ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

Τίτλος Πτυχιακής Εργασίας	Χρήση αρχιτεκτονικής ασύμμετρου Bi-encoder στο πλαίσιο της αυτόματης απόδειξης θεωρημάτων Asymmetric Bi-encoder architecture in the field of automated theorem proving
Ονοματεπώνυμο Φοιτητή	Ηλίας Λιγάτος
Πατρώνυμο	Γεώργιος
Αριθμός Μητρώου	Π20114
Επιβλέπων	Διονύσιος Σωτηρόπουλος, Αναπληρωτής Καθηγητής

Ημερομηνία Παράδοσης **Φεβρουάριος 2026**

Copyright ©

Περίληψη

Η παρούσα πτυχιακή εργασία εξετάζει το πρόβλημα της επιλογής μαθηματικών προτάσεων (premise selection) στο πλαίσιο της αυτοματοποιημένης απόδειξης μαθηματικών θεωρημάτων και προτείνει μια εναλλακτική αρχιτεκτονική ασύμμετρου bi-encoder για τη βελτίωση της διαδικασίας ανάκτησης. Συγκεκριμένα, υλοποιείται ένα σύστημα πυκνής διανυσματικής αναπαράστασης (Dense Retrieval ή Dense Embedding system) στο οποίο οι καταστάσεις απόδειξης (proof states) και οι διαθέσιμες μαθηματικές προτάσεις κωδικοποιούνται μέσω δύο διακριτών νευρωνικών κωδικοποιητών (encoders), επιτρέποντας διαφοροποιημένη μοντελοποίηση των δομικών και σημασιολογικών χαρακτηριστικών κάθε τύπου εισόδου. Η προτεινόμενη αρχιτεκτονική αξιολογείται ως προς την ακρίβεια ανάκτησης, αναδεικνύοντας τον ρόλο της εξειδικευμένης αναπαράστασης στη βελτίωση της απόδοσης συστημάτων αυτοματοποιημένης συλλογιστικής.

Λέξεις Κλειδιά: θεώρημα, μαθηματικά, πρόταση, απόδειξη, Lean, Lean-Dojo, ReProver, retrieval, αυτοματοποίηση, ασυμμετρος, bi-encoder, διανυσματική αναπαράσταση, ΤΝ, διαδραστική απόδειξη θεωρημάτων (ITP), αυτοματοποιημένη απόδειξη θεωρημάτων (ATP)

Abstract

This undergraduate thesis investigates the problem of premise selection within the context of automated theorem proving and proposes an alternative asymmetric bi-encoder architecture to enhance the retrieval process. Specifically, a dense vector representation framework is implemented in which proof states and candidate premises are encoded using two distinct neural encoders, allowing differentiated modeling of their structural and semantic characteristics. The proposed architecture is evaluated in terms of retrieval accuracy, highlighting the importance of specialized representations in improving the performance of neural-assisted formal reasoning systems.

Key Words: theorem, mathematics, premise, proof, Lean, Lean-Dojo, ReProver, retrieval, automation, asymmetric bi-encoder, embedding, AI, interactive theorem proving(ITP), automated theorem proving(ATP)

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

1. ΕΙΣΑΓΩΓΗ

Η ιστορία των μαθηματικών είναι, στην πραγματικότητα, η ιστορία της ανθρώπινης προσπάθειας να τιθασεύσει το άπειρο μέσα από τη δύναμη του ορθολογισμού. Από την εποχή που ο Ευκλείδης έθετε τα θεμέλια της Γεωμετρίας, η **μαθηματική απόδειξη** δεν αντιμετωπίστηκε ποτέ ως μια απλή διαδικασία επαλήθευσης, αλλά ως το «χρυσό πρότυπο» της βεβαιότητας. Σε έναν κόσμο γεμάτο ασάφειες, ένα αποδεδειγμένο θεώρημα αποτελεί μια αιώνια και αμετάβλητη αλήθεια, η οποία παραμένει έγκυρη ανεξαρτήτως χρόνου, τόπου ή πολιτισμού.

Ωστόσο, η φύση της απόδειξης είναι διττή. Από τη μία πλευρά, είναι μια άσκηση **απόλυτης αυστηρότητας**, όπου κάθε βήμα πρέπει να ακολουθεί με μηχανική ακρίβεια τους κανόνες της λογικής συνεπαγωγής. Από την άλλη, είναι μια βαθιά **δημιουργική πράξη**. Η εύρεση μιας απόδειξης μοιάζει συχνά με την πλοήγηση σε έναν σκοτεινό λαβύρινθο: ο μαθηματικός πρέπει να επιστρατεύσει τη διαίσθησή του για να επιλέξει το επόμενο μονοπάτι ανάμεσα σε αναρίθμητες πιθανές κατευθύνσεις, συνδιάζοντας προϋπάρχουσες γνώσεις με νέες, καινοτόμες ιδέες.

Στον 21ο αιώνα, αυτή η παραδοσιακή εικόνα του μαθηματικού με το μολύβι και το χαρτί έρχεται αντιμέτωπη με μια νέα πραγματικότητα: την **εκρηκτική πολυπλοκότητα**. Οι σύγχρονες αποδείξεις, όπως αυτή του *Τελευταίου Θεωρήματος του Fermat* ή η ταξινόμηση των *απλών πεπερασμένων ομάδων*, έχουν γίνει τόσο εκτενείς και δαιδαλώδεις, που η ανθρώπινη ικανότητα για έλεγχο και επαλήθευση αγγίζει τα βιολογικά της όρια. Η πιθανότητα ενός ανεπαίσθητου αλλά μοιραίου σφάλματος ελλοχεύει σε κάθε σελίδα, ενώ η αποθήκευση και η διασύνδεση της τεράστιας μαθηματικής γνώσης καθίσταται πλέον αδύνατη χωρίς ψηφιακά μέσα.

Αυτή η πρόκληση γέννησε το πεδίο της **Αυτοματοποιημένης Απόδειξης Θεωρημάτων (Automated Theorem Proving)**. Η μετάβαση από τη χειρόγραφη σημείωση στον τυπικό κώδικα (formal code) δεν είναι απλώς μια αλλαγή μέσου, αλλά μια ριζική αναθεώρηση του τρόπου με τον οποίο παράγεται η γνώση. Το ερώτημα που τίθεται πλέον δεν είναι μόνο αν ένα θεώρημα είναι σωστό, αλλά αν μπορούμε να κατασκευάσουμε ευφυή συστήματα ικανά να «συλλαμβάνουν» τη λογική δομή των μαθηματικών, να ανακτούν τις κατάλληλες πληροφορίες από τεράστιες βιβλιοθήκες και, τελικά, να ανακαλύπτουν αποδείξεις που υπερβαίνουν την ανθρώπινη επεξεργαστική ισχύ.

1.1 Η Δομή και τα Συστατικά της Τυπικής Απόδειξης (Formal Proof)

Για την κατανόηση του προβλήματος της αυτοματοποίησης, είναι απαραίτητο να οριστεί η δομή μιας τυπικής απόδειξης (formal proof) και τα δομικά στοιχεία που την απαρτίζουν. Σε ένα περιβάλλον τυπικής επαλήθευσης (formal verification environment), η απόδειξη δεν είναι μια ελεύθερη περιγραφή συλλογισμών, αλλά μια αυστηρά ιεραρχική διαδικασία. Κάθε απόδειξη ξεκινά με μια **δήλωση (statement)**, η οποία ορίζει τον τελικό στόχο (target goal) και που πρακτικά αποτελεί την ακριβή διατύπωση αυτού που πρόκειται να αποδειχθεί, γραμμένη σε αυστηρή μαθηματική λογική. Για να φτάσει το σύστημα από την αρχική υπόθεση στην τελική επικύρωση, χρησιμοποιεί μια σειρά από **λογικά βήματα**, καθένα από τα οποία μετασχηματίζει τον τρέχοντα στόχο σε έναν ή περισσότερους απλούστερους υπο-στόχους (subgoals), μέχρι αυτοί να καταστούν τετριμμένοι ή ήδη αποδεδειγμένοι.

Κεντρικό ρόλο σε αυτή τη διαδικασία παίζουν οι **προκείμενες (premises)**. Ως προκείμενες ορίζονται όλα τα προϋπάρχοντα μαθηματικά αντικείμενα —όπως αξιώματα, ορισμοί και ήδη αποδεδειγμένα θεωρήματα— τα οποία ανακαλούνται από τις τυπικές μαθηματικές βιβλιοθήκες (formal libraries) για να υποστηρίξουν ένα συγκεκριμένο βήμα της απόδειξης. Η επιλογή των κατάλληλων προκειμένων αποτελεί μια από τις μεγαλύτερες προκλήσεις της αυτοματοποίησης, καθώς η βιβλιοθήκη μπορεί να περιλαμβάνει δεκάδες χιλιάδες επιλογές, ενώ μόνο ελάχιστες από αυτές είναι πραγματικά σχετικές με τον υπό εξέταση στόχο.

Η μετάβαση από τη μία κατάσταση της απόδειξης στην επόμενη υλοποιείται μέσω των **τακτικών (tactics)**. Οι τακτικές είναι ουσιαστικά εντολές ή μικρά προγράμματα που εφαρμόζουν συγκεκριμένους κανόνες λογικής ή αλγεβρικούς χειρισμούς πάνω στον στόχο. Μια τακτική μπορεί, για παράδειγμα, να εφαρμόσει τη μέθοδο της μαθηματικής επαγωγής ή να εκτελέσει μια

απλοποίηση αριθμητικών παραστάσεων. Η κατάσταση της απόδειξης σε μια δεδομένη χρονική στιγμή, η οποία συχνά αναφέρεται ως **proof state**, περιλαμβάνει τον τρέχοντα στόχο καθώς και όλες τις τοπικές υποθέσεις (hypotheses) που είναι διαθέσιμες και ουσιαστικά πρόκειται για ένα «στιγμιότυπο» της απόδειξης. Η αυτοματοποίηση, επομένως, στοχεύει στη δημιουργία ενός συστήματος που, βλέποντας το proof state, θα μπορεί να επιλέγει την ιδανική τακτική και τις απαραίτητες προκείμενες για να οδηγήσει την απόδειξη στην ολοκλήρωσή της.

1.2 Η Αυτοματοποίηση της Διαδικασίας Απόδειξης Μαθηματικών Θεωρημάτων

Η προσπάθεια για την αυτοματοποίηση της μαθηματικής απόδειξης μετατρέπει μια παραδοσιακά πνευματική και δημιουργική διαδικασία σε ένα δομημένο πρόβλημα υπολογιστικής λογικής. Παρακάτω αναλύονται οι βασικοί άξονες που ορίζουν αυτό το πρόβλημα:

Η Μαθηματική Πρόταση ως Στόχος (Conjecture as a Goal)

Στο πλαίσιο της αυτοματοποίησης, κάθε μαθηματικό θεώρημα αντιμετωπίζεται ως ένας «στόχος» (goal) που πρέπει να επιτευχθεί. Η διαδικασία ξεκινά με μια αρχική κατάσταση, η οποία περιλαμβάνει τη δήλωση της πρότασης που επιθυμούμε να αποδείξουμε. Το σύστημα καλείται να βρει μια σειρά από λογικούς μετασχηματισμούς που θα οδηγήσουν από τις γνωστές υποθέσεις στην τελική απόδειξη. Η δυσκολία εδώ έγκειται στην ακριβή αναπαράσταση της πρότασης: σε αντίθεση με τη φυσική γλώσσα που επιτρέπει ασάφειες, η υπολογιστική αναπαράσταση πρέπει να είναι πλήρης και μονοσήμαντη, ώστε να μπορεί να υποβληθεί σε αυστηρό λογικό έλεγχο.

Ο Χώρος Αναζήτησης και η Λογική Συνεπαγωγή

Το κύριο χαρακτηριστικό του προβλήματος είναι ο αχανής **χώρος αναζήτησης (search space)**. Κάθε βήμα μιας απόδειξης προσφέρει πολυάριθμες επιλογές λογικών κανόνων που μπορούν να εφαρμοστούν. Αυτό δημιουργεί ένα δέντρο πιθανοτήτων που διακλαδίζεται εκθετικά σε κάθε επίπεδο, οδηγώντας σε αυτό που ονομάζουμε **συνδυαστική έκρηξη (combinatorial explosion)**. Η αυτοματοποίηση, επομένως, δεν είναι απλώς θέμα υπολογιστικής ισχύος, αλλά θέμα «στρατηγικής νοημοσύνης»: το σύστημα πρέπει να είναι σε θέση να αξιολογεί ποιοι δρόμοι είναι ελπιδοφόροι και ποιοι οδηγούν σε λογικά αδιέξοδα, περιορίζοντας την αναζήτηση στα πιο πιθανά σενάρια.

Τυπική Επαλήθευση έναντι Διαισθητικής Απόδειξης

Μια κρίσιμη πτυχή της αυτοματοποίησης είναι η μετάβαση από τη «διαισθητική» απόδειξη, όπως την αντιλαμβάνεται ο άνθρωπος, στην **τυπική επαλήθευση (formal verification)**. Αυτό, ουσιαστικά, σημαίνει ότι ενώ ένας μαθηματικός μπορεί να παραλείψει «προφανή» βήματα, ένας υπολογιστής απαιτεί κάθε λεπτομέρεια να είναι ρητά διατυπωμένη. Η αυτοματοποίηση επιδιώκει να γεφυρώσει αυτό το χάσμα, κατασκευάζοντας συστήματα που μπορούν να παράγουν αποδείξεις οι οποίες δεν είναι απλώς «πειστικές», αλλά μαθηματικά αδιάσειστες, καθώς κάθε τους βήμα ελέγχεται από έναν αυστηρό λογικό ελεγκτή.

Η Αυτοματοποίηση ως Πρόβλημα Μοντελοποίησης

Από την πλευρά της Επιστήμης των Υπολογιστών, η απόδειξη θεωρημάτων μοντελοποιείται ως ένα πρόβλημα **μετάβασης καταστάσεων (state-space search)**.

- Η **Κατάσταση (State)** αντιπροσωπεύει όσα έχουν αποδειχθεί μέχρι στιγμής.
- Η **Μετάβαση (Transition)** αντιπροσωπεύει την εφαρμογή ενός λογικού κανόνα ή ενός υπάρχοντος θεωρήματος. Το πρόβλημα της αυτοματοποίησης είναι να βρεθεί ένα «μονοπάτι» (path) από την «κατάσταση της εικασίας» στην «κατάσταση της πλήρους τεκμηριωμένης αλήθειας». Αυτή η μοντελοποίηση επιτρέπει τη χρήση αλγορίθμων τεχνητής νοημοσύνης και μηχανικής μάθησης για την καθοδήγηση της αναζήτησης, μετατρέποντας τη «λογική» σε ένα πρόβλημα βελτιστοποίησης και ανάκτησης πληροφορίας.

Κατηγοριοποίηση των Συστημάτων

Τέλος, η αυτοματοποίηση χωρίζεται σε δύο μεγάλες προσεγγίσεις:

- **Αυτόματοι «Αποδείκτες» (Automated Theorem Provers - ATP):** Συστήματα

που προσπαθούν να βρουν την απόδειξη εντελώς αυτόνομα, συνήθως για προβλήματα περιορισμένης πολυπλοκότητας.

- **Συστήματα Υποβοήθησης (Interactive Proof Assistants):** Συστήματα όπου ο άνθρωπος και η μηχανή συνεργάζονται. Η αυτοματοποίηση σε αυτά τα συστήματα στοχεύει στο να αναλαμβάνει ο υπολογιστής τα πιο τεχνικά και επαναλαμβανόμενα κομμάτια της απόδειξης, επιτρέποντας στον χρήστη να εστιάζει στην υψηλού επιπέδου στρατηγική.

1.3 Η Σημασία του Προβλήματος Αυτοματοποίησης της Διαδικασίας Παραγωγής Αποδείξεων

Η επίλυση του προβλήματος της αυτοματοποιημένης απόδειξης θεωρημάτων υπερβαίνει τα όρια των καθαρών μαθηματικών και αγγίζει κρίσιμους τομείς της σύγχρονης κοινωνίας, αποτελώντας πλέον κρίσιμο παράγοντα για την εξέλιξη της τεχνολογίας και την αξιοπιστία των σύγχρονων ψηφιακών υποδομών. Η σημασία της αυτοματοποίησης αυτής μπορεί να συνοψιστεί στους ακόλουθους άξονες:

Διασφάλιση Κρίσιμων Συστημάτων (Safety-Critical Systems)

Μία από τις σημαντικότερες εφαρμογές του πεδίου εντοπίζεται στον τομέα της τυπικής επαλήθευσης (formal verification), όπου η ικανότητα μιας μηχανής να παράγει αδιάσειστες αποδείξεις εγγυάται την ορθότητα συστημάτων κρίσιμης σημασίας. Σε περιβάλλοντα όπου το ανθρώπινο ή το οικονομικό κόστος ενός σφάλματος είναι δυσβάσταχτο, όπως στον σχεδιασμό επεξεργαστών, στα συστήματα ελέγχου αυτόνομης οδήγησης ή στα κρυπτογραφικά πρωτόκολλα, οι παραδοσιακές μέθοδοι δοκιμών δεν επαρκούν. Η αυτοματοποιημένη απόδειξη μας εξασφαλίζει, μαθηματικώς, ότι ένα σύστημα συμπεριφέρεται ακριβώς όπως προβλέπουν οι προδιαγραφές του για κάθε πιθανή κατάσταση εισόδου, εξαλείφοντας την πιθανότητα απρόβλεπτων αστοχιών που θα μπορούσαν να έχουν καταστροφικές συνέπειες.

Επιτάχυνση της Μαθηματικής Έρευνας

Παράλληλα, η αυτοματοποίηση αυτή λειτουργεί ως καταλύτης για την επιτάχυνση της ίδιας της μαθηματικής έρευνας, η οποία αντιμετωπίζει σήμερα μια κρίση πολυπλοκότητας. Καθώς οι σύγχρονες αποδείξεις γίνονται ολοένα και πιο εκτενείς, η διαδικασία του ομότιμου ελέγχου από την επιστημονική κοινότητα γίνεται εξαιρετικά χρονοβόρα και επιρρεπής σε ανθρώπινα λάθη. Τα αυτοματοποιημένα συστήματα προσφέρουν τα απαραίτητα εργαλεία για τον άμεσο και απόλυτα ακριβή έλεγχο δαιδαλωδών αποδείξεων, διασφαλίζοντας την εγκυρότητά τους. Επιπλέον, η υπολογιστική ισχύς επιτρέπει την εξερεύνηση λογικών μονοπατιών και συνδυασμών που η ανθρώπινη διαίσθηση μπορεί να παραβλέψει, ανοίγοντας τον δρόμο για την ανακάλυψη νέων θεωρημάτων και τη σύνδεση φαινομενικά ασύνδετων μαθηματικών κλάδων.

Η Απόδειξη Θεωρημάτων ως "Benchmark" για την Τεχνητή Νοημοσύνη

Στο πλαίσιο της Τεχνητής Νοημοσύνης (ΤΝ), η απόδειξη θεωρημάτων αναδεικνύεται ως το απόλυτο κριτήριο αξιολόγησης για τις ικανότητες συλλογιστικής (reasoning) των μοντέλων. Σε αντίθεση με άλλες εφαρμογές της ΤΝ, όπως η παραγωγή κειμένου, όπου η επιτυχία είναι συχνά υποκειμενική, στη μαθηματική απόδειξη η ορθότητα είναι «δυαδική» (η απόδειξη που παράχθηκε θα είναι είτε σωστή είτε λάθος) και αντικειμενική. Η ενασχόληση με αυτό το πρόβλημα ωθεί την τεχνολογία πέρα από την απλή στατιστική πρόβλεψη λέξεων, απαιτώντας την ανάπτυξη αρχιτεκτονικών που συνδυάζουν τη δημιουργικότητα των νευρωνικών δικτύων με την αυστηρότητα της συμβολικής λογικής. Αυτή η σύγκλιση είναι απαραίτητη για τη δημιουργία συστημάτων AI που δεν θα «παπαγαλίζουν» απλώς δεδομένα, αλλά θα μπορούν να επιλύουν σύνθετα προβλήματα μέσω δομημένης σκέψης.

Οργάνωση και Προσβασιμότητα της Παγκόσμιας Γνώσης

Τέλος, η αυτοματοποίηση συμβάλλει καθοριστικά στην οργάνωση και την προσβασιμότητα

της παγκόσμιας μαθηματικής γνώσης. Η ψηφιοποίηση των μαθηματικών σε τυπικές βιβλιοθήκες (formal libraries) επιτρέπει σε ερευνητές από όλο τον κόσμο να έχουν πρόσβαση σε ένα επαληθευμένο σώμα γνώσης, το οποίο είναι άμεσα «αναζητήσιμο» και «χρησιμοποιήσιμο» από υπολογιστικά συστήματα. Η δημιουργία ευφυών μηχανισμών ανάκτησης (retrieval) επιτρέπει τη σύνδεση νέων προβλημάτων με υπάρχοντα θεωρήματα, εκδημοκρατίζοντας την πρόσβαση σε εξειδικευμένες γνώσεις και δημιουργώντας ένα σταθερό θεμέλιο για τις μελλοντικές επιστημονικές ανακαλύψεις. Με τον τρόπο αυτό, η ανθρώπινη λογική κωδικοποιείται σε μια μορφή που παραμένει αιώνια, αλάνθαστη και δυναμικά επεκτάσιμη.

1.4 Σκοπός της Πτυχιακής Εργασίας

Ο κύριος σκοπός της παρούσας πτυχιακής εργασίας είναι η σχεδίαση και η πειραματική αξιολόγηση μιας εναλλακτικής αρχιτεκτονικής που αποσκοπεί στην αυτοματοποίηση της επιλογής προκείμενων (premise selection) κατά τη διαδικασία της μαθηματικής απόδειξης ενός θεωρήματος από τον υπολογιστή. Το επίκεντρο της μελέτης εντοπίζεται στη χρήση **ασύμμετρων διπλών κωδικοποιητών (Asymmetric Bi-Encoders)** για τη μοντελοποίηση της σχέσης μεταξύ των διαθέσιμων μαθηματικών «γνώσεων» (προκείμενων) και των τρεχόντων στόχων προς απόδειξη. Η προσέγγιση αυτή βασίζεται στην παραδοχή ότι η δεδομένη κατάσταση μιας απόδειξης σε μια συγκεκριμένη χρονική στιγμή και οι μαθηματικές προτάσεις που είναι αποθηκευμένες σε μια βιβλιοθήκη διαφέρουν ως προς τη δομή και το περιεχόμενο, απαιτώντας έτσι εξειδικευμένους μηχανισμούς κωδικοποίησης για την αποτελεσματική συσχέτισή τους.

Στο πλαίσιο αυτό, η εργασία επιδιώκει να αναπτύξει μια ολοκληρωμένη ροή εργασίας (pipeline) η οποία επιτρέπει τη μετατροπή τυπικών μαθηματικών βιβλιοθηκών (Formal Mathematical Libraries) σε έναν διανυσματικό χώρο υψηλών διαστάσεων. Μέσω της εκπαίδευσης του ασύμμετρου μοντέλου, ο στόχος είναι η δημιουργία ενός συστήματος ανάκτησης που μπορεί να ταυτοποιεί σε πραγματικό χρόνο τα πιο σχετικά λήμματα για κάθε βήμα της απόδειξης, περιορίζοντας δραστικά τη συνδυαστική έκρηξη των πιθανών λογικών κινήσεων. Η εργασία εξετάζει πώς η σημασιολογική αναπαράσταση των μαθηματικών αντικειμένων μπορεί να γεφυρώσει το χάσμα μεταξύ της στατιστικής πρόβλεψης και της αυστηρής λογικής συνεπαγωγής, καθιστώντας την εύρεση αποδείξεων υπολογιστικά βιώσιμη.

Τέλος, η παρούσα μελέτη στοχεύει στην εξαγωγή συμπερασμάτων σχετικά με τη γενικευτική ικανότητα της προτεινόμενης αρχιτεκτονικής. Μέσω της διεξαγωγής πειραμάτων, αξιολογείται η ακρίβεια της ανάκτησης τόσο σε περιβάλλοντα με γνωστά κατά την εκπαίδευση δεδομένα, όσο και σε περιπτώσεις όπου το σύστημα καλείται να διαχειριστεί εντελώς νέες και άγνωστες μαθηματικές έννοιες. Με αυτόν τον τρόπο, η πτυχιακή εργασία φιλοδοξεί να προσφέρει μια τεκμηριωμένη ανάλυση για το πώς η αρχιτεκτονική του ασύμμετρου Bi-Encoder μπορεί να λειτουργήσει ως ένας ισχυρός βοηθός για τα συστήματα τυπικής επαλήθευσης, συμβάλλοντας στην εξέλιξη της αυτοματοποιημένης επιστημονικής συλλογιστικής.

1.5 Δομή της Εργασίας

Η παρούσα πτυχιακή εργασία είναι δομημένη με τρόπο που να καλύπτει σφαιρικά το θεωρητικό υπόβαθρο, τη μεθοδολογία, την υλοποίηση και την αξιολόγηση της πειραματικής εφαρμογής ενός ασύμμετρου bi-encoder στην αρχιτεκτονική του μηχανισμού ανάκτησης (retriever) του μοντέλου αναφοράς (baseline model) ReProver, το οποίο βασίζεται στο framework LeanDojo. Κάθε κεφάλαιο έχει σχεδιαστεί ώστε να προσφέρει στον αναγνώστη μια ολοκληρωμένη εικόνα για το αντικείμενο, τους στόχους, τις τεχνολογικές επιλογές και τα αποτελέσματα της εργασίας.

Αναλυτικότερα, η δομή της εργασίας έχει ως εξής:

- Κεφάλαιο 1: Εισαγωγή

Παρουσιάζεται το θεωρητικό πλαίσιο της τυπικής απόδειξης, ορίζεται το πρόβλημα της αυτοματοποίησης και η σημασία του για την επιστήμη και την τεχνολογία. Επιπλέον, προσδιορίζεται ο σκοπός της εργασίας, εισάγεται η έννοια του ασύμμετρου bi-encoder και περιγράφεται η συνολική δομή της μελέτης.

- Κεφάλαιο 2: Ανασκόπηση Βιβλιογραφίας

Εξετάζονται επιστημονικές εργασίες και προγενέστερα μοντέλα που σχετίζονται με την αυτοματοποιημένη απόδειξη θεωρημάτων και τη νευρωνική ανάκτηση πληροφορίας. Αναλύεται η παρούσα κατάσταση στο πεδίο και εντοπίζονται οι προκλήσεις τις οποίες επιχειρεί να αντιμετωπίσει

η παρούσα αρχιτεκτονική.

- Κεφάλαιο 3: Το framework LeanDojo

Παρουσιάζεται αναλυτικά η αρχιτεκτονική του LeanDojo, ο τρόπος αλληλεπίδρασης με τη γλώσσα Lean 4 και οι βασικές έννοιες των προκειμένων (premises) και των τακτικών. Περιγράφεται η λειτουργική δομή που επιτρέπει τη σύνδεση των μαθηματικών βιβλιοθηκών με μοντέλα μηχανικής μάθησης.

