ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Προανάκληση Δεδομένων στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης με Χρήση Μοντέλων Μηχανικής Μάθησης

Διπλωματική Εργασία
του
Βικέν Γεζεκελιάν

Επιβλέπων: Διονύσιος Ν. Πνευματικάτος
Καθηγητής Ε.Μ.Π.

Εργαστήριο Υπολογιστικών Συστημάτων
Αθήνα, Οκτώβριος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Προανάκληση Δεδομένων στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης με Χρήση Μοντέλων Μηχανικής Μάθησης

Διπλωματική Εργασία
του
Βικέν Γεζεκελιάν

Επιβλέπων: Διονύσιος Ν. Πνευματικάτος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 20η Οκτωβρίου 2022

(Υπογραφή)	(Υπογραφή)	(Υπογραφή)
.....	.....	.....
Διονύσιος Πνευματικάτος	Νεκτάριος Κοζύρης	Βασίλειος Καρακώστας
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Επικ. Καθηγητής Ε.Κ.Π.Α.

Αθήνα, Οκτώβριος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

(Υπογραφή)

.....
Γεζεκελιάν Βικέν
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Η χρήση τεχνικών μηχανικής μάθησης, και συγκεκριμένα τα νευρωνικά δίκτυα, έχουν οδηγήσει σε πρόοδο σε διάφορους τομείς τα τελευταία χρόνια, ενώ έχουν ενθαρρύνει και τη χρήση ηλεκτρονικών υπολογιστών σε πολλά πεδία. Ένας από τους τομείς αυτούς, είναι αυτός της αρχιτεκτονικής υπολογιστών, όπου έχουν υπάρξει ήδη μελέτες σχετικά με την αντικατάσταση ορισμένων μηχανισμών του επεξεργαστή με νευρωνικά δίκτυα, όπως είναι ο προβλεπτής διακλαδώσεων ή προανακλητές στην κρυφή μνήμη του επεξεργαστή. Συγχρόνως, πρόσφατες εργασίες έχουν δείξει πως οι πολυάριθμες αστοχίες της Κρυφής Μνήμης Αναζήτησης Μετάφρασης(TLB) αποτελούν σημαντικό κώλυμα στην προσπάθεια μεγιστοποίησης της απόδοσης των επεξεργαστών. Καθώς η διαχείριση μνήμης σε πολλά συστήματα εκτελείται πλέον με χρήση Εικονικής Μνήμης, η ανάγκη έγκαιρης διάθεσης των απαραίτητων μεταφράσεων εικονικών διευθύνσεων σε φυσικές είναι υψίστης σημασίας για τη διατήρηση υψηλής απόδοσης. Καθώς οι απαιτήσεις και το πλήθος των προγραμμάτων αυξάνονται, με φυσικούς περιορισμούς να αποτρέπουν την αύξηση του μεγέθους της TLB, οι αστοχίες σε αυτή συνεχίζουν να πληθαίνουν οδηγώντας σε επιβάρυνση του συστήματος. Μια από τις σημαντικότερες τεχνικές αντιμετώπισης του προβλήματος αυτού αποτελεί η προανάκληση μεταφράσεων στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης (TLB Prefetching), δηλαδή η φόρτωση δεδομένων σε αυτή από πιο αργά προσπελάσιμες μνήμες, προτού αυτά ζητηθούν. Παρόλα αυτά, έως τώρα δεν έχει υπάρξει κάποια καθολική λύση στο ζήτημα της προανάκλησης μεταφράσεων, με την απόδοση κάθε προανακλητή να κυμαίνεται αισθητά αναλόγως με το πρόγραμμα που χρησιμοποιείται.

Στόχος της παρούσας διπλωματικής εργασίας είναι η αξιολόγηση της χρήσης νευρωνικών δικτύων για την εκτέλεση προανακλήσεων στην Κρυφή Μνήμη Αναζήτησης ΜετάφρασηςTLB του επεξεργαστή. Για τον σκοπό αυτό χρησιμοποιείται μια τροποποιημένη έκδοση του νευρωνικού δικτύου TransFetch, το οποίο στην αρχική του μορφή εκτελεί προανάκληση στην τελευταίου επιπέδου κρυφή μνήμη του επεξεργαστή. Το δίκτυο βασίζεται στον μηχανισμό προσοχής (Attention) για την πρόβλεψη μελλοντικών προσβάσεων στη μνήμη, μέσω ταξινόμησης πολλαπλών ετικετών. Τα δεδομένα εισόδου αρχικά μορφοποιούνται στη μορφή πινάκων που περιλαμβάνουν το ιστορικό προσβάσεων για κάθε αίτημα, ώστε να τροφοδοτηθούν σε ένα Χωρικό Συνελικτικό Δίκτυο (TCN). Στη συνέχεια χρησιμοποιούνται στρώματα Transformer για την πρόβλεψη των πιθανότερων μελλοντικών προσβάσεων. Αφού εκπαιδεύτηκαν μοντέλα για την εύρεση των υπερπαραμέτρων που οδηγούν στη βέλτιστη απόδοση, η δική μας ανάλυση βασίζεται στην αξιολόγηση της σημασίας της έννοιας του χρονισμού κατά την εκτέλεση προανακλήσεων στην TLB, στην αξιολόγηση διαφορετικών μεθόδων εκπαίδευσης για το μοντέλο, καθώς και την εκπαίδευση ενός δικτύου για προανάκληση σε σύνολο εφαρμογών. Τα δεδομένα εκπαίδευσης αποτελούνται από το πλήρες ιστορικό προσβάσεων εφαρμογών και αντλήθηκαν από ίχνη που ανήκουν στις σουίτες GAP, SPEC2006 και SPEC 2017, μέσω του προσομοιωτή ChampSim. Η αξιολόγηση της απόδοσης έγινε επίσης μέσω του προσομοιωτή, με σύγκριση με την εκτέλεση δίχως χρήση προανακλητή, καθώς και με τη χρήση ορισμένων "κλασσικών" προανακλητών στην TLB. Με βάση τα αποτελέσματα της ανάλυσής μας, η χρήση τεχνητών νευρωνικών δικτύων φαίνεται να αποτελεί μια ελκυστική λύση στο ζήτημα της προανάκλησης μεταφράσεων. Βεβαίως, μελλοντικά θα ήταν θεμιτές ορισμένες αλλαγές στον τρόπο λειτουργίας του μοντέλου, ώστε να προσομοιωθεί η χρήση σε πραγματικό σύστημα, καθώς στην παρούσα εργασία έχουν γίνει ορισμένες παραδοχές που θα αναλυθούν παρακάτω.

Λέξεις Κλειδιά

Αρχιτεκτονική Υπολογιστών, Εικονική Μνήμη, Μηχανική Μάθηση, Τεχνητά Νευρωνικά Δίκτυα, Κρυφή Μνήμη Αναζήτησης Μετάφρασης, Προανάκληση Μεταφράσεων, Transformer, Μηχανισμός Προσοχής, ChampSim

Abstract

In the past few years, Machine Learning, and specifically artificial neural networks, have led to breakthroughs in various fields, while simultaneously encouraging the usage of computer systems in those fields. One of said fields is computer architecture, where various studies have attempted to replace certain aspects of the processor via artificial neural networks. Such aspects include the CPUs branch prediction unit, as well as prefetchers to the CPUs caches. At the same time, multiple studies have shown that Translation Lookaside Buffer misses can prove to be a significant hindrance when trying to maximize the performance of computing systems. As memory management of systems is nowadays mostly implemented through the usage of Virtual Memory, the need for timely available translations of virtual to physical addresses is of the utmost importance. As the number of programs and their memory requirements continue to increase, with physical constraints barring an increase in TLB size, the number of TLB misses continues to increase, negatively affecting overall system performance. One of the most used ways to alleviate this problem, is TLB prefetching, i.e., fetching and inserting address translations into the TLB, from slowly accessed types of memory. Nonetheless, there appears to be no universal solution to prefetching, with the performance of prefetchers varying depending on the program they are used on.

In this Diploma Thesis, we attempt to evaluate usage of artificial neural networks in TLB Prefetching. In this context, we use a modified version of TransFetch, which is a neural network modeled to work as a Last Level Cache Prefetcher. The network is based on the Attention Mechanism to identify and predict future accesses, through multiple label classification. Input data is first transformed into arrays, containing the access history of each translation request, and is then fed into a Temporal Convolutional Network. Transformer layers are then used to predict the most probable future accesses. After training networks to find model parameters leading to highest performance, our evaluation is based on the assessment of the concept of prefetching timeliness, the assessment of different methods of network training and the assessment of using a single network as a prefetcher for multiple programs. Training data used is derived from the complete access history of programs and was generated from traces of GAP, SPEC2006 and SPEC2017 suites, using the ChampSim Simulator. Performance evaluation was also completed using the simulator, through comparison of executions using no TLB Prefetcher, and some "classic" TLB prefetching schemes. According to the results, using artificial neural networks appears to be a promising solution in translation prefetching. In the future, changes to the way the model is trained and used would be necessary in order to emulate usage in a real system.

Keywords

Computer Architecture, Virtual Memory, Machine Learning, Artificial Neural Networks, Translation Lookaside Buffer, Address Translation Prefetching, Transformer, Attention Mechanism, ChampSim

Ευχαριστίες

Για την πραγματοποίηση της παρούσας Διπλωματικής Εργασίας θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ.Διονύσιο Πνευματικάτο για την ευκαιρία που μου έδωσε να την εκπονήσω στον τομέα Τεχνολογίας Πληροφορικής και Υπολογιστών (CSLab), καθώς και για την επίβλεψή της. Επίσης, ευχαριστώ ιδιαίτερα τον Δρ. Βασίλειο Καρακώστα για την καθοδήγηση που μου προσέφερε και την εξαιρετική συνεργασία που είχαμε κατά τη διάρκεια εκπόνησης της εργασίας. Θα ήθελα επίσης να ευχαριστήσω και τον Γιώργο Βαβουλιώτη για την πολύτιμη βοήθεια που μου προσέφερε. Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου και την αδερφή μου για την καθοδήγηση και την ηθική συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Γεζεκελιάν Βικέν

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Τα τελευταία χρόνια, ο τομέας της Μηχανικής Μάθησης αναπτύσσεται με ραγδαίους ρυθμούς προσφέροντας λύσεις και νέους τρόπους αξιοποίησης των νευρωνικών δικτύων σε πληθώρα διαφορετικών κλάδων. Η αύξηση των διαθέσιμων υπολογιστικών πόρων των συστημάτων σε συνδυασμό με νέα και διαφορετικά μοντέλα μηχανικής μάθησης ωθούν την ταχύτατη εξέλιξη του τομέα, αφού διευκολύνουν την εκπαίδευση και χρήση των νευρωνικών δικτύων για την αντιμετώπιση ολοένα και πιο σύνθετων και ποικίλων προβλημάτων. Παράλληλα, στον κλάδο της αρχιτεκτονικής υπολογιστών παρουσιάζονται δυσκολίες αύξησης της υπολογιστικής ισχύος, με την ισχύ του νόμου του Moore να πλησιάζει στο τέλος της λόγω φυσικών περιορισμών[1]. Οι δύο αυτές τάσεις, έχουν οδηγήσει στη μελέτη ώστε να βρεθούν τρόποι αξιοποίησης μοντέλων Μηχανικής Μάθησης στην αρχιτεκτονική των υπολογιστικών συστημάτων. Στη συγκεκριμένη διπλωματική εργασία, εξετάζεται η χρήση ενός Τεχνητού Νευρωνικού Δικτύου για την προανάκληση δεδομένων στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης του επεξεργαστή.

1.1 Κίνητρο

Το σύνολο των σύγχρονων λειτουργικών συστημάτων αξιοποιεί την Εικονική Μνήμη ως τεχνική διαχείρισης της μνήμης. Η τεχνική αυτή αντιστοιχεί το σύνολο των διαθέσιμων φυσικών διευθύνσεων σε εικονικές, παρέχοντας με τον τρόπο αυτό στον προγραμματιστή την ψευδαίσθηση ύπαρξης απεριόριστης διαθέσιμης μνήμης. Παράλληλα, οι εφαρμογές ενός συστήματος δύνανται να εκτελούνται απομονωμένες η μία από την άλλη, γεγονός που αυξάνει την ασφάλεια και βελτιώνει τον τρόπο αξιοποίησης της διαθέσιμης φυσικής μνήμης. Παρά τα θετικά στοιχεία που προσφέρει, η χρήση Εικονικής Μνήμης σε ένα σύστημα συνεπάγεται και την εισαγωγή μιας διεπαφής, ονομαζόμενη Πίνακας Σελίδων, υπεύθυνης για την απεικόνιση μεταξύ Φυσικών και Εικονικών διευθύνσεων. Ο πίνακας αυτός περιλαμβάνει το σύνολο των μεταφράσεων για κάθε εφαρμογή, με κάθε αίτημα πρόσβασης στη μνήμη να συνοδεύεται και από μια δαπανηρή και πολλών σταδίων αναζήτηση σε αυτόν[2, 3]. Για την αντιμετώπιση του κόστους των αναζητήσεων αυτών, έχει εισαχθεί μια επιπλέον δομή σε σύγχρονους επεξεργαστές, ονομαζόμενη Κρυφή Μνήμη Αναζήτησης Μετάφρασης (Translation Lookaside-Buffer), η οποία λειτουργεί ως μια κρυφή μνήμη του Πίνακα Σελίδων.

Οι απαιτήσεις μνήμης των σύγχρονων εφαρμογών αυξάνονται συνεχώς, δίχως να είναι δυνατή η αντίστοιχη αύξηση του μεγέθους της Κρυφής Μνήμης Αναζήτησης Μετάφρασης των συστημάτων. Ο λόγος έγκειται στη συχνή ανάγκη πρόσβασης της, καθώς κάθε λειτουργία μνήμης απαιτεί πρόσβαση σε αυτή, καθώς και στους φυσικούς περιορισμούς της υλοποίησής της. Χάρις αυτών, μια απόπειρα αύξησης του μεγέθους της θα οδηγούσε σε υψηλές καθυστερήσεις και σημαντικές αυξήσεις στην ενέργεια που καταναλώνει το σύστημα[4, 5]. Η ασυμφωνία αυτή οδηγεί εφαρμογές με υψηλές απαιτήσεις μνήμης και μη βοηθητική τοπικότητα αιτημάτων στη δαπάνη ενός μεγάλου μέρους του χρόνου εκτέλεσης τους σε άεργους κύκλους, καθώς αναμένουν τις απεικονίσεις των εικονικών διευθύνσεων ώστε να αποκτήσουν πρόσβαση στα απαραίτητα δεδομένα. Η Προανάκληση, δηλαδή η φόρτωση δεδομένων στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης από βραδύτερα προσπελάσιμες μνήμες προτού ζητηθούν αποτελεί μια προσέγγιση για την αντιμετώπιση του προβλήματος αυτού.

Οι παραδοσιακές μέθοδοι προανάκλησης ποικίλουν στη λειτουργία και την πολυπλοκότητά τους, και μπορεί να χρησιμοποιούν από απλούς ευριστικούς κανόνες με χαμηλό αποτύπωμα μνήμης έως και σύνθετους αλγορίθμους

για την πρόβλεψη της κατάλληλης καταχώρησης. Στις μεθόδους αυτές, η ανταπόκριση σε μεγάλο εύρος εφαρμογών με πολύπλοκες αλληλουχίες προσβάσεων στη μνήμη έρχεται συνήθως με τη χρήση περισσότερων πόρων μνήμης και αυξημένων υπολογιστικών απαιτήσεων. Παράλληλα, η λειτουργία των μεθόδων αυτών παραμένει σταθερή ανεξαρτήτως της εφαρμογής στην οποία χρησιμοποιούνται, γεγονός που πολλές φορές οδηγεί σε πολλαπλές αλλεπάλληλες ζημιογόνες προανακλήσεις που μολύνουν τη μνήμη και οδηγούν στη μείωση της απόδοσης. Για τους λόγους αυτούς, πρόσφατες έρευνες αντιμετωπίζουν την προανάκληση στις κρυφές μνήμες του επεξεργαστή, με τη χρήση νευρωνικών δικτύων.

1.2 Συναφείς προσεγγίσεις

Στο παρελθόν, έχουν υπάρξει απόπειρες χρήσης της μηχανικής μάθησης για τη βελτίωση της λειτουργίας διάφορων στοιχείων της αρχιτεκτονικής υπολογιστικών συστημάτων. Μια από τις πρώτες εξ αυτών αποτέλεσε η χρήση ενός Perceptron για την πρόβλεψη διακλαδώσεων[6], με πολλές παρόμοιες δουλειές να ακολουθούν. Ο προβλεπτής διακλαδώσεων αποτελεί μια από τις σημαντικότερες μονάδες του επεξεργαστή, καθώς τεχνικές όπως η σωλήνωση και η εκτέλεση εντολών εκτός σειράς που χρησιμοποιούνται σε υπολογιστικά συστήματα εξαρτώνται και επηρεάζονται από τις αποφάσεις του προβλεπτή, με πιθανές αστοχίες του να επιφέρουν αισθητή επιβράδυνση της εκτέλεσης. Παρά το γεγονός πως σύγχρονοι προβλεπτές επιτυγχάνουν ακρίβεια της τάξεως του 99%, προβλεπτές που κάνουν χρήση νευρωνικών δικτύων, είναι ικανοί να επιταχύνουν αισθητά την εκτέλεση εφαρμογών μέσω της πρόβλεψης συγκεκριμένων "δύσκολων-να-προβλεφθούν" διακλαδώσεων[7], δίχως να δαπανούν περιττούς πόρους[8]. Πλέον, οι προβλεπτές διακλαδώσεων που κάνουν χρήση νευρωνικών δικτύων είναι ικανοί να προσφέρουν σημαντική βελτίωση στην απόδοση υπολογιστικών συστημάτων, ξεπερνώντας τις επιδόσεις των σύγχρονων προβλεπτών.

Ένα ακόμη σημαντικό ζήτημα στην αρχιτεκτονική υπολογιστών, αποτελεί η δημιουργία μιας καθολικής μεθόδους προανάκλησης δεδομένων, η οποία θα είναι ικανή να αποδώσει σε όλο το εύρος των εφαρμογών. Ειδικότερα για τις κρυφές μνήμες του επεξεργαστή, έχουν υπάρξει πολλές μελέτες για τη δημιουργία ενός προανακλητή, η λειτουργία του οποίου θα βασίζεται σε κάποιο νευρωνικό δίκτυο. Οι προσεγγίσεις, τα μοντέλα που χρησιμοποιούνται καθώς και ο τρόπος αντιμετώπισης του προβλήματος ποικίλλουν. Συγκεκριμένα, ενώ το ζήτημα μπορεί να αντιμετωπιστεί και ως πρόβλημα παλινδρόμησης [9], σύγχρονες μελέτες αντιμετωπίζουν την προανάκληση συσχετίζοντας τη με προβλήματα φυσικής γλώσσας, όπου μια αριθμημένη ακολουθία δεδομένων εισόδου χρησιμοποιείται για την πρόβλεψη της επόμενης λέξης, ή στην περίπτωση που μας απασχολεί πρόσβασης [10, 11]. Στο ίδιο πλαίσιο, εφόσον απαιτούνται μοντέλα ικανά να επεξεργαστούν ακολουθιακά δεδομένα εισόδου, χρησιμοποιούνται συνήθως αναδρομικά νευρωνικά δίκτυα, καθώς και επεκτάσεις αυτών, όπως είναι τα δίκτυα Μακράς-Βραχυπρόθεσμης Μνήμης (LSTM) [12, 13]. Ακόμη, ένας μηχανισμός ο οποίος επιστρατεύεται συχνά σε προανακλητές που βασίζονται σε νευρωνικά δίκτυα, είναι ο μηχανισμός της προσοχής. Ο μηχανισμός αυτός επιτρέπει στο δίκτυο να αποφασίσει ποια κομμάτια των δεδομένων εισόδου είναι σημαντικότερα, και κατ΄ επέκταση να εκτελέσει τις μελλοντικές του προβλέψεις με βάση αυτά. Ο μηχανισμός αυτός είναι ικανός να ξεπεράσει πολλά από τα προβλήματα που παρουσιάζουν τα αναδρομικά νευρωνικά δίκτυα, βελτιώνοντας ταυτόχρονα την αποδοτικότητα και την ακρίβειά τους.

1.3 Αντικείμενο διπλωματικής

Σε αυτή τη διπλωματική εργασία, βασιζόμαστε στη λειτουργία του μοντέλου TransFetch[14], το οποίο έχουμε προσαρμόσει ώστε να λειτουργήσει ως προανακλητής στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης.

Η προανάκληση αντιμετωπίζεται ως πρόβλημα ταξινόμησης με βάση το Δέλτα, δηλαδή την απόσταση των διευθύνσεων διαδοχικών προσβάσεων στη μνήμη. Παράλληλα, κατά τη δημιουργία προβλέψεων λαμβάνεται υπόψιν και ο Μετρητής Προγράμματος με σκοπό τη δημιουργία συσχετίσεων μεταξύ μελλοντικών διευθύνσεων και του τωρινού Μετρητή. Το μοντέλο βασίζεται στο συνδυασμό ενός Χωρικού Νευρωνικού Δικτύου και του μηχανισμού Προσοχής Πολλών Κεφαλών, ενώ χρησιμοποιεί ως δεδομένα εκπαίδευσης το ιστορικό προσβάσεων εφαρμογών. Η εκτέλεση πειραμάτων εκτελείται με τη βοήθεια του προσομοιωτή ChampSim, ενώ η Κρυφή Μνήμη Αναζήτησης που χρησιμοποιείται είναι δύο επιπέδων, με το πρώτο επίπεδο να είναι διαχωρισμένο μεταξύ δεδομένων και

εντολών. Το δίκτυο εκπαιδεύεται offline, δηλαδή δίχως σύνδεση στον προσομοιωτή, ώστε να παραγάγει το επιθυμητό αρχείο προανακλήσεων. Οι εφαρμογές που χρησιμοποιούνται στην παρούσα εργασία προέρχονται από τις σουίτες προγραμμάτων GAP, SPEC2006 και SPEC2017. Τα δεδομένα εκπαίδευσης που έχουν δημιουργηθεί αφορούν το ιστορικό προσβάσεων των εφαρμογών αυτών, ενώ έχει ακολουθήσει ανάλυσή τους με σκοπό την εύρεση των κατάλληλων παραμέτρων για το μοντέλο. Ακόμη, έχουν αξιολογηθεί διαφορετικές μέθοδοι εκπαίδευσης του μοντέλου, ενώ έχει υπάρξει και ανάλυση σχετικά με τη σημασία του χρονισμού εκτέλεσης των προανακλήσεων.

Με βάση τα αποτελέσματα των πειραμάτων, φαίνεται να υπάρχει προοπτική αξιοποίησης μοντέλων μηχανικής μάθησης κατά την προανάκτηση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Παρόλα αυτά, σημειώνεται πως η παρούσα εργασία αποτελεί θεωρητική μελέτη για τις δυνατότητες χρήσης νευρωνικών δικτύων ως προανακλητή. Ως εκ τούτου έχουν υπάρξει πολλές παραδοχές, ενώ αγνοούνται πλήρως οι καθυστερήσεις πρόβλεψης του νευρωνικού δικτύου που θα προέκυπταν κατά την εκτέλεση σε πραγματικό χρόνο.

1.4 Διάρθρωση Εργασίας

Στο κεφάλαιο 2, παρουσιάζονται πληροφορίες σχετικά με την Εικονική Μνήμη, την απαραίτητη αρχιτεκτονική υποστήριξη για την εύρυθμη λειτουργία της εντός ενός συστήματος, καθώς και ορισμένους μηχανισμούς προανάκλησης στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης που χρησιμοποιήθηκαν παρακάτω. Το Κεφάλαιο 3 αναφέρεται στη μηχανική μάθηση, και συγκεκριμένα στον τρόπο με τον οποίο εκπαιδεύονται και λειτουργούν ορισμένα είδη Τεχνητών Νευρωνικών Δικτύων. Στο κεφάλαιο 4 παρουσιάζεται το σύνολο της μεθοδολογίας που ακολουθήθηκε κατά την ολοκλήρωση της συγκεκριμένης εργασίας. Αρχικά γίνεται αναφορά στον προσομοιωτή ChampSim, το βασικότερο εργαλείο της παρούσας εργασίας, ενώ στη συνέχεια παρουσιάζεται η διαδικασία που ακολουθήθηκε για τη δημιουργία και εκπαίδευση μιας τροποποιημένης εκδοχής του νευρωνικού δικτύου TransFetch [14], που λειτουργεί ως προανακλητής στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Τα αποτελέσματα των προσομοιώσεων που εκτελέστηκαν παρουσιάζονται και σχολιάζονται στο Κεφάλαιο 5. Τέλος, στο Κεφάλαιο 6 βρίσκεται ο επίλογος της διπλωματικής, σε συνδυασμό με ορισμένες πιθανές μελλοντικές επεκτάσεις.

Κεφάλαιο 2

Η Προανάκληση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης

2.1 Εικονική Μνήμη

Με τον όρο Εικονική Μνήμη (Virtual Memory) αναφερόμαστε στην τεχνική διαχείρισης μνήμης κατά την οποία η κύρια μνήμη (Dynamic Random-Access Memory) παρομοιάζεται με ενός είδους κρυφή μνήμη (cache) για τη, σαφώς πιο αργή, δευτερεύουσα μνήμη, η οποία υλοποιείται συνήθως με σκληρούς δίσκους (Hard Disk Drive) ή δίσκους στερεάς κατάστασης (Solid State Drive). Με την εικονική μνήμη, το λειτουργικό σύστημα εκμεταλλευόμενο τόσο το λογισμικό όσο και το υλικό, αντιστοιχίζει τις διευθύνσεις που χρησιμοποιεί κάποιο πρόγραμμα, ονομαζόμενες εικονικές διευθύνσεις (Virtual Addresses), σε πραγματικές διευθύνσεις εντός της κύριας μνήμης, ονομαζόμενες φυσικές διευθύνσεις (Physical Addresses)[15]. Η διαχείριση της εικονικής μνήμης είναι δυνατή με δύο μεθόδους: τη Σελιδοποίηση (Paging) και την Τμηματοποίηση (Segmentation). Στη συγκεκριμένη διπλωματική εργασία, έχει χρησιμοποιηθεί σύγχρονη αρχιτεκτονική που υποστηρίζει τη διαχείριση εικονικής μνήμης μέσω σελιδοποίησης και η οποία θα αναπτυχθεί περαιτέρω παρακάτω[16, 17].

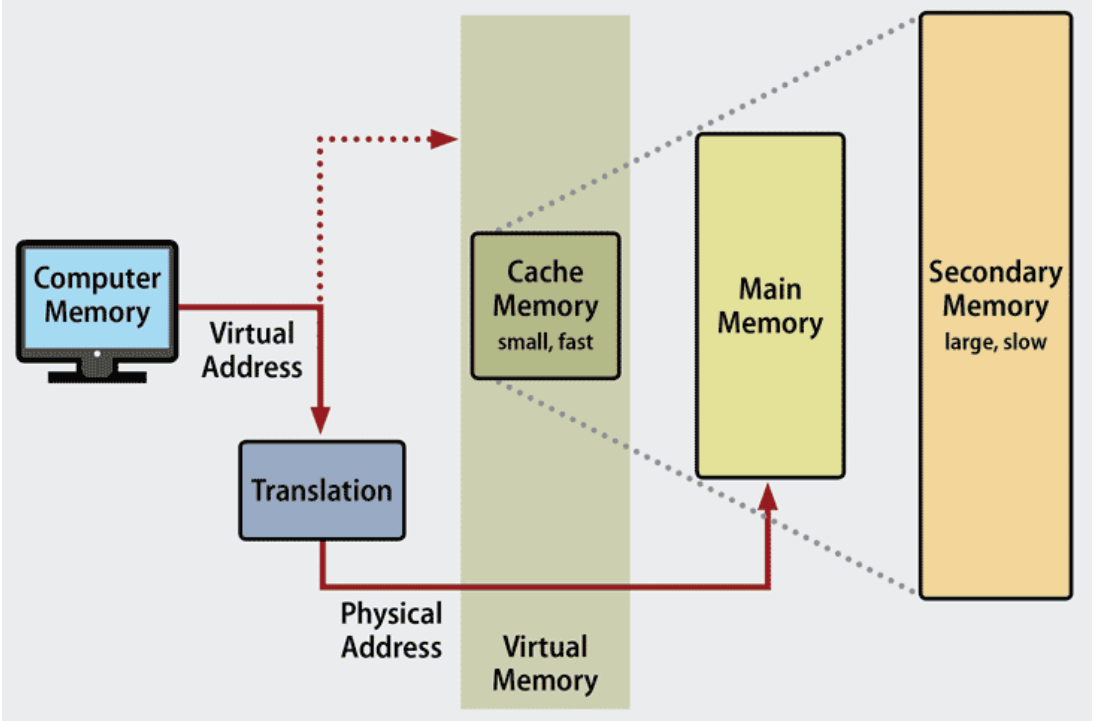
Τα κύρια πλεονεκτήματα που προσφέρει η εικονική μνήμη βρίσκονται στην ασφαλή κοινή χρήση της κύριας μνήμης μεταξύ πολλαπλών προγραμμάτων καθώς και στη χρήση ενός εικονικού χώρου διευθύνσεων μεγαλύτερης χωρητικότητας από της φυσικά διαθέσιμης μνήμης. Στο παρελθόν, ο μερισμός της κύριας μνήμης μεταξύ προγραμμάτων ενείχε πολλούς κινδύνους, λόγω της αδυναμίας του προγραμματιστή να γνωρίζει εκ των προτέρων το πλήθος και τις απαιτήσεις των διεργασιών που εκτελούνται ταυτόχρονα. Μέσω της εικονικής μνήμης, το κάθε πρόγραμμα εκτελείται πρακτικά στον δικό του χώρο διευθύνσεων, εξαλείφοντας το πρόβλημα αυτό. Παράλληλα, ο προγραμματιστής δεν χρειάζεται να ασχοληθεί με το μέγεθος που θα καταλαμβάνει το κάθε πρόγραμμα εντός της κύριας μνήμης, καθώς πλέον η διαχείριση της ιεραρχίας της μνήμης γίνεται αυτόματα μέσω του λειτουργικού συστήματος.

2.1.1 Χώρος Εικονικών Διευθύνσεων

Ως χώρο εικονικών διευθύνσεων ενός προγράμματος, ονομάζουμε το σύνολο των εικονικών διευθύνσεων που έχουν παραχθεί από το λειτουργικό σύστημα και έχουν αντιστοιχηθεί στο πρόγραμμα αυτό. Το λειτουργικό σύστημα είναι υπεύθυνο για τη διαχείριση του συνόλου των χώρων εικονικών διευθύνσεων, καθώς και την ανάθεση της μνήμης σε εικονικές διευθύνσεις.

2.1.2 Χώρος Φυσικών Διευθύνσεων

Με τον όρο χώρος φυσικών διευθύνσεων αναφερόμαστε στο σύνολο των θέσεων μνήμης ενός υπολογιστικού συστήματος, δηλαδή στην κύρια μνήμη του. Το λειτουργικό σύστημα είναι υπεύθυνο για τη δέσμευση των υλικών πόρων του συστήματος και τον βέλτιστο διαμοιρασμό τους μεταξύ των ενεργών προγραμμάτων.



Σχήμα 2.1: Ιεραρχία μνήμης διαφορετικών επιπέδων ενός συστήματος

Source: `https://www.enterprisestorageforum.com/hardware/virtual-memory/`

2.1.3 Μετάφραση Εικονικών σε Φυσικές Διευθύνσεις

Κατά την εκτέλεση κάθε προγράμματος, εκτελούνται διάφορα αιτήματα πρόσβασης στη μνήμη. Τα αιτήματα αυτά, αφορούν τις διευθύνσεις στις οποίες έχει πρόσβαση, αυτές που ανήκουν δηλαδή στο χώρο εικονικών διευθύνσεων του εκάστοτε προγράμματος. Όπως είναι λογικό όμως, το πρόγραμμα θα χρειαστεί πρόσβαση στη φυσική διεύθυνση της μνήμης, ώστε να μπορέσει να εκτελέσει την ενέργεια που επιθυμεί. Η μετάφραση (translation), ή αλλιώς απεικόνιση (mapping), διευθύνσεων είναι η διαδικασία κατά την οποία με τη χρήση υλικού και λογισμικού μια εικονική διεύθυνση ενός προγράμματος μεταφράζεται στη φυσική διεύθυνση που μπορεί έπειτα να χρησιμοποιηθεί για την προσπέλαση της μνήμης. Η διαδικασία αυτή δέχεται ως είσοδο την εικονική διεύθυνση στην οποία έχει ζητηθεί πρόσβαση και παράγει στην έξοδό της την φυσική διεύθυνση που έχει αντιστοιχηθεί από το λειτουργικό σύστημα.

2.2 Διαχείριση Εικονικής Μνήμης μέσω Σελιδοποίησης

Όπως αναφέραμε και παραπάνω, στη συγκεκριμένη διπλωματική θα χρησιμοποιηθεί αποκλειστικά η σελιδοποίηση ως μέθοδος διαχείρισης της εικονικής μνήμης.

Η ιεραρχική δομή της εικονικής μνήμης όταν χρησιμοποιείται η σελιδοποίηση είναι παρόμοια με αυτή της κρυφής μνήμης, χρησιμοποιούνται όμως διαφορετικές ονομασίες. Συγκεκριμένα, η εικονική μνήμη αποτελείται από μπλοκ, δηλαδή σύνολα εικονικών διευθύνσεων τα οποία ονομάζονται σελίδες (Pages). Οι σελίδες αυτές δεν είναι απαραίτητο να εκχωρηθούν με συνεχή τρόπο στη μνήμη, εξαλείφοντας την ανάγκη εύρεσης συνεχών τμημάτων μνήμης κατάλληλου μεγέθους για την φόρτωση του κάθε προγράμματος. Παράλληλα, κατά τη δημιουργία κάθε διεργασίας δεσμεύεται χώρος στο δίσκο για όλες τις σελίδες που θα χρειαστεί, ονομαζόμενος χώρος εναλλαγής (Swap Space). Κάθε μία εκ των εικονικών σελίδων αντιστοιχεί σε μια φυσική σελίδα, δηλαδή σε μια σελίδα που περιλαμβάνει φυσικές διευθύνσεις εντός της μνήμης. Κάθε φυσική σελίδα, μπορεί να αντιστοιχηθεί σε πολλαπλές εικονικές σελίδες, ώστε να είναι δυνατή η κοινή χρήση δεδομένων μεταξύ διαφορετικών προγραμμάτων. Οι διευθύνσεις εντός των σελίδων αποτελούνται από ένα αναγνωριστικό σελίδας και τη μετατόπιση σελίδας (Page Offset). Το αναγνωριστικό σελίδας, ονομαζόμενο αριθμός εικονικής σελίδας (Virtual Page Number) ή αριθμός

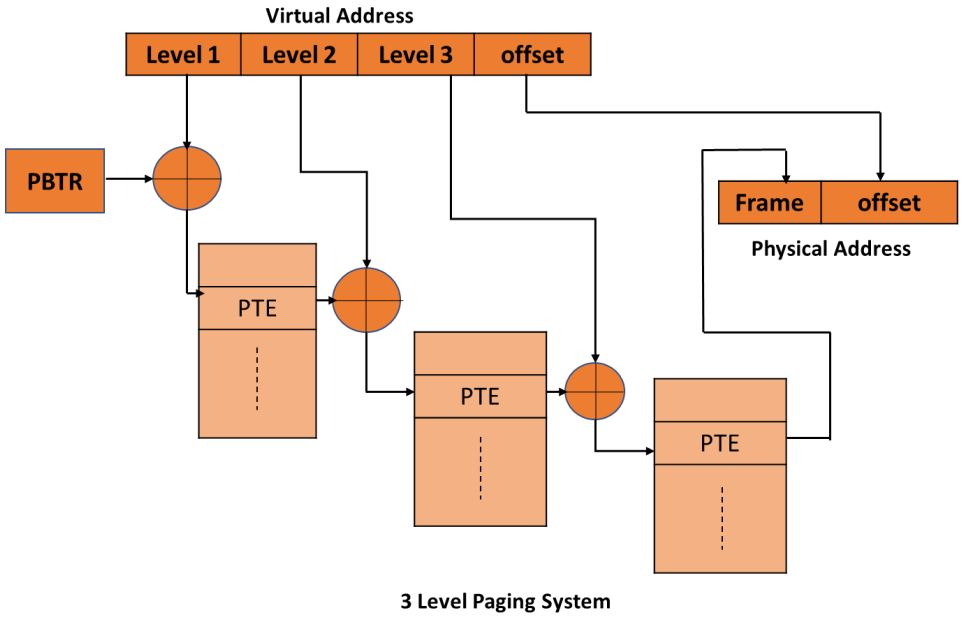
φυσικής σελίδας (Physical Page Number) αναλόγως με τη φύση της εκάστοτε σελίδας, αποτελούν τα πρώτα ψηφία της διεύθυνσης αλλά και το κομμάτι της διεύθυνσης το οποίο μεταφράζεται, καθώς η μετατόπιση εντός της σελίδας μένει σταθερή κατά την απεικόνιση διευθύνσεων. Το πλήθος των ψηφίων που αποτελούν τους αριθμούς εικονικής και φυσικής σελίδας μπορούν να διαφέρουν σε μέγεθος, με τον αριθμό εικονικής σελίδας να είναι συνήθως μεγαλύτερος, ώστε να μπορούν να απεικονιστούν περισσότερες εικονικές διευθύνσεις και να δίνεται στον προγραμματιστή η ψευδαίσθηση ύπαρξης απεριόριστης εικονικής μνήμης.

2.2.1 Πίνακας Σελίδων

Ο πίνακας σελίδων αποτελεί τη δομή δεδομένων που είναι υπεύθυνη για την αποθήκευση απεικονίσεων εικονικών διευθύνσεων σε φυσικές. Αποτελεί κομμάτι του λογισμικού του συστήματος και η διαχείρισή του γίνεται από το λειτουργικό σύστημα, με κάθε εφαρμογή να έχει και τον δικό της πίνακα σελίδων. Οι πίνακες αυτοί αποθηκεύονται στην κύρια μνήμη, με την ακριβή τους θέση να υποδεικνύεται από τον καταχωρητή πίνακα σελίδων (Page Table Base Register) και χρησιμοποιούν ως δείκτη (Index) τον αριθμό εικονικής σελίδας. Οι πίνακες σελίδων μπορούν να οργανωθούν με διαφορετικούς τρόπους, ο καθένας βελτιστοποιημένος για να ανταπεξέλθει σε διαφορετικές απαιτήσεις.

Στην απλούστερη μορφή του, ο Πίνακας Σελίδων αποτελείται από ένα γραμμικό πίνακα που έχει αποθηκευμένες όλες τις απεικονίσεις εικονικών διευθύνσεων σε φυσικές, για κάθε ενεργή εφαρμογή του συστήματος. Η προσέγγιση αυτή, αν και ιδιαίτερα απλή στην υλοποίηση της, δεν είναι ιδιαίτερα αποδοτική σε επίπεδο συστήματος. Το μεγάλο πλήθος διεργασιών ενός συστήματος σε συνδυασμό με το πλήθος των καταχωρήσεων για κάθε διεργασία οδηγεί σε έναν Πίνακα Σελίδων πολύ μεγάλου μεγέθους, με ένα μεγάλο κομμάτι του να παραμένει στην πραγματικότητα κενό, καθώς οι εικονικοί χώροι διευθύνσεων πολλών εφαρμογών είναι ιδιαίτερα αραιοί.

Ο τρόπος οργάνωσης του πίνακα σελίδων στον οποίο θα επικεντρωθεί η συγκεκριμένη διπλωματική, είναι ο Πίνακας Σελίδων Πολλαπλών Επιπέδων ή αλλιώς Ιεραρχικός Τρόπος Οργάνωσης του Πίνακα Σελίδων. Οι Πίνακες Σελίδων Πολλαπλών Επιπέδων είναι δενδρικές δομές που πρακτικά χρησιμοποιούνται για να αποθηκεύσουν Πίνακες Σελίδων, με τα φύλλα του «δέντρου» να αντιστοιχούν σε καταχωρήσεις. Στους Πίνακες Σελίδων Πολλαπλών Επιπέδων οι δείκτες, δηλαδή οι εικονικές διευθύνσεις, σπάνε σε κομμάτια κάθε ένα εκ των οποίων αποτελεί έναν καινούργιο δείκτη για κάθε επίπεδο του δέντρου. Το μέγεθος του κάθε δείκτη αποτελεί σχεδιαστική επιλογή και παραμένει σταθερό για κάθε αρχιτεκτονική. Στους πίνακες αυτούς, κάθε λειτουργία απαιτεί διαδοχικές προσβάσεις μνήμης ίσες σε αριθμό με το πλήθος των επιπέδων του πίνακα. Μια αναπαράσταση ενός Πίνακα Σελίδων τριών επιπέδων φαίνεται στο σχήμα 2.2.



Σχήμα 2.2: Αναπαράσταση Πίνακα Σελίδων 3 Επιπέδων

2.2.1.1 Καταχωρήσεις Πίνακα Σελίδων

Οι πληροφορίες εντός του πίνακα σελίδων αποθηκεύονται υπό τη μορφή καταχωρήσεων (Page Table Entries). Οι καταχωρήσεις αυτές, πέραν της φυσικής διεύθυνσης που αντιστοιχεί στον δείκτη τους, αποθηκεύουν επιπλέον χρήσιμες πληροφορίες, υπό τη μορφή δυαδικών ψηφίων καταστάσεων (Status Bits), για την αύξηση της αποδοτικότητας και ασφάλειας του συστήματος. Ορισμένες από τις πληροφορίες αυτές, που διατηρούνται στους περισσότερους σύγχρονους πίνακες σελίδων, θα παρουσιαστούν εν συντομία παρακάτω.

- *Δυαδικό Ψηφίο Παρουσίας/Απουσίας (Present/Absent Bit):* Χρησιμοποιείται για να δηλώσει την παρουσία ή μη της μετάφρασης, δηλαδή της φυσικής διεύθυνσης που ζητείται στον Πίνακα Σελίδων. Καλείται επίσης και ψηφίο εγκυρότητας (Valid Bit) καθώς στην πραγματικότητα τα ψηφία που αντιστοιχούν στον αριθμό φυσικής σελίδας θα περιέχουν πάντοτε κάποια τιμή, δεν θα είναι όμως πάντα έγκυρη. Στην περίπτωση που ισούται με 0, η διεύθυνση δεν υπάρχει εντός του πίνακα και προκύπτει Σφάλμα Σελίδας, που θα αναλυθεί περαιτέρω σε παρακάτω υποενότητα.
- *Δυαδικά Ψηφία Προστασίας (Protection bits):* Είναι συνήθως δύο σε αριθμό και υποδηλώνουν τα δικαιώματα της διεργασίας αναφορικά με τις λειτουργίες που μπορεί να εκτελέσει στη συγκεκριμένη διεύθυνση. Το πρώτο ονομάζεται ψηφίο διαβάσματος/εγγραφής (Read/Write Bit). Στην περίπτωση που ισούται με 1, ενεργοποιείται το δικαίωμα διαβάσματος και εγγραφής στη μνήμη, ενώ σε περίπτωση που ισούται με 0 επιτρέπεται μόνο το διάβασμα από τη μνήμη. Το δεύτερο ονομάζεται ψηφίο μη εκτέλεσης (No Execute ή NX Bit) και στην περίπτωση που ισούται με 1 αποτρέπει την εκτέλεση των δεδομένων της σελίδας μνήμης.
- *Δυαδικό Ψηφίο Αναφοράς ή Χρήσης (Reference/Use Bit):* Δηλώνει εάν έχει υπάρξει αναφορά στη σελίδα στον τελευταίο κύκλο ρολογιού. Πρακτικά χρησιμοποιείται για την υλοποίηση απλοϊκών μορφών αλγορίθμων αντικατάστασης, όπως είναι ο Least-Recently Used (LRU). Σε ορισμένα συστήματα, χρησιμοποιούνται περισσότερα από ένα δυαδικά ψηφία αναφοράς.
- *Δυαδικό Ψηφίο Κρυφής Μνήμης (Caching Enabled/Disabled Bit):* Χρησιμοποιείται για να ενεργοποιήσει/απενεργοποιήσει την ενημέρωση της κρυφής μνήμης για τη συγκεκριμένη σελίδα. Η λειτουργία αυτή είναι χρήσιμη όταν είναι απαραίτητη η πρόσβαση στις πιο πρόσφατες πληροφορίες, ιδίως όταν προέρχονται από τον χρήστη.
- *Δυαδικό Ψηφίο Τροποποίησης (Modified Bit):* Ονομάζεται συχνά και ακάθαρτο δυαδικό ψηφίο (Dirty Bit) και δηλώνει εάν η σελίδα έχει τροποποιηθεί ή όχι. Εάν το περιεχόμενο έχει τροποποιηθεί, το σύστημα φροντίζει να αποθηκεύσει τις καινούργιες πληροφορίες στο δίσκο σε περίπτωση που υπάρξει αίτημα αντικατάστασης της σελίδας.
- *Δυαδικό Ψηφίο Χρήστη (User Bit):* Καθορίζει ποιες διεργασίες έχουν πρόσβαση στα δεδομένα της σελίδας. Εάν είναι απενεργοποιημένο, η πρόσβαση στη σελίδα είναι δυνατή μόνο από το λειτουργικό σύστημα και όχι σε διεργασίες με δικαιώματα χρήστη.

