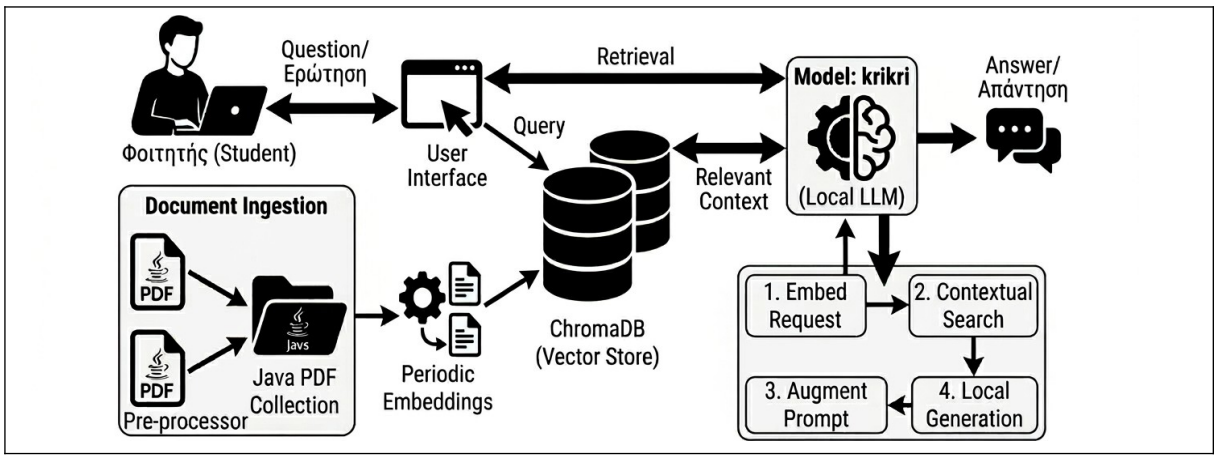
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Διαδραστικού Εκπαιδευτικού Βοηθού με Χρήση Μεγάλων Γλωσσικών Μοντέλων και Τεχνικών Retrieval-Augmented Generation: Εφαρμογή στο Μάθημα «Αντικειμενοστραφής Προγραμματισμός»



Του φοιτητή
Πετσαλάκη Βασίλη
Αρ. Μητρώου: 185328

Επιβλέπων
Αδαμίδης Παναγιώτης
Καθηγητής

Θεσσαλονίκη, Ιούνιος 2026

Τίτλος Π.Ε.: Ανάπτυξη Διαδραστικού Εκπαιδευτικού Βοηθού με Χρήση Μεγάλων Γλωσσικών Μοντέλων και Τεχνικών Retrieval-Augmented Generation: Εφαρμογή στο Μάθημα «Αντικειμενοστραφής Προγραμματισμός»

Κωδικός Π.Ε.: 25304

Ονοματεπώνυμο φοιτητή: Πετσαλάκης Βασίλης

Ονοματεπώνυμο εισηγητή: Αδαμίδης Παναγιώτης

Ημερομηνία ανάληψης Π.Ε.: 29/09/2025

Ημερομηνία περάτωσης Π.Ε.: 24/05/2026

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως πτυχιακή εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Στους γονείς μου,
και σε όσους πίστεψαν σε μένα.

Πρόλογος

Η παρούσα πτυχιακή εργασία αποσκοπεί στην ανάπτυξη ενός διαδραστικού εκπαιδευτικού βοηθού με χρήση Μεγάλων Γλωσσικών Μοντέλων και Τεχνικών Retrieval-Augmented Generation, με εφαρμογή στο μάθημα «Αντικειμενοστραφής Προγραμματισμός» και εκπονήθηκε στο πλαίσιο των σπουδών μου στο τμήμα Μηχανικών Πληροφορικής του ΔΙΠΑΕ. Η αφορμή για την επιλογή του θέματος ήταν η γρήγορη εξέλιξη της Τεχνητής Νοημοσύνης, η οποία με έκανε να θέλω να μάθω πως τα Μεγάλα Γλωσσικά Μοντέλα (LLMs) μπορούν να κάνουν ένα λογισμικό πιο έξυπνο.

Για την παρούσα εργασία, επέλεξα να εστιάσω στον τομέα της εκπαίδευσης, αναπτύσσοντας έναν βοηθό που εξειδικεύεται στο μάθημα του Αντικειμενοστραφούς Προγραμματισμού της σχολής. Ταυτόχρονα, βασικός μου στόχος ήταν η τοπική εκτέλεση του μοντέλου, ώστε να εξασφαλίζεται η προστασία των προσωπικών δεδομένων των χρηστών.

Μέσα από αυτή τη διαδικασία απέκτησα σημαντικές γνώσεις, καθώς ήρθα σε επαφή με τεχνολογίες αιχμής όπως η προσέγγιση RAG. Η εμπειρία αυτή με βοήθησε να κατανοήσω πώς μπορούμε να λύνουμε πραγματικά προβλήματα συνδυάζοντας τον προγραμματισμό με τη χρήση μοντέλων τεχνητής νοημοσύνης.

Περίληψη

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος «εικονικού διδάσκοντα» (virtual tutor), με σκοπό την εξατομικευμένη υποστήριξη των φοιτητών στο μάθημα «Αντικειμενοστραφής Προγραμματισμός». Η εφαρμογή αξιοποιεί τεχνολογίες αιχμής στον τομέα της Τεχνητής Νοημοσύνης, προσφέροντας ένα διαδραστικό περιβάλλον ερωταποκρίσεων πάνω στο επίσημο εκπαιδευτικό υλικό του μαθήματος.

Για την επίτευξη υψηλής ακρίβειας και τον περιορισμό των εσφαλμένων απαντήσεων (hallucinations), υιοθετήθηκε η προσέγγιση Retrieval-Augmented Generation (RAG). Το σύστημα επεξεργάζεται υλικό σε μορφή αρχείων PDF και το αποθηκεύει ως διανυσματικές αναπαραστάσεις (embeddings) στη βάση δεδομένων ChromaDB. Κατά την υποβολή μιας ερώτησης, πραγματοποιείται νοηματική αναζήτηση (similarity search) για την ανάκτηση των σχετικών αποσπασμάτων, τα οποία στη συνέχεια τροφοδοτούνται στο μεγάλο γλωσσικό μοντέλο KriKri (8B), που βασίζεται στην αρχιτεκτονική Llama-3.1. Η υλοποίηση βασίστηκε στη γλώσσα Python (FastAPI) και σε τεχνολογίες HTML/CSS/JS, με το μοντέλο να εκτελείται τοπικά για τη διασφάλιση της ιδιωτικότητας των χρηστών.

Development of an Interactive Educational Assistant using Large Language Models and Retrieval-Augmented Generation: Application in the Object-Oriented Programming Course

Petsalakis Vasilis

Abstract

The subject of this undergraduate thesis is the design and implementation of a "virtual tutor" system, designed to provide personalized learning support for students in the "Object-Oriented Programming" course. The application leverages cutting-edge Artificial Intelligence technologies, offering an interactive Question-Answering environment based on the official educational material of the course. To achieve high accuracy and mitigate the phenomenon of "hallucinations," the Retrieval-Augmented Generation (RAG) approach was adopted. The system processes educational material in PDF format and stores it as vector representations (embeddings) in a ChromaDB database. Upon submitting a query, a semantic similarity search is performed to retrieve relevant excerpts, which are then fed into the KriKri (8B) Large Language Model, based on the Llama-3.1 architecture. The implementation was developed using Python (FastAPI) and HTML/CSS/JS technologies, with the model running locally to ensure user privacy and data independence.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες στον επιβλέποντα καθηγητή μου, κ. Αδαμίδη, για την εμπιστοσύνη που μου έδειξε αναθέτοντάς μου αυτό το θέμα, καθώς και για την πολύτιμη καθοδήγηση και την επιστημονική του υποστήριξη σε όλη τη διάρκεια εκπόνησης της εργασίας.

Επίσης, ευχαριστώ την οικογένεια και τους φίλους μου για την ηθική υποστήριξη και την ενθάρρυνση που μου παρείχαν μέχρι την ολοκλήρωση των σπουδών μου.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες

AI	Artificial Intelligence (Τεχνητή Νοημοσύνη)
AIED	Artificial Intelligence in Education
ANN	Approximate Nearest Neighbor
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
BGE	BAAI General Embedding
BM25	Best Matching 25
CBOW	Continuous Bag-of-Words
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DPR	Dense Passage Retrieval
FAISS	Facebook AI Similarity Search
GGUF	GPT-Generated Unified Format
GloVe	Global Vectors for Word Representation
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
HNSW [13]	Hierarchical Navigable Small World
HTML	HyperText Markup Language
JSON	JavaScript Object Notation
k-NN	k-Nearest Neighbors
LLM	Large Language Model (Μεγάλο Γλωσσικό Μοντέλο)
LSTM	Long Short-Term Memory
MTEB	Massive Text Embedding Benchmark
NLP	Natural Language Processing
PDF	Portable Document Format
RAG	Retrieval-Augmented Generation
RAM	Random Access Memory
REST	Representational State Transfer
RNN	Recurrent Neural Network
SBERT	Sentence-BERT
UI	User Interface

UUID	Universally Unique Identifier
VRAM	Video Random Access Memory

Κεφάλαιο 1ο: Εισαγωγή

1.1 Σκοπός, στόχοι, κίνητρο και σημασία του θέματος

Η μεγάλη ανάπτυξη των Μεγάλων Γλωσσικών Μοντέλων (Large Language Models – LLMs) τα τελευταία χρόνια έχει ανοίξει νέες προοπτικές σε ένα ευρύ φάσμα επιστημονικών και εφαρμοσμένων τομέων, με την τριτοβάθμια εκπαίδευση να συγκαταλέγεται στους πλέον υποσχόμενους χώρους εφαρμογής. Η ικανότητα των LLMs να κατανοούν τη φυσική γλώσσα, να παράγουν συνεκτικό κείμενο και να υποστηρίζουν διαδραστικά περιβάλλοντα ερωταποκρίσεων τα καθιστά κατάλληλα εργαλεία για την ανάπτυξη συστημάτων εκπαιδευτικής υποστήριξης. Ταυτόχρονα, φαίνεται να μπορούν να προσφέρουν εξατομικευμένη και ευέλικτη μάθηση στο πανεπιστημιακό περιβάλλον, τομέας στον οποίο η ανθρώπινη καθοδήγηση αποτελεί σπάνιο πόρο. Η αξιοποίηση τέτοιων τεχνολογιών θεωρείται ιδιαιτέρως χρήσιμη για τον εκσυγχρονισμό των μεθόδων υποστήριξης της μάθησης [1].