- Κεφάλαιο 4: Σύνολα Δεδομένων

Περιγράφονται τα δεδομένα που χρησιμοποιήθηκαν για την εκπαίδευση και την αξιολόγηση του μοντέλου, με ιδιαίτερη έμφαση στις μαθηματικές βιβλιοθήκες της Lean (Mathlib). Αναλύονται οι διαφορετικοί τρόποι διαχωρισμού των δεδομένων (random και novel premises) για την αξιολόγηση της γενικευτικής ικανότητας του συστήματος.

- Κεφάλαιο 5: Η Ροή Εργασίας της Αρχιτεκτονικής Υλοποίησης

Αναλύεται ο σχεδιασμός του ασύμμετρου bi-encoder και η υλοποίηση του pipeline ανάκτησης. Περιγράφονται οι τεχνικές λεπτομέρειες της εκπαίδευσης, οι επιλογές των υπερπαραμέτρων (hyperparameters) και η διαδικασία ενσωμάτωσης του νέου retriever στο ευρύτερο σύστημα του ReProver.

- Κεφάλαιο 6: Πειραματικά Αποτελέσματα και Αξιολόγηση

Παρουσιάζονται και αναλύονται τα αποτελέσματα των πειραμάτων, εστιάζοντας στην ακρίβεια της ανάκτησης (retrieval accuracy) και στη συνολική απόδοση του μοντέλου. Γίνεται σύγκριση των αποτελεσμάτων μεταξύ των διαφορετικών συνόλων δεδομένων και εξάγονται συμπεράσματα για την αποτελεσματικότητα της αρχιτεκτονικής.

- Κεφάλαιο 7: Συμπεράσματα και Μελλοντικές Επεκτάσεις

Συνοψίζονται τα κύρια ευρήματα της εργασίας, αξιολογείται η συμβολή της προτεινόμενης μεθοδολογίας και προτείνονται κατευθύνσεις για μελλοντική έρευνα και βελτίωση των συστημάτων αυτοματοποιημένης απόδειξης.

- Κεφάλαιο 8: Βιβλιογραφία

Παρατίθενται όλες οι επιστημονικές πηγές, τα τεχνικά εγχειρίδια και οι βιβλιοθήκες λογισμικού που χρησιμοποιήθηκαν για την εκπόνηση της εργασίας.

Η παραπάνω δομή διασφαλίζει τη συστηματική παρουσίαση όλων των πτυχών της εργασίας, από τη θεωρητική τεκμηρίωση έως την πρακτική εφαρμογή και ποσοτική αξιολόγηση της προτεινόμενης λύσης.

2. ΑΝΑΣΚΟΠΗΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑΣ

Η ανασκόπηση της διεθνούς βιβλιογραφίας αποτελεί το απαραίτητο θεμέλιο για την κατανόηση της εξέλιξης του πεδίου της αυτοματοποιημένης απόδειξης θεωρημάτων, από τις πρώτες συμβολικές προσεγγίσεις έως τα σύγχρονα νευρωνικά μοντέλα. Σκοπός του παρόντος κεφαλαίου είναι η παρουσίαση των σημαντικότερων ερευνητικών εργασιών που διαμόρφωσαν το τοπίο της μαθηματικής συλλογιστικής μέσω υπολογιστή, δίνοντας ιδιαίτερη έμφαση στη μετάβαση από τους άκαμπτους κανόνες λογικής στην ευέλικτη μάθηση από δεδομένα. Μέσα από την ανάλυση επιλεγμένων μοντέλων και αρχιτεκτονικών, αναδεικνύεται η σταδιακή ανάγκη για αποτελεσματικούς μηχανισμούς ανάκτησης πληροφορίας, οι οποίοι επιτρέπουν στα Μεγάλα Γλωσσικά Μοντέλα να πλοηγούνται σε τεράστιες βιβλιοθήκες τυπικών μαθηματικών. Η ιστορική και τεχνική αυτή αναδρομή κρίνεται απαραίτητη, καθώς τοποθετεί την παρούσα πτυχιακή εργασία στο πλαίσιο της σύγχρονης έρευνας και δικαιολογεί την επιλογή του ασύμμετρου bi-encoder ως εργαλείου για τη βελτιστοποίηση της διαδικασίας απόδειξης.

2.1 Πρωτοπόρες Προσεγγίσεις στη Νευρωνική Ανάκτηση και Απόδειξη Θεωρημάτων

Η συστηματική προσπάθεια ενσωμάτωσης νευρωνικών δικτύων στην απόδειξη θεωρημάτων ξεκίνησε με την εργασία **DeepMath (Alemi et al., 2016)**, η οποία θεωρείται ορόσημο για το πρόβλημα της επιλογής προκειμένων. Οι ερευνητές της Google απέδειξαν για πρώτη φορά ότι μοντέλα βαθιάς μάθησης, όπως τα Συνελικτικά Νευρωνικά Δίκτυα (CNN) και τα δίκτυα Long Short-Term Memory (LSTM), μπορούν να εκπαιδευτούν ώστε να αναγνωρίζουν ποια λήμματα είναι χρήσιμα για την απόδειξη ενός στόχου μέσα από ένα τεράστιο σύνολο δεδομένων (Mizar dataset). Η σημασία του DeepMath έγκειται στο γεγονός ότι ξεπέρασε τις παραδοσιακές στατιστικές μεθόδους (όπως το TF-IDF), αποδεικνύοντας ότι η Τεχνητή Νοημοσύνη μπορεί να συλλάβει τη σημασιολογική δομή των μαθηματικών εκφράσεων και όχι απλώς να μετράει κοινές λέξεις ή σύμβολα.

Στη συνέχεια, η έρευνα επεκτάθηκε σε πιο σύνθετα περιβάλλοντα με το σύστημα **HOList (Bansal et al., 2019)**, το οποίο επικεντρώθηκε στο διαδραστικό σύστημα απόδειξης HOL Light. Το HOList εισήγαγε μια ολοκληρωμένη υποδομή όπου ένας νευρωνικός αποδείκτης (neural prover) μπορούσε να "επικοινωνεί" με το σύστημα απόδειξης σε πραγματικό χρόνο. Η συγκεκριμένη εργασία ανέδειξε τη σημασία της συνδυασμένης μάθησης: το μοντέλο έπρεπε ταυτόχρονα να προβλέπει την κατάλληλη τακτική και να επιλέγει τις σωστές προκείμενες από τη βιβλιοθήκη. Ήταν η πρώτη φορά που είδαμε τη χρήση γραφημάτων (Graph Neural Networks) για την αναπαράσταση μαθηματικών τύπων, γεγονός που επέτρεψε στα μοντέλα να αντιλαμβάνονται τις εσωτερικές εξαρτήσεις και την ιεραρχία των μαθηματικών οντοτήτων με μεγαλύτερη ακρίβεια.

Η πραγματική τομή στη χρήση των Μεγάλων Γλωσσικών Μοντέλων (LLMs) ήρθε με το **GPT-f (Polu & Sutskever, 2020)** από την OpenAI. Αντί για εξειδικευμένες αρχιτεκτονικές, οι ερευνητές χρησιμοποίησαν μοντέλα βασισμένα στον Transformer (παρόμοια με το GPT-3) για να παράγουν αποδείξεις στο σύστημα Metamath. Το GPT-f απέδειξε ότι η πρόβλεψη του επόμενου βήματος μιας απόδειξης ως πρόβλημα παραγωγής κειμένου είναι εξαιρετικά αποτελεσματική, καταφέρνοντας μάλιστα να βρει αποδείξεις για θεωρήματα που δεν είχαν επιλυθεί από προηγούμενους αυτοματοποιημένους αποδείκτες. Αυτή η εργασία προετοίμασε το έδαφος για τα σύγχρονα συστήματα, καθώς έδειξε ότι η τεράστια προ-εκπαίδευση σε κείμενο και κώδικα προσδίδει στο μοντέλο μια "μαθηματική λογική" που μπορεί να εξειδικευτεί στην τυπική επαλήθευση.

2.2 Σύγχρονες Αρχιτεκτονικές Σημασιολογικής Ανάκτησης και Μαθηματικής Αναπαράστασης

Magnushammer: Η Επικράτηση των Transformers στην Επιλογή Προκειμένων

Το Magnushammer αποτελεί μία από τις σημαντικότερες πρόσφατες εξελίξεις στο πρόβλημα της επιλογής προκειμένων (premise selection), μεταφέροντας την επιτυχία των αρχιτεκτονικών Transformer από το πεδίο της επεξεργασία φυσικής γλώσσας στην τυπική επαλήθευση. Η κύρια συνεισφορά της εργασίας έγκειται στην απόδειξη ότι η σημασιολογική αναπαράσταση των μαθηματικών τύπων, μέσω προ-εκπαιδευμένων μοντέλων, μπορεί να υποκαταστήσει πλήρως τους παραδοσιακούς αλγόριθμους που βασίζονταν σε προ-σχεδιασμένα χαρακτηριστικά (hand-engineered features).

Σε αντίθεση με παλαιότερα συστήματα που χρησιμοποιούσαν στατιστικές μεθόδους για την αντιστοίχιση συμβόλων, το Magnushammer εκπαιδεύεται ως ένας ισχυρός «ταξινομητής συνάφειας». Συγκεκριμένα, το μοντέλο αξιοποιεί τον μηχανισμό αυτο-προσοχής (self-attention) για να κωδικοποιήσει τον τρέχοντα στόχο (goal) και μια υποψήφια προκείμενη (premise) σε έναν κοινό διανυσματικό χώρο. Η αρχιτεκτονική του βασίζεται στην ιδέα της Αντιπαραβολικής Μάθησης (Contrastive Learning), όπου το μοντέλο μαθαίνει να "φέρνει κοντά" στον διανυσματικό χώρο τα θεωρήματα που είναι απαραίτητα για μια απόδειξη, ενώ ταυτόχρονα "απομακρύνει" τα άσχετα λήμματα.

Η χρήση του Magnushammer ως «προ-επεξεργαστή» (pre-processor) για το εργαλείο Sledgehammer του Isabelle/HOL οδήγησε σε εντυπωσιακή αύξηση της τάξης του 20-30% στην επίλυση θεωρημάτων που προηγουμένως θεωρούνταν απροσπέλαστα. Αυτό οφείλεται στην ικανότητα του μοντέλου να διαχειρίζεται τεράστια σώματα δεδομένων χωρίς την ανάγκη για εξειδικευμένους κανόνες από ειδικούς, θέτοντας νέα πρότυπα για το τι μπορεί να επιτύχει ένας νευρωνικός retriever στην αυτοματοποιημένη ανακάλυψη λογικών συνδέσεων.

Llemma: Ένα Ανοιχτό Γλωσσικό Μοντέλο για Μαθηματική Συλλογιστική

Ενώ το Magnushammer εστιάζει στη διαδικασία της επιλογής, το μοντέλο Llemma επικεντρώνεται στην ποιότητα της μαθηματικής αναπαράστασης. Πρόκειται για ένα Μεγάλο Γλωσσικό Μοντέλο (LLM) που προέκυψε από τη συνεχή προ-εκπαίδευση (continued pre-training) του μοντέλου CodeLlama πάνω στο σύνολο δεδομένων Proof-Pile-2. Το σύνολο αυτό, μεγέθους 55 δισεκατομμυρίων tokens, περιλαμβάνει επιστημονικά άρθρα από το arXiv, μαθηματικό κώδικα από το GitHub και τυπικές βιβλιοθήκες από συστήματα όπως το Lean, το Coq και το Isabelle.

Η εκτενής αυτή προ-εκπαίδευση επιτρέπει στο Llemma να διαθέτει μια βαθιά κατανόηση των μαθηματικών εννοιών, κάτι που είναι ζωτικής σημασίας για τη δημιουργία ποιοτικών διανυσματικών αναπαραστάσεων (embeddings). Στο πλαίσιο ενός bi-encoder, η ικανότητα του μοντέλου να αντιλαμβάνεται ότι δύο διαφορετικές διατυπώσεις αναφέρονται στην ίδια μαθηματική ιδιότητα (π.χ. η μεταθετικότητα μιας πράξης) είναι αυτή που καθορίζει την επιτυχία της ανάκτησης.

Το Proof-Pile-2 δεν είναι μια απλή συλλογή κειμένων, αλλά ένα μείγμα επιστημονικής γνώσης και κώδικα. Η συμπερίληψη του μαθηματικού κώδικα επιτρέπει στο μοντέλο να μαθαίνει την αλγοριθμική δομή των αποδείξεων, ενώ τα άρθρα προσφέρουν το εννοιολογικό πλαίσιο. Αυτή η συνέργεια επιτρέπει στον παραγόμενο bi-encoder να γεφυρώνει το χάσμα μεταξύ της ανθρώπινης διαίσθησης και της αυστηρής μηχανικής επαλήθευσης. Επιπλέον, το Llemma παρουσιάζει ικανότητες "few-shot learning", ενώ η ανοιχτή φύση του επέτρεψε στην ερευνητική κοινότητα να το χρησιμοποιήσει ως ραχοκοκαλιά (backbone) για την ανάπτυξη εξειδικευμένων συστημάτων ανάκτησης.

Η συνδυαστική μελέτη των δύο αυτών προσεγγίσεων καταδεικνύει ότι η επιτυχής απόδειξη θεωρημάτων απαιτεί τόσο έναν ισχυρό μηχανισμό επιλογής (Magnushammer) όσο και μια βαθιά, προ-εκπαιδευμένη γνώση του μαθηματικού πεδίου (Llemma). Αυτή η ανάγκη οδηγεί νομοτελειακά στην υιοθέτηση αρχιτεκτονικών που ενσωματώνουν και τις δύο πτυχές σε ένα ενιαίο πλαίσιο, προετοιμάζοντας το έδαφος για τα σύγχρονα συστήματα ανάκτησης-παραγωγής (Retrieval-Augmented Generation - RAG).

2.3 Συστήματα Ανάκτησης-Παραγωγής (RAG) και η Αιχμή της Τεχνολογίας στη Lean 4

REAL-Prover (2025): Βηματική Ανάκτηση και το Benchmark FATE-M

Μία από τις βασικές συνεισφορές του REAL-Prover είναι η έμφαση στην έννοια της βηματικής ανάκτησης (stepwise retrieval), δηλαδή της συνεχούς επαναξιολόγησης των σχετικών προκείμενων κατά τη διάρκεια της διαδικασίας απόδειξης. Στα σύγχρονα συστήματα αυτοματοποιημένης απόδειξης που βασίζονται σε retrieval-augmented architectures, η επιλογή των κατάλληλων προκείμενων αποτελεί κρίσιμο παράγοντα για την επιτυχία της απόδειξης. Συστήματα όπως ο ReProver ήδη χρησιμοποιούν μηχανισμούς ανάκτησης που εξαρτώνται από την τρέχουσα κατάσταση της απόδειξης, επαναλαμβάνοντας τη διαδικασία retrieval σε πολλαπλά βήματα καθώς το proof state μεταβάλλεται. Ωστόσο, η βασική ιδέα που αναδεικνύεται στο REAL-Prover είναι ότι η διαδικασία αυτή μπορεί να αξιοποιηθεί ακόμη πιο συστηματικά ως κεντρικός μηχανισμός καθοδήγησης της αναζήτησης. Οι ερευνητές επισημαίνουν ότι στις αποδείξεις μαθηματικών θεωρημάτων, ιδιαίτερα σε περιβάλλοντα υψηλής πολυπλοκότητας, ο στόχος της απόδειξης μετασχηματίζεται συνεχώς μέσω της εφαρμογής τακτικών (tactics). Κάθε νέα κατάσταση απόδειξης μπορεί να απαιτεί διαφορετικό σύνολο σχετικών θεωρημάτων ή λημμάτων. Για τον λόγο αυτό, το REAL-Prover ενισχύει τον δυναμικό κύκλο αλληλεπίδρασης μεταξύ retriever και proof state, επιτρέποντας στο σύστημα να επαναπροσδιορίζει το σχετικό μαθηματικό context σε κάθε βήμα της διαδικασίας απόδειξης. Με αυτόν τον τρόπο, το μοντέλο μπορεί να εντοπίζει λήμματα που καθίστανται χρήσιμα μόνο μετά από ενδιάμεσους μετασχηματισμούς της απόδειξης, προσεγγίζοντας περισσότερο τη στρατηγική που ακολουθεί ένας ανθρώπινος μαθηματικός κατά την επίλυση ενός προβλήματος.

Σε τεχνικό επίπεδο, το σύστημα REAL-Prover εισάγει τον retriever LeanSearch-PS, ο οποίος βασίζεται σε μια αρχιτεκτονική bi-encoder για την αναπαράσταση τόσο των καταστάσεων απόδειξης όσο και των διαθέσιμων προκείμενων. Στην προσέγγιση αυτή, δύο ανεξάρτητα νευρωνικά δίκτυα χρησιμοποιούνται για την κωδικοποίηση των proof states και των μαθηματικών προκείμενων σε έναν κοινό διανυσματικό χώρο ενσωμάτωσης. Η σημαντική διαφοροποίηση από τον ReProver έγκειται στον τρόπο εκπαίδευσης του συστήματος, όπου αξιοποιούνται τα λεγόμενα proof traces, δηλαδή πλήρεις αλληλουχίες βημάτων από πραγματικές αποδείξεις που προέρχονται από τη βιβλιοθήκη Mathlib4. Αντί το μοντέλο να μαθαίνει απλώς επιφανειακές συσχετίσεις μεταξύ περιγραφών θεωρημάτων και στόχων απόδειξης, εκπαιδεύεται ώστε να αναγνωρίζει πώς συγκεκριμένες προκείμενες χρησιμοποιούνται στην πράξη για τη μετάβαση από μία κατάσταση απόδειξης σε μια επόμενη. Με τη χρήση σύγχρονων dense encoders, όπως μοντέλα της οικογένειας E5, το σύστημα δημιουργεί διανυσματικές αναπαραστάσεις που κωδικοποιούν όχι μόνο τη συντακτική μορφή των μαθηματικών εκφράσεων, αλλά και τη λειτουργική τους σημασία μέσα στη διαδικασία απόδειξης. Έτσι καθίσταται δυνατή η ανάκτηση προκείμενων που είναι ουσιαστικά απαραίτητες για την πρόοδο της απόδειξης, ακόμη και όταν δεν υπάρχει άμεση λεξιλογική ομοιότητα με τον τρέχοντα στόχο.

Ένα ακόμη σημαντικό στοιχείο της αρχιτεκτονικής του REAL-Prover είναι η εισαγωγή του framework Jixia-Interactive, το οποίο λειτουργεί ως ενδιάμεσο επίπεδο αλληλεπίδρασης μεταξύ του μοντέλου μηχανικής μάθησης και του περιβάλλοντος εκτέλεσης της Lean 4. Η μετάβαση στη Lean 4 συνοδεύεται από αυξημένη πολυπλοκότητα στη διαχείριση του compiler και των εσωτερικών καταστάσεων του συστήματος απόδειξης. Η συνεχής επαλήθευση τακτικών μπορεί να συνεπάγεται σημαντικό υπολογιστικό κόστος, ιδιαίτερα όταν η διαδικασία απόδειξης περιλαμβάνει μεγάλο αριθμό δοκιμών και εναλλακτικών μονοπατιών. Το Jixia αντιμετωπίζει αυτό το πρόβλημα παρέχοντας έναν αποδοτικό μηχανισμό διαχείρισης του proof environment, επιτρέποντας στο σύστημα να εφαρμόζει τακτικές και να λαμβάνει άμεσα ανατροφοδότηση από

τον Lean compiler χωρίς να απαιτείται επανεκκίνηση της διαδικασίας επαλήθευσης. Παράλληλα, η κατάσταση της απόδειξης οργανώνεται σε μια δομή αναζήτησης τύπου δέντρου, στην οποία καταγράφονται τα διαφορετικά μονοπάτια που εξετάζονται. Η υποδομή αυτή είναι ιδιαίτερα σημαντική για συστήματα που βασίζονται σε retrieval, καθώς επιτρέπει την ταχεία επανεκτίμηση των σχετικών προκειμένων μετά από κάθε μεταβολή του proof state.

Για την αξιολόγηση της απόδοσης του συστήματος, το REAL-Prover εισάγει το σύνολο δεδομένων FATE-M (Formal Algebra Theorem Evaluation – Medium), το οποίο επικεντρώνεται σε προβλήματα αφηρημένης άλγεβρας, όπως θεωρήματα που αφορούν ομάδες, δακτυλίους και πεδία. Η σημασία του συγκεκριμένου benchmark έγκειται στο ότι δίνει έμφαση σε καταστάσεις που χαρακτηρίζονται ως «novel contexts». Σε πολλά πειραματικά σύνολα δεδομένων, τα μοντέλα μπορούν να εμφανίσουν υψηλή απόδοση επειδή έχουν ήδη συναντήσει παρόμοιες δομές ή ακόμα και συγκεκριμένα θεωρήματα κατά τη διαδικασία εκπαίδευσης. Το FATE-M επιχειρεί να περιορίσει αυτό το φαινόμενο, περιλαμβάνοντας προβλήματα που απαιτούν τον συνδυασμό προκειμένων οι οποίες δεν εμφανίζονται μαζί στο training set ή που προέρχονται από διαφορετικά τμήματα της βιβλιοθήκης Mathlib4. Με τον τρόπο αυτό, αξιολογείται η πραγματική ικανότητα του συστήματος να γενικεύει σε νέα μαθηματικά προβλήματα και να εντοπίζει τις κατάλληλες προκείμενες για την επίλυση προβλημάτων που δεν έχουν εμφανιστεί προηγουμένως κατά την εκπαίδευση.

Εκτός από τον μηχανισμό ανάκτησης, το σύστημα REAL-Prover ενσωματώνει επίσης στρατηγικές αναζήτησης που καθοδηγούν τη διαδικασία παραγωγής αποδείξεων. Μία από αυτές είναι η χρήση παραλλαγών της στρατηγικής Best-First Search, η οποία επιτρέπει στο σύστημα να εξετάζει πολλαπλά πιθανά μονοπάτια απόδειξης και να δίνει προτεραιότητα σε εκείνα που εμφανίζουν μεγαλύτερη πιθανότητα επιτυχίας. Σε αυτό το πλαίσιο, η πληροφορία που παρέχεται από τον retriever χρησιμοποιείται ως πρόσθετο σήμα αξιολόγησης για τις προτεινόμενες τακτικές. Εάν οι προκείμενες που ανακτώνται παρουσιάζουν χαμηλή συνάφεια με την τρέχουσα κατάσταση απόδειξης, το σύστημα μπορεί να μειώσει τη βαθμολογία του αντίστοιχου μονοπατιού αναζήτησης και να εξετάσει εναλλακτικές επιλογές. Η ενσωμάτωση της ανάκτησης στη διαδικασία αναζήτησης συμβάλλει στη μείωση των λεγόμενων «hallucinations», δηλαδή περιπτώσεων όπου το μοντέλο προτείνει τακτικές ή θεωρήματα που δεν είναι λογικά συμβατά με το τρέχον proof state. Με αυτόν τον τρόπο, ο μηχανισμός retrieval λειτουργεί ως φίλτρο που περιορίζει την ανεξέλεγκτη παραγωγή πιθανών βημάτων από το γλωσσικό μοντέλο.

Συνολικά, η προσέγγιση του REAL-Prover συμβάλλει στην εξέλιξη των συστημάτων αυτοματοποιημένης απόδειξης θεωρημάτων αναδεικνύοντας τη σημασία της αποτελεσματικής ανάκτησης μαθηματικών γνώσεων κατά τη διάρκεια της διαδικασίας απόδειξης. Αντί να βασίζεται αποκλειστικά στην αύξηση του μεγέθους των γλωσσικών μοντέλων, η προσέγγιση αυτή δίνει έμφαση στη βελτίωση της αλληλεπίδρασης μεταξύ retriever, μοντέλου παραγωγής (generation model) και τυπικού συστήματος επαλήθευσης. Τα πειραματικά αποτελέσματα δείχνουν ότι η ύπαρξη ενός αποδοτικού μηχανισμού ανάκτησης προκείμενων μπορεί να επηρεάσει καθοριστικά την επιτυχία της διαδικασίας απόδειξης, ακόμη και όταν χρησιμοποιούνται μοντέλα μέτριου μεγέθους. Η παρατήρηση αυτή έχει ιδιαίτερη σημασία για την παρούσα πτυχιακή εργασία, καθώς υπογραμμίζει τον κεντρικό ρόλο που διαδραματίζει η ποιότητα των διανυσματικών αναπαραστάσεων και της αρχιτεκτονικής του retriever στην απόδοση των συστημάτων απόδειξης σε περιβάλλοντα όπως η Lean 4.

HybriL (Hybrid Language Models for Theorem Proving, 2024): Υβριδική Ανάκτηση και η Σύγκλιση Σημασιολογικής και Συμβολικής Αναζήτησης

Η κεντρική ιδέα του HybriL (2024-2025) βασίζεται στην παραδοχή ότι η καθαρά νευρωνική ανάκτηση (Dense Retrieval) συχνά αποτυγχάνει στα μαθηματικά λόγω του προβλήματος του "vocabulary gap", όπου διαφορετικά σύμβολα μπορεί να περιγράφουν την ίδια έννοια ή το ίδιο σύμβολο να έχει διαφορετική σημασία σε άλλο context. Οι συγγραφείς υποστηρίζουν ότι τα τυπικά μαθηματικά απαιτούν δύο είδη «μνήμης»: μια **σημασιολογική** (για την κατανόηση της γενικής μαθηματικής ιδέας) και μια **λεκτική/συμβολική** (για την ακριβή εύρεση ταυτοτήτων). Το HybriL εισάγει μια υβριδική προσέγγιση που συνδυάζει τα πυκνά embeddings (Dense), τα οποία εκπαιδεύονται με contrastive loss, με την παραδοσιακή αραιή ανάκτηση (Sparse Retrieval), όπως ο αλγόριθμος BM25. Αυτή η συνέργεια επιτρέπει στο σύστημα να ανακτά λήμματα που είναι είτε εννοιολογικά συναφή (π.χ. θεωρήματα που αφορούν ισομορφισμούς) είτε περιέχουν ακριβώς τους ίδιους εξειδικευμένους μαθηματικούς τελεστές ή ονόματα μεταβλητών που ορίζονται στη Mathlib4.

Μια από τις σημαντικότερες τεχνικές συνεισφορές της συγκεκριμένης εργασίας είναι η

λεπτομερής ανάλυση του «χάσματος απόδοσης» (performance gap) των Bi-Encoders. Παρόλο που οι Bi-Encoders είναι εξαιρετικά γρήγοροι λόγω του "pre-computation" των embeddings, αδυνατούν να συλλάβουν τις λεπτομερείς αλληλεπιδράσεις (fine-grained interactions) μεταξύ των συμβόλων του στόχου και της προκείμενης, καθώς η σύγκριση γίνεται μόνο μέσω ενός απλού εσωτερικού γινομένου (cosine similarity). Το HybriL επιλύει αυτό το δομικό πρόβλημα χρησιμοποιώντας τον Bi-Encoder μόνο για ένα αρχικό «φιλτράρισμα» (candidate retrieval) και στη συνέχεια επιστρατεύει έναν **Cross-Encoder**. Ο Cross-Encoder δέχεται ως είσοδο την ένωση (concatenation) του goal και της premise, επιτρέποντας στους μηχανισμούς self-attention του μοντέλου να συσχετίσουν κάθε σύμβολο του στόχου με κάθε σύμβολο της υποψήφιας προκείμενης, επιτυγχάνοντας έτσι μια πιο έγκυρη ιεράρχηση των λημμάτων με βάση την πραγματική λογική τους συνάφεια με τον εκάστοτε αποδεικτικό στόχο (proof goal).