2.2.1.2 Σφάλμα Σελίδας (Page Fault)

Το Σφάλμα Σελίδας ή αλλιώς Αστοχία Σελίδας (Page Miss) είναι μια διακοπή λογισμικού (Software Interrupt) που προκύπτει σε ορισμένες περιπτώσεις όταν κάποια διεργασία αιτείται πρόσβαση σε κάποια σελίδα. Αναλόγως με τον λόγο για τον οποίο έχει προκύψει σφάλμα σελίδας, ομαδοποιούνται σε μικρά (minor), μεγάλα (major) και μη έγκυρα (Invalid). Τα μικρά σφάλματα σελίδας εγείρονται όταν τα δεδομένα που ζητούνται υπάρχουν στη μνήμη, πιθανώς λόγω της εκτέλεσης κάποιας άλλης εφαρμογής, δεν έχουν σημειωθεί όμως ως διαθέσιμα για τη διεργασία που τα ζητάει. Στην περίπτωση αυτή, ο χειριστής σφαλμάτων μνήμης του λειτουργικού συστήματος αρκεί απλώς να τροποποιήσει την καταχώρηση στον πίνακα σελίδων ώστε να δείχνει στη σωστή διεύθυνση στη μνήμη. Τα μεγάλα σφάλματα σελίδας εγείρονται όταν η πληροφορία δεν υπάρχει στη μνήμη, οπότε πρέπει να φορτωθούν από το δίσκο σε κάποια διαθέσιμη σελίδα, ή στην περίπτωση που αυτό δεν είναι εφικτό στη σελίδα που ορίζει ο αλγόριθμος αντικατάστασης που χρησιμοποιείται, με την ποινή αστοχίας να είναι σημαντικά υψηλότερη. Και στις δύο περιπτώσεις, αφού ολοκληρωθούν οι απαραίτητες ενέργειες η αναφορά εκτελείται εκ νέου. Η τελευταία περίπτωση προκύπτει όταν μια διεργασία προσπαθεί να προσπελάσει κάποια διεύθυνση που δεν αποτελεί κομμάτι του χώρου εικονικών διευθύνσεών της. Αυτό μπορεί να προκύψει σε περιπτώσεις όπου η διεργασία παραβιάζει

τα δικαιώματα πρόσβασης που της έχουν δοθεί και οδηγεί σε σφάλμα κατάτμησης (Segmentation Fault) και τον τερματισμό της εφαρμογής.

2.2.1.3 Διάσχιση Σελίδων (Page Walk)

Η Διάσχιση Σελίδων (Page Walk) είναι η διαδικασία που εκτελείται κατά την αναζήτηση κάποιας απεικόνισης εικονικής διεύθυνσης σε Πίνακες Σελίδων Πολλαπλών Επιπέδων. Κατά την εκτέλεση μιας Διάσχισης Σελίδων, απαιτούνται διαδοχικές προσβάσεις μνήμης στα διάφορα επίπεδα του Πίνακα Σελίδων. Ταυτόχρονα, κατά τη διάρκεια της κάθε πρόσβασης, ελέγχονται και όλα τα προαιρετικά δυαδικά ψηφία που συνοδεύουν τις καταχωρήσεις του Πίνακα Σελίδων. Σε περίπτωση που βρεθεί η ζητούμενη καταχώρηση, επιστρέφεται η φυσική διεύθυνση, ενώ σε αντίθετη περίπτωση καλείται Σφάλμα Σελίδας. Καθώς η διαδικασία αυτή εκτελείται κάθε φορά που ζητείται μετάφραση κάποιας εικονικής διεύθυνσης, θεωρείται χρονοβόρα και μπορεί να κοστίσει αρκετούς κύκλους ρολογιού.

2.2.2 Κρυφή Μνήμη Αναζήτησης Μετάφρασης (Translation Lookaside Buffer)

Παραπάνω, αναφέραμε πως οι πίνακες σελίδων των ενεργών εφαρμογών ενός υπολογιστικού συστήματος αποθηκεύονται εντός της κύριας μνήμης, η οποία τυπικά υλοποιείται με DRAM. Κάθε λειτουργία μνήμης απαιτεί πολλαπλές διαδοχικές προσβάσεις μνήμης ώστε να βρεθεί η απεικόνιση της εικονικής μνήμης που χρησιμοποιείται και να πραγματοποιηθεί η πραγματική λειτουργία μνήμης. Η χρονική επιβάρυνση που προκαλούν οι προσβάσεις αυτές έφεραν την ανάγκη επινόησης της Κρυφής Μνήμης Αναζήτησης Μετάφρασης (Translation Lookaside Buffer ή αλλιώς TLB). Όπως προδίδει και το όνομά της, η TLB αποτελεί στην ουσία μια κρυφή μνήμη για τον πίνακα σελίδων, η οποία όμως σε αντίθεση με τις περισσότερες κρυφές μνήμες δεν υλοποιείται με Static Random-Access Memory (SRAM) αλλά με Content-Addressable Memory (CAM), λόγω της μεταβλητότητας του μεγέθους των σελίδων. Η μνήμη αυτή παρακολουθεί και αποθηκεύει ένα υποσύνολο του Πίνακα Σελίδων. Εκμεταλλευόμενη την τοπικότητα, τόσο χωρική όσο και χρονική, των αναφορών στον πίνακα σελίδων, στην TLB αποθηκεύονται οι μεταφράσεις που χρησιμοποιήθηκαν πιο πρόσφατα. Ενώ κάθε αναζήτηση στην TLB εκτελείται μέσω υλικού, καθώς η ταχύτητα της αναζήτησης είναι καίριας σημασίας για την απόδοση του συστήματος, η διαχείριση της TLB μπορεί να γίνεται είτε μέσω του υλικού είτε μέσω του λογισμικού. Κάθε καταχώρηση εντός της TLB χρησιμοποιεί ως δείκτη ένα κομμάτι του αριθμού εικονικής σελίδας και ως περιεχόμενο τον αριθμό φυσικής σελίδας σε συνδυασμό με ορισμένα από τα προαιρετικά δυαδικά ψηφία που αναφέραμε παραπάνω, καθώς πλέον δεν είναι απαραίτητη η προσπέλαση του πίνακα σελίδων για κάθε αναφορά. Σε κάθε αναφορά μνήμης, γίνεται αναζήτηση του αριθμού εικονικής μνήμης στην TLB. Σε περίπτωση ευστοχίας (TLB hit), επιστρέφεται ο αριθμός φυσικής σελίδας. Σε περίπτωση αστοχίας (TLB miss), ο επεξεργαστής πρέπει να ελέγξει εάν η σελίδα υπάρχει στη μνήμη. Σε περίπτωση που υπάρχει, η μετάφραση φορτώνεται στην TLB από τον πίνακα σελίδων και η εντολή που προκάλεσε την αστοχία εκτελείται εκ νέου. Σε αντίθετη περίπτωση, προκύπτει πραγματικό σφάλμα σελίδας, το οποίο αντιμετωπίζεται όπως αναφέρθηκε παραπάνω. Καθώς η TLB μπορεί να περιέχει συγκεκριμένο αριθμό εγγραφών, υιοθετούνται αλγόριθμοι αντικατάστασης ώστε να αποφασιστεί η καταχώρηση που θα αντικατασταθεί σε κάθε αστοχία.

2.2.2.1 Αρχιτεκτονική TLB

Ο σκοπός της TLB είναι η επιτάχυνση της μετάφρασης εικονικών διευθύνσεων σε φυσικές, ώστε να αυξηθεί η απόδοση της Εικονικής Μνήμης και κατ' επέκταση και των υπολογιστικών συστημάτων που τη χρησιμοποιούν. Η απόδοση της έχει αποδειχθεί κρίσιμη τόσο στη συνολική απόδοση του συστήματος, όσο και στην κατανάλωση ενέργειας αυτού. Για τον λόγο αυτό, έχουν υπάρξει μελέτες ώστε να βελτιστοποιηθεί η δομή και ο τρόπος λειτουργίας της. Όσον αφορά τη θέση της εντός του συστήματος, μπορεί να βρίσκεται μεταξύ του επεξεργαστή και των κρυφών μνημών αυτού, μεταξύ των κρυφών μνημών του επεξεργαστή και της κύριας μνήμης, ή ανάμεσα στα διαφορετικά επίπεδα κρυφής μνήμης του επεξεργαστή και πλέον εντοπίζεται σχεδόν σε όλα τα υπολογιστικά συστήματα που κάνουν χρήση εικονικής μνήμης. Η τοποθέτησή της προσδιορίζει και εάν οι κρυφές μνήμες χρησιμοποιούν φυσική ή εικονική διευθυνσιοδότηση.

Όπως αναφέρθηκε και παραπάνω, η TLB εκμεταλλεύεται την τοπικότητα των αναφορών στον πίνακα σελίδων για τη λειτουργία της. Παρόλα αυτά, με την πάροδο του χρόνου οι ανάγκες μνήμης των εφαρμογών καθώς

και το πλήθος των ενεργών εφαρμογών ενός μέσου συστήματος συνεχίζουν και αυξάνονται, με την TLB να οφείλει να κρατά τις απαραίτητες σελίδες για κάθε μία από τις εφαρμογές αυτές. Παράλληλα, οι προσβάσεις στην TLB (TLB Lookups) αποτελούν κομμάτι της διασωλήνωσης εντολών. Ως αποτέλεσμα, η TLB πρέπει να έχει ικανοποιητικά μικρό χρόνο πρόσβασης, γεγονός που σε συνδυασμό με την αύξηση της κατανάλωσης ενέργειας που θα επέφερε, καθιστά αδύνατη τη συνεχή αύξηση του μεγέθους της. Εφόσον λοιπόν δεν είναι δυνατή η αύξηση του συνόλου της μνήμης που είναι προσπελάσιμο μέσω της TLB, το οποίο ονομάζεται TLB Reach, έχουν προταθεί διαφορετικές τεχνικές ώστε να βελτιωθεί η λειτουργία της[18, 19].

Αρχικά, ομοίως με την οργάνωση των κρυφών μνημών ενός υπολογιστικού συστήματος, έχει προταθεί η Ιεραρχική Οργάνωση της TLB. Η οργάνωση αυτή αποτελείται συνήθως από 2 επίπεδα, μπορεί όμως να περιέχει και περισσότερα. Η αναζήτηση μετάφρασης σε μια Ιεραρχικά Οργανωμένη TLB ξεκινά από το πρώτο επίπεδο και σε περίπτωση που δεν βρεθεί συνεχίζεται στο αμέσως επόμενο. Το μέγεθος κάθε επιπέδου σταδιακά αυξάνεται, ομοίως και με τον χρόνο προσπέλασής του. Με τον τρόπο αυτό, επιτυγχάνεται γρηγορότερη εύρεση των μεταφράσεων που χρησιμοποιούνται συχνότερα, ενώ παράλληλα είναι δυνατή και η αύξηση του συνόλου των καταχωρήσεων της TLB. Σε αρχιτεκτονικές Harvard ή τροποποιημένες αρχιτεκτονικές Harvard, χρησιμοποιούνται ξεχωριστοί χώροι εικονικών διευθύνσεων για τα δεδομένα και τις εντολές. Στο πλαίσιο αυτό έχουν εισαχθεί και ξεχωριστές TLB για τις προσβάσεις εντολών και δεδομένων, ονομαζόμενες Instruction TLB ή αλλιώς ITLB και Data TLB ή αλλιώς DTLB αντίστοιχα. Ο διαχωρισμός αυτός είναι δυνατό να χρησιμοποιηθεί και σε συνδυασμό με την ιεραρχική οργάνωση της TLB.

Μια διαφορετική προσέγγιση για τη βελτίωση της απόδοσης της TLB ενός συστήματος, είναι η αλλαγή του μεγέθους σελίδας που χρησιμοποιείται[20]. Στο πλαίσιο αυτό, έχουν προταθεί ιδέες όπως τα SuperPages[21] και HugePages[22], που αυξάνουν το μέγεθος σελίδας που χρησιμοποιεί η TLB, γεγονός που θεωρητικά επιτρέπει στην TLB να εξυπηρετεί μεγαλύτερο κομμάτι της μνήμης χωρίς κάποιο κόστος και συνεπώς επιφέρει αύξηση του TLB Reach[23]. Παρόλα αυτά οι τεχνικές αυτές είναι πιθανό να οδηγήσουν στον εσωτερικό θρυμματισμό (Internal Fragmentation), εφόσον η χρήση σελίδων μεγάλου μεγέθους δεν είναι απαραίτητη για όλες τις διεργασίες ενός συστήματος, ενώ η αλλαγή του μεγέθους σελίδας παρουσιάζει και σημαντικές δυσκολίες καθώς αποτελεί αρχιτεκτονική παράμετρο του συστήματος. Εναλλακτικά, έχει προταθεί και η χρήση σελίδων πολλαπλών μεγεθών, η οποία με κατάλληλη υλοποίηση επιτρέπει σε εφαρμογές που απαιτούν χρήση σελίδων μεγάλου μεγέθους να τις χρησιμοποιούν, δίχως τον κίνδυνο εσωτερικού θρυμματισμού. Παρόλα αυτά, παρουσιάζονται εκ νέου δυσκολίες ενσωμάτωσης πολλαπλών μεγεθών σελίδας στην TLB λόγω των επιπτώσεων στην αρχιτεκτονική του υπολογιστικού συστήματος, ενώ είναι πιθανό να προκύψει και ένα νέο πρόβλημα, ο εξωτερικός θρυμματισμός (External Fragmentation).

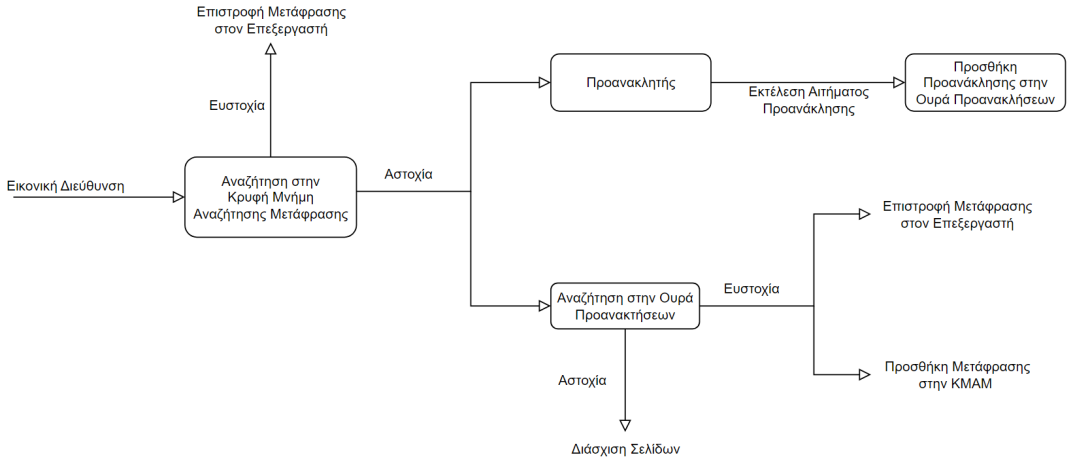
Μια ακόμη μέθοδος επιτάχυνσης της διαδικασίας μετάφρασης διευθύνσεων σε συστήματα που χρησιμοποιούν Πίνακες Σελίδων Πολλαπλών Επιπέδων και επικεντρώνεται στη μείωση του κόστους των αστοχιών στην TLB είναι η εισαγωγή των Κρυφών Μνημών Μετάφρασης (Translation Caches)[24, 25]. Οι μνήμες αυτές, βρίσκονται μεταξύ των επιπέδων του Πίνακα Σελίδων και αποθηκεύουν μερικές μεταφράσεις διευθύνσεων, δίνοντας τη δυνατότητα παράλειψης ενός η και περισσότερων επιπέδων κατά τη Διάσχιση Σελίδων. Παρομοίως, χρησιμοποιούνται και διαφορετικές υλοποιήσεις εντός της Μονάδας Διαχείρισης Μνήμης (Memory Management Unit), κομμάτι της οποίας αποτελεί και η Κρυφή Μνήμη Αναζήτησης Μετάφρασης, για την επιτάχυνση της Εικονικής Μνήμης.

2.3 Η Προανάκληση στην TLB

Η ανάγκη βελτίωσης της απόδοσης της TLB σε συνδυασμό με τη δυσκολία σχεδιασμού της έχουν στρέψει την επιστημονική κοινότητα στη χρήση τεχνικών Προανάκλησης στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης (TLB Prefetching). Όπως έχει αναφερθεί και παραπάνω, οι αναζητήσεις μεταφράσεων επιφέρουν αισθητές δαπάνες σε απόδοση και ενέργεια[5, 26]. Συνεπώς, οι επιτυχείς προανακλήσεις καταχωρήσεων από τον πίνακα σελίδων προτού αυτές ζητηθούν μπορούν να μετριάσουν τον κίνδυνο συμφόρησης του συστήματος. Στον αντίποδα, εφόσον κάθε προανάκληση απαιτεί τη διάσχιση του πίνακα σελίδων και διαδοχικές προσβάσεις μνήμης, η Προανάκληση στην Κρυφή Μνήμη Αναζήτησης είναι μια ιδιαίτερα κοστοβόρα διαδικασία, η οποία σε περίπτωση πολλαπλών μη επιτυχών προανακλήσεων μπορεί ακόμη και να μειώσει τη συνολική απόδοση του συστήματος.

Η ιδέα της Προανάκλησης Μεταφράσεων στην TLB δεν διαφέρει αισθητά από την προανάκληση εντολών ή δεδομένων στις κρυφές μνήμες του επεξεργαστή[27, 28]. Οι δυσκολίες που αναφέρθηκαν όμως παραπάνω έχουν οδηγήσει στη χρήση μιας δομής δεδομένων, της Ουράς Προανακλήσεων (Prefetch Queue/Buffer) κατά την προανάκληση σε αυτήν. Ο ρόλος της είναι η προσωρινή αποθήκευση των μεταφράσεων που φέρει ο προανακλητής, ώστε να αποφευχθεί η συμφόρηση της, ήδη περιορισμένου μεγέθους, TLB. Το μέγεθος της Ουράς Προανακλήσεων είναι σχετικά μικρό, τυπικά 16 έως 64 καταχωρήσεων, ώστε να διατηρεί ικανοποιητικά μικρό χρόνο πρόσβασης.

Όσον αφορά τη διαδικασία που ακολουθείται κατά την αναζήτηση κάποιας μετάφρασης διεύθυνσης σε σύστημα που υλοποιεί προανάκληση στην TLB, ξεκινάει όπως αναφέρθηκε και παραπάνω, με αναζήτηση της μετάφρασης στην TLB. Σε περίπτωση επιτυχούς αναζήτησης επιστρέφεται η ζητούμενη φυσική διεύθυνση και η εκτέλεση της διεργασίας συνεχίζεται κανονικά. Σε αντίθετη περίπτωση, δεν πυροδοτείται αμέσως Διάσχιση Σελίδων όπως προηγουμένως, γίνεται όμως αναζήτηση της μετάφρασης στον Prefetch Buffer. Σε περίπτωση που η αναζήτηση στην ουρά είναι επιτυχής, η εκτέλεση συνεχίζει σαν να είχαμε επιτυχή αναζήτηση στην TLB, ενώ η ζητούμενη μετάφραση αφαιρείται από την ουρά ώστε να μεταφερθεί στην TLB. Σε περίπτωση που έχουμε αστοχία και στον Prefetch Buffer, ενεργοποιείται η Διάσχιση Σελίδων και ακολουθείται η διαδικασία που αναλύθηκε και παραπάνω. Παράλληλα με την αναζήτηση στον Prefetch Buffer, πυροδοτείται ο προανακλητής ο οποίος εισάγει τις νέες προβλέψεις στον Prefetch Buffer, εφόσον δεν υπάρχουν ήδη σε αυτόν. Μια αναπαράσταση αυτής της διαδικασίας φαίνεται στο σχήμα 2.3.



Σχήμα 2.3: Ακολουθία προσβάσεων σε σύστημα που εκτελεί προανάκληση στην TLB

2.3.1 Τεχνικές Προανάκλησης

Στη συγκεκριμένη ενότητα θα παρουσιαστούν συνοπτικά ορισμένοι προανακλητές[29] στην TLB, οι οποίοι θα χρησιμοποιηθούν και ως μέτρο σύγκρισης σε παρακάτω ενότητα. Στις προσομοιώσεις, όλοι οι προανακλητές πυροδοτούνται σε κάθε αστοχία στο τελευταίο επίπεδο της TLB, ανεξαρτήτως από το εάν η αναζήτηση στην Prefetch Queue είναι επιτυχής ή όχι.

- *Ακολουθιακός Προανακλητής (Sequential Prefetcher):* Ο Ακολουθιακός Προανακλητής επιχειρεί να εκμεταλλευτεί τη χωρική τοπικότητα των αναφορών στη μνήμη, οι οποίες υποθέτει πως θα είναι σειριακές για κάθε εφαρμογή. Στη συγκεκριμένη διπλωματική εργασία γίνεται χρήση της απλούστερης μορφής του, στην οποία κάθε φορά που υπάρχει αστοχία στην TLB, γίνεται αίτημα προανάκλησης της ακριβώς επόμενης σελίδας στον Prefetch Buffer. Βεβαίως, υπάρχουν διάφορες παραλλαγές του συγκεκριμένου προανακλητή, με την απόσταση της σελίδας την οποία φέρνει σε σχέση με αυτή που προκάλεσε την αστοχία να είναι παραμετροποιήσιμη.
- *Προανακλητής Αυθαίρετου Βήματος (Arbitrary Stride Prefetcher):* Ο συγκεκριμένος μηχανισμός επιχειρεί να εκμεταλλευτεί τα μοτίβα που προκύπτουν μεταξύ των διαδοχικών προσβάσεων μνήμης κάθε εφαρμογής. Για να το επιτύχει αυτό, χρησιμοποιεί έναν πίνακα, ονομαζόμενο Πίνακα Πρόβλεψης Αναφορών (Reference

Prediction Table), με σκοπό να κρατά χρήσιμες πληροφορίες για τις προηγούμενες προσβάσεις. Συγκεκριμένα, ο πίνακας χρησιμοποιεί τον Μετρητή Προγράμματος (Program Counter) ως δείκτη, ενώ παράλληλα αποθηκεύει:

(i) την τελευταία διεύθυνση στην οποία ζητήθηκε πρόσβαση κατά τη συγκεκριμένη εντολή

(ii) το παρών βήμα, δηλαδή την απόσταση μεταξύ διαδοχικών προσβάσεων στη συγκεκριμένη εντολή

(iii) την κατάσταση (state), που πρακτικά αποτελεί έναν μετρητή για τη σταθερότητα του βήματος

Ο προανακλητής εκτελεί αιτήματα μόνο όταν η κατάσταση του βήματος παραμείνει σταθερή για τουλάχιστον δύο διαδοχικές αναφορές κάποιας εντολής, με σκοπό να αποφευχθεί η υπερχείλιση του Prefetch Buffer με προανακλήσεις που δεν θα χρησιμοποιηθούν ποτέ.

- *Προανακλητής Markov (Markov Prefetcher):* Οι προανακλητές στους οποίους έχουμε αναφερθεί μέχρι στιγμής, επιχειρούν να εντοπίσουν κάποια κανονικότητα στις προσβάσεις μνήμης που εκτελεί η διεργασία, με αποτέλεσμα να αποτυγχάνουν πλήρως στην εύρεση χρήσιμων προνακλήσεων σε περίπτωση που το μοτίβο αυτό δεν υπάρχει. Αντιθέτως, ο προανακλητής Markov επιχειρεί να υπολογίσει την πιθανότητα να χρειαστεί κάποια καταχώρηση την επόμενη φορά που θα υπάρξει αίτημα πρόσβασης στην παρούσα σελίδα. Συγκεκριμένα, δημιουργείται ένας πίνακας προβλέψεων, ο οποίος χρησιμοποιεί ως δείκτη τους αριθμούς εικονικής σελίδας για τις οποίες υπήρξε αστοχία στην TLB, και ως δεδομένα τις ακριβώς προηγούμενες διευθύνσεις που προκάλεσαν αστοχία στην TLB προτού γίνει αίτημα πρόσβασης στη σελίδα που υποδεικνύει ο δείκτης. Σε κάθε νέα αστοχία, ενημερώνεται η γραμμή της ακριβώς προηγούμενης αστοχίας, ενώ εάν ο αριθμός της παρούσας εικονικής σελίδας δεν αντιστοιχεί σε κάποιον δείκτη εντός του πίνακα τότε προστίθεται καινούργια καταχώρηση. Όπως είναι φυσικό, το μέγεθος του πίνακα είναι περιορισμένο, και έτσι χρησιμοποιείται ο αλγόριθμος Least-Recently Used (LRU) τόσο για τις διευθύνσεις κάθε γραμμής, όσο και καταχωρήσεις εντός του πίνακα. Προανακλήσεις πραγματοποιούνται όταν μια διεύθυνση που προκάλεσε αστοχία στην TLB υπάρχει εντός του πίνακα προβλέψεων, στην οποία περίπτωση πραγματοποιούνται προανακλήσεις για κάθε διεύθυνση εντός της γραμμής στην οποία αντιστοιχεί η διεύθυνση.

- *Προανακλητής Απόστασης [30] (Distance Prefetcher):* Ο Προανακλητής Απόστασης χρησιμοποιεί την απόσταση, η οποία ονομάζεται και delta, μεταξύ διαδοχικών προσβάσεων στη μνήμη για να προβλέψει τις επόμενες αστοχίες που θα προκύψουν. Στην ουσία, υιοθετεί ένα Πίνακα Αποστάσεων που έχει αντίστοιχη λειτουργία με τον Πίνακα Προβλέψεων του προανακλητή Markov. Ο Πίνακας Προβλέψεων που διατηρεί, χρησιμοποιεί τις αποστάσεις μεταξύ αστοχιών ως δείκτη και αποθηκεύει το ιστορικό των αποστάσεων που ακολούθησαν με σκοπό να εκτελέσει προβλέψεις για τις πιθανότερες χρήσιμες προανακλήσεις. Αυτός ο τρόπος λειτουργίας επιτρέπει στον Προανακλητή Απόστασης να εντοπίζει μοτίβα εξίσου καλά με τους προανακλητές που έχουν αναφερθεί μέχρι στιγμής, με τις απαιτήσεις μνήμης του να είναι ανάλογες της πολυπλοκότητας των μοτίβων των προσβάσεων μνήμης.

- *Προανακλητής Διορατικότητας (Oracle Prefetcher):* Ο Προανακλητής Oracle αποτελεί έναν προανακλητή που μπορεί να υλοποιηθεί μονάχα εφόσον είναι γνωστό εκ των προτέρων το πλήρες ιστορικό προσβάσεων κάποιας εφαρμογής. Εφόσον τα μελλοντικά αιτήματα πρόσβασης είναι γνωστά, ο προανακλητής αυτός γνωρίζει σε ποιες διευθύνσεις θα προκύψουν μελλοντικές αστοχίες και μπορεί να τις φέρει στην Prefetch Queue προτού ζητηθούν. Η λειτουργία του είναι καθαρά θεωρητική, εφόσον σε ένα κανονικό υπολογιστικό σύστημα δεν είναι δυνατόν να είναι γνωστές εκ των προτέρων οι μελλοντικές προσβάσεις, και χρησιμοποιείται στην παρούσα διπλωματική ως μέτρο σύγκρισης της αποτελεσματικότητας διάφορων προανακλητών, καθώς και ως μέσο εύρεσης της κατάλληλης επικαιρότητας προανακλήσεων στην TLB.

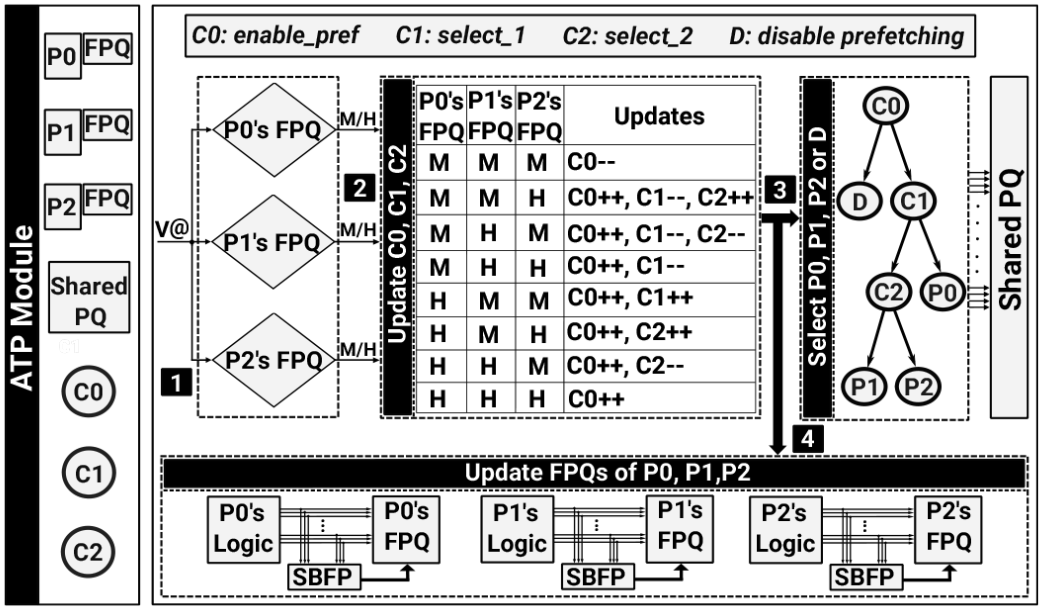
- *Agile TLB Prefetcher (ATP):* Ο Agile TLB Prefetcher [31] αποτελεί έναν προανακλητή στην TLB που επιχειρεί να εκμεταλλευτεί διαφορετικά χαρακτηριστικά των προσβάσεων στη μνήμη για να εντοπίσει μοτίβα, συνδυάζοντας τρεις προανακλητές χαμηλού κόστους. Οι προανακλητές αυτοί είναι ο Προανακλητής Βήματος (Stride Prefetcher), ο Προανακλητής H2 (H2 Prefetcher) και ο Τροποποιημένος Προανακλητής Αυθαίρετου Βήματος(Modified Arbitrary Stride Prefetcher). Η λειτουργία κάθε ενός από αυτούς περιγράφεται εν συντομία παρακάτω:

SP: Ο Προανακλητής Βήματος, αποτελεί μια πιο επιθετική εκδοχή του Ακολουθιακού Προανακλητή, η οποία σε κάθε αστοχία της Κρυφής Μνήμης Αναζήτησης Μετάφρασης εκτελεί αιτήματα προανάκλησης για τις σελίδες που βρίσκονται σε απόσταση έως και 2 από τη σελίδα όπου προέκυψε η αστοχία. Εκτελεί δηλαδή, σε κάθε αστοχία, τέσσερα αιτήματα προανάκλησης, με βήματα -2, -1, +1, +2.

H2P: Ο H2P αποθηκεύει και χρησιμοποιεί τις δύο αποστάσεις μεταξύ των τριών τελευταίων διευθύνσεων που προκάλεσαν αστοχία στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Σε κάθε νέα αστοχία, ο προανακλητής εκτελεί αίτημα προανάκλησης για τις καταχωρήσεις που βρίσκονται στις δύο αυτές αποστάσεις από την παρούσα σελίδα.

MASP: Ο MASP αποτελεί μια βελτιωμένη και πιο επιθετική εκδοχή του Προανακλητή Αυθαίρετου Βήματος. Αρχικά, η εκδοχή αυτή αφαιρεί την απαίτηση σταθερότητας του βήματος, δίνοντας περισσότερες ευκαιρίες προανάκλησης στον μηχανισμό. Με τον τρόπο αυτό, σε κάθε αστοχία εκτελείται προανάκληση της καταχώρησης που απέχει απόσταση ίση με το βήμα από τη σελίδα που προκάλεσε την αστοχία. Παράλληλα, αποθηκεύεται και η προηγούμενη σελίδα που προκάλεσε αστοχία για κάθε Δείκτη Προγράμματος εντός του πίνακα. Με τον τρόπο αυτόν σε κάθε αστοχία στην TLB εκτελείται και μια δεύτερη προανάκληση, αυτή τη φορά στη σελίδα που απέχει απόσταση ίση με την απόσταση μεταξύ της παρούσας και της τελευταίας αστοχίας για τον εκάστοτε Δείκτη Προγράμματος.

Κάθε ένας από τους προανακλητές αυτούς χρησιμοποιεί και μία "ψεύτικη" ουρά προανάκλησης. Οι ουρές αυτές περιέχουν μονάχα τις εικονικές διευθύνσεις που προέβλεψε κάθε προανακλητής και χρησιμοποιούνται για τη μέτρηση της ακρίβειας των προανακλητών, η οποία με τη σειρά της χρησιμοποιείται για την επιλογή του προανακλητή που θα χρησιμοποιηθεί στη συνέχεια. Η δομή επιλογής προανακλητή του ATP είναι δενδρική, και φαίνεται λεπτομερώς στο σχήμα 2.4. Σημειώνεται ότι στην εκδοχή που έχει χρησιμοποιηθεί στην παρούσα εργασία απουσιάζει η SAMPLING-BASED FREE TLB PREFETCHING (SBFP), η οποία λειτουργεί συνδυαστικά με προανακλητές ώστε να βελτιώσει την απόδοσή τους. Όσον αφορά τον τρόπο επιλογής προανακλητή, σε κάθε αστοχία, γίνεται αναζήτηση στις ουρές προανάκλησης των προανακλητών και ενημέρωση των αντίστοιχων μετρητών. Στη συνέχεια, οι μετρητές αυτοί χρησιμοποιούνται ώστε να γίνει η επιλογή μεταξύ των τριών προανακλητή και της επιλογής του να μην εκτελεστεί προανάκληση. Σε κάθε περίπτωση, οι "ψεύτικες" ουρές προανάκλησης όλων των προανακλητών ανανεώνονται με τις προβλέψεις που αυτοί θα εκτελούσαν, κρατώντας έτσι επίκαιρη τη βέλτιστη επιλογή μεταξύ των τριών.



Σχήμα 2.4: Δομή του ATP

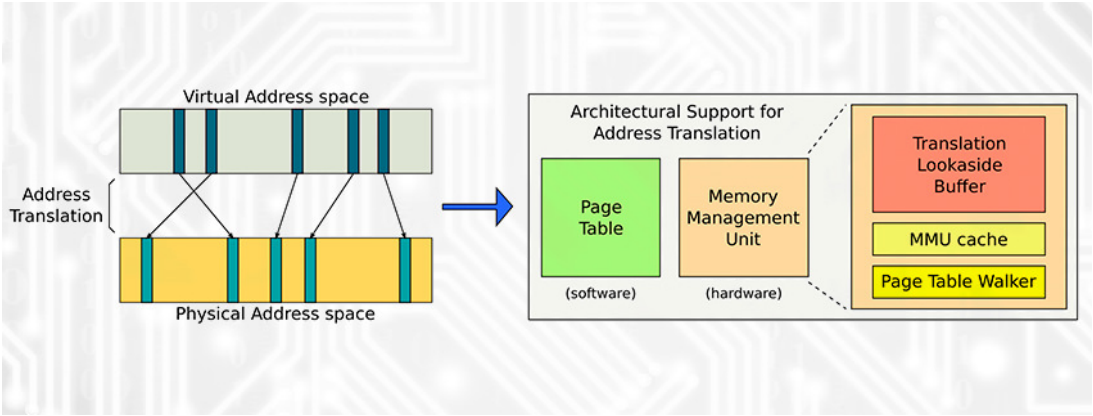
Source: `http://www.cslab.ece.ntua.gr/~knikas/files/papers/2021-isca-tlb_prefetching.pdf`

2.3.2 Προκλήσεις της προανάκλησης

Κατά τη σχεδίαση του μηχανισμού με τον οποίο λειτουργεί ένας προανακλητής, υπάρχουν ορισμένες δυσκολίες οι οποίες πρέπει να ληφθούν υπόψιν ώστε η απόδοσή του στην λειτουργία πληθώρας εφαρμογών να είναι ικανοποιητική. Παράλληλα, κάθε προανακλητής κάνει χρήση ορισμένων παραμέτρων για να αποφασίσει τα δεδομένα, ή στην περίπτωση των προανακλητών TLB την καταχώρηση του πίνακα σελίδων, που θα φέρει. Ορισμένα από τα ζητήματα αυτά θα αναφερθούν συνοπτικά παρακάτω.

2.3.2.1 Δεδομένα Προανάκλησης

Η πλέον σημαντική παράμετρος αξιολόγησης ενός προανακλητή είναι η χρησιμότητα των δεδομένων που θα αποφασίσει να φέρει από τη μνήμη. Για να είναι επιτυχής μια προανάκληση, ο προανακλητής θα πρέπει να είναι ικανός να προβλέψει μελλοντικές αστοχίες που θα προκύψουν κατά την εκτέλεση της εφαρμογής. Σε περίπτωση που οι προβλέψεις του προανακλητή είναι ανακριβείς, θα προκύψει μόλυνση της κρυφής μνήμης με προανακλήσεις που δεν θα χρησιμοποιηθούν ποτέ, και πιθανώς θα είναι και επιβλαβείς καθώς θα αντικαταστήσουν χρήσιμα δεδομένα. Παράλληλα, τίθεται και το ζήτημα του πλήθους των αιτημάτων προανάκλησης, καθώς με την αύξηση του αριθμού των δεδομένων που έρχονται στην κρυφή μνήμη είναι σαφώς πιθανότερο κάποιο από αυτά να είναι χρήσιμα. Βεβαίως, στην περίπτωση που ο προανακλητής δεν είναι ακριβής μια τέτοια αύξηση θα οδηγήσει στη μόλυνση της μνήμης με ταχύτερους ρυθμούς. Στην περίπτωση της προανάκλησης στην TLB, ο κίνδυνος ζημιογόνων προανακλήσεων μετριάζεται από το γεγονός πως χρησιμοποιείται ο Prefetch Buffer. Παρόλα αυτά, ο κίνδυνος αυτός δεν είναι ανύπαρκτος, καθώς επαναλαμβανόμενες προανακλήσεις που δεν χρησιμοποιούνται μπορούν εύκολα να γεμίσουν τον Prefetch Buffer, λόγω του περιορισμένου μεγέθους του, και να καταστήσουν τον προανακλητή μη αποδοτικό. Για τους λόγους αυτούς, η απόφαση για το ποιες αλλά και το πόσες καταχωρήσεις θα έρθουν στη μνήμη είναι ζωτικής σημασίας για την εύρωστη λειτουργία κάθε προανακλητή.



Σχήμα 2.5: Στοιχεία Μνήμης ενός συστήματος

2.3.2.2 Η Επικαιρότητα της Προανάκλησης (Timeliness)

Ένας προανακλητής είναι υπεύθυνος για την πρόβλεψη μελλοντικών αστοχιών σε κάποια κρυφή μνήμη, και τη μεταφορά των κατάλληλων δεδομένων σε αυτή προτού αυτά να χρειαστούν. Ένα πρόβλημα που προκύπτει, ιδίως στην περίπτωση της προανάκλησης στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης, είναι η επικαιρότητα της προανάκλησης, δηλαδή το κατά πόσο οι προβλέψεις του προανακλητή έχουν προλάβει να μεταφερθούν στον Prefetch Buffer προτού αυτά ζητηθούν[32, 33]. Κάθε προανάκληση στην TLB απαιτεί μια διάσχιση σελίδων και ως αποτέλεσμα αρκετούς κύκλους ρολογιού έως ότου ολοκληρωθεί. Είναι λοιπόν πιθανό, ενώ ο προανακλητής εκτελεί σωστή πρόβλεψη, ο χρόνος μέχρι να προκύψει η αστοχία να μην επαρκεί ώστε να μεταφερθούν τα δεδομένα, καθιστώντας την προανάκληση ζημιογόνο καθώς τα δεδομένα θα φτάσουν στην Prefetch Queue αφού προκύψει η αστοχία. Κατ΄ επέκταση, θα έχει υπάρξει ήδη αίτημα ώστε τα εν λόγω δεδομένα να εισαχθούν στην TLB, με αποτέλεσμα να χρησιμοποιηθούν πόροι για τις απαραίτητες προσβάσεις στη μνήμη δίχως όφελος. Για την αντιμετώπιση του συγκεκριμένου προβλήματος, οι προανακλητές θα πρέπει στην ουσία να επιχειρούν να προβλέψουν μελλοντικές αστοχίες σε μεγαλύτερη απόσταση, ενώ σε επίπεδο ανάλυσης είναι χρήσιμος και

ο υπολογισμός των χρονικών διαστημάτων μεταξύ διαδοχικών προσβάσεων στη μνήμη. Χρησιμοποιώντας ως παράδειγμα τον Προανακλητή Απόστασης που αναφέρθηκε και παραπάνω, η λειτουργία του θα μπορούσε να αλλάξει ώστε να καταγράφει και να προβλέπει τις αποστάσεις έως τις πιθανότερες αστοχίες δύο ή και περισσότερα αιτήματα προσβάσεων μετά.

2.3.2.3 Επιβάρυνση της Υλοποίησης

Μια ακόμη παράμετρος που θα πρέπει να ληφθεί υπόψιν κατά τη μεταφορά προανακλητών από τη θεωρία στην πράξη είναι η επιβάρυνση της υλοποίησης[34]. Η υλοποίησή του απλούστερου μηχανισμού προανάκλησης που αναφέρθηκε παραπάνω, δηλαδή ο Ακολουθιακός Προανακλητής, δεν επιφέρει βάρος στο υπολογιστικό σύστημα καθώς δεν χρειάζεται να αποθηκεύσει επιπλέον δεδομένα για τη λειτουργία του. Αντιθέτως, πιο εκλεπτυσμένοι προανακλητές, όπως ο Προανακλητής Markov θα πρέπει να αποθηκεύσουν μεγάλο ποσό μεταδεδομένων. Τα δεδομένα αυτά θα μπορούσαν είτε να αποθηκευτούν στη μνήμη, γεγονός που οδηγεί σε ακριβή και συχνή επικοινωνία, είτε αποθηκεύονται on-chip γεγονός που δεσμεύει χώρο στον επεξεργαστή ενώ παράλληλα απαιτεί και τη χρήση μικρών δομών, οι οποίες σε πολλές περιπτώσεις είναι και ακριβές στην υλοποίησή τους.

Κεφάλαιο 3

Μηχανική Μάθηση & Τεχνητά Νευρωνικά Δίκτυα

Η Μηχανική Μάθηση (Machine Learning - ML) είναι ο τομέας της τεχνητής νοημοσύνης (Artificial Intelligence - AI) που επικεντρώνεται στη χρήση δεδομένων και υπολογιστικών αλγορίθμων, με στόχο τη δημιουργία μοντέλων ικανά να επιλύσουν απαιτητικά προβλήματα, ακόμη και εάν δεν είναι ρητά προγραμματισμένα για αυτό[35, 36]. Η διαδικασία της μηχανικής μάθησης περιλαμβάνει τη τροφοδότηση δεδομένων στο εκάστοτε μοντέλο, τα οποία αναλύει και χρησιμοποιεί για να εκπαιδευθεί στην επίλυση μελλοντικών προβλημάτων ίδιου τύπου, διαφορετικών όμως δεδομένων. Παρακάτω, γίνεται μια εισαγωγή σε βασικές έννοιες της Μηχανικής Μάθησης και των Τεχνητών Νευρωνικών Δικτύων που χρησιμοποιήθηκαν για τη διεκπεραίωση της παρούσας διπλωματικής εργασίας.

3.1 Τύποι αλγορίθμων μηχανικής μάθησης

Κύριο κριτήριο κατηγοριοποίησης των αλγορίθμων μηχανικής μάθησης, αποτελεί η διαδικασία μάθησης που επιλέγεται για την εκπαίδευση τους. Στο πλαίσιο αυτό, οι αλγόριθμοι μπορούν να διαχωριστούν σε δύο κατηγορίες, Μάθηση με Εκπαιδευτή, ή αλλιώς Επιβλεπόμενη Μάθηση (Supervised learning) και Μάθηση χωρίς Εκπαιδευτή. Η δεύτερη κατηγορία μπορεί να κατηγοριοποιηθεί περαιτέρω στη Μη Επιβλεπόμενη Μάθηση (Unsupervised learning) και την Ενισχυτική Μάθηση (Reinforcement learning).

3.1.1 Επιβλεπόμενη Μάθηση (Supervised learning)

Κατά την επιβλεπόμενη μάθηση, το μοντέλο καλείται να μάθει μια συνάρτηση αντιστοίχησης κάποιας εισόδου σε συγκεκριμένη έξοδο, με τη χρήση δεδομένων υπό μορφής ζευγών ενός διανύσματος εισόδου και της επιθυμητής εξόδου. Η συγκεκριμένη μέθοδος, χρησιμοποιείται συνήθως για την επίλυση δύο ειδών προβλημάτων: τα προβλήματα παλινδρόμησης και τα προβλήματα ταξινόμησης.

- *Προβλήματα Ταξινόμησης (Classification):* Στα προβλήματα ταξινόμησης, ή αλλιώς κατηγοριοποίησης, το μοντέλο επιχειρεί να αναγνωρίσει τη διακριτή κατηγορία στην οποία ανήκει κάθε ένα από τα δεδομένα εισόδου που του παρέχονται. Συγκεκριμένα στη δυαδική αναζήτηση, οι τιμές εξόδου μπορούν να είναι είτε 0 είτε 1, ενώ σε προβλήματα πολλαπλών κατηγοριών, κάθε κατηγορία αντιστοιχείται σε μια ετικέτα και η έξοδος παίρνει τη μορφή ενός πίνακα πιθανοτήτων. Τα χαρακτηριστικότερα παραδείγματα τέτοιου είδους προβλημάτων αποτελεί ο διαχωρισμός φωτογραφιών ζώων ή η αναγνώριση ενός αριθμού.
- *Προβλήματα Παλινδρόμησης (Regression):* Στα προβλήματα παλινδρόμησης, η έξοδος του μοντέλου είναι συνεχής, ενώ ο σκοπός είναι η εκτίμηση αυτής με βάση την είσοδο που δίνεται στο μοντέλο. Η αξιολόγηση γίνεται με βάση το σφάλμα από την πραγματική τιμή της εξόδου. Για τους παραπάνω λόγους, τα μοντέλα αυτά χρησιμοποιούνται συνήθως για τη δημιουργία προβλέψεων, λ.χ. της τιμής κάποιας μετοχής στο χρηματιστήριο ή η πρόβλεψη καιρικών φαινομένων.

3.1.2 Μη Επιβλεπόμενη Μάθηση (Unsupervised learning)

Κατά τη μη επιβλεπόμενη μάθηση, το μοντέλο καλείται να αναγνωρίσει πρότυπα και μοτίβα εντός του συνόλου δεδομένων που του παρέχεται, χωρίς κάποια εξωτερική βοήθεια, δηλαδή χωρίς να γνωρίζει την επιθυμητή έξοδο.