Η παρούσα πτυχιακή εργασία αναπτύσσεται εντός αυτού του πλαισίου και εστιάζει στο μάθημα «Αντικειμενοστραφής Προγραμματισμός» του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων, το οποίο αξιοποιείται ως γνωστική βάση για την υλοποίηση ενός λειτουργικού δείγματος συστήματος. Κεντρικό κίνητρο αποτέλεσε η διαπίστωση ότι οι φοιτητές αντιμετωπίζουν συχνά δυσκολίες στην κατανόηση αφηρημένων εννοιών του προγραμματισμού, όπως η κληρονομικότητα, ο πολυμορφισμός ή η αφαιρετικότητα, ενώ η πρόσβαση σε άμεση και εξατομικευμένη υποστήριξη από τους διδάσκοντες παραμένει περιορισμένη λόγω της δυσμενούς αναλογίας φοιτητών προς καθηγητές που επικρατεί στα ελληνικά πανεπιστήμια. Το πρόβλημα δεν είναι, βεβαίως, τοπικό: η ερευνητική κοινότητα έχει επανειλημμένως επισημάνει την ανάγκη για συστήματα ικανά να παρέχουν έγκαιρη και στοχευμένη βοήθεια στους φοιτητές κατά τη διάρκεια της αυτόνομης μελέτης τους [1], [2].

Σκοπός της εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός διαδικτυακού συστήματος εικονικού διδάσκοντα (virtual tutor), το οποίο αξιοποιεί τεχνικές Retrieval-Augmented Generation (RAG) σε συνδυασμό με ένα ανοικτού κώδικα Μεγάλο Γλωσσικό Μοντέλο που εκτελείται αποκλειστικά σε τοπικό εξοπλισμό. Το σύστημα δέχεται ως είσοδο το επίσημο εκπαιδευτικό υλικό του μαθήματος σε μορφή αρχείων PDF και απαντά σε ερωτήματα φοιτητών με τρόπο δομημένο και εκπαιδευτικά κατάλληλο, επιδιώκοντας να προσεγγίσει λειτουργικά τον ρόλο ενός ανθρώπινου διδάσκοντα. Η επιλογή της τοπικής εκτέλεσης του μοντέλου, αντί της χρήσης εμπορικών «APIs», εξυπηρετεί τρεις επιμέρους στόχους: τη διασφάλιση της ιδιωτικότητας των δεδομένων του χρήστη, τον πλήρη έλεγχο της συμπεριφοράς του συστήματος και τη δυνατότητα αναπαραγωγής των αποτελεσμάτων χωρίς εξάρτηση από υπηρεσίες τρίτων.

Οι επιμέρους στόχοι της εργασίας συνοψίζονται ως εξής: πρώτον, η σχεδίαση και υλοποίηση ενός πλήρους RAG pipeline που να συμπεριλαμβάνει την επεξεργασία εκπαιδευτικού υλικού, την ευρετηρίασή του σε διανυσματική βάση δεδομένων, την ανάκτηση σχετικών αποσπασμάτων με βάση τα ερωτήματα του χρήστη και την παραγωγή απαντήσεων με γνώμονα το ανακτημένο περιεχόμενο· δεύτερον, η επιλογή και παραμετροποίηση κατάλληλων ανοικτού κώδικα μοντέλων embedding και γλωσσικής παραγωγής, ικανών να χειριστούν αποδοτικά περιεχόμενο στην ελληνική γλώσσα· τρίτον, η ανάπτυξη ενός φιλικού προς τον χρήστη διαδικτυακού γραφικού περιβάλλοντος αλληλεπίδρασης· και τέταρτον, η συστηματική αξιολόγηση της απόδοσης του συστήματος με έμφαση στη συμβολή του RAG και της στρατηγικής τεμαχισμού του κειμένου στην ποιότητα των παραγόμενων απαντήσεων.

Ο κεντρικός ερευνητικός στόχος αφορά τη διερεύνηση της αποτελεσματικότητας μιας τέτοιας προσέγγισης ως προς την παροχή αξιόπιστης εκπαιδευτικής υποστήριξης. Πιο συγκεκριμένα, εξετάζονται οι δυνατότητες αλλά και οι περιορισμοί των σύγχρονων τεχνολογιών επεξεργασίας φυσικής γλώσσας σε ένα πραγματικό εκπαιδευτικό σενάριο, καθώς και η συμβολή των τεχνικών ανάκτησης πληροφορίας στη βελτίωση της ποιότητας των απαντήσεων, μέσω ενός πρωτοκόλλου αξιολόγησης που συγκρίνει συστηματικά διαφορετικές εκδοχές του συστήματος.

Η εργασία οργανώνεται γύρω από τρεις ερευνητικές ερωτήσεις, οι οποίες εξετάζονται στο Κεφάλαιο 5 και στα Συμπεράσματα. Η πρώτη αφορά την εφικτότητα της υλοποίησης. Συγκεκριμένα, εξετάζεται αν είναι δυνατή η ανάπτυξη ενός ελληνόφωνου εκπαιδευτικού βοηθού βασισμένου σε RAG, ο οποίος να εκτελείται σε τυπικό desktop εξοπλισμό χωρίς χρήση εμπορικών APIs. Η δεύτερη αφορά τη συμβολή των σχεδιαστικών επιλογών στην ποιότητα των απαντήσεων. Εξετάζεται δηλαδή σε ποιο βαθμό η ενσωμάτωση της προσέγγισης RAG και η επιλογή στρατηγικής τεμαχισμού του εκπαιδευτικού υλικού επηρεάζουν την ποιότητα των απαντήσεων του συστήματος. Η τρίτη αφορά την αξιοπιστία του συστήματος σε εκπαιδευτικό πλαίσιο. Η ερώτηση εστιάζει στο αν διατηρεί το σύστημα την αξιοπιστία του όταν αντιμετωπίζει ερωτήματα εκτός του εκπαιδευτικού του πλαισίου, ή παράγει ακατάλληλες απαντήσεις από τη γενική παραμετρική του γνώση. Οι τρεις αυτές ερωτήσεις καθοδηγούν τον σχεδιασμό του πειραματικού πρωτοκόλλου στο Κεφάλαιο 5 και αποτελούν τη βάση της συζήτησης των ευρημάτων στο Κεφάλαιο 6.

Η σημασία του θέματος αναδεικνύεται από περισσότερες της μιας οπτικές γωνίες. Από τεχνολογική άποψη, η εργασία αντιμετωπίζει ένα από τα κεντρικά προβλήματα των σύγχρονων LLMs, δηλαδή την τάση τους να παράγουν μη τεκμηριωμένες ή ακόμη και πραγματολογικά εσφαλμένες απαντήσεις, φαινόμενο γνωστό στη διεθνή βιβλιογραφία ως «ψευδαίσθηση» (hallucination). Το φαινόμενο αυτό είναι ιδιαίτερα προβληματικό στο εκπαιδευτικό πλαίσιο, όπου η ακρίβεια της πληροφορίας είναι σημαντική και λανθασμένες απαντήσεις μπορούν να οδηγήσουν σε παρανοήσεις τις οποίες ο φοιτητής δύσκολα αναγνωρίζει και διορθώνει στη συνέχεια [2], [6].

Σε εκπαιδευτικό επίπεδο, η εργασία συνεισφέρει στον διαρκώς αναπτυσσόμενο κλάδο της Τεχνητής Νοημοσύνης στην Εκπαίδευση (Artificial Intelligence in Education – AIED), προσφέροντας μία συγκεκριμένη υλοποίηση η οποία μπορεί να αξιολογηθεί, να κριθεί και να επεκταθεί σε άλλα γνωστικά αντικείμενα με ελάχιστες τροποποιήσεις. Πρακτική αξία έχει επίσης η αποκλειστική χρήση εργαλείων ανοικτού κώδικα και η τοπική εκτέλεση του γλωσσικού μοντέλου, η οποία διασφαλίζει ανεξαρτησία από εμπορικές υπηρεσίες, προστασία της ιδιωτικότητας και εύκολη αναπαραγωγή των αποτελεσμάτων.

1.2 Μεγάλα Γλωσσικά Μοντέλα στην Εκπαίδευση

Τα τελευταία χρόνια, η ενσωμάτωση των Μεγάλων Γλωσσικών Μοντέλων στην εκπαιδευτική διαδικασία αποτελεί ένα από τα ταχύτερα αναπτυσσόμενα ερευνητικά πεδία στον χώρο της επιστήμης υπολογιστών και της μηχανικής μάθησης, με αυξανόμενη δημοσιευτική δραστηριότητα τα τελευταία έτη. Σε μια εκτενή επισκόπηση της υπάρχουσας βιβλιογραφίας, οι Wang et al. [1] καταγράφουν τις εφαρμογές των LLMs στην εκπαίδευση, ταξινομώντας τες σε τέσσερις βασικές κατηγορίες: υποστήριξη φοιτητών, υποστήριξη διδασκόντων, προσαρμοστική μάθηση και εμπορικά εργαλεία. Τα ευρήματα της επισκόπησης αναδεικνύουν ότι τα LLMs έχουν ήδη επιδείξει αξιόλογες επιδόσεις σε εργασίες όπως η επίλυση προβλημάτων STEM σε επίπεδο δευτεροβάθμιας και τριτοβάθμιας εκπαίδευσης, η διόρθωση γραμματικών λαθών σε κείμενα και η αυτόματη παραγωγή εκπαιδευτικού υλικού. Ταυτόχρονα, οι ίδιοι συγγραφείς εντοπίζουν σημαντικές προκλήσεις οι οποίες σχετίζονται με

την αξιοπιστία των απαντήσεων, την εκπαιδευτική τους καταλληλότητα και ζητήματα ιδιωτικότητας που εγείρονται κατά τη χρήση τέτοιων συστημάτων από ανηλίκους ή εν γένει ευάλωτους χρήστες.