Στην αρχιτεκτονική του HybriL, η διαδικασία της ανάκτησης χωρίζεται σε δύο διακριτά στάδια (multi-stage retrieval), μια πρακτική που δανείζεται από τα σύγχρονα συστήματα ανάκτησης πληροφορίας (Information Retrieval). Στο πρώτο στάδιο, ο Bi-Encoder και ο BM25 λειτουργούν παράλληλα για να επιστρέψουν μια ευρεία λίστα πιθανών θεωρημάτων (candidates). Στο δεύτερο στάδιο, ένας μηχανισμός **Re-ranking** (βασισμένος στον Cross-Encoder) αναλαμβάνει να ταξινομήσει εκ νέου αυτά τα αποτελέσματα. Το HybriL μάλιστα προτείνει μια μέθοδο **Knowledge Distillation**, όπου ο ακριβός Cross-Encoder λειτουργεί ως «δάσκαλος» για να εκπαιδεύσει τον Bi-Encoder να παράγει καλύτερα embeddings. Αυτό δείχνει ότι η ποιότητα της ανάκτησης εξαρτάται από τον τρόπο με τον οποίο τα embeddings συνεργάζονται με την ιεραρχική ταξινόμηση για να φέρουν το «σωστό» λήμμα στις θέσεις Top-1 ή Top-5, μειώνοντας δραματικά τον υπολογιστικό θόρυβο για τον generator.

Το HybriL δίνει τεράστια έμφαση στην «ευαισθησία στα σύμβολα» (symbolic sensitivity), αναγνωρίζοντας ότι τα τυπικά γλωσσικά μοντέλα συχνά «συγχέουν» διαφορετικούς μαθηματικούς τελεστές (π.χ. το «+» και το «-») αν αυτοί εμφανίζονται σε παρόμοιο λεκτικό context. Για να το αντιμετωπίσει αυτό, το paper περιγράφει τη χρήση ενός **Custom Byte-Pair Encoding (BPE)**, ο οποίος έχει προσαρμοστεί ειδικά για τον κώδικα της Lean 4. Αυτός ο tokenizer εξασφαλίζει ότι σύνθετες μαθηματικές εκφράσεις και ειδικοί χαρακτήρες (όπως οι Unicode χαρακτήρες της Lean) δεν τεμαχίζονται σε άσχετα υπο-tokens, αλλά διατηρούν τη σημασιολογική τους ενότητα. Αυτό επιτρέπει στον υβριδικό retriever να αναγνωρίζει την ακριβή δομή μιας μαθηματικής πρότασης, διασφαλίζοντας ότι η υβριδική ανάκτηση θα δώσει απόλυτη προτεραιότητα σε λήμματα που ταιριάζουν δομικά με το προς απόδειξη θεώρημα.

Τα πειράματα του HybriL στη βιβλιοθήκη Mathlib4 έδειξαν ότι η υβριδική προσέγγιση ξεπερνά το απλό RAG (όπως αυτό του ReProver) κατά ένα εντυπωσιακό ποσοστό, ειδικά σε θεωρήματα που ανήκουν στη «μακρά ουρά» (long-tail distribution) της γνώσης. Μέσα από λεπτομερή **Ablation Studies**, το paper αποδεικνύει ότι η αφαίρεση του Sparse Retrieval (BM25) οδηγεί σε κατακόρυφη πτώση της ακρίβειας σε προβλήματα που απαιτούν σπάνια θεωρήματα ή εξειδικευμένους ορισμούς που βρίσκονται στη "μακρά ουρά" της βιβλιοθήκης, ενώ η αφαίρεση του Cross-Encoder μειώνει την ικανότητα του μοντέλου να διακρίνει μεταξύ πολύ παρόμοιων λημμάτων. Η συνέργεια αυτή αποδεικνύεται σωτήρια για την διαδικασία της αυτοματοποιημένης απόδειξης, καθώς περιορίζει τις λανθασμένες επιλογές προκειμένων που συχνά οδηγούν τα LLMs σε λογικά αδιέξοδα, αυξάνοντας το συνολικό ποσοστό επιτυχίας (pass@k) σε σύνθετα benchmarks.

DeepSeek-Prover (2025): Ενισχυτική Μάθηση και Αναζήτηση σε Χώρο Καταστάσεων μέσω MCTS

Η θεμελιώδης φιλοσοφία του DeepSeek-Prover βασίζεται στην πλέον καλά εδραιωμένη παραδοχή ότι η απόδειξη θεωρημάτων δεν αποτελεί ένα πρόβλημα απλής παραγωγής επόμενου token (next-token prediction), αλλά ένα πρόβλημα στρατηγικής πλοήγησης σε έναν εκθετικά αυξανόμενο χώρο καταστάσεων. Ενώ τα παραδοσιακά LLMs προσπαθούν να «μαντέψουν» την επόμενη τακτική (tactic) βασιζόμενα αποκλειστικά σε στατιστικές πιθανότητες, το DeepSeek-Prover αντιμετωπίζει τη Mathlib4 ως έναν γράφο όπου κάθε κόμβος αντιπροσωπεύει ένα αποδεικτικό στάδιο (proof state) και κάθε ακμή μια έγκυρη τακτική. Αυτή η εναλλακτική θεώρηση επιτρέπει στο μοντέλο να μην περιορίζεται σε μια γραμμική συλλογιστική, αλλά να εξερευνά πολλαπλά εναλλακτικά μονοπάτια σε πραγματικό χρόνο, χρησιμοποιώντας το LLM ως «πυξίδα» (policy network) και τον Lean compiler ως έναν αλάνθαστο κριτή (verifier) που επικυρώνει κάθε κίνηση.

Στον πυρήνα του DeepSeek-Prover είναι ο αλγόριθμος Monte-Carlo Tree Search (MCTS), ο

οποίος επιτρέπει στο σύστημα να εκτελεί προσομοιώσεις (rollouts) πριν καταλήξει στην τελική επιλογή ενός βήματος. Σε κάθε στάδιο της απόδειξης, το μοντέλο παράγει μια δέσμη υποψήφιων τακτικών και ο MCTS τις αξιολογεί εκτελώντας δοκιμαστικές διαδρομές για να εκτιμήσει ποια οδηγεί πιο κοντά στην επίλυση. Το σημαντικότερο πλεονέκτημα αυτής της προσέγγισης είναι η δυνατότητα οπισθοδρόμησης (backtracking): εάν μια επιλογή —παρά την υψηλή αρχική της πιθανότητα— οδηγήσει σε λογικό αδιέξοδο ή σφάλμα μεταγλώττισης, ο αλγόριθμος επιστρέφει αυτόματα σε προηγούμενο κόμβο και δοκιμάζει μια εναλλακτική στρατηγική. Αυτή η δυναμική διόρθωση λαθών επιτρέπει στο DeepSeek να επιλύει θεωρήματα που απαιτούν δεκάδες βήματα, εκεί όπου απλούστερα συστήματα θα είχαν αποτύχει λόγω ενός και μόνο λανθασμένου ενδιάμεσου βήματος.

Μια από τις πλέον καινοτόμες πτυχές του συστήματος είναι η χρήση Ενισχυτικής Μάθησης (Reinforcement Learning) μέσω της μεθόδου RLEF (Reinforcement Learning from Freeform Proof Feedback). Το DeepSeek-Prover δεν εκπαιδεύτηκε μόνο με την επίβλεψη ανθρώπινων αποδείξεων (Supervised Learning), αλλά και μέσω μιας συνεχούς διαδικασίας αυτο-βελτίωσης στο περιβάλλον της Lean 4. Το μοντέλο «παίζει» με τον compiler, προσπαθώντας να λύσει χιλιάδες προβλήματα αυτόνομα· κάθε φορά που μια απόδειξη γίνεται δεκτή, το μοντέλο λαμβάνει μια θετική επιβράβευση (reward), η οποία αξιοποιείται για τη **διαρκή εκπαίδευση και βελτίωση του μοντέλου**, διασφαλίζοντας ότι ο μηχανισμός ανάκτησης και παραγωγής "μαθαίνει" να αναγνωρίζει τα πιο ελπιδοφόρα αποδεικτικά μονοπάτια. Αυτή η προσέγγιση επιτρέπει στον εσωτερικό retriever να ανακαλύπτει «μη προφανείς» συνδέσεις και λήμματα που συχνά διαφεύγουν από τους ανθρώπους, καθιστώντας το σύστημα εξαιρετικά δημιουργικό στην εύρεση συντομότερων και πιο αποδοτικών αποδεικτικών μονοπατιών.

Το DeepSeek-Prover βασίζεται στο DeepSeek-Coder, ένα μοντέλο που έχει προ-εκπαιδευτεί σε εκατοντάδες δισεκατομμύρια γραμμές κώδικα (όπως Python, C++ και Java). Οι συγγραφείς απέδειξαν πειραματικά ότι η ικανότητα κατανόησης της δομής του προγραμματισμού μεταφέρεται άμεσα στο πεδίο της τυπικής επαλήθευσης (Formal Verification). Αυτή η **εξοικείωση με ποικίλα προγραμματιστικά πρότυπα** βοηθά τον bi-encoder του μοντέλου να "διαβάζει" πίσω από τη σύνταξη της Lean 4 και έτσι να αντιλαμβάνεται τη «λειτουργική» σημασία ενός θεωρήματος στη Lean (για παράδειγμα, πώς αλληλεπιδρούν τα macros και τα type classes) και όχι μόνο τη γλωσσική του διατύπωση. Αυτό οδηγεί σε μια πολύ πιο ακριβή ανάκτηση προκειμένων, καθώς το μοντέλο μπορεί να «διακρίνει» ποιο λήμμα είναι δομικά συμβατό με τον τρέχοντα στόχο, ακόμη και αν η ονομασία του είναι διαφορετική από την αναμενόμενη.

Ένα κρίσιμο πρόβλημα που επιλύει το DeepSeek-Prover είναι η υπολογιστική επιβάρυνση που προκαλεί η συνεχής κλήση του Lean compiler. Το μοντέλο εισάγει μια στρατηγική «ασύγχρονης επαλήθευσης», όπου ο generator παράγει πολλαπλά υποψήφια βήματα παράλληλα, ενώ ο verifier τα ελέγχει σε ξεχωριστά threads. Αυτό επιτρέπει στο σύστημα να διατηρεί μια υψηλή ταχύτητα αναζήτησης στο δέντρο αποφάσεων χωρίς να «κολλάει» στις καθυστερήσεις του compiler. Επιπλέον, το σύστημα χρησιμοποιεί ένα value function (συνάρτηση αξίας) για να «κλαδεύει» (prune) νωρίς τα μονοπάτια που φαίνονται μη ελπιδοφόρα, διασφαλίζοντας ότι οι πόροι για την ανάκτηση και παραγωγή (RAG) εστιάζονται μόνο στις πιο πιθανές λύσεις, αποφεύγοντας έτσι την υπολογιστική έκρηξη που χαρακτηρίζει την τυφλή αναζήτηση.

DT-Solver (2025): Ιεραρχική Οργάνωση του Αποδεικτικού Χώρου και Στοχευμένη Ανάκτηση ανά Μαθηματικό Πεδίο

Το DT-Solver (2025) εισηγείται μια νέα προσέγγιση στο πρόβλημα της ανάκτησης, εστιάζοντας στο «χάσμα πεδίου» (domain gap) που υπάρχει μέσα στην ίδια τη Mathlib4. Για παράδειγμα, οι συγγραφείς παρατήρησαν ότι η σημασιολογική αναπαράσταση ενός λήμματος στην Τοπολογία μπορεί να απέχει διανυσματικά από ένα λήμμα της Θεωρίας Ομάδων, ακόμα και αν χρησιμοποιούν παρόμοια λογική δομή. Αντί για ένα ενιαίο μοντέλο που προσπαθεί να ισοσταθμίσει αυτές τις διαφορές, το DT-Solver εισάγει έναν προκαταρκτικό ταξινομητή πεδίου (Domain Classifier). Αυτός ο μηχανισμός αναλύει τον αποδεικτικό στόχο και τον κατατάσσει σε μια ιεραρχική δομή μαθηματικών κλάδων. Με αυτόν τον τρόπο, το σύστημα περιορίζει εκ των προτέρων το πεδίο δράσης του retriever, διασφαλίζοντας ότι οι προκείμενες που θα ανακτηθούν θα προέρχονται από το σχετικό μαθηματικό πλαίσιο, εξαλείφοντας τον κίνδυνο παρεμβολών από άσχετα αλλά συντακτικά παρόμοια λήμματα.

Σε αντίθεση με τα παραδοσιακά LLMs που λειτουργούν με μια «greedy» λογική παραγωγής κειμένου, το DT-Solver ενσωματώνει ένα δυναμικό Δέντρο Αποφάσεων (Decision Tree) που λειτουργεί ως ενορχηστρωτής. Κάθε κόμβος του δέντρου αντιπροσωπεύει μια στρατηγική επιλογή

υψηλού επιπέδου: για παράδειγμα, το μοντέλο αποφασίζει πρώτα αν η κατάσταση απαιτεί «απλοποίηση» (simplification), «εφαρμογή λήμματος» (lemma application) ή «επαγωγική διαδικασία» (induction). Μόλις ληφθεί αυτή η δομική απόφαση, ο μηχανισμός ανάκτησης ενεργοποιείται στοχευμένα μόνο για εκείνη την κατηγορία. Αυτό μετατρέπει την ανάκτηση από μια χαοτική αναζήτηση σε ολόκληρη τη βάση δεδομένων σε μια ιεραρχική διαδικασία φιλτραρίσματος, όπου κάθε επίπεδο του δέντρου μειώνει το θόρυβο και αυξάνει την πιθανότητα εντοπισμού της βέλτιστης προκείμενης.

Η τεχνική υπεροχή του DT-Solver εντοπίζεται στη χρήση μιας αρχιτεκτονικής Mixture of Encoders (MoE). Αντί για έναν γενικό Bi-Encoder, το σύστημα διαθέτει μια συστοιχία από εξειδικευμένους encoders, καθένας από τους οποίους έχει εκπαιδευτεί σε διαφορετικά υπο-σύνολα της Lean 4 (π.χ. Ανάλυση, Άλγεβρα, Λογική). Όταν ο μηχανισμός αναγνώρισης μαθηματικού πεδίου (Domain Classifier) προσδιορίζει το context, ενεργοποιείται ο αντίστοιχος expert encoder. Αυτό επιτρέπει τη δημιουργία πολύ πιο «πυκνών» και ακριβέστερων embeddings, καθώς ο encoder δεν χρειάζεται να «θυμάται» όλη τη βιβλιοθήκη, αλλά εστιάζει στις λεπτές εννοιολογικές διαφορές του συγκεκριμένου πεδίου. Αυτό δείχνει πώς η αρχιτεκτονική MoE μπορεί να λύσει το πρόβλημα του «semantic crowding», όπου διαφορετικές έννοιες καταλήγουν να αναπαρίστανται από παρόμοια διανύσματα λόγω υπερφόρτωσης του μοντέλου.

Ένα από τα πιο καινοτόμα χαρακτηριστικά του DT-Solver είναι ο αλγόριθμος κλαδέματος (pruning) που εφαρμόζεται στο δέντρο αποφάσεων. Καθώς το μοντέλο εξερευνά πιθανές αποδείξεις, αξιολογεί τη συνάρτηση αξίας (value function) για κάθε πιθανή διακλάδωση. Εάν ένας συνδυασμός τακτικής και ανακτηθείσας προκείμενης δεν οδηγήσει σε μείωση της πολυπλοκότητας του proof state (με βάση τον Lean compiler), ολόκληρος ο κλάδος απορρίπτεται ακαριαία. Αυτό το «έξυπνο» κλάδεμα αποτρέπει την εκθετική «έκρηξη» που παρατηρείται στην αναζήτηση κατά πλάτος (breadth-first search) και επιτρέπει στο DT-Solver να επιλύει σύνθετα προβλήματα με ένα κλάσμα των υπολογιστικών πόρων που απαιτούν μοντέλα όπως το GPT-4, καθιστώντας το ιδανικό για χρήση σε ακαδημαϊκά περιβάλλοντα.

Η εκπαίδευση του DT-Solver βασίστηκε σε ένα ειδικά διαμορφωμένο dataset, το οποίο δεν περιλάμβανε μόνο ζεύγη (στόχος, απόδειξη), αλλά και μετα-δεδομένα στρατηγικής (strategy tags). Οι ερευνητές χρησιμοποίησαν τεχνικές "knowledge distillation" για να εξάγουν την κρυφή δομή των αποδείξεων που έχουν γραφτεί από ειδικούς στη Lean. Με αυτόν τον τρόπο, ο retriever του DT-Solver έμαθε να αναγνωρίζει «μοτίβα χρησιμότητας» (utility patterns). Για παράδειγμα, έμαθε ότι όταν ένας στόχος περιλαμβάνει ανισότητες, ορισμένα λήμματα της Mathlib4 είναι 80% πιο πιθανό να χρησιμοποιηθούν σε συνδυασμό με την τακτική linarith. Αυτή η συσχέτιση μεταξύ τακτικής και ανάκτησης είναι που προσδίδει στο μοντέλο μια σχεδόν ανθρώπινη «διαίσθηση» κατά την επιλογή των προκειμένων.

Η ανάλυση του DT-Solver καταλήγει σε ένα σημαντικό συμπέρασμα. Η ποιότητα ενός Bi-Encoder δεν κρίνεται μόνο από τη γενική του ακρίβεια (Top-K), αλλά από την ικανότητά του να ανταποκρίνεται σε πολυεπίπεδες δομές λήψης αποφάσεων (hierarchical decision-making structures), όπου η ανάκτηση προκειμένων καθοδηγείται από την προηγούμενη ταξινόμηση του μαθηματικού πεδίου. Το DT-Solver αποδεικνύει ότι η ενσωμάτωση πληροφοριών πεδίου (domain awareness) στα embeddings μπορεί να μεταμορφώσει την απόδοση ενός retriever.

Thor (2024): Νευρο-συμβολική (Neuro-symbolic) Προσέγγιση και Ενοποίηση Γλωσσικών Μοντέλων με Αυτοματοποιημένους Αποδείκτες (ATPs)

Το Thor (2024) εισηγείται μια εξελιγμένη νευρο-συμβολική αρχιτεκτονική, η οποία στοχεύει στην επίλυση ενός από τα μεγαλύτερα μειονεκτήματα των Μεγάλων Γλωσσικών Μοντέλων (LLMs): την έλλειψη εγγυημένης λογικής συνέπειας. Ενώ τα LLMs διαθέτουν μια σχεδόν «διαισθητική» ικανότητα να πλοηγούνται σε τεράστιες βιβλιοθήκες κώδικα όπως η Mathlib4, συχνά παράγουν αποδείξεις που περιέχουν μικρά αλλά καταστροφικά λογικά άλματα (hallucinations). Το Thor αντιμετωπίζει αυτό το ζήτημα χρησιμοποιώντας το LLM ως έναν «παραγωγό υποθέσεων» (hypothesis generator) και αναθέτοντας την τελική επικύρωση σε Αυτοματοποιημένους Αποδείκτες Θεωρημάτων (ATPs) όπως το Vampire και το E. Αυτή η συνέργεια επιτρέπει στο σύστημα να συνδυάζει την ευελιξία της φυσικής γλώσσας με την απόλυτη αυστηρότητα των συμβολικών συστημάτων, δημιουργώντας ένα πλαίσιο όπου η δημιουργικότητα της Τεχνητής Νοημοσύνης ελέγχεται διαρκώς από αδιάψευστους λογικούς κανόνες.

Η μετάφραση από τη γλώσσα Lean 4 σε Λογική Πρώτης Τάξης (First-Order Logic - FOL) δεν είναι μια απλή διαδικασία αντιστοίχισης λέξεων, αλλά ένας βαθύς σημασιολογικός μετασχηματισμός. Η Lean 4 χρησιμοποιεί έναν πλούσιο φορμαλισμό εξαρτημένων τύπων

(Dependent Types), ο οποίος πρέπει να «απογυμνωθεί» από ορισμένες υψηλού επιπέδου δομές για να γίνει κατανοητός από τους ATP solvers, χωρίς όμως να χαθεί η λογική του εγκυρότητα. Το σύστημα Thor ενσωματώνει προηγμένους αλγόριθμους κωδικοποίησης τύπων (type encoding), οι οποίοι μετατρέπουν τις ιεραρχίες των κλάσεων και τα implicit arguments σε ρητά λογικά αξιώματα (axioms).

Η τεχνική "Hammer" αποτελεί το επίκεντρο της αποτελεσματικότητας του Thor. Σε ένα τυπικό μαθηματικό περιβάλλον (formal mathematical environment), υπάρχουν χιλιάδες πιθανά λήμματα που θα μπορούσαν να χρησιμοποιηθούν, δημιουργώντας έναν εκθετικά αυξανόμενο χώρο αναζήτησης που θα «γονάτιζε» οποιονδήποτε κλασικό solver. Το Thor χρησιμοποιεί τον Bi-Encoder για να εκτελέσει μια εξαιρετικά στοχευμένη επιλογή προκειμένων (Premise Selection), περιορίζοντας τα υποψήφια λήμματα στα 16-32 πιο σχετικά. Αυτή η δραστική μείωση επιτρέπει στους ATPs να επικεντρωθούν στην εύρεση του συνδυαστικού μονοπατιού (proof path) αντί να αναλώνονται στην αξιολόγηση άσχετων δεδομένων. Η επιτυχία αυτής της προσέγγισης αποδεικνύει ότι η «ευφυΐα» ενός συστήματος αυτόματης απόδειξης εξαρτάται άμεσα από την ποιότητα του φιλτραρίσματος που επιτυγχάνει ο retriever.

Ένα από τα πιο ισχυρά χαρακτηριστικά του Thor είναι η ικανότητα ανακατασκευής της απόδειξης (Proof Reconstruction) μέσα στο περιβάλλον της Lean. Όταν ο εξωτερικός solver (π.χ. Vampire) βρίσκει μια λύση, το Thor δεν σταματά εκεί· προσπαθεί να μεταφράσει τα βήματα του solver πίσω σε έγκυρο κώδικα Lean 4, χρησιμοποιώντας τακτικές όπως η smt ή η metis. Αν η ανακατασκευή πετύχει, η απόδειξη θεωρείται πλήρως επαληθευμένη. Αυτή η διαδικασία τροφοδοτεί έναν βρόχο αυτο-βελτίωσης, όπου το μοντέλο μαθαίνει από τις επιτυχείς ανακατασκευές ποιες προκείμενες είναι πράγματι «χρήσιμες» (effective premises) και ποιες είναι απλώς «σχετικές» (relevant premises), επιτρέποντας στον Bi-Encoder να βαθμονομεί τις αναπαραστάσεις του με βάση τη τελική έκβαση της απόδειξης.

Οι ερευνητές του Thor παρατήρησαν ένα παράδοξο φαινόμενο: η προσθήκη περισσότερων προκειμένων συχνά μείωνε την πιθανότητα εύρεσης λύσης από τον solver, ένα πρόβλημα γνωστό ως "Premise Overload". Για να το αντιμετωπίσει, το Thor εισάγει μια δυναμική στρατηγική επιλογής k λημμάτων, η οποία βασίζεται στην κατανομή των scores ομοιότητας που παράγει ο Bi-Encoder. Αντί να χρησιμοποιεί έναν σταθερό αριθμό προκειμένων, το σύστημα αναλύει το «χάσμα εμπιστοσύνης» (confidence gap) μεταξύ των κορυφαίων αποτελεσμάτων. Αυτή η προσέγγιση διασφαλίζει ότι το context window του solver παραμένει καθαρό από «θόρυβο», επιτρέποντας στους αλγόριθμους ενοποίησης (unification algorithms) των ATPs να λειτουργούν με τη μέγιστη δυνατή ταχύτητα, γεγονός που οδήγησε σε σημαντική αύξηση των επιδόσεων σε benchmarks όπως το PISA και το MiniF2F.

3. ΤΟ FRAMEWORK LEAN-DOJO

3.1 Η Γλώσσα Lean

Η Lean είναι μια γλώσσα προγραμματισμού ανοιχτού κώδικα και ταυτόχρονα ένας διαδραστικός «αποδείκτης» θεωρημάτων (Interactive Theorem Prover - ITP), η οποία αναπτύχθηκε από τον Leonardo de Moura στη Microsoft Research. Στην τρέχουσα μορφή της (Lean 4), αποτελεί ένα πανίσχυρο εργαλείο που γεφυρώνει το χάσμα μεταξύ του καθαρού προγραμματισμού και της αυστηρής μαθηματικής απόδειξης.

Στον πυρήνα της, η Lean βασίζεται στον **Λογισμό των Επαγωγικών Κατασκευών (Calculus of Inductive Constructions)**. Αυτό σημαίνει ότι κάθε μαθηματική πρόταση αντιμετωπίζεται ως ένας "τύπος" (type) και κάθε απόδειξη ως ένα "πρόγραμμα" που υλοποιεί αυτόν τον τύπο (προσέγγιση Curry-Howard). Η Lean διαθέτει έναν εξαιρετικά μικρό και αξιόπιστο **πυρήνα (kernel)**, ο οποίος αναλαμβάνει τον τελικό έλεγχο κάθε απόδειξης. Αυτή η αρχιτεκτονική διασφαλίζει ότι, αν ο kernel εγκρίνει μια απόδειξη, αυτή είναι **αδιαμφισβήτητα ορθή**, αποκλείοντας κάθε πιθανότητα ανθρώπινου λογικού σφάλματος.

Πέρα από εργαλείο λογικής, η Lean είναι μια πλήρης, υψηλού επιπέδου γλώσσα συναρτησιακού προγραμματισμού (παρόμοια με τη Haskell). Το ιδιαίτερο χαρακτηριστικό της Lean 4 είναι ότι είναι "αυτο-φιλοξενούμενη" (self-hosted), δηλαδή γραμμένη στον εαυτό της. Αυτό επιτρέπει στους χρήστες να αναπτύσσουν δικές τους τακτικές (tactics), δηλαδή προγράμματα που αυτοματοποιούν την εύρεση αποδείξεων, μετατρέποντας τη διαδικασία της απόδειξης από μια χειροκίνητη παράθεση λογικών βημάτων σε μια διαδραστική εμπειρία προγραμματισμού.