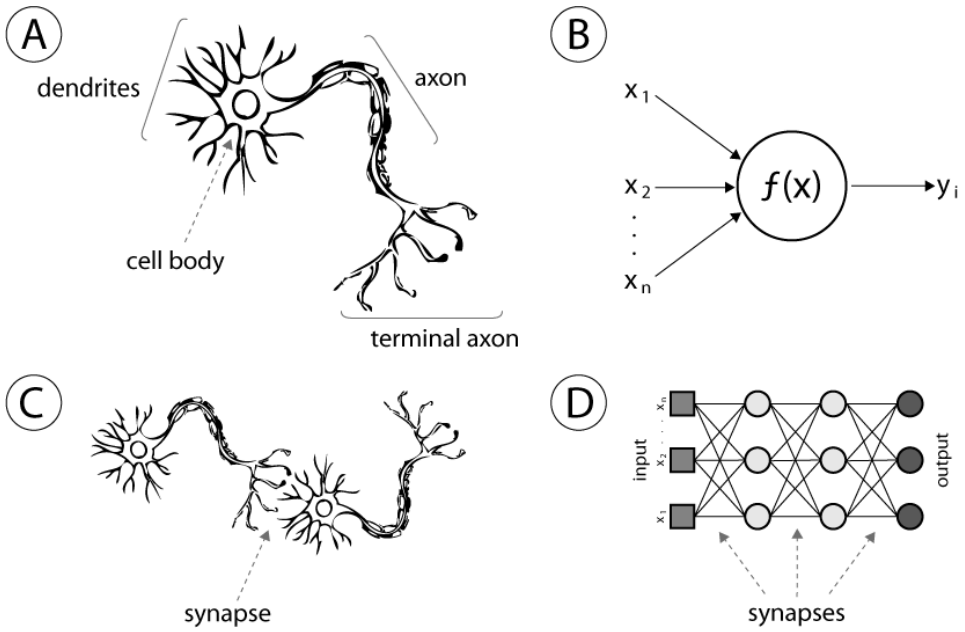
Κατά τη διαδικασία της εκπαίδευσης, το μοντέλο επιχειρεί να βρει ομοιότητες μεταξύ των δεδομένων εισόδου και να τα χωρίσει σε ομάδες (clusters). Παρόλο που η συγκεκριμένη μέθοδος εκπαίδευσης επιτρέπει στον αλγόριθμο να αντιμετωπίσει πιο σύνθετες, και πολλές φορές όχι αυστηρώς καθορισμένες, εργασίες σε σχέση με τα μοντέλα επιβλεπόμενης μάθησης, πολλές φορές τα αποτελέσματα μπορεί να είναι απρόβλεπτα καθώς το μοντέλο είναι το ίδιο υπεύθυνο τόσο για το πλήθος, όσο και τα χαρακτηριστικά των ομάδων που θα χρησιμοποιηθούν. Αλγόριθμοι μη επιβλεπόμενης μάθησης χρησιμοποιούνται ευρέως σε συστήματα συστάσεων, συστήματα ανίχνευσης ανωμαλιών (λ.χ. ελαττωματικών εξαρτημάτων) αλλά και στον τομέα της γενετικής, όπου επιτρέπουν την ανίχνευση σύνθετων μοτίβων.

3.1.3 Ενισχυτική μάθηση (Reinforcement learning)

Κατά την ενισχυτική μάθηση, χρησιμοποιείται ένα σύστημα επιβράβευσης και τιμωρίας, με το μοντέλο πρακτικά να επιχειρεί να επιλέξει σε κάθε βήμα τις κατάλληλες ενέργειες ώστε να μεγιστοποιήσει τον βαθμό επιβράβευσής του. Η κύρια διαφοροποίηση με τους αλγορίθμους επιβλεπόμενης μάθησης είναι πως το μοντέλο δεν γνωρίζει εξαρχής το επιθυμητό αποτέλεσμα, αλλά μόνο το κατά πόσο ορισμένες πράξεις έχουν θετικό ή αρνητικό αντίκτυπο, ενώ απουσιάζουν πλήρως τα ζεύγη εισόδου-εξόδου που αναφέραμε παραπάνω. Οι εφαρμογές της ενισχυτικής μάθησης έχουν μεγάλο εύρος και, μεταξύ άλλων, περιλαμβάνουν την εκπαίδευση σε παιχνίδια, την επεξεργασία φυσικής γλώσσας και συναλλαγές στο χρηματιστήριο.

3.2 Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks)

Με τον όρο Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks) ή απλώς Νευρωνικά Δίκτυα (Neural Networks), αναφερόμαστε σε υπολογιστικά συστήματα η λειτουργία των οποίων είναι εμπνευσμένη από τα βιολογικά νευρωνικά δίκτυα που συνιστούν τον ανθρώπινο εγκέφαλο. Στο πνεύμα αυτό, εισήχθη η ιδέα των τεχνητών νευρώνων, ή αλλιώς αισθητήρων. Οι νευρώνες αυτοί συνιστούν μια ομάδα συνδεδεμένων κόμβων, με κάθε νευρώνα να αντιστοιχείται σε έναν κόμβο, και έχουν την ικανότητα να στέλνουν σήματα σε άλλους νευρώνες μέσω των ακμών που τους συνδέουν, ομοίως με τις συνάψεις που αποστέλλουν ηλεκτρικά σήματα σε ένα βιολογικό εγκέφαλο. Στην περίπτωσή μας, τα σήματα αυτά παίρνουν κάποια πραγματική μαθηματική τιμή, ενώ η επιρροή τους στην έξοδο του δικτύου καθορίζεται από μια σειρά από βάρη που χαρακτηρίζουν κάθε νευρώνα. Οι τιμές αυτές παραμετροποιούνται κατά τη διάρκεια της διαδικασίας της εκπαίδευσης και καθορίζουν την ισχύ του σήματος που αποστέλλεται.



Σχήμα 3.1: Στοιχεία βιολογικών και τεχνητών νευρωνικών δικτύων

Source: `https://bit.ly/3CaZNlR`

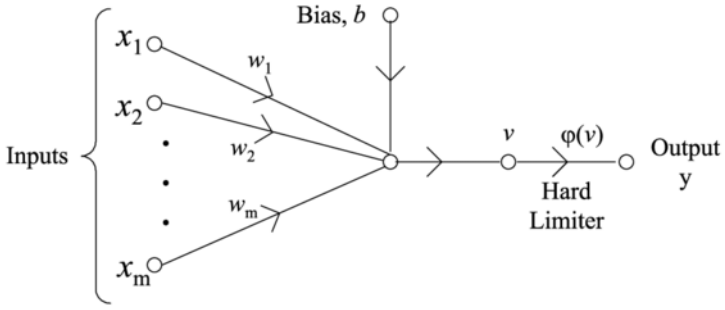
3.2.1 Δομικά Στοιχεία Τεχνητών Νευρωνικών Δικτύων

3.2.1.1 Τεχνητοί "αισθητήρες" - το Perceptron

Το Perceptron, στη μορφή που παρουσιάστηκε από τον F. Rosenblatt το 1957[37], βασίζεται στο μοντέλο ενός μη γραμμικού νευρώνα και αποτελεί το πρώτο νευρωνικό δίκτυο που μπορούσε να περιγραφεί αλγοριθμικά. Συγκεκριμένα, και όπως φαίνεται και παρακάτω στο σχήμα 3.2, αποτελείται από ένα γραμμικό συνδυαστή, ακολουθούμενο από έναν απότομο περιοριστή(hard limiter). Ο κόμβος άθροισης του μοντέλου, υπολογίζει ένα γραμμικό συνδυασμό των εισόδων που δέχεται, ενσωματώνοντας παράλληλα μια εξωτερικά εφαρμοζόμενη πόλωση (bias). Εάν συμβολίσουμε τα συναπτικά βάρη του νευρώνα ως w_1, w_2, ..., w_m ,τις αντίστοιχες εισόδους που εφαρμόζονται στο perceptron ως x_1, x_2, ..., x_m και την εξωτερική πόλωση ως b, μπορούμε να περιγράψουμε την είσοδο του απότομου περιοριστή ως:

$$u = \sum_{i=1}^{m} w_i * x_i + b$$

Το παραγόμενο αποτέλεσμα, εφαρμόζεται έπειτα στον απότομο περιοριστή, ο οποίος παράγει ως έξοδο απόκριση +1 ή -1, εάν η είσοδός του είναι θετική ή αρνητική αντιστοίχως. Ο στόχος του perceptron στη μορφή που περιγράψαμε παραπάνω, ήταν η ταξινόμηση ενός συνόλου εξωτερικά εφαρμοζόμενων διεγέρσεων, δηλαδή εισόδων, σε δύο κλάσεις. Η ταξινόμηση αυτή γίνεται με βάση την έξοδο y του απότομου περιοριστή.

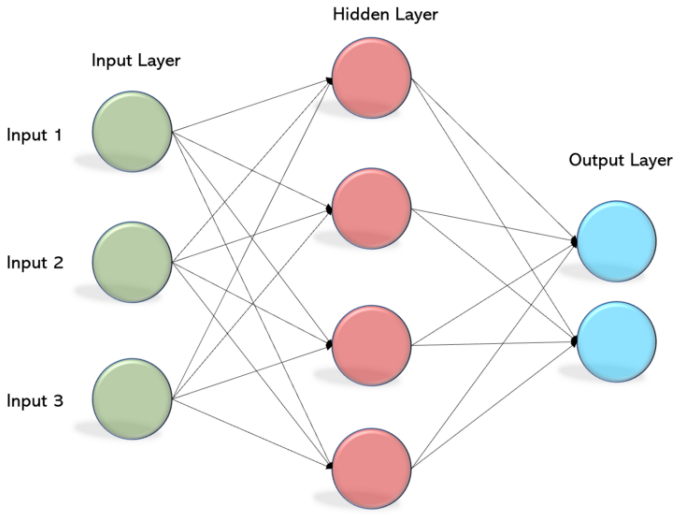


Σχήμα 3.2: Διάταξη ενός Perceptron

3.2.1.2 Το Perceptron Πολλαπλών Στρωμάτων

Τα perceptron πολλών επιπέδων, όπως προδίδει και η ονομασία τους, αποτελούνται από πολλαπλά επίπεδα διατάξεων νευρώνων και συνιστούν πρακτικά την αρχή των βαθιών νευρωνικών δικτύων (Deep Neural Networks). Ένα τέτοιου είδους δίκτυο, αποτελείται από τρία επίπεδα: το επίπεδο εισόδου (input layer), το κρυφό επίπεδο (hidden layer) και το επίπεδο εξόδου (output layer). Τα επίπεδα εισόδου και εξόδου αποτελούνται πάντοτε από ένα στρώμα, σε αντίθεση με το κρυφό επίπεδο το οποίο, ιδίως στην περίπτωση βαθιών νευρωνικών δικτύων, αποτελείται από πολλαπλά στρώματα. Κάθε στρώμα, αποτελείται από ένα σύνολο νευρώνων, το πλήθος των οποίων ονομάζεται πλάτος του επιπέδου (layer width). Ένα παράδειγμα ενός τέτοιου είδους δικτύου, με ένα κρυφό επίπεδο, φαίνεται στο σχήμα 3.3.

Το πλήθος των στρωμάτων και νευρώνων που αποτελούν το δίκτυο εξαρτάται από τις απαιτήσεις του προβλήματος που καλείται να επιλύσει το μοντέλο. Συγκεκριμένα, όπως αναφέραμε και παραπάνω, το επίπεδο εισόδου του δικτύου αποτελείται πάντα από ένα στρώμα, ενώ το πλάτος του ισούται συνήθως με το πλήθος των χαρακτηριστικών των δεδομένων εισόδου, με την επιλογή να προστεθεί ένας επιπλέον νευρώνας για την πόλωση. Όσον αφορά το επίπεδο εξόδου, που επίσης αποτελείται πάντοτε από ένα στρώμα, το πλάτος του εξαρτάται από το είδος της επιθυμητής εξόδου. Για παράδειγμα, στην περίπτωση που η έξοδος πρέπει να παίρνει μια πραγματική τιμή, το στρώμα εξόδου θα αποτελείται από έναν νευρώνα, ενώ σε έναν ταξινομητή μπορεί να χρησιμοποιηθεί ένας κόμβος για κάθε κατηγορία. Όσον αφορά το κρυφό επίπεδο, τόσο το πλήθος των στρωμάτων όσο και το πλάτος τους εξαρτάται από την πολυπλοκότητα του προβλήματος που καλείται να αντιμετωπίσει το δίκτυο. Κατά κανόνα, πολλαπλά κρυφά επίπεδα χρησιμοποιούνται μονάχα κατά την επίλυση σύνθετων και πολύπλοκων προβλημάτων, καθώς καθιστούν δυσκολότερη και τη διαδικασία της εκπαίδευσης.



Σχήμα 3.3: Παράδειγμα διάταξης ενός Perceptron πολλών επιπέδων

3.2.2 Εκπαίδευση Νευρωνικών Δικτύων

Η εκπαίδευση ενός νευρωνικού δικτύου, είναι η επαναλαμβανόμενη διαδικασία κατά την οποία το δίκτυο καλείται να προσαρμοστεί, αλλάζοντας τα βάρη κάθε κόμβου και προαιρετικά κάποιων ορίων, σε απάντηση ορισμένων δεδομένων εισόδου ώστε να μπορέσει να αντιμετωπίσει καλύτερα κάποια εργασία. Η διαδικασία αυτή τυπικά φτάνει στο τέλος της όταν η αύξηση του πλήθους των εποχών, δηλαδή των επαναλήψεων, δεν φέρει μείωση των σφαλμάτων. Για να είναι επιτυχής η εκπαίδευση ενός νευρωνικού δικτύου είναι απαραίτητη η επιλογή των κατάλληλων μεθόδων και παραμέτρων, ορισμένες εκ των οποίων αναπτύσσονται στο παρόν κεφάλαιο.

3.2.2.1 Συνάρτηση Ενεργοποίησης (Activation Function)

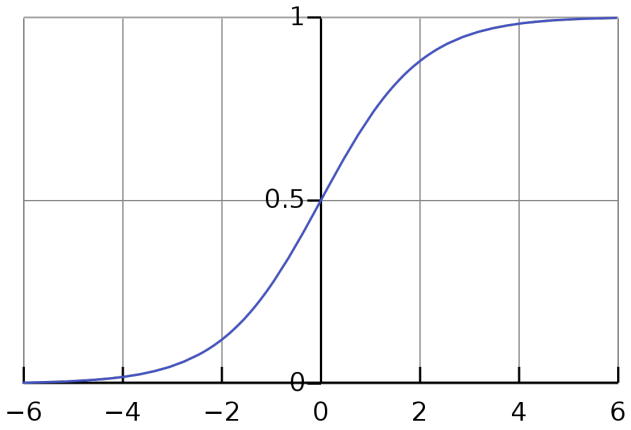
Η συνάρτηση ενεργοποίησης των νευρώνων ενός δικτύου ορίζει τον τρόπο με τον οποίο το σύνολο των εισόδων του κάθε κόμβου μετατρέπεται στην έξοδο του. Η επιλογή της συνάρτησης ενεργοποίησης έχει σημαντική επίπτωση στην απόδοση του νευρωνικού δικτύου. Τυπικά, κάθε επίπεδο του δικτύου χρησιμοποιεί την ίδια συνάρτηση ενεργοποίησης, η οποία είναι συνήθως μη γραμμική και αντιστοιχίζει την είσοδο σε μια τιμή εντός ενός φραγμένου πεδίου τιμών. Στην περίπτωση των perceptron πολλαπλών επιπέδων, το μοντέλο κάθε νευρώνα στο δίκτυο θα πρέπει να περιλαμβάνει μια μη γραμμική συνάρτηση ενεργοποίησης, η οποία είναι διαφορίσιμη. Σε αντίθετη περίπτωση, μπορεί να αποδειχθεί αλγεβρικά πως οποιοσδήποτε αριθμός στρωμάτων, μπορεί να μειωθεί σε δίκτυο δύο στρωμάτων. Μερικές από τις βασικότερες συναρτήσεις ενεργοποίησης παρουσιάζονται παρακάτω.

- *Σιγμοειδής Συνάρτηση (Sigmoid Function):* Η σιγμοειδής συνάρτηση, ονομαζόμενη από τη χαρακτηριστική της μορφή η οποία φαίνεται και στο σχήμα 3.4, αποτελεί μια από τις πρώτες συναρτήσεις ενεργοποίησης που χρησιμοποιήθηκαν στον τομέα των νευρωνικών δικτύων. Μαθηματικά, υπολογίζεται ως:

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

  Όπως προκύπτει και από παραπάνω, η συγκεκριμένη συνάρτηση αντιστοιχεί κάθε τιμή εισόδου σε μία τιμή στο διάστημα (0, 1), με τις μικρότερες τιμές να πλησιάζουν στο 0, και τις μεγαλύτερες να πλησιάζουν στο 1, δίχως όμως να παίρνει ποτέ η συνάρτηση τις τιμές αυτές. Οι μικρές αυτές τιμές όμως, που αντιστοιχίζονται και σε μικρές τιμές κλίσης, είναι πιθανό να δημιουργήσουν πρόβλημα στη διαδικασία της εκπαίδευσης, ιδίως σε περιπτώσεις όπου χρησιμοποιείται η οπισθοδιάδοση(Backpropagation), στην οποία θα γίνει αναφορά παρακάτω, καθώς οι τιμές αυτές πολλαπλασιάζονται διαδοχικά μεταξύ τους, με αποτέλεσμα η τελική τιμή της κλίσης να είναι μικροσκοπική, τα βάρη να μην αλλάζουν αποτελεσματικά τις τιμές τους και η διαδικασία της εκπαίδευσης να γίνεται πολύ αργή ή ακόμη και να σταματά. Το συγκεκριμένο πρόβλημα ονομάζεται το πρόβλημα εξασθένισης κλίσης (Vanishing Gradient Problem) και εμφανίζεται και κατά τη

χρήση διαφορετικών συναρτήσεων ενεργοποίησης.

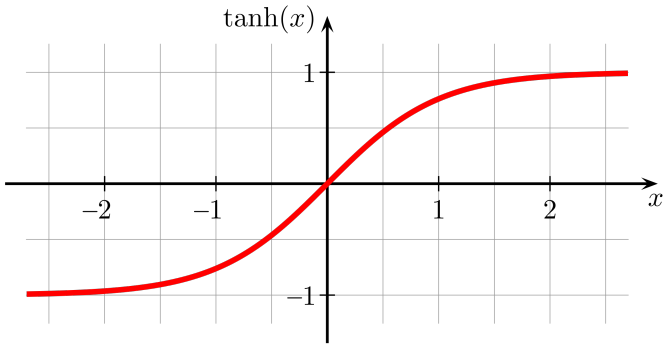


Σχήμα 3.4: Σιγμοειδής συνάρτηση

- *Υπερβολική Εφαπτομένη (Hyperbolic Tangent):* Η συνάρτηση αυτή, η οποία φαίνεται και στο σχήμα 3.5, εμφανίζει αρκετές ομοιότητες με τη σιγμοειδή συνάρτηση, αντιστοιχίζοντας τις τιμές εισόδου στο διάστημα (-1, 1). Μαθηματικά, υπολογίζεται ως:

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

Το πλεονέκτημα που παρουσιάζει η συγκεκριμένη συνάρτηση σε σχέση με τη σιγμοειδή, είναι πως οι τιμές που βρίσκονται κοντά στο 0 δεν μεταβάλλονται σημαντικά, με αποτέλεσμα να συμβάλλουν και αυτές στη διαδικασία της εκπαίδευσης. Παρόλα αυτά, και στη συγκεκριμένη περίπτωση οι πολύ μεγάλες και πολύ μικρές τιμές αντιστοιχίζονται στο +1 και -1 αντίστοιχα, με αποτέλεσμα να εμφανίζεται εκ νέου το πρόβλημα εξασθένισης κλίσης.



Σχήμα 3.5: Υπερβολική Εφαπτομένη

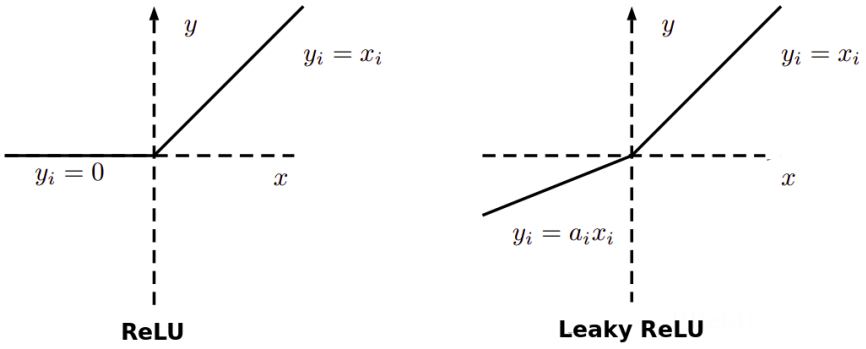
- *Rectified Linear Unit (ReLu) & Leaky ReLu:* Η συγκεκριμένη συνάρτηση, αν και απλή, αποτελεί μία από τις πλέον δημοφιλείς συναρτήσεις ενεργοποίησης[38]. Όπως φαίνεται και στο σχήμα 3.6(α), με τη χρήση της συνάρτησης οι θετικές τιμές δεν μεταβάλλονται, ενώ οι τιμές που είναι μικρότερες του μηδενός μηδενίζονται και κατ΄ επέκταση δεν συνυπολογίζονται στη διαδικασία της εκπαίδευσης. Μαθηματικά, υπολογίζεται ως:

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

Η ιδιότητα της ReLu να μηδενίζει τις μη θετικές τιμές της, καθιστά τους νευρώνες του δικτύου με αρνητική έξοδο ανενεργούς, με συνέπεια τη μείωση του χρόνου εκπαίδευσης και την αύξηση της αποδοτικότητας του μοντέλου. Παρόλα αυτά, η ιδιότητα αυτή δημιουργεί και προβλήματα, καθώς οι νευρώνες που λαμβάνουν αρνητική τιμή κατά τη διάρκεια της εκπαίδευσης πρακτικά απενεργοποιούνται και δεν χρησιμοποιούνται καθόλου. Το πρόβλημα αυτό ονομάζεται "Dying ReLu" problem, και για την επίλυσή του έχει προταθεί μια

παραλλαγή της ReLu, ονομαζόμενη Leaky ReLu(Σχήμα 3.6(β)). Η συγκεκριμένη συνάρτηση αντιστοιχίζει τις αρνητικές τιμές σε κάποια γραμμική συνάρτηση πολύ μικρής κλίσης, δίνοντας έτσι τη δυνατότητα στους νευρώνες που λαμβάνουν αρνητικές τιμές να επανέλθουν(recover). Μαθηματικά, υπολογίζεται ως:

$$f(x) = max(ax, x), \text{ όπου } a \text{ η τιμή της κλίσης}$$



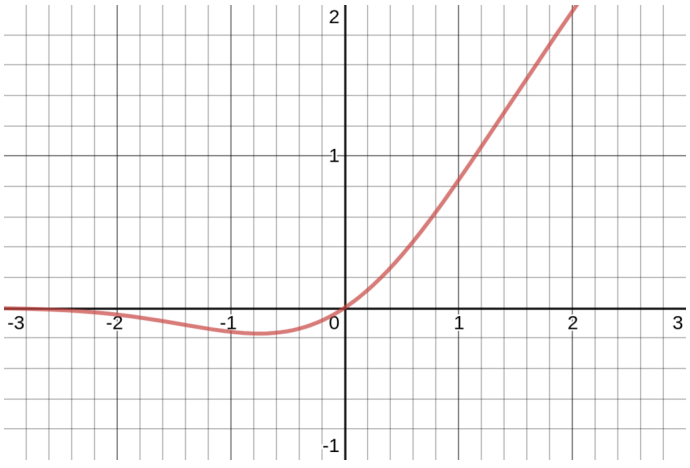
Σχήμα 3.6: ReLu(α) & Leaky ReLu(β)

- *Gaussian Error Linear Unit (GELU):*

  Μια ακόμη συνάρτηση ενεργοποίησης αποτελεί η GELU (Gaussian Error Linear Unit)[39]. Η συνάρτηση αυτή, πρακτικά πολλαπλασιάζει την είσοδο με τη συνάρτηση πυκνότητας της κανονικής κατανομής στην είσοδο αυτή, ενώ μαθηματικά αποτυπώνεται ως:

  $$f(x) = x * \Phi(x)$$

  Σε πολλές περιπτώσεις, λόγω του χρόνου που απαιτείται για τον ακριβή υπολογισμό της τιμής της, χρησιμοποιούνται προσεγγιστικοί τρόποι υπολογισμού της, ακριβείς έως και το 4ο δεκαδικό ψηφίο. Η μορφή της φαίνεται στο σχήμα 3.7. Η συγκεκριμένη συνάρτηση αποφεύγει πλήρως το πρόβλημα της εξασθένισης κλίσης, ενώ , παρόλο που εισήχθη μόλις το 2016, θεωρείται πλέον η βέλτιστη συνάρτηση ενεργοποίησης σε προβλήματα επεξεργασίας φυσικής γλώσσας.



Σχήμα 3.7: GELU

- *Συνάρτηση Softmax:* Η συγκεκριμένη συνάρτηση χρησιμοποιείται συνήθως ως η τελική συνάρτηση ενεργοποίησης κάποιου νευρωνικού δικτύου και όχι σε ενδιάμεσους κόμβους με σκοπό την ομαλοποίηση της εξόδους σε κατανομή πιθανότητας των κλάσεων εξόδου. Χρησιμοποιείται δηλαδή σε προβλήματα κατηγοριοποίησης με περισσότερες από δύο κλάσεις ταξινόμησης, όπου, όπως αναφέρθηκε και παραπάνω, η τιμή

εξόδου κάθε νευρώνα υποδεικνύει την πιθανότητα τα δεδομένα εισόδου να ανήκουν στην κλάση που του αντιστοιχεί. Η συνάρτηση Softmax πρακτικά διαμορφώνει το σύνολο των τιμών αυτών ώστε το άθροισμά τους να ισούται με 1 και να εκφράζονται με κατάλληλο τρόπο οι πιθανότητες που αντιστοιχούν σε κάθε κλάση. Μαθηματικά, υπολογίζεται ως:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{z_j}},\ i = 1, 2, ..., \text{K και } z = (z_1, z_2, ..., z_K)$$

3.2.2.2 Συνάρτηση Κόστους (Cost Function)

Η συνάρτηση κόστους, ή αλλιώς συνάρτηση απώλειας (Loss Function), αποτελεί μια σημαντική παράμετρο ενός νευρωνικού δικτύου, καθώς χρησιμοποιείται ως μετρική της διαδικασίας της εκπαίδευσης σε ένα σύνολο δεδομένων. Πρακτικά υπολογίζει την απόκλιση μεταξύ της αναμενόμενης(πραγματικής) και της υπολογιζόμενης από το μοντέλο τιμής εξόδου και την αναπαριστά με έναν πραγματικό αριθμό. Συνήθως συμβολίζεται ως $J(\theta)$ και είναι η κύρια μέθοδος αξιολόγησης της διαδικασίας εκπαίδευσης κατά την επιβλεπόμενη μάθηση. Παρακάτω αναφέρονται συνοπτικά μερικές από τις συχνότερα χρησιμοποιούμενες συναρτήσεις κόστους.

- *Απώλεια Εγκάρσιας Εντροπίας (Cross Entropy Loss):* Συνήθως χρησιμοποιείται για την αξιολόγηση της επίδοσης ενός μοντέλου ταξινόμησης, όπου η έξοδος είναι μια πιθανότητα με τιμή από 0 έως 1. Στην περίπτωση περισσότερων από δύο κλάσεων, αποτυπώνεται μαθηματικά ως εξής:

  $$J(\theta) = -\frac{1}{m} \sum_{c=1}^{m} y_{o,c} \log(p_{o,c})$$

  , όπου M ο αριθμός των κλάσεων ταξινόμησης, y ένας δυαδικός(0 ή 1) δείκτης που υποδεικνύει εάν η είσοδος o ανήκει στην κλάση c και p η προβλεπόμενη πιθανότητα να ανήκει η είσοδος o στην κλάση c.

- *Μέσο Τετραγωνικό Σφάλμα (Mean Squared Error):* Συνήθως χρησιμοποιείται σε προβλήματα παλινδρόμησης, και υπολογίζει τη μέση τιμή του τετραγώνου της διαφοράς μεταξύ της πραγματικής και της προβλεπόμενης τιμής. Μαθηματικά, υπολογίζεται ως εξής:

  $$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2$$

- *Μέσο Απόλυτο Σφάλμα (Mean Absolute Error):* Η συγκεκριμένη συνάρτηση, όπως και η παραπάνω, χρησιμοποιείται συνήθως σε προβλήματα παλινδρόμησης, υπολογίζει όμως τη μέση απόλυτη τιμή της διαφοράς μεταξύ της πραγματικής και της προβλεπόμενης τιμής. Η συγκεκριμένη συνάρτηση χρησιμοποιείται σπανιότερα σε σχέση με τη συνάρτηση μέσου τετραγωνικού σφάλματος, προτιμάται όμως όταν τα δεδομένα εισόδου περιέχουν πολλαπλές "ακραίες" τιμές, καθώς επηρεάζεται σε μικρότερο βαθμό από αυτές. Μαθηματικά, υπολογίζεται ως εξής:

  $$J(\theta) = \frac{1}{n} \sum_{i=1}^{D} |x_i - y_i|$$

3.2.2.3 Αλγόριθμος Βελτιστοποίησης (Optimization Algorithm)

Ο αλγόριθμος βελτιστοποίησης σε ένα νευρωνικό δίκτυο, είναι υπεύθυνος για την εύρεση των βέλτιστων βαρών των κόμβων για τη μείωση του σφάλματος κατά την αντιστοίχηση εισόδων σε εξόδους, δηλαδή την ελαχιστοποίηση της τιμής της συνάρτησης κόστους. Όπως είναι εμφανές, αποτελεί αναπόσπαστο κομμάτι κάθε δικτύου, καθώς έχει σημαντικές επιπτώσεις τόσο στην ακρίβεια, όσο και στην ταχύτητα εκπαίδευσης του μοντέλου.

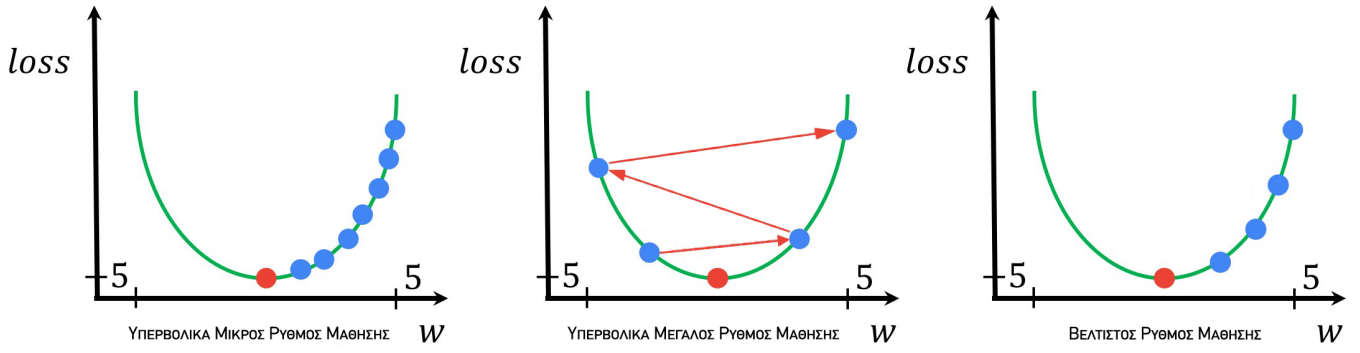
Οι αλγόριθμοι αυτοί, συχνά κάνουν χρήση της τιμής της κλίσης, δηλαδή της μερικής παραγώγου, της συνάρτησής κόστους ως προς το βάρος και την πόλωση κάθε κόμβου του δικτύου. Η συχνότερα χρησιμοποιούμενη μέθοδος για τον υπολογισμό των παραγώγων αυτών, είναι η μέθοδος της οπισθοδιάδοσης (Backpropagation). Πρακτικά, πρόκειται για μια οικογένεια μεθόδων με κύριο χαρακτηριστικό την αναδρομική τους λειτουργία, που τους επιτρέπει να υπολογίσουν με αποδοτικό τρόπο τις ζητούμενες τιμές. Οι αλγόριθμοι αυτοί έπονται του υπολογισμού της εξόδου από το δίκτυο και λειτουργούν διαδίδοντας πληροφορία από το στρώμα εξόδου προς το στρώμα εισόδου, κάνοντας χρήση του κανόνα αλυσίδας ώστε να βρουν την επιρροή κάθε μεταβλητής στην τελική συνάρτηση κόστους.

Ένας από τους αρχικούς και γνωστότερους αλγόριθμους βελτιστοποίησης, είναι ο αλγόριθμος καθόδου με βάση την κλίση (Gradient Descent). Ο συγκεκριμένος αλγόριθμος, και συγκεκριμένα οι παραλλαγές του, είναι οι συχνότερα χρησιμοποιούμενες μέθοδοι για τη βελτιστοποίηση νευρωνικών δικτύων. Στην αρχική του μορφή, βασίζεται στην ελαχιστοποίηση της συνάρτησης κόστους μέσω της κλίσης των παραμέτρων του προβλήματος, στην προκειμένη περίπτωση των βαρών και της πόλωσης του μοντέλου. Η διαδικασία επαναλαμβάνεται με τις ανανεώσεις να είναι αντίθετες της κατεύθυνσης της κλίσης, μέχρι να υπάρξει σύγκλιση, δηλαδή να βρεθεί κάποιο τοπικό ελάχιστο της συνάρτησης. Οι παράμετροι του δικτύου σε κάθε βήμα ανανεώνονται με βάση τη σχέση:

$$\theta_t + 1 = \theta_t - \lambda \nabla_\theta J(\theta)$$

, όπου ϑ το σύνολο των παραμέτρων, λ ο ρυθμός μάθησης και $J(\theta)$ η συνάρτηση κόστους

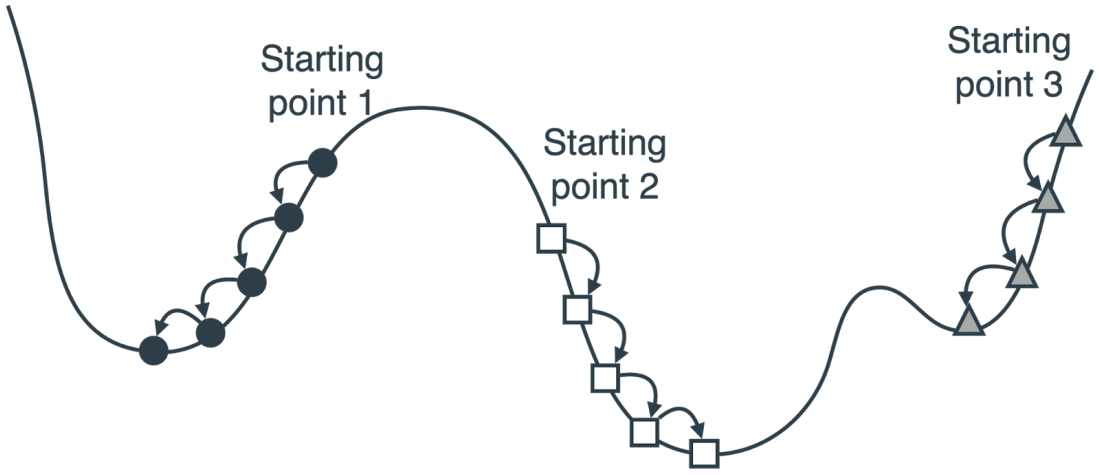
Όσον αφορά τις παραλλαγές του συγκεκριμένου αλγορίθμου, οι διαφοροποιήσεις που συναντώνται αφορούν το πλήθος των δεδομένων που χρησιμοποιείται για τον υπολογισμό των κλίσεων, ανταλλάσσοντας με τον τρόπο αυτό την ακρίβεια της κάθε ενημέρωσης των παραμέτρων με τον απαιτούμενο χρόνο για να ολοκληρωθεί. Μια συχνά χρησιμοποιούμενη παραλλαγή είναι ο αλγόριθμος στοχαστικής καθόδου με βάση την κλίση (Stochastic Gradient Descent). Στον αλγόριθμο αυτό, αντί να γίνεται χρήση ολόκληρου του δείγματος, επιλέγεται τυχαία είτε μια μεμονωμένη τιμή του δείγματος εισόδου, είτε ένα μικρό σύνολο από αυτές (ονομαζόμενο mini-batch) αντικαθιστώντας την πραγματική κλίση με μια εκτίμηση αυτής. Με τον τρόπο αυτό, μειώνεται κατακόρυφα το πλήθος των απαιτούμενων υπολογισμών, και κατ΄ επέκταση ο χρόνος για να ολοκληρωθεί κάθε επανάληψη, ιδίως σε περιπτώσεις όπου το σύνολο των δεδομένων είναι μεγάλο, με σχετικά μικρό αντίκτυπο στην ακρίβεια και τον ρυθμό σύγκλισης.



Σχήμα 3.8: Επιρροή του ρυθμού μάθησης

Βεβαίως, οι αλγόριθμοι αυτοί παρουσιάζουν ορισμένα προβλήματα. Για παράδειγμα, η τιμή του σταθερού ρυθμού μάθησης μπορεί είτε να οδηγήσει στην απόκλιση εάν είναι πολύ μεγάλη, είτε να κάνει το μοντέλο να συγκλίνει με πολύ αργό ρυθμό στην περίπτωση που είναι πολύ μικρή, όπως φαίνεται και στο σχήμα 3.8. Ακόμη, στην αρχική του μορφή, ενώ ο αλγόριθμος είναι ικανός να βρίσκει κάποιο τοπικό ελάχιστο της συνάρτησης κόστους, δεν έχει τρόπο να εξασφαλίσει πως αυτό είναι και το ολικό ελάχιστο της. Για τον λόγο αυτό, σε ορισμένα προβλήματα το τελικό αποτέλεσμα εξαρτάται από τις αρχικές τιμές που δόθηκαν στα βάρη των κόμβων και δεν είναι βέλτιστο, όπως φαίνεται και στο σχήμα 3.9.

Λόγω των προβλημάτων αυτών, έχουν προταθεί περαιτέρω παραλλαγές στους παραπάνω αλγόριθμους. Ένα τέτοιο παράδειγμα είναι και ο αλγόριθμος ADAM (Adaptive Moment Estimation)[40], ο οποίος χρησιμοποιεί μεταβλητό ρυθμό μάθησης, ξεχωριστό μάλιστα για κάθε κόμβο του δικτύου, ενώ κάνει χρήση και της δεύτερης παραγώγου, ή αλλιώς της δεύτερης βαθμίδας κλίσης της συνάρτησης. Μέσω των παραπάνω, επιταχύνεται αισθητά η διαδικασία της σύγκλισης, μετριάζονται όλοι οι κίνδυνοι σχετικά με τον ρυθμό μετάδοσης, ενώ παράλληλα αποφεύγεται πλήρως ο κίνδυνος εγκλωβισμού σε κάποιο τοπικό ελάχιστο.



Σχήμα 3.9: Παράδειγμα τοπικών ελαχίστων που δεν ταυτίζονται με το ολικό ελάχιστο

3.2.2.4 Κανονικοποίηση Μοντέλου(Model Regularization)

Ένα πρόβλημα που είναι πιθανό να προκύψει κατά την εκπαίδευση νευρωνικών δικτύων, είναι αυτό της υπέρ-προσαρμογής (Overfitting). Όταν παρουσιάζεται το πρόβλημα αυτό, το μοντέλο έχει εκπαιδευτεί έτσι ώστε να ευθυγραμμίζεται υπερβολικά με τα δεδομένα εισόδου. Η υπέρ-προσαρμογή προκύπτει εν γένει λόγω του θορύβου που περιέχεται σε κάθε σύνολο δεδομένων, καθώς τα σύνθετα νευρωνικά δίκτυα έχουν τη δυνατότητα να εντοπίζουν τις συσχετίσεις και μοτίβα που προκύπτουν από τον θόρυβο αυτό. Το γεγονός αυτό τα καθιστά πλήρως ακριβή σε προβλέψεις στο αρχικό σύνολο δεδομένων, μη αποδοτικά όμως σε διαφορετικά σύνολα δεδομένων. Για τον λόγο αυτό έχουν προταθεί διάφοροι τρόποι κανονικοποίησης για την αποφυγή της υπέρ-προσαρμογής, μερικούς από τους οποίους θα αναπτύξουμε εν συντομία παρακάτω.

Αρχικά, θα αναφερθούμε στις δύο βασικότερες τεχνικές κανονικοποίησης, την κανονικοποίηση ράχης και την κανονικοποίηση Lasso. Στις δύο τεχνικές αυτές, τροποποιείται η συνάρτηση κόστους του μοντέλου.

- *Κανονικοποίηση Ράχης(Ridge Regularisation):* Κατά την κανονικοποίηση ράχης, προστίθεται στην τιμή της συνάρτησης κόστους το άθροισμα των τετραγώνων όλων των βαρών του μοντέλου, πολλαπλασιασμένο με κάποια ποινή λάθους λ. Η αύξηση της ποινής επιφέρει μείωση της σημασίας των συντελεστών και κατά επέκταση μείωση της πολυπλοκότητας του μοντέλου. Μαθηματικά, αποτυπώνεται ως:

$$J(\theta)' = J(\theta) + \lambda \sum ||w||^2$$

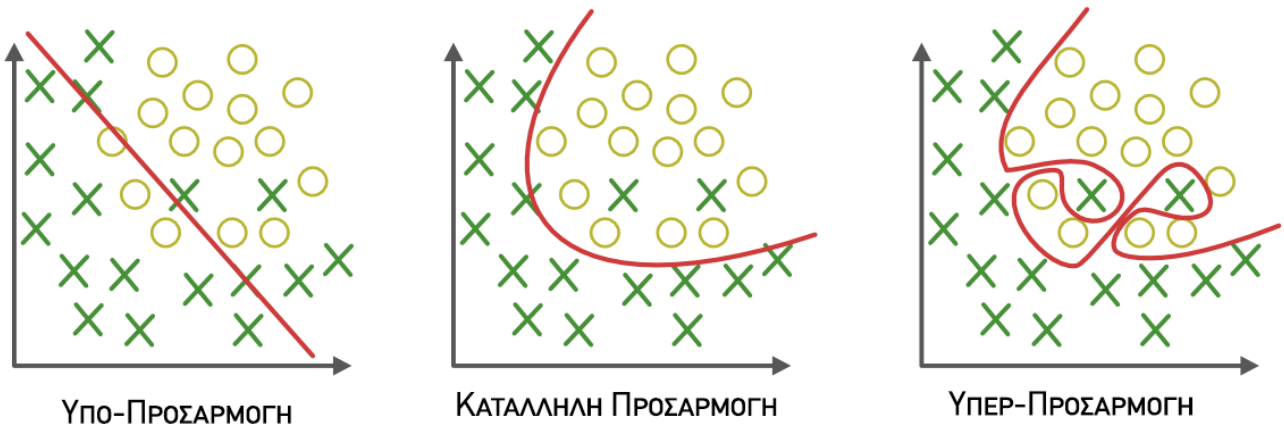
- *Κανονικοποίηση Lasso (Lasso Regularisation):* Κατά την κανονικοποίηση Lasso, προστίθεται στην τιμή της συνάρτησης κόστους το άθροισμα των απόλυτων τιμών όλων των βαρών του μοντέλου, πολλαπλασιασμένο με κάποια ποινή λάθους λ. Μαθηματικά, αποτυπώνεται ως:

$$J(\theta)' = J(\theta) + \lambda \sum ||w||$$

  Στις παραπάνω περιπτώσεις, αναφέραμε πως η αύξηση του συντελεστή λ επιφέρει και μείωση της πολυπλοκότητας του μοντέλου. Η υπερβολική του αύξηση όμως, καθιστά το μοντέλο ανίκανο να αποτυπώσει τις σχέσεις μεταξύ των δεδομένων εισόδου και εξόδου με ακριβή τρόπο. Στην περίπτωση αυτή, λέμε πως το μοντέλο είναι υπό-προσαρμοσμένο (Underfitted). Ένα παράδειγμα υπέρ-προσαρμοσμένου και υπό-προσαρμοσμένου μοντέλου φαίνεται στο σχήμα 3.10

Πέραν της παραπάνω τεχνικής κανονικοποίησης, υπάρχουν και άλλες μέθοδοι ώστε να αποφευχθεί η υπέρ-προσαρμογή ενός δικτύου.

- *Ομαλοποίηση & κλιμάκωση δεδομένων(Scaling & Normalisation):* Σε ένα σύνολο δεδομένων, είναι πολύ πιθανό οι τιμές να διαφέρουν αρκετές τάξεις μεγέθους μεταξύ τους. Καθώς όμως η προσαρμογή των βαρών του δικτύου κατά τη διάρκεια της εκπαίδευσης εξαρτάται από την τιμή των εισόδων, κάποια βάρη θα ενημερωθούν με πολύ μεγαλύτερο ρυθμό από κάποια άλλα, δυσκολεύοντας τη σύγκλιση σε μια βέλτιστη



Σχήμα 3.10: Προσαρμογή ενός δικτύου σε δεδομένα

τιμή. Για το λόγο αυτό, συχνά τα δεδομένα εισόδου ομαλοποιούνται ή κλιμακώνονται. Η κλιμάκωση αφορά την προσαρμογή των δεδομένων ώστε να ανήκουν σε ένα συγκεκριμένο εύρος, ενώ παράλληλα είναι χρήσιμη και σε περιπτώσεις όπου χρησιμοποιούνται διαφορετικές μονάδες μέτρησης στα δεδομένα. Κατά την ομαλοποίηση, τα δεδομένα προσαρμόζονται ώστε να ακολουθούν κανονική κατανομή, δηλαδή την κατανομή του Gauss. Γενικά, η ομαλοποίηση χρησιμοποιείται σε περιπτώσεις όπου το μοντέλο αναμένει η είσοδος να ακολουθεί κανονική κατανομή, μπορεί όμως να προσφέρει πλεονεκτήματα όσον αφορά τον χρόνο σύγκλισης στα περισσότερα μοντέλα.

