικού των προσβάσεων στη μνήμη, αντί αποκλειστικά των misses της L3, την αξιολόγηση περισσότερων παραμέτρων στην περίπτωση των LSTM όπως το lookback size, την αξιολόγηση του μοντέλου εισόδου και σύγκριση με εναλλακτικές προτάσεις, και την άρση του περιορισμού του μηχανισμού προανάκτησης στο πλαίσιο της σελίδας.

1.4 Οργάνωση της Διπλωματικής

Στο Κεφάλαιο 2 καταγράφεται το σύνολο των μεθόδων προανάκτησης, ενώ το Κεφάλαιο 3 αναφέρεται στη μηχανική μάθηση, και ειδικότερα τα τεχνητά νευρωνικά δίκτυα, ως προς τα είδη, τη χρήση, και τις τεχνικές. Στη συνέχεια, επεκτείνοντας το θεωρητικό υπόβαθρο με ορισμένες εφαρμογές νευρωνικών δικτύων για την εκτέλεση προανάκτησης στο κεφάλαιο 4, δίνεται έμφαση στις εργασίες που λειτούργησαν ως βάση της παρούσας διπλωματικής εργασίας. Στο κεφαλαιο 5 περιγράφονται οι βασικές ιδέες, τα εργαλεία καθώς και οι επιλογές σχεδίασης, δηλαδή το σύνολο της μελέτης σε ό,τι αφορά την υλοποίηση. Στο κεφάλαιο 6 παρουσιάζονται τα αποτελέσματα των προσομοιώσεων και, τέλος, στο 7ο Κεφάλαιο εξάγονται τα συμπεράσματα της παρούσας εργασίας καθώς και προτάσεις για περαιτέρω μελέτη.

Κεφάλαιο 2

Προανάκτηση

2.1 Εισαγωγή

Το κενό απόδοσης επεξεργαστή-μνήμης (processor-memory performance gap) ή το τείχος της μνήμης (memory wall) όπως αναφέρεται [24] αποτελεί καίριο πρόβλημα που μαστίζει τα υπολογιστικά συστήματα σε όλο τους το εύρος, από ενσωματωμένα συστήματα μέχρι υπερυπολογιστές. Ενώ η επίδοση των επεξεργαστών αυξάνεται εκθετικά, η μνήμη κλιμακώνει υπό όρους εύρους ζώνης και μεγέθους αλλά όχι και ως προς το χρόνο πρόσβασης. Ως αποτέλεσμα, ο χρόνος που αντιστοιχεί σε έναν κύκλο σε κάποιον σύγχρονο επεξεργαστή είναι τώρα δύο τάξεις μεγέθους μικρότερος από αυτόν που αντιστοιχεί στην καθυστέρηση πρόσβασης στην μνήμη (DRAM). Έτσι μία σύγχρονη εφαρμογή μπορεί να δαπανήσει έως και πάνω από το 50% του συνολικού πλήθους των κύκλων υπολογισμού σε αναμονή για δεδομένα από την κύρια μνήμη [25, 26, 27].

Ένας τρόπος με τον οποίο έχει αντιμετωπιστεί το πρόβλημα αυτό είναι η εισαγωγή βαθιάς ιεραρχίας μνήμης με μικρές, τόσο ως προς τη χωρητικότητα αλλά κυρίως ως προς το χρόνο πρόσβασης, κρυφές μνήμες. Παρόλα αυτά, δεδομένης της αύξησης του μεγέθους των δεδομένων ανά εφαρμογή, καθώς και της μείωσης της κλιμακωσιμότητας των transistors, δεν μπορούμε να βασιστούμε αποκλειστικά στην κλιμάκωση της χωρητικότητας των κρυφών μνημών.

Η προανάκτηση παρακάμπτει την περιορισμένη χωρητικότητα της κρυφής μνήμης (cache) μετακινώντας δεδομένα από επίπεδα μνήμης με υψηλή καθυστέρηση πρόσβασης σε ταχύτερες μνήμες, προτού αυτά ζητηθούν ρητά από την εκτελέσιμη εφαρμογή, μειώνοντας έτσι το ρυθμό αστοχίας (miss rate) και κατά συνέπεια βελτιστοποιώντας την επίδοση. Ο μηχανισμός αφορά εξίσου δεδομένα (Data Prefetching) και εντολές (Instruction Prefetching) του προγράμματος, με εφαρμογή ίσως διαφορετικών τεχνικών, καθώς σε κάθε περίπτωση αντιστοιχούν και διαφορετικά μοντέλα πρόσβασης στη μνήμη. Η προανάκτηση μπορεί να γίνει με χρήση υλικού, λογισμικού ή με συνδυασμό των δύο [28]. Προκειμένου να είναι αποτελεσματική, θα πρέπει να υλοποιείται με τέτοιο τρόπο ώστε να είναι έγκαιρη, ακριβής και με μικρό κόστος υλοποίησης, ενώ ακόμη σημαντικό είναι να ληφθούν υπόψη και οι μεταβολές που μπορεί να επιφέρει στο

σύστημα της μνήμης. Άσκοπη χρήση της μπορεί να είναι επιζήμια για την επίδοση και την κατανάλωση ενέργειας. Η σωστή προανάκτηση είναι σαφώς 'σημασιολογικά μη ορατή' με την έννοια ότι δεν τροποποιεί τα περιεχόμενα των καταχωρητών και της μνήμης και δεν μπορεί να προκαλέσει σφάλματα της εικονικής μνήμης. Για την επίτευξη των παραπάνω αλλά και προκειμένου να ανταποκρίνεται σε όλο το φάσμα των εφαρμογών πρέπει να 'αντιλαμβάνεται' τα διαφορετικά μοντέλα πρόσβασης στη μνήμη σε κάθε εφαρμογή. Μέχρι τώρα έχουν αναπτυχθεί ποικίλοι τρόποι προσέγγισης, ενώ συγκεκριμένη στρατηγική που να οδηγεί σε βέλτιστη απόδοση δεν έχει προταθεί.

2.2 Μετρικές και ορολογία

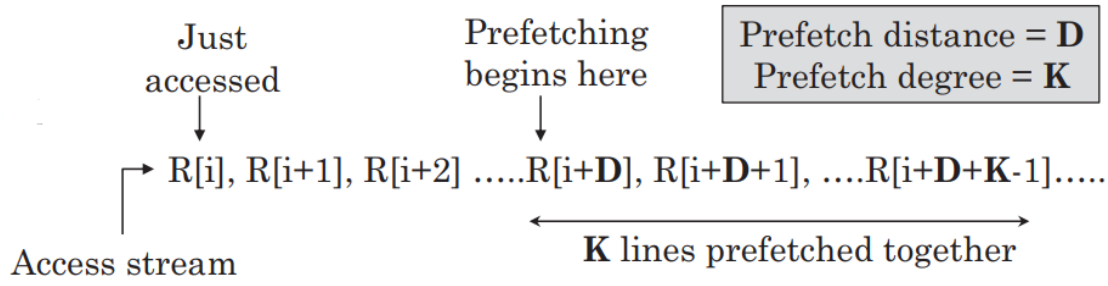
Για το χαρακτηρισμό των prefetches συνήθως χρησιμοποιούνται οι εξής όροι:

Χρήσιμες (useful prefetches)	prefetches που μειώνουν αστοχίες
Επιβλαβείς (harmful prefetches)	prefetches που αντικαθιστούν χρήσιμα δεδομένα στη μνήμη
πλεονάζουσες (redundant prefetching)	prefetches για τα οποία τα δεδομένα που ζητήθηκαν βρίσκονται ήδη στην κρυφή μνήμη
κακές (bad prefetches)	prefetches των οποίων τα δεδομένα ζητούνται αφού αντικατασταθούν στην κρυφή μνήμη
Εγκαιρες/καθυστερημένες (timely/late prefetch)	prefetches των οποίων τα δεδομένα τοποθετούνται στην κρυφή μνήμη πριν (/αφού) ζητηθεί πρόσβαση σε αυτά
εσφαλμένες	Η διεύθυνση μπορεί να προκαλεί εξαίρεση για σφάλματα εικονικών διευθύνσεων και παραβιάσεις της προστασίας

Όσον αφορά τις μεταβλητές, οι δύο κυριότερες που χρησιμοποιούνται στην προανάκτηση είναι ο βαθμός (degree) και η απόσταση (distance). Ο βαθμός αντιστοιχεί στον αριθμό των στοιχείων που φέρνουμε από τη μνήμη ανά εντολή προανάκτησης, ενώ η απόσταση εκφράζει πόσο μακριά βρίσκονται τα δεδομένα που ζητούνται από τα δεδομένα στα οποία γίνεται πρόσβαση κατά την εκκίνηση της προανάκτησης (όμοια, η απόσταση μεταξύ των εντολών για instruction prefetching) (Σχήμα 2.1).

Σημαντικές ακόμη παράμετροι για την απόδοση της προανάκτησης είναι το μέγεθος της κρυφής μνήμης και ο χρόνος μεταφοράς μιας γραμμής της (line transfer interval) από τη κύρια μνήμη και αντίστροφα. Και οι δύο καθορίζουν σε σημαντικό βαθμό το μέγεθος των παραπάνω μεταβλητών καθώς και το χρονικό διάστημα για το οποίο ένα αντικείμενο προανάκτησης μπορεί να παραμείνει πριν αντικατασταθεί στην κρυφή μνήμη.

Τέλος, για την αξιολόγηση της προανάκτησης οι κυρίως χρησιμοποιούμενες μετρικές είναι η



Σχήμα 2.1: *βαθμός και απόσταση προανάκτησης*

κάλυψη (Coverage), που δηλώνει το ποσοστό των αρχικών αστοχιών που εξαλείφθηκαν, η ακρίβεια (Accuracy),που εκφράζει το ποσοστό των ορθών εντολών προανάκτησης και ο χρονισμός (timeliness), που αντιστοιχεί στην χρονική ορθότητα της εκκίνησης της προανάκτησης.

Σημαντικό είναι να σημειωθεί ότι στους αλγορίθμους προανάκτησης οι μετρικές λειτουργούν η μία αντιστρόφως ανάλογα από τις υπόλοιπες. Υλοποίηση επιθετικής πολιτικής, με στόχο την βελτίωση της κάλυψης, μειώνει την ακρίβεια. Όμοια, αλγόριθμοι που βελτιώνουν το χρονισμό μπορεί να οδηγήσουν σε πρόωρες αιτήσεις δεδομένων με αποτέλεσμα τη μόλυνση της κρυφής μνήμης και συνεπώς την αύξηση των αστοχιών, δηλαδή τη μείωση της κάλυψης [29].

2.3 Χαρακτηριστικά Προανάκτησης

Στη μελέτη και την ταξινόμηση των διαφόρων μηχανισμών προανάκτησης ακολουθήθηκαν οι εξής άξονες που αφορούν τα σημαντικότερα ζητήματα που θα πρέπει να επιλυθούν σε κάθε σχεδίαση: Δεδομένα προανάκτησης, χρονική τοποθέτηση, προέλευση και προορισμός, εκκίνηση προανάκτησης.

2.3.1 Δεδομένα Προανάκτησης

Σε ένα πρόβλημα προανάκτησης η πρόβλεψη για το ποια δεδομένα θα φέρουμε από τη μνήμη είναι η πλέον σημαντική απόφαση. Εαν η στρατηγική προανάκτησης δύναται να προβλέψει την εμφάνιση μελλοντικών αστοχιών, τότε είναι εφικτή η προσθήκη των κατάλληλων εντολών ώστε να φέρουμε από τη μνήμη τα απαραίτητα δεδομένα προτού αυτά ζητηθούν ρητά.

Προκειμένου να καλυφθεί η καθυστέρηση που προκαλείται λόγω των αστοχιών της κρυφής μνήμης, η ακρίβεια του μηχανισμού προανάκτησης ως προς τα δεδομένα θα πρέπει να είναι υψηλή. Σε αντίθετη περίπτωση αντί του περιορισμού των αστοχιών θα έχουμε μόλυνση της κρυφής μνήμης, απομακρύνοντας χρήσιμα δεδομένα από αυτή γεγονός που συνεπάγεται συνολική επιβάρυνση του συστήματος με νέες αστοχίες και περεταίρω δέσμευση του εύρους ζώνης της μνήμης.

2.3.2 Χρονική τοποθέτηση πρόανάκτησης

Το σημείο έναρξης της εκτέλεσης μίας εντολής προανάκτησης είναι ζωτικής σημασίας για τη συνολική απόδοση. Τα δεδομένα που έχουν επιλεγεί θα πρέπει να βρίσκονται στην κρυφή μνήμη προτού ζητηθούν ρητά, ωστέ να αποφευχθεί η εκάστοτε αστοχία. Η αποτελεσματικότητα της έγκαιρης προανάκτησης βασίζεται στη συνολική επιβάρυνση λόγω του μηχανισμού, αλλά και στο διαθέσιμο χρόνο μέχρι την επόμενη αστοχία. Αν ο χρόνος για την επόμενη αστοχία δεν επαρκεί για την πρόβλεψη και μεταφορα των δεδομένων, τότε με την αυξηση της απόστασης μπορεί να αποφευχθεί μία καθυστερημένη απόπειρα προανακτησης.

Και για το ζήτημα αυτό του χρονισμού έχουν προταθεί πολλές εναλλακτικές μεθοδολογίες [30][29]. Κάποιες από αυτές είναι εξαρτώμενες από κάποιο γεγονός, όπως η πρόσβαση στη μνήμη, οι αστοχίες της κρυφής μνήμης, τα άλματα ή ακόμη η πρόσβαση σε προηγουμένως προανακτηθέντα δεδομένα. Άλλες, χρησιμοποιούν έναν προπορευόμενο δείκτη προγράμματος, έτσι ώστε να παραμένουν μερικούς κύκλους μπροστά. Όσον αφορά τις υλοποιήσεις λογισμικού, η τοποθέτηση των κατάλληλων εντολών είναι απόφαση του μεταγλωττιστή ή του προγραμματιστή, γεγονός που καθιστά τις υλοποιήσεις αυτές λιγότερο ακριβείς στο ζήτημα του χρονισμού, ενώ σε συνδυαστικές υλοποιήσεις με βοηθητικά νήματα απαιτείται ο περιοδικός συγχρονισμός των νημάτων ώστε να αποφεύγεται πολύ πρόωρη ή καθυστερημένη προανάκτηση. Τέλος, σε πολλές εφαρμογές η ανάλυση των χρονικών διαστημάτων μεταξύ των διαδοχικών προσβάσεων μπορεί να φανεί χρήσιμη για την πρόβλεψη του σωστού χρονικού πλαισίου για την εκκίνηση της προανάκτησης.

Σημειώνεται ότι υπήρξαν υλοποιήσεις που βασίστηκαν στο χρονισμό. Στο [31] παρατήρησαν ότι σε μια ροή ο χρόνος πρόσβασης στα δεδομένα γίνεται με τρόπο προβλέψιμο. Με βάση αυτό, η τεχνική που προτάθηκε αποθηκεύει εκτός των διευθύνσεων και πληροφορίες χρονισμού ώστε να γίνει πρόβλεψη για το πότε συνέβη η αστοχία. Οι πληροφορίες χρονισμού που επιλέχθηκαν στη συγκεκριμένη περίπτωση αφορούν τον αριθμό των αστοχιών.

2.3.3 Προέλευση και προορισμός προανάκτησης

Η ιεραρχία της μνήμης περιλαμβάνει πολλά επίπεδα, όπου μπορεί να χρησιμοποιηθεί η προανάκτηση. Μεγάλα περιθώρια για βελτίωση της επίδοσης προσφέρει η προανάκτηση μεταξύ κρυφών μνημών και κύριας μνήμης, καθώς επίσης και μεταξύ κύριας μνήμης και αποθηκευτικού χώρου. Για το σχεδιασμό μίας μεθόδου προανάκτησης, είναι απαραίτητο να είναι γνωστή η θέση στην οποία βρίσκεται το πιο πρόσφατα επεξεργασμένο αντίγραφο των δεδομένων. Στις υπάρχουσες βαθιές ιεραρχίες που χρησιμοποιούν πολιτική υστεροεγγραφής (write-back policy), τα δεδομένα μπορεί να βρίσκονται σε οποιοδήποτε επίπεδο της ιεραρχίας, ενώ η πολυπλοκότητα αυξάνεται αν θεωρήσουμε πολυπύρηνα συστήματα με ιδιωτικές και μοιραζόμενες κρυφές μνήμες μεταξύ των πυρήνων, οπότε εμπλέκονται και ζητήματα συνάφειας.

Ο προορισμός της προανάκτησης θα πρέπει να είναι σαφώς πιο κοντά στον επεξεργαστή από την πηγή ώστε να βελτιώνεται η απόδοση. Τα δεδομένα της προανάκτησης μπορούν να

τοποθετηθούν είτε σε κάποια κρυφή μνήμη του συστήματος, ιδιωτική ή μοιραζόμενη, είτε να υπάρχει ειδικά αφιερωμένη μνήμη σε αυτά (Prefetch cache). Ενώ η βέλτιστη επιλογή από άποψη απόδοσης είναι η αποθήκευση των δεδομένων στην κρυφή μνήμη του συστήματος, ενέχει ο κίνδυνος μόλυνσης της, σε περίπτωση που ακολουθηθεί επιθετική πολιτική προανάκτησης ή η ακολουθούμενη στρατηγική παρουσιάζει όχι ικανοποιητική ακρίβεια, είτε συνολικά, είτε ειδικά στη δεδομένη εφαρμογή. Για την αντιμετώπιση αυτού, προτάθηκε η χρήση ενός απομονωτή. Σε αυτή την περίπτωση τα δεδομένα που προέρχονται από προανάκτηση τοποθετούνται εκεί έως ότου ζητηθούν από τον επεξεργαστή, οπότε μεταφέρονται στην κρυφή μνήμη, ή αντικατασταθούν από νέα δεδομένα. Με αυτό τον τρόπο το περιεχόμενο της κρυφής μνήμης παραμένει όμοιο με αυτό που αντιστοιχεί σε εκτέλεση χωρίς προανάκτηση, υπάρχει όμως λειτουργική επιβάρυνση για την αναζήτηση των δεδομένων σε αυτή την πρόσθετη μνήμη, είτε παράλληλα, είτε σειριακά με τις κρυφές μνήμες, καθώς και για την εκ νέου μεταφορά τους, το κοστος της οποίας όμως είναι σαφώς ελάχιστο συγκριτικά με το κόστος μεταφοράς από την κύρια μνήμη.

2.3.4 Εκκίνηση της προανάκτησης

Οι εντολές προανάκτησης μπορεί να εκτελούνται είτε από τον ίδιο τον επεξεργαστή που θα χρησιμοποιήσει τα δεδομένα (client-initiated or pull-based prefetching), είτε από κάποιον διαφορετικό επεξεργαστή αφιερωμένο σε αυτό το σκοπό (push-based prefetching). Στην πρώτη κατηγορία ανήκουν κυρίως μονοπύρηνα συστήματα αλλά και επεξεργαστές που υποστηρίζουν πολλαπλά νήματα ανά πυρήνα και καθιστούν δυνατή την απόζευξη των υπολογισμών από τις προσβάσεις στη μνήμη, επιτρέποντας στον ίδιο πυρήνα την υλοποίηση ακόμη και μηχανισμών βοηθητικού νήματος και πρόωρης εκτέλεσης. Με τη δεύτερη κατηγορία αναφερόμαστε τόσο σε πολυπύρηνα συστήματα, οπότε ένας πυρήνας εκτός του υπολογιστικού φέρνει δεδομένα σε μια μοιραζόμενη με τον υπολογιστικό πυρήνα μνήμη, όσο και υλοποιήσεις όπου η προανάκτηση εκκινείται από την κύρια μνήμη ή ακόμη και κάποιον διακομιστή αν αναφερόμαστε σε κατανεμημένα συστήματα.

2.3.5 Ταξινόμηση Μοντέλων Πρόσβασης στη μνήμη

Παρά τον όγκο ερευνητικής εργασίας γύρω από τις μεθόδους προανάκτησης καθώς και γενικότερα της αντιμετώπισης της καθυστέρησης της εκτέλεσης λόγω του κενού μνήμης-επεξεργαστή, δεν υπάρχει ακόμα σε βάθος κατανόηση των μοντέλων πρόσβασης στη μνήμη που ακολουθούνται από τις εφαρμογές. Ως συνέπεια αυτού, παραμένει ακόμα δύσκολη η λήψη απόφασης σχετικά με την επιλογή της κατάλληλης μεθόδου προανάκτησης για κάποια δεδομένη εφαρμογή και η πρόβλεψη του οφέλους που θα μπορούσε να επιτευχθεί. Είναι λογικό πως μια επιτυχής σχεδίαση μηχανισμού προανάκτησης, που ανταποκρίνεται σε όλο το φάσμα των εφαρμογών ενώ παράλληλα ελαχιστοποιεί το κόστος υλοποίησης της, θα πρέπει να άντιλαμβάνεται΄ τα διαφορετικά μοντέλα πρόσβασης στη μνήμη. Στην κατεύθυνση αυτή, πρόσφατη εργασία [32] στοχεύει στο διαχωρισμό και την ταξινόμηση αυτών των μοντέλων πρόσβασης και θέτει αυτά ως πρώτο βήμα πριν το σχεδιασμό και την υλοποίηση.

Οι αλγόριθμοι πρόβλεψης παραδοσιακά αναζητούν στο ιστορικό της εφαρμογής διάφορα μοντέλα πρόσβασης στην κύρια μνήμη ή αστοχιών της κρυφής μνήμης. Το πλήθος των δεδομένων που συλλέγονται καθώς και η επιλογή του μηχανισμού προανάκτησης καθορίζουν τα μοντέλα που μπορούν να εντοπιστούν. Πιθανός διαχωρισμός των μοντέλων μπορεί να προκύψει με βάση τη χωρική απόσταση μεταξύ των προσβάσεων, την επαναληψιμότητα ή το μέγεθος των δεδομένων. Τα χωρικά μοντέλα διαχωρίζονται με βάση την απόσταση μεταξύ των προσβάσεων σε συνεχή ή μη, ενώ τα μη συνεχή μπορούν ακόμη να διαφοροποιηθούν με βάση το βήμα σε σταθερού βήματος, βήματος δύο ή περισσότερων διαστάσεων, αρνητικού βήματος ή τυχαίου. Όσον αφορά την επαναληψιμότητα, έχουμε ακολουθίες που παρατηρούνται μία φορά και άλλες με πολλαπλές εμφανίσεις, ενώ το μήκος των δεδομένων που ζητούνται μπορεί να είναι είτε σταθερό, είτε όχι.

Στο [32] προτείνεται η ταξινόμηση με βάση την παρατήρηση ότι υπάρχει μία αναδρομική σχέση που καθορίζει τη διαφορά μεταξύ διαδοχικών διευθύνσεων φόρτωσης για δεδομένο δείκτη προγράμματος (program counter, PC). Η αναδρομική αυτή σχέση ορίζεται ως:

$$A_n = f(A_n - 1)$$

Όπου A η διεύθυνση μνήμης, το n εκφράζει τη ν-ιοστή εκτέλεση μιας δεδομένης εντολής φόρτωσης και f συνάρτηση που συσχετίζει δύο διαδοχικές εκτελέσεις της εντολής. Η πολυπλοκότητα της συνάρτησης f καθορίζει τις απαιτήσεις του μηχανισμού προανάκτησης, αλλά και τις ικανότητες αυτού για πρόβλεψη.

Μοτίβο	Αναδρομική σχέση	Παράδειγμα	Σημειώσεις
Σταθερό	$A_n = A_{n-1}$	*ptr	
Δέλτα	$A_n = f(A_{n-1} + d)$	διάσχιση πίνακα	d = βήμα
pointer chase	$A_n = Ld(A_{n-1})$	next = current -> next	Load address is derived from the value of previous load
Indirect Delta	$A_n = Ld(B_{n-1} + d)$	*(M[i])	
Indirect Index	$A_n = Ld(B_{n-1+c} + d)$	M[N[i]]	c = base address of M, d = stride
Constant plus offset reference	$A_n = B_n + c_1$, $B_n = Ld(B_{n-1} + c_2)$	linked list traversal	c_1 = data offset, next pointer offset, B_n = address of the n^{th} struct

Πίνακας 2.1: *Συνήθη μοντέλα πρόσβασης*

Τα πλέον συνήθη μοντέλα πρόσβασης παρατίθενται στον πίνακα 2.1. Ενώ ο πίνακας δεν περιλαμβάνει όλα τα πιθανά μοντέλα, οι περισσότερες προσβάσεις αντιστοιχούν αν όχι σε κάποιο από αυτά, σε κάποιο συνδυασμό ή εμφώλευσή τους. Αυτού του είδους η ταξινόμηση μας δίνει τη δυνατότητα, σε δεδομένη εφαρμογή, να αποφασίσουμε τις δυνατότητες του μηχανισμού με βάση το ποσοστό αστοχιών που θέλουμε να πετύχουμε, να αποφανθούμε με δεδομένο μηχανισμό προανάκτησης για την απόδοσή του καθώς και να βγάλουμε συμπεράσματα για τα ζητήματα χρονισμού. Η προσέγγιση όμως αυτή, που λαμβάνει υπόψη το σχεδιασμό και τις δομές δεδομένων, είναι χρήσιμη για ανθρώπινη παρατήρηση, όχι όμως και για αυτοματοποιημένη επεξεργασία. Προτάθηκε, λοιπόν, η δημιουργία πυρήνων προανάκτησης, καθένας εκ των

οποίων αποτελείται από ένα πλήθος πηγών δεδομένων και λειτουργιών σε αυτές, με βάση τα οποία παράγεται η αντίστοιχη αστοχία. Ως λειτουργίες εννοούνται όλες οι εντολές που υλοποιούνται στην υπό μελέτη αρχιτεκτονική. Σε αντιστοιχία με τον προηγούμενο πίνακα στον 2.2 παρατίθενται οι πλέον χρησιμοποιούμενες ακολουθίες λειτουργιών από τις οποίες επωφελείται ο εξεταζόμενος μηχανισμός προανάκτησης. Συνδυάζοντας πολλαπλά μοντέλα στην ίδια κατηγορία ο μηχανισμός γίνεται πιο γενικός για την αντιμετώπιση τυχαίων πολύπλοκων μοντέλων ενώ ακόμη δίνει τη δυνατότητα για πολύπλευρη ανάλυση.

2.4 Μηχανισμοί Προανάκτησης

Πλήθος τεχνικών έχουν μελετηθεί για την υλοποίηση της προανάκτησης, ο διαχωρισμός των οποίων βασίζεται στο είδος της υλοποίησης.

2.4.1 Υλοποιήσεις Υλικού

Η προανάκτηση υλικού θα μπορούσε σε ένα πολύ γενικό πλαίσιο να διαχωριστεί σε προανάκτηση βήματος και σε συσχετιστική προανάκτηση. Η προανάκτηση βήματος αφορά την αναγνώριση σταθερών, επαναλαμβανόμενων διαφορών μεταξύ διαδοχικών διευθύνσεων της μνήμης (deltas). Η συσχετιστική προανάκτηση στοχεύει εξίσου στην αναγνώριση επαναλαμβανόμενων μοτίβων, αλλά όχι τόσο σταθερών. Βασίζεται στην αποθήκευση του ιστορικού των προσβάσεων της μνήμης σε μεγάλους πίνακες και είναι σαφώς πιο αποδοτική στον εντοπισμό μη κοινών μοντέλων.

2.4.1.1 Προανάκτηση με χρήση απομονωτών

Συχνή τακτική στην υλοποίηση προανάκτησης είναι η χρήση απομονωτών ροής, προκειμένου να μην μολύνεται η εκάστοτε κρυφή μνήμη για την οποία προορίζονται οι εντολές, είτε από πρόωρη κλήση αυτών, είτε από λανθασμένη επιλογή τους, τόσο σε μεθόδους προανάκτησης βήματος [33][34][35], όσο και σε μεθόδους συσχετιστικής προανάκτησης. Οι πρώτες υλοποιήσεις βασίστηκαν στη χρήση ενός απομονωτή στον οποίο αποθηκεύονται δεδομένα που προέρ-

Ταξινόμηση	Αντίστοιχο μοντέλο
Σταθερό	Σταθερό
Πρόσθεση	Δέλτα
Πρόσθεση, Πολλαπλασιασμός	Σύνθετο
Φόρτωση, Πρόσθεση	Διασυνδεδεμένη λίστα, indirect index
Φόρτωση, Πρόσθεση, Πολλαπλασιασμός	Σύνθετο

Πίνακας 2.2: *Αντιστοίχιση απλών και σύνθετων λειτουργιών μηχανής σε συνήθη μοντέλα πρόσβασης*

χονται αποκλειστικά από εντολές προανάκτησης, έως ότου γεμίσει. Τα δεδομένα παραμένουν σε αυτόν έως ότου ζητηθούν, οπότε μεταφέρονται στην κρυφή μνήμη, ή αντικαθίστανται από νέα. Αργότερα, υπήρξαν προτάσεις πολλαπλών βημάτων, όπου το πλήθος των απομονωτών είναι ανάλογο των διαφορετικών ροών που δύναται να εντοπιστούν [34], ενώ ακόμη σε ορισμένες εκ των μεθόδων εφαρμόζεται η μεταβολή του βήματος πρόβλεψης[36].

2.4.1.2 Προανάκτηση βήματος

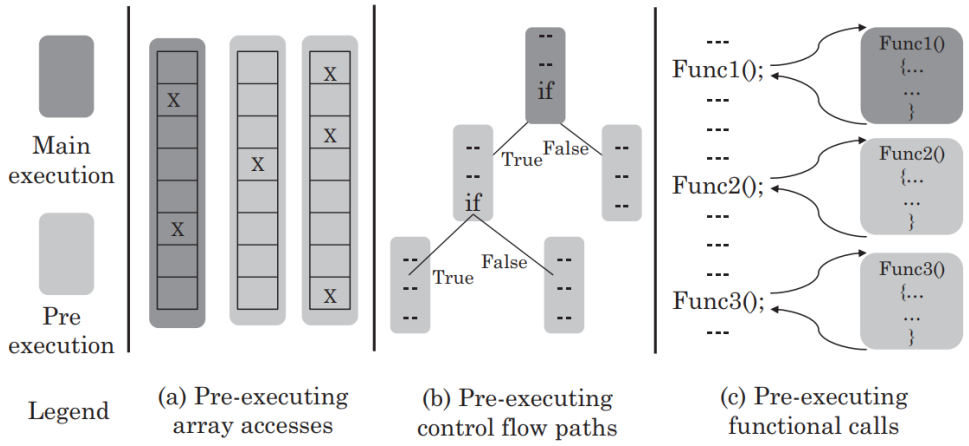
Οι πρώτες υλοποιήσεις βασίστηκαν στην τοπικότητα των δεδομένων. Μελετήθηκαν συνεπώς υλοποιήσεις επόμενης γραμμής και συνεχόμενων θέσεων στη μνήμη. Ακολούθησε η χρήση άλλων βημάτων σταθερών και καθοριζόμενων από τη ροή εντολών, ενώ στη συνέχεια επικράτησαν προτάσεις μεταβαλλόμενου κατά την εκτέλεση βήματος, εφόσον η ακολουθία δεδομένων φαινόταν αναντίστοιχη του αρχικά επιλεγόμενου, καθώς και πολλαπλών βημάτων συνδυαζόμενων σε ένα μοτίβο [36]. Μελετήθηκε ακόμη η παράλληλη καταγραφή πολλαπλών, πιθανώς και επικαλυπτόμενων, βημάτων ώστε να βρεθεί εκείνο που αντιπροσωπεύει καλύτερα την τρέχουσα εφαρμογή, καλύπτοντας το μεγαλύτερο δυνατό ποσοστό προσβάσεων [37].