Στο σημείο αυτό να αναφέρουμε ότι ένας από τους βασικούς πυλώνες της επιτυχίας της Lean είναι η **mathlib**. Πρόκειται για μια τεράστια, βιβλιοθήκη ανοιχτού κώδικα που περιλαμβάνει ένα μεγάλο μέρος των σύγχρονων μαθηματικών (άλγεβρα, ανάλυση, τοπολογία κ.α.) σε ψηφιοποιημένη μορφή. Η ύπαρξη της mathlib επιτρέπει στους ερευνητές να χτίζουν πάνω σε προ υπάρχοντα θεωρήματα, καθιστώντας τη Lean το "χρυσό πρότυπο" για την τυποποίηση σύνθετων μαθηματικών ερευνών, με την υποστήριξη κορυφαίων μαθηματικών όπως ο Terence Tao.

Σήμερα, η Lean βρίσκεται στο επίκεντρο της έρευνας για την Τεχνητή Νοημοσύνη. Λόγω της αυστηρής της δομής, παρέχει ένα ιδανικό περιβάλλον για την εκπαίδευση Μεγάλων Γλωσσικών Μοντέλων (LLMs). Σε αντίθεση με τη φυσική γλώσσα, όπου ένα μοντέλο μπορεί να "παραισθάνεται" (hallucinate), στη Lean το μοντέλο λαμβάνει άμεση και αντικειμενική ανατροφοδότηση (feedback) από τον compiler για το αν το βήμα που πρότεινε είναι σωστό. Έτσι, εργαλεία όπως το LeanDojo μετατρέπουν τη Lean σε ένα "γυμναστήριο" (gym) για την ανάπτυξη της επόμενης γενιάς του AI που θα μπορεί να «συλλογίζεται» με απόλυτη ακρίβεια.

Βασικές γνώσεις για την Lean

Για την κατανόηση της παρούσας πτυχιακής εργασίας χρειάζεται κανείς να γνωρίζει μερικές έννοιες και κάποια χαρακτηριστικά της Lean. Αυτές είναι οι:

- Proof State (Κατάσταση Απόδειξης): Στη Lean, μια απόδειξη δεν είναι ένα στατικό κείμενο, αλλά μια δυναμική διαδικασία. Σε κάθε χρονική στιγμή κατά την διάρκεια μιας απόδειξης, ο χρήστης (ή το μοντέλο Τεχνητής Νοημοσύνης) βρίσκεται σε ένα proof state. Αυτή η κατάσταση αποτελείται από το Local Context (Τοπικό Πλαίσιο) το οποίο περιέχει τις μεταβλητές και τις υποθέσεις που έχουν οριστεί μέχρι εκείνο το σημείο και τον Goal (Στόχος) ο οποίος είναι η μαθηματική πρόταση που απομένει να αποδειχθεί. Για παράδειγμα, αν θέλουμε να αποδείξουμε ότι για κάθε φυσικό αριθμό n, ισχύει n + 0 = n, τότε κατά την διάρκεια της απόδειξης στο «βήμα» της εφαρμογής της επαγωγής το Context μπορεί να περιέχει την υπόθεση h: k + 0 = k και ο στόχος μας θα είναι να δείξουμε ότι (k+1)+0=k+1.
- Tactics (Τακτικές): Οι «tactics» είναι οι εντολές που χρησιμοποιεί ο χρήστης για να μετασχηματίσει το proof state. Κάθε τακτική δέχεται ένα goal και επιστρέφει μηδέν, ένα ή περισσότερα νέα sub-goals. Για παράδειγμα υπάρχει η τακτική «rw [obj]» (rewrite) η

οποία βάσει του obj (το οποίο πρέπει να είναι κάτι που η Lean αναγνωρίζει ως σχέση ισότητας) αντικαθιστά μια έκφραση στον στόχο με μία ισοδύναμη. Μερικές ακόμα είναι η «induction n» που διασπά το πρόβλημα σε βάση και επαγωγικό βήμα (παράγει δύο νέα goals) και η «exact h» που ολοκληρώνει την απόδειξη αν η υπόθεση h ταυτίζεται ακριβώς με τον στόχο. Το LeanDojo εκπαιδεύει νευρωνικά δίκτυα (όπως ο Transformer) να λειτουργούν ως Tactic Predictors. Το μοντέλο δέχεται ως είσοδο το κείμενο του proof state και προβλέπει την επόμενη βέλτιστη τακτική.

- Premises (Προκείμενες): Στη Lean, premise είναι οποιοδήποτε ήδη αποδεδειγμένο θεώρημα, ορισμός ή αξίωμα που είναι αποθηκευμένο στη βιβλιοθήκη (συνήθως στη mathlib). Για παράδειγμα, στην «εντολή» rw [add_assoc] η τακτική είναι η "rw" (rewrite) και η προκείμενη (premise) είναι το «add_assoc».

Για την καλύτερη κατανόηση αυτών μπορούμε να εξετάσουμε το παρακάτω παράδειγμα:

Θα αποδείξουμε ότι για οποιουσδήποτε φυσικούς αριθμούς a, b, c, ισχύει η ιδιότητα:

$$(a + b) + c = a + (c + b)$$

Η απόδειξη σε γλώσσα Lean έχει ως εξής:

```
theorem add_comm_assoc (a b c : ℕ) : (a + b) + c = a + (c + b) :=
by
  rw [add_assoc]
  rw [add_comm b c]
```

Το LeanDojo "βλέπει" αυτή την απόδειξη ως μια ακολουθία από states και tactics. Ας δούμε τι συμβαίνει στο παρασκήνιο:

1. Αρχική Κατάσταση (Initial State)

Μόλις γράψουμε το by, η Lean ανοίγει ένα "παράθυρο" (proof state).

- Variables: a, b, c : ℕ
- Goal: ⊢ (a + b) + c = a + (c + b)

2. Πρώτο Βήμα: Η Τακτική rw [add_assoc]

Εδώ καλούμε την τακτική "rewrite" (rw) χρησιμοποιώντας το θεώρημα της προσεταιριστικής ιδιότητας (add_assoc).

- Τι κάνει: Αναδιατάσσει τις παρενθέσεις στο αριστερό μέλος.
- Νέο Goal: ⊢ a + (b + c) = a + (c + b)

(Στο σημείο αυτό, το LeanDojo θα έπρεπε να "μαντέψει" ότι το add_assoc είναι το κατάλληλο premise από τη βιβλιοθήκη.)

3. Δεύτερο Βήμα: Η Τακτική rw [add_comm b c]

Τώρα θέλουμε να αλλάξουμε τη σειρά των b και c μέσα στην παρένθεση.

- Τι κάνει: Εφαρμόζει την αντιμεταθετική ιδιότητα (add_comm) συγκεκριμένα στα b και c.
- Τελικό Goal: ⊢ a + (c + b) = a + (c + b)

4. Ολοκλήρωση (QED)

Επειδή η αριστερή πλευρά είναι πλέον ολόιδια με τη δεξιά, η Lean "κλείνει" αυτόματα τον στόχο. Το proof state είναι πλέον άδειο (Goals accomplished).

Επιπλέον, αν το ανωτέρω παράδειγμα χρησιμοποιηθεί για εκπαίδευση ενός μοντέλου εντός του framework LeanDojo, αυτό θα προσέφερε στο training set τα εξής παραδείγματα (examples):

Input: State: ⊢ (a + b) + c = a + (c + b) -> *Target Output:* rw [add_assoc]

Input: State: ⊢ a + (b + c) = a + (c + b) -> *Target Output:* rw [add_comm b c]

Τα παραδείγματα αυτά διαβάζονται ως «η είσοδος τάδε έχει ως σωστή έξοδο το τάδε». Με αυτόν τον τρόπο, το εκάστοτε μοντέλο μαθαίνει, ουσιαστικά, να αντιστοιχεί εισόδους σε εξόδους.

3.2 Εισαγωγή και Φιλοσοφία του LeanDojo

Η εμφάνιση του LeanDojo το 2023 σηματοδότησε μια θεμελιώδη μεταβολή στον τρόπο προσέγγισης του πεδίου (paradigm shift) της τυπικής επαλήθευσης. Μέχρι τότε, η αλληλεπίδραση των Μεγάλων Γλωσσικών Μοντέλων (LLMs) με διαδραστικούς αποδείκτες (interactive provers) θεωρημάτων, όπως η Lean 4, ήταν μια διαδικασία εξαιρετικά δύσκαμπτη, συχνά περιορισμένη σε στατικά σύνολα δεδομένων που δεν επέτρεπαν την πραγματική ανατροφοδότηση. Το LeanDojo σχεδιάστηκε ως μια open-source υποδομή που λειτουργεί ως «γέφυρα» επικοινωνίας, επιτρέποντας στα μοντέλα μηχανικής μάθησης όχι απλώς να παράγουν κείμενο που μοιάζει με κώδικα Lean, αλλά να αλληλεπιδρούν δυναμικά με τον compiler σε πραγματικό χρόνο. Αυτή η προσέγγιση επιλύει το πρόβλημα της «τυφλής» παραγωγής (generation), δίνοντας στο μοντέλο τη δυνατότητα να εκτελεί τακτικές, να παρατηρεί τις αλλαγές στο proof state και να διορθώνει τα σφάλματά του βάσει των μηνυμάτων του compiler.

Η κεντρική φιλοσοφία του LeanDojo εδράζεται στην έννοια της ενισχυτικής μάθησης (Reinforcement Learning) και της δυναμικής αλληλεπίδρασης με το αποδεικτικό περιβάλλον της Lean. Παρέχοντας ένα περιβάλλον που προσομοιάζει το "OpenAI Gym", το framework μετατρέπει την απόδειξη θεωρημάτων σε ένα παιχνίδι στρατηγικής όπου το εκάστοτε μοντέλο είναι ο παίκτης. Αυτό σημαίνει ότι αντί το μοντέλο να εκπαιδεύεται μόνο σε έτοιμες αποδείξεις γραμμένες από ανθρώπους, το LeanDojo του επιτρέπει να πειραματίζεται με εναλλακτικές αποδεικτικές διαδρομές που μπορεί να μην υπάρχουν στη βιβλιοθήκη Mathlib, η οποία αποτελεί την βασική πηγή άντλησης δεδομένων εκπαίδευσης. Αυτή η δυναμική αλληλεπίδραση είναι κρίσιμη, καθώς επιτρέπει στο σύστημα να αντιλαμβάνεται τη σημασιολογική βαρύτητα κάθε τακτικής (tactic) και να αναπτύσσει μια βαθύτερη κατανόηση των μαθηματικών δομών μέσω της συνεχούς δοκιμής και επαλήθευσης.

Επιπλέον, μια από τις πιο ρηξικέλευθες συνεισφορές του LeanDojo είναι η ικανότητά του να πραγματοποιεί αυτοματοποιημένη εξόρυξη δεδομένων από οποιοδήποτε αποθετήριο (repository) στο GitHub. Ενώ προγενέστερα συστήματα ήταν «εγκλωβισμένα» σε συγκεκριμένες εκδόσεις της Mathlib, το LeanDojo εισάγει την έννοια του "Repository-level retrieval". Αυτό σημαίνει ότι το μοντέλο μπορεί να αντλήσει γνώση, ορισμούς και προκείμενες από ένα τεράστιο και διαρκώς αναπτυσσόμενο οικοσύστημα μαθηματικών projects. Η δυνατότητα αυτή διασφαλίζει ότι το εκάστοτε μοντέλο εκπαιδεύεται σε μια τεράστια ποικιλία μαθηματικών συμβολισμών και πεδίων, καθιστώντας το ικανό να γενικεύει τη γνώση του ακόμα και σε εξειδικευμένα προβλήματα που δεν είχε συναντήσει κατά την αρχική φάση της εκπαίδευσής του.

Επιπρόσθετα, το LeanDojo δίνει λύση σε ένα από τα πιο δυσεπίλυτα προβλήματα της αυτόματης απόδειξης , το οποίο αφορά τη διαχείριση των προκειμένων (Premise Selection) σε περιβάλλοντα με τεράστιες βιβλιοθήκες. Η υποδομή δεν καταγράφει απλώς τις τακτικές, αλλά χρησιμοποιεί εξελιγμένους αλγόριθμους για να εξάγει το πλήρες γράφημα εξαρτήσεων (dependency graph) κάθε θεωρήματος. Μέσω της ανάλυσης των Abstract Syntax Trees (ASTs), το LeanDojo γνωρίζει ακριβώς ποια λήμματα ήταν ορατά και προσβάσιμα στο μοντέλο σε κάθε συγκεκριμένο proof state. Αυτός ο περιορισμός είναι θεμελιώδης, καθώς αποτρέπει τη διαρροή δεδομένων (data leakage) —τη χρήση δηλαδή πληροφοριών που ορίζονται μεταγενέστερα στην απόδειξη— διασφαλίζοντας έτσι τη λογική εγκυρότητα της διαδικασίας. Παράλληλα, η γνώση του ακριβούς εύρους ορατότητας (scope) επιτρέπει στο μοντέλο να φιλτράρει τον "θόρυβο" της βιβλιοθήκης και να εστιάζει αποκλειστικά σε έγκυρες μαθηματικές οντότητες, καθιστώντας την ανάκτηση των προκειμένων δραστικά πιο στοχευμένη και αποδοτική.

Ένας βασικός πυλώνας της φιλοσοφίας των δημιουργών του LeanDojo είναι η προσβασιμότητα και ο εκδημοκρατισμός της έρευνας στην Τεχνητή Νοημοσύνη. Χρησιμοποιώντας τεχνολογίες όπως το Docker και παρέχοντας μια πλήρη διεπαφή σε Python (Pythonic API), το

framework αίρει τα τεχνικά εμπόδια που παραδοσιακά απέτρεπαν τους ερευνητές της μηχανικής μάθησης από το να ασχοληθούν με τη Lean. Η συμβατότητα (interoperability) αυτή επιτρέπει την απρόσκοπτη ενσωμάτωση προηγμένων αρχιτεκτονικών, όπως οι Transformers και οι Graph Neural Networks, στο pipeline της αποδεικτικής διαδικασίας. Έτσι, η έρευνα μπορεί πλέον να εστιαστεί στη βελτίωση της «νοημοσύνης» του μοντέλου και της ακρίβειας της ανάκτησης, έχοντας ως δεδομένη μια σταθερή, βιομηχανικού επιπέδου υποδομή επικοινωνίας με τον αποδείκτη (prover).

Τελικά, το LeanDojo δεν αποτελεί απλώς ένα εργαλείο, αλλά έναν καταλύτη που ωθεί την έρευνα προς τη βαθιά σημασιολογική κατανόηση των μαθηματικών. Η φιλοσοφία του υποστηρίζει ότι η απόδειξη θεωρημάτων δεν αποτελεί ένα πρόβλημα απλής μετάφρασης κειμένου, αλλά ένα ζήτημα ανακάλυψης γνώσης μέσα σε έναν δομημένο χώρο. Παράλληλα, η ικανότητα του συστήματος να παρέχει πλούσιο πλαίσιο (context) για κάθε αποδεικτικό βήμα είναι αυτή που καθιστά δυνατή την εκπαίδευση εξελιγμένων μοντέλων, οι οποίοι μπορούν να πλοηγηθούν με ακρίβεια στις λεπτές εννοιολογικές "αποχρώσεις" των μαθηματικών ορισμών και θεωρημάτων.

3.3 Αρχιτεκτονική του Συστήματος (General Architecture)

Η αρχιτεκτονική του LeanDojo βασίζεται σε μια δομή πελάτη-εξυπηρετητή (client-server), όπου ο πυρήνας του συστήματος είναι ένας εξειδικευμένος Interaction Server. Ο server αυτός εκτελεί μια τροποποιημένη έκδοση του Lean 4 compiler στο παρασκήνιο, λειτουργώντας ως το «λειτουργικό σύστημα» της αποδεικτικής διαδικασίας. Σε αντίθεση με τις στατικές μεθόδους επεξεργασίας κειμένου, ο server διατηρεί στη μνήμη του την τρέχουσα κατάσταση της απόδειξης (proof state), επιτρέποντας στο σύστημα να δέχεται τακτικές (tactics), να τις επεξεργάζεται σε πραγματικό χρόνο και να επιστρέφει άμεσα το αποτέλεσμα της εκτέλεσής τους.

Για την απρόσκοπτη ενσωμάτωση με σύγχρονα πλαίσια μηχανικής μάθησης (όπως το PyTorch), το LeanDojo παρέχει ένα ολοκληρωμένο Python API. Αυτή η «γέφυρα» επικοινωνίας μετατρέπει τις σύνθετες εσωτερικές λειτουργίες του Lean compiler σε εύχρηστα αντικείμενα στην Python. Μέσω αυτού του API, ένας ερευνητής μπορεί να στείλει μια τακτική ως απλή συμβολοσειρά (string) και να λάβει πίσω μια δομημένη απάντηση που περιλαμβάνει τους νέους στόχους (goals) ή μηνύματα σφάλματος. Αυτή η δυνατότητα σύνδεσης είναι κρίσιμη, καθώς αίρει τα τεχνικά εμπόδια επικοινωνίας μεταξύ του μοντέλου και του αποδείκτη.

Μια από τις πιο σύνθετες πτυχές της αρχιτεκτονικής είναι το Data Extraction Pipeline, το οποίο αναλαμβάνει τη μετατροπή ολόκληρων αποθετηρίων (repositories) από το GitHub σε μορφή αξιοποιήσιμη από μοντέλα AI. Η διαδικασία ξεκινά με τη μεταγλώττιση (compilation) του κώδικα Lean, κατά την οποία το LeanDojo «παρεμβάλλεται» για να εξάγει πληροφορίες που συνήθως χάνονται, όπως οι ενδιάμεσες αποδεικτικές καταστάσεις (proof states) μεταξύ των εφαρμογών των τακτικών, οι ακριβείς αναφορές σε προκείμενες (fully qualified names), ίχνη (traces) που συνδέουν κάθε θεώρημα με τα αρχεία από τα οποία εισάγει (import) γνώση. Το αποτέλεσμα είναι ένα πλούσιο σύνολο δεδομένων που συνδέει κάθε βήμα μιας ανθρώπινης απόδειξης με το ακριβές μαθηματικό πλαίσιο στο οποίο πραγματοποιήθηκε.

Ένα ακόμα βασικό χαρακτηριστικό της αρχιτεκτονικής του LeanDojo είναι το Proof State Representation που υιοθετεί και που πρόκειται για τη διαδικασία κωδικοποίησης της τρέχουσας μαθηματικής κατάστασης σε μορφή δεδομένων. Κάθε κατάσταση αποτελείται από τον στόχο (goal) —δηλαδή αυτό που πρέπει να αποδειχθεί— και το τοπικό πλαίσιο (local context), το οποίο περιλαμβάνει όλες τις ενεργές υποθέσεις (hypotheses) και τις μεταβλητές. Το LeanDojo αναπαριστά αυτή την πληροφορία ως δομημένα αντικείμενα (συχνά σε μορφή JSON), διασφαλίζοντας ότι ο Bi-Encoder λαμβάνει μια «καθαρή» αλλά πλήρη εικόνα. Η ακρίβεια αυτής της αναπαράστασης είναι κρίσιμη, καθώς επιτρέπει στο μοντέλο να διακρίνει τις λεπτές διαφορές μεταξύ παρόμοιων αποδεικτικών σταδίων.

Επιπλέον, το LeanDojo έχει ένα Feedback Loop που είναι η διαδικασία κατά την οποία το σύστημα αξιολογεί τις προτάσεις του μοντέλου. Όταν το μοντέλο παράγει μια τακτική (π.χ. induction n), το LeanDojo την εκτελεί και επιστρέφει άμεσα μια απάντηση. Εάν η τακτική είναι έγκυρη, ο server παρέχει το επόμενο proof state. Εάν είναι λανθασμένη, επιστρέφει το συγκεκριμένο μήνυμα σφάλματος του compiler. Αυτή η αλληλεπίδραση επιτρέπει στο σύστημα να μαθαίνει όχι μόνο από τις επιτυχίες, αλλά και από τις αποτυχίες, χρησιμοποιώντας τα σφάλματα ως οδηγό για τη διόρθωση της στρατηγικής του σε πραγματικό χρόνο.

Τέλος, για να διασφαλιστεί η σταθερότητα και η αναπαραγωγή αποτελεσμάτων, η αρχιτεκτονική του LeanDojo βασίζεται σε Docker Containers. Κάθε περιβάλλον απόδειξης είναι απομονωμένο, περιλαμβάνοντας τη συγκεκριμένη έκδοση της Lean και των βιβλιοθηκών που απαιτούνται. Αυτό λύνει το πρόβλημα του "software rot", διασφαλίζοντας ότι μια απόδειξη που εκτελέστηκε σήμερα θα παραμείνει έγκυρη και στο μέλλον. Επιπλέον, η χρήση containers επιτρέπει την παράλληλη εκτέλεση χιλιάδων αποδεικτών (provers) σε υπολογιστικά νέφη, καθιστώντας την αρχιτεκτονική ικανή να υποστηρίξει εκπαίδευση μοντέλων σε βιομηχανική κλίμακα.

3.4 Το Σύνολο Δεδομένων LeanDojo Dataset

Το σύνολο δεδομένων που χρησιμοποιείται από τους συγγραφείς του LeanDojo εδράζεται στην αξιοποίηση της Mathlib4, της εκτενέστερης και πιο εξελιγμένης βιβλιοθήκης τυπικών μαθηματικών στον κόσμο. Το dataset δεν περιορίζεται σε τεχνητά παραδείγματα, αλλά αντλεί τη γνώση του από χιλιάδες αρχεία που έχουν γραφτεί από την παγκόσμια μαθηματική κοινότητα, καλύπτοντας ένα ευρύτατο φάσμα, από τη στοιχειώδη άλγεβρα έως την προχωρημένη θεωρία αριθμών και την τοπολογία. Πέρα από τη Mathlib4, το LeanDojo επεκτείνεται σε πολυάριθμα

ανεξάρτητα αποθετήρια (repositories) στο GitHub, συγκροτώντας ένα ετερογενές και δυναμικό σώμα δεδομένων που αντικατοπτρίζει τον τρόπο με τον οποίο αναπτύσσονται τα σύγχρονα μαθηματικά σε περιβάλλοντα τυπικής επαλήθευσης.

Επιπλέον, η διαδικασία συλλογής δεδομένων που υλοποιεί το LeanDojo μετατρέπει τη στατική γνώση των αποθετηρίων που βρίσκεται σε μορφή αρχείων Lean σε δυναμικά παραδείγματα εκπαίδευσης. Χρησιμοποιώντας τον Interaction Server, το LeanDojo «ανακατασκευάζει» τη διαδρομή κάθε ανθρώπινης απόδειξης, απομονώνοντας κάθε ενδιάμεσο βήμα, όπως έχουμε πει και παραπάνω. Κάθε τέτοιο βήμα συσχετίζει το proof state (την κατάσταση πριν την εφαρμογή μιας τακτικής) με το αντίστοιχο tactic (την ενέργεια) και τα premises (τις προκείμενες) που επέλεξε ο συγγραφέας της απόδειξης . Αυτή η λεπτομερής χαρτογράφηση επιτρέπει στα μοντέλα μηχανικής μάθησης να διδαχθούν όχι μόνο το τελικό αποτέλεσμα, αλλά και τη στρατηγική λήψης αποφάσεων που οδηγεί σε μια επιτυχημένη απόδειξη.

Τέλος, όσον αφορά το σύνολο δεδομένων του LeanDojo, είναι σημαντικό να αναφέρουμε και ότι ένα από τα σημαντικότερα χαρακτηριστικά του dataset είναι η ιδιότητα του Grounding. Αυτό σημαίνει ότι, σε αντίθεση με απλά σύνολα δεδομένων κειμένου, κάθε αναφορά σε λήμμα ή θεώρημα εντός των δεδομένων είναι άρρηκτα συνδεδεμένη με τον ακριβή ορισμό της στη βιβλιοθήκη. Αυτό επιτρέπει στα μοντέλα, και ιδιαίτερα στον Bi-Encoder, να μην αντιμετωπίζουν τα μαθηματικά ως μια ακολουθία από αφηρημένους χαρακτήρες, αλλά ως ένα συνεκτικό δίκτυο αλληλεξαρτώμενων εννοιών, όπου η σημασία κάθε όρου καθορίζεται από τη θέση του μέσα στο συνολικό γράφημα γνώσης. Περισσότερες λεπτομέρειες για το συγκεκριμένο σύνολο δεδομένων υπάρχουν στο σχετικό κεφάλαιο καθώς η παρούσα πτυχιακή εργασία χρησιμοποιεί το ίδιο σύνολο δεδομένων και κρίθηκε σκόπιμο αυτό να αναλυθεί σε δικό του κεφάλαιο.

3.5 Διαδικασία Ανάκτησης και Χρήση Προκειμένων

Η διαδικασία ανάκτησης (retrieval) αποτελεί το πρώτο και κρισιμότερο στάδιο στην αυτόματη απόδειξη θεωρημάτων, καθώς καλείται να διαχειριστεί τον τεράστιο όγκο της βιβλιοθήκης Mathlib4, η οποία περιλαμβάνει δεκάδες χιλιάδες ορισμούς και θεωρήματα. Λόγω των περιορισμών στο μέγεθος του παραθύρου εισόδου (context window) των σύγχρονων γλωσσικών μοντέλων, είναι πρακτικά αδύνατο να εισαχθεί ολόκληρη η βιβλιοθήκη ως πληροφορία σε κάθε αποδεικτικό βήμα. Το LeanDojo επιλύει αυτό το πρόβλημα υιοθετώντας μια αρχιτεκτονική **ανάκτησης προκειμένων (premise retrieval)**, η οποία λειτουργεί ως ένας ευφυής μηχανισμός φιλτραρίσματος. Ο μηχανισμός αυτός αναλαμβάνει να εντοπίσει ένα περιορισμένο αλλά εξαιρετικά σχετικό υποσύνολο υποψήφιων θεωρημάτων (προκειμένων), μειώνοντας δραστικά την υπολογιστική πολυπλοκότητα και επιτρέποντας στο μοντέλο παραγωγής να εστιάσει αποκλειστικά στις πιο υποσχόμενες λογικές διαδρομές.

Η καρδιά αυτής της ανάκτησης βασίζεται σε έναν **Retriever** (συνήθως αρχιτεκτονικής Bi-Encoder), ο οποίος χρησιμοποιεί προηγμένα νευρωνικά δίκτυα για να μετατρέψει τόσο το τρέχον **proof state** όσο και τις προκείμενες της βιβλιοθήκης σε πυκνά διανύσματα (**dense embeddings**) σε έναν κοινό λανθάνοντα χώρο. Η διαδικασία αυτή δεν περιορίζεται στην απλή σύγκριση λέξεων-κλειδιών, αλλά εστιάζει και στη **σημασιολογική εγγύτητα (semantic similarity)**. Χρησιμοποιώντας μετρικές όπως το εσωτερικό γινόμενο (dot product) ή τη συνημίτονη εγγύτητα (cosine similarity), το σύστημα βαθμολογεί κάθε προκείμενη βάσει του πόσο «κοντά» βρίσκεται μαθηματικά στον τρέχοντα στόχο. Αυτό επιτρέπει στο LeanDojo να ανακτά θεωρήματα που, αν και δεν μοιράζονται κοινή ορολογία με τον στόχο, κρίνονται ως δομικά και λογικά απαραίτητα για την ολοκλήρωση της απόδειξης.