Ένα από τα πρώτα και πλέον σημαντικά έργα στον χώρο είναι εκείνο των Nye, Mee και Core [2], οι οποίοι εξετάζουν αξιολογικά τις δυνατότητες και τους περιορισμούς των παραγωγικών LLMs για χρήση σε συστήματα διδασκαλίας. Οι ερευνητές επισημαίνουν ότι, παρά την επιφανειακή ομοιότητα ανάμεσα σε μια συνομιλία με το ChatGPT και σε έναν εκπαιδευτικό διάλογο, τα LLMs απέχουν σημαντικά από τον τρόπο λειτουργίας ενός έμπειρου διδάσκοντα. Ο τελευταίος αναλαμβάνει πρωτοβουλία κατά τη συνομιλία, υποβάλλει ερωτήματα αντί να απαντά παθητικά σε αυτά, και διαθέτει παιδαγωγική γνώση του αντικειμένου, η οποία συμπεριλαμβάνει τα συχνά λάθη και τις τυπικές παρανοήσεις των μαθητών, στοιχεία που απουσιάζουν από τα τρέχοντα LLMs. Το εύρημα αυτό τεκμηριώνει εμφατικά την ανάγκη για συστήματα τα οποία συνδυάζουν τις γλωσσικές δυνατότητες των LLMs με εξωτερικές πηγές γνώσης και δομημένη εκπαιδευτική λογική.

Μια πιο εφαρμοσμένη προσέγγιση παρουσιάζουν οι Venugopalan et al. [3], οι οποίοι ανέπτυξαν ένα σύστημα υποστήριξης γονέων σε περιβάλλον υβριδικής διδασκαλίας, συνδυάζοντας την ευελιξία του μοντέλου Llama 3 με την εκπαιδευτική νοημοσύνη ενός συμβατικού Intelligent Tutoring System (ITS). Το σύστημα χρησιμοποιεί δεδομένα επίλυσης προβλημάτων σε πραγματικό χρόνο ως πλαίσιο για την παραγωγή στοχευμένων προτάσεων επικοινωνίας, επιτυγχάνοντας εκπαιδευτικά συνεκτικές αποκρίσεις. Ιδιαίτερα σημαντικό εύρημα της εργασίας αυτής είναι ότι η παροχή δεδομένων από το ITS στο LLM δεν επαρκεί από μόνη της· απαιτείται ταυτόχρονα και η παροχή σαφών οδηγιών και ενδεικτικών παραδειγμάτων σχετικά με τον τρόπο αξιοποίησης των δεδομένων. Το εύρημα αυτό υπογραμμίζει τη σημασία της σχεδίασης προτροπών (prompt engineering) στα εκπαιδευτικά συστήματα που βασίζονται σε LLMs.

Μια ολοκληρωμένη αρχιτεκτονική εικονικού διδάσκοντα προτείνεται από τους Chen et al. [4] στο σύστημα ChatTutor. Η αρχιτεκτονική αυτή αποσυνθέτει τη διδακτική διαδικασία σε τρεις αλληλένδετες λειτουργίες: αλληλεπίδραση, αναστοχασμό και αντίδραση, κάθε μια από τις οποίες υλοποιείται μέσω αλυσίδων LLM με εργαλεία και μονάδων δυναμικής μνήμης. Τα αποτελέσματα της πειραματικής αξιολόγησης δείχνουν ότι η χρήση δομημένης μνήμης και προσαρμοστικού αναστοχασμού βελτιώνει σημαντικά τη συνέπεια και τη σταθερότητα του συστήματος κατά τη διάρκεια μακροχρόνιων αλληλεπιδράσεων. Οι ίδιοι συγγραφείς αναγνωρίζουν, ωστόσο, ότι ακόμη και ένα ισχυρό γλωσσικό μοντέλο υπόκειται σε ψευδαισθήσεις όταν του ζητηθεί να ανακαλέσει συγκεκριμένες πραγματολογικές πληροφορίες, και γι' αυτόν τον λόγο προτείνουν την τεχνική Retrieval-Augmented Generation ως πιθανή λύση για τη σύνδεση των απαντήσεων με αξιόπιστο εξωτερικό περιεχόμενο.

Στη μελέτη επίσης των εφαρμογών των LLMs στην εκπαίδευση, η ερευνητική κοινότητα έχει στρέψει την προσοχή της στην αντιμετώπιση του φαινομένου των ψευδαισθήσεων μέσω τεχνικών ανάκτησης. Οι Shuster et al. [5] έδειξαν ότι η ενσωμάτωση RAG σε συνομιλιακά συστήματα μειώνει δραστικά τις αναξιόπιστες αποκρίσεις, καθώς δεσμεύει το μοντέλο σε επαληθεύσιμο εξωτερικό περιεχόμενο αντί να το αφήνει να βασιστεί αποκλειστικά στην παραμετρική του γνώση. Ωστόσο, όπως τεκμηριώνουν οι Zhang και Zhang σε πρόσφατη ανασκόπηση [6], η αποτελεσματικότητα ενός συστήματος RAG δεν εξαρτάται μόνο από το γλωσσικό μοντέλο αλλά από την ποιότητα κάθε επιμέρους σταδίου του pipeline, από την αρχική ανάκτηση έως την τελική παραγωγή της απάντησης.

Συνολικά, η βιβλιογραφική επισκόπηση αναδεικνύει ότι, παρά την ταχεία πρόοδο στον τομέα, εξακολουθεί να υφίσταται έλλειψη συστημάτων τα οποία συνδυάζουν αποτελεσματικά LLMs με τεχνικές RAG για την εξυπηρέτηση συγκεκριμένων εκπαιδευτικών αναγκών, ιδιαίτερα σε γλώσσες

διαφορετικές από την αγγλική. Η παρούσα εργασία επιχειρεί να καλύψει αυτό ακριβώς το κενό, αναπτύσσοντας ένα σύστημα που ανταποκρίνεται στις ιδιαιτερότητες του ελληνικού εκπαιδευτικού πλαισίου και του ελληνόγλωσσου εκπαιδευτικού υλικού.

1.3 Πρόβλημα που αντιμετωπίζεται

Το πρόβλημα που καλείται να αντιμετωπίσει η παρούσα εργασία εκτείνεται σε δύο αλληλένδετα επίπεδα, ένα εκπαιδευτικό και ένα τεχνικό. Η διάκριση αυτή είναι σημαντική διότι η επιτυχής αντιμετώπιση του πρώτου προϋποθέτει τη συστηματική επίλυση του δεύτερου.

Σε εκπαιδευτικό επίπεδο, το ζήτημα αφορά την έλλειψη άμεσης και εξατομικευμένης υποστήριξης των φοιτητών κατά τη διάρκεια της αυτόνομης μελέτης τους. Σε μαθήματα με υψηλή εγγραφή, όπως ο Αντικειμενοστραφής Προγραμματισμός, η ατομική επικοινωνία του φοιτητή με τον διδάσκοντα είναι πρακτικά αδύνατη σε συνεχή βάση. Ο φοιτητής που αντιμετωπίζει δυσκολία στην κατανόηση μιας έννοιας δεν διαθέτει άμεσο κανάλι ανατροφοδότησης εκτός των προγραμματισμένων ωρών διδασκαλίας, και συχνά αναγκάζεται να καταφύγει σε αναζητήσεις ανοικτού διαδικτύου, όπου η πληροφορία είναι διάσπαρτη, συχνά ανακριβής και δύσκολο να συσχετιστεί με το συγκεκριμένο εκπαιδευτικό πλαίσιο του μαθήματος. Ένα σύστημα εικονικού διδάσκοντα, το οποίο βασίζεται αποκλειστικά στο επίσημο εκπαιδευτικό υλικό του μαθήματος και ανταποκρίνεται σε ερωτήματα με εκπαιδευτικά συνεκτικό τρόπο, μπορεί να αποτελέσει σημαντικό συμπλήρωμα της παραδοσιακής διδασκαλίας, χωρίς να υποκαθιστά τον ρόλο του ανθρώπινου διδάσκοντα [1].

Σε τεχνικό επίπεδο, το κεντρικό πρόβλημα αφορά την αναξιοπιστία των Μεγάλων Γλωσσικών Μοντέλων όταν αυτά χρησιμοποιούνται χωρίς πρόσβαση σε εξωτερικές, επαληθεύσιμες πηγές γνώσης. Τα LLMs βασίζουν τη λειτουργία τους σε στατιστικές αναπαραστάσεις γλωσσικών προτύπων, οι οποίες προκύπτουν κατά τη φάση εκπαίδευσής τους, και δεν διαθέτουν ενσωματωμένο μηχανισμό επαλήθευσης της παραγόμενης πληροφορίας. Ως αποτέλεσμα, παράγουν ενίοτε αποκρίσεις που φαίνονται εκ πρώτης όψεως πειστικές αλλά είναι πραγματολογικά εσφαλμένες, φαινόμενο το οποίο έχει καθιερωθεί να αναφέρεται ως «ψευδαίσθηση» ή hallucination [6]. Στο εκπαιδευτικό πλαίσιο, το φαινόμενο αυτό είναι ιδιαίτερα επικίνδυνο: μια λανθασμένη αλλά φαινομενικά πειστική απάντηση μπορεί να εδραιώσει παρανοήσεις στον φοιτητή αντί να τον βοηθήσει να τις ξεπεράσει [7].

Η τεχνική Retrieval-Augmented Generation (RAG) έχει αναδειχθεί ως η πλέον ενδεδειγμένη προσέγγιση για την αντιμετώπιση αυτού του προβλήματος. Τα αποσπάσματα που ανακτώνται από μια εξωτερική βάση γνώσης λειτουργούν ως «άγκυρα» για τη διαδικασία παραγωγής απάντησης, δεσμεύοντας το μοντέλο στο παρεχόμενο περιεχόμενο και περιορίζοντας δραστικά την παραγωγή μη τεκμηριωμένης πληροφορίας [8]. Τα εμπειρικά δεδομένα επιβεβαιώνουν τη χρησιμότητα αυτής της προσέγγισης: οι Shuster et al. [5] έδειξαν ότι η ενσωμάτωση RAG σε συνομιλιακά συστήματα μειώνει το ποσοστό των ψευδαίσθητων αποκρίσεων σε σημαντικό βαθμό σε σχέση με μοντέλα που λειτουργούν χωρίς ανάκτηση, ενώ η βελτίωση αυτή είναι ακόμη εντονότερη σε θέματα τα οποία βρίσκονται εκτός της κατανομής εκπαίδευσης του μοντέλου.