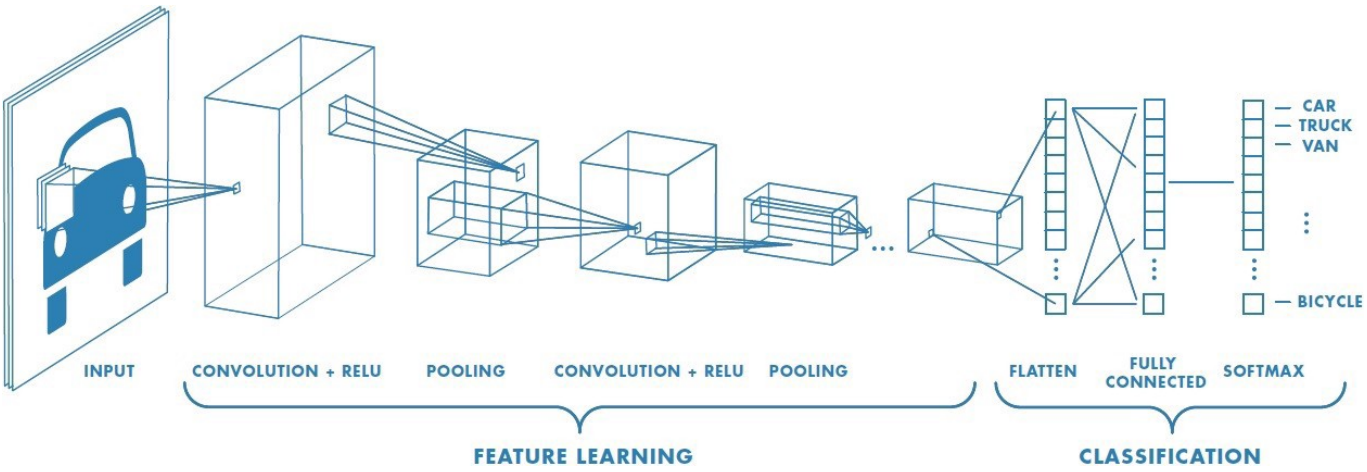
- *Στρώμα Απόσυρσης(Dropout Layer)* Η απόσυρση, είναι μια τεχνική κανονικοποίησης δικτύου που προσομοιώνει την παράλληλη εκπαίδευση ενός μεγάλου αριθμού μοντέλων με διαφορετικές αρχιτεκτονικές. Κατά τη διαδικασία της εκπαίδευσης, στο στρώμα απόσυρσης διάφοροι κόμβοι αποσύρονται, ή αλλιώς προσωρινά αγνοούνται, μέσω του μηδενισμού των βαρών τους. Με τον τρόπο αυτό, το συγκεκριμένο στρώμα νευρώνων μπορεί να συμπεριφέρεται και να αντιμετωπίζεται ως ένα στρώμα με διαφορετικό αριθμό κόμβων και συνδεσιμότητας με το προηγούμενο επίπεδο. Η διαδικασία αυτή πρακτικά προσδίδει μεταβλητότητα και θόρυβο στη διαδικασία της εκπαίδευσης, ανεξάρτητο της αρχικής εισόδου. Η συγκεκριμένη τεχνική εξαλείφει ορισμένες εξαρτήσεις μεταξύ των βαρών των κόμβων και των στρωμάτων του δικτύου, οδηγώντας σε μια συνάρτηση εξόδου που θα γενικεύεται καλύτερα για διαφορετικές εισόδους.

- *Κανονικοποίηση Στρώματος(Layer Normalisation)* Η κανονικοποίησης στρωμάτων αποτελεί μια τεχνική που χρησιμοποιείται σε Αναδρομικά Νευρωνικά Δίκτυα, καθώς και δίκτυα Transformer με σκοπό τη βελτίωση του τρόπου που γενικεύουν, καθώς και την επιτάχυνση της διαδικασίας εκπαίδευσής τους[41]. Η τεχνική αυτή λειτουργεί εντός ενός κρυφού στρώματος, και εκτιμά τα στατιστικά κανονικοποίησης των εισόδων κάθε νευρώνα του προηγούμενου στρώματος ξεχωριστά. Η κανονικοποίηση αυτή πρακτικά εφαρμόζεται σε κάθε χαρακτηριστικό ξεχωριστά ώστε να αυξηθεί η σταθερότητα του δικτύου κατά τη διάρκεια της εκπαίδευσης.

- *Πρόωρη Διακοπή(Early Stopping):* Τέλος, θα αναφερθούμε και σε μία από τις απλούστερες μεθόδους ώστε να αποφευχθεί η υπέρ-προσαρμογή, αυτή της πρόωρης διακοπής. Όπως υποδεικνύει και η ονομασία της, αφορά την πρόωρη διακοπή της διαδικασίας της εκπαίδευσης. Η διακοπή αυτή επέρχεται αφότου ολοκληρωθούν ορισμένες συνθήκες που έχουν οριστεί εξαρχής και συνήθως αφορούν διάφορες μετρικές απόδοσης, όπως η ακρίβεια ή η απώλεια.

3.3 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

Τα συνελικτικά νευρωνικά δίκτυα (CNN), γνωστά και ως ConvNet, αποτελούν μια κατηγορία νευρωνικών δικτύων ικανά να επεξεργαστούν με αποδοτικό τρόπο χωρικά εξαρτώμενες εισόδους, με χαρακτηριστικότερο παράδειγμα τις εικόνες[42]. Πρακτικά πρόκειται για μια κανονικοποιημένη εκδοχή του Perceptron πολλαπλών επιπέδων. Για να επιτύχουν την κανονικοποίηση αυτή, χρησιμοποιούν απλά μοτίβα εντός των δεδομένων για να συναρμολογήσουν μοτίβα αυξημένης πολυπλοκότητας, αποφεύγοντας με τον τρόπο αυτό το φαινόμενο της υπέρ-προσαρμογής που συχνά συνοδεύει πλήρως συνδεδεμένα δίκτυα.
Ένα συνελικτικό νευρωνικό δίκτυο, τυπικά αποτελείται από τρία επίπεδα, το στρώμα συνέλιξης (Convolution Layer), το στρώμα ομαδοποίησης (Pooling Layer) και ένα πλήρως συνδεδεμένο στρώμα (Fully Connected Layer). Μια αναπαράσταση ενός τέτοιου δικτύου φαίνεται στο σχήμα 3.11. Το στρώμα συνέλιξης, που αποτελεί και το κυριότερο δομικό στοιχείο ενός CNN, εκτελεί εσωτερικά γινόμενα μεταξύ δύο πινάκων. Ο πρώτος πίνακας αποτελείται από ένα σύνολο εκπαιδεύσιμων παραμέτρων και ονομάζεται kernel, ενώ ο δεύτερος είναι ο πίνακας εισόδου του στρώματος και αποτελεί ένα υποσύνολο των δεδομένων εισόδου. Το kernel, αν και μικρότερο σε μήκος και πλάτος σε σχέση με την είσοδο, έχει το ίδιο βάθος, δηλαδή το ίδιο πλήθος καναλιών (channels). Για παράδειγμα, στην περίπτωση μιας εικόνας, όπου το βάθος εξαρτάται από τη χρωματική κωδικοποίηση της, θα χρησιμοποιηθούν τρία κανάλια εάν η κωδικοποίηση αυτή είναι σε RGB. Κατά την εμπρόσθια τροφοδότηση το kernel διασχίζει την εικόνα κατά ύψος και πλάτος με συγκεκριμένο βήμα (stride) οπότε παράγεται ένας πίνακας δύο διαστάσεων, ονομαζόμενος χάρτης ενεργοποίησης (Activation Map). Στη συνέχεια, το στρώμα ομαδοποίησης αντικαθιστά ορισμένες από τις εξόδους του δικτύου με τη μέση τιμή των γειτονικών κόμβων, με σκοπό τη μείωση του υπολογιστικού φόρτου. Τέλος, υπάρχει ένα πλήρως συνδεδεμένο στρώμα, το οποίο χρησιμοποιείται για την αντιστοίχηση εισόδου και εξόδου.



Σχήμα 3.11: Δομή ενός CNN

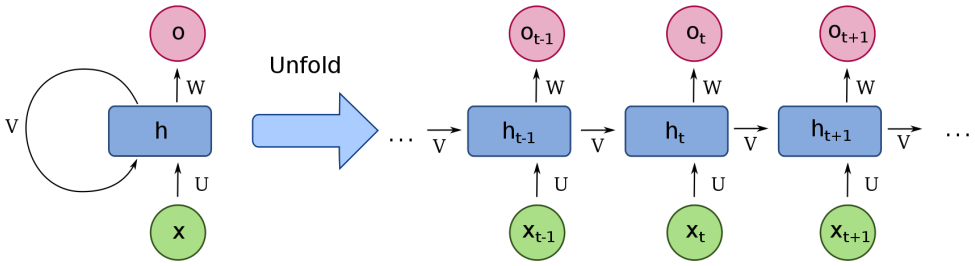
Source: `https://bit.ly/2YB9o0H`

3.4 Επεξεργασία Ακολουθιακών Δεδομένων Εισόδου με Τ.Ν.Δ.

Τα είδη των τεχνητών νευρωνικών δικτύων που έχουμε αναπτύξει μέχρι στιγμής, αν και μπορούν να αντιμετωπίσουν επιτυχώς πληθώρα προβλημάτων, δεν είναι σχεδιασμένα για να αντιμετωπίζουν δεδομένα εισόδου που είναι διατεταγμένα με συγκεκριμένη σειρά. Παρόλο που έχουν υπάρξει προσεγγίσεις στο συγκεκριμένο πρόβλημα με τη χρήση ορισμένων συνελικτικών δικτύων μονοδιάστατης εισόδου, η αρχιτεκτονική υποστήριξη για την επίλυση των συγκεκριμένων προβλημάτων με χρήση νευρωνικών δικτύων επήλθε με την εισαγωγή της ιδιότητας της μνήμης σε αυτά.

3.4.1 Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks)

Τα αναδρομικά νευρωνικά δίκτυα, συνιστούν μια κατηγορία νευρωνικών δικτύων όπου οι συνδέσεις μεταξύ των κόμβων μπορούν να δημιουργούν βρόγχους. Η ιδιότητα αυτή, επιτρέπει σε κόμβους να κρατούν προηγούμενες πληροφορίες τις οποίες μπορούν έπειτα να χρησιμοποιήσουν σε συνδυασμό με τις επόμενες εισόδους τους στις προσεχείς προβλέψεις τους, ενσωματώνοντας εξαρτήσεις μεταξύ των δεδομένων[43, 44]. Το γεγονός αυτό τα καθιστά χρήσιμα στην επίλυση προβλημάτων που περιέχουν ακολουθιακά δεδομένα, όπως είναι η αναγνώριση ομιλίας και γραφής ή η μετάφραση κειμένου, προβλήματα δηλαδή όπου τα δεδομένα εισόδου εξαρτώνται από τις ακριβώς προηγούμενες εισόδους και τα οποία δυσκολεύονται να αντιμετωπίσουν τα δίκτυα εμπρόσθιας τροφοδότησης.



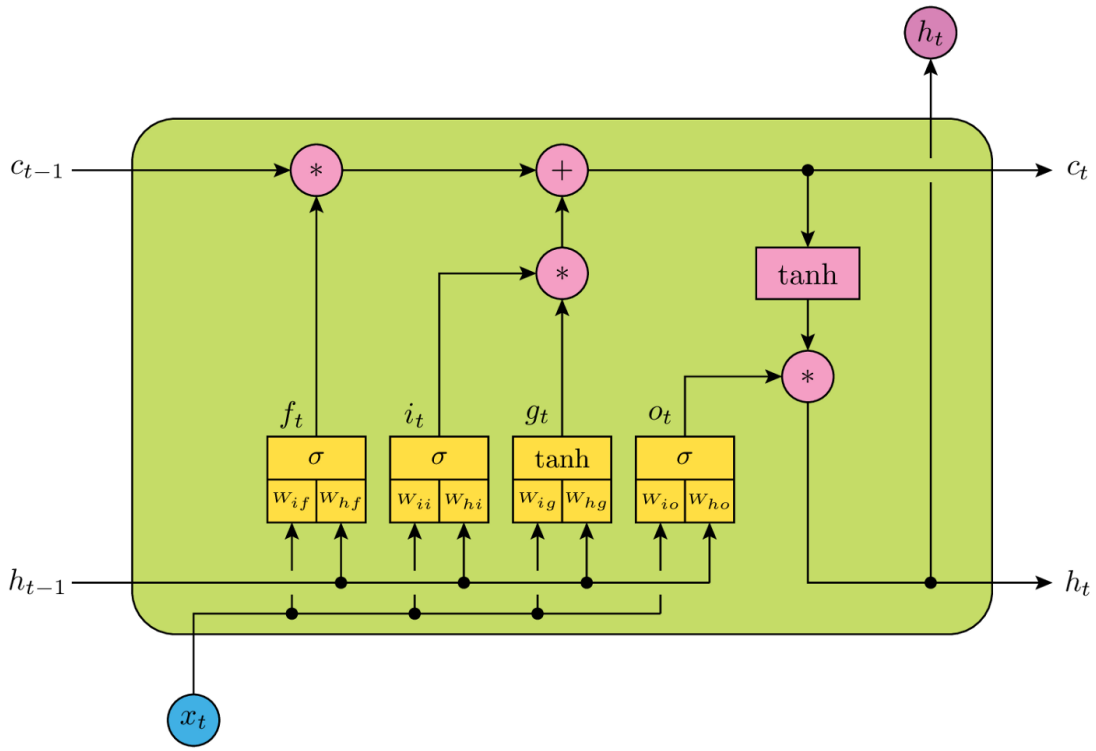
Σχήμα 3.12: Λειτουργία ενός RNN

Source: `https://en.wikipedia.org/wiki/File:Recurrent_neural_network_unfold.svg`

Μια αναπαράσταση ενός αναδρομικού νευρωνικού δικτύου φαίνεται στο σχήμα 3.12, όπου θεωρούμε πως το δίκτυο λαμβάνει ως είσοδο μια ακολουθία διανυσμάτων $[x_1, x_2, ..., x_t]$ στο χρονικό βήμα t, και θεωρούμε πως η μνήμη του δικτύου περνάει στην επόμενη κατάσταση ως v_t. Για την ανανέωση των βαρών στο δίκτυο, χρησιμοποιείται ο αλγόριθμος της οπισθοδιάδοσης στο χρόνο (Backpropagation through time). Ο αλγόριθμος αυτός, λειτουργεί με τρόπο όμοιο με τον αλγόριθμο οπισθοδιάδοσης, ενώ παρουσιάζεται και εδώ το πρόβλημα της εξασθένισης κλίσης, το οποίο γίνεται εντονότερο με την αύξηση των στρωμάτων και της πολυπλοκότητας του δικτύου. Σε πρακτικό επίπεδο, το πρόβλημα αυτό μεταφράζεται σε αδυναμία του δικτύου να λάβει υπόψιν μακροπρόθεσμες εξαρτήσεις. Όπως είναι εμφανές, το πρόβλημα αυτό είναι σημαντικό στις περιπτώσεις όπου χρησιμοποιούνται τα συγκεκριμένα νευρωνικά δίκτυα, με παράδειγμα την επεξεργασία φυσικής γλώσσας όπου είναι πιθανή η εξάρτηση από κάποια λέξη αρκετά βήματα πίσω.

3.4.2 Δίκτυα Μακράς Βραχυπρόθεσμης Μνήμης (LSTM)

Την λύση στα προβλήματα που αναφέρθηκαν παραπάνω επιχείρησε να δώσει η εφεύρεση των Hochreiter & Schmidhuber το 1997[45], τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (Long Short-Term Memory). Τα δίκτυα αυτά αντλούν το όνομα τους από την ιδιότητα τους να επεκτείνουν τη δυνατότητα βραχυπρόθεσμης μνήμης των επαναλαμβανόμενων δικτύων. Σε αντίθεση με τα απλά RNN, τα LSTM αναθέτουν στα δεδομένα επιπρόσθετα βάρη, υπό τη μορφή μιας κρυφής κατάστασης κελιού, τα οποία μεταβάλλονται κατά τη διάρκεια της εκπαίδευσης, ώστε να αξιολογήσουν τη σημασία τους. Παράλληλα, διαθέτουν μια πύλη λήθης (forget gate) που τους επιτρέπει να αποθηκεύουν ή να ξεχνούν τα δεδομένα αναλόγως με το εάν αυτά θεωρούνται σημαντικά ή όχι. Η λειτουργία αυτή, τους επιτρέπει να αποδίδουν πολύ καλύτερα σε σχέση με τα απλά RNN σε μεγάλες ακολουθίες δεδομένων, έως όμως ένα όριο, καθώς με την αύξηση του μήκους των ακολουθιών το μοντέλο αδυνατεί να αποφασίσει ποια δεδομένα είναι σημαντικά και ποια όχι. Ταυτόχρονα, αυξάνονται αισθητά και οι υπολογιστικές απαιτήσεις για το μοντέλο, καθώς όπως και στα επαναλαμβανόμενα νευρωνικά δίκτυα δεν είναι δυνατή η παραλληλοποίηση τους αφού επεξεργάζονται τα δεδομένα μεμονωμένα. Στο σχήμα 3.13 φαίνεται ένα παράδειγμα ενός κελιού LSTM, το οποίο διαθέτει μια πύλη εισόδου, μια πύλη εξόδου καθώς και μια πύλη λήθης.



Σχήμα 3.13: Το κελί ενός LSTM

Source: `https://medium.com/@andre.holzner/lstm-cells-in-pytorch-fab924a78b1c`

3.4.3 Ο Μηχανισμός Προσοχής

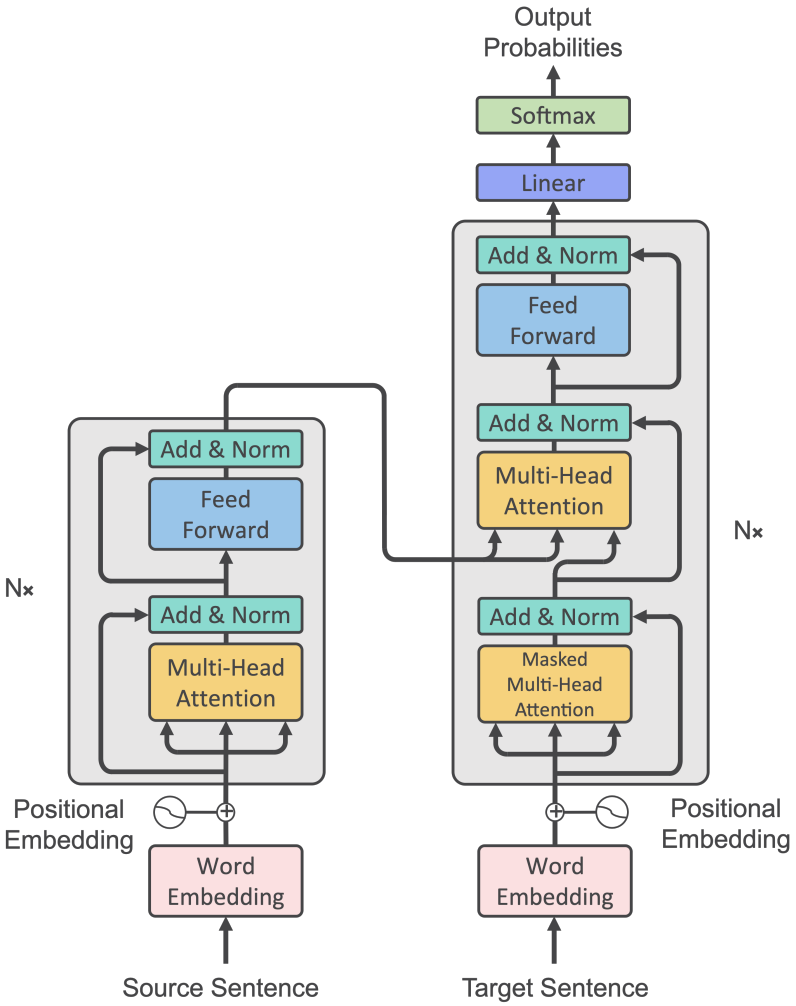
Στον τομέα των βαθιών νευρωνικών δικτύων, ο μηχανισμός προσοχής αποτελεί ένα από τα ισχυρότερα εργαλεία κατά την επεξεργασία των δεδομένων εισόδου. Ο μηχανισμός αυτός προσφέρει τη δυνατότητα αξιολόγησης της σημασίας κάθε μέρους των δεδομένων εισόδου. Μέσω της αξιολόγησης αυτής, η οποία σε αντίθεση με τα βάρη των νευρώνων μπορεί να μεταβληθεί κατά την εκτέλεση, το δίκτυο μπορεί να ανιχνεύσει τις λεπτομέρειες των δεδομένων που έχουν πραγματικά σημασία. Κατά την εκτέλεση, η έξοδος δεν εξαρτάται μόνο από την ακριβώς προηγούμενη κρυφή παράσταση, αλλά είτε από το σύνολο των κρυφών καταστάσεων του δικτύου (Global Attention) είτε από ένα συγκεκριμένο υποσύνολο καταστάσεων (Local Attention). Κατά την εκπαίδευση, ορίζεται ένα επιπλέον σύνολο κόμβων στο δίκτυο, τα βάρη προσοχής, τα οποία έχουν την ιδιότητα να μεταβάλλονται ακόμη και κατά τη διάρκεια της εκτέλεσης, σε αντίθεση με τα υπόλοιπα βάρη του δικτύου.

Η απλούστερη περίπτωση όπου ο μηχανισμός προσοχής είναι αναγκαίος, είναι η περίπτωση μετάφρασης προτάσεων από μία γλώσσα σε κάποια άλλη. Όπως είναι λογικό, η ακολουθία των λέξεων μιας πρότασης μπορεί να διαφέρει από γλώσσα σε γλώσσα. Παρόλα αυτά, τα περισσότερα νευρωνικά δίκτυα είναι ικανά να μεταφράσουν μονάχα κάθε λέξη ξεχωριστά, καταλήγοντας σε νοηματικά λανθασμένες προτάσεις. Αυτό ήρθε να αλλάξει ο μηχανισμός προσοχής, χάριν του οποίου το δίκτυο μπορεί να δίνει βάση σε προηγούμενες ή επόμενες λέξεις από αυτήν που μεταφράζει, και να βρίσκει τη σωστή μετάφραση ολόκληρης της πρότασης.

3.4.4 Το Μοντέλο Transformer

Όπως έχουμε ήδη αναφέρει, τα επαναλαμβανόμενα νευρωνικά δίκτυα δεν μπορούν να παραλληλοποιηθούν λόγω του τρόπου που επεξεργάζονται τα δεδομένα. Το γεγονός αυτό, σε συνδυασμό με τους βρόγχους που υπάρχουν εντός των συγκεκριμένων δικτύων, οδηγεί σε υψηλές υπολογιστικές και χρονικές απαιτήσεις κατά τη διάρκεια της εκπαίδευσης. Τη λύση στο πρόβλημα αυτό έφερε η δημοσίευση **Attention is all you need**[46] που εισήγαγε την αρχιτεκτονική των Transformers. Παρόλο που έχουν δημιουργηθεί για την επεξεργασία ακολουθιακών δεδομένων[47], τα Transformers δεν αποτελούν ένα είδος επαναλαμβανόμενων νευρωνικών δικτύων, αλλά βασίζονται πλήρως στον μηχανισμό προσοχής για τη λειτουργία τους. Το διάγραμμα της αρχιτεκτονικής ενός Transformer,

όπως αυτό προτάθηκε στη δημοσίευση που αναφέραμε παραπάνω, φαίνεται στο σχήμα 3.14. Σε αντίθεση με τα RNN, τα οποία επεξεργάζονται κάθε κομμάτι της ακολουθίας εισόδου μεμονωμένα, τα Transformers είναι σχεδιασμένα ώστε να επεξεργάζονται ένα υποσύνολο της ακολουθίας ταυτόχρονα, γεγονός που τα καθιστά παραλληλοποιήσιμα και μπορεί να μειώσει αισθητά τις χρονικές απαιτήσεις της εκπαίδευσης. Τα μοντέλα αυτά έχουν αποδειχθεί αρκετά αποδοτικά ώστε να αποδίδουν καλύτερα ακόμη και από συνελικτικά δίκτυα στην επεξεργασία εικόνων, εφόσον διατεθούν επαρκή δεδομένα εκπαίδευσης[48].



Σχήμα 3.14: Η αρχιτεκτονική ενός Transformer

Source: `https://arxiv.org/abs/1706.03762`

Όσον αφορά τη λειτουργία τους, η είσοδος μετατρέπεται αρχικά σε διανύσματα με κωδικοποίηση θέσης τα οποία έπειτα τροφοδοτούνται σε ένα κωδικοποιητή έξι στρωμάτων. Ο κωδικοποιητής αυτός αποτελείται από δύο στρώματα, το πρώτο εκ των οποίων υλοποιεί τον μηχανισμό προσοχής πολλών κεφαλών (Multi-Head Self-Attention), ενώ το δεύτερο είναι ένα πλήρως συνδεδεμένο εμπρόσθια τροφοδοτούμενο δίκτυο. Ο σκοπός του, είναι η αναγνώριση των συσχετίσεων ανάμεσα σε κάθε μέρος της εισόδου. Καθώς το κάθε μέρος της εισόδου μπορεί να έχει περισσότερες από μια λειτουργίες στο σύνολο της εισόδου, χρησιμοποιείται ο μηχανισμός προσοχής πολλών κεφαλών, όπως αναφέραμε και παραπάνω. Ο αποκωδικοποιητής αποτελείται από τα ίδια στρώματα, μεταξύ τους όμως υπάρχει και ένα επιπλέον στρώμα προσοχής. Ο σκοπός του είναι η χρήση της ακολουθίας εισόδου σε συνδυασμό με τη μερική έξοδο που έχει ήδη παραχθεί ώστε να προβλέψει το επόμενο κομμάτι της ακολουθίας εξόδου.

Κεφάλαιο 4

Μεθοδολογία & Ανάλυση

Στο παρόν κεφάλαιο θα παρουσιαστεί η μεθοδολογία που έχει ακολουθηθεί κατά τη συγκεκριμένη διπλωματική εργασία. Αρχικά, θα παρουσιαστεί η υποδομή που χρησιμοποιήθηκε για τη συλλογή των απαραίτητων πληροφοριών σχετικά με τα αιτήματα προσβάσεων μνήμης για διάφορα ίχνη (traces). Έπειτα θα παρουσιαστεί η ανάλυση που υπήρξε στην πληροφορία αυτή ώστε να αξιοποιηθεί με τον βέλτιστο δυνατό τρόπο. Τέλος, θα παρουσιαστεί το μοντέλο μηχανικής μάθησης TransFetch, το οποίο έπειτα από ορισμένες αλλαγές αποτελεί το μοντέλο που χρησιμοποιήθηκε για την αξιολόγηση της προανάκλησης στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης με τη χρήση Τεχνητών Νευρωνικών Δικτύων. Στην αξιολόγηση αυτή έχει ληφθεί υπόψιν η επίπτωση της παραμετροποίησης ορισμένων συνθηκών, όπως είναι ο τρόπος χρήσης και το πλήθος των δεδομένων εκπαίδευσης, η προσθήκη της έννοιας της επικαιρότητας των προανακλήσεων και το πλήθος των traces στα οποία εκπαιδεύεται το δίκτυο προτού πραγματοποιηθούν προανακλήσεις.

4.1 Ο προσομοιωτής ChampSim

Στην παρούσα ενότητα θα παρουσιαστεί το βασικότερο εργαλείο αυτής της εργασίας, η υποδομή που χρησιμοποιήθηκε για την ολοκλήρωση των απαραίτητων προσομοιώσεων (Simulations). Η υποδομή αυτή είναι ο προσομοιωτής ChampSim[49], ο οποίος έχει χρησιμοποιηθεί σε πολλαπλούς διαγωνισμούς και ερευνητικές εργασίες σχετικές με την προανάκληση δεδομένων. Ο ChampSim αποτελεί ένα πρόγραμμα γραμμένο σε γλώσσα C++ που χρησιμοποιείται για τη μελέτη της αρχιτεκτονικής συστημάτων. Για την εκτέλεση κάποιας προσομοίωσης, απαιτείται πρώτα η δημιουργία ενός εκτελέσιμου αρχείου στο οποίο ο χρήστης μπορεί να ορίσει τις ακριβείς παραμέτρους της αρχιτεκτονικής που θα προσομοιωθεί, καθώς και τους προανακλητές που θα χρησιμοποιηθούν για κάθε κρυφή μνήμη, τις πολιτικές αντικατάστασης, τον προβλεπτής διακλαδώσεων και το πλήθος των πυρήνων του συστήματος. Για την εκτέλεση προσομοιώσεων, απαιτούνται αρχεία που περιλαμβάνουν το σύνολο των εντολών που εκτελούνται όταν τρέχει κάποια εφαρμογή, ονομαζόμενα traces. Κάθε προσομοίωση δέχεται σαν είσοδο το εκτελέσιμο αρχείο που θα χρησιμοποιηθεί, ένα trace, καθώς και το πλήθος των εντολών για τις οποίες επιθυμεί ο χρήστης να προσομοιώσει την εκτέλεση του εκάστοτε προγράμματος. Στο τέλος της εκτέλεσης δημιουργείται ένα αρχείο κειμένου το οποίο περιλαμβάνει πληροφορίες σχετικές με τη λειτουργία και την απόδοση κάθε μέρους της αρχιτεκτονικής, όπως για παράδειγμα τις εντολές ανά κύκλο ή το πλήθος των αιτημάτων προανάκλησης για κάθε κρυφή μνήμη.

Η συγκεκριμένη έκδοση του προσομοιωτή ChampSim που χρησιμοποιήθηκε σε αυτή τη διπλωματική εργασία, με ορισμένες αλλαγές, προέρχεται από το *Morrigan: A Composite Instruction TLB Prefetcher*[50, 51]. Στην έκδοση αυτή, υπάρχει πλήρως λειτουργική Κρυφή Μνήμη Αναζήτησης Μετάφρασης, αλλά και η δυνατότητα προανάκλησης σε αυτή. Οι αλλαγές που έγιναν αφορούν το είδος των αιτημάτων μεταφράσεων που πυροδοτούν αιτήματα προανάκλησης, καθώς στην αρχική του έκδοση ο προσομοιωτής επικεντρώνεται στην προανάκληση μεταφράσεων αποκλειστικά για εντολές. Η αρχιτεκτονική που χρησιμοποιήθηκε για τη συγκεκριμένη μελέτη περιέχει ξεχωριστές TLBs για εντολές και δεδομένα, καθώς και ιεραρχική οργάνωση της TLB. Συγκεκριμένα, περιλαμβάνει μια L1 DTLB για τα δεδομένα, μια L1 ITLB για τις εντολές, αλλά και μια L2 STLB η οποία μοιράζεται μεταξύ των DTLB και ITLB. Στη μελέτη αυτή η προσοχή εστιάζεται στις αστοχίες δεδομένων που προκύπτουν στην STLB, καθώς αυτές είναι που ευθύνονται κατά κύριο λόγο για την επιβάρυνση ενός υπολογιστικού συστήματος. Το σύνολο των προσομοιώσεων αφορούν συστήματα ενός πυρήνα, με την STLB

να είναι η μόνη κρυφή μνήμη που χρησιμοποιεί προανακλητή. Οι βασικές παράμετροι της αρχιτεκτονικής του προσομοιωτή, όπως χρησιμοποιήθηκαν για κάθε προσομοίωση στην παρούσα εργασία παρουσιάζονται στους πίνακες 4.1 και 4.2. Σημειώνεται πως οι προανακλητές σε όλες τις κρυφές μνήμες του επεξεργαστή παρέμειναν απενεργοποιημένοι σε όλα τα πειράματα. Η τελική μορφή του προσομοιωτή, όπως χρησιμοποιήθηκε για την εκτέλεση πειραμάτων βρίσκεται στο αποθετήριο [52].

Παράμετροι Αρχιτεκτονικής TLB			
	L1 Instruction TLB	L1 Data TLB	L2 Shared TLB
SET	16	16	256
WAY	8	4	6
RQ SIZE	16	16	32
WQ SIZE	16	16	32
PQ SIZE	0	0	64
MSHR SIZE	4	4	4
LATENCY	1	1	8

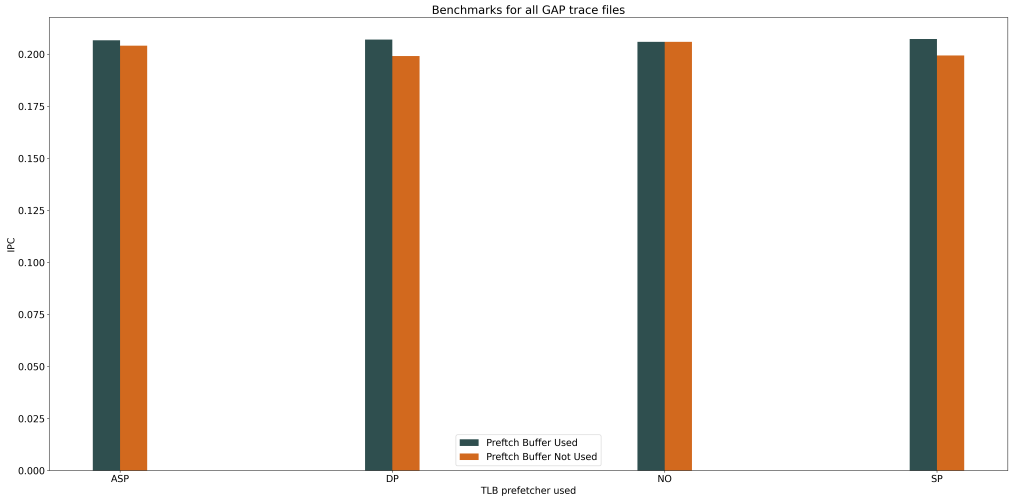
Πίνακας 4.1: Χαρακτηριστικά των TLB στον προσομοιωτή ChampSim

Παράμετροι Αρχιτεκτονικής Cache				
	L1 Instruction Cache	L1 Data Cache	L2 Cache	Last Level Cache
SET	64	64	1024	2048
WAY	8	8	8	16
RQ SIZE	64	64	32	48
WQ SIZE	64	64	32	48
PQ SIZE	32	8	32	64
MSHR SIZE	8	16	32	64
LATENCY	4	4	8	10

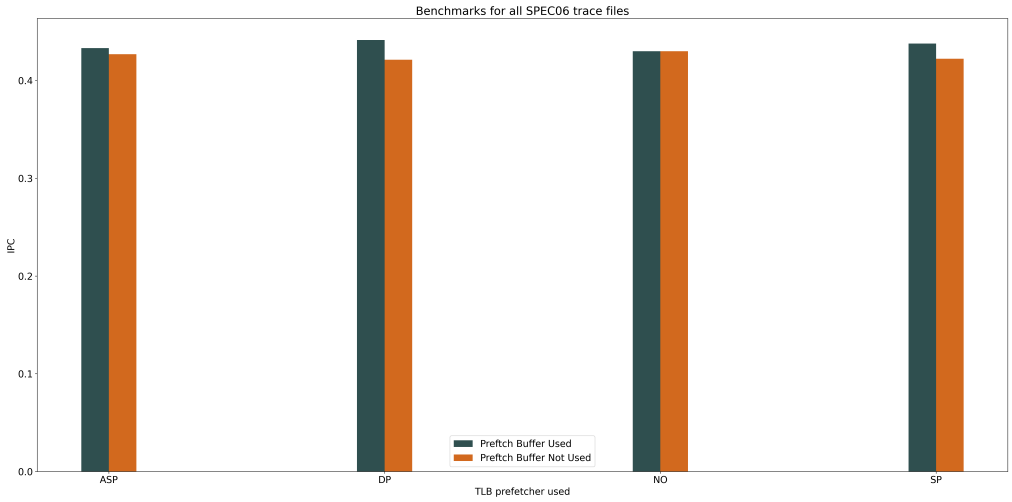
Πίνακας 4.2: Χαρακτηριστικά των Κρυφών Μνημών στον προσομοιωτή ChampSim

Όπως φαίνεται και στους πίνακες παραμέτρων, η TLB δεύτερου επιπέδου κάνει χρήση ουράς προανάκλησης, μεγέθους 64 καταχωρήσεων. Όπως αναφέρθηκε και παραπάνω, η χρήση ουράς προανάκλησης στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης είναι συνηθισμένη τακτική ώστε να αποφευχθεί η μόλυνση της Κρυφής Μνήμης Αναζήτησης Μετάφρασης με άχρηστες προανακλήσεις. Ο βαθμός στον οποίο μπορεί η χρήση της Ουράς Προανάκλησης να επηρεάσει την απόδοση φαίνεται στα σχήματα 4.1, 4.2, 4.3. Σε αυτά, χρησιμοποιείται ως μετρική το IPC, δηλαδή το μέσο πλήθος εντολών που ολοκληρώνεται ανά κύκλο εκτέλεσης, ώστε να συγκριθεί η απόδοση των προανακλητών στην STLB όταν υπάρχει Ουρά Προανακλήσεων και όταν όχι. Οι προανακλητές που χρησιμοποιήθηκαν είναι με τη σειρά ο Προανακλητής Αυθαίρετου Βήματος (ASP), ο Προανακλήτής Απόστασης (DP), η απόδοση χωρίς χρήση κάποιου προανακλητή (NO) και ο Ακολουθιακός Προανακλητής (SP). Κατά τη διαδικασία αυτή, καθώς και στη συνέχεια, χρησιμοποιήθηκε μια συλλογή από traces από τις σουίτες προγραμμάτων GAP[53], SPEC 2006[54] και SPEC 2017[55]. Από τα διαγράμματα αυτά, είναι εμφανές πως η χρήση Ουράς Προανάκλησης είναι ευεργετική για το σύστημα, αφού βελτιώνει αισθητά την απόδοση κάθε προανακλητή. Παράλληλα, εάν δεν χρησιμοποιηθεί ουρά προανάκλησης, η χρήση προανακλητή στην TLB φαίνεται να επιβαρύνει την απόδοση του συστήματος.

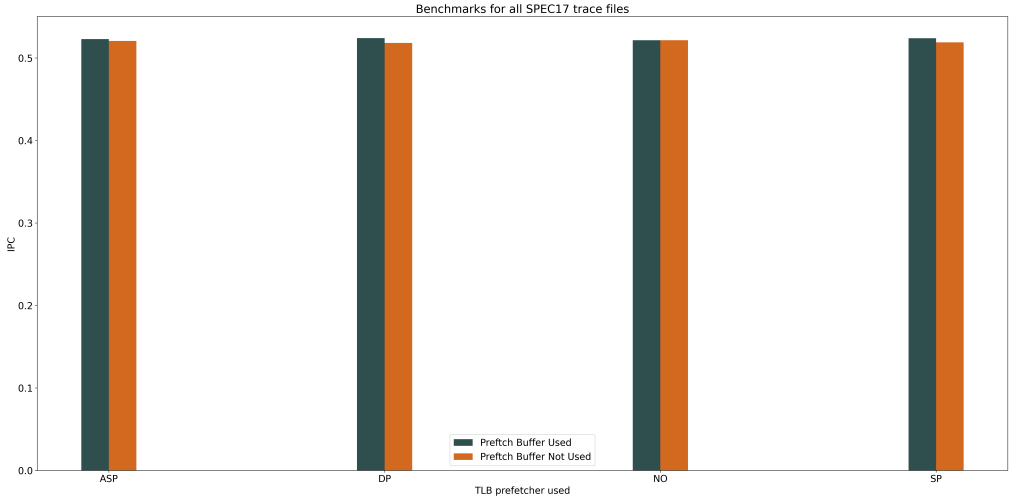
Ακόμη, αναφέρουμε πως ο προσομοιωτής στηρίζεται στην αρχιτεκτονική x86-64. Σε αυτή, ο Πίνακας Σελίδων που χρησιμοποιείται είναι υλοποιημένος σε Ιεραρχική Οργάνωση τεσσάρων επιπέδων, τα οποία ονομάζονται PML4, PDP, PD και PT αντίστοιχα, από το σημαντικότερο στο λιγότερο σημαντικό. Στη συγκεκριμένη αρχιτεκτονική, η διάσχιση σελίδων πραγματοποιείται με τη χρήση ενός επιπλέον καταχωρητή, που ονομάζεται CR3, και διατηρεί τον δείκτη στη ρίζα του πίνακα. Κατά την αναζήτηση μιας καταχώρησης, ο CR3 χρησιμοποιείται για την εύρεση του πρώτου επιπέδου του Πίνακα Σελίδων, δηλαδή το PML4. Στη συνέχεια, η καταχώρηση του PML4, που ουσιαστικά αποτελεί δείκτη προς τη ρίζα του επόμενου επιπέδου, χρησιμοποιείται για την εύρεση της αντίστοιχης καταχώρησης PDP. Η διαδικασία επαναλαμβάνεται έως ότου βρεθεί η καταχώρηση PT, η οποία



Σχήμα 4.1: Απόδοση για τα προγράμματα της σουίτας GAP



Σχήμα 4.2: Απόδοση για τα προγράμματα της σουίτας SPEC2006



Σχήμα 4.3: Απόδοση για τα προγράμματα της σουίτας SPEC2017

οδηγεί στην αναζητούμενη απεικόνιση στον χώρο φυσικών διευθύνσεων.

4.2 Συλλογή Δεδομένων Εκπαίδευσης

Κατά την εκπαίδευση Τεχνητών Νευρωνικών Δικτύων, ένας από τους σημαντικότερους παράγοντες που πρέπει να ληφθεί υπόψιν είναι το σύνολο των δεδομένων εκπαίδευσης που θα χρησιμοποιηθεί. Στην παρούσα εργασία, έχει χρησιμοποιηθεί η μεθοδολογία που προτάθηκε στον διαγωνισμό προανάκλησης που διοργανώθηκε το 2021 από την ISCA (International Symposium on Computer Architecture)[56]. Συγκεκριμένα, για την εκπαίδευση του νευρωνικού δικτύου χρησιμοποιούνται αρχεία κειμένου όπου έχουν καταγραφεί οι αλληλουχίες των αιτημάτων πρόσβασης στην STLB που εκτελεί κάποια εφαρμογή, ονομαζόμενα Load Traces. Καθώς στον διαγωνισμό παρέχονται Load Traces με καταγεγραμμένα τα αιτήματα πρόσβασης στο τελευταίο επίπεδο κρυφής μνήμης του επεξεργαστή, τα απαραίτητα αρχεία για τη συγκεκριμένη εργασία χρειάστηκε να δημιουργηθούν εκ νέου, χρησιμοποιόντας τον προσομοιωτή ChampSim καθώς και τα ίχνη που παρέχονταν από τον διαγωνισμό[57], με τρόπο που θα περιγραφεί παρακάτω.

Για τη συλλογή της απαραίτητης πληροφορίας, χρησιμοποιήθηκε μια έκδοση του προσομοιωτή ChampSim που βρίσκεται στο αποθετήριο [58]. Στη συγκεκριμένη μορφή, ο προσομοιωτής εκτελεί αίτημα «προανάκλησης» σε κάθε αίτημα πρόσβασης στο δεύτερο επίπεδο της Κρυφής Μνήμης Αναζήτησης Μετάφρασης, ανεξαρτήτως του εάν ήταν εύστοχο ή όχι, εφόσον η πρόσβαση αυτή αφορά δεδομένα. Ο προανακλητής που χρησιμοποιήθηκε συνδυαστικά με τη μορφή αυτή, εκτυπώνει στην έξοδο τις πληροφορίες που περιέχει το αίτημα προανάκλησης που δέχτηκε, με την έξοδο να αποθηκεύεται σε κατάλληλο αρχείο. Με τον τρόπο αυτό καταγράφηκαν πληροφορίες αναφορικά με όλα τα αιτήματα δεδομένων στην Κρυφή Μνήμη Αναζήτησης δεύτερου επιπέδου που προέκυψαν κατά την εκτέλεση των πρώτων 100 εκατομμυρίων εντολών. Συγκεκριμένα η πληροφορία που αποθηκεύεται είναι:

- *Αριθμός Εντολής (Instruction ID):* Ο αριθμός εντολής, δηλαδή ένα γνησίως αύξων και μοναδικό αναγνωριστικό για κάθε εντολή που εκτελείται. Η αρίθμηση του αναγνωριστικού αυτού παραμένει ίδια σε πολλαπλές προσομοιώσεις της ίδιας εφαρμογής, ακόμη και εάν ο κύκλος ρολογιού κατά τον οποίο εκτελείται μια εντολή αλλάξει. Η χρησιμότητα του αριθμού αυτού έγκειται στην αναγνώριση της αλληλουχίας με την οποία εκτελούνται τα αιτήματα πρόσβασης στη μνήμη καθώς και, μετέπειτα, στην εκτέλεση αιτημάτων προανάκλησης έπειτα από συγκεκριμένες εντολές.

- *Τρέχων Κύκλος(Current Cycle):* Ο τρέχων κύκλος, δηλαδή ο κύκλος ρολογιού κατά τον οποίο εκτελέστηκε το συγκεκριμένο αίτημα πρόσβασης στη μνήμης

- *Δείκτης Εντολής (Instruction Pointer - IP):* Ονομάζεται και μετρητής προγράμματος (Program Counter – PC) και μας επιτρέπει να αναγνωρίσουμε την εντολή του προγράμματος που προκάλεσε ένα αίτημα πρόσβασης στη μνήμη. Η διαφορά του σε σχέση με τον αριθμό εντολής είναι πως ο μετρητής προγράμματος παραμένει σταθερός για συγκεκριμένες εντολές, όσες φορές και εάν αυτές εκτελεστούν κατά μια προσομοίωση, σε αντίθεση με τον αριθμό εντολής, που όπως αναφέρθηκε και παραπάνω λαμβάνει γνησίως αύξουσες τιμές.

- *Εικονική Διεύθυνση (Virtual Address):* Η εικονική διεύθυνση, δηλαδή ο αριθμός εικονικής σελίδας για τον οποίο έγινε το αίτημα πρόσβασης. Αποτελεί τη σημαντικότερη παράμετρο προς καταγραφή, εφόσον στα αιτήματα προανάκλησης που εκτελούνται θα πρέπει να γνωρίζουμε ποια καταχώρηση του πίνακα σελίδων θέλουμε να έρθει στην STLB.

- *Ευστοχία/Αστοχία αιτήματος(Hit/Miss):* Τέλος, καταγράφεται και εάν το αίτημα πρόσβασης ήταν εύστοχο ή όχι. Βεβαίως εάν εκτελείται προανάκληση σε επόμενες προσομοιώσεις, η τιμή της συγκεκριμένης παραμέτρου σε μελλοντικά πειράματα είναι πιθανόν να αλλάξει, είναι όμως ικανή να προσφέρει χρήσιμη πληροφορία σε συνδυασμό με τον μετρητή προγράμματος σε περιπτώσεις όπου έχουμε αλλεπάλληλες αστοχίες για συγκεκριμένο μετρητή προγράμματος.

4.3 Ανάλυση Δεδομένων Εκπαίδευσης

Όπως αναφέρθηκε και παραπάνω, ένας από τους σημαντικότερους παράγοντες για τη δημιουργία ενός αποδοτικού μοντέλου μηχανικής μάθησης είναι η επιλογή των κατάλληλων δεδομένων εκπαίδευσης. Εξίσου σημαντικός όμως είναι και ο τρόπος με τον οποίο το μοντέλο θα εκμεταλλευτεί και θα χρησιμοποιήσει την πληροφορία που του παρέχεται[59]. Για τον λόγο αυτό, ακολουθεί μια σύντομη ανάλυση των καταγραφών προσβάσεων μνήμης, από την οποία θα συλλεχθεί χρήσιμη πληροφορία για την παραμετροποίηση του μοντέλου μηχανικής μάθησης που θα χρησιμοποιηθεί μετέπειτα.