Πέρα από τη γενική σημασιολογική συνάφεια, το LeanDojo εφαρμόζει παράλληλα μια στρατηγική «γνώσης γενικού πλαισίου» (context-awareness), λαμβάνοντας υπόψη την ιεραρχική δομή των αρχείων της Lean. Η διαδικασία αυτή δεν αντιμετωπίζει τη βιβλιοθήκη ως μια επίπεδη λίστα, αλλά λαμβάνει υπόψιν την **τοπική ορατότητα (local scope)**, δίνοντας προτεραιότητα σε προκείμενες που έχουν ήδη δηλωθεί στο ίδιο αρχείο ή σε αρχεία που έχουν «εισαχθεί» ρητά μέσω εντολών import. Ταυτόχρονα, το σύστημα ενσωματώνει στο διάνυσμα αναζήτησης τις τοπικές υποθέσεις (hypotheses) που έχουν παραχθεί από προηγούμενες τακτικές, διασφαλίζοντας ότι οι ανακτηθείσες προκείμενες είναι όχι μόνο μαθηματικά σχετικές, αλλά και άμεσα προσβάσιμες βάσει των αυστηρών κανόνων ορατότητας του compiler.

Αφού ο Retriever εντοπίσει τις k πιο σχετικές προκείμενες (π.χ. τις 100 επικρατέστερες), η διαδικασία περνά σε μια φάση λεπτομερούς κατάταξης (**re-ranking**). Σε αυτό το στάδιο, το μοντέλο παραγωγής (generator) αναλαμβάνει να αξιολογήσει τη λειτουργική συμβατότητα κάθε προκείμενης με τον συγκεκριμένο στόχο. Για παράδειγμα, αν ο στόχος απαιτεί μια αντικατάσταση

(rewrite), το σύστημα φιλτράρει τις προκείμενες που δεν εκφράζουν ισότητες ή ισοδυναμίες. Η ενδιάμεση αυτή αξιολόγηση είναι κρίσιμη, καθώς απομακρύνει τον «θόρυβο» —δηλαδή θεωρήματα που μπορεί να είναι σημασιολογικά παρεμφερή αλλά τεχνικά μη εφαρμόσιμα— διασφαλίζοντας ότι οι τελικές προτάσεις που θα φτάσουν στον αποδείκτη έχουν την υψηλότερη δυνατή πιθανότητα επιτυχίας.

Η όλη διαδικασία ολοκληρώνεται με τη σύνθεση της τακτικής, όπου οι επιλεγμένες προκείμενες ενσωματώνονται ως ορίσματα σε εντολές της Lean. Το μοντέλο συνδυάζει τη γνώση του για τη σύνταξη των τακτικών (όπως apply ή simp) με τα **πλήρως προσδιορισμένα ονόματα (fully qualified names)** των προκειμένων, αποτρέποντας ασάφειες που θα μπορούσαν να προκύψουν από συνώνυμα λήμματα σε διαφορετικά namespaces. Αν η επιλεγμένη προκείμενη ταιριάζει απόλυτα με τον τύπο του στόχου, ο compiler επικυρώνει το βήμα και η απόδειξη προχωρά, μετατρέποντας την επιτυχημένη ανάκτηση σε ουσιαστική μαθηματική πρόοδο.

Ωστόσο, η ανάκτηση στο LeanDojo δεν είναι μια στατική αναζήτηση, αλλά ένας δυναμικός και «αυτο-διορθούμενος» βρόχος μάθησης. Σε περίπτωση που μια ανακτηθείσα προκείμενη οδηγήσει σε αποτυχία εφαρμογής της (tactic error), ο Interaction Server επιστρέφει το ακριβές μήνυμα σφάλματος του compiler, το οποίο καταγράφεται αμέσως ως «αρνητικό παράδειγμα». Αυτή η πληροφορία είναι ανεκτίμητη για τη συνεχή βελτίωση του Bi-Encoder, καθώς τον διδάσκει να αναγνωρίζει τις λεπτές λεπτομέρειες (nuances) που καθιστούν μια προκείμενη ακατάλληλη παρά τη σημασιολογική της συνάφεια. Με αυτόν τον τρόπο, το σύστημα ακονίζει την ακρίβειά του με κάθε νέα απόπειρα, καθιστώντας τη διαδικασία ανάκτησης έναν εξελισσόμενο μηχανισμό «μαθηματικής ανακάλυψης».

3.6 Η «Γλώσσα» των Τακτικών (Tactics) και η Παραγωγή Αποδείξεων

Στο περιβάλλον της Lean 4, οι τακτικές (tactics) δεν αποτελούν απλώς εντολές, αλλά μια εξειδικευμένη «γλώσσα» μετασχηματισμού λογικών καταστάσεων. Η ενότητα αυτή εστιάζει στον τρόπο με τον οποίο το LeanDojo μοντελοποιεί την παραγωγή αποδείξεων ως μια διαδικασία διαδοχικών αποφάσεων, όπου κάθε τακτική λειτουργεί ως ένα προγραμματιστικό βήμα που τροποποιεί τον τρέχοντα στόχο. Σε αντίθεση με τις παραδοσιακές μεθόδους που αναζητούν ολόκληρους αποδεικτικούς όρους (proof terms), το LeanDojo αξιοποιεί την «υψηλού επιπέδου» σύνταξη των τακτικών, επιτρέποντας στα μοντέλα να αλληλεπιδρούν με τον αποδείκτη με έναν τρόπο που προσομοιάζει την ανθρώπινη μαθηματική συλλογιστική.

Η παραγωγή μιας απόδειξης ξεκινά με την ανάλυση του proof state από το μοντέλο παραγωγής (generator), το οποίο καλείται να συνθέσει την καταλληλότερη τακτική για ένα δεδομένο context της απόδειξης μια δεδομένη χρονική στιγμή. Η διαδικασία αυτή προσδιορίζεται από τη γραμματική και το συντακτικό της Lean, όπου το μοντέλο πρέπει να επιλέξει όχι μόνο τον τύπο της ενέργειας (π.χ. induction, rewrite, apply), αλλά και τις συγκεκριμένες παραμέτρους ή προκείμενες που απαιτούνται. Αυτή η «παραγωγή κειμένου» είναι στην πραγματικότητα μια σύνθετη λογική πράξη: το μοντέλο πρέπει να προβλέψει μια τακτική που θα είναι ταυτόχρονα συντακτικά ορθή και μαθηματικά αποτελεσματική για την εξέλιξη της απόδειξης.

Μια κρίσιμη λεπτομέρεια στην αρχιτεκτονική του LeanDojo είναι η διαχείριση των δομημένων τακτικών. Πολλές εντολές στη Lean 4 επιδέχονται σύνθετες παραμέτρους, όπως λίστες θεωρημάτων ή συγκεκριμένες θέσεις εφαρμογής (locations). Το σύστημα επιβάλλει μια αυστηρή αντιστοίχιση μεταξύ των ανακτηθέντων προκειμένων και της σύνταξης της τακτικής, διασφαλίζοντας ότι το μοντέλο ενσωματώνει τη γνώση του Retriever μέσα σε ένα λειτουργικό string εντολής. Για παράδειγμα, η παραγωγή της τακτικής rw [add_assoc, add_comm] απαιτεί από το μοντέλο να έχει κατανοήσει ότι η συγκεκριμένη σειρά εφαρμογής των θεωρημάτων θα οδηγήσει στην επιθυμητή απλοποίηση του στόχου.

Η παραγωγή των αποδείξεων δεν περιορίζεται σε μια γραμμική ακολουθία, αλλά εξελίσσεται μέσα από έναν δέντρο-ειδή χώρο αναζήτησης. Το LeanDojo υποστηρίζει εξελιγμένους αλγορίθμους αναζήτησης, όπως ο **Best-First Search**, όπου σε κάθε βήμα το μοντέλο μπορεί να παράγει πολλαπλές υποψήφιες τακτικές (candidates). Κάθε πρόταση βαθμολογείται με βάση την πιθανότητα επιτυχίας της και στη συνέχεια εκτελείται από τον Interaction Server. Αυτή η δυναμική αλληλεπίδραση επιτρέπει στο σύστημα να εξερευνά εναλλακτικά αποδεικτικά μονοπάτια, αναγνωρίζοντας πότε μια τακτική οδηγεί σε «αδιέξοδο» ή σε έναν στόχο που είναι πλέον αδύνατο να αποδειχθεί.

Η αποτελεσματικότητα της «γλώσσας» των τακτικών ενισχύεται σημαντικά από την άμεση

απόκριση στις προτεινόμενες τακτικές που παρέχει ο compiler. Όταν το μοντέλο παράγει μια τακτική, το LeanDojo δεν την αποδέχεται παθητικά, αλλά την επικυρώνει σε πραγματικό χρόνο. Εάν η τακτική «κλείσει» τον στόχο, η απόδειξη ολοκληρώνεται επιτυχώς. Εάν όμως η τακτική είναι ημιτελής (π.χ. δημιουργεί νέους υπο-στόχους), το σύστημα τροφοδοτεί τη νέα κατάσταση πίσω στο μοντέλο για το επόμενο βήμα. Αυτός ο διαδραστικός χαρακτήρας της παραγωγής αποδείξεων μετατρέπει τη στατική πρόβλεψη κειμένου σε μια ενεργή διαδικασία επίλυσης προβλημάτων, όπου η «γλώσσα» λειτουργεί ως το εργαλείο πλοήγησης μέσα στον μαθηματικό χώρο.

Τέλος, η αρχιτεκτονική «παραγωγής» του LeanDojo εξασφαλίζει τη γενίκευση σε νέα μαθηματικά πεδία μέσω της χρήσης ενός εμπλουτισμένου context. Το μοντέλο δεν μαθαίνει απλώς να επαναλαμβάνει τακτικές που είδε κατά την εκπαίδευση, αλλά αναπτύσσει μια βαθύτερη κατανόηση του πώς οι τακτικές αλληλεπιδρούν με τους τύπους (types) και τις ιδιότητες των αντικειμένων. Αυτή η ικανότητα «σύνθεσης» νέων αποδεικτικών στρατηγικών είναι που επιτρέπει στο LeanDojo να αντιμετωπίζει πρωτότυπα θεωρήματα, καθιστώντας τη γλώσσα των τακτικών ένα πανίσχυρο μέσο για τη γεφύρωση του χάσματος μεταξύ της ανθρώπινης διαίσθησης και της υπολογιστικής αυστηρότητας.

3.7 Λειτουργική Περιγραφή (Operation-wise Description)

Η λειτουργία του LeanDojo ολοκληρώνεται μέσα από μια σειρά διαδοχικών σταδίων που μετατρέπουν μια μαθηματική δήλωση σε μια πλήρως επαληθευμένη απόδειξη. Η διαδικασία εκκινεί με την εισαγωγή ενός θεωρήματος στον Interaction Server, ο οποίος προετοιμάζει το αρχικό proof state. Σε αυτό το πρώτο στάδιο, το σύστημα αναλύει τον στόχο και τις διαθέσιμες υποθέσεις, δημιουργώντας μια δομημένη αναπαράσταση που θα χρησιμεύσει ως είσοδος για το μοντέλο. Αυτό το αρχικό «στιγμιότυπο» της λογικής κατάστασης είναι κρίσιμο, καθώς καθορίζει ολόκληρο το πλαίσιο μέσα στο οποίο θα κινηθεί η αναζήτηση λύσης.

Μόλις οριστεί η κατάσταση, η σκυτάλη περνά στη διαδικασία ανάκτησης, όπου ο Retriever σαρώνει τη βιβλιοθήκη Mathlib4 για τον εντοπισμό σχετικών προκειμένων. Το σύστημα προσδιορίζει τις κορυφαίες k προκείμενες που παρουσιάζουν τη μεγαλύτερη σημασιολογική συνάφεια με τον τρέχοντα στόχο, λαμβάνοντας υπόψη τόσο τις παγκόσμιες βιβλιοθήκες όσο και το τοπικό πλαίσιο των εισαγόμενων αρχείων. Αυτό το υποσύνολο γνώσης «σερβίρεται» στο μοντέλο παραγωγής (generator), το οποίο καλείται να συνθέσει μια έγκυρη τακτική. Η σύνθεση αυτή δεν είναι τυχαία, αλλά εδράζεται στον συνδυασμό της μαθηματικής διαίσθησης του μοντέλου και των συγκεκριμένων προκειμένων που ανακτήθηκαν στο προηγούμενο βήμα.

Το επόμενο και πλέον κρίσιμο στάδιο είναι η εκτέλεση και επικύρωση της παραχθείσας τακτικής. Η προτεινόμενη εντολή αποστέλλεται πίσω στον Lean compiler μέσω του Python API, όπου ελέγχεται η συντακτική και λογική της ορθότητα. Σε αυτό το σημείο, η διαδικασία «διακλαδίζεται»: εάν η τακτική είναι επιτυχής, ο compiler επιστρέφει τη νέα, απλουστευμένη κατάσταση της απόδειξης· εάν αποτύχει, το σύστημα καταγράφει το σφάλμα και επιτρέπει στο μοντέλο να δοκιμάσει μια εναλλακτική προσέγγιση. Αυτός ο επαναληπτικός βρόχος συνεχίζεται μέχρις ότου ο στόχος «κλείσει» οριστικά, διασφαλίζοντας ότι κάθε βήμα της διαδρομής είναι αυστηρά επαληθευμένο.

Η λειτουργική αυτή ροή κορυφώνεται με την οριστική αποθήκευση της αποδεικτικής διαδρομής (proof trace). Το LeanDojo δεν κρατά μόνο το τελικό αποτέλεσμα, αλλά καταγράφει ολόκληρο το ιστορικό των αποφάσεων, των επιτυχημένων ανακτήσεων και των λανθασμένων δοκιμών. Αυτή η συνολική καταγραφή εξασφαλίζει την δυνατότητα ανασύνθεσης και την «επαληθευσιμότητα» της απόδειξης και παρέχει πολύτιμα δεδομένα για τη μελλοντική βελτίωση των μοντέλων. Με αυτόν τον τρόπο, η λειτουργία του LeanDojo υπερβαίνει τα όρια μιας απλής αυτοματοποίησης, αποτελώντας ένα ολοκληρωμένο οικοσύστημα όπου η ανάκτηση γνώσης και η παραγωγή λογικής συνεργάζονται αρμονικά υπό την αυστηρή εποπτεία ενός τυπικού αποδείκτη.

4. ΣΥΝΟΛΑ ΔΕΔΟΜΕΝΩΝ

Στο προηγούμενο κεφάλαιο αναφερθήκαμε στο σύνολο δεδομένων που χρησιμοποιεί το LeanDojo. Σε εκείνο το κεφάλαιο περιοριστήκαμε σε μια απλή αναφορά καθώς η παρούσα πτυχιακή χρησιμοποιεί το ίδιο σύνολο δεδομένων, οπότε σε αυτήν την ενότητα θα το αναλύσουμε εκτενέστερα.

4.1 Προέλευση και Συλλογή Δεδομένων (Data Sourcing)

Η συγκρότηση ενός αξιόπιστου και αντιπροσωπευτικού συνόλου δεδομένων για την εκπαίδευση μοντέλων αυτόματης απόδειξης εκκινεί από την αξιοποίηση της Mathlib4, της εκτενέστερης βιβλιοθήκης τυπικών μαθηματικών στον κόσμο. Η Mathlib4 δεν αποτελεί απλώς μια συλλογή αρχείων κειμένου, αλλά ένα δυναμικό και ιεραρχικά δομημένο αποθετήριο που καλύπτει το μεγαλύτερο μέρος των σύγχρονων μαθηματικών, από τη γραμμική άλγεβρα έως τη θεωρία κατηγοριών. Το LeanDojo αντλεί την «πρώτη ύλη» του από αυτό το οικοσύστημα, διασφαλίζοντας ότι το μοντέλο εκτίθεται σε μαθηματικές έννοιες υψηλού επιπέδου, οι οποίες έχουν ήδη επικυρωθεί από την παγκόσμια κοινότητα χρηστών της Lean 4.

Πέρα από την κεντρική βιβλιοθήκη, η διαδικασία συλλογής επεκτείνεται σε ένα ευρύ φάσμα ανεξάρτητων αποθετηρίων στο GitHub, τα οποία χρησιμοποιούν τη Lean 4 ως βασικό εργαλείο επαλήθευσης. Αυτή η πολυσυλλεκτική προσέγγιση είναι κρίσιμη για τη γενίκευση του συστήματος, καθώς εισάγει στο σύνολο δεδομένων διαφορετικά στυλ κωδικοποίησης, εξειδικευμένους ορισμούς και ποικίλες αποδεικτικές στρατηγικές. Με την ενσωμάτωση αυτών των εξωτερικών πηγών, το LeanDojo δημιουργεί ένα σώμα δεδομένων που δεν είναι στατικό, αλλά αντικατοπτρίζει την πραγματική και συνεχή εξέλιξη της τυπικής μαθηματικής έρευνας σε παγκόσμια κλίμακα.

Η μετατροπή αυτού του τεράστιου όγκου κώδικα σε αξιοποιήσιμα δεδομένα εκπαίδευσης επιτυγχάνεται μέσω της εξειδικευμένης διαδικασίας του tracing. Χρησιμοποιώντας τον Interaction Server, το LeanDojo «παρακολουθεί» την εκτέλεση κάθε αρχείου Lean σε πραγματικό χρόνο, καταγράφοντας κάθε σημείο όπου εφαρμόζεται μια τακτική. Η διαδικασία αυτή απομονώνει τα στιγμιότυπα της απόδειξης, επιτρέποντας την εξαγωγή του πλήρους proof state (τον στόχο και τις διαθέσιμες υποθέσεις) σε κάθε βήμα. Με αυτόν τον τρόπο, οι στατικές αποδείξεις μετασχηματίζονται σε μια ακολουθία λογικών μεταβάσεων, οι οποίες αποτελούν τη «διδακτική ύλη» για την εκπαίδευση του νευρωνικού δικτύου.

Ιδιαίτερη έμφαση δίνεται, επίσης, και στη διατήρηση της σημασιολογικής ακεραιότητας των δεδομένων κατά τη συλλογή. Το LeanDojo δεν αποθηκεύει μόνο το κείμενο των αποδείξεων, αλλά καταγράφει τις πλήρεις εξαρτήσεις (dependencies) κάθε θεωρήματος. Αυτό σημαίνει ότι για κάθε βήμα απόδειξης, το σύστημα γνωρίζει ακριβώς ποιες προκείμενες (premises) ήταν διαθέσιμες στον χρήστη τη συγκεκριμένη στιγμή, καθώς και ποιες από αυτές επιλέχθηκαν τελικά για την επίλυση του στόχου. Αυτή η λεπτομερής «ιχνηλάτηση» (tracing) μετατρέπει το σύνολο δεδομένων σε έναν πλούσιο γράφο γνώσης, όπου η πληροφορία είναι απόλυτα διασυνδεδεμένη και τεχνικά επαληθεύσιμη.

Τέλος, η στρατηγική συλλογής δεδομένων θωρακίζει το σύστημα έναντι του κινδύνου της «αποστήθισης» μέσω μιας προσεκτικής οργάνωσης των πηγών. Τα δεδομένα κατηγοριοποιούνται με βάση την προέλευσή τους, επιτρέποντας τη δημιουργία εξειδικευμένων συνόλων δοκιμής (test sets) που περιλαμβάνουν εντελώς νέα projects ή θεωρήματα που δεν υπήρχαν στο σύνολο εκπαίδευσης. Αυτή η μεθοδολογία διασφαλίζει ότι η αξιολόγηση του μοντέλου θα αντικατοπτρίζει την πραγματική του ικανότητα να παράγει πρωτότυπη μαθηματική σκέψη και να ανακτά προκείμενες σε άγνωστα περιβάλλοντα, καθιστώντας το dataset ένα αξιόπιστο εργαλείο για την προαγωγή της τεχνητής νοημοσύνης στα μαθηματικά.

4.2 Η Δομή των Αποθηκευμένων Δεδομένων (JSON Schema & Fields)

Η επιτυχία της εκπαίδευσης ενός μοντέλου βασίζεται πρωτίστως στην ποιότητα και την οργάνωση της πληροφορίας που δέχεται, γι' αυτό και μετά τη διαδικασία του tracing, το LeanDojo μετασχηματίζει τις αποδείξεις σε δομημένα αρχεία μορφής JSON. Αυτά τα αρχεία λειτουργούν ως ο απαραίτητος συνδετικός κρίκος μεταξύ του compiler της Lean και του νευρωνικού δικτύου, καθώς δεν περιέχουν απλό κείμενο, αλλά μια ιεραρχική αναπαράσταση των αποδεικτικών βημάτων. Κάθε αντικείμενο μέσα σε ένα αρχείο JSON αντιστοιχεί σε μια συγκεκριμένη χρονική στιγμή της απόδειξης, επιτρέποντας στο μοντέλο να «διαβάζει» την εξέλιξη της λογικής σκέψης

με έναν τρόπο που είναι υπολογιστικά διαχειρίσιμος και απόλυτα οργανωμένος.

Κεντρικό ρόλο σε κάθε εγγραφή του dataset παίζει το πεδίο του proof state, το οποίο συμπυκνώνει τον τρέχοντα στόχο (goal) και τις τοπικές υποθέσεις (hypotheses) που είναι διαθέσιμες στον χρήστη. Το LeanDojo διασφαλίζει ότι η πληροφορία αυτή συνοδεύεται από το απαραίτητο context, όπως το όνομα του αρχείου και τις βιβλιοθήκες που έχουν εισαχθεί, προσφέροντας έτσι μια ολοκληρωμένη εικόνα του αποδεικτικού περιβάλλοντος. Αυτή η λεπτομέρεια επιτρέπει στον Retriever να κατανοήσει όχι μόνο τι πρέπει να αποδειχθεί, αλλά και πού ακριβώς βρίσκεται το πρόβλημα μέσα στο ευρύτερο μαθηματικό οικοσύστημα, γεγονός που περιορίζει δραστικά τον χώρο αναζήτησης στις προκείμενες που είναι όντως προσβάσιμες.

Παράλληλα με την κατάσταση της απόδειξης, το dataset καταγράφει με ακρίβεια την τακτική (tactic) που εφαρμόστηκε από τον συγγραφέα της απόδειξης, καθώς και τα πλήρη ονόματα των προκειμένων (premises) που ανακλήθηκαν. Η πληροφορία αυτή αποθηκεύεται με τη μορφή «fully qualified names», γεγονός που εκμηδενίζει την πιθανότητα ασάφειας μεταξύ ομώνυμων θεωρημάτων που ανήκουν σε διαφορετικά namespaces. Για τον Bi-Encoder, αυτή η άμεση σύζευξη μεταξύ της κατάστασης και των χρησιμοποιούμενων προκειμένων αποτελεί το «χρυσό παράδειγμα» (ground truth), πάνω στο οποίο βασίζεται η εκπαίδευσή του για την αναγνώριση των καταλληλότερων θεωρημάτων σε μελλοντικές αποδείξεις.

Πέρα από τα καθαρά μαθηματικά δεδομένα, κάθε εγγραφή εμπλουτίζεται με μεταδεδομένα (metadata) που αφορούν τη θέση της τακτικής μέσα στον πηγαίο κώδικα, όπως ο αριθμός γραμμής και στήλης. Αυτή η «ιχνηλασιμότητα» (traceability) είναι ζωτικής σημασίας για τη διαδικασία του debugging και την οπτικοποίηση των αποτελεσμάτων, καθώς επιτρέπει στον ερευνητή να ανατρέξει στο αρχικό αρχείο Lean και να κατανοήσει το πλήρες σκεπτικό πίσω από μια συγκεκριμένη απόδειξη. Με αυτόν τον τρόπο, η εκπαίδευση του μοντέλου παραμένει στενά συνδεδεμένη με την πραγματική δομή του κώδικα, διασφαλίζοντας ότι η παραγόμενη γνώση δεν είναι αποκομμένη από το προγραμματιστικό της πλαίσιο.

Η λειτουργική ολοκλήρωση της δομής των δεδομένων επιτυγχάνεται στο τελικό στάδιο της κανονικοποίησης (normalization) των strings. Πριν τα δεδομένα διοχετευθούν στον tokenizer του μοντέλου, εφαρμόζονται τεχνικές καθαρισμού για την αφαίρεση περιττών σχολίων ή κενών διαστημάτων που δεν προσφέρουν λογική αξία στη διαδικασία της απόδειξης. Αυτή η ομοιόμορφη μορφοποίηση εγγυάται ότι το μοντέλο δεν θα επηρεαστεί από αισθητικές διαφοροποιήσεις στον τρόπο γραφής του κώδικα, αλλά θα επικεντρωθεί αποκλειστικά στην ουσία των μαθηματικών συμβόλων και των λογικών σχέσεων, ενισχύοντας έτσι τη σταθερότητα και την απόδοση της ανάκτησης.

4.3 Διαχείριση Προκειμένων στο Dataset (Premise Corpus)

Η αποτελεσματικότητα ενός συστήματος ανάκτησης δεν εξαρτάται μόνο από τον αλγόριθμο αναζήτησης, αλλά και από την οργάνωση του συνόλου των υποψήφιων στοιχείων, το οποίο ονομάζεται **Premise Corpus**. Στο LeanDojo, αυτό το σώμα προκειμένων αποτελείται από κάθε ορισμό, θεώρημα και λήμμα που είναι προσβάσιμο μέσα από τη Mathlib4 και τα συνδεδεμένα αποθετήρια. Η διαχείριση αυτού του τεράστιου όγκου πληροφορίας απαιτεί τη δημιουργία ενός μοναδικού καταλόγου, όπου κάθε προκείμενη αντιστοιχίζεται σε ένα σταθερό αναγνωριστικό. Αυτή η κεντρικοποιημένη (centralized) «δεξαμενή» γνώσης επιτρέπει στον Bi-Encoder να συγκρίνει το τρέχον proof state με ένα πεπερασμένο αλλά εξαιρετικά ευρύ σύνολο μαθηματικών προτάσεων, διασφαλίζοντας ότι καμία διαθέσιμη πληροφορία δεν αγνοείται κατά την αναζήτηση.

Για να μπορέσει το μοντέλο να επεξεργαστεί τις προκείμενες, κάθε εγγραφή στο corpus περιλαμβάνει όχι μόνο το όνομα του θεωρήματος, αλλά και το πλήρες περιεχόμενό του σε μορφή κώδικα (source code). Η συμπερίληψη του σώματος της προκείμενης είναι καθοριστική, καθώς επιτρέπει στον Premise Encoder να εξάγει σημασιολογικά χαρακτηριστικά από τη λογική δομή του θεωρήματος και όχι μόνο από τον τίτλο του. Με αυτόν τον τρόπο, το σύστημα χαρτογραφεί τις προκείμενες στον διανυσματικό χώρο βάσει της πραγματικής μαθηματικής τους ουσίας. Αυτή η διαδικασία μετατρέπει το Premise Corpus από μια απλή λίστα ονομάτων σε έναν πλούσιο σημασιολογικό κατάλογο, όπου παρεμφερή θεωρήματα (π.χ. ιδιότητες των ορίων) καταλαμβάνουν γειτονικές θέσεις στον χώρο των embeddings.