2.4.1.3 Πρόβλεψη βασιζόμενη στο ιστορικό (History-based prediction)

Συχνά αναφερόμενη και ως συσχετιστική προανάκτηση, η μέθοδος που βασίζεται στην καταγραφή του ιστορικού αποτελεί την πλέον χρησιμοποιούμενη τεχνική προανάκτησης υλικού. Σε αυτές τις μεθόδους, μία μηχανή προανάκτησης χρησιμοποιείται για να παρακολουθεί τις προσβάσεις στα δεδομένα ή εναλλακτικά τις αστοχίες της κρυφής μνήμης από τις οποίες αναγνωρίζονται μοτίβα και, με βάση αυτά, πραγματοποιείται η πρόβλεψη των μελλοντικών προσβάσεων στη μνήμη και η έκδοση των αντίστοιχων εντολών. Η υλοποίηση γίνεται εξ ολοκλήρου στο εσωτερικό του επεξεργαστή και δεν απαιτεί καμία παρέμβαση του χρήστη. Οι δομές στις οποίες αποθηκεύεται το ιστορικό ποικίλουν και επιλέγονται ανάλογα με την ακολουθούμενη μέθοδο. Για παράδειγμα, σε στρατηγικές προανάκτησης βήματος διατηρείται ένας πίνακας προβλέψεων αναφορών (reference prediction table, RPT). Άλλες δομές που έχουν προταθεί είναι οι αλυσίδες Markov, για την δημιουργία και αποθήκευση πιθανοτικού διαγράμματος μεταβάσεων, καθώς και πίνακες διαφορών πολλαπλών επιπέδων, για τη μελέτη πέραν της πρόβλεψης των δεδομένων και του χρονισμού.

Με εφαρμογή τέτοιων μοντέλων έχουμε έγκαιρη και αποδοτική προανάκτηση με σημαντικά οφέλη σε ότι αφορά μη κοινά μοντέλα πρόσβασης, στα οποία δεν μπορεί να ανταποκριθεί η προανάκτηση βήματος. Χαρακτηριστικό παράδειγμα συσχετιστικής προανάκτησης, όπως αναφέρθηκε, αποτελεί η μέθοδος Markov [3], στην οποία η υλοποίηση ακολουθεί την υπόθεση ότι η αλληλουχία διευθύνσεων στις οποίες έχουμε αστοχίες ακολουθούν ένα διάγραμμα μεταβάσεων Markov.

Στην πλειονότητα τους οι συσχετιστικές τεχνικές προανάκτησης παρουσιάζουν πρόβλημα αποθηκευτικού χώρου καθώς απαιτείται η καταγραφή δεδομένων για μεγάλες αναδρομικές φάσεις



Σχήμα 2.2: *pre-execution*

της εφαρμογής και λαμβάνεται πλήρες ιστορικό το οποίο ιδανικά αποθηκεύεται στο ολοκληρωμένο για να μην δαπανάται το εύρος ζώνης. Το γεγονός αυτό καθιστά τις μεθόδους αυτές μη πρακτικές, παρά την αποτελεσματικότητα τους. Ως παράδειγμα, στη μέθοδο Markov, λόγω των μεταβολών στην εκτέλεση των προγραμμάτων ένα πλήρες διάγραμμα μπορεί να είναι αδύνατο να κατασκευαστεί και να αποθηκευτεί, είναι όμως εφικτός ο περιορισμός του αριθμού των κόμβων αλλά και του βαθμού καθενός εξ αυτών με βάση τις πιθανότητες επανεμφάνισης. Παρόλα αυτά, η βελτιστοποίηση αυτή αφορά μεμονωμένα τη δεδομένη τεχνική.

Ως γενική λύση, παρουσιάζεται η πρόταση των Nesbit και Smith για διαφορετική οργάνωση της αποθήκευσης του ιστορικού προανάκτησης με χρήση του global history buffer (GHB), ο οποίος αποτελείται από εγγραφές καθολικής διεύθυνσης αστοχίας και ενός δείκτη που οδηγεί στο ιστορικό διευθύνσεων. Με αυτόν τον τρόπο γίνεται αποσύνδεση του κλειδιού προανάκτησης από το απαιτούμενο αποθηκευμένο ιστορικό και συνεπώς δίνεται η δυνατότητα συνδυασμού κι αξιοποίησης περισσότερων τεχνικών προανάκτησης παράλληλα, με χρήση εναλλακτικών κλειδιών [4].

Ένας άλλος τρόπος με τον οποίο αντιμετωπίστηκε το πρόβλημα της απαιτούμενης μνήμης είναι η ομαδοποίηση διευθύνσεων. Μελετήθηκαν οι ακολουθίες που προκύπτουν ανά set με την πρόβλεψη να γίνεται με βάση επαναλαμβανόμενες αλληλουχίες συσχέτισης των tags. Η τεχνική αυτή [38] παρουσιάζει την ιδιά ακρίβεια με συσχετιστική προανάκτηση διευθύνσεων, έχοντας όμως μικρότερη απαίτηση αποθηκευτικού χώρου. Τέλος, μία ακόμη πρόταση [39] αποτελεί ο χωρισμός του αποθηκευτικου χώρου σε ζώνες με βάση τα λιγότερο σημαντικά βιτς, ομαδοποιώντας τις διευθύνσεις στις οποίες είναι κοινά.

2.4.1.4 Πρόωρη εκτέλεση ως μέθοδος προανάκτησης και προανάκτηση βοηθητικου νήματος (Runahead execution)

Σε αυτή τη μέθοδο χρησιμοποιούνται είτε αδρανείς κύκλοι του επεξεργαστή, είτε, σε πολυπύρηνα συστήματα, αδρανείς ή εν αναμονή πυρήνες. Ακόμη, σε επεξεργαστές που υποστη-

ρίζουν την πολυνημάτωση, όσο το κύριο νήμα εκτελεί τη δεδομένη εφαρμογή, ένα άλλο νήμα δύναται να εκτελεί μια πλήρη ή μειωμένη εκδοχή του προγράμματος. Σκοπός αυτών αποτελεί η παραγωγή διευθύνσεων για προανάκτηση. Δεδομένου ότι η βοηθητική εκτέλεση ακολουθεί τον κώδικά της αρχικής εφαρμογής, έχουμε υπολογισμό των διευθύνσεων προανάκτησης αντί της πρόβλεψης αυτών, περιορίζοντας έτσι τις ροές εκτέλεσης από το σύνολο των δυνατών στις πιθανές. Στο σχήμα 2.2 παρουσιάζονται ορισμένες χρήσεις πρόωρης εκτέλεσης για την κάλυψη διαφόρων ροών [40].

Τεχνικές που χρησιμοποιήθηκαν για αυτό το μοντέλο προανάκτησης εκμεταλλεύονται τους πλεονάζοντες πόρους με ποικίλους τρόπους. Στο [41] χρησιμοποιείται μία ξεχωριστή μηχανή πρόωρης εκτέλεσης η οποία δεν επιδρά στην κατάσταση του επεξεργαστή και βελτιώνει την ταχύτητα εκτέλεσης καθώς δεν επηρεάζεται από καθυστερήσεις στον απομονωτή αναδιάταξης (reorder buffer) και τις ουρές ανάκτησης (fetch queue). Στο [42] προτείνεται η παράλληλη εκτέλεση του κύριου κώδικα, χωρίς καμία μεταβολή αυτού, από τόσα νήματα όσες και οι προκύπτουσες εναλλακτικές ροές. Οταν το κύριο νήμα σταματά, περισσότεροι πόροι μπορούν να δοθούν στα βοηθητικά νήματα για την βελτίωση της συνολικής απόδοσης. Μία ακόμη πρόταση αποτελεί η δυνατότητα έναρξης βοηθητικών νημάτων όχι μόνο από το κύριο, αλλά και από τα ήδη εκτελούμενα βοηθητικά νήματα, γεγονός που οδηγεί στη βελτίωση της απόδοσης μέσω της αύξησης της επιθετικότητας [43]. Τέλος, σημαντική είναι η συνεισφορά της πρόωρης εκτέλεσης όταν συνδυάζεται με προσωρινή μνήμη δεικτών (pointer cache), οπότε υποβοηθάται η επιλογή της σωστής ροής, ή χρησιμοποιείται profiling, οπότε εντοπίζονται οι κρίσιμες περιοχές στις οποίες πρέπει να δοθούν επιπλέον πόροι, εκκινώντας βοηθητικά νήματα.

2.4.2 Υλοποιήσεις Λογισμικού

Στην πλειονότητα τους οι σύγχρονοι επεξεργαστές υποστηρίζουν κάποια μορφή εντολών ανάκτησης από τη μνήμη (fetch) οι οποίες μπορούν να χρησιμοποιηθούν για την υλοποίηση προανάκτησης. Η υλοποίηση τους μπορεί να είναι τόσο απλή όσο οι εντολές φόρτωσης από τη μνήμη ή αποθήκευσης σε αυτή, είτε να παρουσιάζει αυξημένη πολυπλοκότητα ώστε να παρέχονται και επιπλέον πληροφορίες στον επεξεργαστή, όπως ο τρόπος με τον οποίο θα χρησιμοποιηθούν τα δεδομένα της προανάκτησης. Βασική προϋπόθεση βέβαια της υλοποίησης είναι η μη πρόκληση εξαιρέσεων και μεγέθυνση της επιβάρυνσης σε σημείο που να καθίσταται ασύμφορη. Διασφαλίζεται δηλαδή η ορθότητα και η προανάκτηση λειτουργεί ως επιπρόσθετη λειτουργία βελτιστοποίησης.

Η πολυπλοκότητα της προανάκτησης λογισμικού, δεν έγκειται σε απαιτήσεις χώρου ή υλικού. Κύριο ζήτημα για τις μεθόδους αυτές αποτελεί ο χρονισμός, η επιλογή δηλαδή του σωστού σημείου για τοποθέτηση των εντολών προανάκτησης μεταξύ αυτών της εκτελούμενης εφαρμογής. Οι εντολές μπορούν να προστεθούν είτε από τον προγραμματιστή, είτε από το μεταγλωττιστή του προγράμματος κατά τη φάση βελτιστοποίησης. Σε αντίθεση με την πλειονότητα των βελτιστοποιήσεων, που έχουν μεγάλη συχνότητα μέσα στον κώδικα ή είναι ιδιαίτερα κουραστική η υλοποίηση τους χωρίς αυτοματοποιημένη μέθοδο, η προανάκτηση μπορεί να υλοποιηθεί αποτελεσματικά από τον προγραμματιστή. Μελέτες υποδεικνύουν πως η τοποθέτηση ακόμη

και λίγων εντολών προανάκτησης μπορεί να επιφέρει σημαντική βελτίωση στην απόδοση [44]. Παρόλα αυτά, αν επιδιώκεται η ελάχιστη δυνατή προσπάθεια ή η εφαρμογή έχει σημαντικές δυνατότητες προανάκτησης, τότε η χρήση του μεταγλωττιστή καθίσταται αναγκαία.

Στις υλοποιήσεις λογισμικού σημαντική είναι η συνεισφορά του profiling, της δυναμικής δηλαδή ανάλυσης του προγράμματος. Για την υλοποίηση του και τη συλλογή των απαιτούμενων πληροφοριών χρησιμοποιούνται πολλές τεχνικές όπως οι διακοπές υλικού, η οργάνωση του κώδικα, η προσομοίωση set εντολών, οι δείκτες απόδοσης, κα.

2.4.3 Συνδυαστικές Υλοποιήσεις

Κάνοντας σύγκριση μεταξύ προανάκτησης υλικού και λογισμικού, παρατηρούμε ότι ενώ η προανάκτηση υλικού μπορεί να ανταποκριθεί σε μεταβολές κατά τη διάρκεια της εκτέλεσης ή σε παραλλαγές της εισόδου καθώς και να προσαρμόσει την επιθετικότητά της, αποτυγχάνει για πολύ μικρές ροές και απαιτεί πολύπλοκες δομές για τον εντοπισμό μη κανονικών μοντέλων. Ομοίως, όταν ο αριθμός των ροών στην εφαρμογή υπερβαίνει τους διαθέσιμους πόρους, μπορεί να μην είναι ικανή για σαφή διαχωρισμό αυτών. Αντίθετα, σε αυτή την περίπτωση, η προανάκτηση λογισμικού μπορεί να προσθέσει ειδικές εντολές για κάθε ροή ξεχωριστά. Άλλες περιπτώσεις όπου η προανάκτηση λογισμικού υπερτερεί είναι η οριοθέτηση βρόχων και διαδικασιών καθώς και η αποφυγή της προανάκτησης εκτός των ορίων αυτών (ιδιαίτερα επιθετική προανάκτηση). Βέβαια, όσον αφορά το χρονισμό, το υλικό ανταποκρίνεται καλύτερα, καθώς για τον ακριβή υπολογισμό του βέλτιστου σημείου εκτέλεσης απαιτείται να ληφθούν υπόψη πολλές παράμετροι. Τέλος, σημαντικό είναι να σημειωθεί ότι στην προανάκτηση λογισμικού μπορεί να αυξηθεί κατά πολύ ο αριθμός των εκτελούμενων εντολών (έως και 100%), το οποίο αποτελεί σημαντική σπατάλη τους εύρους ζώνης ανάκτησης αλλά και εκτέλεσης των εντολών, αν και οι εφαρμογές που χρησιμοποιούν προανάκτηση συνήθως περιορίζονται από τη μνήμη και συνεπώς αύξηση του αριθμού των εντολών δεν οδηγεί απαραίτητα σε αύξηση του χρόνου ολοκλήρωσης.

Με βάση τα παραπάνω, παράλληλη χρήση προανάκτησης υλικού και λογισμικού αυξάνει τη δυνατότητα προσαρμογής σε μεγαλύτερη ποικιλία ροών, ενώ ακόμη οι αιτήσεις προανάκτησης λογισμικού μπορούν να χρησιμοποιηθούν για εκπαίδευση της προανάκτησης υλικού. Παρόλα αυτά, αυτή η επικοινωνία μπορεί να είναι επιζήμια όταν, για παράδειγμα, η προανάκτηση λογισμικού εμποδίζει το υλικό να εντοπίσει τις ροές δεδομένων ή όταν έχουμε λανθασμένα δεδομένα, μόλυνση της κρυφής μνήμης και περιορισμό του εύρους ζώνης.

Οι συνδυαστικές υλοποιήσεις γίνονται όλο και πιο δημοφιλείς σε πολυνηματικούς επεξεργαστές, καθώς δίνεται η δυνατότητα παράλληλης εκτέλεσης πολύπλοκων αλγορίθμων για την πρόβλεψη των μοντέλων πρόβλεψης στη μνήμη [28]. Οι μέθοδοι αυτοί απαιτούν υποστήριξη από το υλικό για την εκτέλεση του βοηθητικού νήματος, αλλά και από το λογισμικό για τον συγχρονισμό με το κύριο νήμα εκτέλεσης της εφαρμογής. Οι στρατηγικές προανάκτησης που βασίζονται σε εκτέλεση βοηθητικού νήματος είτε αναλύουν το ιστορικό των προσβάσεων στα δεδομένα, είτε εκτελούν πρόωρα τμήματα κώδικα που απαιτούν μεγάλο όγκο δεδομένων,

προετοιμάζοντας με αυτό τον τρόπο μία μοιραζόμενη κρυφή μνήμη.

2.5 Προκλήσεις και δυσκολίες στην προανάκτηση

Με βάση τα παραπάνω και πέρα από τις σχεδιαστικές επιλογές που πρέπει να ληφθούν για το σχεδιασμό ενός μηχανισμού προανάκτησης, σημαντικό είναι να αναφερθούν συνοπτικά τα ζητήματα που μειώνουν την απόδοση του μηχανισμού από τη βέλτιστη.

2.5.1 Επιβάρυνση υλοποίησης

Οι πιο απλοί μηχανισμοί προανάκτησης (next-line, stride) είναι εύκολα υλοποιήσιμοι, καλύπτουν όμως μικρό εύρος μοντέλων πρόσβασης στη μνήμη και έχουν συνεπώς περιορισμένη ακρίβεια. Αντίθετα, πιο εκλεπτυσμένοι prefetchers απαιτούν μεγάλο ποσό μεταδεδομένων. Αυτά αποθηκεύονται είτε στη μνήμη, οπότε έχουμε συχνή και ακριβή επικοινωνία για τη μεταφορά από και προς τη μονάδα επεξεργασίας, είτε βρίσκονται on-chip, γεγονός που απαιτεί μικρές δομές ενώ παράλληλα καταλαμβάνονται πολύτιμοι πόροι. Ακόμη, η μεταφορά δεδομένων σχετικά με τη διεύθυνση στην οποία παρουσιάστηκε η εκάστοτε αστοχία, όπως ο δείκτης προγράμματος PC counter καθώς και το σύνολο των συλλεγόμενων μεταδεδομένων, στον prefetcher, ειδικότερα σε χαμηλότερα επίπεδα της ιεραρχίας κρυφές μνήμες, απαιτεί μη τετριμμένες αλλαγές στο chip του επεξεργαστή (πχ wire-routing). Το ίδιο ισχύει και για την αποφυγή πλεοναζουσών εντολών προανάκτησης, οπότε η κρυφή μνήμη πρέπει είτε να υποστεί κάποια μετατροπή, είτε να ελέγχεται διαρκώς [45].

2.5.2 Συμβιβασμοί απόδοσης

Εξαιτίας των χαρακτηριστικών των σύγχρονων επεξεργαστών, όπως η εκτέλεση εντολών εκτός σειράς, η μείωση των αστοχιών μέσω της προανάκτησης μπορεί να μην οδηγεί απαραίτητα σε βελτίωση της απόδοσης. Αφαιρώντας τις καθυστερήσεις πρόσβασης, δίνεται η δυνατότητα στον επεξεργαστή να προχωρήσει σημαντικά πιο μπροστά σε μονοπάτια εκτέλεσης που δεν έχουν μελετηθεί προηγουμένως και πιθανώς θα οδηγούν σε νέες αστοχίες οι οποίες δεν έχουν ληφθεί υπόψη [29]. Ακόμη προκειμένου να έχουμε τα δεδομένα από τη μνήμη έγκαιρα (timely prefetch) πρέπει να δοθεί σημασία σε πολλές πληροφορίες χρονισμού, όπως cache miss latencies, γεγονός ιδιαίτερα δύσκολο για υλοποιήσεις λογισμικού.

2.5.3 Μόλυνση της κρυφής μνήμης και διαμοιρασμός πόρων

Όσο ο όγκος των δεδομένων που φέρνουμε στην κρυφή μνήμη αυξάνεται, τόσο αυξάνεται ο ρυθμός των hits χάρη στην καλύτερη αξιοποίηση του αποθηκευτικού χώρου. Αυτό όμως συμβαίνει μόνο μέχρι το πλήθος των δεδομένων που απαιτούνται να ξεπεράσει το μέγεθος της

κρυφής μνήμης. Από αυτό το σημείο και έπειτα τα blocks που φορτώνονται μέσω της προανάκτησης αντικαθιστούν δεδομένα που έχουν ζητηθεί ρητά από την εκτελούμενη εφαρμογή ή άλλα δεδομένα προανάκτησης. Όσο αυξάνεται ο βαθμός, ο ρυθμός και η απόσταση της προανάκτησης, τόσο αυξάνονται και οι πιθανότητες να αντικατασταθούν δεδομένα που είναι ακόμη χρήσιμα για την εκτέλεση, δημιουργώντας έτσι νέες αστοχίες που δεν θα είχαν συμβεί χωρίς την παρέμβαση στη ροή των δεδομένων, γεγονός το οποίο με τη σειρά του μπορεί να οδηγήσει σε εκ νέου ενεργοποίηση του μηχανισμού. Κάποιες τεχνικές για να επιλυθεί το πρόβλημα αυτό, χρησιμοποιούν για τα δεδομένα που προέρχονται από εντολές προανάκτησης ειδικούς επιπρόσθετους καταχωρητές. Για να γίνει όμως, είτε παράλληλη, είτε σειριακή, χρήση αυτών απαιτούνται τόσο ενέργεια όσο και επιπλέον χρόνος, ενώ ακόμη η μέθοδος απαιτείται αναδιάρθρωση της αρχιτεκτονικής του chip για την τοποθέτηση των καταχωρητών και αποκλείει το διαμοιρασμό του χώρου μεταξύ demand fetched και prefetched δεδομένων. Είναι σαφές ότι η επίτευξη ισορροπίας απαιτεί προσεκτική επιλογή των παραμέτρων και της επιθετικότητας.

2.5.4 Προκλήσεις αξιοπιστίας

Τέλος, η προανάκτηση μπορεί να αυξήσει το soft error rate αυξάνοντας το χρόνο παραμονής των δεδομένων στην κρυφή μνήμη [46]. Ακόμη αυξάνοντας τις εγγραφές, μπορούν να προκληθούν μόνιμα σφάλματα και να μειωθεί η διάρκεια ζωης σε μη πτητικές μνήμες (δεν χάνουν τα δεδομένα δε διακοπή της τροφοδοσίας) που έχουν περιορισμένη αντοχή εγγραφών.

Κεφάλαιο 3

Μηχανική Μάθηση και Τεχνητά Νευρωνικά Δίκτυα

Η Μηχανική Μάθηση (Machine Learning - ML) αποτελεί έναν από τους πλέον γνωστούς κλάδους της τεχνητής νοημοσύνης (Artificial Intelligence - AI) και αφορά τη μελέτη αλγορίθμων αυτόματης βελτιστοποίησης μέσω εκπαίδευσης για την επίτευξη ενός επιθυμητού αποτελέσματος. Το αποτέλεσμα αυτό καθορίζει και τις μετρικές για την αξιολόγηση της απόδοσης και της ορθής λειτουργίας της μεθόδου. Είναι στενά συνδεδεμένη και συχνά συγχέεται με υπολογιστική στατιστική, καθώς τόσο η μεν οσο και η δε επικεντρώνονται στην πρόβλεψη μέσω υπολογιστών, ενώ η ανάλυση των μοντέλων της αποτελεί τομέα της θεωρητικής πληροφορικής.

3.1 Τύποι προβλημάτων και εργασιών

Κύριο κριτήριο μεταξύ των μεθόδων μηχανικής μάθησης αποτελούν τα δεδομένα εισόδου με τα οποία τροφοδοτείται ο εκάστοτε αλγόριθμος. Έτσι διαχωρίζουμε σε τρεις διακριτές μεθόδους: Μάθηση με Επίβλεψη (Supervised Learning), Μάθηση χωρίς Επίβλεψη (Unsupervised Learning) και Ενισχυμένη Μάθηση (Reinforcement Learning).

3.1.1 Μάθηση με επίβλεψη

Κατά την επιτηρούμενη μάθηση ο αλγόριθμος τροφοδοτείται με επισημειωμένα δεδομένα εκπαίδευσης (labeled training data), τα οποία είναι ζεύγη ενός διανύσματος εισόδου και μίας επιθυμητής τιμής εξόδου. Τα προβλήματα μάθησης με επίβλεψη χωρίζονται σε:

- Προβλήματα παλινδρόμησης (Regression): Τα δεδομένα εισόδου χαρακτηρίζονται από μια τιμή (τιμή εξόδου), συνήθως συνεχή, την οποία ο αλγόριθμος καλείται μετα την εκπαίδευση του να προβλέψει. Ένα από τα πιο χαρακτηριστικά προβλήματα παλινδρόμησης αποτελεί η πρόβλεψη των μετοχών στο χρηματιστήριο.
- Προβλήματα ταξινόμησης (Classification): Στην περίπτωση αυτή η επιθυμητή έξοδος ανήκει σε ένα σύνολο τιμών (διακριτών), το οποίο αντιπροσωπεύει τις κατηγορίες ή

κλάσεις στις οποίες θέλουμε να ταξινομήσουμε τα δεδομένα εισόδου. Παράδειγμα προβλήματος ταξινόμησης αποτελεί ο διαχωρισμός φωτογραφιών ανθρώπων με βάση το φύλο.

3.1.2 Μάθηση χωρίς επίβλεψη

Σε αυτή την περίπτωση τα δεδομένα που δίνονται προς εκπαίδευση δεν είναι επισημειωμένα και συνεπώς δεν δίνεται κατά την εκπαίδευση η επιθυμητή έξοδος. Αντίθετα, το σύστημα επιδιώκει να δημιουργήσει συστάδες (Clusters) και να τοποθετήσει σε αυτές τα δεδομένα που παρουσιάζουν μεταξύ τους τις μεγαλύτερες ομοιότητες. Σε μεγάλο βαθμό τα προβλήματα αυτά δεν είναι αυστηρώς καθορισμένα εξ αρχης, καθώς δεν γνωρίζουμε το πλήθος των ομάδων που θα πρέπει ή θα θέλαμε να δημιουργηθούν. Παράδειγμα αλγορίθμων μάθησης χωρίς επίβλεψη είναι οι γενετικοί αλγόριθμοι, οι οποίοι αποτελούν μια μέθοδο εύρεσης βέλτιστων λύσεων και χρησιμοποιούνται στη δημιουργία δεδομένων για την προσέγγιση συναρτήσεων.

3.1.3 Ενισχυμένη Μάθηση

Ως συνδυασμός των δύο προηγούμενων μεθόδων, στην ενισχυμένη μάθηση ο αλγόριθμος τροφοδοτείται με δεδομένα μη επισημασμένα, αλλά διατίθεται ένα σύστημα τιμωρίας και ανταμοιβής (Punish and Reward Method) μέσω του οποίου ελέγχεται η πρόοδος προς τη σωστή κατεύθυνση. Το πλεονέκτημα της μεθόδου αυτής συνίσταται στην εξ ορισμού δυνατότητα αλληλεπίδρασης του συστήματος μάθησης με το περιβάλλον. Γνωστές εφαρμογές ενισχυτικής μάθησης αποτελούν ο έλεγχος ρομπότ καθώς και επιτραπέζια παιχνίδια όπως το σκάκι.

3.2 Τεχνητά Νευρωνικά δίκτυα

3.2.1 Ιστορική αναδρομή

Παρόλο που το ενδιαφέρον της επιστημονικής κοινότητας παρουσίασε ραγδαία αύξηση, ουσιαστικά, την τελευταία δεκαετία, οδηγώντας στην ευρύτατη μελέτη και χρήση τους, το πρώτο υπολογιστικό μοντέλο νευρωνικού δικτύου δημιουργείται το 1943 (McCulloch & Pitts). Ακολούθησε, την ίδια δεκαετία, το μαθηματικό μοντέλο της μάθησης του εγκεφάλου (Hebbian learning), το οποίο εισήγαγε ο Donald Hebb και προσομοίωσαν στη συνέχεια οι Fahrley και Clark (1954). Το 1957 ο Rosenblatt προτείνει το στοιχειώδες τεχνητό νευρωνικό δίκτυο του απλού αισθητήρα που ονόμαστηκε Perceptron, ενώ το 1969 αποδεικνύεται μαθηματικά η αδυναμία επίλυσης μη γραμμικών προβλημάτων με μοντέλα ενός επιπέδου (Minsky & Papert). Η ευρέως γνωστή μέθοδος της οπισθοδρόμησης ή μέθοδος ανάστροφης μετάδοσης (back propagation) παρουσιάστηκε επίσημα και έγινε γνωστή με αυτό τον όρο απο τους Rumelhart, Hinton & Williams το 1986, παρόλο που αναπτύχθηκε ανεξαρτητα από πολλούς ερευνητές

ήδη από τις αρχές της δεκαετίας του 60, ενώ ο πρώτος που πρότεινε την εφαρμογή της στα νευρωνικά δίκτυα ήταν ο Paul Werbos το 1974.

Σημαντικό είναι να αναφερθεί στο σημείο αυτό πως αφετηρία για την επανεκκίνηση των μελετών που οδήγησε στό σημερινό εύρος χρήσης των νευρωνικων δικτύων μετά το πέρας της δεκαετίας του 60, οπότε παρατηρείται μία παύση των εξελίξεων, δεν αποτελεί άλλο παρά η τεχνολογική πρόοδος στο κομμάτι των VLSI, που οδήγησε στην αύξηση της υπολογιστικής ισχύος και κατέστησε υλοποιήσιμα τα τεχνητά νευρωνικά δίκτυα και πέραν του θεωρητικού επιπέδου.

Το 1989 αναπτύχθηκαν τα συνελικτικά νευρωνικά δίκτυα που αποτέλεσαν τη βασική τεχνολογία για την ανάλυση εικόνων. Στις αρχές της δεκαετίας που ακολούθησε τα αποτελέσματα σε επίπεδο απόδοσης δεν ήταν τα αναμενόμενα, κυρίως λόγω των δεδομένων που χρησιμοποιούνταν για την εκπαίδευση, ενώ ακόμη δεν υπήρχε σε βάθος κατανόηση των μοντέλων αλλά και της απόδοσης και αποτελεσματικότητας τους. Ακολούθησε, το 1995, η ανάπτυξη των δικτύων μακράς-βραχέας μνήμης (Long short-term memory - LSTM), τα οποία χρησιμοποιούνται ευρύτατα σήμερα στην επεξεργασία φυσικής γλώσσας (Natural Language Processing - NLP) και άλλες χρονοεξαρτώμενες εφαρμογές.

Παρόλο που μεχρι το σημείο αυτό έχουν προταθεί όλα τα μοντέλα που χρησιμοποιούνται μέχρι και σήμερα και αποτελούν τη βάση της μηχανικής μάθησης, η εφαρμογη τους δεν αποδίδει εξίσου με άλλες, απλούστερες και ίσως πιο κατανοητές μεθόδους, οπότε και δεν εφαρμόζονται. Το 2010 η ήδη υπάρχουσα τεχνολογία των πολυεπίπεδων μοντέλων έρχεται ξανά στο προσκήνιο με τον όρο deep learning. Δυο ακόμη σημαντικά στοιχεία που έδωσαν την απαιτούμενη ώθηση για εξέλιξη των νευρωνικών δικτύων ήταν η χρήση της κάρτας γραφικών (GPU) ως υπολογιστικής πλατφόρμας, παρέχοντας τη δυνατότητα της παραλληλοποίησης, αλλά και η ανάπτυξη του ImageNet, μιας συλλογής δεδομένων που περιελάμβανε ήδη πάνω από ένα εκατομμύριο κατηγοριοποιημένες εικόνες. Ο συνδυασμός των δύο στην υλοποίηση ενός πολυεπίδου συνελικτικού δικτύου έδωσε αποτελέσματα συγκρίσιμα με άλλες δημοφιλείς χρησιμοποιούμενες μεθόδους, κινητοποιώντας εκ νέου την επιστημονική κοινότητα για ενασχόληση και μελέτη. Τέλος, αξιοσημείωτη είναι, το 2015, η επίτευξη απόδοσης που ξεπερνά την ανθρώπινη ικανότητα με την δημιουργία του AlphaGo, αλγορίθμου που σχεδιάστηκε με βαση τα συνελικτικά νευρωνικά δίκτυα (CNN) και την ενισχυτική μάθηση (reinforcement learning) και φανέρωσε το εύρος των δυνατοτήτων τους.