Η απλή εφαρμογή RAG, ωστόσο, δεν εξαλείφει αυτόματα το πρόβλημα. Οι Zhang και Zhang [6] τεκμηριώνουν διεξοδικά ότι ψευδαισθήσεις μπορούν να εμφανιστούν σε πολλαπλά στάδια ενός RAG pipeline. Κατά τη φάση της ανάκτησης, το πρόβλημα μπορεί να προκύψει λόγω χαμηλής ποιότητας των ανακτηθέντων αποσπασμάτων, σημασιολογικής αναντιστοιχίας ανάμεσα στο ερώτημα και στο ανακτηθέν περιεχόμενο, ή ανεπαρκούς στρατηγικής τεμαχισμού των εγγράφων. Κατά τη φάση της παραγωγής, το πρόβλημα μπορεί να οφείλεται σε σύγκρουση ανάμεσα στην ανακτημένη πληροφορία και στην εσωτερική γνώση του μοντέλου, με το τελευταίο να προτιμά ενίοτε την παραμετρική του

«μνήμη» έναντι του παρεχόμενου context. Ως αποτέλεσμα, ένα αξιόπιστο σύστημα απαιτεί προσεκτική σχεδίαση κάθε επιμέρους σταδίου του pipeline, από την επιλογή στρατηγικής τεμαχισμού των εγγράφων και του μοντέλου embedding έως τη στρατηγική επανακατάταξης (reranking) των αποτελεσμάτων [6], [8].

Το πρόβλημα περιπλέκεται περαιτέρω στο συγκεκριμένο πλαίσιο της παρούσας εργασίας, καθώς το εκπαιδευτικό υλικό είναι στην ελληνική γλώσσα, ενώ η συντριπτική πλειονότητα των διαθέσιμων μοντέλων embedding και γλωσσικής παραγωγής έχει εκπαιδευτεί κυρίως σε αγγλόφωνο περιεχόμενο. Η σημασιολογική αναπαράσταση ελληνικού εξειδικευμένου κειμένου αποτελεί από μόνη της τεχνική πρόκληση, η οποία επηρεάζει άμεσα την ποιότητα της ανάκτησης και, κατ' επέκταση, την αξιοπιστία των τελικών απαντήσεων. Η ανάπτυξη ενός RAG pipeline που να ανταποκρίνεται αποτελεσματικά σε αυτές τις προκλήσεις, μέσω της αξιοποίησης πολυγλωσσικών μοντέλων embedding και μηχανισμών επανακατάταξης εκπαιδευμένων σε πολυγλωσσικά δεδομένα, αποτελεί τον κεντρικό τεχνικό στόχο της παρούσας εργασίας.

1.4 Δομή της εργασίας

Η παρούσα εργασία είναι οργανωμένη σε έξι κεφάλαια, τα οποία ακολουθούν μια λογική πορεία από το θεωρητικό υπόβαθρο προς τη συγκεκριμένη υλοποίηση και αξιολόγηση.

Το Κεφάλαιο 2 παρέχει το θεωρητικό υπόβαθρο το οποίο είναι απαραίτητο για την κατανόηση του συστήματος που αναπτύχθηκε. Ειδικότερα, παρουσιάζεται η αρχιτεκτονική Transformer, η θεωρία των διανυσματικών αναπαραστάσεων κειμένου από τα κλασικά μοντέλα έως τα σύγχρονα πολυγλωσσικά μοντέλα, οι αρχές λειτουργίας των διανυσματικών βάσεων δεδομένων και της πυκνής ανάκτησης, και η προσέγγιση RAG με έμφαση στις τεχνικές αντιμετώπισης ψευδαισθήσεων.

Το Κεφάλαιο 3 περιγράφει τη σχεδίαση του συστήματος. Παρουσιάζεται το συνολικό αρχιτεκτονικό διάγραμμα και αναλύονται ξεχωριστά τα επιμέρους στάδια του pipeline: η επεξεργασία και ευρετηρίαση του εκπαιδευτικού υλικού, η κατανόηση και αναδιατύπωση των ερωτημάτων του χρήστη, η ανάκτηση και επανακατάταξη των σχετικών αποσπασμάτων, και η παραγωγή της τελικής απάντησης με διαχείριση της μνήμης του διαλόγου σε πολυγύρους συνομιλίας.

Το Κεφάλαιο 4 τεκμηριώνει την υλοποίηση του συστήματος, καλύπτοντας τις επιλογές τεχνολογιών και εργαλείων για κάθε στάδιο, συμπεριλαμβανομένης της συγκριτικής παρουσίασης των εναλλακτικών μοντέλων LLM και embedding που δοκιμάστηκαν κατά την πορεία της ανάπτυξης. Ταυτόχρονα, περιγράφεται το περιβάλλον εκτέλεσης και οι πρακτικοί περιορισμοί που επέβαλε η χρήση τοπικού υπολογιστικού εξοπλισμού.

Το Κεφάλαιο 5 παρουσιάζει την πειραματική αξιολόγηση του συστήματος. Συγκρίνονται πέντε διαφορετικές εκδοχές του pipeline ως προς την επίδραση της ενσωμάτωσης RAG και της στρατηγικής τεμαχισμού στις παραγόμενες απαντήσεις, με βάση ένα πρωτόκολλο αξιολόγησης τριβάθμιας κλίμακας που εφαρμόζεται σε ένα σύνολο αντιπροσωπευτικών ερωτήσεων.

Τέλος, το Κεφάλαιο 6 παρουσιάζει τα κύρια ευρήματα και συμπεράσματα της εργασίας, τη συνεισφορά της στον τομέα των εκπαιδευτικών συστημάτων βασισμένων σε LLMs και προτάσεις για μελλοντική επέκταση και βελτίωση.

Μια τέτοια δομή δεν είναι τυχαία αλλά αντικατοπτρίζει τη λογική πορεία από το αφηρημένο στο συγκεκριμένο: το Κεφάλαιο 2 θέτει το θεωρητικό πλαίσιο, το Κεφάλαιο 3 μετατρέπει το πλαίσιο αυτό σε συγκεκριμένες σχεδιαστικές αποφάσεις, και το Κεφάλαιο 4 υλοποιεί τις αποφάσεις σε πραγματικό κώδικα. Κάθε επόμενο κεφάλαιο προϋποθέτει το προηγούμενο, δημιουργώντας μια συνεκτική

αλληλουχία από το «γιατί» (Κεφ. 1–2) στο «πώς θα σχεδιαστεί» (Κεφ. 3), στο «πώς υλοποιείται» (Κεφ. 4), στο «πόσο αποδίδει» (Κεφ. 5) και τέλος στο «τι μάθαμε» (Κεφ. 6).

Από πλευράς ανάγνωσης, η εργασία μπορεί να προσεγγιστεί είτε γραμμικά από το πρώτο έως το τελευταίο κεφάλαιο, είτε επιλεκτικά ανάλογα με το ενδιαφέρον του αναγνώστη. Ένας αναγνώστης εξοικειωμένος με τη θεωρία των LLMs και των RAG συστημάτων μπορεί να παραλείψει το Κεφάλαιο 2 και να μεταβεί απευθείας στα Κεφάλαια 3 και 4. Αντίστροφα, ένας αναγνώστης που ενδιαφέρεται κυρίως για τη μεθοδολογική αξιολόγηση μπορεί να εστιάσει στο Κεφάλαιο 5. Τα Συμπεράσματα του Κεφαλαίου 6 προσφέρουν μια αυτοτελή σύνοψη των κύριων ευρημάτων για αναγνώστες που αναζητούν μια γρήγορη εικόνα των αποτελεσμάτων χωρίς τις λεπτομέρειες της υλοποίησης.

Κεφάλαιο 2ο: Θεωρητικό Υπόβαθρο

2.1 Μεγάλα Γλωσσικά Μοντέλα (LLMs)

Τα Μεγάλα Γλωσσικά Μοντέλα (Large Language Models, LLMs) αποτελούν σήμερα το κεντρικό πεδίο έρευνας στην Επεξεργασία Φυσικής Γλώσσας και έχουν αναδιαμορφώσει ριζικά τον τρόπο με τον οποίο οι μηχανές αλληλεπιδρούν με τη γλώσσα. Πρόκειται για βαθιά νευρωνικά δίκτυα με δεκάδες έως εκατοντάδες δισεκατομμύρια παραμέτρους τα οποία εκπαιδεύονται σε τεράστιο όγκο κειμένων και αποκτούν γενικευμένες γλωσσικές ικανότητες που εκτείνονται πολύ πέρα από τη στενή λειτουργία της πρόβλεψης της επόμενης λέξης. Οι ικανότητες αυτές περιλαμβάνουν τη συνομιλία σε φυσική γλώσσα, τη συνοπτική περιγραφή μακροσκελών κειμένων, τη μετάφραση ανάμεσα σε γλώσσες, τη συγγραφή και ανάλυση κώδικα, τη λύση μαθηματικών ασκήσεων, καθώς και ένα ευρύ φάσμα εργασιών που παραδοσιακά θεωρούνταν ότι απαιτούν εξειδικευμένη μηχανική για κάθε επιμέρους περίπτωση [17].