Μια ιδιαίτερη πρόκληση στη διαχείριση του corpus είναι η αντιμετώπιση της δυναμικής φύσης των προκειμένων. Στη Lean 4, η ορατότητα μιας προκείμενης εξαρτάται από τις εντολές import

και το συγκεκριμένο namespace στο οποίο βρίσκεται ο χρήστης. Το LeanDojo διαχειρίζεται αυτή την πολυπλοκότητα αποθηκεύοντας για κάθε προκείμενη την πλήρη διαδρομή της (module path). Κατά την εκπαίδευση και τη δοκιμή, το σύστημα είναι σε θέση να φιλτράρει το corpus, ώστε ο retriever να «βλέπει» μόνο τις προκείμενες που θα ήταν νομίμως διαθέσιμες σε έναν «άνθρωπο-αποδείκτη» (human-prover) στο συγκεκριμένο σημείο του κώδικα. Αυτή η προσέγγιση αποτρέπει τη χρήση «μελλοντικών» θεωρημάτων (data leakage), διασφαλίζοντας την ακεραιότητα της αποδεικτικής διαδικασίας.

Επιπλέον, το Premise Corpus υποστηρίζει τη διαφοροποίηση μεταξύ διαφορετικών τύπων προκειμένων, όπως θεωρήματα, ορισμοί, αξιώματα και τοπικές υποθέσεις. Το LeanDojo κατηγοριοποιεί αυτά τα στοιχεία, επιτρέποντας στο μοντέλο να αναπτύξει διαφορετικές στρατηγικές ανάκτησης ανάλογα με το αν ο στόχος απαιτεί την εφαρμογή ενός γνωστού λήμματος (apply) ή την απλοποίηση ενός ορισμού (unfold). Η κατηγοριοποιημένη αυτή οργάνωση ενισχύει την ακρίβεια του συστήματος, καθώς βοηθά τον Bi-Encoder να κατανοήσει τον λειτουργικό ρόλο που παίζει κάθε προκείμενη μέσα σε μια απόδειξη, μειώνοντας τις πιθανότητες ανάκτησης άσχετων ή μη λειτουργικών πληροφοριών.

Η λειτουργία του corpus ολοκληρώνεται με τον συνεχή συγχρονισμό του με τις εκδόσεις της Mathlib4. Καθώς η μαθηματική κοινότητα προσθέτει νέα θεωρήματα, το LeanDojo επιτρέπει την ανανέωση της δεξαμενής προκειμένων χωρίς να απαιτείται πλήρης επανεκπαίδευση του μοντέλου από την αρχή. Αυτή η επεκτασιμότητα είναι ζωτικής σημασίας, καθώς επιτρέπει στο σύστημα να παραμένει επίκαιρο και να ενσωματώνει τις τελευταίες εξελίξεις στα τυπικά μαθηματικά. Έτσι, το Premise Corpus δεν αποτελεί ένα στατικό σύνολο δεδομένων, αλλά ένα ζωντανό αποθετήριο γνώσης που εξελίσσεται παράλληλα με τη μαθηματική έρευνα, παρέχοντας στον Retriever τα απαραίτητα εφόδια για την αντιμετώπιση ολοένα και πιο σύνθετων προβλημάτων.

4.4 Διαχωρισμός Δεδομένων (Data Splitting & Splitting Strategies)

Η ορθή επιλογή στρατηγικής για τον διαχωρισμό των δεδομένων σε σύνολα εκπαίδευσης (train), επικύρωσης (val) και ελέγχου (test) είναι καθοριστική για την αξιοπιστία ενός μοντέλου αυτόματης απόδειξης. Στο LeanDojo, ο διαχωρισμός αυτός δεν πραγματοποιείται αποκλειστικά με τυχαίο τρόπο, καθώς κάτι τέτοιο θα οδηγούσε σε υπερεκτίμηση της απόδοσης λόγω της έντονης αλληλεξάρτησης των μαθηματικών θεωρημάτων. Αντίθετα, εφαρμόζονται πολλαπλές στρατηγικές διαχωρισμού, οι οποίες στοχεύουν στον έλεγχο της ικανότητας του μοντέλου να διαχειρίζεται διαφορετικά επίπεδα δυσκολίας και «άγνωστης» πληροφορίας, διασφαλίζοντας ότι η παραγόμενη γνώση είναι ανθεκτική και εφαρμόσιμη σε νέα προβλήματα.

Η πρώτη και πιο βασική προσέγγιση είναι το **Random Split**, όπου τα αποδεικτικά βήματα (τα οποία είναι και τα παραδείγματα βάσει των οποίων γίνεται η εκπαίδευση του μοντέλου) κατανέμονται τυχαία μεταξύ των συνόλων. Παρόλο που αυτή η μέθοδος παρέχει μια γενική εικόνα της απόδοσης, ενέχει τον κίνδυνο της «διαρροής δεδομένων» (data leakage), καθώς το μοντέλο μπορεί να έχει εκπαιδευτεί σε θεωρήματα που βρίσκονται στο ίδιο αρχείο με αυτά του test set. Για την αντιμετώπιση αυτού του προβλήματος, το LeanDojo εισάγει τον διαχωρισμό **Novel Premises Split**, ο οποίος αποτελεί μια πολύ πιο αυστηρή δοκιμασία. Σε αυτή την περίπτωση, το σύνολο ελέγχου περιλαμβάνει θεωρήματα που απαιτούν τη χρήση προκειμένων οι οποίες δεν εμφανίστηκαν ποτέ κατά τη διάρκεια της εκπαίδευσης. Αυτή η στρατηγική αναγκάζει τον Retriever να βασιστεί αποκλειστικά στη σημασιολογική κατανόηση του κώδικα και όχι σε απομνημονευμένα ονόματα θεωρημάτων.

Η κατηγοριοποιημένη αυτή προσέγγιση στον διαχωρισμό επιτρέπει στους ερευνητές να εντοπίσουν με ακρίβεια τα όρια του μοντέλου. Αναλύοντας την απόκλιση στην απόδοση μεταξύ του Random Split και του Novel Premises, μπορούμε να κατανοήσουμε σε ποιο βαθμό ο Bi-Encoder βασίζεται σε επιφανειακές συσχετίσεις ή σε βαθύτερες λογικές δομές. Η προσεκτική αυτή οργάνωση των δεδομένων θωρακίζει τη μεθοδολογία έναντι της υπερπροσαρμογής (overfitting) και παρέχει μια κλιμακωτή κλίμακα αξιολόγησης, η οποία ξεκινά από την απλή ανάκληση γνώσης και κορυφώνεται στην ανακάλυψη πρωτότυπων αποδεικτικών μονοπατιών σε άγνωστα μαθηματικά πεδία.

4.5 Προεπεξεργασία και Καθαρισμός (Data Preprocessing)

Η διαδικασία της προεπεξεργασίας αποτελεί έναν κρίσιμο ενδιάμεσο σταθμό, καθώς

διασφαλίζει ότι το μοντέλο θα επικεντρωθεί στη λογική δομή των μαθηματικών και όχι σε θόρυβο που προκύπτει από το στυλ γραφής του κώδικα. Το πρώτο στάδιο περιλαμβάνει τον **καθαρισμό του κειμένου (text sanitization)**, όπου αφαιρούνται συστηματικά τα σχόλια, τα περιττά κενά διαστήματα και οι ειδικοί χαρακτήρες μορφοποίησης που δεν φέρουν αποδεικτική αξία. Με αυτή την απλοποίηση, ο Bi-Encoder έρχεται αντιμέτωπος με μια ομοιόμορφη αναπαράσταση, γεγονός που μειώνει την πολυπλοκότητα του λεξιλογίου (vocabulary size) και επιτρέπει την ταχύτερη σύγκλιση κατά την εκπαίδευση.

Μια ιδιαίτερη τεχνική πρόκληση που αντιμετωπίζεται σε αυτό το στάδιο είναι η **διαχείριση του μεγέθους των καταστάσεων (state truncation)**. Στα τυπικά μαθηματικά, ένας στόχος (goal) μπορεί να συνοδεύεται από δεκάδες ή και εκατοντάδες υποθέσεις στο τοπικό context, με αποτέλεσμα το συνολικό μήκος του string να υπερβαίνει τα όρια εισόδου των Transformers (συνήθως 512 ή 1024 tokens). Το LeanDojo εφαρμόζει έξυπνες στρατηγικές περικοπής, δίνοντας προτεραιότητα στον ίδιο τον στόχο και στις πιο πρόσφατα ορισμένες υποθέσεις. Αυτή η ιεράρχηση εξασφαλίζει ότι οι πιο άμεσα σχετικές πληροφορίες παραμένουν εντός του παραθύρου προσοχής (attention window) του μοντέλου, αποφεύγοντας την απώλεια απαραίτητων δεδομένων (ο τρέχων στόχος, οι τοπικές υποθέσεις, οι τύποι των μεταβλητών) για τη λήψη αποφάσεων.

Επιπλέον, η προεπεξεργασία περιλαμβάνει την **κανονικοποίηση των προκειμένων (premise normalization)**. Κάθε θεώρημα που ανακτάται μετατρέπεται στη βασική του μορφή, αφαιρώντας τοπικές παραμετροποιήσεις που ενδέχεται να εισάγουν θόρυβο στην εκπαιδευτική διαδικασία και να μειώσουν την ακρίβεια του αλγορίθμου. Παράλληλα, το σύστημα φιλτράρει τις τακτικές που δεν μεταβάλλουν την κατάσταση της απόδειξης (non-productive tactics), ώστε το dataset να περιέχει μόνο ουσιαστικές λογικές μεταβάσεις. Αυτό το «φιλτράρισμα ποιότητας» εγγυάται ότι το μοντέλο θα διδαχθεί μόνο αποτελεσματικές αποδεικτικές διαδρομές, αποτρέποντας την εκμάθηση περιττών βημάτων ή βημάτων που θα οδηγούσαν στην εμφάνιση κύκλων τα οποία θα μείωναν την αποδοτικότητά του σε πραγματικές συνθήκες.

Τέλος, ένα σημαντικό κομμάτι της προεπεξεργασίας αφορά την **κωδικοποίηση των ονομάτων (identifier encoding)**. Το LeanDojo χρησιμοποιεί τεχνικές sub-word tokenization, οι οποίες επιτρέπουν στο μοντέλο να αναλύει σύνθετα ονόματα θεωρημάτων στα συνθετικά τους μέρη (π.χ. το Nat.add_comm αναλύεται σε Nat, add και comm). Αυτή η προσέγγιση επιτρέπει στον Bi-Encoder να αναπτύξει μια μορφολογική κατανόηση της βιβλιοθήκης Mathlib, δίνοντάς του τη δυνατότητα να προβλέπει τη συνάφεια προκειμένων ακόμα και αν δεν έχει δει το πλήρες όνομά τους στο παρελθόν, αρκεί να αναγνωρίζει τα επιμέρους μαθηματικά συστατικά τους.

4.6 Προεπεξεργασία και Καθαρισμός (Data Preprocessing)

Η ποσοτική ανάλυση του συνόλου δεδομένων αναδεικνύει το εύρος και την πολυπλοκότητα των τυπικών μαθηματικών που περιλαμβάνονται στο LeanDojo. Το Premise Corpus, η κεντρική «δεξαμενή» από την οποία αντλεί πληροφορίες ο Retriever, περιλαμβάνει **εκατοντάδες χιλιάδες μοναδικές προκείμενες**, οι οποίες καλύπτουν το σύνολο της βιβλιοθήκης Mathlib4 και πολυάριθμα εξωτερικά αποθετήρια (repositories). Αυτός ο τεράστιος όγκος δεδομένων διασφαλίζει ότι το μοντέλο εκτίθεται σε μια εξαιρετικά πλούσια ποικιλία μαθηματικών δομών, από απλές αριθμητικές πράξεις έως σύνθετα θεωρήματα της σύγχρονης ανάλυσης και άλγεβρας, καθιστώντας το dataset ένα από τα πληρέστερα στον τομέα της αυτόματης απόδειξης.

Πέρα από το πλήθος των προκειμένων, η ανάλυση των αποδεικτικών βημάτων (proof steps) αποκαλύπτει τη βάση πάνω στην οποία εκπαιδεύεται ο Bi-Encoder. Κάθε θεώρημα αποτελείται κατά μέσο όρο από μια σειρά τακτικών, δημιουργώντας ένα σύνολο δεδομένων με εκατομμύρια ζεύγη «Κατάστασης-Τακτικής» (State-Tactic pairs). Η στατιστική κατανομή αυτών των βημάτων δείχνει ότι, ενώ οι περισσότερες αποδείξεις ολοκληρώνονται σε λίγα βήματα, υπάρχει ένας σημαντικός αριθμός σύνθετων θεωρημάτων που απαιτούν δεκάδες αλληλένδετες τακτικές. Αυτή η διαβάθμιση δυσκολίας επιτρέπει στο μοντέλο να αναπτύξει σταδιακά την ικανότητα ανάκτησης, ξεκινώντας από προφανή λήμματα και καταλήγοντας σε εξειδικευμένες προκείμενες που απαιτούν βαθύτερη λογική συσχέτιση.

Ιδιαίτερο ενδιαφέρον παρουσιάζει η συχνότητα χρήσης των προκειμένων, η οποία ακολουθεί μια κατανομή «μακράς ουράς» (long-tail distribution). Ένας μικρός αριθμός βασικών θεωρημάτων εμφανίζεται σε πολύ μεγάλο ποσοστό των αποδείξεων, ενώ η πλειονότητα των προκειμένων χρησιμοποιείται σπάνια και σε πολύ εξειδικευμένα πλαίσια. Η συγκεκριμένη στατιστική ιδιαιτερότητα αποτελεί τη βασική πρόκληση για τον Bi-Encoder, καθώς καλείται να διατηρήσει την ακρίβειά του όχι μόνο στις κοινές προκείμενες, αλλά και στην αναγνώριση των σπάνιων θεωρημάτων που είναι απαραίτητα για την επίλυση προχωρημένων μαθηματικών προβλημάτων.

Τέλος, ο διαχωρισμός των δεδομένων (data splitting) παράγει συγκεκριμένα υποσύνολα που επιτρέπουν την αντικειμενική αξιολόγηση της ικανότητας γενίκευσης του μοντέλου. Τα στατιστικά στοιχεία του Novel Premises split δείχνουν ότι ένα σημαντικό ποσοστό των θεωρημάτων στο test set απαιτεί προκείμενες που δεν έχουν χρησιμοποιηθεί ποτέ στο σύνολο εκπαίδευσης. Αυτή η ποσοτική διαφορά επιβεβαιώνει ότι η αξιολόγηση δεν βασίζεται στην απλή ανάκληση αποθηκευμένης γνώσης, αλλά στην ικανότητα του μοντέλου να πλοηγείται σε έναν διαρκώς επεκτεινόμενο «μαθηματικό» χώρο.

5. Η ΡΟΗ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΥΛΟΠΟΙΗΣΗΣ

Στο παρόν κεφάλαιο θα παρουσιαστεί λεπτομερώς η εναλλακτική αρχιτεκτονική που προτείνεται στην παρούσα πτυχιακή εργασία. Αυτή περιλαμβάνει, ουσιαστικά, την χρήση διαφορετικού encoder για την παραγωγή των embeddings των premises και διαφορετικού για αυτήν των states, σχηματίζοντας έτσι μια εναλλακτική αρχιτεκτονική για έναν retriever. Ο συγκεκριμένος ασύμμετρος bi-encoder retriever κατόπιν αντικαθιστά τον αντίστοιχο του LeanDojo και έτσι μπορούμε να συγκρίνουμε αυστηρά τις δυο διαφορετικές αρχιτεκτονικές για έναν retriever.

5.1 Θεωρητικό Υπόβαθρο των Bi-Encoders και των Embeddings

Η θεμελιώδης αρχή της σύγχρονης ανάκτησης πληροφοριών (Information Retrieval) βασίζεται στη μετατροπή ακατέργαστων συμβολοσειρών κειμένου σε αριθμητικά διανύσματα εντός ενός πολυδιάστατου χώρου, μια διαδικασία γνωστή ως «παραγωγή» **embeddings**. Στο πλαίσιο των γλωσσικών μοντέλων, τα embeddings δεν αποτελούν απλές κωδικοποιήσεις λέξεων, αλλά πυκνές (dense) αναπαραστάσεις που περικλείουν μέσα τους τη σημασιολογική και συντακτική ουσία των μαθηματικών εκφράσεων. Χρησιμοποιώντας αρχιτεκτονικές βασισμένες σε Transformers, το σύστημα είναι σε θέση να αντιληφθεί ότι δύο θεωρήματα που χρησιμοποιούν διαφορετική ορολογία αλλά περιγράφουν την ίδια λογική ιδιότητα (π.χ. την αντιμεταθετικότητα), πρέπει να βρίσκονται σε κοντινή απόσταση στον διανυσματικό αυτό χώρο. Αυτή η ικανότητα «νοηματικής» τοποθέτησης των διανυσμάτων στον χώρο αποτελεί την καρδιά του retriever, καθώς μετατρέπει το πρόβλημα της εύρεσης προκειμένων από μια απλή αναζήτηση λέξεων-κλειδιών σε ένα πρόβλημα γεωμετρικής γειτνίασης.

Η αρχιτεκτονική του **Bi-Encoder** αποτελεί το πλέον διαδεδομένο πρότυπο για την υλοποίηση τέτοιων συστημάτων ανάκτησης σε μεγάλες κλίμακες δεδομένων. Σε αντίθεση με τους Cross-Encoders, οι οποίοι επεξεργάζονται ταυτόχρονα το ερώτημα και το έγγραφο (μια διαδικασία εξαιρετικά δαπανηρή υπολογιστικά), ο Bi-Encoder χρησιμοποιεί δύο διακριτούς μηχανισμούς κωδικοποίησης για να παράγει ανεξάρτητα embeddings για την κατάσταση της απόδειξης και τις διαθέσιμες προκείμενες. Η διαφοροποίηση αυτή είναι κρίσιμη, καθώς επιτρέπει τον υπολογισμό των διανυσμάτων εκ των προτέρων για ολόκληρη τη βιβλιοθήκη Mathlib4. Με αυτόν τον τρόπο, όταν το μοντέλο καλείται να προτείνει ένα θεώρημα, δεν χρειάζεται να επανεκτελέσει το νευρωνικό δίκτυο για κάθε μία από τις εκατοντάδες χιλιάδες προκείμενες, αλλά απλώς να αναζητήσει τα ήδη αποθηκευμένα διανύσματα που παρουσιάζουν τη μεγαλύτερη συνάφεια με το τρέχον state.

Η μέτρηση αυτής της συνάφειας επιτυγχάνεται μέσω μαθηματικών συναρτήσεων ομοιότητας, με επικρατέστερη το **συνημίτονο γωνίας (Cosine Similarity)** ή το εσωτερικό γινόμενο (Dot Product). Όταν τα διανύσματα δύο προτάσεων δείχνουν προς την ίδια κατεύθυνση στον πολυδιάστατο χώρο, η τιμή της ομοιότητας προσεγγίζει τη μονάδα, υποδηλώνοντας υψηλή συνάφεια. Στην περίπτωση του retriever, ο στόχος είναι η εκπαίδευση των encoders με τέτοιο τρόπο ώστε το διανυσματικό αποτύπωμα του proof state να «ευθυγραμμίζεται» με το αποτύπωμα της προκείμενης που όντως οδηγεί στη λύση. Αυτή η γεωμετρική προσέγγιση επιτρέπει στο σύστημα να διαχειρίζεται την τεράστια πολυπλοκότητα των μαθηματικών βιβλιοθηκών, παρέχοντας μια κλιμακούμενη λύση που μπορεί να λειτουργεί σε πραγματικό χρόνο κατά τη διάρκεια της αποδεικτικής διαδικασίας, προσφέροντας στον χρήστη ή στο μοντέλο παραγωγής (generator) τις πιο σωστές επιλογές προκειμένων.

Τέλος, η επιτυχία της συγκεκριμένης θεωρητικής προσέγγισης εξαρτάται άμεσα από την ικανότητα του μοντέλου να διατηρεί τη **σημασιολογική συνοχή** μεταξύ διαφορετικών επιπέδων αφαίρεσης. Οι encoders πρέπει να είναι σε θέση να «συμπιέζουν» την πληροφορία ενός ολόκληρου proof state, το οποίο μπορεί να περιέχει πολλαπλές υποθέσεις και σύνθετους στόχους, σε ένα διάνυσμα σταθερού μεγέθους, χωρίς να χάνεται η κρίσιμη λεπτομέρεια που καθιστά μια προκείμενη εφαρμόσιμη. Αυτή η διαδικασία της «σημασιολογικής» συμπίεσης πληροφορίας αποτελεί την κύρια πρόκληση στην εκπαίδευση του Bi-Encoder. Μέσω της χρήσης μεγάλων προ-εκπαιδευμένων μοντέλων και της «εξειδικευμένης προσαρμογής» (fine-tuning) στα δεδομένα της Lean 4, το σύστημα μαθαίνει να αναγνωρίζει μοτίβα και λογικές δομές, μετατρέποντας τον άψυχο κώδικα σε έναν ζωντανό χάρτη μαθηματικής γνώσης όπου η ανακάλυψη νέων αποδείξεων γίνεται πλέον ζήτημα σωστής διανυσματικής πλοήγησης.

5.2 Η Συμμετρική Αρχιτεκτονική (Baseline)

Η συμμετρική αρχιτεκτονική αποτελεί την καθιερωμένη προσέγγιση στα περισσότερα συστήματα ανάκτησης που βασίζονται σε Bi-Encoders, υιοθετώντας τη λογική των **κοινών βαρών (shared weights)** μεταξύ των δύο κλάδων του δικτύου. Σε αυτό το μοντέλο, ένας μοναδικός encoder αναλαμβάνει να επεξεργαστεί τόσο το ερώτημα (query) (εδώ, proof state) όσο και το έγγραφο-στόχο (εδώ, premise), επιβάλλοντας στο σύστημα να χρησιμοποιεί τις ίδιες ακριβώς παραμέτρους για την εξαγωγή χαρακτηριστικών και από τα δύο. Η επιλογή αυτή πηγάζει από την παραδοχή ότι, αφού τα δεδομένα προέρχονται από την ίδια γλώσσα προγραμματισμού και το ίδιο μαθηματικό πλαίσιο, η βέλτιστη διανυσματική αναπαράσταση μπορεί να επιτευχθεί μέσω ενός ενιαίου σημασιολογικού πρίσματος, η οποία απλοποιεί την εκπαιδευτική διαδικασία και μειώνει τον αριθμό των παραμέτρων που πρέπει να βελτιστοποιηθούν.

Στο πλαίσιο του LeanDojo, η συμμετρική αυτή προσέγγιση εξυπηρετεί τη δημιουργία ενός «οικουμενικού» διανυσματικού χώρου, όπου κάθε μαθηματική οντότητα, ανεξάρτητα από τον ρόλο της, καταλαμβάνει μια θέση βάσει του πηγαίου κώδικα Lean που την ορίζει. Χρησιμοποιώντας έναν κοινό encoder, το σύστημα διασφαλίζει ότι αν ένας στόχος και μια προκείμενη είναι ταυτόσημοι σε επίπεδο κειμένου, τα embeddings τους θα συμπίπτουν απόλυτα. Αυτή η ιδιότητα είναι ιδιαίτερα χρήσιμη για απλές περιπτώσεις ανάκτησης, όπου η λύση είναι μια άμεση εφαρμογή ενός θεωρήματος που μοιάζει συντακτικά με τον στόχο. Ωστόσο, η συμμετρία αυτή λειτουργεί συχνά ως ένας «στενός κορσές», καθώς αναγκάζει το μοντέλο να αγνοήσει το γεγονός ότι ο τρόπος που διατυπώνεται ένας στόχος σε μια ενεργή απόδειξη διαφέρει ριζικά από τον τρόπο που ορίζεται ένα αυτοτελές θεώρημα στη βιβλιοθήκη Mathlib4.

Η κύρια αδυναμία της συμμετρικής αρχιτεκτονικής εντοπίζεται στην **πληροφοριακή ασυμμετρία** που χαρακτηρίζει τα δεδομένα της Lean 4. Ένα proof state είναι μια δυναμική και συχνά χαοτική δομή, η οποία περιλαμβάνει πολλαπλά contexts, ονόματα τοπικών μεταβλητών και ενδιάμεσα βήματα που δεν υπάρχουν στους επίσημους ορισμούς της βιβλιοθήκης. Από την άλλη πλευρά, οι προκείμενες είναι στατικά, καλογραμμένα και αυστηρά δομημένα κομμάτια γνώσης. Επιβάλλοντας έναν κοινό encoder, το baseline μοντέλο του LeanDojo υποχρεώνεται να βρει έναν «μέσο όρο» αναπαράστασης που να ταιριάζει και στα δύο, με κίνδυνο να χάσει τις λεπτές αποχρώσεις που καθιστούν μια προκείμενη ιδανική για ένα συγκεκριμένο, πολύπλοκο state. Αυτή η υπολογιστική δυσκαμψία μπορεί να οδηγήσει σε λανθασμένες ανακτήσεις, ειδικά όταν η σχέση μεταξύ στόχου και προκείμενης είναι βαθιά σημασιολογική και όχι απλά επιφανειακή-συντακτική.

Τέλος, η συμμετρική υλοποίηση λειτουργεί ως το απαραίτητο μέτρο σύγκρισης για κάθε νέα πρόταση, καθώς οριοθετεί τις δυνατότητες των μοντέλων που βασίζονται στην απλή σύζευξη ομοειδών αναπαραστάσεων. Στην παρούσα εργασία, το baseline μοντέλο χρησιμοποιείται για να προσπαθήσει να αναδείξει πού ακριβώς αποτυγχάνει η ενιαία κωδικοποίηση και πώς η έλλειψη εξειδίκευσης ενδεχομένως περιορίζει την απόδοση του retriever σε «δύσκολα» σενάρια, όπως το novel premises split. Η ανάλυση της συμμετρικής αυτής αρχιτεκτονικής δεν αποσκοπεί μόνο στην παρουσίαση μιας υπάρχουσας λύσης, αλλά στη θεμελίωση της ανάγκης για μια πιο ευέλικτη δομή. Κατανοώντας τους περιορισμούς των κοινών βαρών, μπορούμε να εκτιμήσουμε την αξία της μετάβασης σε ένα ασύμμετρο σχήμα, το οποίο θα επιτρέπει στο σύστημα να «βλέπει» τον στόχο και την προκείμενη μέσα από δύο διαφορετικούς, αλλά απόλυτα συντονισμένους, «φακούς».

5.3 Προτεινόμενη Ασύμμετρη Αρχιτεκτονική (Asymmetric Bi-Encoder)

Η κεντρική «καινοτομία» της παρούσας εργασίας έγκειται στην εισαγωγή μιας ασύμμετρης αρχιτεκτονικής, η οποία έρχεται να ανατρέψει την παραδοσιακή στρατηγική των κοινών βαρών. Αντί να χρησιμοποιούμε έναν ενιαίο encoder, επιστρατεύουμε δύο διαφορετικά νευρωνικά δίκτυα, το καθένα εκ των οποίων διαθέτει τις δικές του ανεξάρτητες παραμέτρους. Αυτός ο διαχωρισμός επιτρέπει στο σύστημα να αναπτύξει εξειδικευμένες στρατηγικές επεξεργασίας για τα δύο σκέλη της ανάκτησης: τον State Encoder και τον Premise Encoder. Με αυτόν τον τρόπο, το μοντέλο αποκτά τη δυνατότητα να «μάθει» τις ιδιαιτερότητες κάθε πλευράς χωρίς οι βελτιστοποιήσεις της μιας να παρεμβαίνουν στην απόδοση της άλλης, προσφέροντας μια πολύ πιο ευέλικτη προσέγγιση στη διανυσματική αναπαράσταση των μαθηματικών.