3.2.2 Εκπαίδευση Νευρωνικών Δικτύων

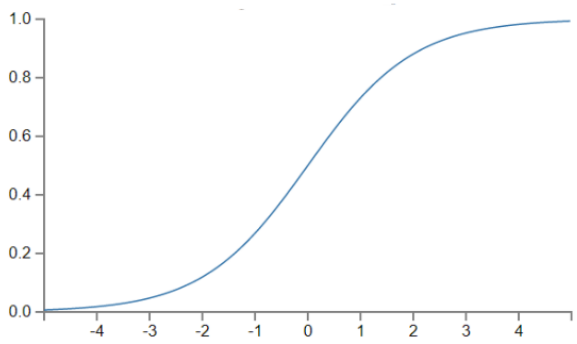
Σε κάθε νευρωνικό δίκτυο η διαδικασία της εκπαίδευσης πρέπει να δύναται να ολοκληρωθεί σε πεπερασμένο χρόνο, ελαχιστοποιώντας παράλληλα κατά το μέγιστο την καταναλισκόμενη υπολογιστική ισχύ. Για την επίτευξη του σκοπού αυτού πολύ σημαντική αποδεικνύεται η επιλογή των κατάλληλων μεθόδων, εργαλείων αλλά και των δεδομένων εκπαίδευσης, ενώ ακόμη κύρια ιδιότητα αποτελεί και η γενίκευση, χωρίς την οποία το δίκτυο δεν μπορεί να ανταποκριθεί σε νέα δεδομένα, πέραν του συνόλου εκπαίδευσης και ελέγχου.

Σε πρακτικό επίπεδο, η εκπαίδευση ενός νευρωνικού δικτύου δεν αποτελεί αλλο παρά μία επαναληπτική διαδικασία κατά την οποία οι παράμετροι του δικτύου μεταβάλλονται με σκοπό τον υπολογισμό της επιθυμητής εξόδου (πρόβλεψη). Το πλήθος των επαναλήψεων, οι οποίες αναφέρονται ως εποχές, καθορίζει την ικανότητα γενίκευσης και κατηγοριοποίησης του μοντέλου. Με βάση την ανανέωση των παραμέτρων η εκπαίδευση διαχωρίζεται σε On-Line και Batch Learning. Στην πρώτη περίπτωση κάθε μεμονωμένο στιγμιότυπο των δεδομένων εκπαίδευσης ανανεώνει τα βάρη του δικτύου. Αντίθετα, στη δεύτερη περίπτωση, τα δεδομένα εισόδου χωρίζονται σε πακέτα (batches) με βάση τα οποία ως σύνολο υπολογίζονται τα νέα βάρη. Η διαδικασία της εκπαιδευσης λειτουργεί κυρίως με υπολογισμό διαφορών και παραγώγων, ενώ εξαρτάται από πολλές παραμέτρους όπως ο αριθμός των επαναλήψεων καθώς και οι συναρτήσεις ενεργοποίησης, κόστους και βελτιστοποίησης. Παρακάτω αναπτύσσονται ορισμένες από αυτές.

3.2.2.1 Συνάρτηση Ενεργοποίησης (Activation Function)

Οι συναρτήσεις ενεργοποίησης αποτελούν ένα από τα βασικότερα στοιχεία της βαθιάς μάθησης, αφού καθορίζει την έξοδο κάθε κόμβου, και συνεπώς του μοντέλου στο σύνολό του, δεδομένης μίας εισόδου ή ενός συνόλου εισόδων. Επηρεάζει συνεπώς κάθε μετρική του μοντέλου, από την υπολογιστική απόδοση της εκπαίδευσης του εως την ικανότητα και το βαθμό σύγκλισης και την ακρίβεια (accuracy) του. Μερικες βασικές συναρτήσεις ενεργοποίησης παρουσιάζονται παρακάτω.

- Σιγμοειδής συνάρτηση (Sigmoid function): Αποτελεί μια από τις πρώτες συναρτήσεις που χρησιμοποιήθηκαν χρονικά. Αποτέλεσμα της συνάρτησης για κάθε τιμή της εισόδου βρίσκεται στο διάστημα (0,1) με τις μικρότερες τιμές να αντιστοιχίζονται κοντά στο 0 και τις μεγαλύτερες κοντά στο ένα, χωρίς όμως να λαμβάνει ποτέ αυτές τις τιμές (Σχήμα 3.1). Το γεγονός πως οι τιμές κλιμακώνουν καθαυτόν τον τρόπο, οδηγεί σε πολύ μικρές τιμές κλίσης, κάνοντας τη μάθηση αρκετά αργή, με πιθανότητα ακόμη και να σταματήσει. Το φαινόμενο αυτό ονομάζεται εξασθένιση κλίσης (Vanishing Gradient) και προκαλεί αρκετά προβλήματα στην διαδικασία μάθησης.



Σχήμα 3.1: *Σιγμοειδής συναρτηση*

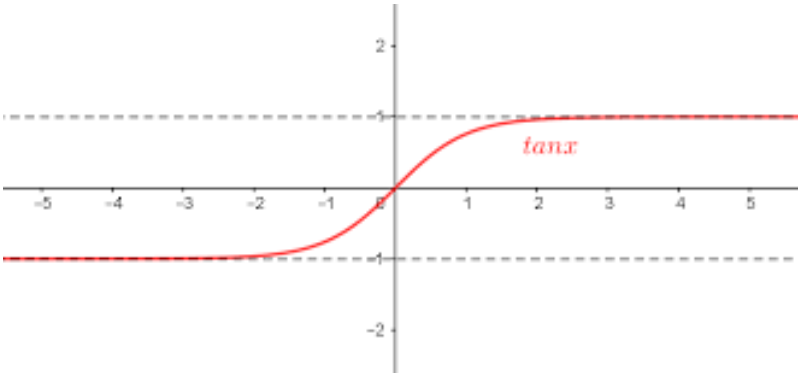
Μαθηματικά η σιγμοειδής συνάρτηση υπολογίζεται ως:

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

- Υπερβολική εφαπτομένη (Hyperbolic tangent): Στη συνάρτηση αυτή το αποτέλεσμα για κάθε τιμή εισόδου με βάση ένα κατώφλι αντιστοιχίζεται στο διάστημα (-1,1) (Σχήμα 3.2). Σε σύγκριση με τη σιγμοειδή συνάρτηση δεν μεταβάλλει αισθητά τις τιμές που βρίσκονται κοντά στο 0 με συνέπεια να συμβάλλουν και εκείνοι κατά τη διαδικασία διάδοσης. Το πλεονέκτημα αυτό αποτελεί το βασικό λόγο για τον οποίο χρησιμοποιείται σε επαναλαμβανόμενα νευρωνικά δίκτυα (Recurrent Neural Networks). Το πρόβλημα εξασθένησης κλίσης εμφανίζεται και σε αυτή την περίπτωση αλλά σε μικρότερο βαθμό συγκριτικά με τη σιγμοειδή συνάρτηση.

  Η υπερβολική εφαπτομένη δίνεται από τη σχέση:

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



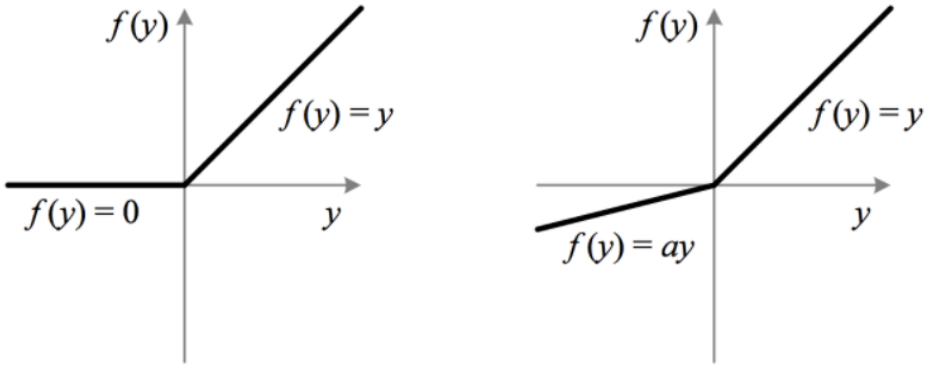
Σχήμα 3.2: *Υπερβολική εφαπτομένη*

- Ανορθωμένη γραμμική συνάρτηση ή συνάρτηση ράμπας (Rectified Linear Unit - ReLU): Αποτελεί την πλέον δημοφιλή και χρησιμοποιούμενη συνάρτηση ενεργοποίησης. Με τη χρήση της συνάρτησης αυτής οι είσοδοι των οποίων οι τιμές είναι μικρότερες του μηδενός δεν συνυπολογίζονται κατα τη διαδικασία της μάθησης όπως γίνεται φανερό τόσο από τον τύπο όσο και από το αντίστοιχο σχήμα (Σχήμα 3.3).

  Η ReLU δίνεται από την εξίσωση:

$$f(x) = max(0, x)$$

  Το γεγονός αυτό καθιστά το σύνολο των νευρώνων που έχουν αρνητικές τιμές ανενεργό με συνέπεια τη μείωση του χρόνου εκμάθησης του δικτύου κάνοντας το ιδιαίτερα αποδοτικό. Η ιδια ιδιότητα δημιουργεί και προβλήματα, καθώς η οριζόντια τιμη των αρνητικών τιμών συνεπάγεται σταθερή και μηδενική παράγωγο, δηλαδή μηδενική μεταβολή της τιμής καθόλη τη διάρκεια της εκπαίδευσης. Ο περιορισμός αυτός στη βιβλιογραφία είναι γνωστός και ως ′dying ReLU′ δηλώνοντας την αδυναμία επαναχρησιμοποίησης νευρώνα



Σχήμα 3.3: *ReLU(a) και Leaky ReLU(β)*

που έχει λάβει αρνητική τιμή. Για την αντιμετώπιση του προβλήματος αυτού, αντί της RelU χρησιμοποιείται μία παραλλαγή της, η Leaky ReLU. Στην περίπτωση αυτή, αντί της αντιστοίχης κάθε αρνητικής τιμής στην οριζόντια μηδενική γραμμή, χρησιμοποιείται μία γραμμική συνάρτηση πολύ μικρής κλίσης, ώστε να δύναται κάθε νευρώνας που έχει λάβει αρνητική τιμή να ανακάμπτει (recover).

- Συνάρτηση Softmax: Είναι μία γενίκευση της σιγμοειδούς συνάρτησης και χρησιμοποιείται όταν οι επιθυμητές κλάσεις ταξινόμησης είναι πάνω από δύο. Ουσιαστικά κανονικοποιεί τις τιμές της εξόδου ώστε, κατανεμημένες στο διάστημα [0,1] με συνολικό αθροισμα το 1, να δηλώνουν την πιθανότητα του τυχαίου δείγματος εισόδου να ανήκει σε κάθε κλάση.

  Η συναρτηση softmax δίνεται από τον τύπο:

$$Softmax(z_i) = \frac{e^{z_j}}{\sum_{j=0}^{N} e^{z_j}}$$

3.2.2.2 Συνάρτηση Κόστους (Cost Function)

Η συνάρτηση Κόστους χρησιμοποιείται ως μετρική για τον έλεγχο της επαναληπτικής διαδικασίας της εκπαίδευσης. Συνήθως συμβολίζεται με J(ϑ) και υπολογίζει την απόκλιση της τιμής εξόδου για τις δεδομένες παραμέτρους από την επιθυμητή τιμή. Η πλέον γνωστή συνάρτηση κόστους είναι η Απώλεια Διατροπικής Εντροπίας (Crossentropy Loss), η οποία υπολογίζεται ως εξής:

$$J(\theta) = -H(y,p) = -\sum_{i=0}^{N} \widehat{y}_i log(p_i, j)$$

όπου N ο συνολικός αριθμός κλάσεων, y_i η εκτιμώμενη τιμή για την παρατήρηση i και $p_i j = p(\widehat{y_i} = j|x)$ η posterior πιθανότητα το i δείγμα να ανήκει στην j κλάση. Η παραπάνω εξίσωση υπολογίζει το κόστος κάθε δείγματος εισόδου. Για τον υπολογισμό του συνολικού κόστους αρκεί να υπολογίσουμε τον αριθμητικό μέσο όρο των επιμέρους σφαλμάτων.

Μία ακόμη γνωστή και συχνά χρησιμοποιούμενη συνάρτηση κόστους είναι το Μέσο Τετραγωνικό Σφάλμα (Mean Squared error), το οποίο υπολογίζει την τετραγωνική απόσταση μεταξύ επιθυμητής τιμής και πρόβλεψης και δίνεται από τη σχέση:

$$J(\theta) = \frac{1}{n}\sum_{i=0}^{M}(Y_i - \widehat{y}_i)^2$$

Η συνάρτηση αυτή δεν υπολογίζει πιθανοτικές διαφορές αλλά καθαρά αριθμητικές, γεγονός που δημιουργεί αρκετά προβλήματα αφού οι κλάσεις δεν είναι κατάνάγκη διατεταγμένοι αριθμοί.

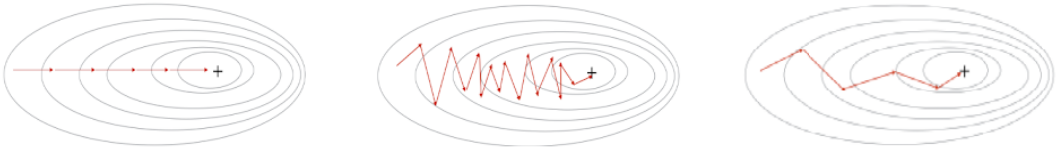
3.2.2.3 Τεχνικές και Αλγόριθμοι Βελτιστοποίησης - Optimization Techniques and Algorithms

Οι αλγόριθμοι βελτιστοποίησης είναι υπεύθυνοι για τη μείωση των απωλειών και την μέγιστη δυνατή ακρίβεια των αποτελεσμάτων. Πολλοί έχουν αναπτυχθεί τα τελευταία χρόνια, με κάθε πρόταση να παρουσιάζει τόσο πλεονεκτήματα όσο και μειονεκτήματα. Η πλέον γνωστή τεχνική που χρησιμοποιείται για την εκπαίδευση των τεχνιτών νευρωνικών δικτύων είναι η Οπισθοδιάδοση ή, εναλλακτικά, Οπίσθια Ανατροφοδότηση Σφάλματος (Back Propagation). Αποτελεί στην πραγματικότητα μια οικογένεια μεθόδων που ακολουθούν έναν αλγόριθμο βελτιστοποίησης βασιζόμενο στην κλίση. Κύριο χαρακτηριστικό τους είναι επαναληπτική και αναδρομική μέθοδος για τον υπολογισμό εκ νέου των βαρών του δικτύου. Πιο συγκεκριμένα, τα νεα βάρη επανακαθορίζονται με τον υπολογισμό της κλίσης του κόστους ως προς την κάθε παράμετρο του δικτύου. Η διαδικασία αυτή έπεται της εμπρόσθιας τροφοδότησης (Feed Forward) του δικτύου κατά την οποία τροφοδοτείται το δίκτυο με μία είσοδο x, βάση της οποίας υπολογίζεται η προβλεπόμενη τιμή εξόδου y. Η προσαρμογή του κάθε βάρους πραγματοποιείται με την πρόσθεση της κλίσης της συνάρτησης κόστους ως προς το βάρος του νευρώνα k, $\frac{\partial J(\theta)}{\partial w_k}$. Η διαδικασία αυτή εκτελείται επαναληπτικά από την έξοδο του δικτύου προς την είσοδο και βασίζεται στον υπολογισμό μερικών παραγώγων με τον κανόνα της αλυσίδας. Βασικό πλεονέκτημα της μεθόδου αποτελεί το γεγονός ότι εγγυάται τον γρήγορο υπολογισμό αλλά και την άμεση αναπροσαρμογή τυχόν αυθαίρετης τιμής πόλωσης σε κάποιον νευρώνα.

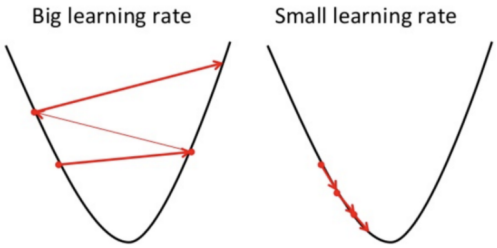
Μία άλλη, εξίσου γνωστή, μέθοδος είναι σύγκλιση με ελάττωση της παραγώγου, γνωστή και ως αλγόριθμος απότομης καθόδου (Gradient descent). Η λειτουργία του αλγορίθμου βασίζεται στην ελαχιστοποίηση (ή τη μεγιστοποίηση) μίας συνάρτησης κόστους με τη χρήση της κλίσης(gradient) των παραμέτρων του προβλήματος. Ο υπολογισμός επαναλαμβάνεται έως ότου επέλθει σύγκλιση ή να ολοκληρωθεί ένας ορισμένος αριθμός επαναλήψεων (Termination Criteria) και οι παράμετροι του δικτύου ανανεώνονται με βάση την παρακάτω εξίσωση:

$$\theta_{t+1} = \theta_t - \lambda \nabla_\theta J(\theta)$$

με ϑ το σύνολο των παραμέτρων του προβλήματος, λ τον ρυθμό μάθησης (learning rate) και $J(\theta)$ τη συνάρτηση κόστους, όπως αναλύθηκε παραπάνω. Στην πλειονότητα των περιπτώσεων εφαρμόζονται παραλλαγές του αρχικού αλγορίθμου, όπως ο Στοχαστικός αλγόριθμος τάχι-



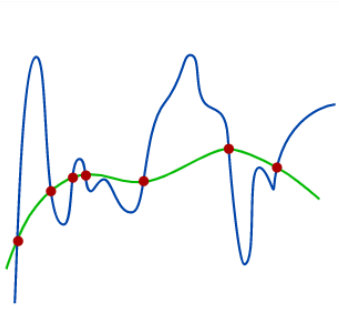
Σχήμα 3.4: *Σύγκλιση: (α)gradient descent, (β)stochastic gradient descent, (γ)mini-batch gradient descent*



Σχήμα 3.5: *Ρυθμός μάθησης (α)Μεγάλο βήμα (β) Μικρό βήμα*

στης κατάβασης(Stochastic Gradient Descent - SGD) στον οποίο αντί της χρήσης όλου του δείγματος για κάθε ανανέωση παραμέτρου σε μία δεδομένη επανάληψη, όπως ορίζεται από τη μέθοδο, χρησιμοποιείται μόνο ένα μεμονομένο στοιχείο, είτε ένα υποσύνολο στοιχείων (mini-batch) αυτού, γεγονός που καθιστά τη μέθοδο σαφώς ταχύτερη, αλλά με μεγαλύτερη απόκλιση σφάλματος, αν και οχι σε σημείο που να καθιστά τη βελτιστοποίηση μη αποδοτική.

Σαφώς ο αλγόριθμος έχει ορισμένα μειονεκτήματα, τα οποία οδηγούν στη χρήση περαιτέρω παραλλαγών. Ο σταθερός ρυθμός μάθησης λ καθώς και η πολύ αργή του σύγκλιση σε προβλήματα με μεγάλο πλήθος δεδομένων λόγω της μεγάλης διασποράς της κλίσης οδήγησαν στη δημιουργία αλγορίθμων προσαρμοσμένων σε αυτά. Ο αλγόριθμος Adam (adaptive moment estimation) βασίζεται στη λειτουργία του SGD χρησιμοποιώντας μεταβλητό ρυθμό μάθησης για κάθε παράμετρο-βάρος του δικτύου, χρησιμοποιώντας τόσο την πρώτη όσο και τη δεύτερη βαθμίδα κλίσης (First-Second Moment of Gradient) για τη λειτουργία του, ενώ παράλληλα, για να αποφύγει φαινόμενα ταλάντωσης όπως παρουσιάζονται στο σχήμα 3.5 χρησιμοποιεί δύο μεταβλητές παράλειψης (Forget Variables) που επιταχύνουν τη διαδικασία σύγκλισης. Μεταβλητό ρυθμό μάθησης χρησιμοποιεί και ο αλγόριθμος Adagrad που ταυτόχρονα προσαρμόζει μεγάλες ενημερώσεις για σπάνιες παραμέτρους και μικρότερες για πιο συχνές, γεγονός που τον καθιστά θεωρητικά ιδιαίτερα εύρωστο, ενώ παρουσιάζει σημαντική μείωση του ρυθμού μάθησης με την παροδο των εποχών, καθιστώντας αδύνατη την εκπαιδευση. Για τη βελτίωσή του προτάθηκε ως επέκταση ο Adadelta, ο οποίος χρησιμοποιεί αποκλειστικά τη δεύτερη βαθμίδα κλίσης ως ρυθμό μάθησης σε συνδυασμό με τους προηγούμενα χρησιμοποιούμενους που ανήκουν σε ένα χρονικό παράθυρο. Δηλαδή ο ρυθμός μάθησης σε μία εποχή επηρεάζεται από τους N προηγούμενους ρυθμούς αντί του συνόλου, το οποίο χρησιμοποιούσε ο Adagrad.



Σχήμα 3.6: *Συναρτήσεις με μηδενικό κόστος στα επιλεγμένα σημεία, με τη μία εξ αυτών (μπλε) να είναι πιο πολύπλοκη, προσαρμοσμένη με μεγαλύτερη ακρίβεια στα δεδομένα εισόδου και συνεπως με μικρότερη δυνατότητα γενίκευσης (overfitted)*

3.2.2.4 Κανονικοποίηση Δικτύου (model regularization)

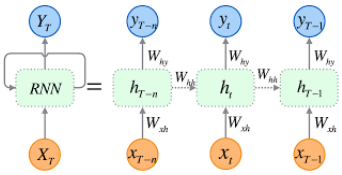
Πέραν του ζητήματος της βελτιστοποίησης, σημαντικό είναι να καταγραφουν και τεχνικές που αφορούν τη μείωση του υπολογιστικού κόστους αλλά και την ικανότητα γενίκευσης του μοντέλου. Η κανονικοποίηση ουσιαστικά ωθεί το μοντέλο σε παραμέτρους που αντιπροσωπεύουν λιγότερο πολύπλοκες συναρτήσεις, μειώνοντας με αυτό τον τρόπο τον κίνδυνο της υπερπροσαρμογής (overfitting).

Ένας τρόπος για την εφαρμογή κανονικοποίησης είναι η προσθήκη στη συνάρτηση κόστους ενός επιπλέον όρου $R(f)$. Στην πλειονότητα των προβλημάτων μηχανικής μάθησης χρησιμοποιείται ως όρος κανονικοποίησης η L_2 νόρμα των παραμέτρων και η συνάρτηση κόστους εκφράζεται από τη σχέση:

$$J'(\theta) = J(\theta) + \gamma \sum_i (\theta_i)$$

όπου γ ο συντελεστής κανονικοποίησης που καθορίζει τη συμβολή του όρου κανονικοποίησης στη συνάρτηση κόστους. Έχει αποδειχθεί [47] ότι η κανονικοποίηση οδηγεί σε τιμές παραμέτρων που μειώνουν αισθητά το κόστος, και αποτρέπει από τη λήψη τιμών που οδηγούν σε άνοδο του, με αποτέλεσμα τη μείωση της πιθανότητας ταλάντωσης του αλγορίθμου βελτιστοποίησης και συνεπών την ταχύτερη σύγκλιση αυτού.

Μία ακόμη τεχνική, πέραν της προσθήκης του όρου κανονικοποίησης στη συνάρτηση κόστους, αποτελεί ο τυχαίος μηδενισμός βαρών (Dropout)[48]. Στην περίπτωση αυτή ορισμένα από τα βάρη του δικτύου μηδενίζονται στοχαστικά με σκοπό τον περιορισμό των νευρώνων εκπαίδευσης σε ορισμένα χαρακτηριστικά για τη δημιουργία μίας γενικότερης συνάρτησης υπολογισμού της εξόδου, η οποία θα γενικεύει καλύτερα.



Σχήμα 3.7: *Ανάλυση αναδρομικού νευρωνικού δικτύου*

3.2.3 Επαναλαμβανόμενα Νευρωνικά δίκτυα - Recurrent Neural Networks (RNN)

Τα επαναλαμβανόμενα νευρωνικά δίκτυα είναι μία κατηγορία δικτύων που χρησιμοποιούνται σε ακολουθιακά δεδομένα, όπως τα σήματα φωνής, το κείμενο ή το βίντεο. Βασική διαφορά σε σχέση με τα δίκτυα εμπρόσθιας τροφοδότησης αποτελεί η χρήση της εσωτερικής τους μνήμης για την αποθήκευση πληροφορίας που χρησιμοποιείται κατά την πρόβλεψη της επόμενης κατάστασης. Η έξοδος δηλαδή του εκάστοτε νευρώνα τη χρονική στιγμή t εξαρτάται πλέον τόσο από την είσοδο x_t όσο και από την έξοδο του νευρώνα την $t-1$, την οποία συμβολίζουμε h_{t-1}. Με αυτόν τον τρόπο τα RNN δύνανται να εξάγουν αποτελέσματα με βάση τα συμφραζόμενα, γεγονός πολύ σημαντικό στα δεδομένα ακολουθιακού τύπου. Σημαντικό σε αυτό το σημείο ειναι να αναφερθεί πως είναι δυνατή η ανίχνευση χρονικών εξαρτήσεων ακόμη και σε μη ακολουθιακά δεδομένα.

Όπως σε όλα τα νευρωνικά δίκτυα και στα επαναλαμβανόμενα υπάρχει ένα σύνολο παραμέτρων θ, οι οποίες εκπαιδεύονται με βάση τα δεδομένα. Πλέον, όμως, των βαρών W_i που συνδέουν την είσοδο Q_i με την έξοδο Y, χρησιμοποιείται ένα διάνυσμα βαρών U_i το οποίο επιδρά πάνω στην προηγούμενη έξοδο του νευρώνα, ρυθμίζοντας έτσι τη συμβολή της στον υπολογισμό της επόμενης κατάστασης. Οι εξισώσεις που περιγράφουν τη λειτουργία ενός RNN είναι οι εξής:

$$h_t = f_h(W_h x_t + U_h h_{t-1} + b_h)$$

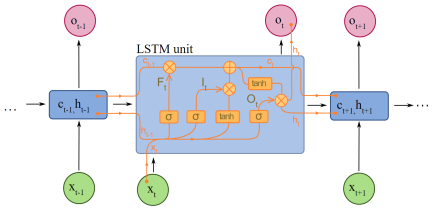
$$y_t = f_y(W_y h_t + b_y)$$

με f_h, f_y τις συναρτήσεις ενεργοποίησης για τα h και y αντίστοιχα, W_h, W_y, U_h τα βάρη του δικτύου, x_t η είσοδος του δικτύου τη χρονική στιγμή t, h_t η κρυφή έξοδος του δικτύου τη χρονική στιγμή t και b_h, b_y οι αντίστοιχες τιμές πόλωσης (bias). Τέλος, όπως και προηγουμένως, μπορουν να δημιουργηθούν βαθιές ιεραρχίες με τη χρήση πολλαπλών στρωμάτων RNN κυττάρων.

Τρείς είναι οι πλέον χρησιμοποιούμενες μορφές αναδρομικού νευρωνικού δικτύου: τα Vanilla RNN, τα οποία αποτελούν την απλούστερη μορφή αναδρομικών νευρωνικών δικτύων και η λειτουργία τους ουσιαστικά περιγράφηκε παραπάνω, τα δίκτυα Μακράς Βραχυπρόθεσμης Μνήμης (LSTM) καθώς και οι φραγμένες επαναλαμβανόμενες μονάδες (GRUs), των οποίων οι διαφορές και η λειτουργία αναλύονται στη συνέχεια. Σημαντική είναι τέλος και η αναφορά στα αμφίδρομα RNNs.

3.2.3.1 Long Short-term Memory (LSTM)

Τα δίκτυα μακράς βραχέας μνήμης, τα οποία προτάθηκαν από τους Hochreiter και Scmidhuber [22] το 1997, αποτελούν μία λύση για το πρόβλημα εξασθένησης κλίση (Vanishing Gradient). Στην περίπτωση των RNNs η εξασθένηση αυτή των παραγώγων καθιστά αδύνατη τη μοντελοποίηση χρονικών εξαρτήσεων μακράς διάρκειας. Το LSTM, σε αντίθεση με τα vanilla RNN, διαθέτει μία πύλη λήθης(forget gate) η οποία δίνει τη δυνατότητα στην μονάδα να αποκόπτει μικρές ή μεγάλες τιμές κλίσης ώστε να αποκλείονται φαινόμενα εξαφάνισης ή έκρηξης κλίσης, ενώ ταυτόχρονα έχει τη δυνατότητα να διατηρεί την εξάρτηση απομακρυσμένων εισόδων της ακολουθίας, επιτρέποντας την επίλυση προβλημάτων με βάση τα συμφραζόμενα.



Σχήμα 3.8: *Κύτταρο (LSTM)*

Αναλυτικά, όπως φαίνεται και στο σχήμα 3.8, το κύτταρο LSTM αποτελείται από τις πύλες εισόδου (input gate,i_i), εξόδου (output gate, o_t) και λήθης (forget gate, f_t). Οι εξισώσεις που ακολουθούν περιγράφουν τη λειτουργία του τη χρονική στιγμή t και είναι μία από τις εκδοχές των χρησιμοποιούμενων LSTM, ίσως και η πλέον χρησιμοποιούμενη. Παρόλα αυτά είναι συνηθες να χρησιμοποιούνται και τροποποιημένες μορφές όπως η πρόταση των Gers and Schmidhuber[49].

$$f_i = \sigma(W_f[h_{t-1}, x_t] + b_f$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i$$

$$\tilde{C_t} = tanh(W_c[h_{t-1}, x_t] + b_c$$

$$C_t = f + t * C_{t-1} + i_t * \tilde{C_t}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o$$

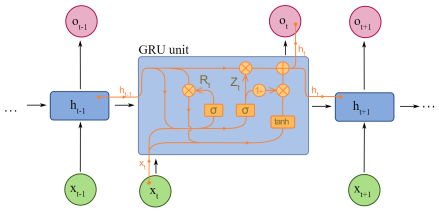
$$h_t = o_t * tanh(C_t)$$

όπου $\sigma()$ η σιγμοειδής συνάρτηση και β οι πολώσεις των πυλών.

Τέλος, σε ότι αφορά τη λειτουργία του κυττάρου συνοψίζονται τα εξής χαρακτηριστικά: στη θύρα εισόδου ελέγχεται η ροή των διανυσμάτων εισόδου $[x_t, h_{t-1}]$, στη θύρα λήθης ελέγχεται η μετάβαση ή όχι του C_t στο επόμενο κύτταρο του δικτύου και η κρυφή κατάσταση ενσωματώνει ή όχι την παρελθοντική πληροφορία του δικτύου στην τρέχουσα είσοδο.

3.2.3.2 Gated Recurrent Units (GRUs)

Όσον αφορά τα GRUs, αποτελούν ένα μηχανισμό φραγής στα επαναλαμβανόμενα νευρωνικά δίκτυα, που προτάθηκε το 2014 [50]. Η δομή του προσομοιάζει εκείνη του LSTM, διαθέτει όμως λιγότερες παραμέτρους αφού λείπει από αυτό μία πύλη εξόδου. Παρά την αλλαγή αυτή η αποδοση του παραμένει ίδια με αυτή του LSTM σε δεδομένες εφαρμογές μοντελοποίησης πολυφωνικής μουσικής, σήματος ομιλίας αλλά και φυσικής γλώσσας[51][52], ενώ έχει αποδειχθεί ότι είναι βέλτιστη σε συγκεκριμένα μικρότερα σε όγκο αλλά και σε συχνότητα δεδομένα[53][54].



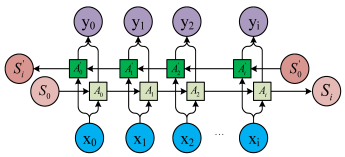
Σχήμα 3.9: *Κύτταρο (GRU)*

3.2.3.3 Αμφίδρομα RNN

Στα αμφίδρομα επαναλαμβανόμενα νευρωνικά δίκτυα δύο κρυφά στρώματα αντίθετων κατευθύνσεων συνδέονται σε ένα κοινό αποτέλεσμα. Με αυτό τον τρόπο, η τρέχουσα κατάσταση ενημερώνεται τόσο με βάση πληροφορίες από το παρελθόν όσο από μελλοντικές καταστάσεις. Η λειτουργία αυτή παρουσιάζεται στο σχήμα 3.10.