4.3.1 Η Απαραίτητη Επικαιρότητα Προανάκλησης

Όπως αναφέρθηκε και στο Κεφάλαιο 2, πέραν της εύρεσης των κατάλληλων δεδομένων προανάκλησης, είναι πολύ σημαντική η διασφάλιση πως τα δεδομένα προανάκλησης θα είναι διαθέσιμα όταν γίνει τελικά το αίτημα πρόσβασης σε αυτά. Σε μια τυπική TLB, το κόστος αστοχίας σε κύκλους, εφόσον υπάρξει ευστοχία στον πίνακα σελίδων, μπορεί να κυμαίνεται από 10 έως 100 κύκλους. Συγκεκριμένα στην έκδοση του προσομοιωτή ChampSim που χρησιμοποιήθηκε, η καθυστέρηση πρόσβασης στις TLB πρώτου επιπέδου (ITLB & DTLB) ισούται με 1 κύκλο, στην TLB δεύτερου επιπέδου (STLB) 8 κύκλους και στον πίνακα σελίδων περίπου 100 έως 300 κύκλους. Συγκεκριμένα, η μέση καθυστέρηση λόγω Διάσχισης Σελίδων που προέκυψε από αστοχία δεδομένων στην STLB υπολογίστηκε ίση με 124 κύκλους. Στο πλαίσιο αυτό, υπολογίστηκε το ποσοστό των διαδοχικών προσβάσεων και διαδοχικών αστοχιών δεδομένων στην STLB που προκύπτουν εντός 124 κύκλων και εντός 300 κύκλων μεταξύ τους αντίστοιχα. Τα αποτελέσματα φαίνονται στον πίνακα 4.3.

Ποσοστό Διαδοχικών	Εφαρμογές Σουίτας GAP	Εφαρμογές Σουίτας SPEC2006	Εφαρμογές Σουίτας SPEC2017	Σύνολο των εφαρμογών
Αστοχιών εντός 124 κύκλων	5.468%	6.711%	5.828%	6.041%
Αστοχιών εντός 300 κύκλων	11.792%	31.103%	22.917%	23.316%
Προσβάσεων εντός 124 κύκλων	50.964%	30.571%	35.221%	36.898%
Προσβάσεων εντός 300 κύκλων	74.426%	52.497%	49.383%	55.449%

Πίνακας 4.3: Ποσοστά διαδοχικών προσβάσεων και αστοχιών εντός αριθμού κύκλων

Στον πίνακα αυτόν, βλέπουμε πως σε όλες τις σουίτες σημαντικό ποσοστό διαδοχικών αστοχιών και προσβάσεων λαμβάνουν χώρα σε σχετικά μικρή απόσταση κύκλων. Στο ίδιο πλαίσιο λοιπόν, υλοποιήθηκε και χρησιμοποιήθηκε ένας Oracle Prefetcher. Εφόσον έχουμε δημιουργήσει ήδη το πλήρες ιστορικό προσβάσεων μνήμης για κάθε ίχνος που μας αφορά, μπορούμε να δημιουργήσουμε έναν Oracle Prefetcher, ο οποίος σε κάθε εντολή εκτελεί αίτημα προανάκλησης για την αστοχία που προκύπτει έπειτα από N σε αριθμό αιτήματα πρόσβασης από την τρέχουσα εντολή. Ο αριθμός αυτός, N, αποτέλεσε παράμετρο κατά την εκτέλεση πειραμάτων, με τα αποτελέσματα για τα ίχνη ομαδοποιημένα ανά σουίτας να παρουσιάζονται στο πίνακα 4.4. Συγκεκριμένα, φαίνεται το μέσο πλήθος εντολών ανά κύκλο(Avg. IPC), καθώς και το μέσο κέρδος σε σχέση με την περίπτωση όπου δεν χρησιμοποιείται προανακλητής(Geo Gains) για τις εφαρμογές κάθε σουίτας ξεχωριστά και για το σύνολο των εφαρμογών, ενώ λαμβάνονται υπόψιν προανακλητές με τιμή "Επικαιρότητας" από 1 έως και 18. Από τον πίνακα αυτόν, επιχειρούμε να βρούμε την τιμή του N για την οποία τα αιτήματα που εκτελούνται έχουν την απαραίτητη "Επικαιρότητα", δηλαδή εκτελούνται σε κατάλληλο χρονικό πλαίσιο ώστε να οδηγήσουν στη βέλτιστη απόδοση του συστήματος. Η τιμή αυτή φαίνεται να είναι μεταξύ 9 και 12, καθώς εάν είναι μικρότερη ορισμένα από τα αιτήματα εκτελούνται πολύ αργά, ενώ αν είναι μεγαλύτερη τα αιτήματα φαίνεται να διαγράφονται από την Ουρά Προανάκλησης προτού χρησιμοποιηθούν.

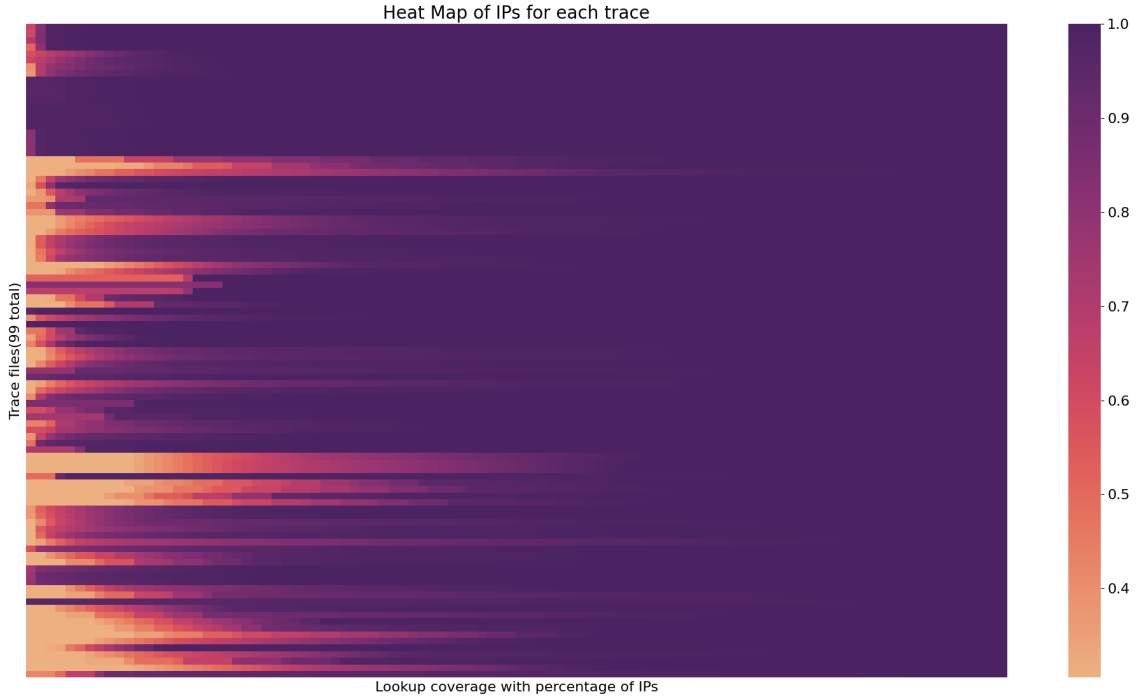
Προανακλητής	Εφαρμογές Σουίτας GAP		Εφαρμογές Σουίτας SPEC2006		Εφαρμογές Σουίτας SPEC2017		Σύνολο των εφαρμογών	
Μετρική	Avg. IPC	Geo Gains	Avg. IPC	Geo Gains	Avg. IPC	Geo Gains	Avg. IPC	Geo Gains
No	0.218	0.00%	0.424	0.00%	0.529	0.00%	0.433	0.00%
Oracle1	0.220	0.58%	0.438	4.06%	0.537	2.96%	0.441	2.79%
Oracle2	0.220	0.85%	0.442	5.22%	0.538	3.32%	0.443	3.37%
Oracle3	0.221	0.99%	0.445	6.01%	0.538	3.52%	0.444	3.73%
Oracle4	0.221	1.03%	0.448	6.51%	0.538	3.52%	0.445	3.89%
Oracle5	0.221	1.08%	0.450	6.84%	0.539	3.54%	0.446	4.00%
Oracle6	0.221	1.08%	0.451	7.07%	0.539	**3.57%**	0.446	4.09%
Oracle7	0.221	1.09%	0.452	7.26%	**0.539**	3.53%	0.446	4.12%
Oracle8	0.221	**1.09%**	0.453	7.47%	0.539	3.52%	0.447	4.18%
Oracle9	**0.221**	1.09%	0.454	7.55%	0.539	3.52%	**0.447**	4.20%
Oracle10	0.221	1.08%	0.454	7.63%	0.538	3.47%	0.447	4.20%
Oracle11	0.221	1.08%	0.455	7.67%	0.538	3.45%	0.447	4.20%
Oracle12	0.221	1.07%	**0.455**	7.72%	0.538	3.46%	0.447	**4.22%**
Oracle13	0.221	1.07%	0.455	7.72%	0.538	3.43%	0.447	4.21%
Oracle14	0.221	1.06%	0.455	7.74%	0.538	3.41%	0.447	4.20%
Oracle15	0.221	1.07%	0.455	7.74%	0.538	3.38%	0.447	4.19%
Oracle16	0.221	1.06%	0.455	**7.75%**	0.538	3.37%	0.447	4.18%
Oracle17	0.221	1.06%	0.455	7.73%	0.538	3.35%	0.447	4.17%
Oracle18	0.221	1.06%	0.455	7.72%	0.538	3.33%	0.447	4.16%

Πίνακας 4.4: Πίνακας Αποτελεσμάτων Oracle Prefetcher

4.3.2 Ανάλυση με Βάση τον Μετρητή Προγράμματος

Όπως αναφέρθηκε και παραπάνω, ο Μετρητής Προγράμματος μας επιτρέπει να αναγνωρίσουμε την εντολή από την οποία προήλθε ένα αίτημα πρόσβασης. Η πληροφορία αυτή μπορεί να φανεί ιδιαίτερα χρήσιμη σε περιπτώσεις όπου μια εντολή εκτελείται πολλαπλές φορές, εφόσον είναι πιθανό η αλληλουχία των επόμενων εντολών να παραμένει επίσης σταθερή, και κατ' επέκταση εύκολα προβλέψιμη από τον προανακλητή που χρησιμοποιείται. Στο σχήμα 4.4 φαίνεται για κάθε εφαρμογή το ποσοστό των IPs από τις οποίες προέρχεται η πλειοψηφία των αιτημάτων πρόσβασης στην STLB. Στον κατακόρυφο άξονα φαίνονται τα αποτελέσματα για κάθε ξεχωριστή εφαρμογή, στον οριζόντιο το ποσοστό των IPs και η χρωματική διαβάθμιση δείχνει το ποσοστό των αιτημάτων πρόσβασης που προέρχονται από το συγκεκριμένο ποσοστό IPs.

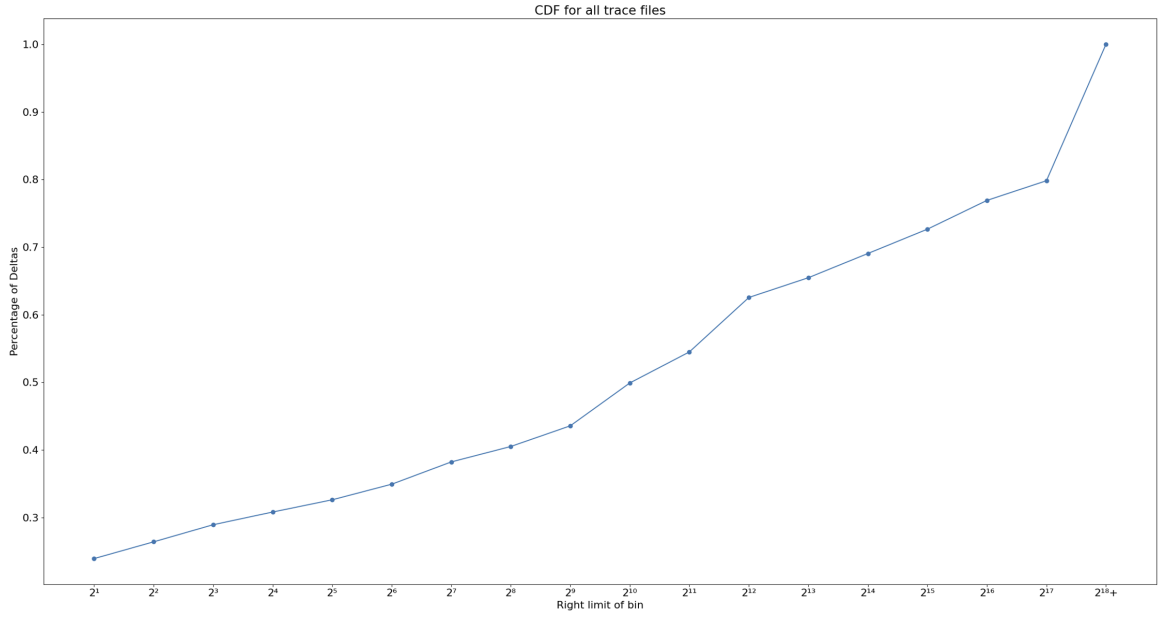
Από τη συγκεκριμένη ανάλυση βλέπουμε πως στην πλειοψηφία των περιπτώσεων ένα πολύ μικρό ποσοστό των IPs ευθύνονται για την πλειοψηφία των αιτημάτων πρόσβασης στη μνήμη. Επίσης, υπολογίζοντας τον ρυθμό αστοχίας για κάθε Μετρητή Προγράμματος ξεχωριστά, βλέπουμε πως παρόλο που η πλειοψηφία των αστοχιών προέρχεται από λίγες εντολές, οι εντολές αυτές έχουν μειωμένο ρυθμό αστοχίας σε σχέση με τη μέση εντολή. Με βάση τα παραπάνω, ο μετρητής προγράμματος φαίνεται να αποτελεί μια σημαντική παράμετρο κατά την εκπαίδευση του νευρωνικού δικτύου στην προανάκληση, καθώς μπορεί να του επιτρέψει να εντοπίσει πιθανώς «προβληματικές» αλληλουχίες προσβάσεων στη μνήμη που οδηγούν σε αλλεπάλληλες αστοχίες, καθώς και να δώσει βάση στην κάλυψη των αναγκών μνήμης των συχνότερα χρησιμοποιούμενων εντολών.



Σχήμα 4.4: Heat Map που δείχνει το ποσοστό των IP από όπου προέρχεται η πλειοψηφία των αιτημάτων

4.3.3 Ανάλυση με Βάση το Δέλτα

Κατά την προανάκληση σε κρυφή μνήμη, ένας από τους συχνότερους τρόπους με τον οποίο αξιοποιούνται οι διευθύνσεις στις οποίες προκύπτουν αστοχίες, είναι μέσω του υπολογισμού της διαφοράς μεταξύ δύο διαδοχικών διευθύνσεων στις οποίες υπήρξε αίτημα πρόσβασης που προκάλεσε αστοχία. Η διαφορά αυτή ονομάζεται και απόσταση (Distance) ή αλλιώς Δέλτα (Delta). Η χρήση των διαφορών αυτών περιορίζει αισθητά το πλήθος των πιθανών τιμών προανάκλησης, επιτρέποντας με τον τρόπο αυτό την αντιμετώπιση του προβλήματος της προανάκλησης ως ένα πρόβλημα ταξινόμησης. Η αθροιστική συνάρτηση κατανομής (Cumulative Distribution Function) της απόλυτης τιμής των Deltas των διευθύνσεων στις οποίες υπήρξαν αιτήματα πρόσβασης μνήμης για όλα τα ίχνη, φαίνεται στο σχήμα 4.5.



Σχήμα 4.5: C.D.F. που δείχνει την κατανομή μεγέθους των Δέλτα

Στον οριζόντιο άξονα βρίσκονται τα δεξιά όρια των «κάδων», που είναι δυνάμεις του 2, με το αριστερό όριο του κάθε κάδου να ισούται με το δεξί όριο του αμέσως προηγούμενου, όπου ταξινομήθηκαν οι αποστάσεις και στον κατακόρυφο άξονα το αθροιστικό ποσοστό των Deltas που ανήκουν σε κάθε κάδο. Βλέπουμε πως η συνάρτηση αυτή έχει σχεδόν γραμμική κλίση, γεγονός που σημαίνει πως οι αποστάσεις μεταξύ διαδοχικών διευθύνσεων είναι σχετικά ίσα κατανεμημένες στους «κάδους». Παράλληλα, εξετάζοντας τις αποστάσεις αυτές βρίσκουμε πως ορισμένες τιμές είναι συγκριτικά πολύ μικρές σε σχέση με τις μέγιστες αποστάσεις, δίχως όμως να υπάρχουν ακραίες τιμές τις οποίες θέλουμε να αφαιρέσουμε. Για τους λόγους αυτούς, δεν πραγματοποιείται κλιμάκωση ή κανονικοποίηση των δεδομένων εισόδου, εφόσον δεν θα προσέφερε κάποια επωφελή μετατροπή των δεδομένων.

4.4 Προανάκληση Δεδομένων στην TLB με χρήση Τ.Ν.Δ.

Το πρόβλημα της προανάκλησης σε κρυφή μνήμη με τη χρήση μηχανικής μάθησης είναι ένα ζήτημα το οποίο μπορεί να αντιμετωπιστεί με πληθώρα τρόπων και κατ' επέκταση με πληθώρα διαφορετικών κλάσεων νευρωνικών δικτύων. Το πρώτο βήμα για την αντιμετώπιση οποιουδήποτε ζητήματος μηχανικής μάθησης, είναι ο τύπος αλγορίθμου που θα επιλέξουμε να χρησιμοποιήσουμε. Κατά την προανάκληση επιθυμούμε να βρούμε τις διευθύνσεις εκείνες των οποίων η ύπαρξη στη κρυφή μνήμη θα επιφέρει βελτίωση της απόδοσης του υπολογιστικού συστήματος. Εξ ορισμού, είναι εμφανές πως το συγκεκριμένο πρόβλημα δεν μπορεί να αντιμετωπιστεί μέσω αλγορίθμων μη επιβλεπόμενης μάθησης. Παράλληλα, η αντιμετώπισή του μέσω ενισχυτικής μάθησης παρουσιάζει πολλές δυσκολίες, καθώς απαιτεί γνώση για την επιρροή κάθε απόφασης του δικτύου, δηλαδή στη συγκεκριμένη περίπτωση προανάκλησης, στη συνολική επίδοση του συστήματος. Με βάση τα παραπάνω, το πρόβλημα της προανάκλησης αντιμετωπίζεται συνήθως μέσω κάποιου αλγορίθμου επιβλεπόμενης μάθησης, είτε ως πρόβλημα ταξινόμησης είτε ως πρόβλημα παλινδρόμησης. Παρόλο που προβλήματα πρόβλεψης αντιμετωπίζονται συχνά μέσω παλινδρόμησης, η προανάκληση αντιμετωπίζεται ως πρόβλημα ταξινόμησης, καθώς το γεγονός πως το πλήθος των πιθανών εξόδων ισούται με το μέγεθος του σποραδικού χώρου εικονικών διευθύνσεων, σε συνδυασμό με τον πλεονασμό που συχνά υπάρχει στα δεδομένα εισόδου που χρησιμοποιούνται, οδηγεί στην κακή απόδοση μοντέλων παλινδρόμησης.

Εάν αντιμετωπιστεί ως πρόβλημα ταξινόμησης, η προανάκληση παρουσιάζει αρκετές ομοιότητες με προβλήματα φυσικής γλώσσας (Natural Language Problems). Συγκεκριμένα, και στις δύο περιπτώσεις μπορούμε να θεωρήσουμε πως επιχειρούμε να προβλέψουμε την επόμενη λέξη, ή αντίστοιχα διεύθυνση, με βάση μια γνωστή ακολουθία λέξεων, ή αντίστοιχα διευθύνσεων. Παρόλα αυτά, είναι εμφανές πως το πλήθος των μοναδικών διευθύνσεων, ιδίως σε συστήματα εικονικής διεύθυνσης, είναι αρκετές τάξεις μεγέθους μεγαλύτερο από το πλήθος των λέξεων που χρησιμοποιούνται σε προβλήματα NLP. Όπως αναφέρθηκε και παραπάνω, η χρήση των Deltas στη θέση των διευθύνσεων μπορεί να μετριάσει το πρόβλημα αυτό, το οποίο ονομάζεται και Έκρηξη Κλάσεων (Class Explosion). Ακόμη, πρέπει να ληφθεί υπόψιν το γεγονός πως στην πραγματικότητα δεν είναι γνωστή η επιρροή που θα έχει μια προανάκληση στην επίδοση του συστήματος, καθώς και σε περιπτώσεις όπου αυτή είναι επιτυχής, είναι πιθανό το αίτημα να εκτελεστεί πολύ αργά, και έτσι να μην είναι χρήσιμο, ή να αντικαταστήσει εξίσου χρήσιμα δεδομένα στη μνήμη.

Η επιλογή του σωστού είδους νευρωνικού δικτύου αποτελεί πολύ σημαντικό παράγοντα για τη δημιουργία ενός αποδοτικού μοντέλου. Γενικά, υπάρχουν ποικίλα νευρωνικά δίκτυα, το κάθε ένα εκ των οποίων είναι κατάλληλο και για την αντιμετώπιση διαφορετικών προβλημάτων. Συνήθως η καταλληλότερη προσέγγιση είναι η επιλογή ενός μοντέλου που έχει αποδειχθεί πως είναι ικανό να αποδώσει σε παρόμοια προβλήματα. Παραδείγματος χάριν, προβλήματα όπου τα δεδομένα παρουσιάζουν χωρικές εξαρτήσεις, όπως οι εικόνες, αντιμετωπίζονται συνήθως με τη χρήση Συνελικτικών Νευρωνικών Δικτύων. Αντιθέτως, προβλήματα όπου τα δεδομένα παρουσιάζουν ακολουθιακές, ή αλλιώς χρονικές εξαρτήσεις, αντιμετωπίζονται καλύτερα μέσω αναδρομικών νευρωνικών δικτύων, όπως είναι τα δίκτυα Μακράς Βραχυπρόθεσμης Μνήμης (LSTM). Με μια πρώτη εκτίμηση, η προανάκληση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης φαίνεται να ανήκει στη δεύτερη κατηγορία, καθώς η πρόβλεψη της επόμενης διεύθυνσης εξαρτάται από την ακολουθία των προηγούμενων διευθύνσεων. Ωστόσο, μέσω της κατάλληλης κωδικοποίησης των δεδομένων εισόδου, συγκεκριμένα τη μετατροπή τους σε πίνακα, το ιστορικό προσβάσεων μπορεί να τροφοδοτηθεί σαν είσοδο και σε ένα συνελικτικό δίκτυο, το οποίο μπορεί να παρατηρήσει έπειτα συσχετίσεις μεταξύ των δεδομένων. Ακόμη, αν και δεν έχει αποτελέσει αντικείμενο μελέτης στη συγκεκριμένη διπλωματι-

κή εργασία, σημειώνεται πως απλά συνελικτικά δίκτυα μπορούν να αναπαρασταθούν στο λογισμικό μέσω της χρήσης λογικών πυλών, γεγονός που σημαίνει πως είναι πιθανή η ενσωμάτωση ενός τέτοιου είδους δικτύου στην επεξεργαστική μονάδα ενός υπολογιστικού συστήματος.

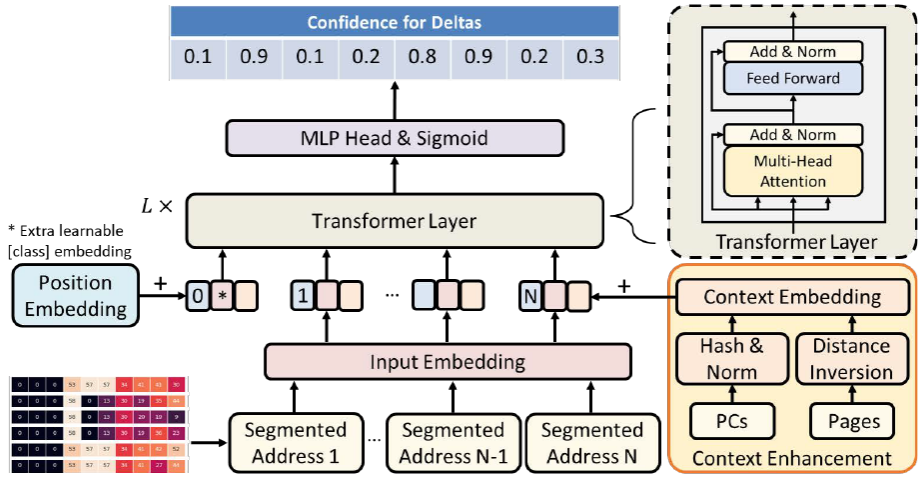
4.4.1 Το μοντέλο TransFetch

Στην ενότητα αυτή θα αναφερθούμε στο κύριο αντικείμενο μελέτης αυτής της διπλωματικής εργασίας, το TransFetch. Αρχικά, θα παρουσιαστεί αναλυτικά ο τρόπος με τον οποίο λειτουργεί το TransFetch, παράλληλα με τις απαραίτητες αλλαγές που έχουμε πραγματοποιήσει ώστε να εκτελεί προανάκληση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Στη συνέχεια, θα αναφερθούμε στα κομμάτια του δικτύου που έχουμε παραμετροποιήσει, και η απόδοση των οποίων θα αξιολογηθεί στο Κεφάλαιο 5. Στο τέλος του παρόντος κεφαλαίου θα αναφερθούμε και στις υπερπαραμέτρους που οδήγησαν σε μοντέλα βέλτιστης απόδοσης και που χρησιμοποιήθηκαν παρακάτω.

Το TransFetch αποτελεί ένα νευρωνικό δίκτυο κατασκευασμένο για την προανάκληση στην κρυφή μνήμη τελευταίου επιπέδου (Last Level Cache). Το μοντέλο που προτείνεται βασίζεται στην προσοχή πολλών κεφαλών και αντιμετωπίζει την προανάκληση ως ένα πρόβλημα ταξινόμησης πολλαπλών ετικετών (Labels). Για τον σκοπό αυτό χρησιμοποιείται ένας πίνακας δυαδικών τιμών (Bitmap) που αποθηκεύει τα μελλοντικά Δέλτα και τα χρησιμοποιεί ως ετικέτες κατά την εκπαίδευση. Το αρχικό μοντέλο εκπαιδεύεται ώστε να υπολογίζει την πιθανότητα να λάβει κάθε δυαδικό ψηφίο εντός του πίνακα την τιμή 1. Η πιθανότητα που προσδίδει το μοντέλο σε κάθε δυαδικό ψηφίο αναφέρεται ως ή «αυτοπεποίθηση» (Confidence) του μοντέλου στην τιμή αυτή, και χρησιμοποιείται για την εκτέλεση μεγαλύτερου βαθμού αιτημάτων προανάκλησης. Παράλληλα, για την επίλυση του προβλήματος έκρηξης κλάσεων, εκτελείται κατάτμηση των διευθύνσεων εισόδου (Address Segmentation). Συγκεκριμένα, κάθε διεύθυνση χωρίζεται σε S στο πλήθος κομμάτια ίσου μήκους, κάθε ένα εκ των οποίων αναπαρίσταται με έναν ακέραιο με μέγιστο μέγεθος 2^s. Με τον τρόπο αυτό το λεξικό που χρησιμοποιεί το μοντέλο περιορίζεται αισθητά, καθώς δεν χρειάζεται να περιλάβει το συνολικό μέγεθος των διευθύνσεων που χρησιμοποιούνται. Για τη συγκεκριμένη εργασία, έχουν υπάρξει απαραίτητες αλλαγές στον τρόπο με τον οποίο γίνεται η επεξεργασία των δεδομένων εισόδου, έχουν προστεθεί ορισμένες λειτουργικότητες εντός του μοντέλου και έχει υπάρξει ανάλυση και ολοκλήρωση πειραμάτων για την εύρεση των καταλληλότερων υπερπαραμέτρων για την επίτευξη της μέγιστης απόδοσης του μοντέλου στην προανάκληση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης[60].

Παρόμοιες ερευνητικές εργασίες στο παρελθόν έχουν χρησιμοποιήσει αναδρομικά δίκτυα, και κατά κύριο λόγο LSTM, σε συνδυασμό με στρώματα Προσοχής για την επίτευξη υψηλής απόδοσης και ακρίβειας στην προανάκληση δεδομένων στην κρυφή μνήμη. Η επιλογή αυτή όμως καθιστά πολύ δύσκολη την παραλληλοποίηση των μοντέλων λόγω της αναδρομικής τους λειτουργίας, ενώ παράλληλα προσάπτονται υψηλές καθυστερήσεις κατά τη διάρκεια παραγωγής προβλέψεων σε αληθινό χρόνο. Για τους λόγους αυτούς, η κεντρική δομή του TransFetch βασίζεται αποκλειστικά στην Προσοχή πολλών κεφαλών (Multi-Head Attention), η χρήση της οποίας έχει αποδειχθεί επίσης αποτελεσματική για την εκπαίδευση δικτύων στην προανάκληση δεδομένων στο παρελθόν. Η βάση του δικτύου υλοποιείται μέσω ενός Συνελικτικού Νευρωνικού Δικτύου, και συγκεκριμένα ενός Χωρικού Συνελικτικού Δικτύου (Temporal Convolutional Network). Τα δίκτυα αυτά προτάθηκαν για πρώτη φορά το 2016[61] για την τμηματοποίηση ενεργειών στον χρόνο σε βίντεο (Video-Based Action Segmentation). Στην ουσία τα Χωρικά Συνελικτικά Δίκτυά καταφέρνουν να συνδυάσουν τις λειτουργίες ενός CNN και ενός RNN, ειδικεύονται στην επεξεργασία ακολουθιακών δεδομένων με χωρικό τρόπο, ενώ παράλληλα μπορούν να εκπαιδευθούν πολύ γρήγορα συγκριτικά με τα αναδρομικά δίκτυα. Τα TCN χάρη στο μεταβλητό μέγεθος εισόδου, τις σταθερές κλίσεις, τη δυνατότητα παραλληλοποίησής τους αλλά και της μεγαλύτερης μνήμης που έχουν είναι ικανά να αποδώσουν καλύτερα σε πληθώρα εργασιών σε σχέση με τα LSTM. Τα βασικά χαρακτηριστικά τους είναι η παραγωγή εξόδου ίσου μήκους με την είσοδο και το γεγονός πως δεν υπάρχει διαρροή πληροφορίας από τις μελλοντικές εισόδους κατά την επεξεργασία της τωρινής. Για την επίτευξη αυτών, το πρώτο στρώμα των TCN είναι ένα μονοδιάστατο πλήρως συνδεδεμένο στρώμα συνέλιξης, με το μήκος του κάθε στρώματος να παραμένει ίδιο με το μήκος του στρώματος εισόδου. Παράλληλα, οι συνελίξεις που πραγματοποιούνται είναι αιτιατές, δηλαδή το αποτέλεσμα κάθε συνέλιξης κατά μια χρονική στιγμή εξαρτάται αποκλειστικά από την έξοδο του προηγούμενου στρώματος έως και τη χρονική στιγμή αυτή.

Όπως αναφέρθηκε και παραπάνω, η δομή του TransFetch βασίζεται στη χρήση ενός TCN σε συνδυασμό με τον



Σχήμα 4.6: Δομή του μοντέλου TransFetch

μηχανισμό προσοχής πολλαπλών κεφαλών. Για να τροφοδοτηθούν τα δεδομένα εισόδου στο πρώτο στρώμα του δικτύου, το οποίο είναι ένα μονοδιάστατο πλήρως συνδεδεμένο συνελικτικό στρώμα, θα πρέπει να μορφοποιηθούν σε πίνακα. Ο πίνακας αυτός δημιουργείται για κάθε είσοδο κρατώντας ιστορικό μεγέθους L για κάθε διεύθυνση και μέσω της κατάτμησης των διευθύνσεων που αποθηκεύονται. Με τον τρόπο αυτό, δημιουργείται ένας πίνακας μεγέθους (L+1) * C, όπου C το πλήθος των τμημάτων στα οποία διαχωρίζεται η διεύθυνση. Στην αρχική μορφή του δικτύου, η κατάτμηση των διευθύνσεων πραγματοποιείται με δύο τρόπους. Η διεύθυνση μπορεί να διαχωρισθεί στα στοιχεία της, δηλαδή τον αριθμό σελίδας, τον αριθμό block και τη μετατόπιση εντός της σελίδας, είτε να διαχωρισθεί σε κομμάτια συγκεκριμένου μεγέθους. Στη συγκεκριμένη εργασία, γίνεται χρήση του ιστορικού πρόσβασης σε εικονικές σελίδες, οπότε δεν έχει νόημα ο διαχωρισμός της διεύθυνσης στα στοιχεία της. Στο στρώμα συνέλιξης, κάθε είσοδος πρέπει να διαχωρισθεί σε κομμάτια (patches), ώστε να ακολουθηθεί η διαδικασία της συνέλιξης, όπως αναφέρεται στην ενότητα 3.3. Για το συγκεκριμένο σκοπό, ο πίνακας εισόδου διαχωρίζεται στις ξεχωριστές γραμμές του και προστίθεται σε κάθε μια από αυτές η απαραίτητη κωδικοποίηση. Αρχικά προστίθεται κωδικοποίηση θέσης σε συνδυασμό με μια εκπαιδεύσιμη κωδικοποίηση ταξινόμησης[62]. Οι κωδικοποιήσεις αυτές επιτρέπουν στο μοντέλο να λάβει υπόψιν τις χωρικές εξαρτήσεις των δεδομένων, και να παράγει προβλέψεις λαμβάνοντας υπόψιν όλα τα τμήματα μιας διεύθυνσης. Παράλληλα, η είσοδος ενισχύεται και με κωδικοποίηση συμφραζομένων (Context Embedding), με σκοπό τη χρήση του αριθμού σελίδας σε συνδυασμό με τον κατακερματισμό (Hash) του μετρητή προγράμματος για την αύξηση της απόδοσης του μοντέλου. Στην παρούσα εργασία, η κωδικοποίηση αυτή χρησιμοποιείται με σκοπό τη συσχέτιση συγκεκριμένων τιμών του Μετρητή Προγράμματος με συγκεκριμένες μελλοντικές διευθύνσεις. Στη συνέχεια, η είσοδος τροφοδοτείται στο στρώμα Transformer του μοντέλου. Το στρώμα αυτό βασίζεται στον μηχανισμό προσοχής πολλών κεφαλών συνδυαστικά με ένα δίκτυο εμπρόσθιας τροφοδότησης. Το στρώμα αυτό δέχεται την είσοδο την και τη μετατρέπει σε τρεις πίνακες μέσω γραμμικής προβολής. Οι 3 αυτοί πίνακες, που ονομάζονται q (Query), k (Key), v (Value) χρησιμοποιούνται για τον παράλληλο υπολογισμό και συνένωση της τιμής του μηχανισμού Προσοχής κάθε κεφαλής. Στη συνέχεια, ακολουθεί το στρώμα ομαδοποίησης (Pooling), στο οποίο οι κωδικοποιήσεις και η πληροφορία που έχει συλλεχθεί από τα προηγούμενα στρώματα αξιοποιούνται για τη δημιουργία διευθύνσεων με τη χρήση των ξεχωριστών τμημάτων. Τέλος, η έξοδος των στρωμάτων τροφοδοτείται σε ένα Perceptron Πολλαπλών Στρωμάτων (Multi-Layer Perceptron), που χρησιμοποιείται για την ταξινόμηση πολλαπλών ετικετών.

4.4.2 Τρόποι Εκπαίδευσης του Μοντέλου

Το σύνολο των δεδομένων εκπαίδευσης αντλείται από το ιστορικό προσβάσεων για τις 100 εκατομμύρια πρώτες εντολές κάθε ίχνους. Ο τρόπος όμως που θα αξιοποιηθεί το ιστορικό αυτό, το πλήθος των δεδομένων που θα χρησιμοποιηθούν ως δεδομένα εκπαίδευσης και ο τρόπος με τον οποίο θα τροφοδοτηθούν στο δίκτυο έχουν πολλή μεγάλη σημασία για την ακρίβεια του μοντέλου που θα προκύψει.

Στην αρχική του μορφή, το TransFetch χρησιμοποιεί το ιστορικό που προκύπτει από τις πρώτες 40 εκατομμύρια εντολές για την εκπαίδευση του μοντέλου, τις επόμενες 10 εκατομμύρια εντολές για την επαλήθευση του μοντέλου και την εισαγωγή μικρών διορθώσεων στα βάρη του, με το μοντέλο να είναι υπεύθυνο για την παραγωγή του αρχείου προανάκλησης για τις τελευταίες 50 εκατομμύρια εντολές. Ακόμη, σημειώνεται πως κάθε μοντέλο εκπαιδεύεται αποκλειστικά σε ένα trace και παράγει προανακλήσεις για το trace αυτό.

Στην παρούσα διπλωματική εργασία, έχουν αξιολογηθεί διαφορετικοί τρόποι εκπαίδευσης του ίδιου μοντέλου. Αρχικά, έχει χρησιμοποιηθεί ο τρόπος εκπαίδευσης που χρησιμοποιεί και το TransFetch, ο οποίος χρησιμοποιείται και σε άλλες εργασίες παρόμοιου περιεχομένου[10]. Καθώς το ιστορικό πρόσβασης που έχουμε καταγράψει αποτελείται από το σύνολο των αιτημάτων πρόσβασης στη μνήμη, ενώ το δίκτυο χρειάζεται να εκτελέσει εν τέλει προανακτήσεις μόνο για τις αστοχίες στην STLB, αφού δημιουργηθεί το ιστορικό για όλες τις προσβάσεις, στη συνέχεια τροφοδοτούνται στην είσοδο του δικτύου μονάχα οι προσβάσεις στις οποίες υπήρξε αστοχία. Ακόμη, έχουμε εκτελέσει και πειράματα όπου το μοντέλο εκπαιδεύεται μέσω δειγματοληψίας. Σημειώνεται πως εφόσον το μοντέλο κατά την αξιολόγηση μιας εισόδου λαμβάνει υπόψιν και το ιστορικό των προσβάσεων που πραγματοποιήθηκαν πριν από αυτή, δεν είναι δυνατή η πραγματικά τυχαία δειγματοληψία για την εκπαίδευση του μοντέλου. Για τον λόγο αυτό, τα μοντέλα έχουν εκπαιδευτεί με δύο επιπλέον τρόπους:

> *(i)* Αρχικά διαβάζεται το σύνολο των δεδομένων από τα οποία θέλουμε να αποκτήσουμε δειγματοληπτικά την είσοδο, ώστε να αποκτήσουμε την απαραίτητη πληροφορία για κάθε πρόσβαση. Έπειτα, επιλέγονται τυχαία δείγματα από το σύνολο ώστε να τροφοδοτηθούν στο μοντέλο ως είσοδος.
>
> *(ii)* Επιλέγονται τυχαία 20 παράθυρα, κάθε ένα εκ των οποίων περιλαμβάνει 1 εκατομμύριο συνεχείς εντολές. Στη συγκεκριμένη περίπτωση το ιστορικό προσβάσεων για τις πρώτες εντολές κάθε διαστήματος θα περιέχει λανθασμένες πληροφορίες, θεωρούμε όμως πως το πλήθος των δεδομένων επισκιάζει την απουσία αυτή.

Όπως αναφέρθηκε και παραπάνω, οι συγκεκριμένοι τρόποι εκπαίδευσης του μοντέλου αντλούν τα δεδομένα εισόδου από το ιστορικό προσβάσεων μιας εφαρμογής, και παράγουν ένα αρχείο προανακλήσεων για τη συγκεκριμένη εφαρμογή. Παρόλο που η διαδικασία αυτή αποτελεί συνήθη τακτική κατά την εκπαίδευση νευρωνικών δικτύων για προανάκληση, στη συγκεκριμένη εργασία έχει γίνει απόπειρα εκπαίδευσης του δικτύου σε δεδομένα που αντλούνται από ένα σύνολο εφαρμογών, και η μετέπειτα αξιολόγησή της απόδοσής του για κάθε ένα από αυτά.

Σε κάθε περίπτωση, η εκπαίδευση των μοντέλων και η παραγωγή του αρχείου προανακλήσεων αποτελεί διαδικασία πλήρως ξεχωριστή του προσομοιωτή ChampSim. Συγκεκριμένα, το μοντέλο εκπαιδεύεται χρησιμοποιώντας κάποιον από τους τρόπους που αναφέρθηκαν παραπάνω ώστε να παραγάγει το αρχείο προανακλήσεων, το οποίο περιλαμβάνει συνδυασμούς αριθμού εντολής και της διεύθυνσης που το μοντέλο θεωρεί πως πρέπει να προανακληθεί κατά την εκτέλεση της εν λόγω εντολής. Στη συνέχεια, χρησιμοποιείται ο προσομοιωτής ChampSim σε συνδυασμό με έναν απλό προανακλητή ο οποίος χρειάζεται να διαβάσει απλώς το αρχείο που παρήγαγε το μοντέλο και να εκτελέσει τα αντίστοιχα αιτήματα προανάκλησης κατά την εκτέλεση των κατάλληλων εντολών. Κατά τη διαδικασία αυτή, δεν λαμβάνεται υπόψιν η καθυστέρηση πρόβλεψης (Inference Latency) που θα προσέθετε η χρήση ενός νευρωνικού δικτύου για την προανάκληση σε πραγματικό σύστημα. Σημειώνεται ακόμη πως το μοντέλο εκπαιδεύεται στο σύνολο των αιτημάτων προσβάσεων στη μνήμη και μπορεί να παράγει αίτημα προανάκλησης για κάθε έναν από τους αντίστοιχους αριθμούς εντολής, εφόσον έχει αρκετά υψηλό confidence. Παρόλα αυτά, κατά την εκτέλεση της προσομοίωσης ο ChampSim εκτελεί αιτήματα προανάκλησης μονάχα όταν προκύπτει αστοχία στην STLB, ανεξάρτητα από το αρχείο προανακλήσεων. Ως εκ τούτου, δεν εκτελούνται όλα τα αιτήματα προανάκλησης που προτείνει το μοντέλο, παρά μόνο όσα συμπίπτουν με αστοχίες στην STLB.

Ακόμη, στο TransFetch έχουμε προσθέσει την έννοια της επικαιρότητας (Timeliness). Κατά τη διαδικασία της

εκπαίδευσης και επαλήθευσης, το μοντέλο χρησιμοποιεί και επιχειρεί να προβλέψει την αλληλουχία των αιτημάτων πρόσβασης στη μνήμη. Λόγω αυτού, κατά τη δημιουργία του αρχείου προανακλήσεων, το μοντέλο επιχειρεί εκ νέου να προβλέψει την αλληλουχία προσβάσεων στη μνήμη. Από την παραπάνω ανάλυση όμως, προκύπτει πως για να θεωρηθεί επιτυχές ένα αίτημα προανάκλησης στην TLB, θα πρέπει να εκτελεστεί αρκετές εντολές προτού χρειαστεί. Για να επιτευχθεί λοιπόν η εισαγωγή της έννοιας της Επικαιρότητας στο μοντέλο με τρόπο που είναι εύκολα παραμετροποιήσιμος, έχουν πραγματοποιηθεί αλλαγές στον τρόπο με τον οποίο διαβάζονται τα δεδομένα εισόδου. Συγκεκριμένα, χρησιμοποιείται μια ουρά η οποία περιλαμβάνει τους αριθμούς εντολής για κάθε αίτημα πρόσβασης στη μνήμη, και τους τροφοδοτεί με καθυστέρηση στο νευρωνικό δίκτυο. Με τον τρόπο αυτό, όταν το μοντέλο εκτελεί αιτήματα προανάκλησης, αυτά θα αφορούν διευθύνσεις που θα είναι απαραίτητες σε επόμενες εντολές. Με τον τρόπο αυτό, γίνεται εύκολη η παραμετροποίηση της Επικαιρότητας, αλλάζοντας το μέγεθος της ουράς που χρησιμοποιείται. Η μεταβολή του μεγέθους της ουράς οδηγεί σε ίση μεταβολή του αριθμού των εντολών μετά την παρούσα, για την οποία εκτελείται εν τέλει η προανάκληση.

4.4.3 Παράμετροι Μοντέλου

Πέραν των τροποποιήσεων που ήταν απαραίτητες για την λειτουργία του TransFetch ως προανακλητή στην TLB, εκτελέστηκαν πειράματα για την εύρεση των παραμέτρων που οδηγούν σε μοντέλο βέλτιστης απόδοσης και ακρίβειας προβλέψεων. Ορισμένες από τις παραμέτρους που εξετάστηκαν καθώς και μια σύντομη περιγραφή της λειτουργίας τους εντός του δικτύου παρουσιάζεται παρακάτω.

1. **Παράμετροι Εισόδου και εξόδου**

   (α′) *Split Bits:* Το πλήθος των ψηφίων που θα περιέχονται σε κάθε τμήμα μετά την κατάτμηση της διεύθυνσης

   (β′) *LookBack:* Το πλήθος των προηγούμενων εγγραφών που λαμβάνονται υπόψιν κατά την επεξεργασία κάθε αιτήματος

   (γ′) *Bit Map Size:* Το μέγεθος του δυαδικού χάρτη που χρησιμοποιείται. Το μέγεθος ισούται και με τη μέγιστη απόσταση από την τωρινή σελίδα που λαμβάνεται υπόψιν

   (δ′) *Forward Window:* Το πλήθος των μελλοντικών Δέλτα που λαμβάνονται υπόψιν για κάθε αίτημα πρόσβασης

   (ε′) *Degree/Filter size:* Το μέγιστο πλήθος προανακλήσεων που μπορεί να εκτελεστεί σε κάθε εντολή, και το πλήθος των προανακλήσεων που επιτρέπονται εν τέλει. Στην περίπτωση της προανάκλησης σε TLB, έχουμε θέσει τον βαθμό προανάκλησης ίσο με 16, επιτρέπουμε όμως εν τέλει την εκτέλεση μονάχα μίας προανάκλησης, αυτής με το υψηλότερο confidence, εφόσον είναι αρκετά υψηλό.