Ο State Encoder σχεδιάστηκε με γνώμονα τη διαχείριση της δομικής πολυπλοκότητας που χαρακτηρίζει ένα «ενεργό» proof state. Οι καταστάσεις αυτές δεν είναι απλές μαθηματικές προτάσεις, αλλά σύνθετα contexts που περιλαμβάνουν στόχους και υποστόχους, τοπικές μεταβλητές και ενδεχομένως έναν μεγάλο αριθμό υποθέσεων που συχνά εκτείνονται σε πολλές γραμμές κώδικα. Η χρήση ενός αποκλειστικού encoder για αυτή τη λειτουργία επιτρέπει στο μοντέλο να εστιάζει στη δυναμική φύση της απόδειξης, μαθαίνοντας να ξεχωρίζει ποιες υποθέσεις είναι κρίσιμες για το επόμενο βήμα και ποιες αποτελούν δευτερεύουσες πληροφορίες. Αυτή η εξειδίκευση είναι δυσκολότερη ή αδύνατη σε συμμετρικά μοντέλα, όπου ο encoder αναγκάζεται να συμβιβάσει-εξισορροπήσει την «κατανόηση» που έχει για ένα μεταβαλλόμενο state με την «κατανόηση» για ένα στατικό θεώρημα.

Από την άλλη πλευρά, ο Premise Encoder είναι βελτιστοποιημένος για την κωδικοποίηση της στατικής γνώσης που βρίσκεται αποθηκευμένη στη βιβλιοθήκη Mathlib4. Οι προκείμενες (θεωρήματα και ορισμοί) έχουν μια πολύ πιο σταθερή και τυποποιημένη μορφή, η οποία απαιτεί μια διαφορετικού τύπου σημασιολογική ανάλυση. Ο Premise Encoder εκπαιδεύεται να αναγνωρίζει τη μαθηματική ουσία των θεωρημάτων μέσα από τη σύνταξή τους, τοποθετώντας τα στον διανυσματικό χώρο με τρόπο που να αναδεικνύει τις μεταξύ τους λογικές συγγένειες. Καθώς δεν επιβαρύνεται με την επεξεργασία των proof states, μπορεί να αφιερώσει όλο το παραμετρικό του εύρος στην «κατάκτηση» της ορολογίας και της δομής των βιβλιοθηκών, λειτουργώντας ως ένας ψηφιακός ταξινομητής που οργανώνει τη γνώση με απόλυτη ακρίβεια.

Η συνεργατική λειτουργία αυτών των δύο εξειδικευμένων νευρωνικών δικτύων επιτυγχάνει μια ακριβέστερη αντιστοίχιση state με premise στον κοινό διανυσματικό χώρο. Παρόλο που οι encoders λειτουργούν ανεξάρτητα, εκπαιδεύονται ώστε τα τελικά τους embeddings να «ευθυγραμμίζονται» αποτελεσματικά μέσω της συνάρτησης ομοιότητας. Η ασυμμετρία αυτή επιτρέπει στον State Encoder να αναζητά πληροφορία με τα δικά του σημασιολογικά κριτήρια και στον Premise Encoder να κωδικοποιεί τη γνώση με τα δικά του, ενώ η σύγκλιση επιτυγχάνεται μέσω της κοινής διαδικασίας εκπαίδευσης. Η συγκεκριμένη προσέγγιση όχι μόνο στοχεύει στη βελτίωση της ακρίβειας ανάκτησης (Retrieval Accuracy), αλλά ενδέχεται να επιτρέπει την καλύτερη κατανόηση της λειτουργικής διαφοροποίησης μεταξύ των δύο encoders, καθώς μας επιτρέπει να αναλύσουμε πώς κάθε encoder αντιλαμβάνεται διαφορετικά το ίδιο μαθηματικό περιεχόμενο ανάλογα με τον λειτουργικό του ρόλο.

5.4 Διαδικασία Εκπαίδευσης (Training Pipeline)

Η εκπαίδευση της ασύμμετρης αρχιτεκτονικής αποτελεί το πιο κρίσιμο στάδιο της υλοποίησης, καθώς είναι η φάση όπου οι δύο ανεξάρτητοι encoders μαθαίνουν να συντονίζονται σε έναν κοινό διανυσματικό χώρο. Η διαδικασία αυτή βασίζεται στη μεθοδολογία της Αντιπαραθετικής Μάθησης (Contrastive Learning), η οποία στοχεύει στη μεγιστοποίηση της ομοιότητας μεταξύ των «θετικών» ζευγών και στην ελαχιστοποίηση της ομοιότητας μεταξύ των «αρνητικών». Στην περίπτωσή μας, ένα θετικό ζεύγος αποτελείται από ένα συγκεκριμένο proof state και την προκείμενη που χρησιμοποιήθηκε επιτυχώς για την επίλυσή του στο dataset. Η πρόκληση έγκειται στο να διδαχθούν οι δύο encoders να παράγουν embeddings που, παρά τις δομικές τους διαφορές, θα ευθυγραμμίζονται γεωμετρικά όταν υπάρχει λογική συνάφεια.

Για την υλοποίηση αυτής της στρατηγικής, υιοθετείται η προσέγγιση του In-batch Negative Contrastive Learning, όπως αυτή εφαρμόζεται στο μοντέλο ReProver. Από μαθηματικής άποψης, η διαδικασία διαμορφώνεται ως ένα πρόβλημα ταξινόμησης (classification) εντός του εκάστοτε training batch, χρησιμοποιώντας τη loss συνάρτηση Cross-Entropy πάνω στις τιμές ομοιότητας του batch. Για κάθε proof state, το μοντέλο καλείται να ξεχωρίσει τη σωστή προκείμενη ανάμεσα σε n υποψήφιες που λειτουργούν ως «θόρυβος» (negative samples). Η ασύμμετρη φύση του Bi-Encoder απαιτεί την ταυτόχρονη ενημέρωση των παραμέτρων και των δυο encoders μέσω του gradient flow και προς τους δύο, αναγκάζοντάς τους να προσαρμόσουν τους εσωτερικούς τους μηχανισμούς αναπαράστασης έτσι ώστε να παράγουν αναπαραστάσεις που γειτνιάζουν στον κοινό διανυσματικό χώρο.

Όπως και στην περίπτωση με τον συμμετρικό bi-encoder του LeanDojo έτσι και εδώ, μια σημαντική λεπτομέρεια της εκπαιδευτικής διαδικασίας είναι η στρατηγική επιλογής των αρνητικών δειγμάτων. Εκτός από τα τυχαία αρνητικά δείγματα που προκύπτουν μέσα από το batch (in-batch negatives), το σύστημα ωθεί τον State Encoder και τον Premise Encoder να αναπτύξουν μια περισσότερο λεπτομερή διακριτική ικανότητα, ξεχωρίζοντας προκείμενες που μπορεί να μοιάζουν συντακτικά αλλά να είναι λογικά άσχετες με το τρέχον γενικό πλαίσιο (context). Αυτή η τεχνική αναγκάζει το μοντέλο να σταματήσει να βασίζεται σε απλές ομοιότητες κειμένου και να αρχίσει να αντιλαμβάνεται τα βαθύτερα λογικά χαρακτηριστικά που καθιστούν μια προκείμενη έγκυρη για έναν συγκεκριμένο στόχο ή υποστόχο.

Η ολοκλήρωση της εκπαίδευσης οδηγεί σε έναν σωστά ευθυγραμμισμένο διανυσματικό χώρο ως προς και τους δυο encoders, όπου η απόσταση μεταξύ των embeddings αντικατοπτρίζει πλέον την αποδεικτική χρησιμότητα. Παρόλο που ο State Encoder και ο Premise Encoder δεν μοιράζονται βάρη, η διαδικασία της αντιπαραθετικής μάθησης τους έχει διδάξει να μεταφράζουν τα δεδομένα εισόδου τους σε μια κοινή διανυσματική γλώσσα. Το αποτέλεσμα αυτού του pipeline είναι ένας retriever που δεν αναγνωρίζει απλώς κώδικα, αλλά αναπαριστά με ακρίβεια τη λογική εξάρτηση μεταξύ ερωτήματος και λύσης στον διανυσματικό χώρο. Αυτό το στάδιο προετοιμάζει το έδαφος για την τελική αξιολόγηση, όπου η ικανότητα του ασύμμετρου μοντέλου να ευθυγραμμίζει άγνωστα states με τις κατάλληλες προκείμενες θα συγκριθεί με την απόδοση του συμμετρικού baseline.

5.5 Ροή Δεδομένων και Υλοποίηση (Implementation Pipeline)

Η αρχιτεκτονική ενός ασύμμετρου Bi-Encoder απαιτεί μια προσεκτικά σχεδιασμένη ροή δεδομένων, ώστε να γεφυρωθεί το χάσμα μεταξύ της στατικής γνώσης (βιβλιοθήκη Mathlib4) και της δυναμικής διαδικασίας της απόδειξης. Η υλοποίηση ακολουθεί το μοντέλο της **Ανάκτησης Πυκνών Διανυσμάτων (Dense Retrieval)**, το οποίο αναλύεται στα παρακάτω στάδια:

5.5.1 Offline Indexing: Η Διαμόρφωση της Γνωσιακής Βάσης:

Η πρώτη φάση αφορά την επεξεργασία των προκειμένων (premises). Επειδή ο αριθμός των διαθέσιμων θεωρημάτων στη Lean 4 είναι εξαιρετικά μεγάλος, η αναζήτηση σε μορφή κειμένου θα ήταν απαγορευτικά αργή.

- **Tokenization & Encoding:** Κάθε προκείμενη περνά από τον εξειδικευμένο Premise Encoder. Το μοντέλο μετατρέπει τις συντακτικές δομές της Lean σε «πυκνά» διανύσματα (dense embeddings).
- **Vector Database:** Τα διανύσματα αυτά αποθηκεύονται σε μια βάση δεδομένων διανυσμάτων. Χρησιμοποιούνται αποδοτικές δομές δεδομένων που επιτρέπουν τη γρήγορη αναζήτηση γειτονικών σημείων σε χώρους πολλών διαστάσεων.
- **Στατικότητα:** Το πλεονέκτημα εδώ είναι ότι η αντιστοίχιση των προκειμένων στον διανυσματικό χώρο γίνεται μία φορά. Ο Premise Encoder λειτουργεί ως ένας «βιβλιοθηκάριος» που έχει ήδη ταξινομήσει όλα τα βιβλία στα σωστά ράφια πριν ξεκινήσει η αναζήτηση.

5.5.2 Online Retrieval: Η Δυναμική Αντιστοίχιση

Όταν ο Tactic Generator (ή ο χρήστης) ζητά μια προκείμενη για να προχωρήσει η απόδειξη, το σύστημα εκτελεί τα εξής βήματα σε πραγματικό χρόνο:

1. **Context Processing:** Ο State Encoder λαμβάνει το πλήρες proof state (τον τρέχοντα στόχο και τις διαθέσιμες υποθέσεις). Λόγω της ασύμμετρης φύσης του, είναι εκπαιδευμένος να αγνοεί τον θόρυβο της σύνταξης και να εστιάζει στα δομικά στοιχεία που απαιτούν επίλυση.
2. **Query Vector Generation:** Παράγεται ένα διάνυσμα-ερώτημα (query embedding) το οποίο αντιπροσωπεύει τη «ζήτηση» για «μαθηματική γνώση» τη δεδομένη στιγμή.
3. **Similarity Search:** Το σύστημα εκτελεί μια αναζήτηση **K-Nearest Neighbors (K-NN)** στον ενοποιημένο διανυσματικό χώρο. Υπολογίζεται η απόσταση μεταξύ του query vector και όλων των αποθηκευμένων premise vectors χρησιμοποιώντας την ομοιότητα συνημιτόνου. Όσο μεγαλύτερη είναι η τιμή, τόσο πιο σχετική θεωρείται η προκείμενη με το τρέχον State.
4. **Ranking & Output:** Οι k επικρατέστερες προκείμενες παραδίδονται στον Generator.

5.5.3 Η Σημασία της Ασυμμετρίας στην Υλοποίηση

Η επιλογή δύο διαφορετικών encoders προσπαθεί να επιλύσει το πρόβλημα της **σημασιολογικής μετατόπισης (semantic shift)**. Σε ένα συμμετρικό μοντέλο, ο encoder θα έπρεπε να εξισορροπήσει τον τρόπο που βλέπει έναν "στόχο" (που είναι μια ερώτηση) με τον τρόπο που βλέπει ένα "λήμμα" (που είναι μια απάντηση). Με την ασύμμετρη υλοποίηση, ο κάθε encoder «μιλάει τη γλώσσα του» ρόλου του, αλλά και οι δύο καταλήγουν να συνεννοούνται στον ίδιο διανυσματικό χώρο, εξασφαλίζοντας μια πολύ πιο ακριβή **αντιστοίχιση (mapping)**.

5.6 Ενσωμάτωση στο LeanDojo (Integration)

Η ενσωμάτωση της ασύμμετρης αρχιτεκτονικής στο οικοσύστημα του LeanDojo απαίτησε μια ριζική αναδιάρθρωση του τρόπου με τον οποίο το σύστημα διαχειρίζεται τα γλωσσικά μοντέλα και τα δεδομένα τους. Στην αρχική υλοποίηση του ReProver, ο retriever βασιζόταν στην κλάση PremiseRetriever, η οποία χρησιμοποιούσε έναν μοναδικό encoder και έναν tokenizer για όλες τις λειτουργίες. Στη νέα μας υλοποίηση, εισάγεται η κλάση DualEncoderRetriever, η οποία τροποποιεί τη βασική δομή του pl.LightningModule ώστε να φιλοξενεί δύο διακριτά instances: τον context_encoder για την επεξεργασία της αποδεικτικής κατάστασης και τον premise_encoder για την επεξεργασία των προκειμένων. Αυτός ο διαχωρισμός σε επίπεδο κώδικα επιτρέπει στο σύστημα να φορτώνει ανεξάρτητα βάρη για κάθε encoder, δίνοντας τη δυνατότητα χρήσης ακόμα και διαφορετικών προ-εκπαιδευμένων μοντέλων για το state και το premise αντίστοιχα.

Σε επίπεδο διαχείρισης δεδομένων, η προσαρμογή επεκτάθηκε στο DualRetrievalDataModule, το οποίο αντικατέστησε το τυπικό RetrievalDataModule. Η βασική διαφορά εντοπίζεται στη μέθοδο collate, όπου πλέον χρησιμοποιούνται δύο ξεχωριστοί tokenizers: ο context_tokenizer και ο premise_tokenizer. Ενώ στην αρχική μορφή όλα τα δεδομένα περνούσαν από την ίδια διαδικασία tokenization, στην ασύμμετρη εκδοχή το proof state και οι προκείμενες (θετικές και αρνητικές) τυγχάνουν εξειδικευμένης προ-επεξεργασίας. Αυτό διασφαλίζει ότι οι είσοδοι (inputs) που φτάνουν στους δύο encoders είναι βελτιστοποιημένες για το συγκεκριμένο λεξιλόγιο και τη σύνταξη που καλείται να διαχειριστεί ο κάθε κλάδος, ενισχύοντας την ακρίβεια της τελικής αντιστοίχισης στον διανυσματικό χώρο.

Η υλοποίηση της εκπαιδευτικής διαδικασίας στο DualEncoderRetriever αντικατοπτρίζει τη νέα λογική ροής της πληροφορίας μέσω της μεθόδου forward. Σε αντίθεση με το συμμετρικό μοντέλο που καλούσε επανειλημμένα την ίδια μέθοδο _encode, ο ασύμμετρος Bi-Encoder δρομολογεί το context_ids στον context_encoder και τα pos_premise_ids/neg_premises_ids στον premise_encoder. Η πληροφορία του σφάλματος (MSE loss) υπολογίζεται πάνω στον πίνακα ομοιοτήτων που προκύπτει από τον πολλαπλασιασμό των διαφορετικών embeddings, αναγκάζοντας τους δύο encoders να «συνεργαστούν» για την ελαχιστοποίηση της απώλειας. Η χρήση του cpu_checkpointing και στις δύο περιπτώσεις διασφαλίζει ότι η εκπαίδευση δύο ξεχωριστών μοντέλων παραμένει εφικτή εντός των ορίων της διαθέσιμης μνήμης GPU, επιτρέποντας την ταυτόχρονη ενημέρωση των βαρών τους.

Μια κρίσιμη τεχνική λεπτομέρεια στην υλοποίηση είναι ο τρόπος με τον οποίο διαχειριζόμαστε το indexing της Mathlib4. Στην κλάση DualEncoderRetriever, η μέθοδος reindex_corpus χρησιμοποιεί αποκλειστικά τον premise_encoder για να δημιουργήσει το corpus_embeddings. Αυτό αποτελεί σημαντική βελτιστοποίηση, καθώς ο context_encoder παραμένει ανενεργός κατά τη φάση της ευρετηρίασης, εξοικονομώντας υπολογιστικούς πόρους. Αντίθετα, κατά την εκτέλεση της μεθόδου retrieve σε πραγματικό χρόνο, το σύστημα χρησιμοποιεί τον context_tokenizer και τον context_encoder για να μετατρέψει το τρέχον state σε query vector, το οποίο στη συνέχεια συγκρίνεται με το ήδη υπάρχον index. Αυτή η λειτουργική διάκριση επιβεβαιώνει τη σπονδυλωτή φύση της αρχιτεκτονικής, επιτρέποντας την ταχεία ανάκτηση χωρίς την ανάγκη επανυπολογισμού των βιβλιοθηκών.

Τέλος, η ενσωμάτωση αυτή επιτρέπει μια συστηματική αξιολόγηση αυτής της μεταβολής της αρχιτεκτονικής μέσω των μετρικών Recall@k και MRR, οι οποίες υπολογίζονται στη μέθοδο validation_step. Επειδή η δομή των δεδομένων εξόδου (predictions) στο on_predict_epoch_end παραμένει συμβατή με το αρχικό format του LeanDojo (αρχεία .pickle), μπορούμε να συγκρίνουμε απευθείας την απόδοση του συμμετρικού PremiseRetriever με τον νέο DualEncoderRetriever. Η επιτυχής ενοποίηση του κώδικα αποδεικνύει ότι η εισαγωγή της ασυμμετρίας δεν αποτελεί απλώς μια θεωρητική βελτίωση, αλλά μια εφαρμόσιμη αρχιτεκτονική αλλαγή που αξιοποιεί πλήρως τις δυνατότητες του LeanDojo για την αυτόματη απόδειξη θεωρημάτων.

6. ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΑΞΙΟΛΟΓΗΣΗ

Στο παρόν κεφάλαιο παρουσιάζονται και αναλύονται τα αποτελέσματα της πειραματικής αξιολόγησης της προτεινόμενης ασύμμετρης αρχιτεκτονικής σε σύγκριση με το μοντέλο αναφοράς (Baseline) του ReProver. Η αξιολόγηση πραγματοποιήθηκε στο benchmark dataset του LeanDojo (Mathlib4), χρησιμοποιώντας τα σύνολα δεδομένων "Random" και "Novel Premises" για τον έλεγχο της γενικευτικής ικανότητας των μοντέλων. Μέσα από τη χρήση των μετρικών ανάκτησης Recall@1, Recall@10 και MRR, επιχειρείται μια ποσοτική αποτίμηση της επίδρασης που έχει ο διαχωρισμός των encoders στην ακρίβεια εντοπισμού των κατάλληλων προκειμένων (premises) για την απόδειξη θεωρημάτων.

6.1 Πειραματικό Περιβάλλον

Η πειραματική διαδικασία για την αξιολόγηση της ασύμμετρης αρχιτεκτονικής του retriever σχεδιάστηκε με γνώμονα τη διασφάλιση της μέγιστης δυνατής ακρίβειας και αναπαραγωγής των αποτελεσμάτων, εντός ενός οικοσυστήματος υψηλών υπολογιστικών απαιτήσεων. Η ανάγκη για την ταυτόχρονη διαχείριση δύο διακριτών encoders, οι οποίοι επεξεργάζονται διαφορετικά είδη μαθηματικής πληροφορίας (proof states και premises), επέβαλε τη χρήση μιας υποδομής που να μπορεί να υποστηρίξει το αυξημένο αποτύπωμα μνήμης και την πολυπλοκότητα των υπολογισμών. Η επιλογή του συγκεκριμένου περιβάλλοντος έγινε ώστε να εξαλειφθούν τυχόν περιορισμοί στο hardware που θα μπορούσαν να αλλοιώσουν τα αποτελέσματα, επιτρέποντας στο μοντέλο να εκπαιδευτεί απρόσκοπτα στη βιβλιοθήκη Mathlib4 μέσω του framework LeanDojo.

Το κεντρικό υπολογιστικό φορτίο διεκπεραιώθηκε από την κορυφαία μονάδα επεξεργασίας NVIDIA A100-SXM4, η οποία προσφέρει 40GB μνήμης VRAM τύπου HBM2e. Η συγκεκριμένη κάρτα γραφικών, βασισμένη στην αρχιτεκτονική Ampere, διαθέτει Tensor Cores τρίτης γενιάς, οι οποίοι είναι βελτιστοποιημένοι για πράξεις πινάκων μεγάλης κλίμακας, απαραίτητες για τον υπολογισμό των cosine similarities μεταξύ των χιλιάδων υποψήφιων προκειμένων. Η διαθέσιμη μνήμη των 40GB υπήρξε καθοριστική, καθώς επέτρεψε τη φόρτωση των δύο ανεξάρτητων ByT5-small μοντέλων μαζί με τους αντίστοιχους tokenizers, διατηρώντας παράλληλα επαρκή χώρο για τα gradients και τα optimizer states κατά τη διάρκεια του backpropagation, χωρίς να χρειαστεί υπερβολική μείωση του batch size.

Σε επίπεδο λογισμικού και οργάνωσης του κώδικα, χρησιμοποιήθηκε το framework PyTorch Lightning, το οποίο παρείχε μια modular δομή για την ενσωμάτωση του DualEncoderRetriever και του DualRetrievalDataModule. Η χρήση της βιβλιοθήκης DeepSpeed σε συνδυασμό με το Triton επέτρεψε τη βελτιστοποίηση των CUDA kernels, αυξάνοντας την ταχύτητα εκπαίδευσης και μειώνοντας την κατανάλωση μνήμης μέσω τεχνικών όπως το gradient checkpointing. Παρόλο που το σύστημα εντόπισε ότι ο κατάλογος cache του Triton autotune βρισκόταν σε δικτυακό σύστημα αρχείων (NFS), πραγματοποιήθηκαν οι απαραίτητες ρυθμίσεις στις περιβαλλοντικές μεταβλητές ώστε να αποφευχθούν καθυστερήσεις κατά την έξοδο των διεργασιών, διασφαλίζοντας τη σταθερότητα του pipeline.

Το σύνολο δεδομένων που αξιοποιήθηκε είναι το benchmark του LeanDojo (Mathlib4), το οποίο αποτελεί την τρέχουσα αιχμή του δόρατος στην αυτοματοποιημένη απόδειξη θεωρημάτων για τη γλώσσα Lean 4. Η εκπαίδευση, για αρχή, επικεντρώθηκε στο random split, το οποίο περιλαμβάνει 118.517 παραδείγματα στο train set, προσφέροντας μια ευρεία βάση για τη μάθηση των μαθηματικών συσχετίσεων. Η αξιολόγηση, στην συνέχεια, επεκτάθηκε και στο novel premises split, μια ιδιαίτερα απαιτητική δοκιμασία όπου το μοντέλο καλείται να ανακτήσει προκείμενες που δεν έχουν εμφανιστεί ποτέ κατά την εκπαίδευση. Αυτός ο διαχωρισμός είναι ζωτικής σημασίας για την παρούσα πτυχιακή εργασία, καθώς ελέγχει αν η ασύμμετρη αρχιτεκτονική βοηθά στη βαθύτερη κατανόηση της μαθηματικής δομής ή αν οδηγεί σε απλή απομνημόνευση των δεδομένων εκπαίδευσης.

Τέλος, οι παράμετροι του μοντέλου ρυθμίστηκαν ώστε να ευθυγραμμίζονται με το ByT5-small, μια έκδοση του T5 που λειτουργεί σε επίπεδο bytes, καθιστώντας την ιδανική για τη διαχείριση της σύνταξης κώδικα Lean χωρίς την ανάγκη για πολύπλοκα λεξιλόγια. Η εκπαίδευση βασίστηκε στη βελτιστοποίηση της συνάρτησης σφάλματος MSE Loss, η οποία εφαρμόζεται πάνω στις προβλεπόμενες τιμές ομοιότητας σε σχέση με τις πραγματικές ετικέτες (labels) του batch. Η επιλογή αυτή, σε συνδυασμό με τον διαχωρισμό των tokenizers για το proof state και τα premises,

δημιούργησε ένα εξειδικευμένο περιβάλλον μάθησης που επιτρέπει την εξαγωγή ασφαλών συμπερασμάτων για τη συμπεριφορά της νέας ασύμμετρης αρχιτεκτονικής έναντι του παραδοσιακού συμμετρικού retriever.

6.2 Μετρικές Αξιολόγησης

Η αξιολόγηση της ικανότητας του μοντέλου να εντοπίζει τις κατάλληλες προκείμενες μέσα από ένα τεράστιο σύνολο υποψηφίων απαιτεί τη χρήση εξειδικευμένων μετρικών από το πεδίο της Ανάκτησης Πληροφοριών (Information Retrieval). Δεδομένου ότι για κάθε αποδεικτικό βήμα (tactic) μπορεί να υπάρχουν μία ή περισσότερες ορθές προκείμενες, η επιτυχία του συστήματος δεν κρίνεται μόνο από το αν θα βρει την πληροφορία, αλλά και από τη θέση (rank) που αυτή καταλαμβάνει στις προτάσεις του. Για τον σκοπό αυτό, επιστρατεύτηκαν οι μετρικές Recall@k και Mean Reciprocal Rank (MRR), οι οποίες προσφέρουν μια ολοκληρωμένη εικόνα για την ποιότητα της αντιστοίχισης στον διανυσματικό χώρο και τη χρηστικότητα του retriever σε ένα πραγματικό σύστημα αυτόματης απόδειξης.