Η θεμελιώδης αρχιτεκτονική στην οποία βασίζονται σχεδόν όλα τα σύγχρονα LLMs είναι το μοντέλο Transformer, το οποίο παρουσιάστηκε από τους Vaswani et al. το 2017 [18]. Πριν από την εμφάνιση του Transformer, οι κυρίαρχες αρχιτεκτονικές για επεξεργασία ακολουθιών ήταν τα Recurrent Neural Networks (RNNs) και οι παραλλαγές τους, όπως τα Long Short-Term Memory (LSTM) και Gated Recurrent Unit (GRU) δίκτυα. Τα μοντέλα αυτά επεξεργάζονταν το κείμενο σειριακά, ένα token τη φορά, δημιουργώντας δύο θεμελιώδη προβλήματα. Το πρώτο είναι η αδυναμία αποτελεσματικής αποτύπωσης μακρινών εξαρτήσεων μέσα στο κείμενο, καθώς η πληροφορία των πρώτων tokens υπέφερε από το φαινόμενο της εξαφάνισης κλίσεων (vanishing gradients) αφού μεταδιδόταν σε πολλά βήματα. Το δεύτερο είναι η αδυναμία παραλληλοποίησης της εκπαίδευσης, καθώς κάθε βήμα εξαρτιόταν από την έξοδο του προηγούμενου, περιορίζοντας σοβαρά την ταχύτητα εκμάθησης σε μεγάλα σώματα δεδομένων.

Η κρίσιμη καινοτομία του Transformer είναι η αντικατάσταση του σειριακού μηχανισμού των RNNs από έναν μηχανισμό αυτοπροσοχής (self-attention). Ο μηχανισμός αυτός επιτρέπει σε κάθε token να λαμβάνει υπόψη όλα τα υπόλοιπα tokens της ακολουθίας και να υπολογίζει σχέσεις μεταξύ τους ανεξάρτητα από την απόστασή τους [18]. Μαθηματικά, κάθε token προβάλλεται σε τρία διανύσματα, ερώτημα (query), κλειδί (key) και τιμή (value). Η σχετικότητά του με τα άλλα tokens υπολογίζεται ως το κλιμακωμένο εσωτερικό γινόμενο του ερωτήματος με τα κλειδιά, στο οποίο εφαρμόζεται συνάρτηση softmax για την παραγωγή βαρών προσοχής. Τα βάρη αυτά χρησιμοποιούνται στη συνέχεια ως συντελεστές για τον υπολογισμό ενός σταθμισμένου αθροίσματος των τιμών, παράγοντας μια νέα αναπαράσταση για κάθε token η οποία ενσωματώνει πληροφορία από ολόκληρη την ακολουθία. Η προσοχή αυτή υλοποιείται συνήθως σε πολλαπλές «κεφαλές» (multi-head attention), οι οποίες επιτρέπουν στο μοντέλο να παρακολουθεί ταυτόχρονα διαφορετικά είδη σχέσεων ανάμεσα στα tokens.

Πέρα από τον μηχανισμό προσοχής, το Transformer περιλαμβάνει θέσεις κωδικοποίησης (positional encodings) οι οποίες εισάγουν πληροφορία για τη σειρά των tokens στην ακολουθία, πληροφορία η οποία θα χανόταν διαφορετικά, δεδομένου ότι η αυτοπροσοχή από μόνη της είναι αμετάθετη. Το μοντέλο οργανώνεται σε πολλαπλά διαδοχικά στρώματα, κάθε ένα από τα οποία περιλαμβάνει έναν μηχανισμό αυτοπροσοχής και ένα feed-forward νευρωνικό δίκτυο εφαρμοζόμενο κατά θέση, συνδεδεμένα με residual connections και layer normalization για τη σταθερότητα της εκπαίδευσης. Η δυνατότητα πλήρους παραλληλοποίησης των υπολογισμών σε όλα τα tokens, σε συνδυασμό με την αποτελεσματική αποτύπωση μακρινών εξαρτήσεων, οδήγησε στη δυνατότητα εκπαίδευσης δραματικά

μεγαλύτερων μοντέλων σε πολύ μεγαλύτερα σύνολα δεδομένων και αποτέλεσε τον τεχνολογικό θεμέλιο όλων των σύγχρονων γλωσσικών μοντέλων.

Λίγα χρόνια μετά την εμφάνιση του Transformer, οι Devlin et al. παρουσίασαν το BERT (Bidirectional Encoder Representations from Transformers) [19], ένα μοντέλο το οποίο εκμεταλλεύτηκε την αρχιτεκτονική Transformer για να εισαγάγει μια νέα προσέγγιση προεκπαίδευσης: τη masked language modeling. Στη μέθοδο αυτή, ένα ποσοστό των tokens της εισόδου αποκρύπτεται τυχαία και το μοντέλο εκπαιδεύεται να τα προβλέψει χρησιμοποιώντας το συνολικό πλαίσιο, τόσο αριστερά όσο και δεξιά της κρυμμένης θέσης. Αυτή η προσέγγιση, γνωστή ως αμφίδρομη (bidirectional), αντικατέστησε την παραδοσιακή μονοκατευθυντική μοντελοποίηση γλώσσας και επέτρεψε στο BERT να παράγει σημαντικά πιο εκφραστικές αναπαραστάσεις κειμένου, επιτυγχάνοντας state-of-the-art αποτελέσματα σε έντεκα διαφορετικές εργασίες NLP κατά την εποχή της δημοσίευσής του [19]. Το BERT καθιέρωσε επίσης το μοτίβο pretrain-then-finetune, στο οποίο ένα μοντέλο εκπαιδεύεται αρχικά σε ένα γενικό σώμα κειμένων χωρίς ετικέτες και στη συνέχεια προσαρμόζεται σε συγκεκριμένες εργασίες μέσω σχετικά μικρών συνόλων δεδομένων με ετικέτες.

Επόμενο κρίσιμο σημείο καμπής υπήρξε η εμφάνιση της οικογένειας GPT (Generative Pre-trained Transformer) της OpenAI, και ιδιαίτερα του GPT-3 των Brown et al. [20]. Το GPT-3, με 175 δισεκατομμύρια παραμέτρους, ήταν δεκαπλάσιο σε μέγεθος από το προηγούμενο μεγαλύτερο μη αραιό γλωσσικό μοντέλο και επέδειξε μια ιδιότητα η οποία δεν είχε παρατηρηθεί σε μικρότερα μοντέλα. Είχε την ικανότητα εκτέλεσης νέων εργασιών μέσω in-context learning, δηλαδή χωρίς καμία ενημέρωση βαρών, αλλά μόνο μέσω της παροχής μερικών παραδειγμάτων στο prompt [20]. Η δυνατότητα αυτή είχε θεμελιώδεις συνέπειες για την εξέλιξη των LLMs. Φάνηκε ότι η κλιμάκωση σε πολύ μεγάλα μεγέθη μοντέλου και δεδομένων οδηγεί σε αναδυόμενες ικανότητες οι οποίες δεν προϋπήρχαν σε μικρότερες κλίμακες και καθιέρωσε τα LLMs ως ευέλικτα εργαλεία γενικής χρησης για την επίλυση προβλημάτων σε συγκεκριμένες εργασίες αντί για εξειδικευμένες εργαλειοθήκες.

Από τότε, το πεδίο έχει γνωρίσει εκρηκτική ανάπτυξη. Με την εμφάνιση εμπορικών μοντέλων όπως το ChatGPT, το Claude και το Gemini, σε συνδυασμό με την παράλληλη ανάπτυξη ανοικτών μοντέλων όπως η οικογένεια Llama της εταιρείας Meta, έχει καταστήσει τα LLMs διαθέσιμα τόσο σε ερευνητικά όσο και σε εμπορικά πλαίσια. Ιδιαίτερα σημαντική για την παρούσα εργασία είναι η οικογένεια Llama 3, και συγκεκριμένα το Llama 3.1, το οποίο παρουσιάστηκε από την ομάδα AI της Meta το 2024 [17]. Η οικογένεια Llama 3.1 περιλαμβάνει μοντέλα διαφορετικών μεγεθών, από 8 έως 405 δισεκατομμύρια παραμέτρους, με εγγενή υποστήριξη πολυγλωσσικής επεξεργασίας, κώδικα, συλλογισμού και χρήσης εργαλείων. Επίσης πολύ σημαντικό είναι ότι διατίθεται ελεύθερα για έρευνα και εμπορική χρήση. Το Llama 3.1-8B συγκεκριμένα αποτελεί τη βάση πάνω στην οποία έχει χτιστεί το ελληνικό μοντέλο Llama-Krikri [16], το οποίο χρησιμοποιείται στην παρούσα εργασία.

Παρά τις εντυπωσιακές ικανότητες των LLMs, υπάρχουν σημαντικοί περιορισμοί οι οποίοι είναι κρίσιμοι για την κατανόηση της επιλογής του RAG ως αρχιτεκτονικής. Ο πρώτος και σημαντικότερος είναι το φαινόμενο των ψευδαισθήσεων (hallucinations). Τα LLMs μπορούν να παράγουν κείμενο το οποίο φαίνεται εύλογο και συνεκτικό, αλλά περιέχει πραγματολογικά εσφαλμένες πληροφορίες οι οποίες δεν υποστηρίζονται από την εκπαίδευσή τους [21]. Το φαινόμενο αυτό είναι ιδιαίτερα προβληματικό στο εκπαιδευτικό πλαίσιο, όπου η ακρίβεια της πληροφορίας είναι πολύ σημαντική. Ο δεύτερος περιορισμός είναι η στατική φύση της γνώσης τους. Πρακτικά, όταν ολοκληρωθεί η εκπαίδευση ενός LLM, η γνώση του «παγώνει» στην ημερομηνία εκείνη και δεν μπορεί να ενσωματώσει μεταγενέστερες πληροφορίες χωρίς πλήρη επανεκπαίδευση, μια διαδικασία η οποία είναι απαγορευτικά δαπανηρή. Ο τρίτος περιορισμός είναι οι τεράστιες υπολογιστικές απαιτήσεις. Η

εκπαίδευση ενός σύγχρονου LLM απαιτεί χιλιάδες κάρτες γραφικών να λειτουργούν ταυτόχρονα επί μήνες, ενώ ακόμη και η εκτέλεσή του σε παραγωγικό περιβάλλον προϋποθέτει εξειδικευμένη υποδομή. Ο τέταρτος περιορισμός είναι αυτός του context window. Κάθε μοντέλο έχει ένα πεπερασμένο μέγεθος εισόδου, που περιορίζει την ποσότητα πληροφορίας η οποία μπορεί να παρασχεθεί σε μία κλήση.