2. **Παράμετροι στρώματος Transformer:**

   (α′) *Depth:* Το βάθος του στρώματος Transformer, δηλαδή το πλήθος των στρωμάτων που περιλαμβάνει

   (β′) *Heads:* Το πλήθος των κεφαλών προσοχής του στρώματος Transformer

   (γ′) *Dimension:* Η διάσταση του στρώματος Transformer

3. **Παράμετροι Εκπαίδευσης**

   (α′) *Optimizer:* Ο αλγόριθμος βελτιστοποίησης που χρησιμοποιείται

   (β′) *Learning Rate:* Ο ρυθμός μάθησης που θα αξιοποιήσει ο αλγόριθμος βελτιστοποίησης

   (γ′) *Epochs:* Το μέγιστο πλήθος εποχών για τις οποίες θα εκπαιδευτεί το μοντέλο

   (δ′) *Early Stop:* Το μέγιστο πλήθος των εποχών δίχως βελτίωση της ακρίβειας του μοντέλου έως ότου σταματήσει η διαδικασία εκπαίδευσης

Στον πίνακα 4.5 παρουσιάζονται οι τιμές που οδηγούν στη βέλτιστη απόδοση, όσον αφορά τον γεωμετρικό μέσο όρο των εντολών που εκτελούνται ανά κύκλο.

	Παράμετρος	Τιμή
Είσοδος/Έξοδος	Split Bits	6
	Lookback	8
	Forward Window	128
	Bitmap Size	256
	Degree	16
Στρώμα Transformer	Depth	2
	Heads	4
	Dimension	128
Παράμετροι Εκπαίδευσης	Optimizer	ADAM
	Epochs	200
	Learning Rate	$2 * 10^{-4}$

Πίνακας 4.5: Βέλτιστες Υπερπαράμετροι Μοντέλου

Κεφάλαιο 5

Πειραματική Αξιολόγηση & Παρουσίαση Αποτελεσμάτων

Αντικείμενο του κεφαλαίου αυτού είναι η παρουσίαση των αποτελεσμάτων των προσομοιώσεων που εκτελέστηκαν και η αξιολόγηση της χρήσης της τροποποιημένης έκδοσής του TransFetch ως προανακλητή στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Οι μετρικές που χρησιμοποιήθηκαν για τη σύγκριση των αποτελεσμάτων είναι η ακρίβεια των προανακλητών, η κάλυψη των αστοχιών και το κέρδος συγκριτικά με την περίπτωση όπου δεν χρησιμοποιείται προανάκληση στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Η ακρίβεια ορίστηκε ως το ποσοστό των αναζητήσεων στην Ουρά Προανακλήσεων που ήταν επιτυχείς, δια τον αριθμό των προανακλήσεων που πραγματοποιήθηκαν, ενώ η κάλυψη υπολογίστηκε διαιρώντας τον αριθμό των επιτυχών αναζητήσεων στην Ουρά Προανακλήσεων, δια τον αριθμό των αστοχιών στην STLB στην περίπτωση όπου δεν χρησιμοποιείται προανακλητής σε αυτή. Τέλος, το κέρδος υπολογίστηκε ως η αύξηση του αριθμού εντολών ανά κύκλο (IPC - Instructions per Cycle) σε σχέση με την εκτέλεση δίχως προανακλητή στην STLB. Και για τις τρεις μετρικές, λαμβάνεται ο γεωμετρικός μέσος όρος των αποτελεσμάτων για όλες τις εφαρμογές που λαμβάνονται υπόψιν ανά περίπτωση.

Ακόμη, έχουν επιλεχθεί ορισμένες, ιδιαίτερα TLB - intensive εφαρμογές, για να τονιστούν σε μεγαλύτερο βαθμό οι διαφοροποιήσεις μεταξύ των προανακλητών. Οι εφαρμογές αυτές επιλέχθηκαν με βάση τις αστοχίες STLB που παρουσιάζουν ανά 1000 εντολές (MPKI). Συγκεκριμένα, ως TLB - intensive θεωρήθηκαν οι εφαρμογές που έχουν τουλάχιστον 1 αστοχία STLB ανά 1000 εντολές. Εξαίρεση αποτελούν οι εφαρμογές της σουίτας GAP στην ενότητα "Αξιολόγηση μεθόδων εκπαίδευσης", όπου η τιμή αυτή ορίστηκε χαμηλότερα λόγω του μικρού αριθμού εφαρμογών που τηρούσαν τις προϋποθέσεις. Οι παράμετροι του προσομοιωτή ChampSim έχουν παραμείνει σταθερές και ίσες με αυτές που παρουσιάστηκαν στο Κεφάλαιο 4 κατά την εκτέλεση του συνόλου των προσομοιώσεων.

Ως σημείο αναφοράς, έχουν χρησιμοποιηθεί ορισμένοι από τους προανακλητές που αναφέρθηκαν στο Κεφάλαιο 2 και δεν βασίζονται στη μηχανική μάθηση. Οι προανακλητές αυτοί, στους οποίους παρακάτω αναφερόμαστε ως "κλασσικούς" προανακλητές, είναι ο Ακολουθιακός Προανακλητής(SP), ο Προανακλητής Αυθαίρετου Βήματος(ASP), ο Προανακλητής Απόστασης (DP) και ο Agile TLB Prefetcher(ATP). Ακόμη, χρησιμοποιούνται και Προανακλητές Oracle διαφορετικών τιμών "επικαιρότητας", ως προανακλητές βέλτιστων δυνατοτήτων, καθώς και για την επισήμανση της σημασίας της επικαιρότητας προανακλήσεων.

Στις αξιολογήσεις που ακολουθούν, τα μοντέλα που χρησιμοποιήθηκαν βασίζονται στην έκδοση του TransFetch που οδήγησε σε μοντέλα βέλτιστης ακρίβειας και απόδοσης, δηλαδή αυτή που παρουσιάζεται στην ενότητα 4.4.3. Οι υπερπαράμετροι παραμένουν σταθερές για το σύνολο των πειραμάτων, ενώ αλλαγές στο δίκτυο έχουν γίνει όπου αυτό ήταν απαραίτητο, όπως θα παρουσιαστούν παρακάτω. Στη συνέχεια, για κάθε ενότητα του κεφαλαίου παρουσιάζονται παρατηρήσεις επί των αποτελεσμάτων και ακολουθούν τα διαγράμματα που χρησιμοποιήθηκαν για την εξαγωγή τους.

5.1 Αξιολόγηση της βέλτιστης "Επικαιρότητας"

Στην ενότητα αυτή αξιολογείται η επιρροή της τιμής της 'Επικαιρότητας' στη λειτουργία του δικτύου ως προανακλητή στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης. Όπως αναφέρθηκε και παραπάνω, η έννοια της επικαιρότητας προστέθηκε στο δίκτυο κατά την επεξεργασία της εισόδου. Στα αποτελέσματα που παρατίθενται, φαίνονται για κάθε μετρική τα αποτελέσματα των βασικών προανακλητών, των προανακλητών νευρωνικού δικτύου για τιμές επικαιρότητας 0, 6, 9 και 12 και των προανακλητών Oracle για τις ίδιες τιμές επικαιρότητας. Σημειώνεται πως για τιμή επικαιρότητας ίση με 0, το νευρωνικό δίκτυο επιχειρεί να προβλέψει την αμέσως επόμενη διεύθυνση στην οποία θα υπάρξει αστοχία. Τα νευρωνικά δίκτυα καθώς και οι προανακλητές Oracle σημειώνονται στα διαγράμματα συνδυαστικά με την τιμή επικαιρότητάς τους. Στα πειράματα που εκτελέστηκαν, έχει δημιουργηθεί ξεχωριστό μοντέλο για κάθε εφαρμογή, με τα δεδομένα εκπαίδευσης να αντλούνται από το σύνολο των προσβάσεων που προέκυψαν για τις πρώτες 40 εκατομμύρια εντολές. Για κάθε μετρική, παρουσιάζεται η επίδοση των προανακλητών για τα προγράμματα κάθε σουίτας ξεχωριστά, η επίδοση στο σύνολο των προγραμμάτων, και τέλος η επίδοση των προανακλητών στα TLB-intensive προγράμματα.

5.1.1 Ακρίβεια

Η πρώτη μετρική με βάση την οποία θα αξιολογηθούν τα μοντέλα είναι η ακρίβεια, δηλαδή το ποσοστό των προανακλήσεων που εκτελέστηκαν και οδήγησαν σε ευστοχία στην Ουρά Προανάκλησης. Τα αποτελέσματα για κάθε σουίτα ξεχωριστά καθώς και για το σύνολο των εφαρμογών φαίνονται στα διαγράμματα 5.1. Στο διάγραμμα που αφορά το σύνολο των εφαρμογών, βλέπουμε πως ανεξαρτήτως τιμής επικαιρότητας, η ακρίβεια του νευρωνικού δικτύου μας φαίνεται να παρουσιάζει μια μικρή μείωση καθώς η επικαιρότητα αυξάνεται, παραμένει όμως συγκρίσιμη με τους υπόλοιπους προανακλητές. Συγκεκριμένα, η ακρίβεια του μοντέλου ξεπερνάται μονάχα από τον Προανακλητή Αυθαίρετου Βήματος, λόγω της συνθήκης σταθερότητας βήματος που περιλαμβάνει προτού εκτελεστεί προανάκληση, και τους Προανακλητές Oracle, οι οποίοι αστοχούν μόνο εφόσον τα δεδομένα προανάκλησης που εκτελούν εισαχθούν στην STLB λόγω κάποιας επόμενης εντολής, προτού προστεθούν στην Ουρά Προανάκλησης. Λαμβάνοντας υπόψιν κάθε σουίτα προγραμμάτων ξεχωριστά, φαίνεται πως η ακρίβεια του δικτύου παραμένει σταθερά συγκρίσιμη με αυτή των υπόλοιπων κλασσικών προανακλητών, πέραν του ASP. Εξαίρεση αποτελούν οι εφαρμογές της σουίτας SPEC06. Στις εφαρμογές αυτές, η ακρίβεια όλων των προανακλητών φαίνεται να είναι υψηλότερη, με την ακρίβεια των νευρωνικών δικτύων να είναι εμφανώς βελτιωμένη και να ξεπερνά όλους τους υπόλοιπους προανακλητές για τιμή επικαιρότητας ίση με 6. Όσον αφορά την ακρίβεια των μοντέλων με βάση την τιμή επικαιρότητας, οι διαφορές φαίνεται να είναι σχετικά μικρές, όμως στη γενική περίπτωση η μεγαλύτερη ακρίβεια φαίνεται πως προκύπτει για τιμή επικαιρότητας ίση με 6.

Περνώντας στην αξιολόγηση της ακρίβειας για τις TLB-intensive εφαρμογές, στα διαγράμματα 5.2 βλέπουμε πως η ακρίβεια όλων των προανακλητών, και ιδιαίτερα των νευρωνικών δικτύων, έχει αυξηθεί αισθητά. Συγκεκριμένα, στις εφαρμογές αυτές φαίνεται πως ξεπερνούν σταθερά σε ακρίβεια όλους τους προανακλητές, με εξαίρεση τους προανακλητές ASP και Oracle, ενώ η βέλτιστη τιμή επικαιρότητας φαίνεται να είναι και πάλι ίση με 6, καθώς παρουσιάζει βελτιωμένη ακρίβεια όταν λαμβάνονται υπόψιν όλα τα προγράμματα. Η αύξηση ακρίβειας στα μοντέλα κατά την αξιολόγηση σε TLB-intensive εφαρμογές είναι αναμενόμενη, καθώς οι εφαρμογές αυτές περιέχουν πιο πυκνό ιστορικό προσβάσεων και παρέχουν στο δίκτυο περισσότερα δεδομένα ώστε να εκπαιδευτεί. Συγκεκριμένα στις εφαρμογές της σουίτας SPEC06 βλέπουμε πως για τιμή επικαιρότητας ίση με 6, το δίκτυο έχει μεγαλύτερη ακρίβεια ακόμη και από τους προανακλητές Oracle.

5.1.2 Κάλυψη

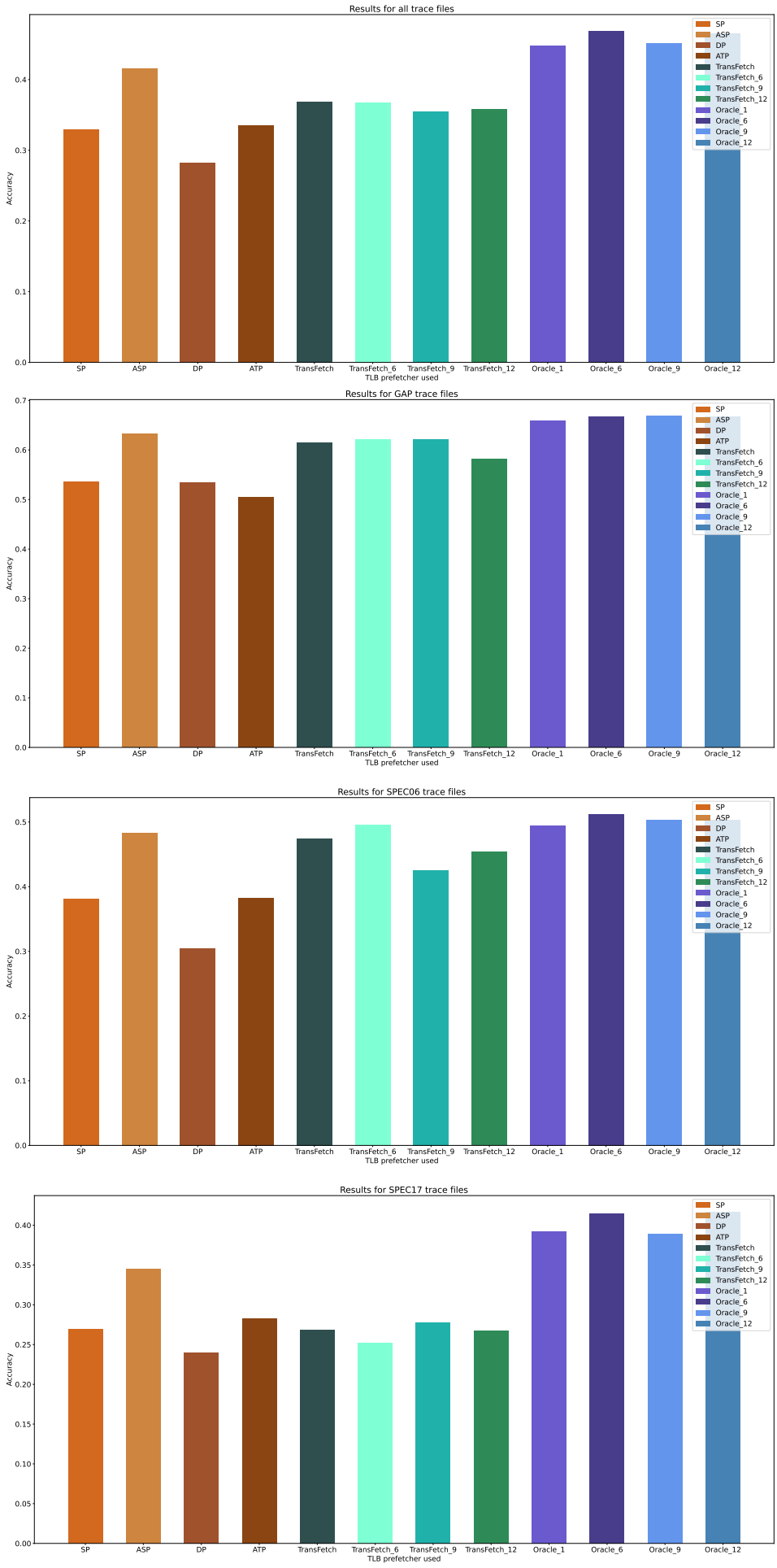
Όσον αφορά την κάλυψη, δηλαδή το ποσοστό των αστοχιών στην STLB για τις οποίες υπήρξαν επιτυχείς αναζητήσεις στην ουρά προανακλήσεων, τα αποτελέσματα φαίνονται στα σχήματα 5.3. Σε αυτά, βλέπουμε πως τα νευρωνικά δίκτυα φαίνεται να αποδίδουν χειρότερα σε σχέση με τους υπόλοιπους προανακλητές, για όλες τις σουίτες. Το γεγονός αυτό πιθανώς έγκειται στη σχετικά υψηλή τιμή αυτοπεποίθησης που είναι απαραίτητη για την εκτέλεση προανακλήσεων. Άλλωστε, οι έννοιες της ακρίβειας και της κάλυψης κατά την προανάκληση μπορούν να θεωρηθούν ως αντιστρόφως ανάλογες, καθώς υψηλή ακρίβεια προανάκλησης συνήθως συνεπάγεται λιγότερα αιτήματα προανάκλησης. Αντιθέτως, υψηλή τιμή κάλυψης είναι πολύ πιθανό να συνοδεύεται από πολ-

λαπλά αιτήματα προανάκλησης τα οποία δεν ήταν επιτυχή και οδηγεί σε μόλυνση της μνήμης. Περνώντας στα διαγράμματα 5.4, που αφορούν τις TLB-intensive εφαρμογές, τα αποτελέσματα φαίνεται να είναι σαφώς καλύτερα για το δίκτυό μας. Συγκεκριμένα, οι τιμές κάλυψης του δικτύου για τις εφαρμογές αυτές έχει σχεδόν διπλασιαστεί, και ξεπερνά τους κλασσικούς προανακλητές που έχουν χρησιμοποιηθεί, πέραν του ATP. Συγκεκριμένα για τη σουίτα προγραμμάτων SPEC06, βλέπουμε πως το σύνολο των προανακλητών παρουσιάζουν σημαντικά χαμηλότερη κάλυψη σε σχέση με τα αποτελέσματα σε όλες τις εφαρμογές της σουίτας. Παρόλα αυτά, το νευρωνικό δίκτυο φαίνεται να μην παρουσιάζει πολύ μεγάλη πτώση, και ως εκ τούτου είναι συγκρίσιμο με τους υπόλοιπους προανακλητές. Ακόμη, στις εφαρμογές της σουίτας SPEC17, τα δίκτυα φαίνεται να ξεπερνούν σε κάλυψη όλους τους προανακλητές, φυσικά με εξαίρεση τους προανακλητές Oracle.

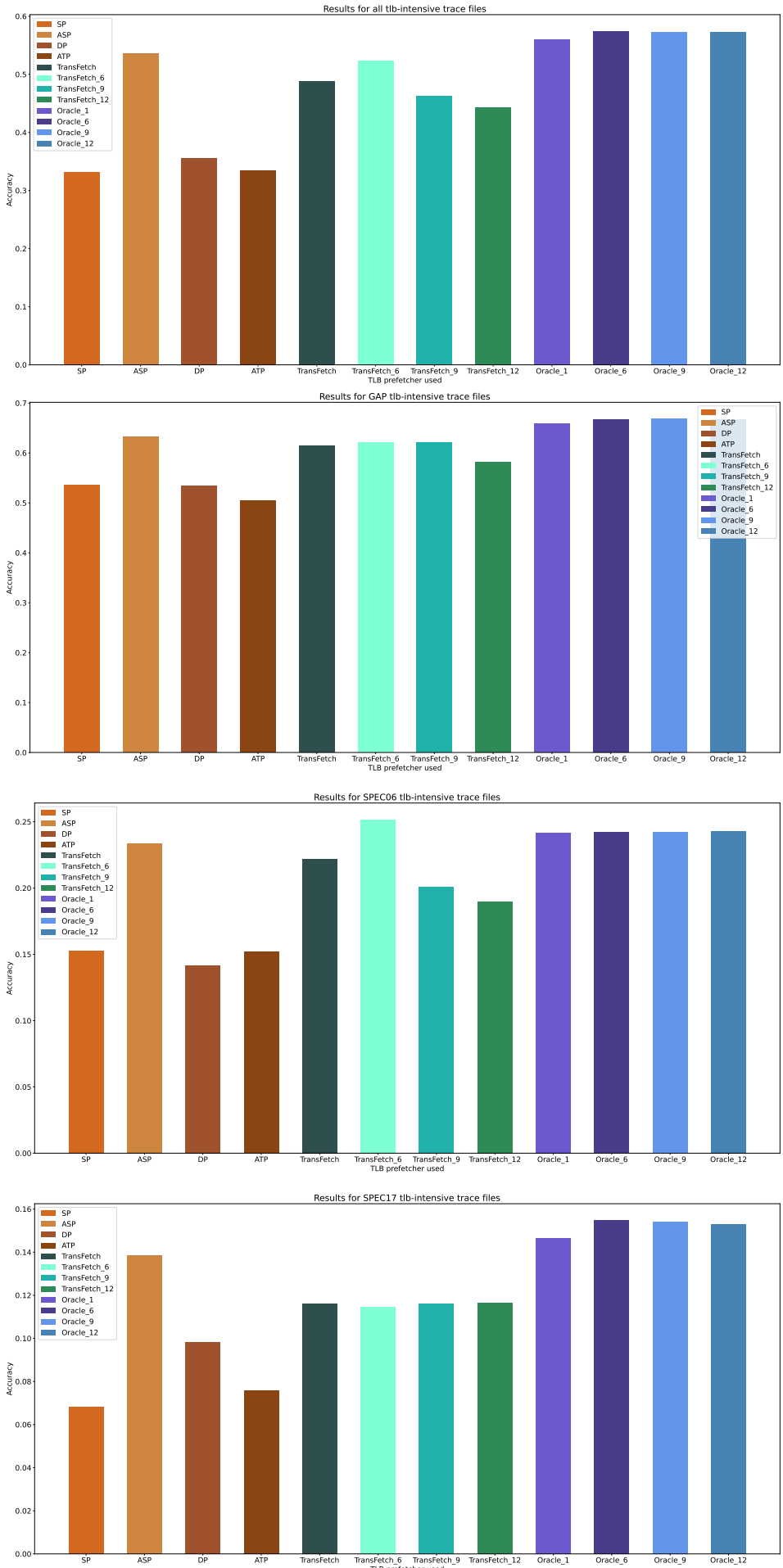
5.1.3 Αύξηση Επίδοσης

Στις μετρικές που έχουν ληφθεί υπόψιν μέχρι στιγμής, οι διαφορετικές εκδόσεις του δικτύου φαίνεται να μην παρουσιάζουν αισθητές διαφορές όσον αφορά την απόδοσή τους. Η χρησιμότητα της έννοιας της επικαιρότητας κατά την προανάκληση, γίνεται όμως εμφανής στα διαγράμματα 5.5, συγκρίνοντας τόσο την απόδοση των διαφορετικών εκδοχών του νευρωνικού δικτύου, όσο και τις εκδοχές των προανακλητών Oracle. Στα διαγράμματα αυτά βλέπουμε το κέρδος που παρέχει στις εντολές που εκτελούνται ανά κύκλο κάθε προανακλητής, σε σχέση με την περίπτωση όπου δεν χρησιμοποιείται προανάκληση. Συγκρίνοντας την απόδοση των προανακλητών για το σύνολο των εφαρμογών, βλέπουμε πως η απλή εκδοχή του νευρωνικού δικτύου ξεπερνά ή φτάνει την απόδοση κάθε κλασσικού προανακλητή, με εξαίρεση τον ATP. Με την εισαγωγή της έννοιας της επικαιρότητας, η απόδοση του δικτύου αυξάνεται σημαντικά, ξεπερνώντας με διαφορά τους κλασσικούς προανακλητές για τιμές ίσες ή μεγαλύτερες του 6, εάν ληφθεί υπόψιν το σύνολο των εφαρμογών. Συγκεκριμένα για τη σουίτα GAP, ο ATP και ο SP φαίνεται να ξεπερνούν σε απόδοση όλες τις εκδοχές του δικτύου και να πλησιάζουν την απόδοση του Oracle1. Το γεγονός αυτό, πιθανώς οφείλεται στη φύση των εφαρμογών της σουίτας GAP τα οποία αφορούν κατά κύριο λόγο επεξεργασία γράφων, και φαίνεται να αποδίδουν καλά με ακολουθιακά σχήματα προανάκλησης. Παράλληλα, για την ίδια σουίτα, βλέπουμε πως οι τιμές κέρδους στη γενική περίπτωση είναι αρκετά χαμηλές σε σχέση με τις εφαρμογές των υπόλοιπων σουίτων. Βλέπουμε επίσης πως η βέλτιστη τιμή επικαιρότητας, καθώς και η σημασία της έννοιας στην προανάκληση διαφέρει αισθητά ανά εφαρμογή. Συγκεκριμένα για τις εφαρμογές της σουίτας SPEC17, βλέπουμε πως η βέλτιστη τιμή επικαιρότητας για το δίκτυο ισούται με 12, ενώ έχει σχετικά μικρότερη επίπτωση στην απόδοση. Αντιθέτως, ο προανακλητής Oracle φαίνεται να παρουσιάζει βέλτιστη απόδοση για τιμή επικαιρότητας ίση με 6. Επίσης, στην ίδια σουίτα, ο προανακλητής Oracle 1, ο οποίος εκτελεί αίτημα προανάκλησης για την ακριβώς επόμενη αστοχία, ξεπερνά σε απόδοση όλες τις εκδοχές του δικτύου μας, κάτι που δεν ισχύει στις εφαρμογές της σουίτας SPEC06, όπου η τιμή της επικαιρότητα φαίνεται να έχει μεγαλύτερη επιρροή στην απόδοση. Το γεγονός αυτό παρόλα αυτά εξηγείται και από τα αποτελέσματα της ανάλυσης του ιστορικού προσβάσεων για τις εφαρμογές κάθε σουίτας, όπου είδαμε πως το ποσοστό των διαδοχικών προσβάσεων που εκτελούνται εντός 300 κύκλων για τις εφαρμογές της σουίτας SPEC17 είναι χαμηλότερο σε σχέση με τις υπόλοιπες εφαρμογές. Λαμβάνοντας υπόψιν την γενική περίπτωση, η βέλτιστη τιμή επικαιρότητας για το δίκτυο μας φαίνεται να ισούται με 9, και για τον λόγο αυτό θα διατηρηθεί και κατά την εκτέλεση των παρακάτω προσομοιώσεων.

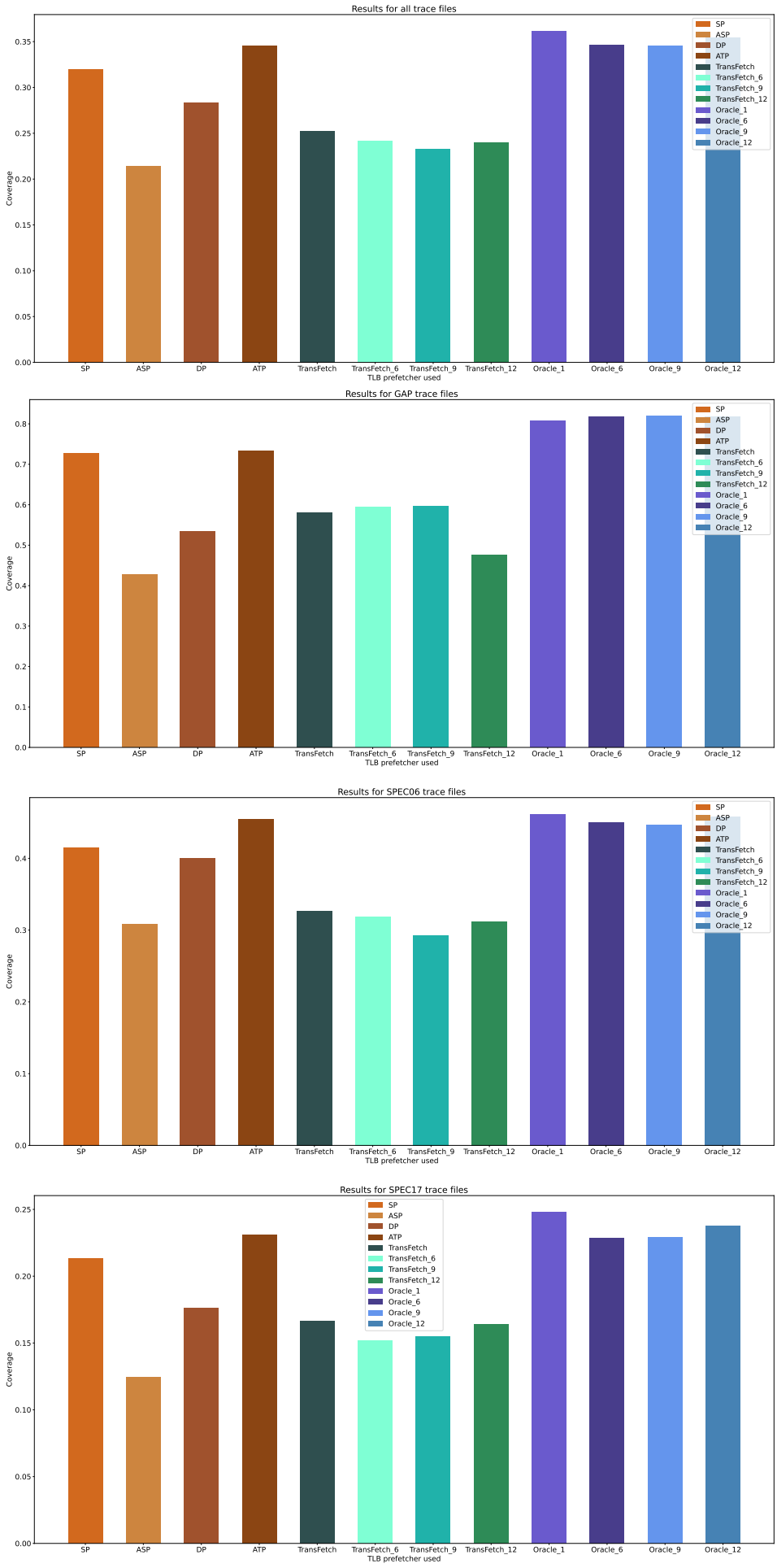
Τα αποτελέσματα για τις TLB-intensive εφαρμογές, τα οποία φαίνονται στα σχήματα 5.6, φαντάζουν πανομοιότυπα με τα αποτελέσματα για το σύνολο των εφαρμογών, με μοναδική εξαίρεση τις τιμές κατά τις οποίες αυξήθηκε η απόδοση. Το γεγονός αυτό πιθανώς οφείλεται στην παρομοίως ανεπαρκή απόδοση όλων των προανακλητών σε εφαρμογές με πολύ χαμηλά ποσοστά αστοχιών στην STLB. Στις εφαρμογές αυτές, ακόμη και εάν δεν χρησιμοποιηθεί προανακλητής το πλήθος των αστοχιών είναι χαμηλό, είτε επειδή δεν υπάρχουν πολλές προσβάσεις στην STLB, είτε επειδή η πλειοψηφία των καταχωρήσεων του πίνακα σελίδων που χρησιμοποιεί η εφαρμογή χωράει στην STLB. Ως εκ τούτου, η χρήση προανακλητή δεν επιφέρει σημαντική αύξηση της επίδοσης, όσο αποδοτικός και εάν είναι. Για την πιο λεπτομερή σύγκριση της απόδοσης των προανακλητών σε τέτοιου είδους ίχνη, θα είχε νόημα η μείωση του μεγέθους της STLB, γεγονός που θα αναδείκνυε την ικανότητα κάθε προανακλητή να εκτελεί ακριβείς, έγκαιρες και σημαντικές προβλέψεις ώστε να βελτιώσει την απόδοση.



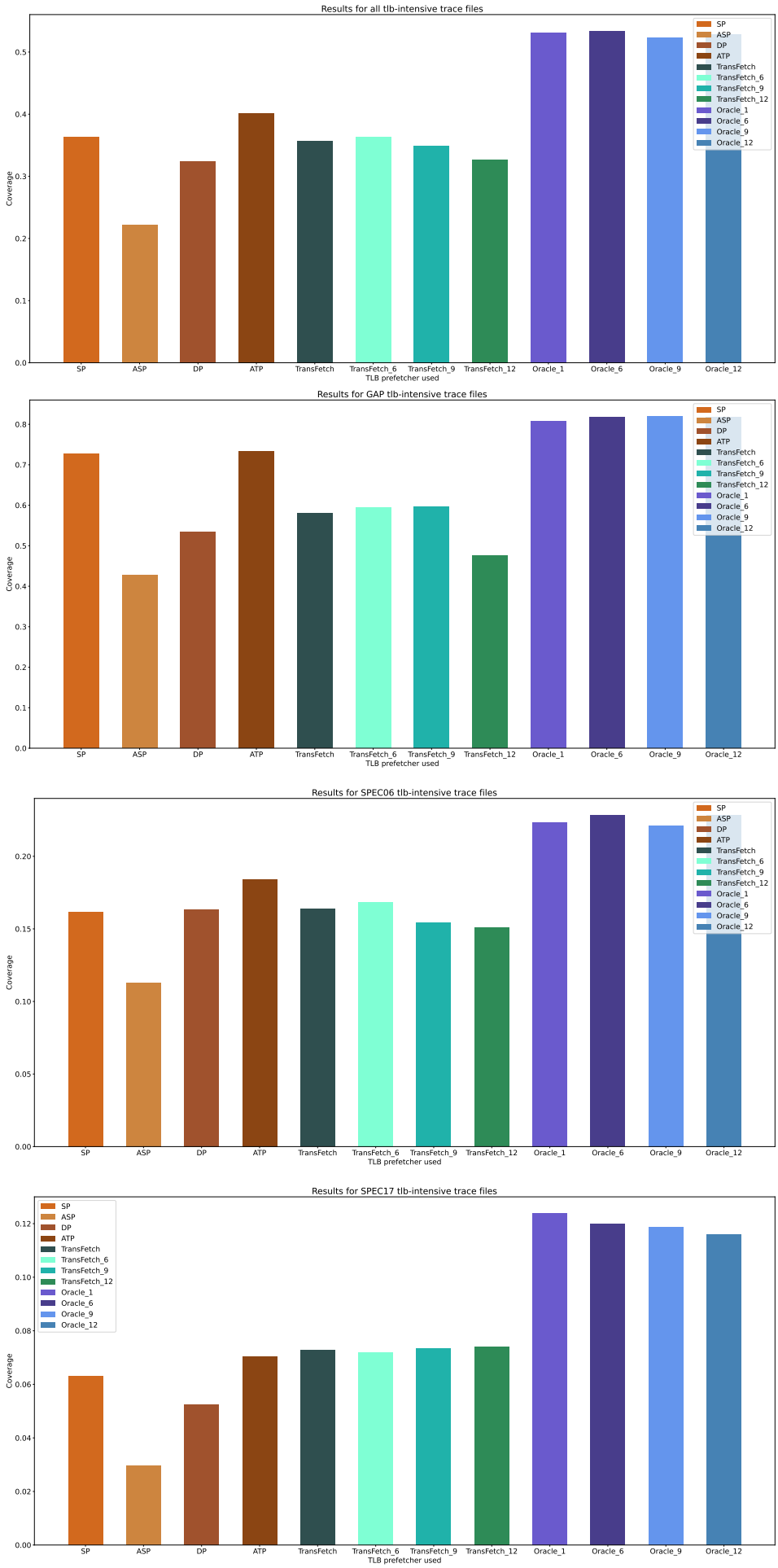
Σχήμα 5.1: Τα αποτελέσματα ακρίβειας για κάθε σουίτα - Αξιολόγηση Επικαιρότητας



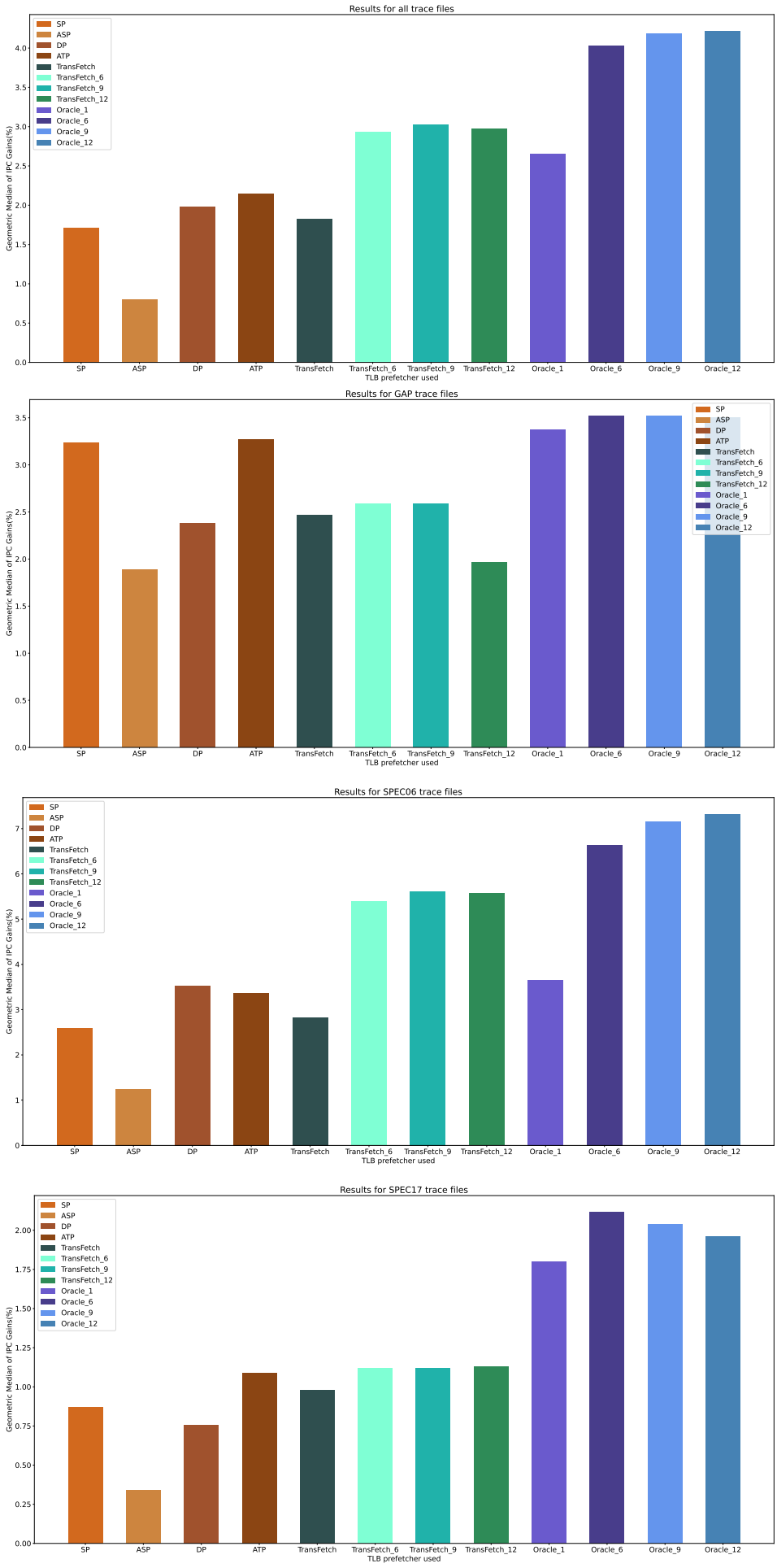
Σχήμα 5.2: Τα αποτελέσματα ακρίβειας για τις TLB-intensive εφαρμογές - Αξιολόγηση Επικαιρότητας



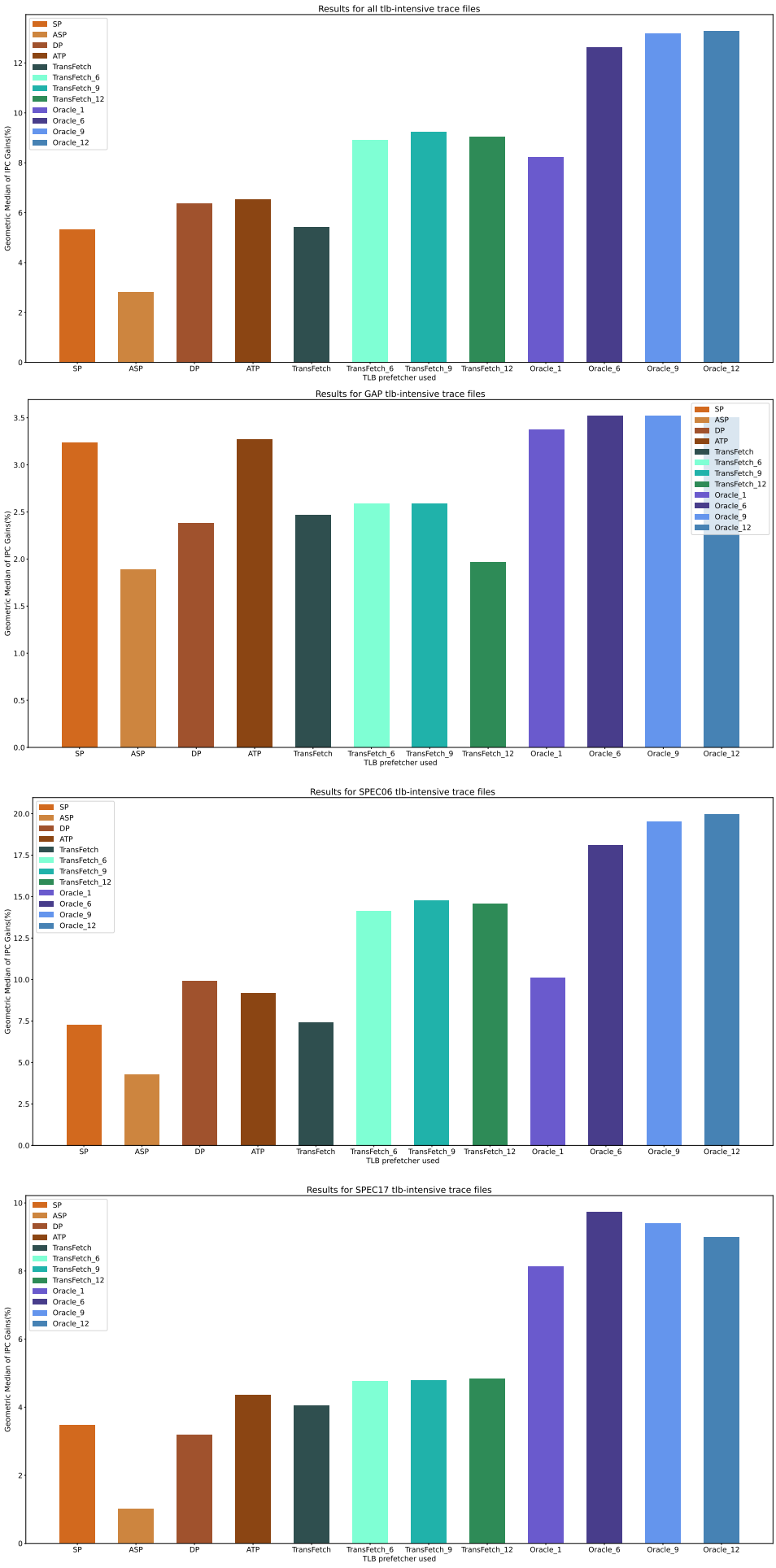
Σχήμα 5.3: Τα αποτελέσματα κάλυψης για κάθε σουίτα - Αξιολόγηση Επικαιρότητας



Σχήμα 5.4: Τα αποτελέσματα κάλυψης για τις TLB-intensive εφαρμογές - Αξιολόγηση Επικαιρότητας



Σχήμα 5.5: Η αύξηση της επίδοσης για κάθε σουίτα - Αξιολόγηση Επικαιρότητας



Σχήμα 5.6: Η αύξηση της επίδοσης για τις TLB-intensive εφαρμογές - Αξιολόγηση Επικαιρότητας

5.2 Αξιολόγηση μεθόδων εκπαίδευσης

Στην ενότητα αυτή αξιολογείται η επιρροή του τρόπου με τον οποίο διαβάζονται τα δεδομένα εισόδου από το δίκτυο στην απόδοσή του ως προανακλητής. Αρχικά, αναφέρουμε πως λόγω των υψηλών απαιτήσεων μνήμης για την εκπαίδευση των μοντέλων αυτών, δεν ήταν δυνατή η αύξηση του πλήθους των εντολών που λαμβάνονται υπόψιν κατά τη δημιουργία των απαραίτητων ιστορικών προσβάσεων. Για τον λόγο αυτό, στην παρούσα ενότητα η εκπαίδευση των δικτύων αφορά τις προσβάσεις που αντλούνται από 20 εκατομμύρια εντολές. Αρχικά, εκπαιδεύτηκαν δίκτυα που αντλούσαν απλώς τις προσβάσεις που πραγματοποιούνταν κατά τις 20 εκατομμύρια πρώτες εντολές. Στη συνέχεια, εκπαιδεύτηκαν δίκτυα τα οποία διάβαζαν τις προσβάσεις που προέκυπταν κατά τις 40 εκατομμύρια πρώτες εντολές, ώστε να χτιστεί το ιστορικό προσβάσεων για κάθε αίτημα, και στη συνέχεια επέλεγαν δειγματοληπτικά τις μισές από αυτές. Τέλος, εκπαιδεύτηκαν και δίκτυα στις προσβάσεις που αντιστοιχούσαν σε 20 παράθυρα του 1 εκατομμυρίου το κάθε ένα. Σε κάθε έναν από τους 3 αυτούς τρόπους εκπαίδευσης, τα μοντέλα εκπαιδεύονται εν τέλει στις προσβάσεις που αντιστοιχούν σε 20 εκατομμύρια εντολές. Τα μοντέλα σημειώνονται στα διαγράμματα ως Classic, Samples και Windows, αντίστοιχα με τη σειρά που αναφέρθηκαν. Σημειώνεται ακόμα πως σε όλα τα μοντέλα είχε προστεθεί τιμή επικαιρότητας ίση με 9. Ακόμη, αναφέρεται πως στην παρούσα ενότητα ενδιαφέρον έχει και η αξιολόγηση της επίδοσης των μοντέλων TransFetch και TF Classic, τα οποία έχουν εκπαιδευτεί με τον ίδιο τρόπο, σε διαφορετικό όμως πλήθος δεδομένων. Και οι δύο εκδοχές του δικτύου έχουν εκπαιδευτεί με μέγιστο αριθμό εποχών ίσο με 200, καθώς όμως η διαδικασία εκπαίδευσης σταματά εάν δεν υπάρξει βελτίωση της ακρίβειας για ορισμένο αριθμό εποχών, το πλήθος εποχών εκπαίδευσης ανά πρόγραμμα πιθανώς διαφέρει μεταξύ των δύο δικτύων.