Η κρυφή κατάσταση του δικτύου τη χρονική στιγμή t αποτελεί τη συνένωση των δύο κρυφών καταστάσεων που προκύπτουν από τα δύο αντίστροφα συνδεδεμένα RNN.

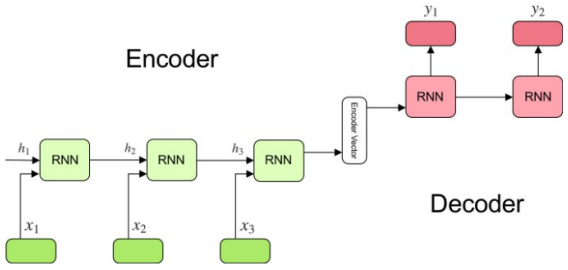
$$h_t = \overrightarrow{h_t} || \overleftarrow{h_t}$$



Σχήμα 3.10: *Αμφίδρομο αναδρομικό νευρωνικό δίκτυο*

3.2.3.4 Μηχανισμός προσοχής Attention Mechanism

Ο μηχανισμός προσοχής αποτελεί αδιαμφισβήτητα ένα από τα ισχυρότερα εργαλεία βαθιάς μάθησης. Εφαρμόζεται στο ανώτερο στρώμα του αναδρομικού νευρωνικού δικτύου και δίνει έμφαση στα μέρη της ακολουθίας που επιδρούν στην ταξινόμηση, αγνοώντας τα λοιπά.



Σχήμα 3.11: *Encoder-decoder sequence to sequence model*

RNN Encoder-Decoder

Το μοντέλο αυτο, γνωστό και ως Seq2Seq, είναι μία διμερής αρχιτεκτονική μηχανικής μάθησης, για την αντιστοίχιση μιας ακολουθίας εισόδων σε μία ακολουθία εξόδων. Αρχικά προτάθηκε σε εφαρμογές μεταγλώττισης, αλλά μπορεί να χρησιμοποιηθεί και σε άλλες εργασίες αντιστοίχισης ακολουθίας οπως η δημιουργία λεζάντας ή η ανάκτηση ερώτησης.

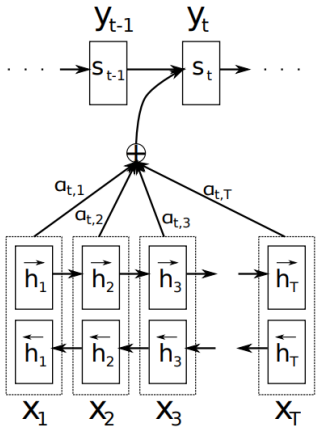
Δύο ανεξάρτητες προτάσεις [50][55] συνδυάζουν δύο αναδρομικά νευρωνικά δίκτυα, αναφερόμενα ως κωδικοποιητής κι αποκωδικοποιητής. Ο κωδικοποιητής διαβάζει την ακολουθία εισόδων, ενώ ο αποκωδικοποιητής παράγει την ακολουθία εξόδου με την κρυφή κατάσταση να συνδέει τα δύο τμήματα. Και τα δύο μέρη αποτελούνται από κύτταρα αναδρομικών νευρωνικών δικτύων ή παραλλαγές όπως LSTMs ή GRUs.

Ευθυγράμμιση και Μεταγλώττιση (Align and Translate)

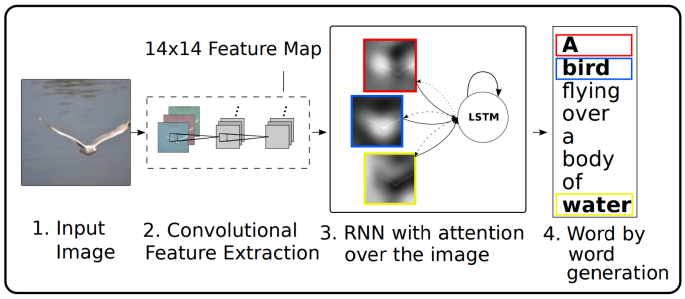
Ένα πιθανό πρόβλημα του μοντέλου encoder-decoder είναι πως δεν είναι δυνατό να αναπαρασταθεί κάθε πληροφορία με διάνυσμα δεδομένου μήκους στην τελευταία κρυφή κατάσταση του κωδικοποιητή h_t. Για την επίλυση αυτού προτάθηκε [56] η χρησιμοποίηση ενός διανύσματος για τα συμφραζόμενα με το οποίο ευθυγραμμίζονται οι είσοδοι προέλευσης και προορισμού (Σχήμα 3.12). Το διάνυσμα αυτό διατηρεί πληροφορίες από όλες τις κρυφές καταστάσεις των κυττάρων κωδικοποίησης. Με αυτόν τον τρόπο, το μοντέλο δύναται να εστιάσει στο επιθυμητό τμήμα της εισόδου και να βελτιώσει την εκπαίδευση πολύπλοκων σχέσεων μεταξύ πηγής και προορισμού.

Visual attention

Το 2015 [57] προτάθηκε μία δομή προσοχής που επεκτείνει το μοντέλο encoder-decoder με σκοπό την ευθυγράμμιση της εισόδου με την έξοδο, στην περίπτωση εικόνας, αντιμετωπίζοντας το πρόβλημα της λεζάντας(image captioning problem). Ένα συνελικτικό στρώμα χρησιμοποιείται για να εξάγει χαρακτηριστικά από την εικόνα εισόδου και επεξεργάζεται αυτά τα χαρακτηριστικά χρησιμοποιώντας αναδρομικό νευρωνικό δίκτυο με μηχανισμό προσοχής. Οι παραγόμενες λέξεις-λεζάντες συνδέονται με συγκεκριμένα μέρη της εικόνας, τονίζοντας τα σχετιζόμενα αντικείμενα που απεικονίζονται (Σχήμα 3.14). Η απόπειρα αυτή αποτελεί μία από τις πρώτες προσπάθειες εφαρμογής μηχανισμού προσοχής σε αντικείμενο πέραν του προβλήματος της μεταγλώττισης.



Σχήμα 3.12: *γραφική απεικόνιση μοντέλου για την δημιουργία εισόδου προορισμού y_t δεδομένης εισόδου $(x_1, x_2, .., x_T)$*



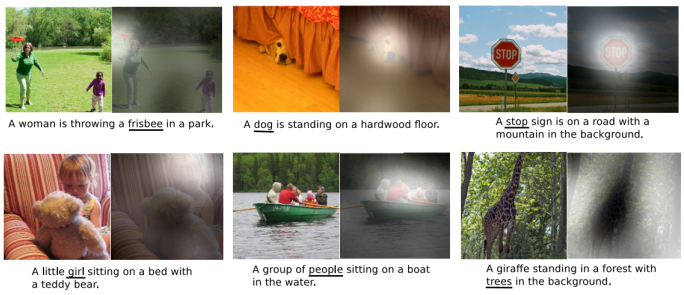
Σχήμα 3.13: *Δίκτυο επίλυσης προβλήματος λεζάντας (image captioning problem)*

Ιεραρχικό δίκτυο προσοχής

Με το ιεραρχικό δίκτυο προσοχής (hierarchical attention network - HAN) [1] αποδείχθηκε ότι αφενός ο μηχανισμός προσοχής μπορεί να εφαρμοστεί σε διάφορα επίπεδα του δικτύου και αφετέρου οτι είναι δυνατή η χρήση του σε προβλήματα ταξινόμησης. Το HAN συνδυάζει δύο δίκτυα κωδικοποίησης, ένα που εστιάζει στις λέξεις και δημιουργεί μία πρόταση ενδιαφέροντος, και ένα που εστιάζει στις προτάσεις οδηγεί στη έξοδο. Ως συνέπεια εξάγονται δύο συμπεράσματα: πρώτον ποιά πρόταση συμβάλει στην κατηγοριοποίηση του αρχείου και δεύτερον ποιές λέξεις σε αυτή ειναι οι εξέχουσες.

Transformer

Η αρχιτεκτονική Transformer(Σχήμα 3.16, η οποία προτάθηκε το 2017 [23], αποτέλεσε ένα από τα μεγαλύτερα επιτεύγματα της δεκαετίας τον τομέα επεξεργασίας φυσικής γλώσσας. Το μοντέλο αυτό δεν βασίζεται, ούτε επεκτείνει αναδρομικά νευρωνικά δίκτυα. Αντίθετα χρησιμοποιεί αποκλειστικά μηχανισμούς προσοχής για να εξάγει εξαρτήσεις μεταξύ εισόδου και εξόδου, ενώ η δομή του δίνει τη δυνατότητα περαιτέρω παραλληλοποίησης. Το στρώμα αυτοπροσοχής πολλαπλών κεφαλών (multi-head self-attention) επιτρέπει στο μοντέλο να εστιάζει συνδυαστικά σε πληροφορίες από διαφορετικές αναπαραστάσεις υποπεριοχών σε διαφορετικές θέσεις, γεγονός μη εφικτό στην περίπτωση απλού μηχανισμού προσοχής. Η πρόταση



Σχήμα 3.14: *Visual Attention*

pork belly = delicious . || scallops? || I don't even like scallops, and these were a-m-a-z-i-n-g . || fun and tasty cocktails. || next time I in Phoenix, I will go back here. || Highly recommend.

Σχήμα 3.15: *Παράδειγμα λειτουργίας από το [1]. Οι προτάσεις ένα και τρία συμβάλουν περισσότερο σημασιολογικά, ενώ στο κομμάτι των λέξεων υπογραμμίζονται ως σημαντικότερες οι delicious και amazing*

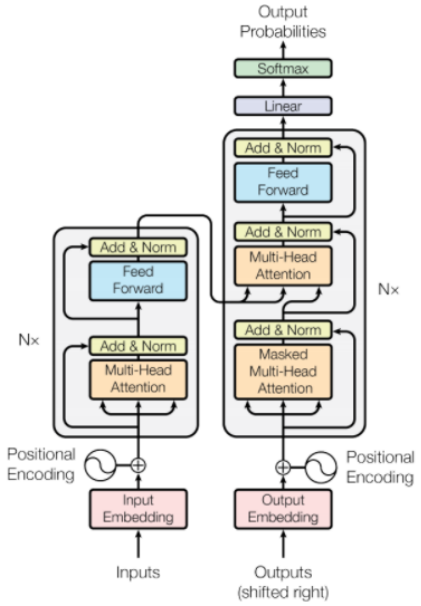
αυτή κινητοποίησε την επιστημονική κοινότητα, οδηγώντας σε ανάπτυξη των μοντέλων αυτοπροσοχής, όπως οι αμφίδρομες αναπαραστάσεις κωδικοποιητών από Transformers (Bidirectional Encoder Representations from Transformers -BERT)[58].

Η λογική αυτή ακολουθήθηκε και σε προβλήματα όρασης (Visual Transformer)[59]. Συνελικτικά δίκτυα αντικαταστάθηκαν με Transformer, αντί ο μηχανισμός να χρησιμοποιηθεί συμπληρωματικά, ξεπερνώντας σε απόδοση κάθε καινοτόμο και μεγάλης κλίμακας συνελικτικό δίκτυο όταν χρησιμοποιηθούν επαρκή δεδομένα εκπαίδευσης.

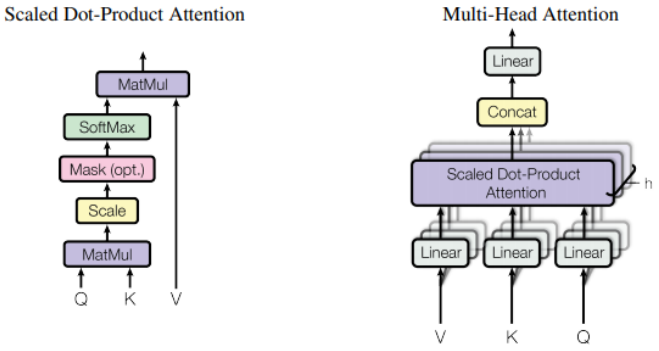
3.2.4 Σύνοψη

Συνοψίζοντας, τα νευρωνικά δίκτυα και η μηχανική μάθηση στο συνολό της αποτελλούν ενάν ραγδαία αναπτυσσόμενο κλάδο. Το εύρος των εφαρμογών σε συνδυασμό με την απόδοση των μοντέλων, η οποία είναι καλή ακόμη και σε εφαρμογές που η αναλυτική μοντελοποίηση είναι ιδιαίτερα πολύπλοκη, ωθούν στην ανάπτυξή της.

Η προσέγγιση και μοντελοποίηση του προβλήματος της προανάκτησης, όπως θα δούμε παρακάτω, θυμίζει τα προβλήματα φυσικής γλώσσας και γι αυτό πλήθος εργασιών (ενδεικτικά [21], [19]), έχουν εστιάσει στη χρήση αναδρομικών νευρωνικών δικτύων. Με βάση τα όσα αναλύθηκαν, ενδιαφέρον φαίνεται να παρουσιάζει η αντίστοιχη προσέγγιση με χρήση μηχανισμών προσοχής, μία αρχική υλοποίηση της οποία παρουσιάζεται στην παρούσα διπλωματική εργασία.



Σχήμα 3.16: *Transformer - Αρχιτεκτονική μοντέλου*



Σχήμα 3.17: *Σύγκριση μηχανισμού προσοχής απλού γινομένου με μηχανισμό προσοχής πολλαπλών κεφαλών*

Κεφάλαιο 4

Σχετική βιβλιογραφία

Στο κεφάλαιο 2 αναλύθηκαν οι παράμετροι που θα πρέπει να ληφθούν υπόψη για την υλοποίηση ενός ολοκληρωμένου μηχανισμού προανάκτησης. Σε αυτό το πλαίσιο παρουσιάστηκαν αρκετές μέθοδοι και σχετικές εργασίες προκειμένου να αναδειχθούν αξιοσημείωτες ιδέες και σχεδιαστικές επιλογές.

Το παρόν κεφάλαιο λειτουργεί ως επέκταση του εισαγωγικού εκείνου κεφαλαίου, εστιάζοντας στις μεθόδους που αφορούν την προανάκτηση με χρήση νευρωνικών δικτύων και ιδίως στις εργασίες που δημιούργησαν τη βάση της διπλωματικής αυτής εργασίας.

4.1 Βασικές ιδέες

Η προσέγγιση του ζητήματος της προανάκτησης με τη χρήση νευρωνικών δικτύων φαίνεται να διέπεται από ορισμένες κοινές ιδέες και σχεδιαστικές επιλογές.

Ενώ εξ ορισμού το prefetching είναι πρόβλημα παλινδρόμησης (αναζητούμε την τιμή της επόμενης διεύθυνσης που θα ζητηθεί σε ένα εύρος συνεχών τιμών), λαμβάνοντας υπόψη αντίστοιχους χώρους, όπως η παραγωγή εικόνας και ήχου, στους οποίους έχει εφαρμοστεί επιτυχώς διακριτοποίηση του συνόλου τιμών, αντιμετωπίζεται ως πρόβλημα ταξινόμησης (αναζητούμε σε ενα σύνολο διακριτών τιμών την τιμή που με μεγαλύτερη πιθανότητα μας δίνει τη λύση). Αφορμή για την αναζήτηση της νέας αυτής οπτικής του προβλήματος αποτέλεσε κατά κύριο λόγο η κακή απόδοση των μοντέλων παλινδρόμησης [21].

Προσεγγίζοντας το από αυτή την οπτική, το prefetchng θυμίζει τα προβλήματα φυσικής γλώσσας, με τη συλλογιστική πορεία να είναι η εξής: ΄με βάση μία ακολουθία λέξεων (αντίστοιχα διευθύνσεων) αναζητούμε την επόμενη λέξη (αντίστοιχα διευθυνση) δεδομένου ενός συνόλου, του λεξικού της γλώσσας (του συνόλου των χρησιμοποιούμενων από την εφαρμογή διευθύνσεων, αντίστοιχα)΄. Με βάση τα παραπάνω, προτάθηκαν αναδρομικά νευρωνικά δίκτυα και συγκεκριμένα κύτταρα LSTM, τα οποία αποτελούν την πλέον διαδεδομένη λύση σε NLP προβλήματα.

Για να είναι υλοποιήσιμη μία λύση σε ένα πρόβλημα ταξινόμησης χωρίς τη δέσμευση υπερβολικών πόρων είναι απαραίτητος ο περιορισμός πέραν του μεγέθους του δικτύου, των κλάσεων που αναπαριστούν τις πιθανές επόμενες διευθύνσεις. Για το σκοπό αυτό, σημαντική είναι η παρατήρηση πως η πλέον ωφέλιμη προανάκτηση είναι αυτή που αφορά διεύθυνση στην ίδια γραμμή της κρυφής μνήμης και δευτερευόντως αν είναι στο πλαίσιο της σελίδας, οπότε μπορεί να καθοριστεί ένα πρώτο όριο. Ένας άλλος τρόπος περιορισμού του πλήθους των διευθύνσεων είναι με βάση τη συχνότητα εμφάνισης. Στην περίπτωση αυτή βεβαια μειώνεται η κάλυψη του μοντέλου, ανάλογα του μεγέθους του 'λεξικού' αυτού που διατηρείται. Μία εναλλακτική λύση, η οποία συμβάλλει και στην αντιμετώπιση της μεταβλητότητας των διευθύνσεων σε κάθε εκτέλεση λόγω της δυναμικής εκχωρησης τους, είναι η χρήση Δέλτα αντί των διευθύνσεων, μέγεθος που παραμένει σταθερό. Ως Δέλτα ορίζεται η διαφορά της τρέχουσας διεύθυνσης από την προηγούμενη.

Οι παρατηρήσεις αυτές φυσικά δεν αφορούν μόνο την προανάκτηση ως πρόβλημα ταξινόμησης, αλλά έχουν χρησιμοποιηθεί ευρέως και σε άλλες μεθόδους. Η φύση όμως του προβλήματος σε αυτή την περίπτωση καθιστά αναγκαία μία τέτοια προσέγγιση.

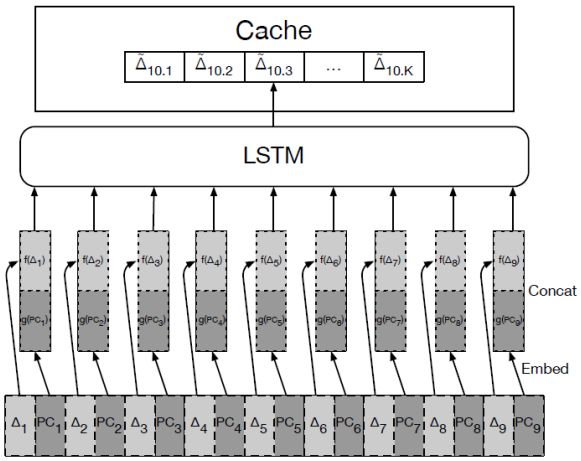
4.2 LSTM embedding and clustering

Η εργασία Learning Memory Access Patterns [21] είναι η πρώτη εκ των δύο, στις οποίες βασίστηκε η δεδομένη εργασία. Σε αυτή παρουσιάζονται αναλυτικά οι παραπάνω ιδέες και εφαρμόζονται για την υλοποίηση δύο νευρωνικών μοντέλων.

Το πρώτο μοντέλο που προτείνεται είναι ένα μεγάλο δίκτυο που αφορά, διαχειρίζεται και αξιοποιεί το συνολο των δεδομένων της εκτελούμενης εφαρμογής. Τα δεδομένα που συλλέγονται αφορούν το σύνολο της εφαρμογής και είναι δείκτες προγράμματος (PCs) και η ακολουθία των διευθύνσεων αστοχίας για καθέναν από αυτούς. Για την είσοδο γίνονται αποδεκτά όλα τα Δελτα αρκεί να έχουν τουλάχιστον δέκα (10) εμφανίσεις στο σύνολο. Αντίθετα για την έξοδο, προκειμένου να περιοριστεί το σύνολο των κλασεων, δημιουργείται ένα λεξικό πιθανών αποτελεσμάτων το οποίο περιλαμβάνει τα 50.000 πιο συχνά εμφανιζόμενα Δέλτα. Τόσο στην είσοδο όσο και στην έξοδο χρησιμοποιείται one-hot κωδικοποίηση. Σε κάθε timestep εισάγονται PC και Δέλτα σε ένα στρώμα ενσωμάτωσης (embedding), συνεννώνονται (concatenate) και διαμορφώνουν έτσι την είσοδο ενός νευρωνικού δικτύου δύο LSTM στρωμάτων (Σχήμα 4.1). Τα δεκα Δέλτα με τη μεγαλύτερη πιθανότητα, όπως προκύπτουν από την έξοδο του δικτύου, χρησιμοποιούνται για την εκτέλεση προανάκτησης.

Στην προσέγγιση αυτή, σημαντικό είναι να σημειωθεί πως το μεγάλο λεξιλόγιο αυξάνει το υπολογιστικό αποτύπωμα αλλά και τον απαιτούμενο αποθηκευτικό χώρο. Απο την άλλη πλευρά, ο περιορισμός του λεξιλόγιου θέτει άνω φράγμα στην ακρίβεια του μηχανισμού, ενώ ένα ακόμη ζήτημα που προκύπτει είναι η μέθοδος διαχείρισης των σπανίως εμφανιζόμενων Δελτα που σε κάθε περίπτωση δεν έχουν ενσωματωθεί στο λεξικό.

Η δεύτερη πρόταση αφορμάται από την τοπικότητα που συχνά παρουσιάζουν τα δεδομενα και



The embedding LSTM model. f and g represent embedding functions.

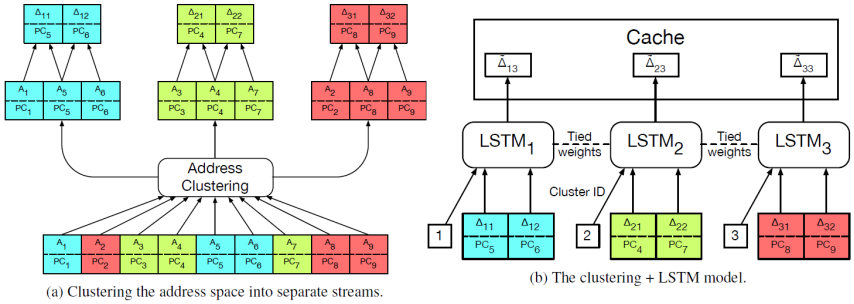
Σχήμα 4.1: *Embedding LSTM model - Learning Memory Access Patterns*

στοχεύει στη σχεδίαση ενός μηχανισμού που θα εστιάζει αποκλειστικά σε ένα πλαίσιο τοπικά, αντί της μοντελοποίησης του συνολικού μοτίβου πρόσβασης της εφαρμογής. Συγκεκριμένα, με τη χρήση του αλγορίθμου k-means, το σύνολο των διευθύνσεων που έχουν καταγραφεί χωρίζονται σε έξι, στην περίπτωση του δεδομένου πειράματος, clusters. Ο υπολογισμός των Δέλτα και των λεξιλογίων, όπως περιγράφηκαν παραπάνω, πραγματοποιείται στο πλαίσιο των clusters, γεγονός που συνεπάγεται σημαντική μείωση του μεγέθους τους. Αυτό δίνει τη δυνατότητα παράλειψης του πίνακα αντιστοιχίσεων (embedding layer), κανονικοποιώντας τις τιμές των Δέλτα και εισάγοντας αυτές τις πραγματικές τιμές στο LSTM. Κάθε cluster μοντελοποιείται μεμονωμένα, προκειμένου όμως να μειωθεί σε μεγαλύτερο βαθμό το μέγεθος του δικτύου, πραγματοποιείται διαμοιρασμός των βαρών σε ένα μοντέλο, ενώ με τη χρήση cluster ids προσδιορίζονται και χρησιμοποιούνται διαφορετικά biases για κάθε περιοχή. Η περιγραφόμενη δομή αναπαρίσταται στο Σχήμα 4.2.

Στα μειονεκτήματα αυτής της προσέγγισης σημειώνονται η μη αξιοποίηση της συνολικής εικόνας του μοντέλου πρόσβαση καθώς και η ανάγκη προεπεξεργασίας των δεδομένων για την ταξινόμηση τους σε clusters.

4.3 Παράμετροι Νευρωνικού Δικτύου

Στην εργασία Understanding Memory Access Patterns for Prefetching, οπού και πάλι ακολουθούνται οι ίδιες βασικές αρχές σχεδίασης, δίνεται έμφαση στην επιλογή των παραμέτρων του νευρωνικού δικτύου. Συγκεκριμένα, ο αριθμός των LSTM κυττάρων και το lookback size, δηλαδή το μέγεθος του ιστορικού που δίνεται στο νευρωνικό δίκτυο μαζί με την τρέχουσα κατάσταση, φαίνεται να έχουν ιδιαίτερη σημασία. Στο Σχήμα 4.3 παρουσιάζεται η μεταβολή της ακρίβειας του μοντέλου με την αύξηση των εν λόγω παραμέτρων.



(a) Clustering the address space into separate streams.

(b) The clustering + LSTM model.

Figure 4. The clustering + LSTM data processing and model.

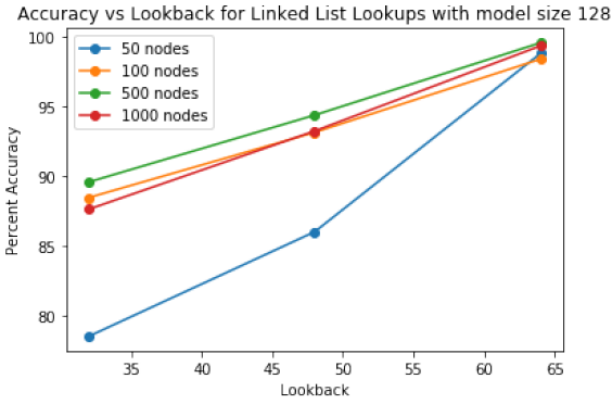
Σχήμα 4.2: *Clustering, data processing and LSTM models - Learning Memory Access Patterns*

Παρατηρούμε ότι η αυξηση του lookback size συμβάλλει σημαντικά στη βελτίωση της ακρίβειας του νευρωνικού μοντέλου ανεξάρτητα της πολυπλοκότητας του δοθέντος προγράμματος (Σχηματα 4.3α΄,4.3β΄), ενώ το μέγεθος του δικτύου, δηλαδή ο αριθμός των LSTM κυττάρων, επηρεάζει την απόδοση εφαρμογών μεγάλου πλήθους παράλληλων ροών, με την ακρίβεια να είναι σταθερά υψηλή στις λοιπές εφαρμογές (Σχήμα 4.3γ΄)

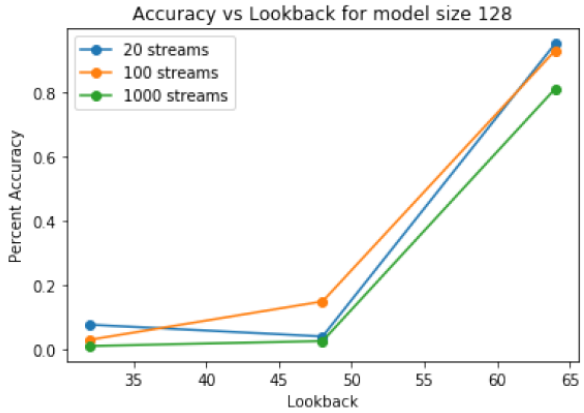
Αξιοσημείωτη είναι επίσης η δημιουργία μίας επιπλέον κλάσης στο σύνολο των αποτελεσμάτων η οποία στοχεύει στην αντιμετώπιση του θορύβου. Αντιστοιχίζεται δηλαδή σε προσβάσεις που δεν εμπίπτουν σε κάποια εκ των λοιπών ροών καθώς και σε Δέλτα μικρού αριθμού εμφανίσεων που δεν έχουν συμπεριληφθεί στο σύνολο τιμών.

Τέλος, σημαντική παραδοχή σε σχέση με την πρώτη εργασία που αναλύθηκε είναι η συλλογή δεδομένων που δεν σχετίζονται μόνο με τις αστοχίες της κρυφής μνήμης. Αντίθετα το ιστορικό περιλαμβάνει όλες τις προσβάσεις της κρυφής μνήμης πρωτου επιπέδου και τις αστοχίες αυτής του τελευταίου επιπέδου. Με αυτόν τον τρόπο επιτυγχάνεται μεγαλύτερη κάλυψη και καλύτερη αποτύπωση του συνολικού μοντέλου πρόσβασης στη μνήμη, με στόχο την εξάλειψη των αστοχιών με το μεγαλύτερο κόστος.

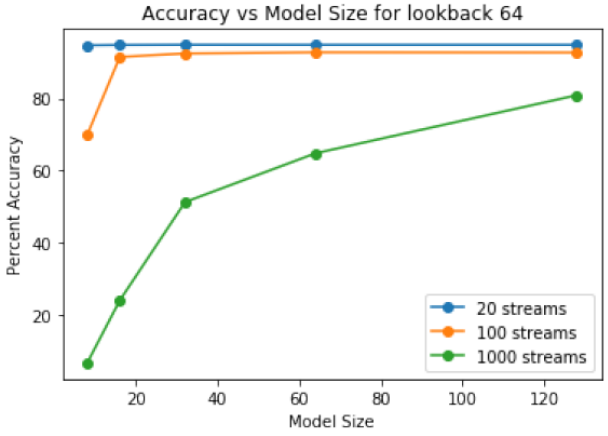
Αναφέρεται πως το μοντέλο που χρησιμοποιείται σε αυτή την περίπτωση αποτελείται από ένα LSTM στρώμα και ένα πυκνό (dense), το οποίο παρέχει την έξοδο. Από το δίκτυο λαμβάνεται μία πρόβλεψη, αυτή με τη μεγαλυτερη πιθανότητα και με βάση αυτή ελέγχεται η ακρίβεια. Το μοντέλο αξιολογείται σε microbenchmarks που δημιουργούνται στο πλαίσιο της εργασίας.



(α′) *Μεταβολή της ακρίβειας ως συνάρτηση του Λοοκβασκ σιζε, σε προγράμματα αναζήτησης Διασυνδεδεμένης λίστας διαφορετικού αριθμού κόμβων*



(β′) *Μεταβολή της ακρίβειας ως συνάρτηση του Λοοκβασκ σιζε, σε προγράμματα διαφορετικού αριθμού παράλληλων ροών*



(γ′) *Μεταβολή της ακρίβειας ως συνάρτηση του μεγέθους του δικτύου, σε προγράμματα διαφορετικού αριθμού παράλληλων ροών*

Σχήμα 4.3: *Διαγράμματα μεταβολής της ακρίβειας του νευρωνικού δικτύου - Understanding Memory Access Patterns for Prefetching*

Κεφάλαιο 5

Ανάλυση και σχεδίαση

Στο κεφάλαιο αυτό παρουσιάζεται η μελέτη που έγινε για την υλοποίηση του μηχανισμού προανάκτησης μέσω της περιγραφής των βασικών ιδεών, των εργαλείων καθώς και των επιλογών σχεδίασης.

5.1 Μέθοδος προανάκτησης

Για την αντιμετώπιση του προβλήματος της προανάκτησης με χρήση μηχανικής μάθησης ακολουθήθηκαν η συλλογιστική αλλά και οι προτάσεις των P. Braun και H. Litz όπως αναλύονται στο άρθρο τους Understanding memory access patterns for prefetching [19].