Ένα τελευταίο ζήτημα που είναι ιδιαίτερα σημαντικό για την παρούσα εργασία αφορά την απόδοση των LLMs σε γλώσσες πέραν της αγγλικής. Τα περισσότερα LLMs εκπαιδεύονται κυρίως σε αγγλικά δεδομένα, με αποτέλεσμα η απόδοσή τους σε γλώσσες λιγότερο δημοφιλείς όπως τα ελληνικά, να είναι σαφώς χαμηλότερη. Το ζήτημα αυτό έχει κινητοποιήσει την ανάπτυξη εξειδικευμένων μοντέλων για συγκεκριμένες γλώσσες, όπως το Meltemi και στη συνέχεια το Krikri, τα οποία αναπτύχθηκαν από το Ινστιτούτο Επεξεργασίας του Λόγου (ΙΕΛ) του Ερευνητικού Κέντρου Αθηνά ειδικά για την ελληνική γλώσσα [16]. Το Krikri, το οποίο χρησιμοποιείται στο παρόν σύστημα, βασίζεται στο Llama 3.1-8B και εκπαιδεύεται επιπλέον σε ένα μεγάλο όγκο ελληνικών κειμένων, επιτυγχάνοντας σημαντικά καλύτερη απόδοση στα ελληνικά σε σύγκριση με τα γενικά μοντέλα του ίδιου μεγέθους.

Ένα τελευταίο πρακτικό ζήτημα, κρίσιμο για την τοπική εκτέλεση LLMs σε μη εξειδικευμένο εξοπλισμό, είναι η τεχνική της κβάντωσης (quantization). Στη μορφή που εκπαιδεύεται ένα γλωσσικό μοντέλο, κάθε παράμετρός του αποθηκεύεται τυπικά ως αριθμός κινητής υποδιαστολής 16 ή 32 bits, πράγμα που καθιστά τα μοντέλα δεκάδων δισεκατομμυρίων παραμέτρων πολύ απαιτητικά σε μνήμη για τον μέσο υπολογιστή. Η κβάντωση είναι μια τεχνική συμπίεσης κατά την οποία οι παράμετροι μετατρέπονται σε αναπαραστάσεις χαμηλότερης ακρίβειας (συνήθως 4 ή 8 bits), με αποτέλεσμα δραματική μείωση του απαιτούμενου όγκου μνήμης, τυπικά της τάξης του τετραπλάσιου για κβάντωση 4 bits, με σχετικά μικρή απώλεια στην ποιότητα των απαντήσεων. Η μορφή είναι η GGUF (GPT-Generated Unified Format), η οποία έχει καθιερωθεί στο οικοσύστημα του llama.cpp, παρέχει μια τυποποιημένη δομή αρχείου για την αποθήκευση quantized μοντέλων μαζί με τα μεταδεδομένα τους (tokenizer, αρχιτεκτονικές παράμετροι, πρότυπα chat). Με αυτό τον τρόπο επιτρέπουν την εύκολη φόρτωση και εκτέλεση τους με ελάχιστο προγραμματιστικό κόπο. Η συγκεκριμένη εκδοχή που χρησιμοποιείται στην παρούσα εργασία είναι η Q4_K_M, μια βελτιωμένη μεταβλητή 4-bit quantization, η οποία εφαρμόζει διαφορετικές στρατηγικές σε διαφορετικά τμήματα του μοντέλου για βέλτιστη ισορροπία μεταξύ μεγέθους και ποιότητας.

2.2 Διανυσματικές Αναπαραστάσεις (Embeddings)

Οι διανυσματικές αναπαραστάσεις λέξεων και κειμένων, γνωστές ως embeddings, αποτελούν έναν από τους θεμελιώδεις πυλώνες της σύγχρονης Επεξεργασίας Φυσικής Γλώσσας. Η βασική ιδέα είναι η αντιστοίχιση διακριτών μονάδων γλώσσας, λέξεων, προτάσεων ή ολόκληρων κειμένων σε διανύσματα πραγματικών αριθμών σταθερής διάστασης, με τέτοιο τρόπο ώστε σημασιολογικά παρόμοιες μονάδες να βρίσκονται κοντά στον διανυσματικό χώρο. Αυτή η αναπαράσταση είναι κρίσιμη καθώς τα νευρωνικά δίκτυα απαιτούν αριθμητικές εισόδους, και η απλοϊκή κωδικοποίηση one-hot (όπου κάθε λέξη αναπαρίσταται ως ένα διάνυσμα με ένα μοναδικό μη μηδενικό στοιχείο) αποτυγχάνει να αποτυπώσει οποιαδήποτε σχέση μεταξύ των λέξεων. Επιπροσθέτως, όλα τα ζεύγη λέξεων είναι εξίσου «μακρινά» υπό την ευκλείδεια απόσταση, ανεξάρτητα από τη σημασιολογική τους συγγένεια.

Η μαθηματική πρόοδος στον τομέα αυτόν ξεκίνησε ουσιαστικά με την εργασία των Mikolov et al. το 2013, οι οποίοι παρουσίασαν το Word2Vec, δύο απλές αλλά εξαιρετικά αποτελεσματικές

αρχιτεκτονικές για την παραγωγή διανυσματικών αναπαραστάσεων λέξεων από μεγάλα σώματα κειμένων [22]. Η πρώτη αρχιτεκτονική, ονομαζόμενη Continuous Bag-of-Words (CBOW), εκπαιδεύει ένα ρηχό νευρωνικό δίκτυο να προβλέπει μια κεντρική λέξη δοθέντος του γλωσσικού της πλαισίου (των γειτονικών λέξεων). Η δεύτερη, γνωστή ως Skip-gram, εκπαιδεύει το δίκτυο να κάνει το αντίθετο, να προβλέπει δηλαδή τις γειτονικές λέξεις δοθείσας της κεντρικής. Και στις δύο περιπτώσεις, τα βάρη του κρυφού επιπέδου του δικτύου χρησιμοποιούνται στο τέλος ως οι τελικές διανυσματικές αναπαραστάσεις των λέξεων.

Το εντυπωσιακό εύρημα της εργασίας ήταν ότι τα παραγόμενα embeddings αποτύπωναν όχι μόνο τη σημασιολογική γειτνίαση, αλλά και γραμμικές σχέσεις μεταξύ εννοιών [22]. Η πλέον χαρακτηριστική ιδιότητα που δημοσιοποιήθηκε ευρέως ήταν η σχέση τύπου «βασιλιάς − άντρας + γυναίκα ≈ βασίλισσα», η οποία δείχνει ότι ο διανυσματικός χώρος κωδικοποιεί αφηρημένες σχέσεις (όπως το γένος ή η ιδιότητα της βασιλείας) ως σταθερές μεταθέσεις στον χώρο. Η ιδιότητα αυτή ήταν αναπάντεχη και θεμελίωσε την ιδέα ότι η κατανομή των λέξεων στον διανυσματικό χώρο έχει σημασιολογική δομή πέρα από τη στενή έννοια της ομοιότητας. Το Word2Vec, σε συνδυασμό με τις παραλλαγές του και με άλλες σύγχρονες προσεγγίσεις όπως το GloVe, αποτέλεσαν για αρκετά χρόνια τη βάση για πλήθος εφαρμογών NLP.

Ωστόσο, τα παραδοσιακά embeddings είχαν το θεμελιώδες μειονέκτημα ότι ήταν στατικά. Κάθε λέξη αντιστοιχούσε σε ένα και μοναδικό διάνυσμα, ανεξάρτητα από το γλωσσικό πλαίσιο στο οποίο εμφανιζόταν. Έτσι, για παράδειγμα η λέξη «bank», η οποία έχει πολλές σημασίες (με την έννοια της τράπεζας ή της όχθης ποταμού), λάμβανε την ίδια αναπαράσταση και στις δύο περιπτώσεις, με αποτέλεσμα η σημασιολογική πληροφορία να είναι μέση μεταξύ των δύο σημασιών και άρα ελλιπής και για τις δύο. Η λύση στο πρόβλημα αυτό ήρθε με την εμφάνιση των συμφραζόμενων (contextual) embeddings, κυρίως μέσω του BERT [19]. Πλέον, κάθε εμφάνιση μιας λέξης στο κείμενο λαμβάνει τη δική της, εξαρτώμενη από το συγκεκριμένο περιβάλλον, αναπαράσταση, παραγόμενη από το πλήρες Transformer δίκτυο το οποίο «βλέπει» ολόκληρη την πρόταση κατά την κωδικοποίηση. Οι contextual αναπαραστάσεις αυτού του είδους είναι σημαντικά πιο εκφραστικές και αποτελούν τη βάση όλων των σύγχρονων μοντέλων embedding.

Λίγο μετά την εμφάνιση του Word2Vec, οι Pennington et al. παρουσίασαν το GloVe (Global Vectors for Word Representation), ένα εναλλακτικό μοντέλο το οποίο συνδύαζε τα πλεονεκτήματα της καθολικής matrix factorization με τις τοπικές προσεγγίσεις context window [25]. Σε αντίθεση με το Word2Vec, το οποίο εκπαιδεύεται αποκλειστικά μέσω πρόβλεψης γειτονικών λέξεων, το GloVe εκμεταλλεύεται απευθείας τις καθολικές στατιστικές συνεμφάνισης λέξεων σε όλο το σώμα κειμένων μέσω ενός log-bilinear regression μοντέλου. Το αποτέλεσμα ήταν διανυσματικές αναπαραστάσεις με συγκρίσιμη ή βελτιωμένη απόδοση σε εργασίες αναλογίας και σημασιολογικής ομοιότητας, οι οποίες για αρκετά χρόνια χρησιμοποιήθηκαν εκτενώς σε πληθώρα εφαρμογών NLP πριν την επικράτηση των συμφραζόμενων αναπαραστάσεων που εισήγαγε το BERT.

Ένα επόμενο σημαντικό βήμα αφορούσε τη γενίκευση από embeddings λέξεων σε embeddings ολόκληρων προτάσεων ή παραγράφων. Απλές προσεγγίσεις, όπως ο μέσος όρος των embeddings των λέξεων μιας πρότασης, απέδιδαν ανεπαρκώς σε εργασίες σημασιολογικής ομοιότητας. Η απευθείας χρήση του BERT για τη μέτρηση ομοιότητας δύο προτάσεων ήταν θεωρητικά δυνατή αλλά υπολογιστικά δαπανηρή: η ομαδοποίηση 10.000 προτάσεων θα απαιτούσε περίπου 50 εκατομμύρια forward passes, δηλαδή δεκάδες ώρες εκτέλεσης ακόμη και σε ισχυρό εξοπλισμό.