5.2.1 Ακρίβεια

Ξεκινώντας από την ακρίβεια των μοντέλων για το σύνολο των εφαρμογών, η εικόνα είναι παρόμοια με αυτή που είχαμε παραπάνω, και φαίνεται στα σχήματα 5.7. Τα δίκτυα που χρησιμοποιήθηκαν παρουσιάζουν στη γενική περίπτωση ακρίβεια συγκρίσιμη και καλύτερη εν σχέσει με τους ¨κλασσικούς' προανακλητές, με εξαίρεση τον ASP, ο οποίος έχει την υψηλότερη ακρίβεια πέραν του προανακλητή Oracle, για τους λόγους που αναφέρθηκαν και παραπάνω. Συγκρίνοντας τα δίκτυα μεταξύ τους, βλέπουμε αρχικά πως η εκπαίδευση μέσω δειγματοληψίας φαίνεται να λειτουργεί ευεργετικά για το δίκτυο, καθώς η ακρίβειά των δικτύων TF Sample και TF Window ξεπερνά ακόμη και αυτή του αρχικού δικτύου (TransFetch 9), το οποίο έχει εκπαιδευτεί σε διπλάσιο όγκο δεδομένων. Η ακρίβεια του δικτύου TF Classic φαίνεται να είναι μειωμένη σε σχέση με αυτή του αρχικού μοντέλου, γεγονός αναμενόμενο, εφόσον έχει εκπαιδευτεί στο ιστορικό προσβάσεων που προκύπτει από τις μισές σε πλήθος εντολές εν σχέσει με το αρχικό δίκτυο. Εξετάζοντας τα αποτελέσματα για κάθε σουίτα ξεχωριστά, βλέπουμε πως δεν υπάρχουν σημαντικές μεταβολές στη συγκριτική απόδοση των προανακλητών ανά σουίτα, με τα αποτελέσματα να είναι σε κάθε περίπτωση όμοια μεταξύ τους.

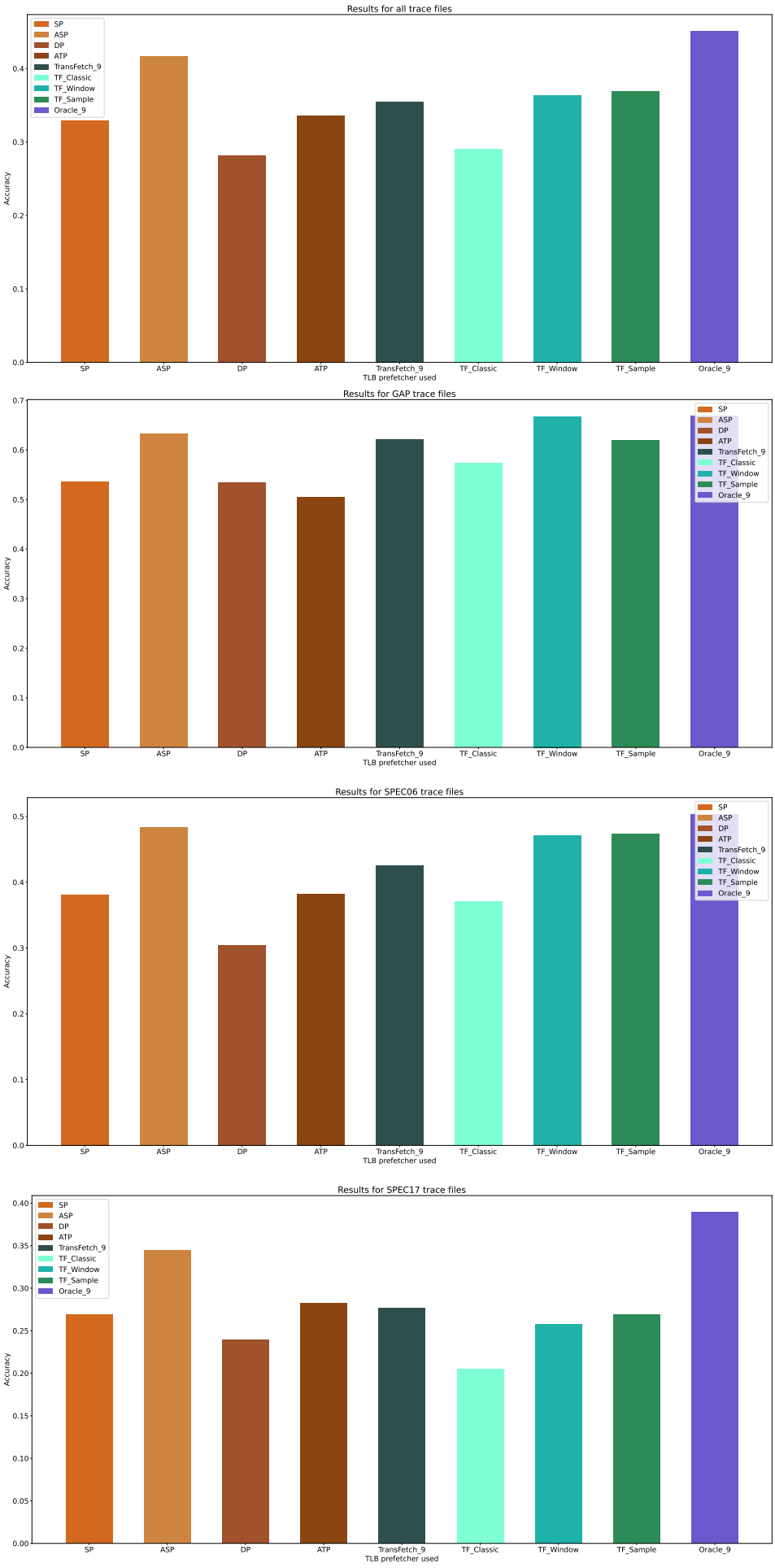
Περνώντας στην περίπτωση των TLB-intensive εφαρμογών, στα σχήματα 5.8 βλέπουμε την ακρίβεια όλων των προανακλητών να αυξάνεται εάν ληφθεί υπόψιν το σύνολο των εφαρμογών. Συγκεκριμένα για τα δίκτυα που χρησιμοποιήθηκαν, η μεγαλύτερη βελτίωση φαίνεται να προέρχεται από το δίκτυο TF Classic. Συγκεκριμένα, ενδιαφέρον είναι το γεγονός πως για τα TLB-intensive ίχνη της σουίτας SPEC06, ο προανακλητής TF Classic παρουσιάζει μεγαλύτερη ακρίβεια σε σχέση με το δίκτυο TransFetch 9, το οποίο έχει εκπαιδευτεί σε διπλάσιο όγκο δεδομένων. Σημειώνεται ταυτόχρονα ότι ο όγκος των δεδομένων εισόδου που τροφοδοτούνται στα δίκτυα, θα είναι σαφώς μεγαλύτερος για τις TLB-intensive εφαρμογές, καθώς σε αυτές το ίδιο πλήθος εντολών θα περιλαμβάνει μεγαλύτερο αριθμό αιτημάτων μετάφρασης. Το γεγονός αυτό, πιθανώς οφείλεται στην υπερπροσαρμογή του δεύτερου δικτύου στα δεδομένα εισόδου, και οδηγεί στο δίκτυο που έχει εκπαιδευτεί σε λιγότερες προσβάσεις να έχει μεγαλύτερη ακρίβεια. Ομοίως με παραπάνω, τα δίκτυα TF Sample και TF Window παρουσιάζουν τη μέγιστη τιμή ακρίβειας για όλους τους προανακλητές, πλην των ASP και Oracle. Τέλος, αναφέρεται πως παρόλο που η ακρίβεια για το σύνολο των εφαρμογών φαίνεται να αυξάνεται περνώντας στις TLB-intensive, η αύξηση αυτή οφείλεται κατά κύριο λόγο στο γεγονός πως οι εφαρμογές της σουίτας GAP, στις οποίες όλοι οι προανακλητές παρουσιάζουν σημαντικά υψηλότερη ακρίβεια, παραμένουν σταθερές σε πλήθος, ενώ οι υπόλοιπες μειώνονται.

5.2.2 Κάλυψη

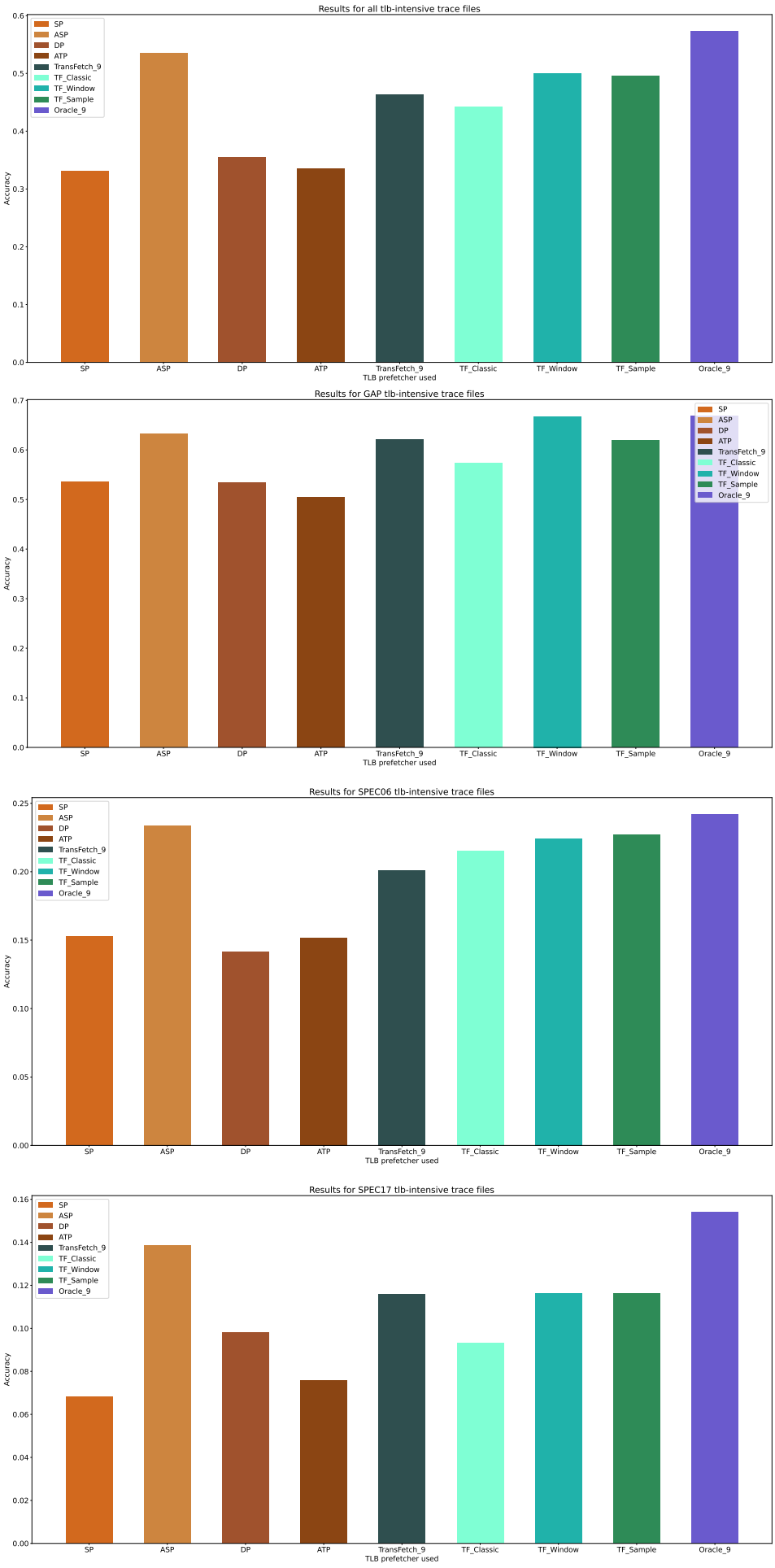
Στην αξιολόγηση της κάλυψης των προανακλητών, τα αποτελέσματα της οποία φαίνονται στα σχήματα 5.9 τα δίκτυα μας φαίνεται να υστερούν σε σχέση με τους υπόλοιπους προανακλητές. Λαμβάνοντας υπόψιν τα συνολικά αποτελέσματα, καθώς και τα αποτελέσματα για κάθε σουίτα ξεχωριστά, φαίνεται πως όλοι οι προανακλητές πέραν του ASP παρουσιάζουν καλύτερη ή συγκρίσιμη τιμή κάλυψης σε σχέση με τα δίκτυα που χρησιμοποιήθηκαν. Συγκρίνοντας τα δίκτυα μεταξύ τους, βλέπουμε πως η επίδοση τους δεν διαφέρει σημαντικά. Παρόλα αυτά ενδιαφέρον έχει και πάλι η επίδοση του δικτύου TF Sample, το οποίο εκπαιδεύτηκε μέσω δειγματοληψίας και ξεπερνά και πάλι σε επίδοση, αν και όχι σημαντικά, το δίκτυο TransFetch. Λαμβάνοντας υπόψιν τις TLB-intensive, στο σχήμα 5.10 βλέπουμε πως η εικόνα για τα δίκτυα είναι εμφανώς βελτιωμένη. Στη γενική περίπτωση, τα δίκτυά μας φαίνεται να υστερούν σε ποσοστά κάλυψης συγκριτικά με τους προανακλητές ATP και SP, ενώ συγκριτικά μεταξύ τους φαίνεται να έχουν την ίδια απόδοση. Εξαίρεση αποτελούν οι εφαρμογές της σουίτας SPEC17, στις οποίες τα δίκτυα παρουσιάζουν την καλύτερη τιμή κάλυψης συγκριτικά με τους κλασσικούς προανακλητές.

5.2.3 Αύξηση Επίδοσης

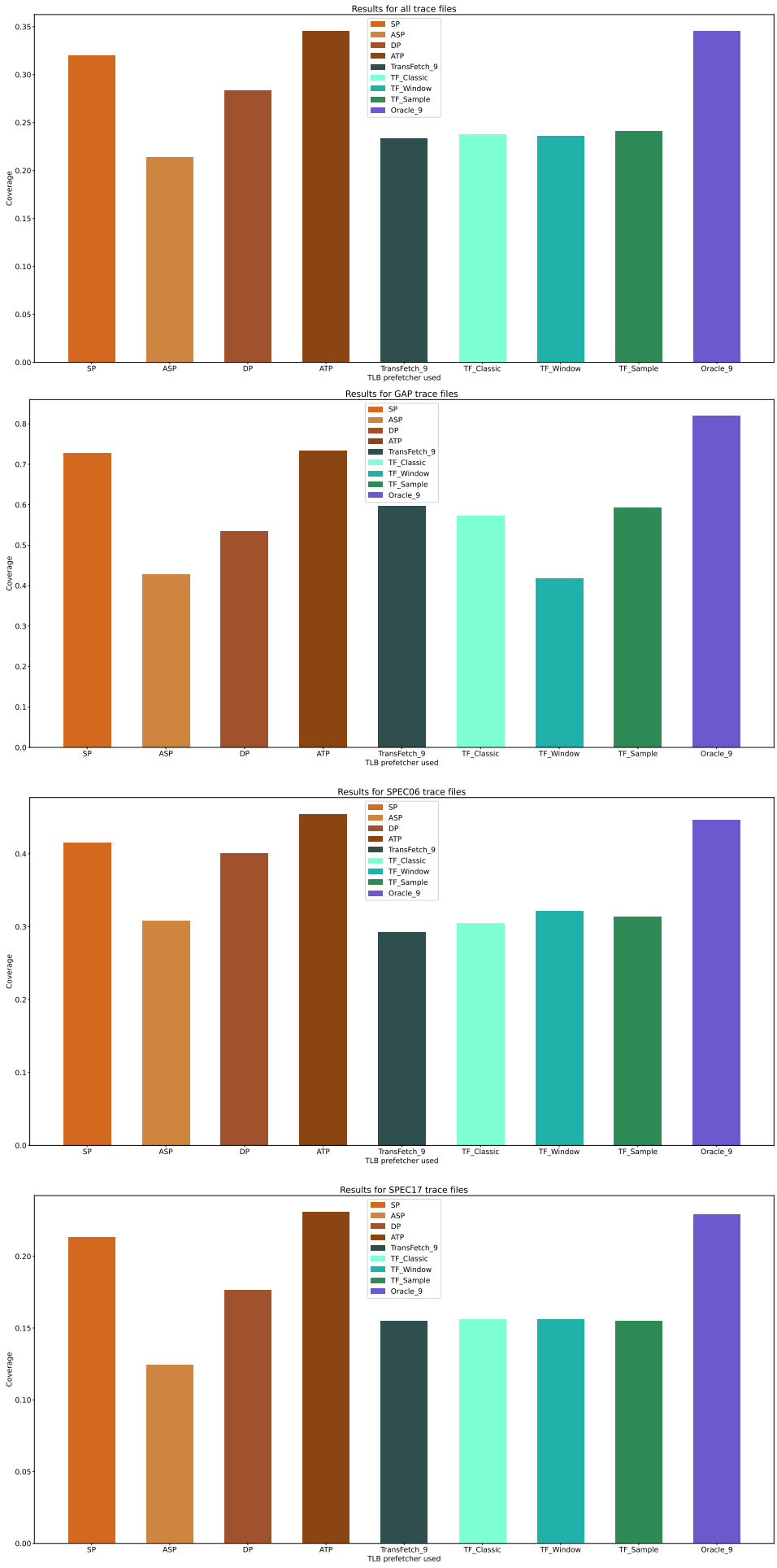
Η σύγκριση των ποσοστών αύξησης επίδοσης για όλους τους προανακλητές φαίνεται στα σχήματα 5.11. Παρατηρώντας τις διαφορές μεταξύ των εκδοχών του δικτύου, παρατηρούμε αρχικά πως η μείωση του πλήθους των δεδομένων εκπαίδευσης δεν επιφέρει σημαντική πτώση της συνολικής απόδοσης. Για το γεγονός αυτό, πιθανώς να οφείλεται η υπερπροσαρμογή του αρχικού δικτύου όταν το ιστορικό προσβάσεων κάποιου ίχνους είναι πολύ πυκνό. Παρόλα αυτά, η εκπαίδευση μέσω δειγματοληψίας φαίνεται να παρουσιάζει καλύτερη επίδοση και σε αυτή τη μετρική, καθιστώντας την, την αποδοτικότερη μέθοδο εκπαίδευσης βάση επίδοσης. Παρόλα αυτά, σημειώνεται πως λόγω της διαδικασίας που ακολουθήθηκε για την εκπαίδευση των δικτύων μέσω δειγματοληψίας, οι απαιτήσεις μνήμης για τον τρόπο εκπαίδευσης του δικτύου TF Sample είναι μεγαλύτερες για την εκπαίδευση σε ίδιο αριθμό εντολών, καθώς πρέπει να διαβαστούν αρχικά περισσότερα δεδομένα από αυτά που θα χρησιμοποιηθούν. Παρόλα αυτά, φαίνεται πως το δίκτυο είναι ικανό να αξιοποιήσει το ιστορικό προσβάσεων με συγκριτικά καλύτερο τρόπο, γεγονός που σημαίνει πως δεν απαιτείται η τροφοδότηση μεγάλου όγκου δεδομένων για την ικανοποιητική εκπαίδευση του δικτύου. Το γεγονός αυτό υποστηρίζει και η απόδοση του μοντέλου TF Window, το οποίο παρουσιάζει ίδια, και ελαφρώς καλύτερη απόδοση, δίχως τον παραπάνω περιορισμό, αφού το σύνολο δεδομένων εκπαίδευσης αντλείται απευθείας από τις προσβάσεις 20 εκατομμύρια εντολών. Η αξιολόγηση της επίδοσης των προανακλητών στις TLB-intensive εφαρμογές, που φαίνεται στα σχήματα 5.12, δεν μας βοηθά να εξάγουμε περισσότερα συμπεράσματα, καθώς και πάλι η επίδοση των προανακλητών είναι βελτιωμένη, δίχως όμως να υπάρχουν διακυμάνσεις στην επίδοση των προανακλητών συγκριτικά μεταξύ τους.



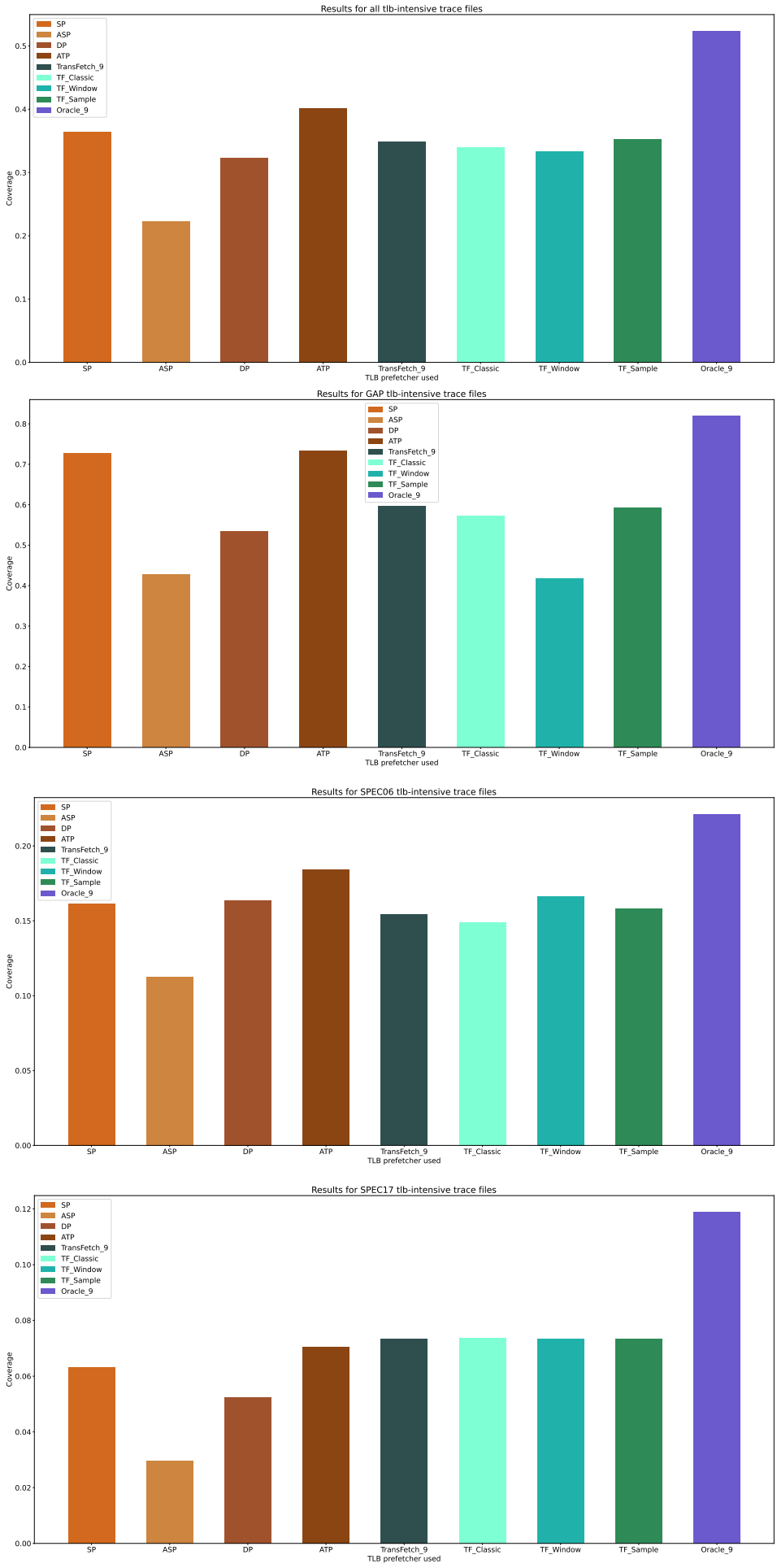
Σχήμα 5.7: Τα αποτελέσματα ακρίβειας για κάθε σουίτα - Αξιολόγηση μεθόδων εκπαίδευσης



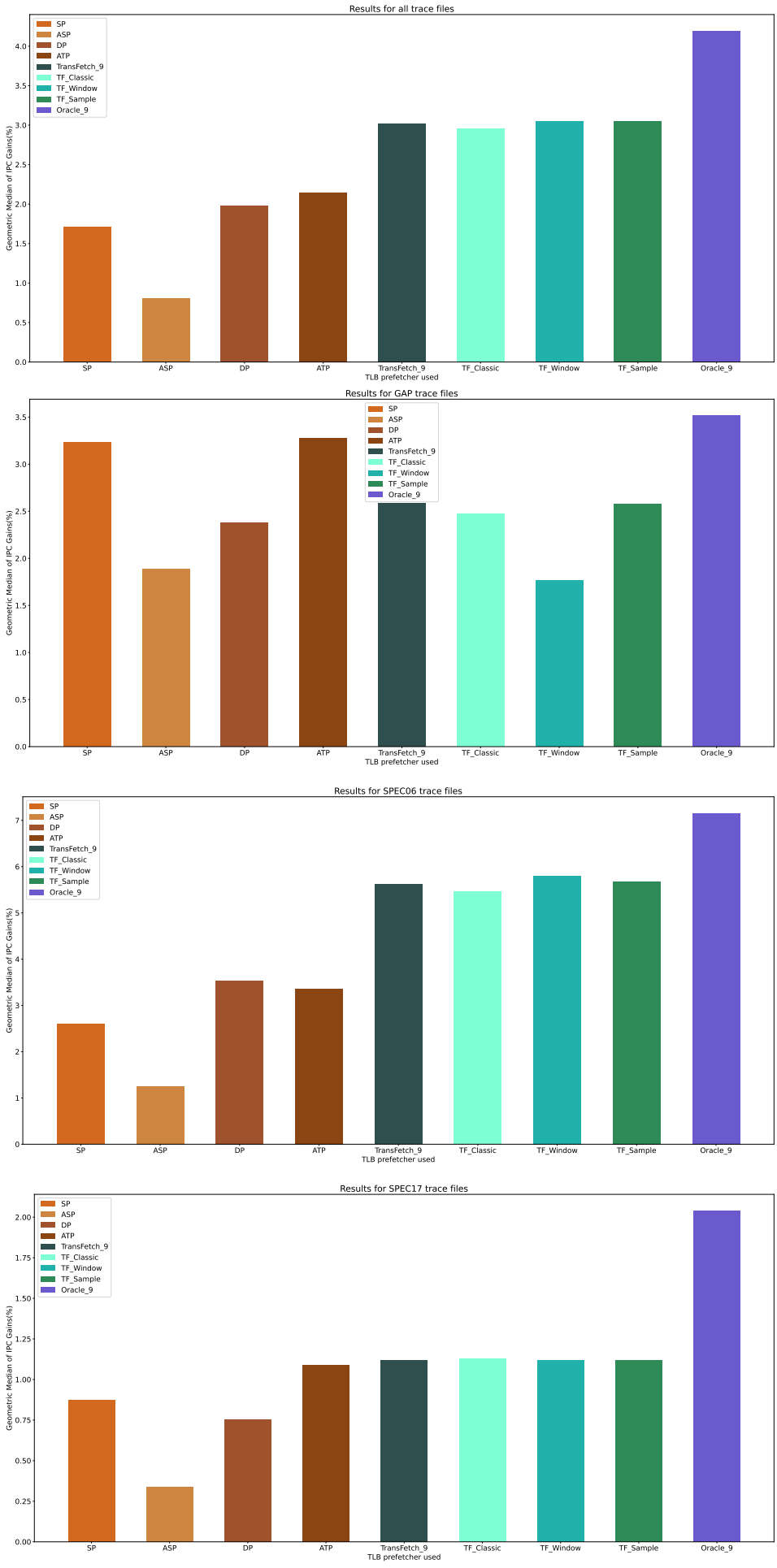
Σχήμα 5.8: Τα αποτελέσματα ακρίβειας για τις TLB-intensive εφαρμογές - Αξιολόγηση μεθόδων εκπαίδευσης



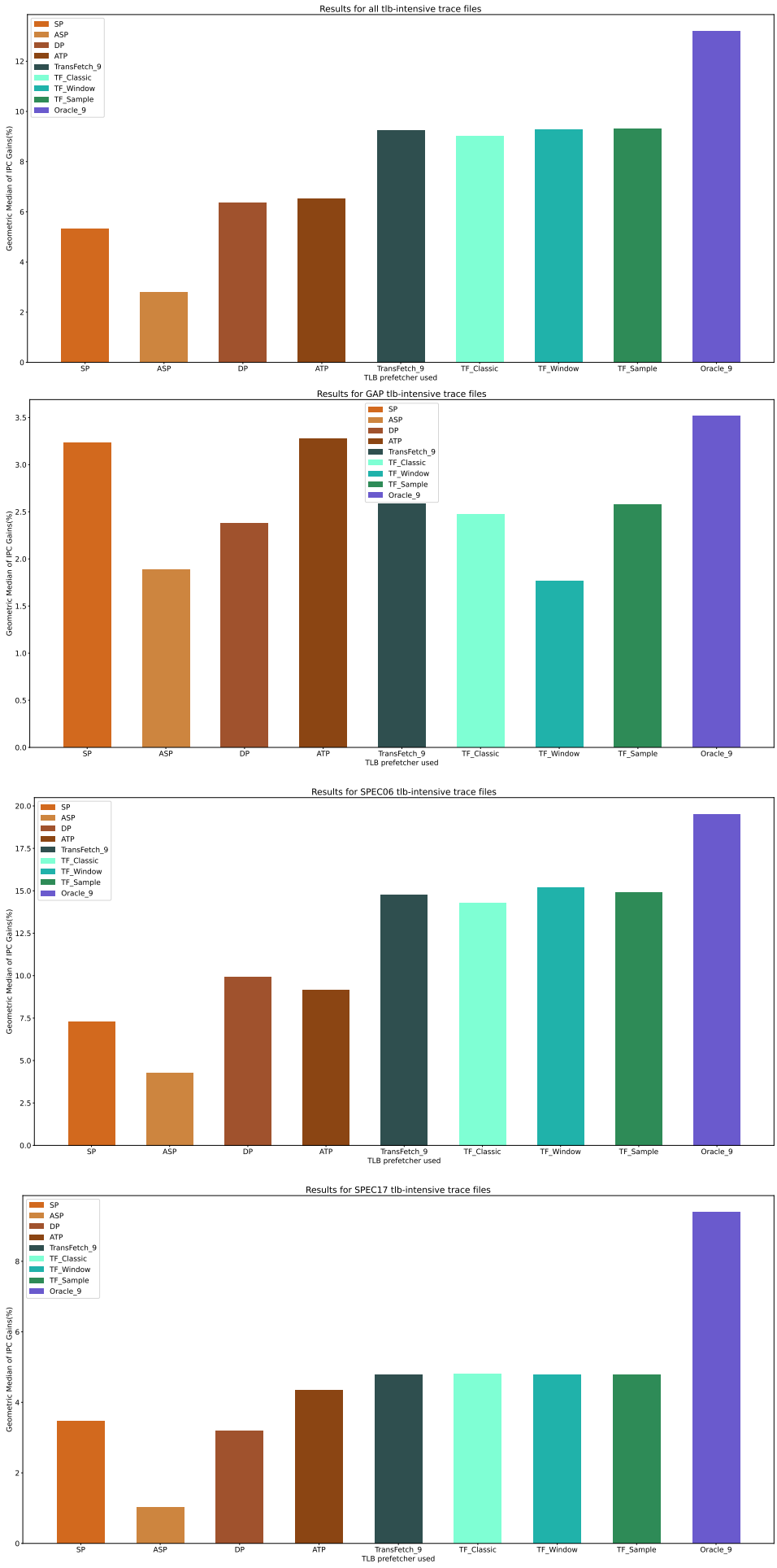
Σχήμα 5.9: Τα αποτελέσματα κάλυψης για κάθε σουίτα - Αξιολόγηση μεθόδων εκπαίδευσης



Σχήμα 5.10: Τα αποτελέσματα κάλυψης για τις TLB-intensive εφαρμογές - Αξιολόγηση μεθόδων εκπαίδευσης



Σχήμα 5.11: Η αύξηση της επίδοσης για κάθε σουίτα - Αξιολόγηση μεθόδων εκπαίδευσης



Σχήμα 5.12: Η αύξηση της επίδοσης για τις TLB-intensive εφαρμογές - Αξιολόγηση μεθόδων εκπαίδευσης

5.3 Αξιολόγηση Εκπαίδευσης δικτύων σε σύνολα εφαρμογών

Στην ενότητα αυτή αξιολογείται η δυνατότητα εκπαίδευση του νευρωνικού δικτύου ώστε να λειτουργήσει αποδοτικά σε σύνολα εφαρμογών. Παραπάνω, είδαμε πως τα νευρωνικά δίκτυα είναι σε θέση να ξεπεράσουν σε απόδοση και ακρίβεια τους ¨κλασσικούς' προανακλητές στην STLB. Παρόλα αυτά, τα δίκτυα που έχουν χρησιμοποιηθεί έως τώρα αφορούν το κάθε ένα από ένα ίχνος. Είναι εμφανές, πως σε πραγματικές συνθήκες η χρήση ξεχωριστών δικτύων για κάθε εφαρμογή ενός συστήματος δεν είναι αποδοτική, ακόμη και εάν αυτά είναι ήδη εκπαιδευμένα, λόγω των πόρων μνήμης που θα ήταν απαραίτητοι. Για τον λόγο αυτό, επιχειρούμε να εκπαιδεύσουμε το δίκτυο σε σύνολα εφαρμογών που παρουσιάζουν ομοιότητες και να αξιολογήσουμε την απόδοσή τους. Η χρήση του δικτύου στην υπάρχουσα μορφή του οδηγεί σε χαμηλό αριθμό προανακλήσεων, οι οποίες μάλιστα δεν είναι ακριβείς Για τους λόγους αυτούς, αρχικά προστέθηκαν στο μοντέλο πιο επιθετικά στρώματα απόσυρσης, με την ελπίδα να αποφευχθεί πλήρως η υπερπροσαρμογή του δικτύου στα δεδομένα εισόδου και το μοντέλο να γενικεύει καλύτερα. Παράλληλα, η απαραίτητη τιμή αυτοπεποίθησης προκείμενου το μοντέλο να εκτελέσει πρόβλεψη μειώθηκε αισθητά, ώστε να αυξηθεί το πλήθος των προανακτήσεων που εκτελούνται.

Τα traces ομαδοποιήθηκαν ανά μετροπρόγραμμα, με κάθε μοντέλο να χρησιμοποιείται για ένα μετροπρόγραμμα. Για την εκπαίδευση, χρησιμοποιήθηκαν δύο διαφορετικές τεχνικές. Στην πρώτη, το μοντέλο ακολουθεί τη διαδικασία εκπαίδευσης και επαλήθευσης για 10 και 3 εκατομμύρια εντολές αντίστοιχα, για όλα τα ίχνη του μετροπρογράμματος, εκπαιδεύεται δηλαδή συνολικά στις προσβάσεις που αντιστοιχούν από 40 έως 80 εκατομμύρια εντολές. Στη συνέχεια, παράγει αρχείο προανακλήσεων ξεχωριστά για κάθε ένα από τα ίχνη. Στη δεύτερη, το μοντέλο ακολουθεί τη διαδικασία εκπαίδευσης για 20 εκατομμύρια εντολές για όλα τα ίχνη του μετροπρογράμματος. Αφού εκπαιδευτεί για όλα τα ίχνη, ακολουθεί επαλήθευση σε 10 εκατομμύρια εντολές και παραγωγή αρχείου προανακλήσεων, για κάθε ίχνος ξεχωριστά. Οι μέθοδοι αυτές σημειώνονται στα διαγράμματα ως TF Multi no Val και TF Multi αντίστοιχα. Η διαφοροποίηση μεταξύ των δύο έγκειται στο γεγονός πως στην πρώτη περίπτωση το δίκτυο καλείται να παραγάγει το αρχείο προανακλήσεων με βάση την πληροφορία που έχει δεχθεί από όλα τα προγράμματα, ενώ στη δεύτερη, όπου η διαδικασία επαλήθευσης για κάθε ίχνος εκτελείται ακριβώς πριν την παραγωγή του αρχείου προανακλήσεων, το δίκτυο έχει την ευκαιρία να προσαρμόσει ελαφρά τα βάρη του, ώστε να αυξηθεί η ακρίβεια των προανακλήσεων. Τα δεδομένα εισόδου αντλούνται από τις 20 εκατομμύρια πρώτες εντολές, δίχως δειγματοληψία, και έχει προστεθεί τιμή επικαιρότητας ίση με 9. Σημειώνεται πως ορισμένες εφαρμογές παραλείφθηκαν από την παρούσα ανάλυση, καθώς λόγω των υψηλών απαιτήσεων μνήμης και των εκτενών ιστορικών προσβάσεων που τους αντιστοιχούσαν δεν ήταν δυνατή η παραγωγή αρχείων προανακλήσεων σε αυτά. Λόγω αυτού, τα διαγράμματα των TLB-intensive εφαρμογών της σουίτας GAP στην παρούσα ανάλυση ταυτίζονται με αυτά που αφορούν όλα τα προγράμματα της σουίτας.

5.3.1 Ακρίβεια

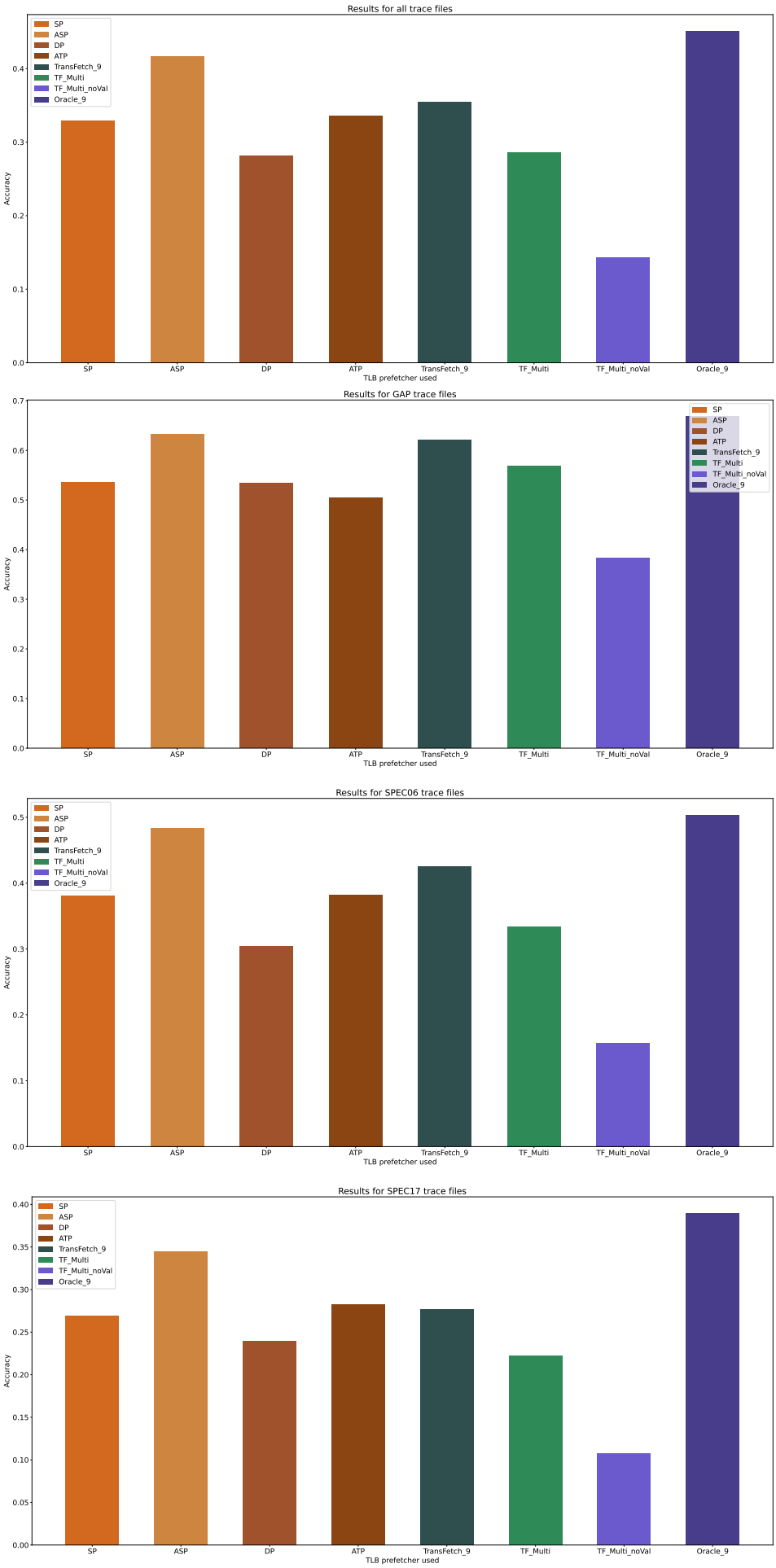
Η σύγκριση των προανακλητών με βάση την ακρίβειά τους στο σύνολο των εφαρμογών φαίνεται στα σχήματα 5.13. Σε αυτά, βλέπουμε πως στη γενική περίπτωση η ακρίβεια των δικτύων που έχουν εκπαιδευθεί σε πολλαπλά ίχνη είναι χαμηλότερη συγκριτικά και με τους κλασσικούς προανακλητές αλλά και με την αρχική έκδοση του δικτύου. Τα αποτελέσματα είναι ακόμη χειρότερα, στην περίπτωση που το δίκτυο δεν έχει την ευκαιρία να προσαρμοστεί στο ίχνος για το οποίο καλείται να εκτελέσει προανάκληση. Το γεγονός αυτό δικαιολογείται, καθώς τα ίχνη στα οποία εκπαιδεύτηκε κάθε μοντέλο δεν έχουν απαραίτητα όμοια μοτίβα πρόσβασης μεταξύ τους, αλλά αποτελούν το κάθε ένα μέρος του ίδιου μετροπρογράμματος. Αξιολογώντας την ακρίβεια προανακλήσεων σε TLB-intensive εφαρμογές, η οποία φαίνεται στα σχήματα 5.14, βλέπουμε πως η επίδοση των μοντέλων αυξάνεται συγκριτικά με τους υπόλοιπους προανακλητές, με τον προανακλητή TF Multi να ξεπερνά σε ακρίβεια όλους τους κλασσικούς προανακλητές πέραν του προανακλητή ASP. Ενδιαφέρον παρουσιάζει και η περίπτωση της σουίτας SPEC17, όπου η ακρίβεια του προανακλητή TF Multi ξεπερνά την ακρίβεια του αρχικού μας μοντέλου, πιθανώς επειδή τα προγράμματα της συγκεκριμένης σουίτας ακολουθούν παρόμοια μοτίβα προσβάσεων καθ όλη τη διάρκεια της εκτέλεσής τους. Σε κάθε περίπτωση, η απόδοση του μοντέλου εάν δεν έχει ευκαιρία να προσαρμοστεί στο συγκεκριμένο ίχνος για το οποίο καλείται να εκτελέσει προανάκληση παραμένει απογοητευτική και χειρότερη από όλα τα υπόλοιπα σχήματα προανάκλησης που χρησιμοποιούνται.

5.3.2 Κάλυψη

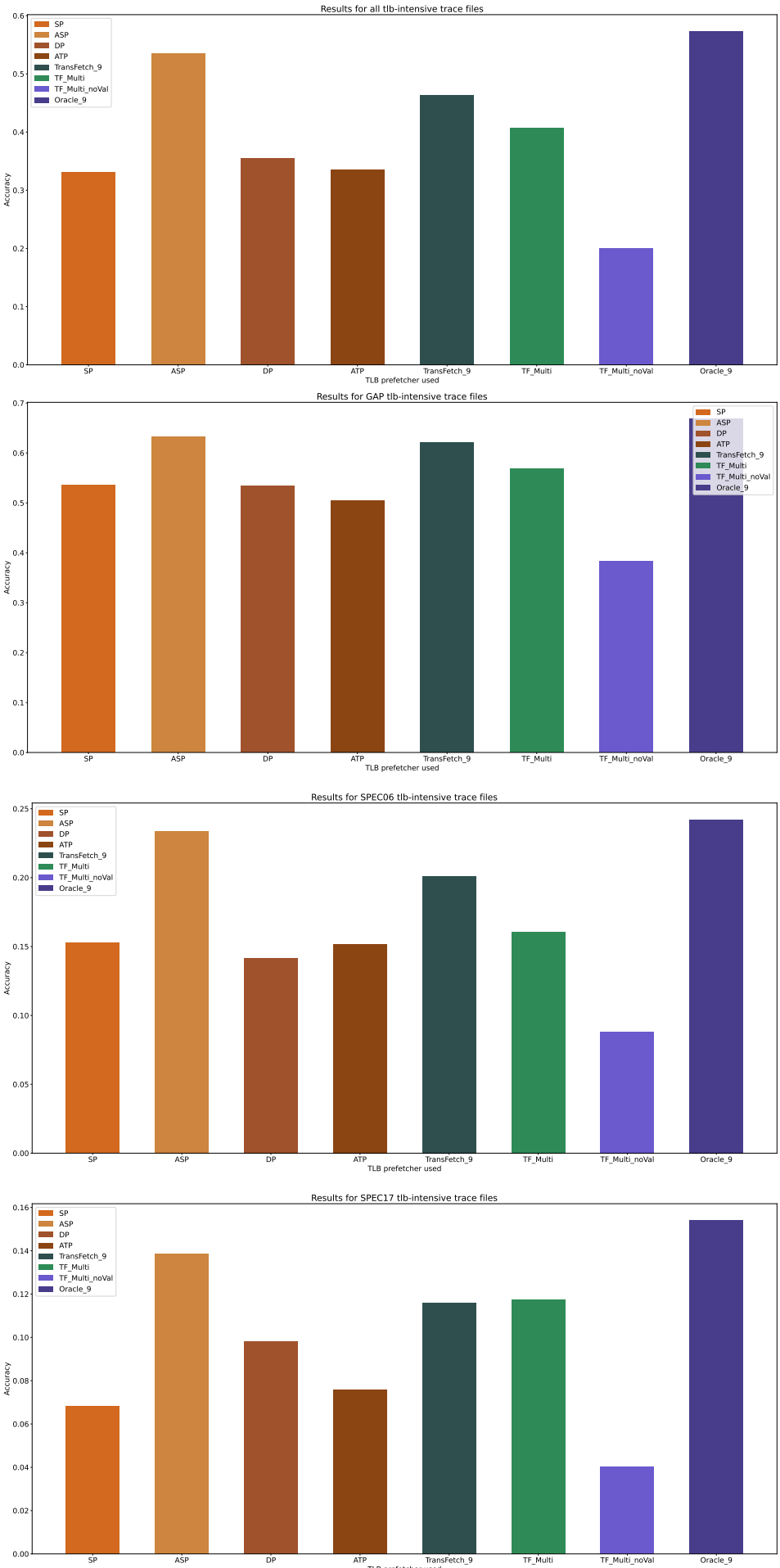
Τα διαγράμματα που αφορούν την κάλυψη των προανακλητών για όλες τις εφαρμογές και τις εφαρμογές κάθε σουίτας ξεχωριστά βρίσκονται στα σχήματα 5.15. Σε αυτά, βλέπουμε πως η έκδοση TF Multi του δικτύου έχει παρόμοια και ελαφρώς υψηλότερα ποσοστά κάλυψης των αστοχιών στην STLB σε όλες τις περιπτώσεις. Το γεγονός αυτό είναι πολύ πιθανό να οφείλεται στη μείωση της τιμής της απαραίτητης αυτοπεποίθησης προκειμένου να εκτελεστεί πρόβλεψη, για το δίκτυο TF Multi. Παρόλα αυτά, φαίνεται πως και οι δύο εκδόσεις του δικτύου υστερούν, σε σχέση με τους κλασσικούς προανακλητές. Απογοητευτική είναι και πάλι η επίδοση του δικτύου TF Multi noVal, η οποία παρουσιάζει με μεγάλη διαφορά τα μικρότερα ποσοστά κάλυψης από όλους τους προανακλητές. Η επίδοση αυτή μάλιστα δεν διαφοροποιείται για την περίπτωση των TLB-intensive εφαρμογών, τα ποσοστά κάλυψης για τα οποία φαίνονται στα σχήματα 5.16. Τα υπόλοιπα δίκτυα φαίνεται να εμφανίζουν ελαφρώς καλύτερη απόδοση συγκριτικά με τους υπόλοιπους προανακλητές σε σχέση με προηγουμένως. Η επίδοση του δικτύου TF Multi φαίνεται και πάλι να είναι αυξημένη για τα προγράμματα της σουίτας SPEC17. Η επίδοση αυτή, πιθανώς σημαίνει πως οι αλληλουχίες πρόσβασης των μετροπρογραμμάτων της σουίτας ευνοούν τη χρήση μοναδικού δικτύου για κάθε σύνολο ιχνών.