Αρχικά η προανάκτηση αντιμετωπίζεται ως πρόβλημα πρόβλεψης. Αναζητείται δηλαδή η επόμενη πρόσβαση στη μνήμη, δεδομένων N προηγούμενων προσβάσεων. Συνήθη χαρακτηριστικά που χρησιμοποιούνται για να προσδιορίσουν την κάθε πρόσβαση στη μνήμη είναι η διεύθυνση καθώς και η εντολή που σχετίζεται με αυτή, η οποία προσδιορίζεται με τη χρήση του δείκτη προγράμματος (program counter). Δεδομένης μίας εντολής φόρτωσης, η ακόλουθία των προσβάσεων στη μνήμη που της αντιστοιχούν είναι συνήθως λιγότερο πολύπλοκη και συνεπώς πιο εύκολα προβλέψιμη. Η αντιστοίχιση, λοιπόν, διεύθυνσης-εντολής μπορεί να βοηθήσει στον διαχωρισμό παράλληλων ροών. Μία άλλη προσέγγιση του ίδιου προβλήματος μπορεί να το κατατάξει σε πρόβλημα ταξινόμησης, αν θεωρηθεί πως η επόμενη διεύθυνση που αναζητείται είναι μία από τις πιο συχνά χρησιμοποιούμενες διευθύνσεις. Στην περίπτωση αυτή είναι αναγκαίο ο αριθμός των πιθανών αποτελεσμάτων να είναι περιορισμένος και κατα το δυνατό μικρός. Όσον αφορά, όμως, τον αριθμό των διευθύνσεων της μνήμης, αυτός μπορεί να είναι πολύ μεγάλος. Αντίθετα, η χρήση deltas, δηλαδή διαφορών μεταξύ των διαδοχικών διευθύνσεων μπορεί να περιορίσει σημαντικά το σύνολο τιμών, ιδιαίτερα όταν η εφαρμογή ακολουθεί τυπικά μοντέλα πρόσβασης, όπως η διάσχιση πίνακα. Σε αυτό το χαρακτηριστικό βασίζονται και πολλοί σύγχρονοι prefetchers βήματος και άλλοι.

Ακόμη όμως και σε αυτή την περίπτωση λαμβάνοντας υπόψη τυχαίες προσβάσεις οι οποίες πιθανώς απέχουν σημαντικά από τον κύρια χρησιμοποιούμενο χώρο μνήμης (μη τοπικότητα

των δεδομένων) οι πιθανές τιμές εκτείνονται σε εύρος όσο και οι ίδιες οι διευθύνσεις. Για την αντιμετώπιση αυτού, και λαμβάνοντας υπόψη την υποδομή (χρησιμοποιούμενος προσομοιωτής), στην οποία prefetches εκτός του ορίου της τρέχουσας σελίδας δεν πραγματοποιούνται (περιορισμός που βέβαια μπορεί να αρθεί), θέτονται ως όρια τιμών και αυτα που καθορίζονται από το μέγεθος της σελίδας.

5.2 Υλοποίηση μοντέλου πρόβλεψης

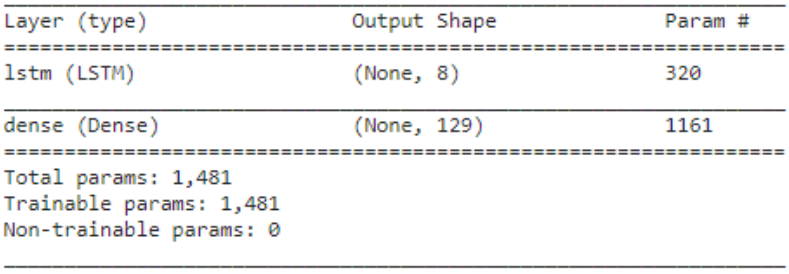
5.2.1 Keras

Το keras είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα γραμμένη σε γλώσσα Python που αφορά τεχνητά νευρωνικά δίκτυα και λειτουργεί ως διεπαφή για τη βιβλιοθήκη Tensorflow. Αναπτύχθηκε με κυριότερο στόχο την ταχύτερη εκτέλεση πειραμάτων, παρέχοντας σημαντικές απλοποιήσεις και δομικά στοιχεία για το σχεδιασμό λύσεων υψηλής ταχύτητας επανάληψης, ενώ ακόμη αξιοσημείωτα χαρακτηριστικά αποτελουν η κλιμακωσιμότητα αλλά και η δυνατότητα χρήσης σε πολλαπλές πλατφόρμες (CPUs, GPUs, TPUs).

5.2.2 Επιλογή και υλοποίηση μοντέλων

Σε συνεχεια της συλλογιστικής πορείας που περιγράφεται παραπάνω η προανάκτηση προσεγγίζεται με τον ίδιο τρόπο όπως ένα πρόβλημα επεξεργασίας φυσικής γλώσσας (Natural Language Processing - NLP). Η συσχέτιση γίνεται ως εξής: σε ένα NLP πρόβλημα, δεδομένης μίας ακολουθίας N λέξεων, πρόβλεψε την επόμενη. Όμοια, σε ένα πρόβλημα προανάκτησης αναζητούμε την επόμενη διεύθυνση δεδομένης μίας ακολουθίας N PCs ή deltas που αντιπροσωπεύουν την απόσταση μεταξύ διαδοχικών PCs. Όπως και στην προανάκτηση, στην επεξεργασία φυσικής γλώσσας υπάρχουν εξίσου πολύπλοκες ακολουθίες και συσχετισμοί μεταξύ των λέξεων σε μία πρόταση, όπως οι ακολουθίες προσβάσεων στη μνήμη σε μία εφαρμογή.

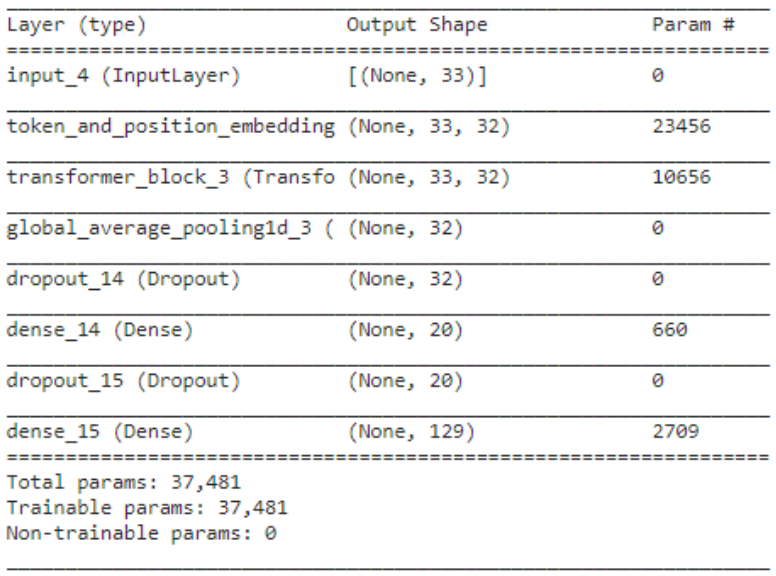
Τα βαθιά νευρωνικά δίκτυα (Deep Neural Networks - DNNs) έχουν συμβάλλει σημαντικά στην πρόοδο διαφόρων πεδίων μεταξύ των οποίων και η επεξεργασία φυσικής γλώσσας. Οι δυνατότητες τους προκύπτουν από τον αλγόριθμο οπισθοδρόμησης, ο οποίος επιτρέπει αποτελεσματικά τη σύγκλιση σε τοπικά βέλτιστα. Η πρόβλεψη στα DNNs προκύπτει με την επεξεργασία ενός διανύσματος εισόδου μέσω διαδοχικών επιπέδων νευρώνων. Το αποτέλεσμα, δηλαδή το διάνυσμα εξόδου που προκύπτει, μεταφράζεται στην επιθυμητή πρόβλεψη. Ένας τύπος DNN ευρέως χρησιμοποιούμενος για NLP προβλήματα είναι τα δίκτυα μακράς βραχέας μνήμης (Long Short-term Memory - LSTM). Είναι σχεδιασμένα για χρονικές ακολουθίες και παρέχουν το πλεονέκτημα μικρότερων μοντέλων λόγω του διαμοιρασμού βαρών. Κάθε στρώμα έχει δεδομένο πλάτος το οποίο καθορίζεται από το διάνυσμά του και μπορεί να μεταβληθεί ενώ ακόμη ορίζεται η παράμετρος lookback η οποία σε αυτό το πλαίσιο καθορίζει το πλήθος των περασμένων προσβάσεων που το μοντέλο λαμβάνει υπόψιν του για την πρόβλεψη της επόμενης.



Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 8)	320
dense (Dense)	(None, 129)	1161

Total params: 1,481
Trainable params: 1,481
Non-trainable params: 0

Σχήμα 5.1: *Layers και αριθμός παραμέτρων για μοντέλοLSTM πλάτους 8 (8 cells)*



Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, 33)]	0
token_and_position_embedding	(None, 33, 32)	23456
transformer_block_3 (Transfo	(None, 33, 32)	10656
global_average_pooling1d_3 (	(None, 32)	0
dropout_14 (Dropout)	(None, 32)	0
dense_14 (Dense)	(None, 20)	660
dropout_15 (Dropout)	(None, 20)	0
dense_15 (Dense)	(None, 129)	2709

Total params: 37,481
Trainable params: 37,481
Non-trainable params: 0

Σχήμα 5.2: *Layers και αριθμός παραμέτρων για μοντέλοtransformer*

Στη δεδομένη εφαρμογή χρησιμοποιήθηκαν, όπως προτείνεται, ένα LSTM layer και ένα Dense layer το οποίο παρέχει και την έξοδο. Από τις παραμέτρους μεταβαλλόμενο μέγεθος θεωρήθηκε το πλάτος του LSTM στρώματος ενώ το lookback size διατηρήθηκε σταθερό και ίσο με 32. Τέλος, τα μοντέλα εκπαιδεύτηκαν με συστοιχίες δεδομένων (batches) των 64 ή των 128 και ο αλγόριθμος βελτιστοποίησης που χρησιμοποιήθηκε είναι ο ADAM.

Σε συνέχεια αυτού, υλοποιήθηκε μηχανισμός προσοχής διπλής κεφαλής για την αντικατάσταση του LSTM μοντέλου, η δομή του οποίου παρουσιάζεται στο Σχήμα 5.2. Αφορμή για το δεδομένο πείραμα υπήρξε η εργασία Attention Is All You Need [23] στην οποία προτείνεται η αντικατάσταση με μηχανισμό προσοχής αντί της χρήσης του συνδυαστικά με κάποιο άλλο αναδρομικό ή συνελικτικό νευρωνικό δίκτυο.

Η συγκεκριμένη δομή είναι αυτή που προτείνεται για text classification με την υλοποίηση ενός Transformer block ως keras layer (Text Classification with Transformer).

5.2.3 Εκπαίδευση

Στην εργασία [19], η οποία όπως αναφέρθηκε λειτούργησε ως βήμα για τη δεδομένη εργασία, χρησιμοποιούνται microbenchmarks για την εκπαίδευση αλλά και την αξιολόγηση των μοντέλων. Συγκεκριμένα δημιουργούνται τέσσερις εφαρμογες: Η πρώτη υλοποιεί τη διάσχιση πίνακα εξετάζοντας ακολουθιακές προσβάσεις στη μνήμη, η δεύτερη τη διάσχιση πίνακα από structs, η τρίτη τη διάσχιση αμετάβλητης λίστας σταθερού μήκους, η οποία δημιουργήθηκε έτσι ώστε η ακολουθία να περιλαμβάνει 5 έως 7 Δέλτα που επαναλαμβάνονται, και η τέταρτη συνδυασμό πολλαπλών μοντέλων πρόσβασης.

Στην παρούσα εργασία, δεν έγινε προσπάθεια αναπαραγωγής των ακολουθιών αυτών όπως περιγράφονται. Για λόγους εξοικονόμησης χρόνου αλλά και για την αξιολόγηση του προτεινόμενου μοντέλου σε πραγματικές εφαρμογές χρησιμοποιήθηκαν έτοιμα ίχνη που παραχωρήθηκαν για το τρίτο πρωτάθλημα προανάκτησης και αφορούν τα μετροπρογράμματα της παραγράφου 5.4.

Για την εκπαίδευση των μοντέλων χρησιμοποιούνται τυχαία, με δειγματοληψία, 2000 εγγραφές ιστορικού, όπως αυτές περιγράφονται παρακάτω. Στην περίπτωση του LSTM τα δεδομένα μετατρέπονται σε one-hot κωδικοποίηση, έτσι ώστε να αποφεύγεται ο αριθμητικός συσχετισμός μεταξύ των τιμών των διαφορών μεταξύ των διευθύνσεων και να αντιμετωπίζεται κάθε τιμή ως ξεχωριστή κατηγορία, δεδομένου ότι αντιμετωπίζουμε την προανάκτηση ως πρόβλημα ταξινόμησης. Δεδομένων των ορίων σελίδας, οι πιθανές κατηγορίες είναι 129. Με βάση αυτό και προκειμένου να επιτύχουμε ομοιομορφία των δεδομένων στην είσοδο του μοντέλου ώστε να διατηρήσουμε όσο το δυνατον απλούστερο το χρησιμοποιούμενο μοντέλο, επιλέγουμε την αναπαράσταση των instruction pointers με χρήση των 6 λιγότερο σημαντικών bits. Ο αριθμός αυτός επαρκεί για την αναπαράσταση των ips, καθώς είτε ο συνολικός αριθμός τους είναι μικρότερος από αυτούς που μπορούμε να αναπαραστήσουμε, είτε θεωρούμε το εύρος για κάθε δεδομένη εφαρμογή είναι σχετικά περιορισμένο. Φυσικά στο πλαίσιο των επιλεγμένων μετροπρογραμμάτων συναντώνται εφαρμογές που δεν εμπίπτουν σε αυτό το πλαίσιο και συνεπώς η απόφαση αυτή θα αξιολογηθεί ως σχεδιαστική επιλογή. Ακόμη, προκειμένου να αντιμετωπίσουμε ανομοιομορφίες των δεδομένων σε ότι αφορά τη συχνότητα εμφάνισης χρησιμοποιούμε υπερδειγματοληψία (oversampling). Από τα δεδομένα αυτά το 80% χρησιμοποιείται για την εκπαίδευση (training data), ενώ το υπόλοιπο 20% χρησιμοποιείται για επαλήθευση (validation data). Τέλος, τα μοντέλα και τα βάρη τους αποθηκεύονται είτε σε μορφή json, είτε στη μορφή SavedModel της βιβλιοθήκης tensorflow.

5.2.4 Αξιολόγηση

Η αξιολόγηση των μοντέλων έγινε, αρχικά, εκτός του προσομοιωτή έτσι ώστε να επιλεγεί ένα εκ των έξι (6) μοντέλων που θα χρησιμοποιηθεί για την τελική αξιολόγηση της μεθόδου προανάκτησης. Τα μοντέλα που εκπαιδεύτηκαν για χρήση του νευρωνικού δικτύου χωρίς την παράλληλη χρήση άλλου είδους προανάκτησης στις κρυφές μνήμες πρώτου και δεύτερου

επιπέδου αξιολογήθηκαν με το πλήρες ιστορικό που είχε ληφθεί με την εκτέλεση πλήρους προσομοίωσης (ως πλήρη προσομοίωση ορίζεται το σύνολο 250 εκατομμυρίων εντολών, εκ των οποίων οι 50 εκατομμύρια πρώτες χρησιμοποιούνται ως warmup). Στα λοιπά μοντέλα, χρησιμοποιείται δείγμα δέκα χιλιάδων εγγραφων που επιλέγονται όπως και τα δεδομένα εκπαίδευσης τυχαία με δειγματοληψία από το σύνολο. Ο περιορισμός αυτός πραγματοποιήθηκε για περιορισμό του απαιτούμενου για την αξιολόγηση χρόνου.

Η διαδικασία πραγματοποιείται για κάθε εγγραφή ξεχωριστά. Από αυτή δίνονται ως είσοδος τα πρωτα 33 στοιχεία (PC και ακολουθία deltas 32 στοιχείων), ενώ το 34 στοιχείο της εγγραφης λειτουργεί ως στοιχείο ελέγχου με το οποίο συγκρίνεται η έξοδος του νευρωνικού δικτύου. Χρησιμοποιούνται μετρητές για το αν η πρώτη πρόβλεψη του νευρωνικού είναι σωστή, όμοια για τη δεύτερη πρόβλεψη καθώς και για το ποσοστό ορθών προβλέψεων αν χρησιμοποιούσαμε προανάκτηση επόμενης γραμμής ή της προηγούμενης διαφοράς για το δεδομένο PC. Το μοντέλο που εμφανίζει την υψηλότερη απόδοση είναι και αυτό που χρησιμοποιείται στον προσομοιωτή.

5.3 ChampSim simulator

Ο ChampSim είναι ένας προσομοιωτής για μελέτη της μικροαρχιτεκτονικής ο οποίος έχει χρησιμοποιηθεί σε διαγωνισμούς που αφορούν τόσο την προανάκτηση όσο και την πρόβλεψη διακλάδωσης και την αντικατάσταση δεδομένων στην κρυφή μνήμη. Η ευρεία χρήση του στα πρωταθλήματα αυτά συνεπάγεται τη συνεχή υποστήριξη σε ότι αφορά προβλήματα, δυσλειτουργίες αλλά και επεκτάσεις των αρχικών δυνατοτήτων, ενώ συμπληρωματικά το σύνολο των συμμετοχών είναι διαθέσιμο προς μελέτη, παρέχοντας, πέρα από τις προτάσεις καθεαυτές πάνω στο προς εξέταση ζητημα, μέσο αξιολόγησης κάθε νέας υλοποίησης.

Όσον αφορά τη λειτουργία του, ο ChampSim δέχεται ως είσοδο ένα ίχνος (trace) από την προς εκτέλεση εφαρμογή και διακριτά αρχεία κώδικα που αφορούν καθένα εκ των προαναφερθέντων θεμάτων. Ως έξοδο λαμβάνουμε μετρήσεις απόδοσης για όλα τα τμήματα της εκάστοτε χρησιμοποιούμενης αρχιτεκτονικής.

Στη συγκεκριμένη διπλωματική ο προσομοιωτής χρησιμοποιείται για τη συλλογή των δεδομένων εκπαίδευσης των μοντέλων(dataset) καθώς και για τον έλεγχο της απόδοσης.

Για τους σκοπους αυτούς δεν χρειάστηκε παρέμβαση στον κύριο κώδικα του προσομοιωτή, παρά μόνο η προσθήκη επιλογών (options) που χρησιμοποιούνται για τον προσδιορισμό είτε των αρχείων εξόδου στα οποία αποθηκεύονται τα συλλεγόμενα δεδομένα για την εκπαίδευση των νευρωνικών μοντέλων (επιλογή -outfile 'outfile_name'), είτε του μοντέλου που πρόκειται να χρησιμοποιηθεί σε περίπτωση εκτέλεσης του προσομοιωτή για την αξιολόγηση της μεθόδου προανάκτησης (επιλογή -ml_model 'model name'). Μία ακόμα επιλογή (-2pred) υλοποιήθηκε έτσι ώστε να χρησιμοποιηθεί για προανάκτηση, πέραν της πρώτης πρόβλεψης του νευρωνικού, και η δεύτερη πιθανοτικά επόμενη διεύθυνση. Σε αυτή την περίπτωση αυξάνεται η απόδοση του μοντέλου σε ότι αφορά τις προβλέψεις σε σύγκριση με την επόμενη διεύθυνση, ενώ αξιολογείται

η συμβολή αυτή στη συνολική απόδοση της εκτελούμενης εφαρμογής βάση του μέσου αριθμού εντολών ανά κύκλο (Instructions Per Cycle - IPC) που αποτελεί και τη βασική μετρική αξιολόγησης στα εν λόγω πειράματα. Τέλος, θα μπορούσαν να προστεθούν αντίστοιχα options για τον προσδιορισμό παραμέτρων όπως ο βαθμός της προανάκτησης, τα οποίο όμως δεν χρησιμοποιήθηκαν προς εξέταση.

Οι κύριες λειτουργίες (καταγραφή ιστορικού και προανάκτηση) υλοποιούνται μέσω δύο αρχείων για την κρυφή μνήμη τελευταίου επιπέδου (LLC). Και στις δύο λειτουργίες δημιουργείται στον προσομοιωτή μία δομή στην οποία αποθηκεύονται program counters - PCs καθένας από τους οποίους συνοδεύεται από την τελευταία διεύθυνση μνήμης που αφορούσε η δεδομένη εντολή φόρτωσης και μία λίστα δύο άκρων 32 θέσεων για την αποθήκευση των διαφορών (deltas). Τα δεδομένα αυτά της δομής για τον εκάστοτε PC θεωρούνται χρήσιμα από τη στιγμή που οι εμφανίσεις του περάσουν ένα δεδομένο όριο threshold που καθορίζεται. Στη δεδομένη περίπτωση το όριο αυτό έχει οριστεί επίσης στον αριθμό 32 έτσι ώστε να μην περιορίζεται περαιτέρω η συλλογή, παρά μόνο όσο απαιτείται για τη συμπλήρωση της εισόδου του νευρωνικού δικτύου. Σε περίπτωση συμπλήρωσης του συνόλου των 32 η ουρά λειτουργεί σαν FIFO και τα νέα δεδομένα είτε καταγράφονται στο αρχείο εξόδου που έχει δοθεί (1η περίπτωση), είτε δίνονται ως είσοδος στο νευρωνικό δίκτυο για την πρόβλεψη. Εναλλακτικά, αν τα δεδομένα δεν επαρκούν για το συγκεκριμένο PC αποθηκεύεται απλά στη δομή το τρέχον delta, ανανεώνεται η διευθυνση μνήμης και στην περίπτωση ελέγχου απόδοσης υλοποιείται προανάκτηση επόμενης γραμμής.

Για την ενσωμάτωση του νευρωνικού δικτύου (υλοποίηση σε γλώσσα Python) στον προσομοιωτή (υλοποίηση σε γλώσσα C++) γίνεται χρήση της βιβλιοθήκης 'Pyhon.h'. Μέσω αυτής αρχικοποιείται ο interpreter της Python και ορίζονται το module και η συνάρτηση αυτού που θα χρησιμοποιηθεί. Σε γλώσσα python έχουν οριστεί δύο συναρτήσεις. Μία για την αρχικοποίηση και φόρτωση του μοντέλου που έχει δοθεί ως είσοδος (pref_init) και η κύρια που χρησιμοποιείται για την πρόβλεψη (pref_operate). Σε συνέχεια των αρχικοποιήσεων και εφόσον αυτά επαρκούν, μετατρέπονται τα δεδομένα (PC και ακολουθία από deltas) και προστίθενται σε μία λίστα, η οποία δίνεται ως όρισμα στη συνάρτηση πρόβλεψης. Σε αυτή γίνονται οι απαραίτητες μετατροπές ετσι ώστε να προκύψει η προκαθορισμένη από την εκπαίδευση μορφή εισόδου και επιστρέφονται δύο διαφορές από την τρέχουσα διεύθυνση για την επόμενη. Από αυτές χρησιμοποιείται η μία ή και οι δύο για προανάκτηση, ανάλογα με την επιλογή που έχουμε θέσει (-2pred) και το αν κάποια εκ των δύο προβλέψεων είναι μηδενική. Ακολουθεί ο υπολογισμός της επόμενης διεύθυνσης και η κλίση της συνάρτησης prefetch_line, η οποία ελέγχει πως η ζητούμενη διεύθυνση βρίσκεται στην ίδια σελίδα και δημιουργεί ένα prefetch packet.

Οι βασικές παράμετροι του προσομοιωτή, όπως χρησιμοποιήθηκαν στην παρούσα εργασία, παρουσιάζονται στον πίνακα 5.1.

Τέλος, ο κώδικας που αφορά τις μετατροπές και τις προσθήκες που πραγματοποιήθηκαν στο πλαίσιο του προσομοιωτή βρίσκονται στο αντίστοιχο repository[60].

Χαρακτηριστικά Champsim				
	Instruction TLB	Data TLB	Second Level TLB	L1 Instruction Cache
SET	16	16	128	64
WAY	4	4	12	8
RQ SIZE	16	16	32	64
WQ SIZE	16	16	32	64
PQ SIZE	0	0	0	8
MSHR SIZE	8	8	16	16
LATENCY	1	1	8	4
	L1 Data Cache	L2 Cache	Last Level Cache	
SET	64	1024	NUM_CPUS2048	
WAY	12	8	16	
RQ SIZE	64	32	NUM_CPUSL2C_MSHR_SIZE	
WQ SIZE	64	32	NUM_CPUSL2C_MSHR_SIZE	
PQ SIZE	8	16	NUM_CPUS32	
MSHR SIZE	16	32	NUM_CPUS64	
LATENCY	5	10	20	

Πίνακας 5.1: *Χαρακτηριστικά μεγέθη των caches στον προσομοιωτή ChampSim*

5.4 Μετροπρογράμματα

Για την αξιολόγηση της απόδοσης των μεθόδων προανάκτησης χρησιμοποιήθηκαν μετροπρογράμματα της σουίτας SPEC CPU®2017[61], η οποία εστιάζει στο τμήμα πραγματικών εφαρμογών υψηλής υπολογιστικής έντασης. Ακολουθεί μία καταγραφή όλων των μετροπρογραμμάτων που χρησιμοποιήθηκαν η οποία περιλαμβάνει και μία σύντομη περιγραφή του καθενός.

600.perlbench_s Αποτελεί μία περιορισμένη εκδοχή της γλώσσας σεναρίων Perl v5.22.1, στην οποία έχει αφαιρεθεί η πλειονότητα των χαρακτηριστικών που αφορούν το λειτουργικό σύστημα.

602.gcc_s Βασίζεται στην έκδοση 4.5.0 του GCC. Παράγει κώδικα για έναν επεξεργαστή IA32 και λειτουργεί ως μεταγλωττιστής με πολλές από τις σημαίες βελτιστοποιήσεων ενεργοποιημένες. Το δεδομένο μετροπρόγραμμα έχει τροποποιηθεί με τέτοιο τρόπο ώστε να ενσωματώνει περισσότερο κώδικα συγκριτικά με τα συστήματα Unix προκειμένου να διαρκεί περισσότερο και να χρησιμοποιεί περισσότερη μνήμη ώστε να προκύπτει ο απαιτούμενος χρόνος εκτέλεσης για το SPEC 2017.

603.bwaves_s Το δεδομένο μετροπρόγραμμα προσομοιώνει αριθμητικά κύματα εκρηξης. Ο αλγόριθμος που εφαρμόζεται επιλύει επαναληπτικά μη συμμετρικά συστήματα γραμμικών εξισώσεων.

605.mcf_s Ανήκει στα μετροπρογράμματα που αφορούν τη συνδυαστική βελτιστοποίηση. Προέρχεται από το πρόγραμμα MCF που χρησιμοποιείται για τον προγραμματισμό οχημάτων δημόσιων συγκοινωνιών. Η εκδοση του μετροπρογράμματος χρησιμοποιεί σχεδόν αποκλειστι-

κά ακέραιες αριθμητικές τιμές.

607.cactuBSSN_s Εντάσσεται στην κατηγορία μετροπρογραμμάτων που έχουν σχέση με τη φυσική και τη θεωρία της σχετικότητας. Βασισμένο στο υπολογιστικό πλαίσιο Cactus, αφορά την επίλυση των εξισώσεων του Αϊνστάιν στο κενό. Στο δεδομένο μετροπρόγραμμα χρησιμοποιείται η διατύπωση BSSN των εξισώσεων με πεπερασμένες διαφορές στο χώρο και μια ρητή μέθοδο ολοκλήρωσης στο χρόνο ενώ βασίζεται στο σύστημα διαχείρισης μνήμης και επικοινωνίας PUGH (Parallel Uniform Grid Hierarchy) για το διαμοιρασμό εργασιών μεταξύ των επεξεργαστών.

619.lbm_s Το συγκεκριμένο μετροπρόγραμμα αφορά στον τομέα της ρευστομηχανικής. Για σκοπούς συγκριτικής αξιολόγησης αλλά και για ευκολη βελτιστοποίηση σε διαφορετικές αρχιτεκτονικές ο κώδικας χρησιμοποιεί εκτενώς μακροεντολές που αποκρύπτουν τις λεπτομέρειες των προσβάσεων στα δεδομένα.

620.omnetpp_s Πρόκειται για μετροπρόγραμμα που αφορά την προσομοίωση διακριτών συμβάντων στη δικτύωση και βασίζεται στο σύστημα προσομοίωσης OMNeT++. Το χρησιμοποιούμενο φορτίο αναφοράς αποτελείται από ένα μεγάλο δίκτυο (10 Gbps) το οποίο περιλαμβάνει έξι μεταγωγείς κορμού, οκτώ μικρά , είκοσι μεσαία και δώδεκα μεγάλα τοπικά δίκτυα.

621.wrf_s Το δεδομένο μετροπρόγραμμα αφορά την πρόγνωση του καιρού.

623.xalancbmk_s Αποτελεί μία τροποποιημένη έκδοση του Xalan-C++, έναν XSLT επεξεργαστή υλοποιημένο σε ένα μεταφέρσιμο υποσύνολο της C++, ο οποίος αφορά στη μετατροπή XML σε HTML. Στην έκδοση του μετροπρογράμματος έχουν πραγματοποιηθεί τροποποιήσεις που περιλαμβάνουν το συνδυασμό κώδικα ώστε να προκύψει ένα αυτόνομο εκτελέσιμο, την αφαίρεση ασυμβατοτήτων του μεταγλωττιστή και τη βελτίωση της απόδοσης, τη μεταβολή της εξόδου ώστε να προβάλλονται και ενδιάμεσες τιμές, την αφαίρεση μεγάλου μέρους μή εκτελούμενου κώδικα που δημιουργούσε αναντιστοιχίες μεταξύ των εκτελέσεων σε διαφορετικές πλατφόρμες, την βελτιστοποίηση στο κομμάτι των δεδομένων και άλλα.

625.x264_s Ανήκει στα μετροπρογράμματα αφορούν τη συμπίεση βίντεο. Η βιβλιοθήκη x264 αλλά και η εφαρμογή συντηρούνται και διανέμονται απο το VideoLan project και το δεδομένο μετροπρόγραμμα που αποτελεί μέρος της σουίτας προέρχεται από ληφθέν στιγμιότυπο.

627.cam4_s Το συγκεκριμένο μετροπρόγραμμα αφορα την μοντελοποίηση της ατμόσφαιρας και αποτελεί μέρος του δημόσιου CESM-1.0.2 ένω η έκδοση του CAM (Community Atmosphere Model είναι η CAM-5.0. Για την κατα το δυνατόν βέλτιστη επικύρωση, το μετροπρόγραμμα διαμορφώθηκε κατάλληλα και χρησιμοποιεί μια μια συνθήκη κάτω του ορίου της θάλασσας, διαμόρφωση που χρησιμοποιείται και απο το Aqua-Planet Experiment Project, ενώ διατηρεί και συμβατότητα προς τα πίσω για τη χρήση του πακέτου cam4, που χρησιμοποιήθηκε στην πλειονότητα των προσομοιώσεων του CMIP5 του ερευνητικού προγραμματος παγκόσμιου κλίματος.

628.pop2_s Όμοια με το cam4_s, το pop2_s ανήκει στην κατηγορία μετροπρογραμμάτων κλιματικής μοντελοποίησης. Το μετροπρόγραμμα βασίζεται στο CESM1.0 το οποίο αποτελείται από τέσσερα διακριτά μοντέλα που προσομοιώνουν ταυτόχρονα την ατμόσφαιρα της γης, τον ωκεανό, την επιφάνεια της γης και τους παγετώνες, ενώ διαθέτη ένα κεντρικό συζευκτικό τμήμα, το CESM που επιτρέπει τη διεξαγωγή ερευνών των κλιματικών καταστάσεων παρελθόντος, παρόντος και μέλλοντος.