Η λύση δόθηκε από την εργασία των Reimers και Gurevych το 2019, οι οποίοι παρουσίασαν το Sentence-BERT (SBERT) [23]. Το SBERT είναι μια τροποποίηση του BERT η οποία χρησιμοποιεί

siamese και triplet αρχιτεκτονικές δικτύου για να παράγει σταθερού μεγέθους διανυσματικές αναπαραστάσεις προτάσεων, οι οποίες μπορούν να συγκριθούν απευθείας μέσω cosine similarity. Η προσέγγιση αυτή μειώνει τον υπολογιστικό χρόνο για την εύρεση του πιο όμοιου ζεύγους από περίπου 65 ώρες σε μόλις 5 δευτερόλεπτα, διατηρώντας την ακρίβεια του BERT [23].

Το Sentence-BERT άνοιξε τον δρόμο για μια ολόκληρη οικογένεια μοντέλων embedding προτάσεων, όπως το Universal Sentence Encoder της Google, τα Contriever και DPR [11], και πιο πρόσφατα τα E5, GTE, BGE και άλλα. Τα μοντέλα αυτά εκπαιδεύονται τυπικά με τεχνική contrastive learning, κατά την οποία το μοντέλο «τραβά» σημασιολογικά σχετικά ζεύγη κειμένων κοντά στον διανυσματικό χώρο και «σπρώχνει» άσχετα ζεύγη μακριά. Η εκπαίδευση αυτή γίνεται συνήθως με την χρήση μεγάλων συνόλων δεδομένων ζευγών (query, passage) τα οποία αντλούνται από ποικίλες πηγές όπως ερωτηματολόγια-απαντήσεις, επιστημονικά άρθρα και τις περιλήψεις τους, ζεύγη μεταφρασμένων προτάσεων και άλλα. Η ποιότητα των παραγόμενων embeddings βελτιώνεται σταθερά με το χρόνο και τα σύγχρονα μοντέλα όπως το multilingual-E5 [12] υποστηρίζουν πλέον δεκάδες γλώσσες, συμπεριλαμβανομένων των ελληνικών και επιτυγχάνουν υψηλή απόδοση σε πολλαπλά benchmarks (π.χ. MTEB).

Ένα σημαντικό ζήτημα κατά τη χρήση embeddings για αναζήτηση ομοιότητας είναι η επιλογή της μετρικής. Στην πράξη, η πλέον διαδεδομένη επιλογή είναι η ομοιότητα συνημιτόνου (cosine similarity), η οποία ορίζεται ως το εσωτερικό γινόμενο δύο διανυσμάτων διαιρεμένο με το γινόμενο των μέτρων τους. Η μετρική αυτή έχει την ιδιότητα ότι αποτιμά μόνο τη γωνία μεταξύ των διανυσμάτων αγνοώντας το μέγεθός τους, πράγμα που είναι επιθυμητό σε εφαρμογές σημασιολογικής αναζήτησης καθώς το μέγεθος του embedding ενός κειμένου μπορεί να επηρεάζεται από παράγοντες άσχετους με το νόημα (π.χ. μήκος κειμένου). Όταν τα embeddings είναι L2-κανονικοποιημένα, δηλαδή κάθε διάνυσμα έχει μοναδιαίο μέτρο τότε η cosine similarity ανάγεται απλώς στο εσωτερικό γινόμενο. Αυτό το πράγμα επιτρέπει αποδοτικότερους υπολογισμούς και απλούστερη ενσωμάτωση σε διανυσματικές βάσεις δεδομένων όπως η ChromaDB που χρησιμοποιείται στην παρούσα εργασία.

2.3 Διανυσματικές Βάσεις Δεδομένων

Η εμφάνιση των embeddings και η ανάγκη αποθήκευσης και αναζήτησης μεγάλου όγκου διανυσμάτων σε σύγχρονες εφαρμογές οδήγησε στην ανάπτυξη μιας νέας κατηγορίας συστημάτων αποθήκευσης: των διανυσματικών βάσεων δεδομένων (vector databases). Πρόκειται για εξειδικευμένα συστήματα σχεδιασμένα με μοναδικό σκοπό την αποδοτική αποθήκευση διανυσμάτων υψηλής διάστασης και τη γρήγορη ανάκτηση των πλησιέστερων γειτόνων ενός δοθέντος διανύσματος ερωτήματος. Σε αντίθεση με τις παραδοσιακές σχεσιακές βάσεις δεδομένων, οι οποίες βελτιστοποιούνται για ερωτήματα βάσει ακριβούς αντιστοίχισης τιμών (equality queries), οι διανυσματικές βάσεις βελτιστοποιούνται για αναζήτηση ομοιότητας, δηλαδή την εύρεση των διανυσμάτων που βρίσκονται πιο κοντά σε ένα ερώτημα υπό κάποια μετρική απόστασης.

Το θεμελιώδες πρόβλημα που καλούνται να λύσουν οι διανυσματικές βάσεις είναι το εξής: δοθέντος ενός διανύσματος-ερωτήματος, να βρεθούν γρήγορα τα k πιο «κοντινά» διανύσματα μέσα σε μια συλλογή που μπορεί να περιέχει εκατομμύρια ή δισεκατομμύρια εγγραφές. Αυτό είναι γνωστό ως αναζήτηση k πλησιέστερων γειτόνων (k-Nearest Neighbors, k-NN). Η απλούστερη λύση, η εξαντλητική αναζήτηση (brute-force search), συγκρίνει το ερώτημα με κάθε αποθηκευμένο διάνυσμα ένα-ένα και επιστρέφει τα k πιο όμοια. Αυτή η προσέγγιση είναι ακριβής αλλά έχει γραμμική πολυπλοκότητα ως προς το μέγεθος της συλλογής, καθιστώντας την απαγορευτική για συλλογές

μεγάλης κλίμακας. Σε ένα dataset με ένα εκατομμύριο διανύσματα διάστασης 1024, που είναι τυπικές τιμές για σύγχρονα embedding μοντέλα, μία αναζήτηση απαιτεί περίπου ένα δισεκατομμύριο αριθμητικές πράξεις, χρόνος που δεν είναι αποδεκτός για διαδραστικές εφαρμογές.

Η λύση παρέχεται από την προσεγγιστική αναζήτηση πλησιέστερων γειτόνων (Approximate Nearest Neighbor search, ANN), η οποία θυσιάζει μια μικρή ποσότητα ακρίβειας για δραματικές βελτιώσεις στην ταχύτητα. Αντί να εγγυηθεί την εύρεση των πραγματικά k πλησιέστερων γειτόνων, η ANN επιστρέφει ένα σύνολο διανυσμάτων τα οποία είναι με υψηλή πιθανότητα πολύ κοντά στο ερώτημα, με χρονική πολυπλοκότητα υπογραμμική ή λογαριθμική ως προς το μέγεθος της συλλογής. Στις πρακτικές εφαρμογές, η μικρή απώλεια ακρίβειας είναι συνήθως αμελητέα. Σε ένα RAG σύστημα, η διαφορά μεταξύ των «πραγματικά πλησιέστερων» και των «σχεδόν πλησιέστερων» διανυσμάτων σπάνια επηρεάζει την ποιότητα των τελικών αποτελεσμάτων, καθώς όλα τα διανύσματα σε αυτήν την εγγύτητα είναι σημασιολογικά σχετικά με το ερώτημα.

Ως πλέον διαδεδομένη σύγχρονη προσέγγιση στην ANN αναζήτηση έχει καθιερωθεί ο αλγόριθμος Hierarchical Navigable Small World (HNSW), ο οποίος προτάθηκε από τους Malkov και Yashunin [13]. Ο HNSW χτίζει μια ιεραρχική δομή γραφήματος στην οποία κάθε κόμβος αντιστοιχεί σε ένα διάνυσμα και οι ακμές συνδέουν γειτονικά διανύσματα στον χώρο. Η ιεραρχία αποτελείται από πολλαπλά επίπεδα. Τα πάνω επίπεδα περιέχουν λίγους κόμβους με ακμές μεγάλης εμβέλειας, λειτουργώντας ως «αυτοκινητόδρομοι» για γρήγορη κίνηση στον χώρο, ενώ τα κάτω επίπεδα περιέχουν όλους τους κόμβους με ακμές μικρότερης εμβέλειας για ακριβή τοπική αναζήτηση. Η αναζήτηση ξεκινά από ένα σημείο του ανώτερου επιπέδου και προχωρά προς τα κάτω, σε κάθε βήμα κινούμενη προς τον γείτονα ο οποίος είναι πλησιέστερος στο ερώτημα, έως ότου δεν υπάρχει πλέον καλύτερη κίνηση. Αυτή η προσέγγιση επιτυγχάνει λογαριθμική εμπειρική πολυπλοκότητα αναζήτησης με πολύ υψηλή ακρίβεια ανάκτησης και έχει γίνει το πρότυπο στις σύγχρονες διανυσματικές βάσεις δεδομένων [13].

Σήμερα υπάρχει ένα ευρύ οικοσύστημα διανυσματικών βάσεων, το οποίο καλύπτει διαφορετικές ανάγκες χρήσης. Σε μία άκρη του φάσματος βρίσκονται βιβλιοθήκες χαμηλού επιπέδου όπως η FAISS της Meta, η οποία παρέχει εξαιρετικά αποδοτικούς αλγορίθμους ANN για χρήση σε C++ και Python αλλά χωρίς ενσωματωμένες λειτουργίες αποθήκευσης, replication ή διαχείρισης συλλογών. Στην άλλη άκρη βρίσκονται πλήρη συστήματα αποθήκευσης όπως το Milvus, το Qdrant και το Weaviate, τα οποία προσφέρουν επιπρόσθετες λειτουργίες όπως οριζόντια κλιμάκωση, υποστήριξη μεταδεδομένων, REST APIs και ενσωμάτωση σε σύγχρονες υποδομές cloud. Ενδιάμεσα, συστήματα όπως η ChromaDB η οποία χρησιμοποιείται στην παρούσα εργασία, προσφέρουν μια ισορροπία. Η συγκεκριμένη βάση είναι ελαφριά, ενσωματώνονται εγγενώς σε Python εφαρμογές χωρίς την ανάγκη ξεχωριστού διακομιστή και ταυτόχρονα παρέχει τις κρίσιμες λειτουργίες όπως μόνιμη αποθήκευση, υποστήριξη μεταδεδομένων και αναζήτηση HNSW. Η επιλογή του ανάλογου συστήματος εξαρτάται από το μέγεθος της εφαρμογής, την απαιτούμενη κλιμάκωση και την ευκολία ανάπτυξης.