5.3.3 Αύξηση Επίδοσης

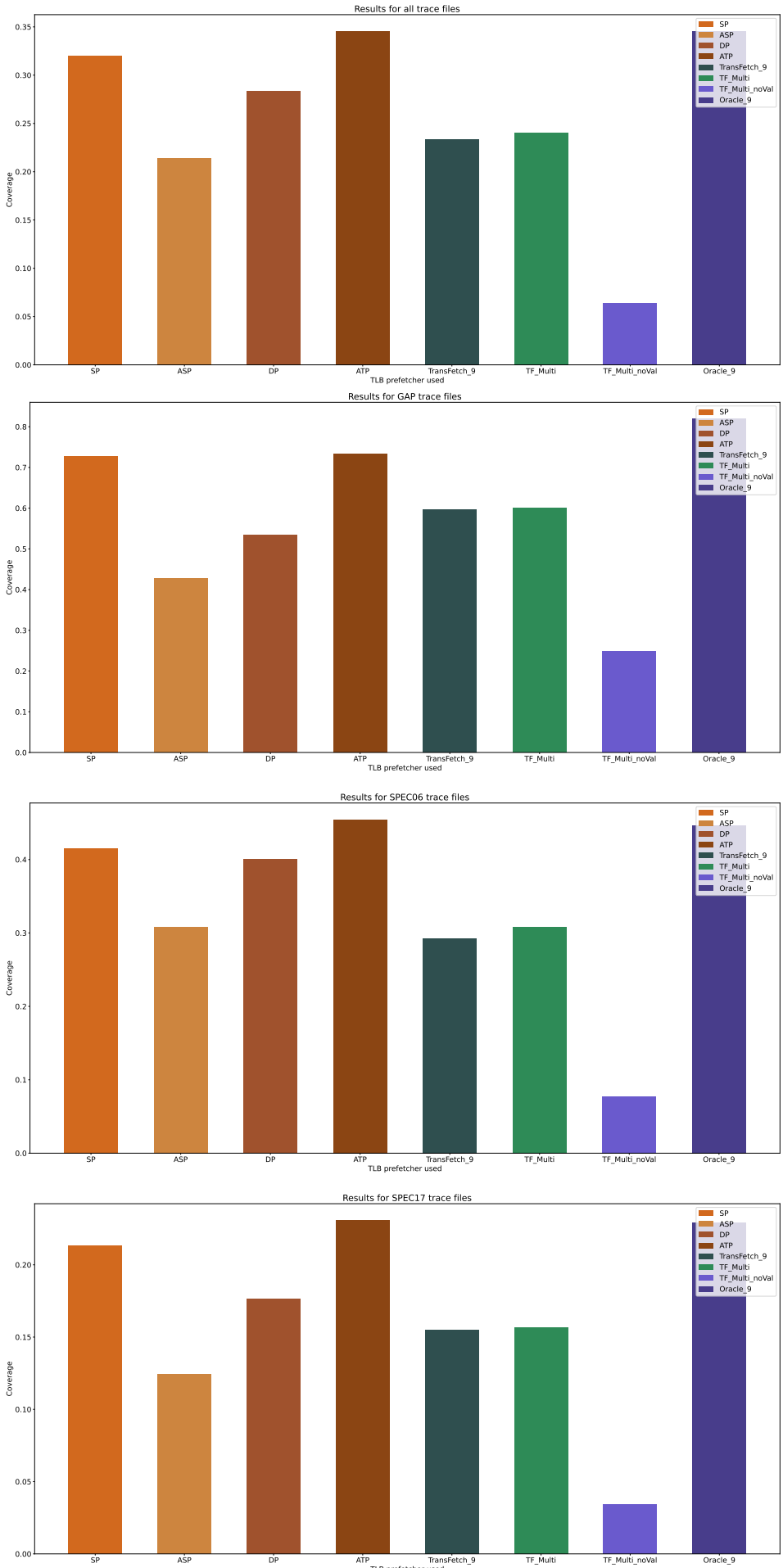
Τέλος, παρατίθενται στα σχήματα 5.17, 5.18 τα αποτελέσματα αύξησης του IPC για όλα τα προγράμματα και για τις TLB-intensive εφαρμογές αντίστοιχα. Όπως και παραπάνω, η συνολική εικόνα των διαγραμμάτων δεν διαφέρει αισθητά, πέραν των υψηλότερων ποσοστών αύξησης της επίδοσης για τα TLB-intensive ίχνη. Ξεκινώντας από το δίκτυο TF Multi noVal, βλέπουμε πως η επίδοση του είναι απογοητευτική για κάθε σουίτα προγραμμάτων, καθώς παρέχει ελάχιστες αυξήσεις στην επίδοση των εφαρμογών. Το γεγονός αυτό δεν προκαλεί εντύπωση, καθώς όπως είδαμε παραπάνω τόσο το ποσοστό ακρίβειας, όσο και το ποσοστό κάλυψής του είναι πολύ χαμηλό για το σύνολο των εφαρμογών. Περνώντας όμως στο δίκτυο TF Multi, βλέπουμε πως η επίδοση του είναι συγκρίσιμη, και ελαφρώς υψηλότερη από την αρχική έκδοση του δικτύου που χρησιμοποιήθηκε, ξεπερνώντας ταυτόχρονα στη γενική περίπτωση όλους τους κλασσικούς προανακλητές. Ο λόγος για αυτό, πιθανώς έγκειται στο γεγονός πως τα δίκτυα που δημιουργήθηκαν για την εκτέλεση προανακλήσεων σε σύνολα εφαρμογών εκπαιδεύτηκαν σε 20 εκατομμύρια εντολές ανά ίχνος, δηλαδή συνολικά από 80 έως 160 εκατομμύρια εντολές αναλόγως με το ποσό των ιχνών που αντιστοιχεί σε κάθε ομάδα προγραμμάτων.



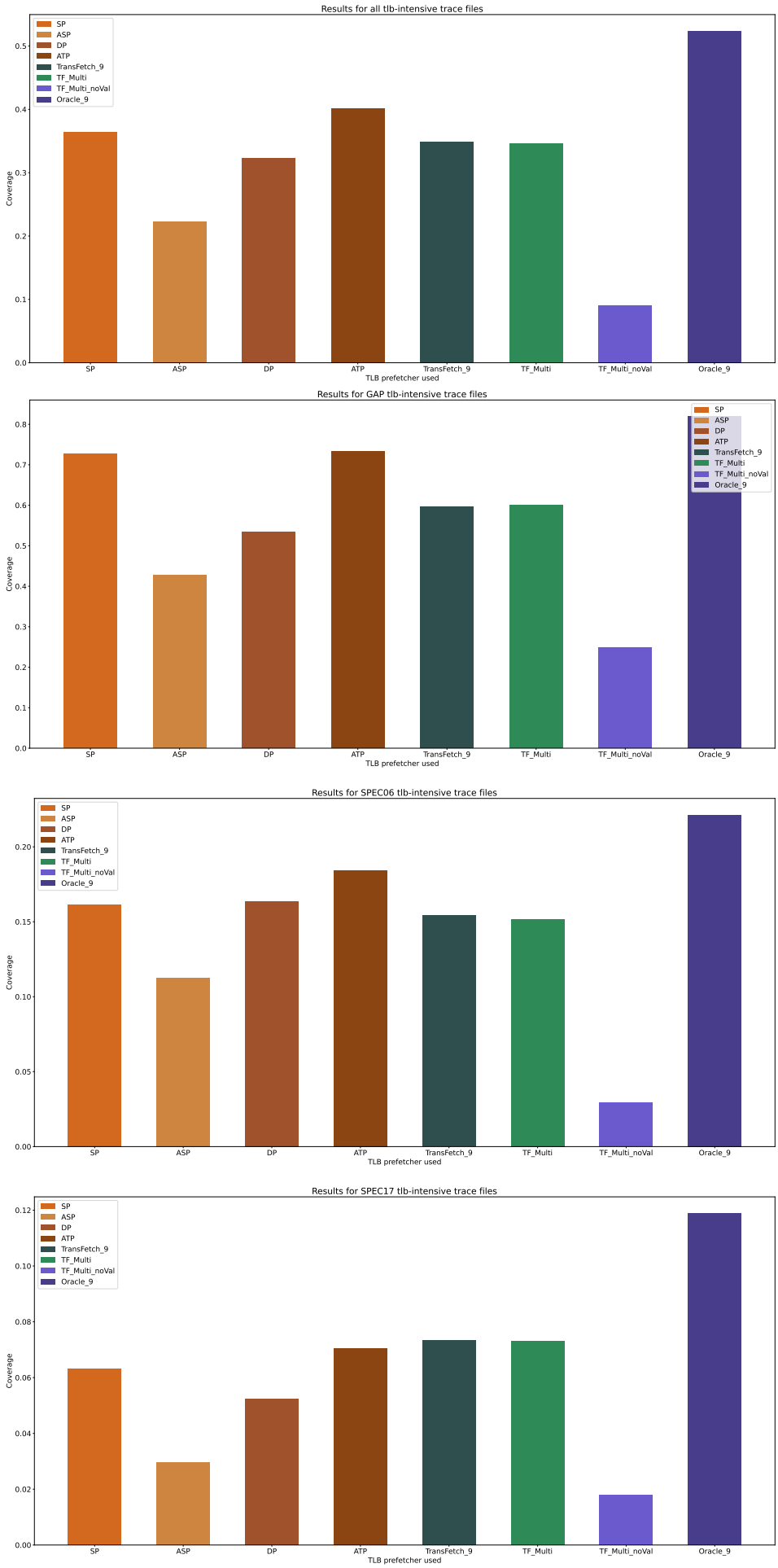
Σχήμα 5.13: Τα αποτελέσματα ακρίβειας για κάθε σουίτα - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών



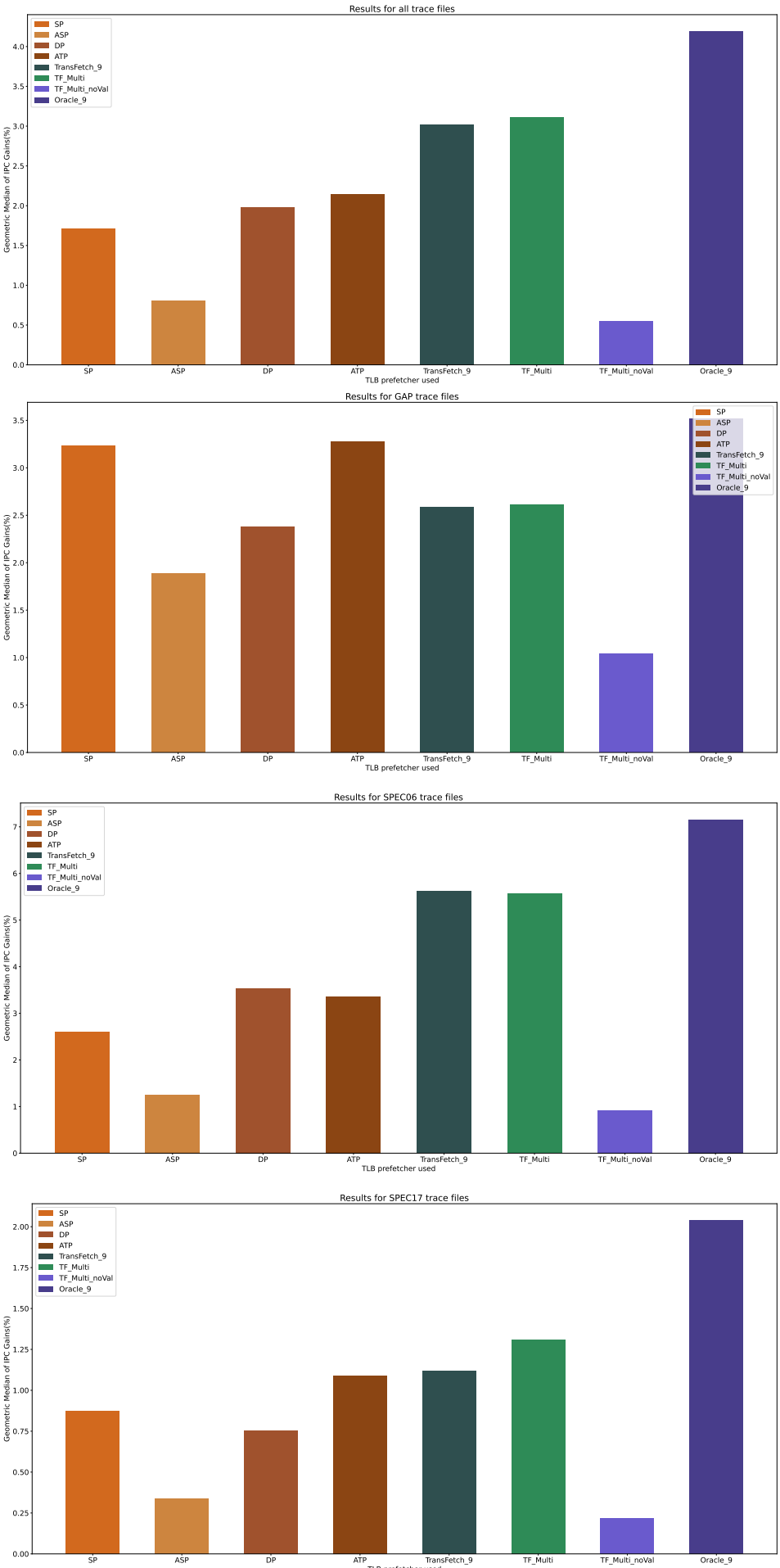
Σχήμα 5.14: Τα αποτελέσματα ακρίβειας για τις TLB-intensive εφαρμογές - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών



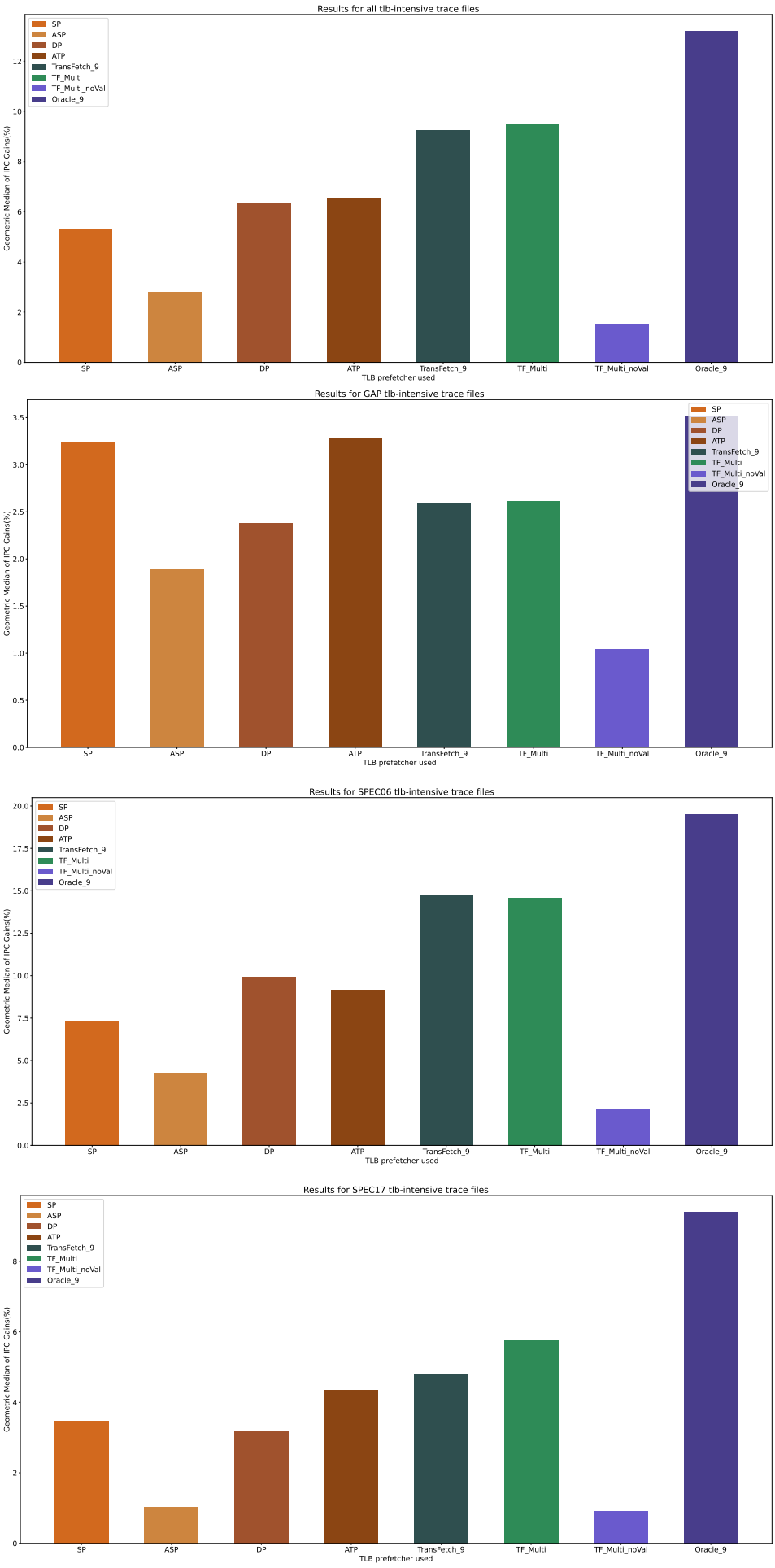
Σχήμα 5.15: Τα αποτελέσματα κάλυψης για κάθε σουίτα - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών



Σχήμα 5.16: Τα αποτελέσματα κάλυψης για τις TLB-intensive εφαρμογές - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών



Σχήμα 5.17: Η αύξηση της επίδοσης για κάθε σουίτα - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών



Σχήμα 5.18: Η αύξηση της επίδοσης για τις TLB-intensive εφαρμογές - Αξιολόγηση εκπαίδευσης σε σύνολα εφαρμογών

Κεφάλαιο 6

Συμπεράσματα & Μελλοντικές επεκτάσεις

Η Κρυφή Μνήμη Αναζήτησης Μετάφρασης αποτελεί ένα σημαντικό τμήμα των σύγχρονων υπολογιστικών συστημάτων, με την επίδοσή της να είναι άρρηκτα συνδεδεμένη με την επίδοση ολόκληρου του συστήματος. Στην προσπάθεια μεγιστοποίησης της χρησιμότητας της TLB, ένας από τους σημαντικότερους μηχανισμούς είναι αυτός της προανάκλησης. Παρά την τεράστια πρόοδο στον τομέα των προανακλητών, δεν έχει υπάρξει ακόμη κάποιος καθολικά αποδοτικός προανακλητής, με τις επιδόσεις ορισμένων μετροπρογραμμάτων να μαρτυρούν υψηλά περιθώρια βελτίωσης. Η χρήση τεχνητών νευρωνικών δικτύων αποτελεί μια ελκυστική και πολλά υποσχόμενη προσέγγιση στο συγκεκριμένο ζήτημα. Η παρούσα μελέτη αποτέλεσε μια προσπάθεια τέτοιου είδους προσέγγισης. Στο κεφάλαιο αυτό θα παρουσιαστούν τα συμπεράσματα που προκύπτουν με βάση την παραπάνω ανάλυση, καθώς και ορισμένες προτάσεις μελλοντικής εξέλιξης του μηχανισμού που προκύπτουν βάση αυτής.

6.1 Συμπεράσματα

Με βάση τα αποτελέσματα του παραπάνω κεφαλαίου, φαίνεται πως η χρήση νευρωνικών δικτύων για την προανάκληση μπορεί να φανεί ιδιαίτερα αποδοτική, παρέχοντας υψηλή ακρίβεια και αύξηση της επίδοσης, με το μοντέλο που χρησιμοποιήθηκε να ξεπερνά κατά πολλούς τρόπους τους κλασσικούς μηχανισμούς προανάκλησης. Παράλληλα, η πρόβλεψη των μελλοντικών προσβάσεων με βάση το ιστορικό αιτημάτων πρόσβασης φαίνεται πως μπορεί να αποτελέσει μια καθολική μέθοδο προανάκλησης, καθώς τα νευρωνικά δίκτυα είναι σε θέση να ανταποκριθούν ικανοποιητικά σε πληθώρα διαφορετικών προγραμμάτων, εφόσον εκπαιδευτούν στο ιστορικό προσβάσεων τους. Η χρήση ενός Χωρικού Συνελικτικού Δικτύου σε συνδυασμό με τον μηχανισμό Προσοχής Πολλών Κεφαλών επιτρέπει την παράλληλη, και κατ΄ επέκταση γρήγορη, εκπαίδευση σε νέα δεδομένα και οδηγεί σε μοντέλα υψηλής ακρίβειας. Παράλληλα, τόσο από τη σύγκριση των διαφορετικών μοντέλων, όσο και από τη σύγκριση των προανακλητών Oracle γίνεται εμφανής η σημασία της έννοιας της επικαιρότητας των προανακλήσεων. Ακόμη, οι διαφορετικές μέθοδοι εκπαίδευσης του δικτύου στα δεδομένα εισόδου, φανέρωσαν την ικανότητα εκπαίδευσης των δικτύων τόσο σε μειωμένο πλήθος αιτημάτων πρόσβασης, όσο και σε δεδομένα που έχουν υποστεί δειγματοληψία, με την εκπαίδευση μέσω δειγματοληψίας να παρουσιάζει και καλύτερα αποτελέσματα εφόσον παρέχεται το ιστορικό για κάθε πρόσβαση. Τέλος, είδαμε πως το μοντέλο που χρησιμοποιήθηκε είναι ικανό να εκπαιδευτεί σε ένα σύνολο ιχνών και, μετά από μια διαδικασία επαλήθευσης ώστε να προκύψουν διορθώσεις στα βάρη τους, να παράγουν ακριβείς προανακλήσεις για κάθε ένα από τα ίχνη αυτά. Χάρις αυτών, θεωρούμε πως τα τεχνητά νευρωνικά δίκτυα, και γενικώς ο τομέας της μηχανικής μάθησης μπορούν να αξιοποιηθούν και να προσφέρουν στην προσπάθεια δημιουργία ενός καθολικού προανακλητή στην Κρυφή Μνήμη Αναζήτησης Μετάφρασης.

Παρόλα αυτά, κατά τη διεκπεραίωση της παρούσας εργασίας υπήρξαν πολλές παραδοχές καθώς και δυσκολίες. Αρχικά, σημειώνεται πως η διαδικασία εκπαίδευσης που χρησιμοποιήθηκε, αν και ιδιαίτερα αποδοτική, έχει υψηλές απαιτήσεις μνήμης, καθιστώντας την εκπαίδευση μεγάλου αριθμού δικτύων ιδιαίτερα κοστοβόρα διαδικασία. Παράλληλα, η απόδοση των δικτύων φαίνεται να παρουσιάζει αισθητή πτώση εφόσον υπάρξει απόπειρα γενίκευσής τους για διαφορετικές εφαρμογές στην περίπτωση όπου δεν είναι αρκετά μεγάλος ο αριθμός εντολών εκπαίδευσης και δεν δίνεται στο δίκτυο ευκαιρία να προσαρμοστεί σε αυτές, όπως δηλαδή θα γινόταν και κατά την εκτέλεση σε πραγματικό σύστημα. Ο συνδυασμός των δύο αυτών γεγονότων, καθιστά τη χρήση νευρωνικών δικτύων ως προανακλητή υπό πραγματικές συνθήκες ιδιαίτερα δύσκολη, καθώς από τη μία δεν φαίνεται να είναι εύκολη η

ενσωμάτωση ενός μέρους της διαδικασίας εκπαίδευσης στην εκτέλεση εφαρμογών και από την άλλη δεν φαίνεται να είναι εξίσου αποδοτική η χρήση προ-εκπαιδευμένων μοντέλων.

6.2 Μελλοντικές επεκτάσεις

Τα αποτελέσματα της παραπάνω πειραματικής αξιολόγησης, καθώς και οι παραδοχές που έγιναν στο πλαίσιο της παρούσας εργασίας φανερώνουν τη δυνατότητα περαιτέρω μελέτης σχετικά με την υλοποίηση ενός καθολικά αποδοτικού μηχανισμού προανάκλησης βασισμένο στη μηχανική μάθηση.

- **Περαιτέρω βελτιώσεις σε επίπεδο μοντέλου**
  Ξεκινώντας από πιθανές βελτιώσεις στα μοντέλα που χρησιμοποιήθηκαν, θα είχε ενδιαφέρον η εκτέλεση πολλαπλών προανακλήσεων ανά αριθμό εντολής. Στην παρούσα εργασία, ο αριθμός αιτημάτων προανάκλησης που επιτρεπόταν να εκτελέσει το δίκτυο ήταν σταθερά ίσος με 1, ανεξαρτήτως της αυτοπεποίθησης, ώστε να ταιριάζει με τον αριθμό τον προανακλήσεων που εκτελούν και οι υπόλοιποι προανακλητές στις περισσότερες περιπτώσεις. Ταυτόχρονα, συνήθως αποφεύγεται η εκτέλεση πολλών αιτημάτων προανάκλησης στην TLB, καθώς η Ουρά Προανακλήσεων μπορεί να γεμίσει αρκετά γρήγορα. Παρόλα αυτά, θεωρούμε πως θα μπορούσε να επιτραπεί στο δίκτυο η εκτέλεση περισσότερων προανακλήσεων, εφόσον ξεπερνά κάποιο, πιθανώς υψηλότερο, σχήμα αυτοπεποίθησης, με σκοπό τη βελτίωση της απόδοσης του, καθώς βλέπουμε πως σε πολλές περιπτώσεις τόσο η ακρίβεια, όσο και η κάλυψη των μοντέλων είναι σχετικά υψηλή. Ακόμη, θα είχε ενδιαφέρον και η εκπαίδευση των δικτύων σε αισθητά μικρότερο όγκο δεδομένων, αφού υπάρξει δειγματοληψία, ώστε να αξιολογηθεί η ικανότητα του δικτύου να εντοπίσει αλληλουχίες πρόσβασης με μικρότερο αριθμό δεδομένων εισόδου.

- **Διερεύνηση διαφορετικών μοντέλων**
  Όπως και οποιοδήποτε πρόβλημα στο οποίο επιχειρεί να προσφέρει λύση ο τομέας της μηχανικής μάθησης, έτσι και η προανάκληση μεταφράσεων δύναται να αντιμετωπιστεί με πληθώρα διαφορετικών μοντέλων. Στη συγκεκριμένη ανάλυση αναδείχθηκε η αξία του μηχανισμού Προσοχής Πολλών Κεφαλών στη δημιουργία ενός ακριβούς και αποδοτικού μοντέλου. Παρόλα αυτά, κατά την εκτέλεση σε πραγματικό σύστημα, φαίνεται πως θα ήταν απαραίτητη η χρήση ενός μοντέλου ανά εφαρμογή για την πλήρη εκμετάλλευση των δυνατοτήτων του μοντέλου που αναπτύχθηκε. Παρά λοιπόν την καλή επίδοση του μοντέλου στις μετρικές που χρησιμοποιήθηκαν, είναι απαραίτητη η περαιτέρω διερεύνηση διαφορετικών μοντέλων και τρόπων εκπαίδευσης για τη δημιουργία του ζητούμενου προανακλητή. Συγκεκριμένα, για την λειτουργία σε ένα πραγματικό σύστημα ίσως ήταν θεμιτή η χρήση ενισχυτικής μάθησης[63], καθώς θα έδινε στον προανακλητή τη δυνατότητα να μαθαίνει εν-πτήσει και να προσαρμόζεται στις απαιτήσεις της εκάστοτε εφαρμογής, βοηθώντας στην αντιμετώπιση ορισμένων προβλημάτων που αναφέρθηκαν παραπάνω.

- **Εκτέλεση προανακτήσεων κατά την εκτέλεση**
  Όπως ήδη αναφέρθηκε, η συγκεκριμένη εργασία δεν λαμβάνει υπόψιν τις σημαντικές καθυστερήσεις πρόβλεψης που θα συνόδευαν τη χρήση ενός νευρωνικού δικτύου ως προανακλητή. Οι καθυστερήσεις αυτές, ιδιαίτερα στην περίπτωση όπου η υλοποίηση του προανακλητή βασιζόταν αποκλειστικά σε λογισμικό, θα προκαλούσαν σημαντική μείωση της απόδοσης, ενώ θα κατέρριπταν και την ανάλυση που προηγήθηκε σχετικά με την επικαιρότητα των προανακλήσεων. Στο πλαίσιο αυτό, θα ήταν θεμιτή ο υπολογισμός των καθυστερήσεων αυτών σε κάποιο σενάριο πιο ρεαλιστικής εκτέλεσης. Στη συνέχεια, θα γινόταν δυνατή και η εκ νέου εκπαίδευση του δικτύου με μεγαλύτερη την τιμή της παραμέτρου "Επικαιρότητας", όπως αυτή ορίστηκε παραπάνω, ώστε να φανεί εάν το δίκτυο είναι δυνατό να ανταπεξέλθει σε αυτή.

- **Σχεδιασμός σε επίπεδο υλικού**
  Η υλοποίηση του μηχανισμού προανάκλησης σε ένα σύστημα μπορεί να γίνει είτε μέσω λογισμικού είτε μέσω υλικού, είτε μέσω κάποιου συνδυασμού των δύο. Σε περιπτώσεις όπου η υλοποίηση εκτελείται καθαρά μέσω του λογισμικού, τα δεδομένα αποθηκεύονται στη μνήμη, γεγονός που οδηγεί σε συχνή και ακριβή επικοινωνία για την εκτέλεση προανακλήσεων, προσθέτοντας ακόμη περισσότερες δυσκολίες στη διαδικασία εύρεσης του κατάλληλου χρονισμού εκτέλεσης τους. Αντιθέτως, προανακλητές που βασίζονται στο υλικό είναι ικανοί να ανταποκριθούν ταχύτερα σε μεταβολές κατά τη διάρκεια εκτέλεσης των προγραμμάτων, αδυνατούν όμως συχνά να αντιμετωπίσουν πιο σύνθετες αλληλουχίες προσβάσεων, ενώ καταλαμβάνουν

και χώρο στην επεξεργαστική μονάδα, καθώς θα πρέπει να ενσωματωθούν σε αυτή. Για τους λόγους αυτούς, η χρήση συνδυαστικών υλοποιήσεων γίνεται ολοένα και πιο δημοφιλής[64]. Παρόλα αυτά, όσον αφορά τα νευρωνικά δίκτυα, οι υλοποιήσεις μέσω υλικού, ιδίως υλοποιήσεις ικανές να ενσωματωθούν εντός του επεξεργαστή, αφορούν κατά κύριο λόγο απλούστερα μοντέλα, ενώ παρουσιάζουν και μειωμένη απόδοση εν σχέσει με τις υλοποιήσεις λογισμικού.

- **Εκτέλεση προσομοιώσεων σε ρεαλιστικά συστήματα**
  Στην παρούσα εργασία, το σύνολο των προσομοιώσεων που πραγματοποιήθηκαν καθώς και τα αποτελέσματα που προέκυψαν αφορούν συστήματα ενός πυρήνα. Σε πραγματικά συστήματα όμως, όπου υπάρχουν περισσότεροι πυρήνες και εκτελούνται πολλές διεργασίες ταυτόχρονα, οι επιδόσεις των νευρωνικών δικτύων που χρησιμοποιήθηκαν είναι πολύ πιθανό να διαφέρουν, με την TLB να είναι υπεύθυνη για τη διατήρηση καταχωρήσεων του πίνακα σελίδων για όλες τις διαφορετικές διεργασίες ανά πυρήνα.

Βιβλιογραφία

[1] Ilkka Tuomi. «The Lives and Death of Moore's Law». In: *First Monday* 7 (Apr. 2002). DOI: 10.5210/fm.v7i11.1000.

[2] Arkaprava Basu et al. «Efficient Virtual Memory for Big Memory Servers». In: *SIGARCH Comput. Archit. News* 41.3 (June 2013), pp. 237–248. ISSN: 0163-5964. DOI: 10.1145/2508148.2485943. URL: https://doi.org/10.1145/2508148.2485943.

[3] Abhishek Bhattacharjee. «Large-Reach Memory Management Unit Caches». In: MICRO-46 (2013), pp. 383–394. DOI: 10.1145/2540708.2540741. URL: https://doi.org/10.1145/2540708.2540741.

[4] Bruce L. Jacob and Trevor N. Mudge. «A Look at Several Memory Management Units, TLB-Refill Mechanisms, and Page Table Organizations». In: ASPLOS VIII (1998), pp. 295–306. DOI: 10.1145/291069.291065. URL: https://doi.org/10.1145/291069.291065.

[5] Toni Juan, Tomas Lang, and Juan J. Navarro. «Reducing TLB Power Requirements». In: ISLPED '97 (1997), pp. 196–201. DOI: 10.1145/263272.263332. URL: https://doi.org/10.1145/263272.263332.

[6] Daniel A. Jiménez and Calvin Lin. «Dynamic branch prediction with perceptrons». In: *Proceedings HPCA Seventh International Symposium on High-Performance Computer Architecture* (2001), pp. 197–206.

[7] Chit-Kwan Lin and Stephen J. Tarsa. «Branch Prediction Is Not A Solved Problem: Measurements, Opportunities, and Future Directions». In: *2019 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, Nov. 2019. DOI: 10.1109/iiswc47752.2019.9042108. URL: https://doi.org/10.1109%5C%2Fiiswc47752.2019.9042108.

[8] Stephen J Tarsa et al. *Improving Branch Prediction By Modeling Global History with Convolutional Neural Networks*. 2019. DOI: 10.48550/ARXIV.1906.09889. URL: https://arxiv.org/abs/1906.09889.

[9] Ji-Tae Yun et al. «Effective data prediction method for in-memory database applications». In: *The Journal of Supercomputing* 76 (Jan. 2020). DOI: 10.1007/s11227-019-03050-x.

[10] Zhan Shi et al. «A Hierarchical Neural Model of Data Prefetching». In: ASPLOS '21 (2021), pp. 861–873. DOI: 10.1145/3445814.3446752. URL: https://doi.org/10.1145/3445814.3446752.

[11] Leeor Peled, Uri Weiser, and Yoav Etsion. «A Neural Network Prefetcher for Arbitrary Memory Access Patterns». In: *ACM Trans. Archit. Code Optim.* 16.4 (Oct. 2019). ISSN: 1544-3566. DOI: 10.1145/3345000. URL: https://doi.org/10.1145/3345000.

[12] Peter Braun and Heiner Litz. «Understanding Memory Access Patterns for Prefetching». In: *International Workshop on AI-assisted Design for Architecture (AIDArc), held in conjunction with ISCA* (). URL: https://par.nsf.gov/biblio/10187649.

[13] Pengmiao Zhang et al. *TransforMAP: Transformer for Memory Access Prediction*. 2022. DOI: 10.48550/ARXIV.2205.14778. URL: https://arxiv.org/abs/2205.14778.

[14] Pengmiao Zhang et al. «Fine-Grained Address Segmentation for Attention-Based Variable-Degree Prefetching». In: CF '22 (2022), pp. 103–112. DOI: 10.1145/3528416.3530236. URL: https://doi.org/10.1145/3528416.3530236.

[15] B. Jacob and T. Mudge. «Virtual memory in contemporary microprocessors». In: *IEEE Micro* 18.4 (1998), pp. 60–75. DOI: 10.1109/40.710872.

[16] David A. Patterson and John L. Hennessy. *Computer Architecture: A Quantitative Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990. ISBN: 1558800698.

[17] David A. Patterson and John L. Hennessy. *Computer Organization and Design: The Hardware/Software Interface*. 4th ed. Morgan Kaufmann, 2014. ISBN: 9780080886138.

[18] Vasileios Karakostas et al. «Redundant Memory Mappings for Fast Access to Large Memories». In: *SIGARCH Comput. Archit. News* 43.3S (June 2015), pp. 66–78. ISSN: 0163-5964. DOI: 10.1145/2872887.2749471. URL: https://doi.org/10.1145/2872887.2749471.

[19] Binh Pham et al. «Increasing TLB reach by exploiting clustering in page translations». In: *2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA)*. 2014, pp. 558–567. DOI: 10.1109/HPCA.2014.6835964.

[20] Guilherme Cox and Abhishek Bhattacharjee. «Efficient Address Translation for Architectures with Multiple Page Sizes». In: *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS '17. Xi'an, China: Association for Computing Machinery, 2017, pp. 435–448. ISBN: 9781450344654. DOI: 10.1145/3037697.3037704. URL: https://doi.org/10.1145/3037697.3037704.

[21] Mark Swanson, Leigh Stoller, and John Carter. «Increasing TLB Reach Using Superpages Backed by Shadow Memory». In: *SIGARCH Comput. Archit. News* 26.3 (Apr. 1998), pp. 204–213. ISSN: 0163-5964. DOI: 10.1145/279361.279388. URL: https://doi.org/10.1145/279361.279388.

[22] *Transparent huge pages in 2.6.38*. URL: https://lwn.net/Articles/423584/.

[23] Binh Pham et al. «CoLT: Coalesced Large-Reach TLBs». In: *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*. 2012, pp. 258–269. DOI: 10.1109/MICRO.2012.32.

[24] Thomas W. Barr, Alan L. Cox, and Scott Rixner. «Translation Caching: Skip, Don't Walk (the Page Table)». In: *SIGARCH Comput. Archit. News* 38.3 (June 2010), pp. 48–59. ISSN: 0163-5964. DOI: 10.1145/1816038.1815970. URL: https://doi.org/10.1145/1816038.1815970.

[25] Vasileios Karakostas et al. «Performance analysis of the memory management unit under scale-out workloads». In: *2014 IEEE International Symposium on Workload Characterization (IISWC)*. 2014, pp. 1–12. DOI: 10.1109/IISWC.2014.6983034.

[26] I. Kadayif et al. «Compiler-directed physical address generation for reducing dTLB power». In: *IEEE International Symposium on - ISPASS Performance Analysis of Systems and Software, 2004.* 2004, pp. 161–168. DOI: 10.1109/ISPASS.2004.1291368.

[27] Andreas Kurth et al. «Scalable and Efficient Virtual Memory Sharing in Heterogeneous SoCs with TLB Prefetching and MMU-Aware DMA Engine». In: (2018), pp. 292–300. DOI: 10.1109/ICCD.2018.00052.

[28] Grant Ayers et al. «Classifying Memory Access Patterns for Prefetching». In: ASPLOS '20 (2020), pp. 513–526. DOI: 10.1145/3373376.3378498. URL: https://doi.org/10.1145/3373376.3378498.

[29] Steven P. Vanderwiel and David J. Lilja. «Data Prefetch Mechanisms». In: *ACM Comput. Surv.* 32.2 (June 2000), pp. 174–199. ISSN: 0360-0300. DOI: 10.1145/358923.358939. URL: https://doi.org/10.1145/358923.358939.

[30] G.B. Kandiraju and A. Sivasubramaniam. «Going the distance for TLB prefetching: an application-driven study». In: (2002), pp. 195–206. DOI: 10.1109/ISCA.2002.1003578.

[31] Georgios Vavouliotis et al. «Exploiting Page Table Locality for Agile TLB Prefetching». In: (2021), pp. 85–98. DOI: 10.1109/ISCA52012.2021.00016.

[32] Huaiyu Zhu, Yong Chen, and Xian-He Sun. «Timing local streams: Improving timeliness in data prefetching». In: *Proceedings of the International Conference on Supercomputing* (Jan. 2010), pp. 169–178. DOI: 10.1145/1810085.1810110.

[33] Philip Emma et al. «Exploring the limits of prefetching». In: *IBM Journal of Research and Development* 49 (Jan. 2005), pp. 127–144. DOI: 10.1147/rd.491.0127.

[34] Santhosh Verma and David Koppelman. «The interaction and relative effectiveness of hardware and software data prefetch». In: *Journal of Circuits, Systems and Computers* 21 (Apr. 2012). DOI: 10.1142/S0218126612400026.

[35] Ioannis Vlahavas et al. *Artificial Intelligence*. Jan. 2002. ISBN: 960-7013-28-X.

[36] Simon S. Haykin. *Neural networks and learning machines*. Third. Upper Saddle River, NJ: Pearson Education, 2009.

[37] F. Rosenblatt. *The perceptron - A perceiving and recognizing automaton*. Tech. rep. 85-460-1. Ithaca, New York: Cornell Aeronautical Laboratory, Jan. 1957.
[38] Bing Xu et al. *Empirical Evaluation of Rectified Activations in Convolutional Network*. 2015. DOI: 10.48550/ARXIV.1505.00853. URL: https://arxiv.org/abs/1505.00853.
[39] Dan Hendrycks and Kevin Gimpel. «Gaussian Error Linear Units (GELUs)». In: (2016). DOI: 10.48550/ARXIV.1606.08415. URL: https://arxiv.org/abs/1606.08415.
[40] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.
[41] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. «Layer Normalization». In: (2016). DOI: 10.48550/ARXIV.1607.06450. URL: https://arxiv.org/abs/1607.06450.
[42] Y. Lecun et al. «Gradient-based learning applied to document recognition». In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.
[43] Alex Sherstinsky. «Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network». In: *Physica D: Nonlinear Phenomena* 404 (Mar. 2020), p. 132306. DOI: 10.1016/j.physd.2019.132306. URL: https://doi.org/10.1016%5C%2Fj.physd.2019.132306.
[44] Samy Bengio et al. «Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks». In: (2015). DOI: 10.48550/ARXIV.1506.03099. URL: https://arxiv.org/abs/1506.03099.
[45] Sepp Hochreiter and Jürgen Schmidhuber. «Long Short-term Memory». In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.
[46] Ashish Vaswani et al. «Attention Is All You Need». In: (2017). DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.
[47] Radostin Cholakov and Todor Kolev. «Transformers predicting the future. Applying attention in next-frame and time series forecasting». In: (2021). DOI: 10.48550/ARXIV.2108.08224. URL: https://arxiv.org/abs/2108.08224.
[48] Alexey Dosovitskiy et al. «An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale». In: (2020). DOI: 10.48550/ARXIV.2010.11929. URL: https://arxiv.org/abs/2010.11929.
[49] ChampSim. *ChampSim/ChampSim: Champsim Repository*. URL: https://github.com/ChampSim/ChampSim.
[50] Georgios Vavouliotis et al. «Morrigan: A Composite Instruction TLB Prefetcher». In: MICRO '21 (2021), pp. 1138–1153. DOI: 10.1145/3466752.3480049. URL: https://doi.org/10.1145/3466752.3480049.
[51] Georgios Vavouliotis. *Morrigan: A Composite Instruction TLB Prefetcher*. Sept. 2021. DOI: 10.5281/zenodo.5517143. URL: https://doi.org/10.5281/zenodo.5517143.
[52] Vikengeze. *BenchSim*. URL: https://github.com/vikengeze/BenchSim.
[53] *GAP SUITE*. URL: http://gap.cs.berkeley.edu/index.html.
[54] *SPEC2006 SUITE*. URL: https://www.spec.org/cpu2006/.
[55] *SPEC2017 SUITE*. URL: https://www.spec.org/cpu2017/.
[56] *MLARCHSYS - ml Prefetching competition*. URL: https://sites.google.com/view/mlarchsys/isca-2021/ml-prefetching-competition?authuser=0.
[57] URL: https://utexas.app.box.com/s/2k54kp8zvrqdfaa8cdhfquvcxwh7yn85/folder/132805808026.
[58] Vikengeze. *TraceSim*. URL: https://github.com/vikengeze/TraceSim.
[59] Milad Hashemi et al. «Learning Memory Access Patterns». In: (2018). DOI: 10.48550/ARXIV.1803.02329. URL: https://arxiv.org/abs/1803.02329.
[60] Vikengeze. *TransFetchTLB*. URL: https://cutt.ly/NB8NMfB.
[61] Colin Lea et al. «Temporal Convolutional Networks: A Unified Approach to Action Segmentation». In: (2016). DOI: 10.48550/ARXIV.1608.08242. URL: https://arxiv.org/abs/1608.08242.
[62] Jacob Devlin et al. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». In: (2018). DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805.
[63] Rahul Bera et al. «Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning». In: MICRO '21 (2021), pp. 1121–1137. DOI: 10.1145/3466752.3480114. URL: https://doi.org/10.1145/3466752.3480114.

[64] Santhosh Verma and David Koppelman. ≪The interaction and relative effectiveness of hardware and software data prefetch≫. In: *Journal of Circuits, Systems and Computers* 21 (Apr. 2012). DOI: 10.1142/S0218126612400026.