631.deepsjeng_s Το δεδομένο μετροπρόγραμμα βασίζεται στο World Computer Speed-Chess Champion του 2008. Ανήκει στην κατηγορία μετροπρογραμμάτων τεχνιτής νοημοσύνης αφου χρησιμοποιεί τον αλγόριθμο Άλφα-Βήτα και αναγνώριση μοτίβων. Η έκδοση αξιολόγησης του μετροπρογράμματος που χρησιμοποιείται έχει μεταγλωττιστεί με περιορισμό μνήμης(SMALL_MEMORY

638.imagick_s Ανήκει στα μετροπρογράμματα διαχείρισης εικόνων. Το ImageMagick είναι μία σουίτα για τη δημιουργία την επεξεργασία, τη σύνθεση ή την μετατροπή εικόνων bitmap. Η έκδοση του μετροπρογράμματος χρησιμοποιεί το τμήμα μετατροπης για την πραγματοποίηση διαφόρων τροποποιήσεων στις εικόνες εισόδου.

641.leela_s Είναι μία μηχανή για το παιχνίδι Go που λειτουργεί με εκτίμηση θέσης βάσει του Monte Carlo, επιλεκτική αναζήτηση σε δέντρα με βάση τα άνω όρια εμπιστοσύνης και αποτίμηση μετακίνησης βαση των αξιολογήσεων Elo.

644.nab_s Ανήκει στα μετροπρογράμματα μοριακής μοντελοποίησης και υλοποιεί εντατικούς υπολογισμούς κινητής υποδιαστολής.

649.fotonik3d_s Στο συγκεκριμένο μετροπρόγραμμα γίνεται υπολογισμός του συντελεστή μετάδοσης ενός φωτονικού κυματοδηγού με χρηση της πεπερασμένης διαφοράς στο πεδίο του χρόνου (FDTD) για τις εξισώσεις Maxwell, ενώ για τον τερματισμό τον υπολογισμών χρησιμοποιείται η μέθοδος μονοαξονικού στρωματος τέλειας προσαρμογής FDTD για διηλεκτρικά υλικά.

654.roms_s Το ROMS είναι ένα σύστημα μοντελοποίησης ωκεανών. Για τη λειτουργία του συνήθως απαιτεί το NETCDF τα δεδομένα του μετροπρογράμματος όμως δεν εκτελούν κλήσεις σε αυτό έτσι ώστε να μην απαιτείται κώδικάς από αυτό παρα μόνο ορισμένες σταθερές.

657.xz_s Ανήκει στα μετροπρογράμματα που αφορούν στη συμπίεση δεδομένων. Βασίζεται στο XZ Utils 5.0.5 του Lasse Collin με ορισμένες διαφορές που περιλαμβάνουν: την ενσωμάτωση του pxz, τον αποκλεισμό ενεργειών εισόδου/εξόδου πέραν της αρχικής, εκτελεί εξ ολοκλήρου τη συμπίεση ή την αποσυμπίεση στην μνήμη, χρησιμοποιεί κατά προτεραιότητα ρουτίνες μεταφέρσιμες σε σχέση με εξειδικευμένες για την εκάστοτε πλατφόρμα.

5.5 Τελική αξιολόγηση μηχανισμών προανάκτησης

Συνολικά τα προτεινόμενα μοντέλα αξιολογήθηκαν με βάση τον συνολικό αριθμό εντολών ανα κύκλο, τη βασική μετρική που παράγει ο προσομοιωτής Champsim, για ένα σύνολο διακοσίων πενήντα εκατομμυρίων εντολών (πλήρης προσομοίωση). Ο συνολικός αυτός αριθμός εντολών ανα κύκλο που προέκυψε για κάθε μετροπρόγραμμα αξιολογείται συγκριτικά τόσο με το αποτέλεσμα εκτέλεσης χωρίς τη χρήση προανάκτησης, που χρησιμοποιείται και ως βάση (baseline), αλλά και με άλλες μεθόδους.

Οι μέθοδοι που επιλέχθηκαν για την αξιολόγηση σχετίζονται με αυτές που χρησιμοποιήθηκαν στην αξιολόγηση στη φάση εκπαίδευσης και επιλογής του τελικού μοντέλου, όπως αυτή περιγράφηκε στην παράγραφο 5.2.4, έτσι ώστε να αξιολογηθεί και η αντιστοιχία μεταξύ των σταδίων αυτών. Αναλυτικά, χρησιμοποιήθηκε προανάκτηση επόμενης γραμμής για κάθε κρυφή μνήμη μεμονωμένα, για κάθε ζεύγος αλλά και για όλες ταυτόχρονα. Ακόμη χρησιμοποιήθηκε ως άνω όριο ή στόχος, ο μηχανισμός bouquet που είχε προταθεί στο τρίτο προτάθλημα προανάκτησης (DPC3), στο οποίο και αξιολογήθηκε ως βέλτιστη μεταξύ των συνολικά δεκατεσσάρων υποβολών. Αποτελείται από τέσσερις prefetchers που αντιστοιχούν σε τέσσερις κατηγορίες ροών: έναν σταθερού βήματος, έναν μεταβλητού βήματος, έναν καθολικής ροής και έναν επόμενης γραμμής. Στους τρεις πρώτους χρησιμοποιείται ο δείκτης προγράμματος για το διαχωρισμό ροών, ενώ μία ακόμη σημαντική σχεδιαστική επιλογή είναι η διατήρηση ενός ενιαίου πίνακα για όλες τις κατηγορίες ροών. Η προανάκτηση εκτελείται στις δύο πρώτες caches.

Κεφάλαιο 6

Αξιολόγηση

Αντικείμενο αυτού του κεφαλαίου είναι η παρουσίαση και ο σχολιασμός των αποτελεσμάτων για τη σύγκριση των προς αξιολόγηση μεθόδων προανάκτησης τόσο μεταξύ τους όσο και με άλλες μεθόδους και προτάσεις. Τα αποτελέσματα αυτά αφορούν τα μετροπρογράμματα που παρουσιάστηκαν στο υποκεφάλαιο 5.4.

6.1 Στατιστικά Μετροπρογραμμάτων

Δεδομένων ορισμένων σχεδιαστικών επιλογών αλλά και των μεθόδων προανάκτησης που επελέγησαν για την συγκριτική αξιολόγηση των μεθόθων που παρουσιάζονται στην παρούσα εργασία, είναι αναγκαίο στο σημείο αυτό να συμπεριληφθούν χαρακτηριστικά των μετροπρογραμμάτων, τα οποία πρόκειται να επηρεάσουν την απόδοση.

6.1.1 Αναπαράσταση δεικτών προγράμματος

Όπως έχει αναφερθεί, για λόγους απλούστερης κωδικοποίησης των δεδομένων εισόδου στα νευρωνικά μοντέλα γίνεται χρήση των έξι λιγότερο σημαντικών bit για την αναπαράσταση των δεικτών προγράμματος. Συνεπώς, ενέχει ο κίνδυνος συγκερασμού ροών διαφορετικών PCs σε μία εγγραφή ιστορικού, όπως αυτές εισάγονται στο δίκτυο. Στον παρακάτω πίνακα (Πίνακας 6.1) παρουσιάζονται τα πλήθη των μοναδικών PCs και των αντίστοιχων αναπαραστάσεων τους στα δεδομένα εκπαίδευσης των μετροπρογραμμάτων.

Από τα στοιχεία ξεχωρίζουν δύο περιπτώσεις, των μετροπρογραμμάτων 607 και 627, στις οποίες ο αριθμός των δεικτών προγράμματος είναι σημαντικά μεγαλύτερος από το μέγιστο πλήθος αναπαραστάσεων που μπορούμε να έχουμε δεδομένης της κωδικοποίησης και συνεπώς αναμένεται αντίστοιχα χαμηλότερη απόδοση συγκριτικά με άλλα μετροπρογράμματα. Σημαντικό είναι επίσης να σημειωθεί πως ακόμη και σε περιπτώσεις που το μέγιστο πλήθος των αναπαραστάσεων επαρκεί για την κάλυψη όλων των PCs, ο τρόπος με τον οποίο έχει πραγματοποιηθεί η κωδικοποίηση δεν συνεπάγεται αυτή την πλήρη αντιστοίχιση, όπως για παράδειγμα στο 649

benchmark	#PCs	#PC representations
600.perlbench	60	60
602.gcc	14	14
603.bwaves	10	10
605.mcf	26	26
607.cactuBSSN	451	127
619.lbm	10	10
620.omnetpp	128	109
621.wrf	15	15
623.xalancbmk	77	71
625.x264	66	63
627.cam4	621	124
628.pop2		
631.deepsjeng	7	7
638.imagick		
641.leela	54	52
644.nab	19	
649.fotonik3d	8	7
654.roms	2	
657.xz	44	42

Πίνακας 6.1: *Αντιστοιχία πλήθους μοναδικών δεικτών προγράμματος και της αντίστοιχής αναπαράστασης τους*

οπού οκτώ διαφορετικά PCs αντιστοιχίζονται σε επτά αναπαραστάσεις. Είναι συνεπώς φανερό πως η επιλογή της δεδομένης κωδικοποίησης η οποία εξυπηρετεί λόγους απλότητας και ταχύτητας σε ότι αφορά τις μετατροπές αλλά και τη δομή των νευρωνικών δικτύων, δύναται να περιορίσει τις δυνατότητες των μοντέλων.

6.1.2 Αντίκτυπος μεγέθους σελίδας

Μία άλλη παράμετρος που είναι σημαντική για την αξιολόγηση των αποτελεσμάτων είναι το πλήθος των εγγραφών που είναι εκτός σελίδας (4KB), καθώς η πρόβλεψη τους σωστά ως καταστάσεις στις οποίες δεν εκτελείται προανάκτηση φυσικά δεν συμβάλλει στην συνολική απόδοση, την αύξηση δηλαδή των εντολών ανά κύκλο που χρησιμοποιείται ως κύρια μετρική, ενώ προσμετρώνται θετικά στην ακρίβεια των μοντέλων. Στον πίνακα που ακολουθεί (Πίνακας 6.2) παρουσιάζονται για κάθε μετροπρόγραμμα τα βήματα (deltas) που εμφανίζονται με μεγαλύτερη συχνότητα, καθώς και το ποσοστό των προβλέψεων για μη εκτέλεση προανάκτησης (delta = 0) που αντιστοιχεί σε διευθύνσεις εκτός σελίδας και θα μπορούσαν να πραγματοποιηθούν με άρση αυτού του περιορισμού.

Όπως είναι φανερό, μεγάλο μέρος των μετροπρογραμμάτων διαθέτουν μηδενικές εγγραφές οι οποίες στην πλειοψηφία τους αντιστοιχούν σε επόμενες προσβάσεις εκτός του ορίου της τρέχουσας σελίδας. Το γεγονός αυτό συνεπάγεται σταθερή και όχι βελτιούμενη απόδοση, ενώ χαρακτηριστικό δείγμα είναι αυτό του μετροπρογράμματος 638 στο οποίο δεν προκύπτουν διαφορές εντός σελίδας με αποτέλεσμα να μην υπάρχουν δυνατότητες προανάκτησης στο πλαίσιο εργασίας μας.

	most_freq	%	2nd_most_freq	%	% of zeros representing out of page deltas
600.perlbench	0	51,23	3	5,86	100
602.gcc	2	53,98	1	20,28	99,98
603.bwaves	5	86,09	0	7,80	100
605.mcf	0	57,14	1	30,34	99,89
607.cactuBSSN	1	97,38	0	1,66	99,94
619.lbm	5	39,96	2	20,08	99,94
620.omnetpp	0	82,78	3	2,31	99,82
621.wrf	1	76,37	0	11,57	100
623.xalancbmk	0	80,99	-1	4,08	95,84
625.x264	0	29,62	1	2,08	99,99
627.cam4	-25	21,49	1	21,29	99,87
628.pop2	1	78,27	0	16,91	47,87
631.deepsjeng	0	63,16	1	5,31	99,99
638.imagick	0	100			95,42
641.leela	1	72,72	0	16,94	99,99
644.nab	1	90,95	0	5,13	100
649.fotonik3d	1	54,35	2	17,15	90,40
654.roms	1	97,92	0	2,08	100
657.xz	0	87,27	1	4,26	99,95

Πίνακας 6.2: *Πίνακας αναπαράστασης των 2 Δέλτα με τις περισσότερες εμφανίσεις στο ιστορικό για κάθε μετροπρόγρμμα και ποσοστό των μηδενικών που αφορά προσβάσεις εκτός σελίδας.*

6.2 Σύγκριση μοντέλων πρόβλεψης

Στην ενότητα αυτή γίνεται σύγκριση των προβλέψεων των νευρωνικών δικτύων που μελετήθηκαν ως προς την ακρίβεια τους. Ως σημεία αναφοράς χρησιμοποιήθηκαν και αντίστοιχα αποτελέσματα δύο βασικών μεθόδων προανάκτησης, αυτών της επόμενης γραμμής και της πρόβλεψης με βάση το τελευταίο εμφανιζόμενο Δέλτα. Όπως και στα νευρωνικά δίκτυα, στην περιπτωση του προηγούμενου δέλτα το ιστορικό διατηρείται ανά instruction pointer ώστε να υπάρχει αντιστοιχία μεταξύ των μεθόδων. Για την εκπαίδευση και την αξιολόγηση των μοντέλων ακολουθήθηκαν οι πρακτικές όπως αυτές περιγράφηκαν στις παραγράφους 5.2.3 και 5.2.4.

6.2.1 Μοντέλα χωρίς παράλληλη χρήση άλλης μεθόδου προανάκτησης

Η ενότητα αυτή αφορά τα πρώτα μοντέλα που εκπαιδεύτηκαν και αξιολογήθηκαν με χρήση δεδομένων που συλλέχθηκαν με εκτέλεση των μετροπρογραμμάτων και αντικατοπτρίζουν τις αστοχίες στην κρυφή μνήμη τρίτου επιπέδου χωρίς την ύπαρξη κάποιου άλλου μηχανισμού προανάκτησης στις L1 και L2 caches.

Από τους πίνακες 6.3 και 6.4 αλλά και από το σχήμα 6.1 είναι φανερό πως η χρήση νευρωνικού δικτύου για την πρόβλεψη της επόμενης διαφοράς υπερέχει των δύο απλούστερων μεθόδων στην πλειονότητα των μετροπρογραμμάτων. Σαφώς υπάρχουν εξαιρέσεις, όπως η περίπτωση του 607 για το οποίο, και με βάση τον πίνακα 6.2, αναμένουμε τόσο η πρόβλεψη για προ-

	next_line	previous_delta	LSTM_cells8	LSTM_cells16	LSTM_cells32	LSTM_cells64	LSTM_cells128	Transformer
600.perlbench	0,0375	0,4559	0,6071	0,6125	0,6684	0,7058	0,7124	0,6706
602.gcc	0,2002	0,2160	0,8554	0,8751	0,8723	0,9024	0,9084	0,8774
603.bwaves	0	0,7507	0,8545	0,8545	0,8545	0,8545	0,8545	0,8440
605.mcf	0,2914	0,8497	0,9444	0,9431	0,9435	0,9387	0,9393	0,9472
607.cactuBSSN	0,9763	0,9562	0,9763	0,9763	0,9763	0,9763	0,9763	0,8378
619.lbm	0	0,2987	0,8045	0,8055	0,8109	0,8125	0,8115	0,7802
620.omnetpp	0,0135	0,7475	0,8328	0,8450	0,8523	0,8470	0,8520	0,8249
621.wrf	0,6829	0,7847	0,8350	0,9353	0,9460	0,9455	0,9615	0,9029
623.xalancbmk	0,0087	0,8055	0,8453	0,8534	0,8537	0,8489	0,8545	0,8346
625.x264	0,1985	0,6323	0,7563	0,7537	0,7667	0,7722	0,7693	0,8012
627.cam4	0,2045	0,5360	0,7053	0,7059	0,7038	0,7101	0,7162	0,6824
628.pop2	0,7924	0,8603	0,9079	0,9100	0,8922	0,9123	0,9140	0,8691
631.deepsjeng	0,01199	0,3939	0,6309	0,6326	0,6323	0,63032	0,6263	0,3678
638.imagick								
641.leela	0,7297	0,9007	0,9338	0,9296	0,9378	0,9379	0,9313	0,9403
644.nab	0,9072	0,9037	0,9072	0,9382	0,9396	0,9397	0,9395	0,9359
649.fotonik3d	0,5440	0,3397	0,5529	0,5444	0,5672	0,5994	0,6270	0,63111
654.roms	0,9793	0,9587	0,9793	0,9793	0,9793	0,9793	0,9793	0,9895
657.xz	0,0450	0,8560	0,9055	0,9056	0,9054	0,9038	0,9043	0,7455

Πίνακας 6.3: *Ακρίβεια μοντέλων μιας πρόβλεψης*

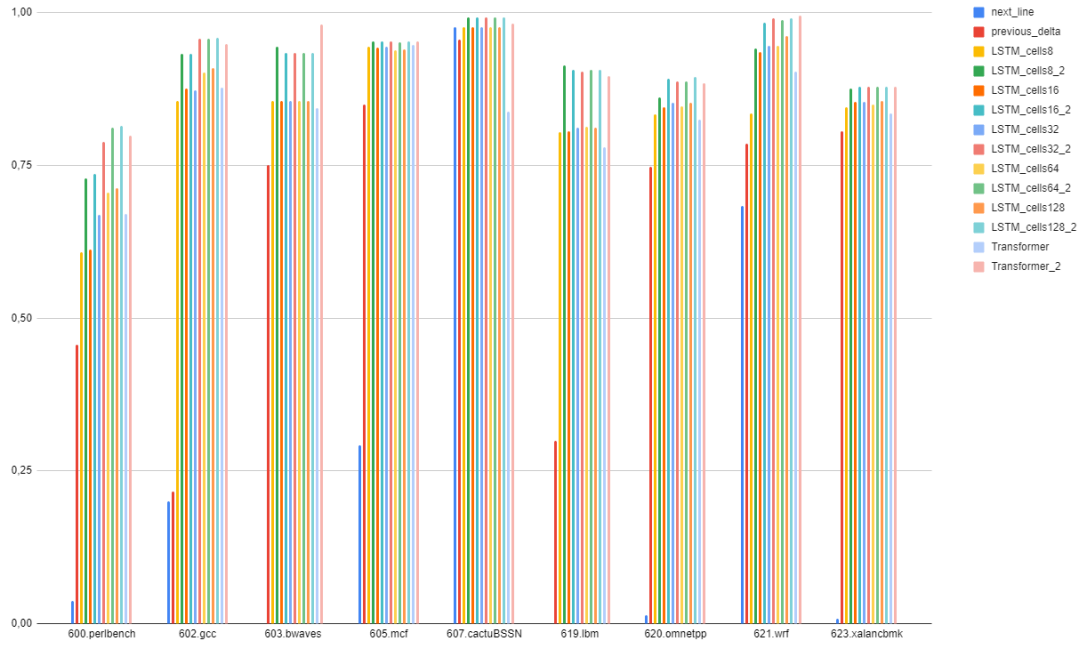
	LSTM_cells8	LSTM_cells16	LSTM_cells32	LSTM_cells64	LSTM_cells128	Transformer
600.perlbench	0,7289	0,7361	0,7882	0,8118	0,8150	0,7988
602.gcc	0,9321	0,9320	0,9568	0,9569	0,9590	0,9489
603.bwaves	0,94438	0,9338	0,9338	0,9338	0,9338	0,9797
605.mcf	0,9524	0,9530	0,9525	0,9512	0,9529	0,9532
607.cactuBSSN	0,9926	0,9924	0,9925	0,9925	0,9926	0,9813
619.lbm	0,9133	0,9068	0,9038	0,9067	0,9062	0,8959
620.omnetpp	0,8603	0,8919	0,8873	0,8870	0,8946	0,8841
621.wrf	0,9414	0,9830	0,9902	0,9872	0,9913	0,9943
623.xalancbmk	0,8752	0,8779	0,8781	0,8785	0,8786	0,8778
625.x264	0,9386	0,9378	0,9433	0,9482	0,9466	0,9485
627.cam4	0,7719	0,7735	0,7793	0,7857	0,7903	0,7624
628.pop2	0,9240	0,9253	0,9244	0,9292	0,9532	0,9487
631.deepsjeng	0,6828	0,6794	0,6797	0,6779	0,6755	0,5820
638.imagick						
641.leela	0,9704	0,9656	0,9719	0,9724	0,9680	0,9747
644.nab	0,9596	0,9596	0,9596	0,9592	0,9587	0,9547
649.fotonik3d	0,7651	0,7289	0,7855	0,8083	0,8272	0,8626
654.roms	1	1	1	1	1	1
657.xz	0,9176	0,9184	0,9182	0,9182	0,9178	0,9727

Πίνακας 6.4: *Ακρίβεια μοντέλων διπλής πρόβλεψης*

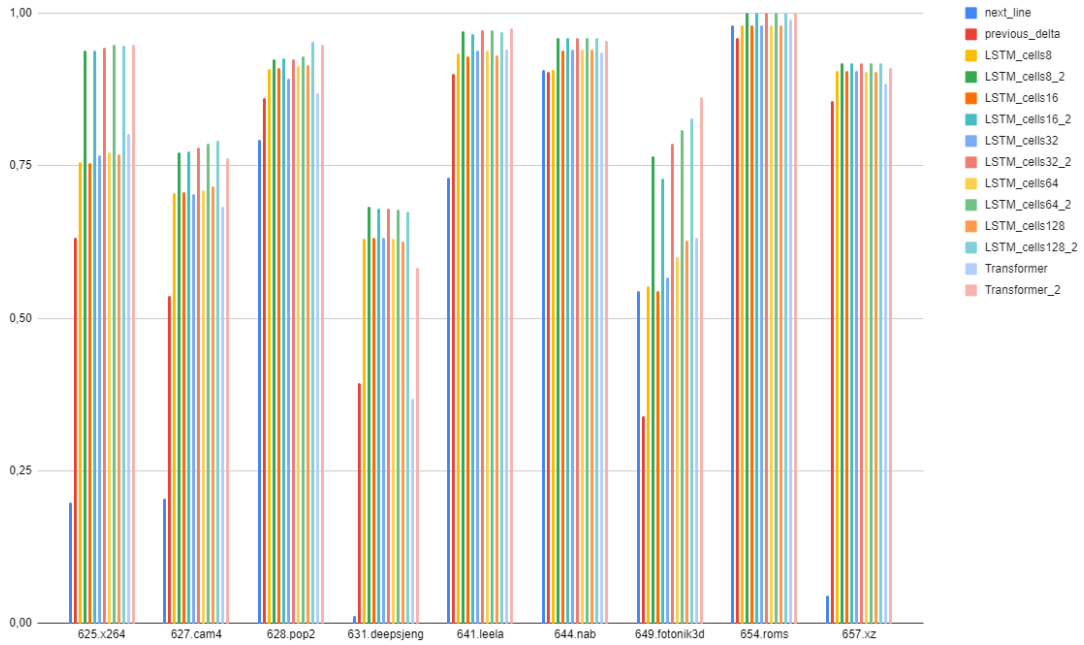
ανάκτηση επόμενης γραμμής όσο και του προηγούμενου Δέλτα θα παρέχουν τα επιθυμητά αποτελέσματα λόγω της συντριπτικής πλειοψηφίας των μοναδιαίων βημάτων στο συνολικό πλήθος. Όμοια και για τα μετροπρογράμματα 644 και 654.

Ιδιαίτερα σημαντική φαίνεται ακόμη η συμβολή της διπλής πρόβλεψης των νευρωνικών δικτύων, με βελτίωση της απόδοσης έως και κατά 40% (μετροπρόγραμμα 649). Το γεγονός αυτό υποδεικνύει τον λόγο για τον οποίο χρησιμοποιείται για την τελική αξιολόγηση των μοντέλων η διπλή πρόβλεψη, ενώ παράλληλα υποδηλώνει εμμέσως την αδυναμία πλήρους πρόβλεψης του μοντέλου πρόσβασης από το νευρωνικό δίκτυο σε μετροπρογράμματα με πιο πολύπλοκα access patterns, με τις παραμέτρους και τις επιλογές σχεδίασης που έχουν πραγματοποιηθεί.

Σε ότι αφορά τα κύτταρα LSTM, και με βάση το σχήμα 6.2, παρατηρούμε ότι κατα κύριο λόγο είτε η απόδοση παραμένει σταθερή, είτε βελτιώνεται με την αύξησή τους. Στις περιπτώσεις σταθερής καμπύλης με συνολικά υψηλή ακρίβεια αντιστοιχούν μετροπρογράμματα με σχετικά απλά μοντέλα πρόσβασης στη μνήμη, ενώ καμπύλες έντονα ανοδικές με την αύξηση των LSTM κυττάρων υποδεικνύουν πιο πολύπλοκες ακολουθίες που απαιτούν αντίστοιχα και

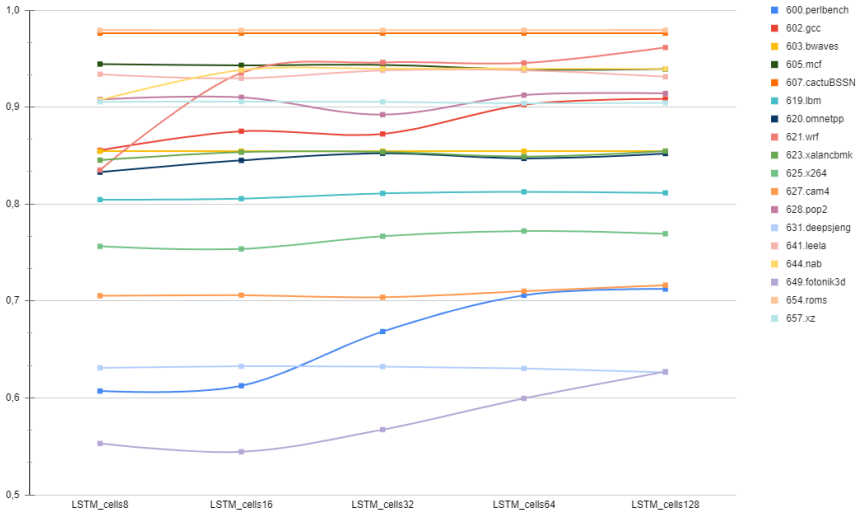


(α′)

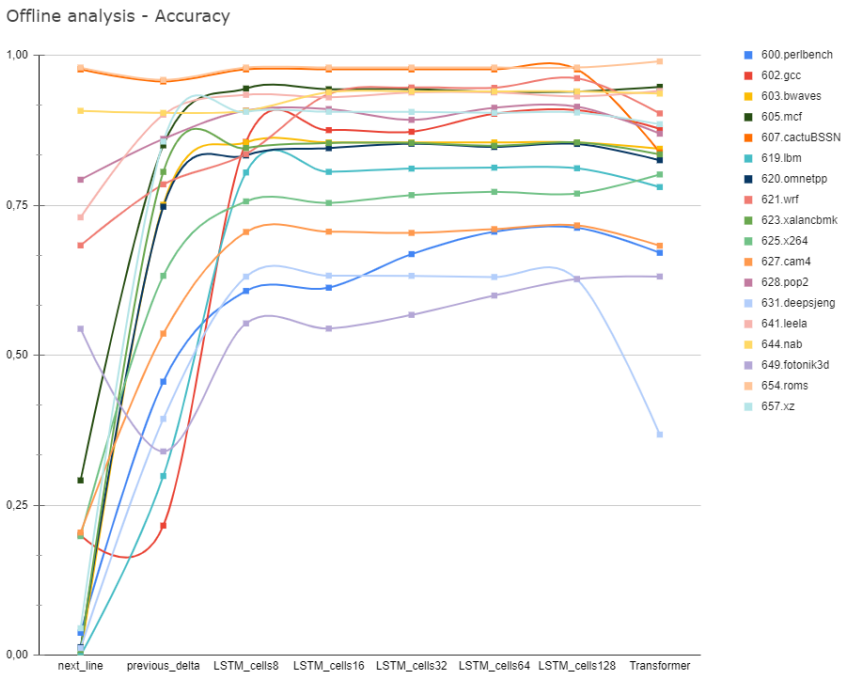


(β′)

Σχήμα 6.1: *Ποσοστό επιτυχημένων προβλέψεων για καθένα από τα σχήματα προανάκτησης σε δεδομένα που αφορούν την εκτέλεση των μετροπρογραμμάτων χωρίς την παράλληλη χρήση άλλου σχήματος προανάκτησης*



Σχήμα 6.2: *Γράφημα μεταβολής ακρίβειας σε σχέση με τον αριθμό των LSTM κυττάρων*



Σχήμα 6.3: *Πλήρες διάγραμμα απόδοσης όλων των συγκρινόμενων μεθόδων - offline analysis*

περισσότερες παραμέτρους για την σωστή προσέγγιση τους. Η παρατήρηση αυτή συμπίπτει με αυτές της εργασίας Understanding Memory Access Patterns for Prefetching[19] από τα διαγράμματα της οποίας, στα οποία γίνεται και αναλυτικότερη αναφορά στο κεφάλαιο 4, ο αριθμός των κυττάρων έχει σημασία σε εφαρμογές μεγάλου αριθμού παράλληλων ροών. Σταθερά χαμηλή ακρίβεια, όπως στην περίπτωση του 631, υποδηλώνει αδυναμία ορθής προσαρμογης του μοντέλου. Τέλος, μικρές διακυμάνσεις προς τα κάτω οι οποίες δημιουργούν ελάχιστα στη γραφική παράσταση μπορεί να οφείλονται και στο σημείο διακοπής της εκπαίδευσης του δικτύου.

	next_line	previous_delta	LSTM_cells8	LSTM_cells16	LSTM_cells32	LSTM_cells64	LSTM_cells128	Transformer
600.perlbench	0,5710	0,3797	0,7854	0,8014	0,7966	0,7793	0,7677	0,7482
602.gcc	0,6323	0,5336	0,6593	0,6519	0,6901	0,6601	0,6553	0,6510
603.bwaves	0,2699	0,3676	0,7853	0,7906	0,7998	0,7766	0,7908	0,8162
605.mcf	0,5684	0,5141	0,5762	0,7620	0,7981	0,7862	0,8028	0,7728
607.cactuBSSN	0,9506	0,9243	0,9472	0,9536	0,9518	0,9537	0,9543	0,9403
619.lbm	0,7555	0,6862	0,7842	0,8035	0,8041	0,7885	0,7914	0,7906
620.omnetpp	0,5145	0,3793	0,7213	0,7395	0,7638	0,7108	0,7513	0,6979
621.wrf	0,8329	0,6832	0,8425	0,81857	0,845	0,87	0,9	0,9014
623.xalancbmk	0,3185	0,4982	0,6473	0,6483	0,6485	0,6393	0,6329	0,5577
625.x264	0,3019	0,5515	0,6549	0,6961	0,6983	0,6866	0,6883	0,6658
627.cam4	0,3499	0,4397	0,5986	0,6171	0,6114	0,6004	0,5978	0,5313
628.pop2	0,8048	0,7788	0,8920	0,8779	0,8834	0,8808	0,8822	0,8658
631.deepsjeng	0,5607	0,3315	0,7142	0,7015	0,6660	0,6460	0,6284	0,5611
638.imagick	0,1564	0,9590	0,9680	0,9682	0,9685	0,9629	0,8843	0,9610
641.leela	0,7119	0,7195	0,8364	0,8294	0,8343	0,8304	0,8105	0,7961
644.nab	0,9243	0,9034	0,9095	0,9242	0,9267	0,9281	0,9275	0,9238
649.fotonik3d	0,7290	0,5686	0,7218	0,7252	0,7230	0,7253	0,7195	0,6646
654.roms	0,9797	0,9594	0,9797	0,9797	0,9797	0,9797	0,9797	0,6351
657.xz	0,5567	0,3629	0,7822	0,7355	0,7848	0,7661	0,7704	0,7455