Η μετρική Recall@k αποτελεί τον βασικό δείκτη απόδοσης και ορίζεται ως το ποσοστό των περιπτώσεων στις οποίες τουλάχιστον μία από τις απαραίτητες προκείμενες περιλαμβάνεται μέσα στις k πρώτες προτάσεις του μοντέλου. Στην παρούσα εργασία, εστιάζω κυρίως στις τιμές Recall@1 και Recall@10. Το Recall@1 είναι η πιο αυστηρή μετρική, καθώς απαιτεί η ορθή προκείμενη να είναι η κορυφαία επιλογή του συστήματος, ενώ το Recall@10 αντικατοπτρίζει μια πιο ρεαλιστική συνθήκη χρήσης, όπου ένας Tactic Generator (όπως ο ByT5) λαμβάνει μια μικρή λίστα υποψηφίων για να επιλέξει την καταλληλότερη.

Παράλληλα, χρησιμοποιήθηκε η μετρική Mean Reciprocal Rank (MRR), η οποία προσφέρει μια πιο λεπτομερή ανάλυση της ποιότητας της κατάταξης σε σχέση με το Recall. Το MRR υπολογίζει τον μέσο όρο των αντιστρόφων των θέσεων στις οποίες εμφανίστηκε η πρώτη ορθή προκείμενη για κάθε ερώτημα. Αν η ορθή προκείμενη βρίσκεται στην πρώτη θέση, το reciprocal rank είναι 1, αν βρίσκεται στη δεύτερη είναι 0.5, και ούτω καθεξής, ενώ αν δεν βρεθεί καθόλου εντός του ορίου αναζήτησης, η τιμή είναι 0. Η μετρική αυτή είναι ιδιαίτερα ευαίσθητη στην ακρίβεια των πρώτων θέσεων, επιβραβεύοντας τα μοντέλα που τοποθετούν τη γνώση όσο το δυνατόν πιο κοντά στην κορυφή της λίστας.

Η επιλογή αυτών των μετρικών επιτρέπει την άμεση σύγκριση μεταξύ της αρχιτεκτονικής του ασύμμετρου και του συμμετρικού μοντέλου του ReProver, αναδεικνύοντας τις λεπτές διαφορές στη συμπεριφορά τους. Ενώ το Recall@k μετρά την «εξαντλητικότητα» (exhaustivity) της ανάκτησης, το MRR αξιολογεί την ακρίβεια κατάταξης (ranking precision) των αποτελεσμάτων στις κορυφαίες θέσεις. Στην περίπτωση των μαθηματικών αποδείξεων, ένα υψηλό MRR είναι εξαιρετικά επιθυμητό, καθώς μειώνει τον θόρυβο που δέχεται ο generator και περιορίζει τον χώρο αναζήτησης, καθιστώντας τη διαδικασία της απόδειξης πιο αποδοτική και λιγότερο επιρρεπή σε σφάλματα που προκύπτουν από άσχετες προκείμενες.

Τέλος, η αξιολόγηση μέσω αυτών των δεικτών πραγματοποιήθηκε σε όλα τα splits του dataset (train, val, test), προσφέροντας μια σφαιρική εικόνα για τη γενικευτική ικανότητα του μοντέλου. Ιδιαίτερη έμφαση δόθηκε στα αποτελέσματα του test set, καθώς εκεί κρίνεται η ικανότητα του Retriever να διαχειρίζεται εντελώς νέα αποδεικτικά πλαίσια. Η σύγκριση των τιμών Recall και MRR μεταξύ του random και του novel premises split αποτελεί το τελικό κριτήριο για το αν ο διαχωρισμός των encoders (ασυμμετρία) ενισχύει τη σημασιολογική κατανόηση των μαθηματικών εννοιών ή αν η κοινή αναπαράσταση του συμμετρικού μοντέλου παραμένει πιο αποτελεσματική για τη συγκεκριμένη κλίμακα δεδομένων.

6.3 Συγκριτική Ανάλυση Αποτελεσμάτων

Η ποσοτική αξιολόγηση των δύο μοντέλων πραγματοποιήθηκε με τη χρήση των προκαθορισμένων splits του LeanDojo, προσφέροντας μια άμεση σύγκριση μεταξύ της συμμετρικής και της ασύμμετρης προσέγγισης. Τα αποτελέσματα παρατίθενται διαχωρισμένα σε δύο κύριες κατηγορίες: το random split, το οποίο αντιπροσωπεύει τη συνήθη ικανότητα ανάκτησης σε γνωστά πλαίσια, και το novel premises split, το οποίο αποτελεί την απόλυτη δοκιμασία γενίκευσης σε προκείμενες που δεν χρησιμοποιήθηκαν κατά την εκπαίδευση. Οι τιμές που καταγράφηκαν αποτυπώνουν την επίδραση του διαχωρισμού των encoders στην ακρίβεια της μαθηματικής αναζήτησης.

Στον Πίνακα 1 παρουσιάζονται τα αποτελέσματα για το random split. Παρατηρούμε ότι το baseline μοντέλο υπερέχει ελαφρώς σε όλες τις μετρικές στο test set, επιτυγχάνοντας Recall@1 11.08% έναντι 9.35% του ασύμμετρου retriever. Η διαφορά αυτή, αν και υπαρκτή, δείχνει ότι η ασύμμετρη αρχιτεκτονική καταφέρνει να διατηρήσει ένα ανταγωνιστικό επίπεδο απόδοσης, παρά την αυξημένη πολυπλοκότητα που εισάγει η ανάγκη ευθυγράμμισης δύο διαφορετικών διανυσματικών χώρων. Το MRR του ασύμμετρου retriever (0.254) υποδεικνύει ότι, παρά την ελαφριά πτώση, η ορθή πληροφορία παραμένει σε υψηλές θέσεις κατάταξης.

Πίνακας 1: Αποτελέσματα στο Random Split

Μοντέλο	Dataset Split	Recall@1(%)	Recall@10(%)	MRR
Συμμετρικός Bi-encoder	Train	27.85	68.50	0.595
	Val	12.49	36.83	0.311
	Test	11.08	35.54	0.288
Ασύμμετρος Bi-encoder	Train	18.07	52.98	0.446
	Val	10.35	33.27	0.274
	Test	9.35	31.41	0.254

Στον Πίνακα 2 εξετάζονται τα αποτελέσματα για το novel premises split, όπου η δυσκολία του προβλήματος αυξάνεται σημαντικά. Εδώ, το baseline μοντέλο σημειώνει Recall@1 7.66% στο test set, ενώ ο ασύμμετρος retriever ακολουθεί με 6.12%. Είναι αξιοσημείωτο ότι η ποσοστιαία πτώση από το random στο novel split είναι αναλογικά παρόμοια και στα δύο μοντέλα. Αυτό υποδηλώνει ότι η υστέρηση του ασύμμετρου retriever δεν οφείλεται σε αδυναμία γενίκευσης, αλλά σε μια σταθερή διαφορά δυναμικής που πιθανώς σχετίζεται με τη δυσκολία σύγκλισης των δύο ανεξάρτητων encoders κατά την εκπαίδευση.

Πίνακας 2: Αποτελέσματα στο Novel Premises Split

Μοντέλο	Dataset Split	Recall@1(%)	Recall@10(%)	MRR
Συμμετρικός Bi-encoder	Train	22.10	59.01	0.506
	Val	8.12	25.47	0.238
	Test	7.66	25.82	0.234
Ασύμμετρος Bi-encoder	Train	13.39	42.51	0.356
	Val	6.00	22.15	0.191
	Test	6.12	22.86	0.209

Η ανάλυση των παραπάνω δεδομένων αποκαλύπτει ότι η συμμετρική αρχιτεκτονική του ReProver παραμένει πιο αποδοτική για το συγκεκριμένο μέγεθος μοντέλου και δεδομένων. Η χρήση κοινών βαρών (weight sharing) στο συμμετρικό μοντέλο φαίνεται να λειτουργεί ως ένας

ισχυρός ρυθμιστής (regularizer), ο οποίος διευκολύνει τη μάθηση κοινών σημασιολογικών προτύπων μεταξύ της κατάστασης της απόδειξης και των διαθέσιμων θεωρημάτων. Παρόλα αυτά, η επιτυχής λειτουργία της ασύμμετρης έκδοσης επιβεβαιώνει ότι η νέα προσέγγιση είναι τεχνικά ορθή και ικανή να παράγει αξιοποιήσιμα αποτελέσματα, προσφέροντας παράλληλα μεγαλύτερη αρχιτεκτονική ευελιξία.

Συμπερασματικά, αν και οι απόλυτοι αριθμοί ευνοούν το αρχικό μοντέλο, η διαφορά δεν είναι χαώδης. Η απόδοση του ασύμμετρου retriever στο test set (Recall@10 πάνω από 31% στο random και 22% στο novel) αποδεικνύει ότι το μοντέλο έχει «μάθει» να συσχετίζει το context με τα premises, παρά τις δυσκολίες που εισάγει ο πλήρης διαχωρισμός των δύο κλάδων επεξεργασίας. Τα ευρήματα αυτά θέτουν τις βάσεις για μια βαθύτερη ερμηνεία των αιτιών αυτής της απόκλισης, η οποία θα αναλυθεί στην επόμενη ενότητα.

6.4 Ερμηνεία και Σχολιασμός

Η υπεροχή του συμμετρικού μοντέλου έναντι της ασύμμετρης αρχιτεκτονικής, αν και οριακή σε ορισμένες μετρικές, αναδεικνύει ορισμένες θεμελιώδεις προκλήσεις στη διαδικασία της μαθηματικής ανάκτησης. Η κυριότερη αιτία αυτής της απόκλισης εντοπίζεται στην απουσία του Weight Sharing (κοινά βάρη). Στο συμμετρικό μοντέλο, ο μοναδικός encoder εκπαιδεύεται να αναπαριστά τόσο τα proof states όσο και τα premises στον ίδιο ακριβώς διανυσματικό χώρο χρησιμοποιώντας τις ίδιες παραμέτρους. Αυτό λειτουργεί ως ένας ισχυρός εγγενής περιορισμός (inductive bias), ο οποίος αναγκάζει το μοντέλο να μάθει μια κοινή «γλώσσα» για το context και το target, διευκολύνοντας την ευθυγράμμιση των διανυσμάτων τους. Αντίθετα, το ασύμμετρο μοντέλο πρέπει να συγκλίνει σε δύο διαφορετικούς χώρους αναπαράστασης ταυτόχρονα, γεγονός που καθιστά το πρόβλημα της βελτιστοποίησης σημαντικά πιο σύνθετο.

Μια δεύτερη κρίσιμη παράμετρος αφορά τη Χωρητικότητα του Μοντέλου (Model Capacity) σε σχέση με το διαθέσιμο σύνολο δεδομένων. Με τον διαχωρισμό των encoders, ο αριθμός των εκπαιδεύσιμων παραμέτρων του retriever ουσιαστικά διπλασιάζεται. Παρόλο που αυτό θεωρητικά επιτρέπει στο μοντέλο να μάθει πιο εξειδικευμένες αναπαραστάσεις για τις προκείμενες, στην πράξη απαιτεί πολύ περισσότερα δεδομένα ή μεγαλύτερο αριθμό εποχών εκπαίδευσης (training epochs) για να φτάσει στο ίδιο επίπεδο ακρίβειας με το πιο «συμπαγές» baseline μοντέλο. Η υστέρηση που παρατηρήθηκε στο train set του ασύμμετρου μοντέλου (R@1 18.07% έναντι 27.85%) υποδηλώνει ενδεχομένως ότι το μοντέλο χρειαζόταν περισσότερο χρόνο για να προσαρμοστεί στις λεπτές συντακτικές διαφορές της Lean 4, καθώς κάθε κλάδος έπρεπε να ξεκινήσει τη μάθηση από διαφορετική αφετηρία.

Επιπλέον, η φύση των μαθηματικών δεδομένων στη Mathlib4 παρουσιάζει υψηλή ομοιογένεια. Οι καταστάσεις απόδειξης (proof states) και οι ορισμοί των προκειμένων (premises) μοιράζονται κοινή σύνταξη, σύμβολα και δομή. Στον ασύμμετρο retriever, η χρήση δύο διαφορετικών tokenizers και encoders στερεί από το σύστημα τη δυνατότητα να εκμεταλλευτεί αυτή την ομοιότητα. Ενώ σε άλλες εφαρμογές (π.χ. αναζήτηση κειμένου σε φυσική γλώσσα) η ασυμμετρία μεταξύ ερώτησης και απάντησης είναι ευεργετική, στον χώρο του τυπικού ελέγχου αποδείξεων η σημασιολογική απόσταση μεταξύ «ερώτησης» (state) και «απάντησης» (premise) είναι μικρή. Συνεπώς, η επιβολή μιας ασύμμετρης δομής ίσως εισάγει έναν περιττό βαθμό ελευθερίας που δυσκολεύει την ακριβή ιεράρχηση των αποτελεσμάτων.

Παρά την πτώση των επιδόσεων, η ανάλυση του MRR (0.254 στο random test) αποκαλύπτει μια ενθαρρυντική πτυχή: ο ασύμμετρος retriever δεν αποτυγχάνει να βρει την πληροφορία, αλλά συχνά την κατατάσσει λίγες θέσεις χαμηλότερα από την κορυφή. Αυτό υποδεικνύει ότι ο διαχωρισμός των κλάδων επεξεργασίας επιτρέπει στο μοντέλο να κατανοήσει τη γενική συνάφεια (relevance), αλλά στερείται της «χειρουργικής» ακρίβειας που προσφέρει το weight sharing στην επιλογή της μίας και μοναδικής ορθής προκείμενης. Η σταθερή απόδοση στο novel premises split (Recall@10 στο 22.86%) επιβεβαιώνει ότι η αρχιτεκτονική διαθέτει στιβαρή γενικευτική ικανότητα και δεν περιορίζεται σε απλή απομνημόνευση, γεγονός που την καθιστά μια έγκυρη εναλλακτική προσέγγιση για μελλοντικά πειράματα με μεγαλύτερης κλίμακας μοντέλα.

Συμπερασματικά, τα πειραματικά δεδομένα υποδηλώνουν ότι για τη συγκεκριμένη κλίμακα παραμέτρων (ByT5-small), η συμμετρική προσέγγιση είναι πιο αποδοτική λόγω της καλύτερης αξιοποίησης των περιορισμένων πόρων. Ωστόσο, η επιτυχής υλοποίηση και λειτουργία του ασύμμετρου retriever αποτελεί μια σημαντική τεχνική συνεισφορά, καθώς αποδεικνύει ότι η αρχιτεκτονική του LeanDojo μπορεί να υποστηρίξει πιο σύνθετες δομές ανάκτησης. Τα ευρήματα αυτά παρέχουν μια σαφή κατεύθυνση για μελλοντική έρευνα, υποδεικνύοντας ότι η ασυμμετρία ίσως αποδώσει καλύτερα σε περιβάλλοντα όπου το proof state και οι προκείμενες διαφέρουν ριζικά σε μορφή ή όταν χρησιμοποιούνται τεχνικές προ-εκπαίδευσης (pre-training) που στοχεύουν ειδικά στην ευθυγράμμιση διαφορετικών encoders.

6.5 Συμπεράσματα

Η ολοκλήρωση της πειραματικής διαδικασίας και η ανάλυση των αποτελεσμάτων παρέχουν μια σαφή εικόνα για τη συμπεριφορά της ασύμμετρης αρχιτεκτονικής στο πλαίσιο της αυτόματης απόδειξης θεωρημάτων. Το βασικό συμπέρασμα που προκύπτει είναι ότι ο ασύμμετρος retriever αποτελεί μια τεχνικά έγκυρη και λειτουργική προσέγγιση, η οποία όμως, υπό τις παρούσες συνθήκες εκπαίδευσης και κλίμακας μοντέλου, υστερεί ελαφρώς σε απόλυτη ακρίβεια έναντι του παραδοσιακού συμμετρικού μοντέλου. Η διαφορά των δύο ποσοστιαίων μονάδων στο Recall@1 (9.35% έναντι 11.08%) αναδεικνύει τη δυσκολία του διαχωρισμού των «σημασιολογικών χώρων», αλλά ταυτόχρονα επιβεβαιώνει ότι η αρχιτεκτονική "ByT5-small" μπορεί να υποστηρίξει την ασυμμετρία χωρίς να καταρρέει η ικανότητα ανάκτησης.

Ιδιαίτερα σημαντική είναι η διαπίστωση ότι η υστέρηση του μοντέλου είναι ομοιόμορφη σε όλα τα επίπεδα αξιολόγησης. Η σταθερή απόδοση στο novel premises split, όπου ο ασύμμετρος retriever πέτυχε Recall@10 22.86%, αποδεικνύει ότι το σύστημα δεν βασίζεται στην απλή απομνημόνευση των δεδομένων εκπαίδευσης, αλλά έχει αναπτύξει έναν βαθμό γενικευτικής ικανότητας. Το γεγονός ότι το μοντέλο καταφέρνει να εντοπίζει άγνωστα premises σε ένα τόσο απαιτητικό περιβάλλον όπως η Mathlib4, υποδηλώνει ότι η στρατηγική των δύο encoders διατηρεί τη μαθηματική λογική, παρά την αυξημένη πολυπλοκότητα στη σύγκλιση των βαρών.

Από την ανάλυση των μετρικών, προκύπτει ότι το MRR αποτελεί τον πλέον αποκαλυπτικό δείκτη για τη φύση της ασυμμετρίας. Η τιμή 0.254 στο random test δείχνει ότι το ασύμμετρο μοντέλο "καταλαβαίνει" το context της απόδειξης, τοποθετώντας τις ορθές προκείμενες μέσα στην πρώτη τετράδα των αποτελεσμάτων κατά μέσο όρο. Η αδυναμία του να κατακτήσει την πρώτη θέση με την ίδια συχνότητα όπως το baseline μοντέλο αποδίδεται κυρίως στην έλλειψη weight sharing, η οποία στερεί από το μοντέλο τη δυνατότητα για μια πιο "σφιχτή" ευθυγράμμιση μεταξύ των embeddings του proof state και των premises.

Επιπλέον, η εμπειρία από την εκτέλεση των πειραμάτων στο σύστημα NVIDIA A100 ανέδειξε τις προκλήσεις διαχείρισης πόρων που εισάγει η προτεινόμενη αρχιτεκτονική. Η χρήση δύο ανεξάρτητων tokenizers και η ανάγκη για κάποιο gradient checkpointing επιβεβαιώνουν ότι ο ασύμμετρος retriever είναι μια πιο "βαριά" προσέγγιση, η οποία ίσως απαιτεί μεγαλύτερη υπολογιστική προσπάθεια για να αποδώσει τα μέγιστα. Παρόλα αυτά, η επιτυχής ενσωμάτωση εργαλείων όπως το DeepSpeed και το Triton στο pipeline της Mathlib4 αποτελεί από μόνη της μια σημαντική τεχνική παρακαταθήκη για μελλοντικές δοκιμές σε μεγαλύτερα μοντέλα.

Συνοψίζοντας, το Κεφάλαιο 6 τεκμηριώνει ότι ενώ η συμμετρική αρχιτεκτονική παραμένει η βέλτιστη επιλογή για μοντέλα μικρής κλίμακας, ο ασύμμετρος retriever ανοίγει νέους δρόμους για την εξειδίκευση των αναπαραστάσεων στη μαθηματική ανάκτηση. Τα αποτελέσματα αυτά δεν αποτελούν απλώς μια σύγκριση αριθμών, αλλά μια βαθύτερη διερεύνηση των ορίων του LeanDojo.

7. ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Η παρούσα πτυχιακή εργασία διερεύνησε τη δυνατότητα εισαγωγής μιας ασύμμετρης αρχιτεκτονικής (Dual Encoder) στο σύστημα ανάκτησης προκειμένων του LeanDojo, χρησιμοποιώντας τη βιβλιοθήκη Mathlib4. Μέσα από τη σχεδίαση και την υλοποίηση ενός μοντέλου με διακριτούς encoders για το proof state και τα premises, αποδείχθηκε ότι η συγκεκριμένη αρχιτεκτονική, που «αρχικοποιήθηκε» βάσει του ByT5-small, είναι ικανή να διαχειριστεί τον διαχωρισμό των σημασιολογικών ρόλων, παρά τις προκλήσεις που θέτει η έλλειψη κοινών βαρών. Αν και τα αποτελέσματα έδειξαν μια ελαφριά υπεροχή του συμμετρικού μοντέλου, η επιτυχής εκπαίδευση και η σταθερή απόδοση στο novel premises split επιβεβαιώνουν ότι η προτεινόμενη προσέγγιση αποτελεί μια στιβαρή εναλλακτική για τη μαθηματική αναζήτηση.

Το κυριότερο συμπέρασμα που εξάγεται είναι ότι η συμμετρία (single encoder) λειτουργεί ως ένας αποτελεσματικός ρυθμιστής για μοντέλα περιορισμένης κλίμακας παραμέτρων, καθώς εκμεταλλεύεται τη συντακτική ομοιότητα μεταξύ των μαθηματικών εκφράσεων. Η δική μας υλοποίηση ανέδειξε ότι η ασυμμετρία (dual encoder) απαιτεί ενδεχομένως μεγαλύτερο όγκο δεδομένων ή πιο εξελιγμένες στρατηγικές ευθυγράμμισης (alignment) για να ξεπεράσει το φράγμα του weight sharing. Παρόλα αυτά, η τεχνική υποδομή που χρησιμοποιήθηκε, με τη χρήση DeepSpeed και Triton σε περιβάλλον NVIDIA A100, αποτελεί μια σημαντική παρακαταθήκη, αποδεικνύοντας ότι το framework του ReProver μπορεί να επεκταθεί σε πιο σύνθετες μορφές ανάκτησης.

Όσον αφορά τις μελλοντικές επεκτάσεις, μια άμεση κατεύθυνση θα ήταν η χρήση μεγαλύτερων προ-εκπαιδευμένων μοντέλων, όπως το ByT5-base ή το ByT5-large. Η αύξηση των παραμέτρων θα επέτρεπε στους δύο encoders να αναπτύξουν πιο πλούσιες αναπαραστάσεις, ξεπερνώντας τους περιορισμούς που παρατηρήθηκαν στην έκδοση small. Επιπλέον, η εισαγωγή μιας φάσης Cross-Encoder μετά την αρχική ανάκτηση (reranking) θα μπορούσε να βελτιώσει σημαντικά τις τιμές του Recall@1 και του MRR, καθώς ένα μοντέλο που εξετάζει ταυτόχρονα το state και το candidate premise μπορεί να συλλάβει λεπτότερες «σημασιολογικές αποχρώσεις» που διαφεύγουν από το dot product των embeddings.

Μια άλλη ενδιαφέρουσα προοπτική είναι η εφαρμογή τεχνικών Contrastive Learning με τη χρήση "δύσκολων αρνητικών" δειγμάτων (hard negatives) κατά την εκπαίδευση. Αντί το μοντέλο να μαθαίνει από τυχαία premises, θα μπορούσε να εκπαιδευτεί να διακρίνει τις ορθές προκείμενες από άλλες που είναι συντακτικά παρόμοιες αλλά μαθηματικά άσχετες. Η συγκεκριμένη ιδέα είναι ήδη υλοποιημένη σε κάποιο «πρωτόγονο» στάδιο εντός του framework LeanDojo μέσω της χρήσης hard negatives και in-file negatives κατά την εκπαίδευση. Παρόλα αυτά ενδέχεται να υπάρχουν καλύτεροι τρόποι ανεύρεσης τέτοιων προκειμένων (από το να τις συλλέγει κανείς από το ίδιο αρχείο Lean για παράδειγμα) και κατ' επέκταση σχηματισμού πιο αποτελεσματικών batches που θα οδηγούν σε «εξυπνότερα» μοντέλα. Αυτό θα βοηθούσε τον ασύμμετρο retriever να αναπτύξει την "χειρουργική" ακρίβεια που του έλειψε στα ανωτέρω πειράματά, ενισχύοντας την ικανότητά του να ιεραρχεί σωστά τα αποτελέσματα στην κορυφή της λίστας (Top-1).

Τέλος, σε ένα ευρύτερο πλαίσιο, η έρευνα θα μπορούσε να επεκταθεί προς την κατεύθυνση της πολυτροπικής μάθησης (multimodal learning) ή της αξιοποίησης γραφημάτων γνώσης (Knowledge Graphs) για την αναπαράσταση των σύνθετων αλληλεξαρτήσεων μεταξύ των μαθηματικών οντοτήτων. Μια τέτοια προσέγγιση θα επέτρεπε στον premise encoder να μην περιορίζεται αποκλειστικά στη «γραμμική» ανάγνωση του κώδικα Lean, αλλά να αντιλαμβάνεται τη θέση κάθε θεωρήματος μέσα στην ευρύτερη ιεραρχική δομή και το δίκτυο εξαρτήσεων της Mathlib4. Η ενσωμάτωση αυτής της "δομικής συνάφειας" (structural context) στον ασύμμετρο retriever θα προσέφερε μια επιπλέον διάσταση κατανόησης, επιτρέποντας στο σύστημα να συσχετίζει προκείμενες με βάση τη λογική τους γειτνίαση και όχι μόνο τη συντακτική τους ομοιότητα. Συνολικά, η διερεύνηση αυτών των υβριδικών μοντέλων ανάκτησης, που συνδυάζουν τη σημασιολογία του κειμένου με τη γεωμετρία των μαθηματικών βιβλιοθηκών, αποτελεί μια από τις πιο υποσχόμενες προκλήσεις για τη δημιουργία της επόμενης γενιάς εργαλείων αυτόματης απόδειξης.

8. ΒΙΒΛΙΟΓΡΑΦΙΑ

1. K. Yang et al., "LeanDojo: Theorem Proving with Retrieval-Augmented Language Models," *arXiv preprint arXiv:2306.15626*, 2023.
2. L. de Moura and S. Ullrich, "The Lean 4 Theorem Prover and Programming Language," in *International Conference on Automated Deduction (CADE)*, pp. 625-635, Springer, 2021.
3. The mathlib Community, "The Lean 4 mathematical library," [Online]. Available: https://github.com/leanprover-community/mathlib4.
4. J. Rasley et al., "DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505-3506, 2020.
5. A. A. Alemi et al., "DeepMath - Deep Learning for Abstract Mathematical Reasoning," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, 2016.
6. K. Bansal et al., "HOList: An Environment for Machine Learning of Higher-Order Logic Theorem Proving," *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.
7. S. Polu and I. Sutskever, "Generative Language Modeling for Automated Theorem Proving," *arXiv preprint arXiv:2009.03393*, 2020. (GPT-f)
8. B. Li et al., "Thor: Wielding Hammers to Yield Proofs," *Advances in Neural Information Processing Systems (NeurIPS)*, 2022/2024.
9. J. Azerbayev et al., "Llemma: An Open Language Model for Mathematics," *arXiv preprint arXiv:2310.10631*, 2023.
10. "DeepSeek-Prover: Advancing Theorem Proving via Reinforcement Learning and MCTS," *DeepSeek-AI Technical Report*, 2025.
11. "REAL-Prover: Retrieval-Enhanced Automated Logical Prover," *arXiv preprint*, 2025.
12. "HybriL: Hybrid Language Models for Theorem Proving," International Conference on Learning Representations (ICLR), 2024.
13. "DT-Solver: Dynamic Tree Search for Formal Mathematical Reasoning," arXiv preprint, 2025.
14. V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain Question Answering," *EMNLP*, 2020.
15. L. Xue et al., "ByT5: Towards a Token-Free Future with Byte-Level Models," *TACL*, vol. 10, 2022.