Πίνακας 6.5: *Ακρίβεια μοντέλων πρόβλεψης*

	LSTM_cells8_2	LSTM_cells16_2	LSTM_cells32_2	LSTM_cells64_2	LSTM_cells128_2	Transformer_2
600.perlbench	0,9481	0,9513	0,9495	0,9490	0,9439	0,9230
602.gcc	0,8205	0,8277	0,8440	0,8280	0,8160	0,7942
603.bwaves	0,8788	0,8974	0,8811	0,8765	0,8848	0,9144
605.mcf	0,9521	0,9615	0,9747	0,9729	0,9729	0,9701
607.cactuBSSN	0,9746	0,9805	0,9803	0,9808	0,9806	0,9727
619.lbm	0,9079	0,8992	0,9206	0,8969	0,9055	0,9094
620.omnetpp	0,9502	0,9497	0,9503	0,9480	0,9522	0,9384
621.wrf	0,9436	0,9332	0,9471	0,9614	0,9725	0,9781
623.xalancbmk	0,9387	0,9399	0,9379	0,9341	0,9352	0,8850
625.x264	0,8646	0,9022	0,8984	0,8882	0,8703	0,8598
627.cam4	0,8005	0,7974	0,7986	0,7907	0,7936	0,7040
628.pop2	0,9572	0,9587	0,9595	0,9576	0,9588	0,9486
631.deepsjeng	0,8586	0,8398	0,8375	0,8290	0,8245	0,7487
638.imagick	0,9915	0,9928	0,9967	0,9967	0,9964	0,9973
641.leela	0,9697	0,9707	0,9703	0,9710	0,9668	0,9574
644.nab	0,9269	0,9615	0,9686	0,9684	0,9698	0,9686
649.fotonik3d	0,8137	0,8338	0,8522	0,8605	0,8613	0,8225
654.roms	1	1	1	1	1	1
657.xz	0,9789	0,9795	0,9782	0,9765	0,9750	0,9727

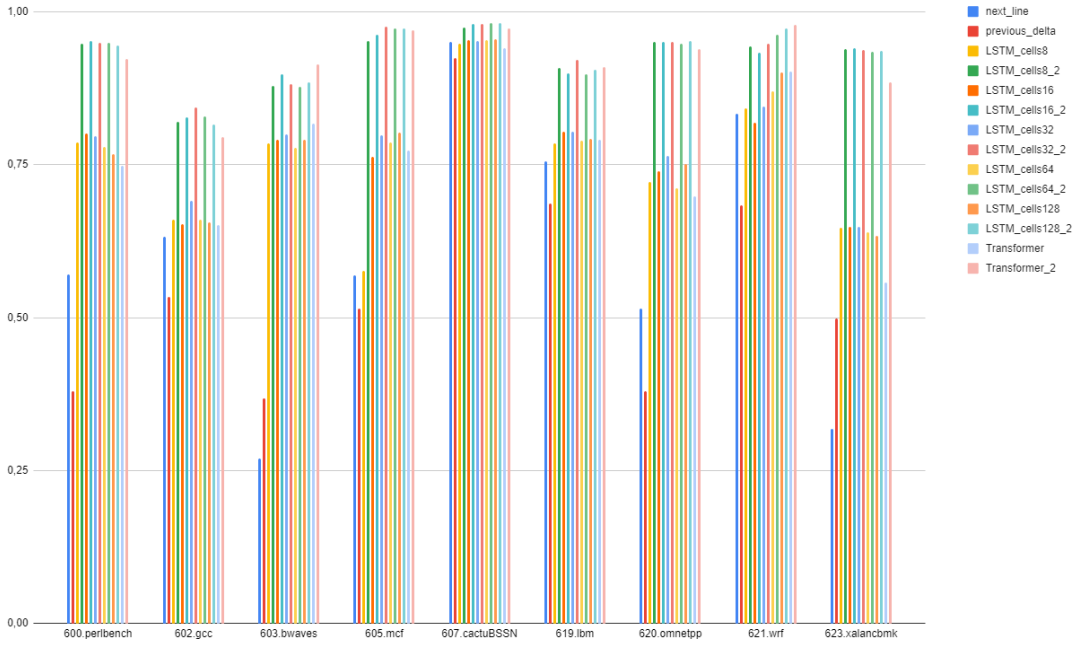
Πίνακας 6.6: *Ακρίβεια μοντέλων διπλής πρόβλεψης*

Σημειώνεται πως για το μετροπρόγραμμα 638 δεν υπάρχουν εγγραφές στους πίνακες 6.3 και 6.4 καθώς το συλλεγόμενο ιστορικό αποτελώντας αποκλειστικά από εγγραφές εκτός του ορίου της σελίδας, για τις οποίες δεν εκτελούμε προανάκτηση. Περιλαμβάνεται όμως στους πίνακες για λόγους συνοχής των αποτελεσμάτων.

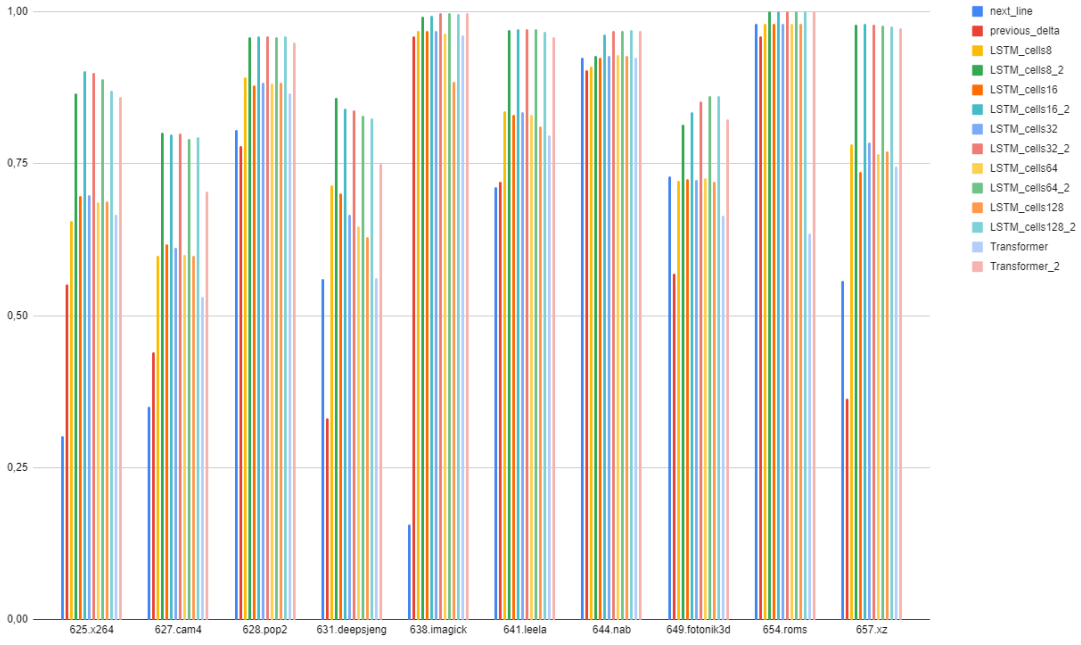
6.2.2 Μοντέλα με παράλληλη χρήση προανάκτησης επόμενης γραμμής στις κρυφές μνήμες πρώτου και δευτέρου επιπέδου

Στην ενότητα αυτή παρουσιάζονται τα αποτελέσματα των μοντέλων που αναπτύχθηκαν με δεδομένα που λήφθηκαν και πάλι με τον ίδιο τρόπο απο τον προσομοιωτή, με ενεργοποίηση όμως μηχανισμού προανάκτησης επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου.

Από τους πίνακες 6.5 και 6.6 αλλά και από το σχήμα 6.4, παρατηρούμε ότι η προανάκτηση επόμενης γραμμής έχει συγκριτικά καλύτερα αποτελέσματα σε σχέση με την πρώτη περίπτωση, γεγονός που επιβεβαιώνει, ως ένα βαθμό, την αλλαγή του μοντέλου πρόσβασης στη μνήμη και



(α′)

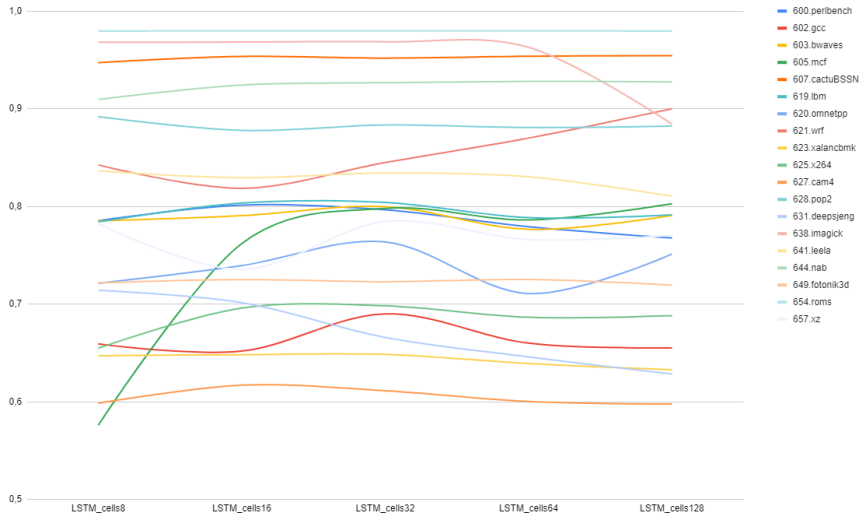


(β′)

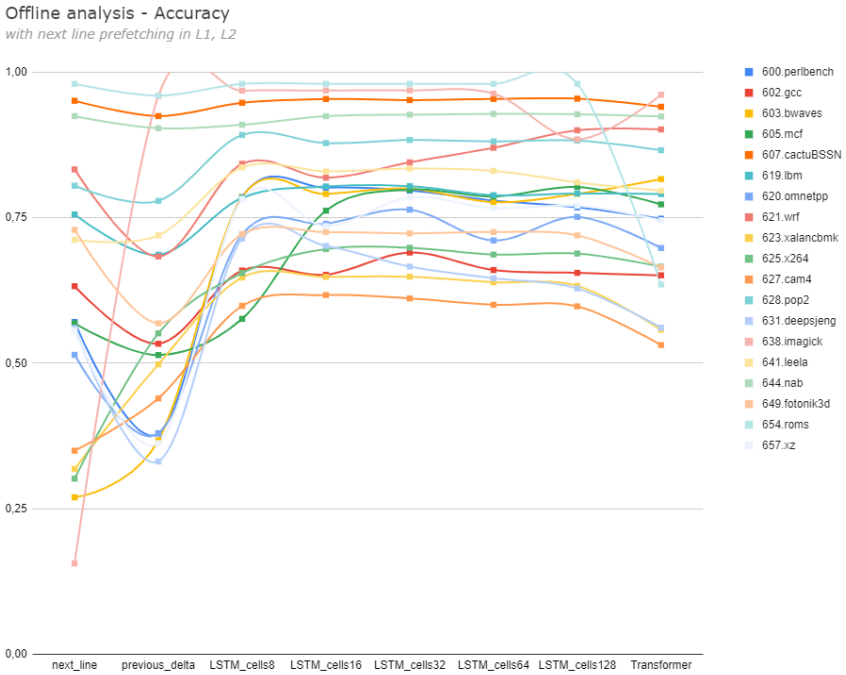
Σχήμα 6.4: *Ποσοστό επιτυχημένων προβλέψεων για καθένα απο τα σχήματα προανάκτησης σε δεδομένα που αφορούν την εκτέλεση των μετροπρογραμμάτων με παράλληλη χρήση προανάκτησης επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου*

συνεπώς την ανάγκη για εκ νέου ανάλυση. Και πάλι, η διπλή πρόβλεψη των δικτύων φαίνεται να συμβάλει σημαντικά στη βελτίωση της απόδοσης και μάλιστα σε μεγαλύτερο βαθμό (έως και 58%, μετροπρόγραμμα 623). Οι δύο αυτές παρατηρήσεις συνδυαστικά, προσδιορίζουν μοτίβα πρόσβασεις των οποίων οι περισσότερες εγγραφές αντιστοιχούν σε μοναδιαίο βήμα

με παρένθετες τυχαίες ή μη κανονικές προσβάσεις, που καθιστούν ιδιαίτερα δύσκολη την πρόβλεψη.



Σχήμα 6.5: *Γράφιμα μεταβολής ακρίβειας σε σχέση με τον αριθμό των LSTM κυττάρων*



Σχήμα 6.6: *Πλήρες διάγραμμα απόδοσης όλων των συγκρινόμενων μεθόδων - offline analysis*

Σε ό,τι αφορά τα χρησιμοποιούμενα LSTM κύτταρα, δηλαδή το μέγεθος του χρησιμοποιούμενου δικτύου (Σχήμα 6.5), σε αυτή την περίπτωση διακρίνονται περιπτώσεις όπου μεγαλύτερος αριθμός κυττάρων οδηγεί σε μικρότερη απόδοση. Το γεγονός αυτό δικαιολογείται από απλά μοντέλα πρόσβασεις στα οποία παρεμβάλλονται τυχαίες προσπελάσεις μνήμης, οι οποίες, δοθέντων περισσότερων παραμέτρων, υπολογίζονται ως μέρος της κύριας ροής.

	Επιλεγόμενο Μοντέλο (α)	Επιλεγόμενο Μοντέλο (β)
600.perlbench	LSTM_cells128	LSTM_cells16
602.gcc	LSTM_cells128	LSTM_cells32
603.bwaves	Transformer	Transformer
605.mcf	Transformer	LSTM_cells32
607.cactuBSSN	LSTM_cells8	LSTM_cells64
619.lbm	LSTM_cells8	LSTM_cells32
620.omnetpp	LSTM_cells128	LSTM_cells128
621.wrf	Transformer	Transformer
623.xalancbmk	LSTM_cells128	LSTM_cells16
625.x264	Transformer	LSTM_cells16
627.cam4	LSTM_cells128	LSTM_cells8
628.pop2	LSTM_cells128	LSTM_cells32
631.deepsjeng	LSTM_cells8	LSTM_cells8
638.imagick	#N/A	Transformer
641.leela	Transformer	LSTM_cells64
644.nab	LSTM_cells8	LSTM_cells128
649.fotonik3d	LSTM_cells32	LSTM_cells64
654.roms	LSTM_cells8	LSTM_cells8
657.xz	LSTM_cells16	LSTM_cells16

Πίνακας 6.7: *Επιλεγόμενα μοντέλα για (α) την αξιολόγηση αποκλειστικά του νευρωνικού μοντέλου και (β) την αξιολόγηση του μοντέλου σε συνδυασμό με προανάκτηση επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου*

Τέλος, αξιολογώντας την απόδοση του σχήματος Transformer, δεν μπορούμε να συμπεράνουμε με βάση τα αποτελέσματα για την υπεροχή του ή όχι έναντι των LSTM δικτύων, με την πλειονότητα βέβαια των μετροπρογραμμάτων να παρουσιάζουν μικρότερη ακρίβεια.

6.3 Συγκριτική αξιολόγηση μεθόδων προανάκτησης στο πλαίσιο του προσομοιωτή Champsim

Από τα μοντέλα που παρουσιάστηκαν στην παραπάνω παράγραφο επιλέχθηκε για κάθε ένα εκ των μετροπρογραμμάτων, αυτό με τη μεγαλύτερη ακρίβεια. Συγκεκριμένα, τα μοντέλα που επιλέχθηκαν παρουσιάζονται στον Πίνακα 6.7. Σε όλες τις εκτελέσεις τα μοντέλα που χρησιμοποιήθηκαν είναι διπλής πρόβλεψης, λόγω της σαφώς καλύτερης κάλυψης. Στη συνέχεια, παρουσιάζονται τα αποτελέσματα της εκτέλεσης στο πλαίσιο του προσομοιωτή τόσο σε απόλυτη μορφη (εντολές ανά κύκλο) στον πίνακα 6.8 και το σχήμα 6.7 αλλά και σε σχετική (επιτάχυνση) στον πίνακα 6.9 και το αντίστοιχο σχήμα (Σχήμα 6.8).

Από τα δεδομένα αυτά παρατηρούμε πως πλήθος μετροπρογραμμάτων από τα εξεταζόμενα δεν επηρεάζονται από τη χρησιμοποίηση ή όχι μηχανισμού προανάκτησης. Χαρακτηριστικό παράδειγμα αποτελεί το μετροπρόγραμμα 648, το οποίο δεν συμπεριλήφθηκε στην έως τώρα ανάλυση, καθώς δεν παρουσιάζει στην εκτέλεση του misses στην κρυφή μνήμη τρίτου επιπέδου και συνεπώς δεν υπάρχει ιστορικό για την εκπαίδευση των νευρωνικών δικτύων. Ακόμη όμως και οι μέθοδοι που αφορούν τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου δεν επηρεάζουν ουσιαστικά την εκτέλεση του προγράμματος, παρα μόνο μεταβάλλουν ελάχιστα των αριθμό των hits και misses. Ακόμα, γίνεται φανερό πως η χρήση προανάκτησης δεν έχει πάντα τα

	no no no	next_line no no	no next_line no	no no next_line	next_line next_line no	no next_line next_line	next_line next_line next_line	no - no LSTM / transformer	ipcp ipcp ipcp	next_line next_line LSTM
600.perlbench	1,3853	1,3879	1,3883	1,3879	1,3898	1,3896	1,3906	1,3885	1,3904	1,3906
602.gcc	0,4368	0,7157	0,6885	0,6116	0,7432	0,7106	0,7627	0,6177	0,7792	0,7647
603.bwaves	1,1686	1,1729	1,1727	1,1722	1,1730	1,1727	1,1732	1,1719	1,1730	1,1732
605.mcf	0,3772	0,3920	0,3909	0,3875	0,3829	0,3834	0,3665	0,4074	0,4086	0,3665
607.cactuBSSN	1,1931	1,9273	1,9649	1,8805	1,9766	1,9692	1,9965	1,8792	1,8394	1,9963
619.lbm	0,3964	0,4087	0,4105	0,4103	0,4348	0,4437	0,4507	0,4505	0,4667	0,4536
620.omnetpp	0,3361	0,3513	0,3521	0,3510	0,3524	0,3539	0,3550	0,3441	0,3502	0,3551
621.wrf	1,1071	1,1082	1,1081	1,1080	1,1082	1,1081	1,1083	1,1080	1,1083	1,1083
623.xalancbmk	0,5987	0,5956	0,5840	0,5998	0,5729	0,5838	0,5730	0,6009	0,5790	0,5718
625.x264	1,4329	1,4698	1,4708	1,4669	1,4724	1,4724	1,4734	1,4587	1,4777	1,4748
627.cam4	1,0527	1,0611	1,0651	1,0628	1,07015	1,0675	1,0724	1,0561	1,0728	1,0715
628.pop2	1,1556	1,2922	1,2933	1,2676	1,3254	1,3156	1,3561	1,2649	1,3817	1,3579
631.deepsjeng	0,9633	0,9653	0,9654	0,9654	0,9667	0,9667	0,9677	0,9663	0,9660	0,9677
638.imagick	2,3702	2,3848	2,3831	2,3826	2,3848	2,3831	2,3847		2,3894	2,3847
641.leela	0,7428	0,7504	0,7487	0,7473	0,7505	0,7488	0,7520	0,7474	0,7572	0,7521
644.nab	1,1942	1,2313	1,2174	1,2150	1,2318	1,2175	1,2319	1,2150	1,2324	1,2320
648.exchange2	1,3268	1,3268	1,3268	1,3268	1,3268	1,3268	1,3268		1,3268	
649.fotonik3d	0,6070	1,0532	1,0179	0,9601	1,0922	1,0523	1,2756	1,0785	1,9265	1,3804
654.roms	1,0130	1,0216	1,0210	1,0197	1,0216	1,0210	1,0216	1,0197	1,0217	1,0216
657.xz	1,0147	1,0200	1,0199	1,0197	1,0137	1,0160	1,0107	1,0128	1,0106	1,0106

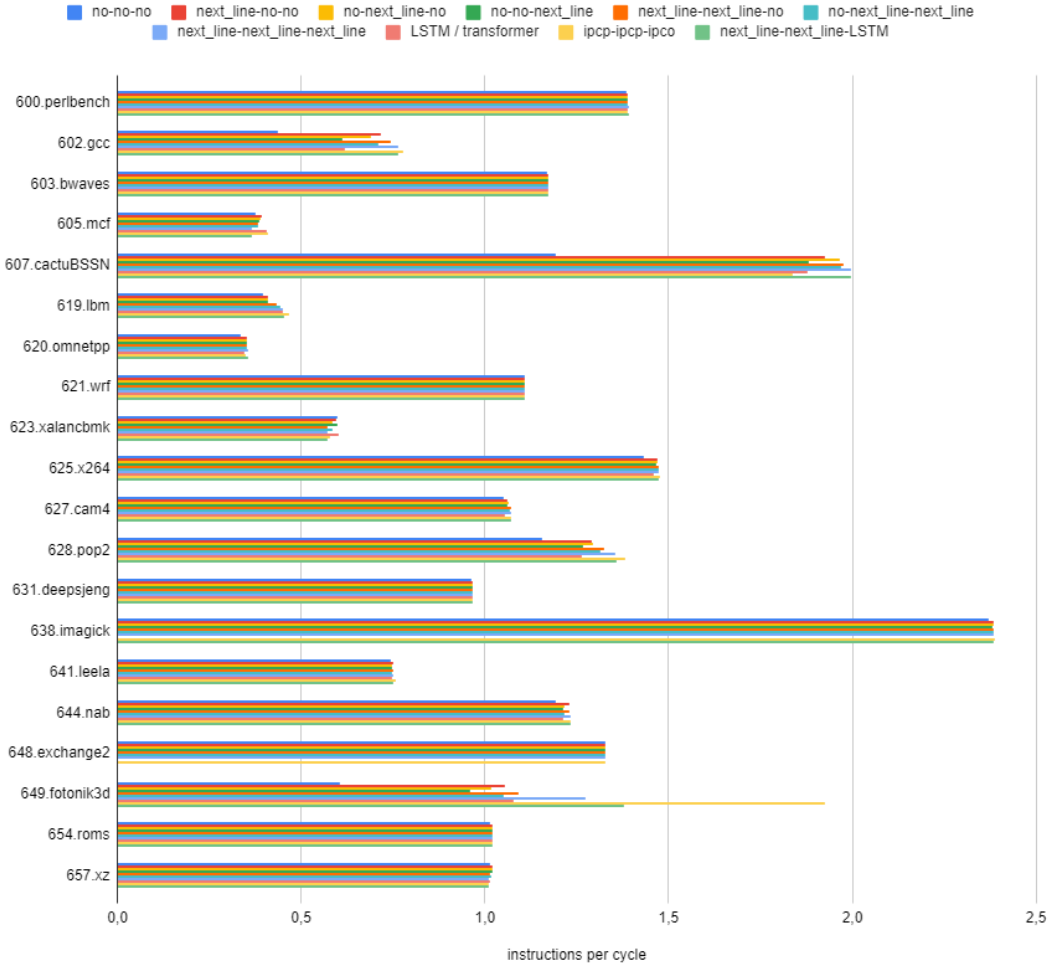
Πίνακας 6.8: *Απόδοση μεθόδων προανάκτησης σε εντολές ανά κύκλο*

	next_line no no	no next_line no	no no next_line	next_line next_line no	no next_line next_line	next_line next_line next_line	no - no LSTM / transformer	ipcp ipcp ipcp	next_line next_line LSTM
600.perlbench	0,1898	0,2166	0,1862	0,3212	0,3104	0,3833	0,2303	0,3667	0,3804
602.gcc	63,8419	57,6189	40,0109	70,1402	62,6926	74,6134	41,4141	78,3937	75,06398839
603.bwaves	0,3714	0,3586	0,3089	0,3774	0,3586	0,3937	0,2876	0,3782	0,3979
605.mcf	3,9256	3,6364	2,7310	1,5226	1,6342	-2,8294	8,0215	8,3171	-2,8294
607.cactuBSSN	61,5340	64,6880	57,6176	65,6737	65,0526	67,3392	57,5108	54,17071781	67,3265833
619.lbm	3,1066	3,5647	3,5148	9,7107	11,9557	13,7115	13,6534	17,7462	14,4406
620.omnetpp	4,5140	4,7366	4,4340	4,8407	5,2739	5,6243	2,3818	4,1850	5,640394015
621.wrf	0,0912	0,0840	0,0741	0,0957	0,08852	0,1003	0,0804	0,1012	0,1003
623.xalancbmk	-0,5231	-2,4546	0,1832	-4,3003	-2,4850	-4,2926	0,3626	-3,2907	-4,4870
625.x264	2,5724	2,6464	2,3728	2,7566	2,7559	2,8236	1,8026	3,1237	2,9241
627.cam4	0,8017	1,1817	0,9575	1,65763	1,4012	1,8685	0,3239	1,9084	1,7887
628.pop2	11,8181	11,9116	9,6937	14,6945	13,8448	17,3477	9,45656	19,5665	17,5017
631.deepsjeng	0,2075	0,2154	0,2157	0,3460	0,3522	0,4495	0,3058	0,2756	0,4495
638.imagick	0,6147	0,5434	0,5253	0,6147	0,5434	0,6135	0,0	0,8126	0,6135
641.leela	1,0203	0,7885	0,6105	1,0373	0,8118	1,2387	0,6143	1,9373	1,2470
644.nab	3,1025	1,9410	1,7443	3,1502	1,954	3,1594	1,7442	3,1980	3,161112042
648.exchange2	0	0	0	0	0	0	0	0	-100
649.fotonik3d	73,5116	67,6848	58,1707	79,9316	73,3486	110,1434	77,6730	217,3732	127,4000
654.roms	0,8559	0,7937	0,6614	0,8559	0,7937	0,8559	0,6614	0,8569	0,8559
657.xz	0,5204	0,5174	0,4997	-0,0936	0,1301	-0,3942	-0,1853	-0,4011	-0,3952
Average	11,6038	11,0337	9,2256	12,6666	12,0409	14,6575	10,8170	20,4510	10,5790

Πίνακας 6.9: *Απόδοση μεθόδων προανάκτησης, επιτάχυνση ως προς την εκτέλεση χωρίς προανάκτηση*

επιθυμητά αποτελέσματα. Στην περίπτωση του μετροπρογράμματος 623 η απόδοση μειώνεται όταν εφαρμόζεται κάποια από της εξεταζόμενες μεθόδους που αφορά τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου. Το γεγονός αυτό φυσικά μπορεί να οφείλεται σε διαφορετικό μοντέλο, αντίθετο ως προς αυτό της επόμενης γραμμής, τα εξίσου όμως αρνητικά αποτελέσματα και με τη χρήση του bouquet prefetcher, δίνουν κίνητρο περαιτέρω διερεύνηση της ακολουθίας σφαλμάτων στις περιπτώσεις αυτές.

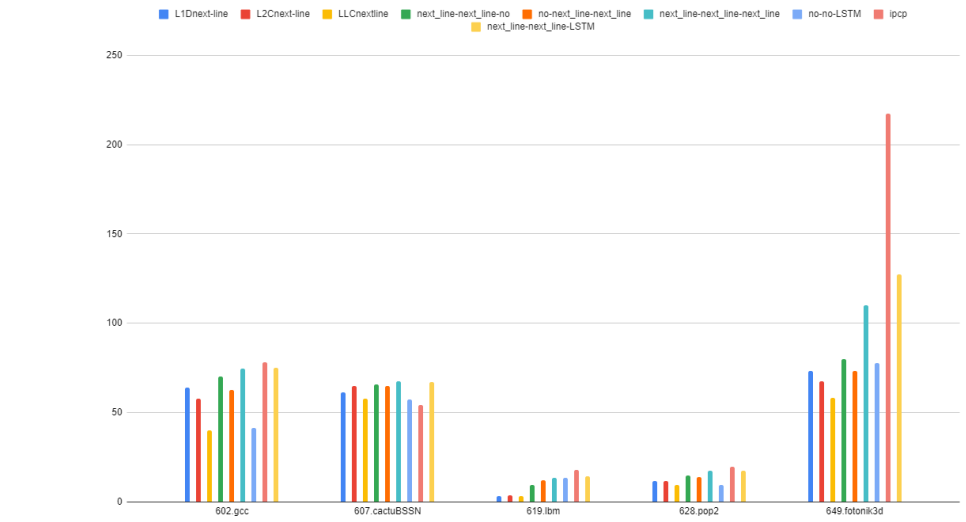
Ακόμη, μεταξύ των μετροπρογραμμάτων υπάρχουν μοντέλα των οποίων η απόδοση του μηχανισμού δεν είναι αντίστοιχη της αναμενόμενης, βάση της ακρίβειας του μοντέλου κατά την offline αξιολόγηση του. Για παράδειγμα, το μετροπρόγραμμα 627 φαίνεται να αποδίδει καλύτερα με προανάκτηση επόμενης γραμμής, συγκριτικά με τα νευρωνικά μοντέλα, ενώ με βάση τα



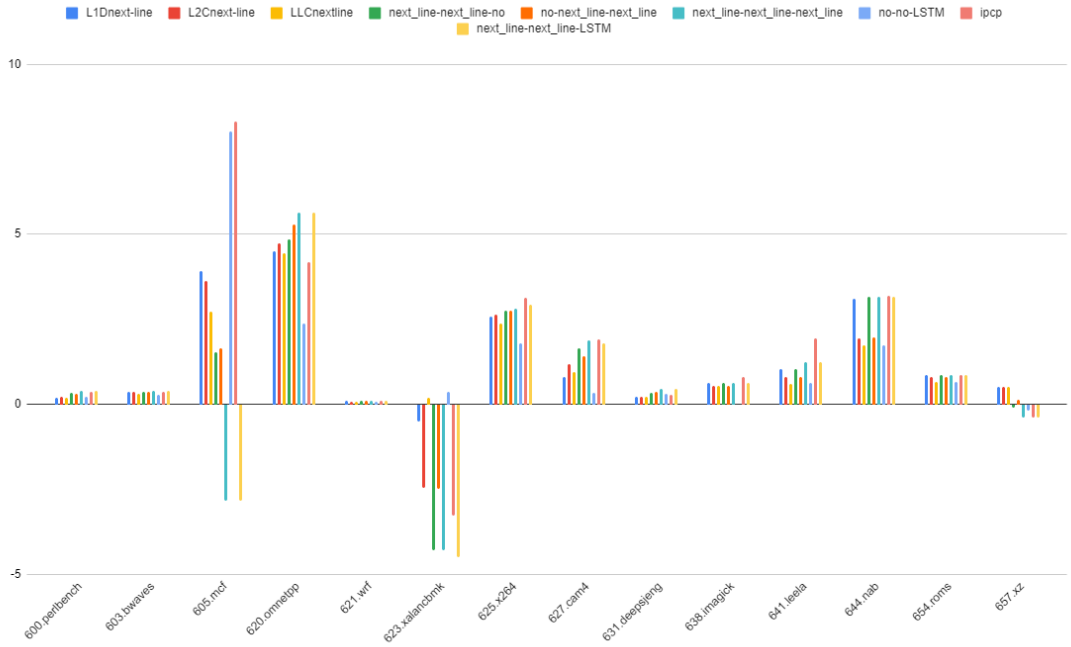
Σχήμα 6.7: *Απόδοση μεθόδων προανάκτησης σε εντολές ανά κύκλο*

αποτελέσματα ακρίβειας (Πίνακες 6.3 και 6.5) αναμένουμε το αντίθετο και ιδίως αφού χρησιμοποιούνται μοντέλα διπλής πρόβλεψης, τα οποία παρουσιάζουν ακόμη μεγαλύτερη ακρίβεια. Προκειμένου να προσδιοριστεί εάν το αποτέλεσμα αυτό οφείλεται σε λανθασμένη πρόβλεψη του μοντέλου, είτε σε ζητήματα χρονισμού, τα οποία δεν έχουν ληφθεί υπόψιν στη δεδομένη εργασία, προστέθηκαν οι κατάλληλες μεταβλητές έτσι ώστε να αποθηκεύεται η τιμη πρόβλεψης και να συγκρίνεται με την επόμενη τιμή για δεδομένο PC. Η ακρίβεια του μοντέλου που υπολογίστηκε με αυτόν τον τρόπο ήταν σημαντικά πιο χαμηλή από την αναμενόμενη και συγκεκριμένα 49,39% αντί του 80,05%.

Πολλές είναι οι παράμετροι στις οποίες μπορεί να αποδοθεί αυτό το αποτέλεσμα. Αφενός, πολύ σημαντικό είναι το γεγονός ότι ο αριθμός των μοναδικών δεικτών προγράμματος είναι πολύ υψηλότερος από εκείνον που μπορεί να αναπαρασταθεί με βάση την κωδικοποίηση. Αφετέρου, εάν το μοντέλο πρόσβασης αποτελείται κυρίως από προσβάσεις επόμενης γραμμής, λοιπές προσβάσεις που ίσως αποτελούν θόρυβο (τυχαίες προσβάσεις) εκλαμβάνονται πιθανώς από το μοντέλο ως μέρος της ροής. Τέλος, αντιστοιχίζοντας το αποτέλεσμα offline ανάλυσης με αυτό της εκτέλεσης στο πλαίσιο εντός του προσομοιωτή, γίνεται η υπόθεση πως οι αιτήσεις προανάκτησης δεν επηρεάζουν το μοντέλο πρόσβασης.



(α′)



(β′)

Σχήμα 6.8: *Διάγραμμα επιτάχυνσης μηχανισμών με βάση την εκτέλεση χωρίς τη χρήση προανάκτησης*

Με βάση τα παραπάνω και προκειμένου να αξιολογηθούν οι παράμετροι αυτοί πραγματοποιήθηκε για το δεδομένο μετροπρόγραμμα ανάλυση από την οποία προέκυψαν τα εξής: Για την εκδοχή Α (χωρίς προανάκτηση στις κρυφές μνήμες πρώτου και δευτέρου επιπέδου) στις συνολικά 184687 εγγραφές εμφανίζονται 12514 διαφορετικοί δείκτες προγράμματος, ενώ για την εκδοχή Β (προανάκτηση επόμενης γραμμής στις κρυφές μνήμες πρώτου και δευτέρου επιπέδου) στις 172921 αντίστοιχα 12293, αριθμοί δέκα φορές πολλαπλάσιοι από τις δυνατές αναπαραστάσεις. Βέβαια μόνο περίπου τετρακόσιοι από αυτούς έχουν εμφανίσεις πάνω από τις τριάντα τρεις (33) και στις δύο περιπτώσεις, που απαιτούνται για την χρησιμοποίηση τους

στην εκπαίδευση του νευρωνικού αλλά και την πρόβλεψη κατα τη διάρκεια της εκτέλεσης. Σε συνέχεια αυτού, αναζητήθηκαν παράθυρα αποτελούμενα από τριάντα τρία Δέλτα (τριάντα δύο που αποτελούν την ακολουθία εισόδου και ένα για την επιθυμητή πρόβλεψη) μεταξύ του ιστορικού για κάθε PC χωρίς τη χρήση προανάκτησης αλλά και με αυτη για την κρυφή μνήμη τελευταίου επιπέδου. Από τις ακολουθίες που ελέγχθηκαν (496.449.536) υπήρξε αντιστοιχία σε μόλις 28917. Από αυτές βέβαια οι περισσότερες ανήκουν στα δύο PCs που εμφανίζονται με μεγαλύτερη συχνότητα (10% των συνολικών εμφανίσεων).

Πέραν της ανάλυσης αυτής, πραγματοποιήθηκαν αντίστοιχες μετρήσεις ακρίβειας και για άλλα μετροπρογράμματα για τον έλεγχο της σωστής λειτουργίας του μηχανισμού, η οποία και διαπιστώθηκε.

Κεφάλαιο 7

Συμπεράσματα και Μελλοντικές Επεκτάσεις

Η μελέτη αυτή αποτέλεσε μία προσέγγιση του ζητήματος της πρανάκτησης με χρήση νευρωνικών δικτύων. Στο κεφάλαιο αυτό παρουσιάζονται τα συμπεράσματα καθώς και προτάσεις που προκύπτουν βάση αυτών για περαιτέρω ανάλυση και εξέλιξη του μηχανισμού υπό αυτή τη σκοπιά.

7.1 Συμπεράσματα

Από τα αποτελέσματα της πειραματικής αξιολόγησης για καθένα από τα σχήματα προανάκτησης που χρησιμοποιήθηκαν προκύπτουν τα εξής:

Η χρήση νευρωνικών δικτύων για την πρόβλεψη μελλοντικών προσβάσεων μπορεί να φανεί εξαιρετικά σημαντική ως καθολική μέθοδος, καθώς ανταποκρίνεται ικανοποιητικά καλά στην πλειοψηφία των μετροπρογραμμάτων, με την απόδοση να είναι συγκρίσιμη με αυτή του προανακτητή bouquet, ο οποίος χρησιμοποιήθηκε κατά μία έννοια ως βέλτιστη προσέγγιση. Από την άλλη πλευρά, η εφαρμογή του μηχανισμού σε συνδυασμό με προανάκτηση επόμενης γραμμής για τις κρυφές μνήμες πρώτου και δευτέρου επιπέδου δεν φαίνεται να συμβάλλει σημαντικά στην απόδοση. Μεταξύ των μοντέλων LSTM και transformer, δεν προκύπτουν επαρκή δεδομένα για την συγκριτική αξιολόγηση και ταξινόμηση των δύο μοντέλων, ενώ σε ότι αφορά τον αριθμό των χρησιμοποιούμενων LSTM κυττάρων, η αύξηση τους συντελεί στην πρόβλεψη και πιο πολύπλοκων μοντέλων. Η χρησιμοποίηση διπλής πρόβλεψης του μοντέλου, αφενός οδηγεί σε μεγαλύτερη ακρίβεια και συνεπώς δυνητικά σε μεγαλύτερο IPC, συνεπάγεται όμως και πολλαπλές αιτήσεις προανάκτησης που περιορίζουν το εύρος ζώνης της μνήμης. Εκπαίδευση και χρήση μοντέλων μεγαλύτερης ακρίβειας θα μπορούσαν και είναι σκόπιμο να αντικαταστήσουν το μηχανισμό αυτό. Τέλος, όπως προκύπτει και από τα αποτελέσματα της παραγράφου 6.3 για το μετροπρόγραμμα 627 η χρησιμοποιούμενη κωδικοποίηση (αναπαράσταση δεικτών προγράμματος και Δέλτα με χρήση των έξι λιγότερο σημαντικών ψηφίων) δεν καλύπτει ολοκληρωτικά τις ανάγκες σε πλήθος αναπαραστάσεων για όλα τα μετροπρογράμματα.

7.2 Μελλοντικές Επεκτάσεις

Τόσο τα ιδια τα αποτελέσματα της πειραματικής αξιολόγησης, όσο και οι παραδοχές και οι απλοποιήσεις στο πλαίσιο του σχεδιασμού αφήνουν το περιθώριο για περαιτέρω μελέτη και ενασχόληση σχετικά με την εφαρμογή των νευρωνικών δικτύων για την υλοποίηση ενός καθολικά αποδοτικού μηχανισμού προανάκτησης.

Ξεκινώντας από τα δεδομένα τα οποία χρησιμοποιήθηκαν για την εκπαίδευση των νευρωνικών δικτύων,το ιστορικό προκύπτει από τις αστοχίες της κρυφής μνήμης τρίτου επιπέδου. Από την περαιτέρω ανάλυση που πραγματοποιήθηκε για το μετροπρόγραμμα 627 γίνεται φανερό πως η ακολουθία αυτή των αστοχιών μεταβάλλεται με τη χρήση κάποιου prefetcher. Για την αντιμετώπιση αυτού, λύση αποτελεί η καταγραφή του συνόλου των προσβάσεων στη μνήμη για κάθε δείκτη προγράμματος, τόσο των misses όσο και των hits, συνθήκη όμως που μεταβάλλει σημαντικά πέρα από την κατανομή και το μέγεθος του dataset.

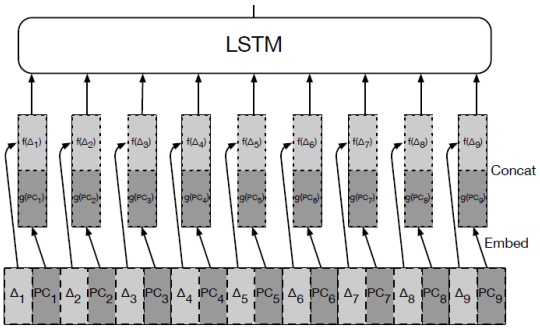
Σε ότι αφορά την υλοποίηση των νευρωνικών δικτύων, όπως έχει ήδη προταθεί και από άλλες εργασίες, σημαντική παράμετρος θεωρείται το lookback size, το οποίο δεν έχει αξιολογηθεί στη δεδομένη εργασία (ως παράμετρος διατηρήθηκε σταθερή και ίση με 32). Μεταβολή του lookback size μπορεί να οδηγήσει σε σημαντικά καλύτερη απόδοση του μοντέλου για αρκετά μικρό μέγεθος αυτού [19].

Όπως έχει ήδη αναφερθεί, σημείο της δεδομένης εργασίας που προσφέρεται για πειραματισμό και μελέτη αποτελεί η χρησιμοποιούμενη κωδικοποίηση, έτσι ώστε να παραμένει απλή στην υλοποίηση, ενώ ταυτόχρονα να καλύπτει τις απαιτήσεις σε πλήθος αναπαραστάσεων. Για την αντιμετώπιση αυτού ίσως θα ήταν ωφέλιμη διαφορετική διαχείριση των PCs από την ακολουθία των Δέλτα. Στην υλοποίηση όμως αυτή απαιτείται πολυπλοκότερη δομή του νευρωνικού δικτύου.

Ακόμη, διαφορετική θα μπορούσε να είναι η μορφή της εισόδου. Αντί της χρησιμοποίησης του PC στην αρχή της ακολουθίας, ως το πρώτο εκ των στοιχείων, θα μπορούσε να ενσωματωθεί ως πληροφορία σε κάθε ένα από τα επιμέρους στοιχεία αυτής, όπως διαμορφώνεται η είσοδος στην εργασία Learning Memory access patterns [21] (Σχήμα 7.1). Σημαντικό είναι να σημειωθεί πως η επιλογή να μην ακολουθεί το συγκεκριμένο μοντέλο πραγματοποιήθηκε λόγω της κατηγοριοποίησης και χρήσης Δέλτα που αφορούν κάθε φορά μόνο έναν δείκτη προγράμματος.

Μία σημαντική παράμετρος, που αφορά το κομμάτι της προανάκτησης και όχι του νευρωνικού δικτύου, η οποία δεν έχει ληφθεί υπόψιν στην παρούσα μελέτη, είναι ο χρονισμός. Ακόμη και σε περιπτώσεις που η ακρίβεια της πρόβλεψης είναι πολύ υψηλή, εάν η αίτηση προανάκτησης δεν πραγματοποιηθεί εγκαίρως δεν αξιοποιούνται οι δυνατότητες του σχεδιαζόμενου μηχανισμού και συνεπώς η επιτάχυνση σε εντολές ανά κύκλο είναι ανακόλουθη των αποτελεσμάτων της offline ανάλυσης του μοντέλου.

Ένας εκ των περιορισμών που χρησιμοποιήθηκαν στη δεδομένη υλοποίηση των μηχανισμών



Σχήμα 7.1: *Γράφημα διαμόρφωσης εισόδου, όπως προτείνεται στην εργασία Learning Memory Access Patterns*

και φυσικά επηρεάζουν τη μορφή των δεδομένων που δέχονται τα νευρωνικά δίκτυα είναι ο καθορισμός του ορίου προανάκτησης στο μέγεθος της σελίδας. Αξιοσημείωτη φαίνεται η αξιολόγηση της βελτίωσης της απόδοσης με την άρση αυτού του περιορισμού, ιδίως αν ληφθούν υπόψη περιπτώσεις μετροπρογραμμάτων όπου δεν καταγράφονται στο ιστορικό μελλοντικές προσβάσεις στο πλαίσιο αυτής. Η μεταβολή αυτή, φυσικά οδηγεί σε σημαντικά μεγαλύτερο πλήθος πιθανών Δέλτα, και συνεπώς αύξηση του μεγέθους του νευρωνικού δικτύου, όσο η προανάκτηση αντιμετωπίζεται ως πρόβλημα ταξινόμησης. Στην αντιμετώπιση αυτού θα μπορούσε συμβάλει η πρόταση του Pierre Michaud στο "Best-Offset Hardware Prefetching" [62] όπου χρησιμοποιούνται αποκλειστικά offsets των οποίων οι πρώτοι παράγοντες δεν περιλαμβάνουν αριθμούς μεγαλύτερους του 5. Συμφωνα με την ανάλυση του, η επιλογή αυτή έχει τα πλεονεκτήματα αφενός της επιλογής Δέλτα μέσα από ένα σημαντικά μικρότερο πλήθος και αφετέρου την επιλογή Δέλτα (μικρότερα) με συγκριτικά περισσότερες εμφανίσεις στο σύνολο.

Τέλος, για την περίπτωση στην οποία διατηρείται η διπλή πρόβλεψη για λόγους υψηλής ακρίβειας, ωφέλιμη θα ήταν πιθανώς η λειτουργία κάποιου μηχανισμού στραγγαλισμού (throttling mechanism) προκειμένου να περιοριστούν πολλαπλές επιβλαβείς αιτήσεις προανάκτησης, σε περιπτώσεις όπου οι προβλέψεις είναι ατυχείς ή συνολικά ο μηχανισμός προανάκτησης δεν συμβάλλει θετικά στην απόδοση της εκτελούμενης εφαρμογής.

Παραρτήματα

Παράρτημα Α΄

Μοντέλο Transformer

Ο παρακάτω κώδικας αφορά την υλοποίηση και εκπαίδευση του μοντέλου Transformer

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import pandas as pd
from numpy import array
from sklearn.model_selection import train_test_split
from keras.models import model_from_json

from imblearn.over_sampling import RandomOverSampler #SMOTE
    sm = RandomOverSampler()#SMOTE()

class TransformerBlock(layers.Layer):
    def __init__(self, embed_dim, num_heads, ff_dim, rate=0.1):
      super(TransformerBlock, self).__init__()
      self.att = layers.MultiHeadAttention(num_heads=num_heads, key_dim=embed_-
dim)
      self.ffn = keras.Sequential(
        [layers.Dense(ff_dim, activation="relu"), layers.Dense(embed_dim),]
      )
      self.layernorm1 = layers.LayerNormalization(epsilon=1e-6)
      self.layernorm2 = layers.LayerNormalization(epsilon=1e-6)
      self.dropout1 = layers.Dropout(rate)
      self.dropout2 = layers.Dropout(rate)

    def call(self, inputs, training):
      attn_output = self.att(inputs, inputs)
      attn_output = self.dropout1(attn_output, training=training)
      out1 = self.layernorm1(inputs + attn_output)
      ffn_output = self.ffn(out1)
      ffn_output = self.dropout2(ffn_output, training=training)
      return self.layernorm2(out1 + ffn_output)

    def get_config(self):
```

```
      config = super().get_config().copy()
      config.update(
          'att':self.att,
          'ffn':self.ffn,
          'layernorm1':self.layernorm1,
          'layernorm2':self.layernorm2,
          'dropout1':self.dropout1,
          'dropout2':self.dropout2
        )
      return config

class TokenAndPositionEmbedding(layers.Layer):
    def __init__(self, maxlen, vocab_size, embed_dim):
      super(TokenAndPositionEmbedding, self).__init__()
      self.token_emb = layers.Embedding(input_dim=vocab_size, output_dim=embed_-
dim)
      self.pos_emb = layers.Embedding(input_dim=maxlen, output_dim=embed_-
dim)

    def call(self, x):
      maxlen = tf.shape(x)[-1]
      positions = tf.range(start=0, limit=maxlen, delta=1)
      positions = self.pos_emb(positions)
      x = self.token_emb(x)
      return x + positions

    def get_config(self):
      config = super().get_config().copy()
      config.update(
          'token_emb':self.token_emb,
          'pos_emb':self.pos_emb
        )
      return config

data = pd.read_csv("input_file.txt", header = None)
data.columns = ['ip','lsb8', 'lsb7', 'lsb6', 'x1','x2', 'x3', 'x4','x5',
'x6', 'x7','x8', 'x9', 'x10','x11', 'x12', 'x13','x14', 'x15', 'x16','x17',
'x18', 'x19','x20', 'x21', 'x22','x23', 'x24', 'x25','x26', 'x27', 'x28','x29',
'x30','x31', 'x32', 'y']

x = data.loc[:,'lsb6':'x32']
y = data.loc[:, 'y']
yy = array(y) + 64
xx = array(x) +64

x_res,y_res=sm.fit_resample(xx,yy)
X_train, X_test, y_train, y_test = train_test_split(x_res, y_res, test_-
size=0.2, random_state=1)

vocab_size =700
```

```
maxlen =33


embed_dim = 32 # Embedding size for each token
num_heads = 2 # Number of attention heads
ff_dim = 32 # Hidden layer size in feed forward network inside transformer

inputs = layers.Input(shape=(maxlen,))
embedding_layer = TokenAndPositionEmbedding(maxlen, vocab_size, embed_-
dim)
x = embedding_layer(inputs)
transformer_block = TransformerBlock(embed_dim, num_heads, ff_dim)
x = transformer_block(x)
x = layers.GlobalAveragePooling1D()(x)
x = layers.Dropout(0.1)(x)
x = layers.Dense(20, activation="relu")(x)
x = layers.Dropout(0.1)(x)
outputs = layers.Dense(129, activation="softmax")(x)

model = keras.Model(inputs=inputs, outputs=outputs)

model.summary()


model.compile("adam", "sparse_categorical_crossentropy", metrics=["accuracy"]
history = model.fit(
    xx, yy, batch_size=128, epochs=1, shuffle=True, validation_data=(X_-
test, y_test)
)

Model = "trained_model"
model.save(Model)
```

Βιβλιογραφία

[1] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola και Eduard Hovy. *Hierarchical Attention Networks for Document Classification. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, σελίδες 1480–1489, San Diego, California, 2016. Association for Computational Linguistics.

[2] Jean Loup Baer και Tien Fu Chen. *An Effective On-Chip Preloading Scheme to Reduce Data Access Penalty. Proceedings of the 1991 ACM/IEEE Conference on Supercomputing*, Supercomputing '91, σελίδα 176–186, New York, NY, USA, 1991. Association for Computing Machinery.

[3] Doug Joseph και Dirk Grunwald. *Prefetching Using Markov Predictors. SIGARCH Comput. Archit. News*, 25(2):252–263, 1997.

[4] K. J. Nesbit και J. E. Smith. *Data Cache Prefetching Using a Global History Buffer. 10th International Symposium on High Performance Computer Architecture (HPCA-'04)*, σελίδες 96–96, 2004.

[5] Changhee Jung, Daeseob Lim, Jaejin Lee και Y. Solihin. *Helper thread prefetching for loosely-coupled multiprocessor systems. Proceedings 20th IEEE International Parallel Distributed Processing Symposium*, σελίδες 10 ππ.–, 2006.

[6] Daniel A Jiménez και Calvin Lin. *Dynamic branch prediction with perceptrons. Proceedings HPCA Seventh International Symposium on High-Performance Computer Architecture*, σελίδες 197–206. IEEE, 2001.

[7] Renée St Amant, Daniel A Jiménez και Doug Burger. *Low-power, high-performance analog neural branch prediction. 2008 41st IEEE/ACM International Symposium on Microarchitecture*, σελίδες 447–458. IEEE, 2008.

[8] Daniel A Jiménez. *An optimized scaled neural branch predictor. 2011 IEEE 29th International Conference on Computer Design (ICCD)*, σελίδες 113–118. IEEE, 2011.

[9] Elba Garza, Samira Mirbagher-Ajorpaz, Tahsin Ahmad Khan και Daniel A Jiménez. *Bit-level perceptron prediction for indirect branches. 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA)*, σελίδες 27–38. IEEE, 2019.

[10] André Seznec. *Tage-sc-l branch predictors again. 5th JILP Workshop on Computer Architecture Competitions (JWAC-5): Championship Branch Prediction (CBP-5)*, 2016.

[11] Stephen J Tarsa, Chit Kwan Lin, Gokce Keskin, Gautham Chinya και Hong Wang. *Improving branch prediction by modeling global history with convolutional neural networks. arXiv preprint arXiv:1906.09889*, 2019.

[12] Engin Ipek, Onur Mutlu, José F Martínez και Rich Caruana. *Self-optimizing memory controllers: A reinforcement learning approach. ACM SIGARCH Computer Architecture News*, 36(3):39–50, 2008.

[13] Janani Mukundan και Jose F Martinez. *MORSE: Multi-objective reconfigurable self-optimizing memory scheduler. IEEE International Symposium on High-Performance Comp Architecture*, σελίδες 1–12. IEEE, 2012.

[14] Sai Manoj PD, Hao Yu, Hantao Huang και Dongjun Xu. *A Q-learning based self-adaptive I/O communication for 2.5 D integrated many-core microprocessor and memory. IEEE Transactions on Computers*, 65(4):1185–1196, 2015.

[15] Shibo Wang και Engin Ipek. *Reducing data movement energy via online data clustering and encoding. 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, σελίδες 1–13. IEEE, 2016.

[16] Wonkyung Kang και Sungjoo Yoo. *Dynamic management of key states for reinforcement learning-assisted garbage collection to reduce long tail latency in SSD. Proceedings of the 55th Annual Design Automation Conference*, σελίδες 1–6, 2018.

[17] Zhaoxia Deng, Lunkai Zhang, Nikita Mishra, Henry Hoffmann και Frederic T Chong. *Memory cocktail therapy: A general learning-based framework to optimize dynamic tradeoffs in nvms. Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, σελίδες 232–244, 2017.

[18] Jiang Xiao, Zhuang Xiong, Song Wu, Yusheng Yi, Hai Jin και Kan Hu. *Disk failure prediction in data centers via online learning. Proceedings of the 47th International Conference on Parallel Processing*, σελίδες 1–10, 2018.

[19] P. Braun και H. Litz. *Understanding memory access patterns for prefetching. in International Workshop on AI-assisted Design for Architecture (AIDArc), held in conjunction with ISCA*, 2019.

[20] Ji Tae Yun, Su Kyung Yoon, Jeong Geun Kim και Shin Dug Kim. *Effective data prediction method for in-memory database applications. The Journal of Supercomputing*, 76, 2020.

[21] Milad Hashemi, Kevin Swersky, Jamie Smith, Grant Ayers, Heiner Litz, Jichuan Chang, Christos Kozyrakis και Parthasarathy Ranganathan. *Learning memory access patterns. International Conference on Machine Learning*, σελίδες 1919–1928. PMLR, 2018.

[22] Sepp Hochreiter και Jürgen Schmidhuber. *Long Short-term Memory. Neural computation*, 9:1735–80, 1997.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser και Illia Polosukhin. *Attention Is All You Need. CoRR*, αβς/1706.03762, 2017.

[24] Wm. A. Wulf και Sally A. McKee. *Hitting the Memory Wall: Implications of the Obvious. SIGARCH Comput. Archit. News*, 23(1):20–24, 1995.

[25] C. Kozyrakis, A. Kansal, S. Sankar και K. Vaid. *Server Engineering Insights for Large-Scale Online Services. IEEE Micro*, 30(4):8–19, 2010.

[26] Michael Ferdman, Almutaz Adileh, Onur Kocberber, Stavros Volos, Mohammad Alisafaee, Djordje Jevdjic, Cansu Kaynak, Adrian Daniel Popescu, Anastasia Ailamaki και Babak Falsafi. *Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. SIGPLAN Not.*, 47(4):37–48, 2012.

[27] Svilen Kanev, Juan Pablo Darago, Kim Hazelwood, Parthasarathy Ranganathan, Tipp Moseley, Gu Yeon Wei και David Brooks. *Profiling a Warehouse-Scale Computer. SIGARCH Comput. Archit. News*, 43(3Σ):158–169, 2015.

[28] Santhosh Verma και David Koppelman. *The interaction and relative effectiveness of hardware and software data prefetch. Journal of Circuits, Systems and Computers*, 21, 2012.

[29] P. Emma, A. Hartstein, T. Puzak και V. Srinivasan. *Exploring the limits of prefetching. IBM J. Res. Dev.*, 49:127–144, 2005.

[30] Huaiyu Zhu, Yong Chen και Xian He Sun. *Timing local streams: Improving timeliness in data prefetching.* σελίδες 169–178, 2010.

[31] Huaiyu Zhu, Yong Chen και Xian He Sun. *Timing Local Streams: Improving Timeliness in Data Prefetching. Proceedings of the 24th ACM International Conference on Supercomputing*, ICS '10, σελίδα 169–178, New York, NY, USA, 2010. Association for Computing Machinery.

[32] Christos Kozyrakis Grant Ayers, Heiner Litz και Parthasarathy Ranganathan. *Classifying Memory Access Patterns for Prefetching. In Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '20)*, Lausanne, Switzerland. ACM, New York, NY, USA, 2020.

[33] J.D. Gindele. *Buffer Block Prefetching Method. IBM Technical Disclosure Bulletin*, 20:696–697, 1977.

[34] Norman P. Jouppi. *Improving Direct-Mapped Cache Performance by the Addition of a Small Fully-Associative Cache and Prefetch Buffers. Proceedings of the 17th Annual*

International Symposium on Computer Architecture, ISCA '90, σελίδα 364–373, New York, NY, USA, 1990. Association for Computing Machinery.

[35] S. Palacharla και R. E. Kessler. *Evaluating Stream Buffers as a Secondary Cache Replacement.* ISCA '94, σελίδα 24–33, Washington, DC, USA, 1994. IEEE Computer Society Press.

[36] Sorin Iacobovici, Lawrence Spracklen, Sudarshan Kadambi, Yuan Chou και Santosh G. Abraham. *Effective Stream-Based and Execution-Based Data Prefetching. Proceedings of the 18th Annual International Conference on Supercomputing*, ICS '04, σελίδα 1–11, New York, NY, USA, 2004. Association for Computing Machinery.

[37] Taesu Kim, Dali Zhao και Alexander V. Veidenbaum. *Multiple Stream Tracker: A New Hardware Stride Prefetcher. Proceedings of the 11th ACM Conference on Computing Frontiers*, CF '14, New York, NY, USA, 2014. Association for Computing Machinery.

[38] Z. Hu, M. Martonosi και S. Kaxiras. *TCP: tag correlating prefetchers. The Ninth International Symposium on High-Performance Computer Architecture, 2003. HPCA-9 2003. Proceedings.*, σελίδες 317–326, 2003.

[39] Saurabh Sharma, Jesse G. Beu και Thomas M. Conte. *Spectral Prefetcher: An Effective Mechanism for L2 Cache Prefetching. ACM Trans. Archit. Code Optim.*, 2(4):423–450, 2005.

[40] Steve VanderWiel και David J Lilja. *A survey of data prefetching techniques. Procs. of the 23rd International Symposium on Computer Architecture.* Citeseer, 1996.

[41] M. Annavaram, J. M. Patel και E. S. Davidson. *Data prefetching by dependence graph precomputation. Proceedings 28th Annual International Symposium on Computer Architecture*, σελίδες 52–61, 2001.

[42] Chi-Keung Luk. *Tolerating memory latency through software-controlled pre-execution in simultaneous multithreading processors. Proceedings 28th Annual International Symposium on Computer Architecture*, σελίδες 40–51, 2001.

[43] J. D. Collins, Hong Wang, D. M. Tullsen, C. Hughes, Yong-Fong Lee, D. Lavery και J. P. Shen. *Speculative precomputation: long-range prefetching of delinquent loads. Proceedings 28th Annual International Symposium on Computer Architecture*, σελίδες 14–25, 2001.

[44] T. Mowry και A. Gupta. *Tolerating Latency through Software-controlled Prefetching in Shared-memory Multiprocessors. Journal of Parallel and Distributed Computing*, 12(2):87–106, 1991.

[45] Glenn Reinman, Brad Calder και Todd Austin. *Fetch Directed Instruction Prefetching. Proceedings of the 32nd Annual ACM/IEEE International Symposium on Microarchitecture*, MICRO 32, σελίδα 16–27, USA, 1999. IEEE Computer Society.

[46] S. Mittal και J. S. Vetter. *A Survey of Techniques for Modeling and Improving Reliability of Computing Systems. IEEE Transactions on Parallel and Distributed Systems*, 27(4):1226–1238, 2016.

[47] Yoshua Bengio Ian Goodfellow και Aaron Courville. *Deep Learning*. The MIT Press, Cambridge, MA, USA, 2016.

[48] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever και Ruslan Salakhutdinov. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[49] F. A. Gers και J. Schmidhuber. *Recurrent nets that time and count. Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, τόμος 3, σελίδες 189–194 ολ.3, 2000.

[50] Kyunghyun Cho, Bartvan Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk και Yoshua Bengio. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. CoRR*, αβς/1406.1078, 2014.

[51] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo και Yoshua Bengio. *Light Gated Recurrent Units for Speech Recognition. IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, 2018.

[52] Yuanhang Su, Yuzhong Huang και C.-C. Jay Kuo. *On Extended Long Short-term Memory and Dependent Bidirectional Recurrent Neural Network. CoRR*, αβς/1803.01686, 2018.

[53] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho και Yoshua Bengio. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. CoRR*, αβς/1412.3555, 2014.

[54] Gruber N και Jockisch. *Are GRU Cells More Specific and LSTM Cells More Sensitive in Motive Classification of Text? Front. Artif. Intell*, . 3:40. δοι: 10.3389/φραι.2020.00040, 2020.

[55] Ilya Sutskever, Oriol Vinyals και Quoc V. Le. *Sequence to Sequence Learning with Neural Networks. CoRR*, αβς/1409.3215, 2014.

[56] Dzmitry Bahdanau, Kyunghyun Cho και Y. Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate. ArXiv*, 1409, 2014.

[57] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel και Yoshua Bengio. *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. CoRR*, αβς/1502.03044, 2015.

[58] Jacob Devlin, Ming-Wei Chang, Kenton Lee και Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. CoRR*, αβς/1810.04805, 2018.

[59] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit και Neil Houlsby. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, 2020.

[60] Aikaterini Karali. *Data Prefetching in Cache Using Machine Learning Models*. `https://github.com/Catherinekarali/champsim_NN_prefetching_extension`, 2021.

[61] *SPEC CPU*. `https://www.spec.org/cpu2017/`, 2017.

[62] Pierre Michaud. *Best-offset hardware prefetching. 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, σελίδες 469–480, 2016.