Ένα σημαντικό σημείο αφορά τις ρίζες της σύγχρονης ANN αναζήτησης. Αν και ο αλγόριθμος HNSW κυριαρχεί στις σημερινές διανυσματικές βάσεις δεδομένων, μεγάλο μέρος της υποκείμενης έρευνας προήλθε από την κοινότητα της αναζήτησης ομοιότητας εικόνων και κυρίως από την εργασία των Johnson et al. με τη βιβλιοθήκη FAISS, η οποία παρουσίασε εξαιρετικά αποδοτικές υλοποιήσεις για αναζήτηση σε συλλογές δισεκατομμυρίων διανυσμάτων τόσο σε CPU όσο και σε GPU [26]. Η FAISS εισήγαγε τεχνικές όπως η product quantization και η βελτιστοποιημένη k-selection οι οποίες παρέχουν σημαντική επιτάχυνση σε σχέση με παραδοσιακές προσεγγίσεις. Αυτό, αποτέλεσε σημείο αναφοράς για την ανάπτυξη πολλών επόμενων συστημάτων. Οι σύγχρονες διανυσματικές βάσεις,

συμπεριλαμβανομένης της ChromaDB ενσωματώνουν συχνά ιδέες από τη FAISS για τη βελτιστοποίηση των εσωτερικών τους δομών, αν και εκθέτουν στον προγραμματιστή ένα πιο υψηλό επίπεδο αφαίρεσης.

Από τεχνικής πλευράς, οι διανυσματικές βάσεις υποστηρίζουν τυπικά διάφορες μετρικές ομοιότητας όπως ευκλείδεια απόσταση, εσωτερικό γινόμενο και cosine similarity. Η επιλογή της κατάλληλης μετρικής είναι άρρηκτα συνδεδεμένη με τον τρόπο εκπαίδευσης του μοντέλου embedding. Κάθε μοντέλο εκπαιδεύεται με μια συγκεκριμένη μετρική στο δίκτυο του και η χρήση διαφορετικής μετρικής κατά την αναζήτηση μπορεί να οδηγήσει σε υποβαθμισμένα αποτελέσματα. Τα περισσότερα σύγχρονα μοντέλα embedding, συμπεριλαμβανομένου του multilingual-E5 που χρησιμοποιείται στην παρούσα εργασία, εκπαιδεύονται με cosine similarity, και επομένως η ίδια μετρική πρέπει να χρησιμοποιείται και κατά την αναζήτηση στη διανυσματική βάση. Πέραν της μετρικής αυτής, οι διανυσματικές βάσεις υποστηρίζουν και φιλτράρισμα βάσει μεταδεδομένων, επιτρέποντας σύνθετα ερωτήματα της μορφής «βρες τα k πιο όμοια διανύσματα τα οποία προέρχονται από το αρχείο X ή έχουν ετικέτα Y», μια λειτουργικότητα η οποία είναι κρίσιμη για εφαρμογές με πολλαπλές πηγές δεδομένων.

2.4 Retrieval-Augmented Generation (RAG)

Το Retrieval-Augmented Generation (RAG) αποτελεί μια αρχιτεκτονική προσέγγιση η οποία συνδυάζει τις δυνατότητες παραγωγής κειμένου των μεγάλων γλωσσικών μοντέλων με τις δυνατότητες αναζήτησης πληροφορίας των διανυσματικών βάσεων. Αυτή η προσέγγιση αναπτύχθηκε ως απάντηση σε συγκεκριμένους περιορισμούς των LLMs, όπως το φαινόμενο των ψευδαισθήσεων, τη στατική φύση της παραμετρικής γνώσης και την αδυναμία ενσωμάτωσης νέας ή εξειδικευμένης πληροφορίας χωρίς εκ νέου εκπαίδευση του μοντέλου [5], [6]. Στον πυρήνα του RAG βρίσκεται μια απλή αλλά ισχυρή ιδέα η οποία λέει ότι αντί το γλωσσικό μοντέλο να βασίζεται αποκλειστικά στη γνώση που έχει αποθηκεύσει στα βάρη του, ενισχύεται κατά τον χρόνο εκτέλεσης με σχετικές πληροφορίες οι οποίες ανακτώνται δυναμικά από μια εξωτερική βάση γνώσης.

Η αρχική εργασία που καθιέρωσε τον όρο RAG και διατύπωσε την έννοια σε συστηματική μορφή ήταν η δημοσίευση των Lewis et al. στο NeurIPS 2020 [9]. Στην εργασία αυτή παρουσιάστηκε μια αρχιτεκτονική η οποία συνδύαζε έναν dense retriever με ένα seq2seq γλωσσικό μοντέλο και εκπαιδεύτηκε από άκρο σε άκρο για διάφορες knowledge-intensive εργασίες. Το θεμελιώδες εύρημα ήταν ότι η ενσωμάτωση εξωτερικής γνώσης μέσω ανάκτησης βελτίωνε σημαντικά την απόδοση σε σύγκριση με καθαρά παραμετρικές προσεγγίσεις, ενώ ταυτόχρονα παρείχε μεγαλύτερη διαφάνεια καθώς οι απαντήσεις μπορούσαν πλέον να ανιχνεύονται πίσω στα συγκεκριμένα έγγραφα από τα οποία προέκυψαν. Η αρχιτεκτονική αυτή άνοιξε τον δρόμο για μια πλήρη νέα κατηγορία συστημάτων η οποία κατά τα επόμενα χρόνια εξελίχθηκε σε ένα από τα πιο ενεργά πεδία έρευνας στο χώρο των LLMs.

Σε ένα τυπικό RAG pipeline, η επεξεργασία ενός ερωτήματος ακολουθεί μια σειρά διακριτών σταδίων τα οποία μαζί συνθέτουν αυτό που συχνά ονομάζεται «pipeline» [10]. Το πρώτο στάδιο είναι η ευρετηρίαση (indexing), η οποία εκτελείται μία φορά εκτός σύνδεσης. Κατά την διαδικασία αυτή, η συλλογή εγγράφων σπάζεται σε μικρότερα τμήματα (chunks), καθένα από τα οποία μετατρέπεται σε διάνυσμα embedding και αποθηκεύεται σε μια διανυσματική βάση. Το δεύτερο στάδιο είναι η ανάκτηση (retrieval), η οποία εκτελείται κατά τον χρόνο της ερώτησης. Το ερώτημα του χρήστη μετατρέπεται σε διάνυσμα και χρησιμοποιείται για την εύρεση των k πιο όμοιων chunks στη βάση. Το

τρίτο στάδιο είναι η επαύξηση (augmentation), κατά την οποία τα ανακτηθέντα chunks συναρμολογούνται σε ένα πλαίσιο και παρέχονται στο γλωσσικό μοντέλο ως μέρος του prompt. Το τέταρτο και τελευταίο στάδιο είναι η παραγωγή (generation), όπου το μοντέλο παράγει την τελική απάντηση λαμβάνοντας υπόψη τόσο το αρχικό ερώτημα όσο και το ανακτημένο πλαίσιο [9], [10].

Η βασική αυτή αρχιτεκτονική, γνωστή ως «naive RAG», αποτελεί το σημείο εκκίνησης αλλά έχει αρκετούς γνωστούς περιορισμούς. Ο πρώτος είναι η χαμηλή ποιότητα της ανάκτησης όταν το ερώτημα είναι σύντομο, ασαφές ή διατυπωμένο διαφορετικά από το εκπαιδευτικό υλικό. Αυτό είναι περιοριστικό γιατί ο χρήστης θα πρέπει πάντα να είναι όσο πιο συγκεκριμένος μπορεί, πράγμα που δείχνει ότι το μοντέλο δεν βγάζει συμπεράσματα από μόνο του. Ο δεύτερος είναι η αδυναμία χειρισμού μεγάλων διαλόγων με πολλές ανταλλαγές μηνυμάτων μεταξύ χρήστη και μοντέλου, όπου τα ερωτήματα εξαρτώνται από προηγούμενες ανταλλαγές. Ο τρίτος είναι η ευαισθησία στην ποιότητα του chunking. Αν τα chunks είναι πολύ μεγάλα, μπορεί να είναι υπερβολικά διάχυτα σημασιολογικά, ενώ αν είναι πολύ μικρά, μπορεί να χάσουν κρίσιμο πλαίσιο. Ο τέταρτος είναι η πιθανή ανεπάρκεια του γλωσσικού μοντέλου να αξιοποιήσει σωστά το παρεχόμενο πλαίσιο, είτε αγνοώντας το και βασιζόμενο στην παραμετρική του γνώση είτε επαναλαμβάνοντάς το μηχανικά χωρίς σύνθεση. Για την αντιμετώπιση αυτών των περιορισμών, προτείνονται αρκετές βελτιώσεις στη βασική αρχιτεκτονική RAG.

Η πρώτη είναι η αναδιατύπωση ερωτήματος (query rewriting), η οποία χρησιμοποιεί το ίδιο το γλωσσικό μοντέλο για να μετατρέψει εξαρτώμενα από context ερωτήματα σε αυτόνομες διατυπώσεις πριν την ανάκτηση, προσέγγιση η οποία έχει μελετηθεί συστηματικά από τους Ma et al. [27] υπό το όνομα «Rewrite-Retrieve-Read» και έχει αποδειχθεί ιδιαίτερα αποτελεσματική σε συστήματα RAG με εκτεταμένους διαλόγους.

Ένα από τα βασικά πλεονεκτήματα του RAG είναι η ικανότητά του να μετριάζει το πρόβλημα των ψευδαισθήσεων των LLMs [5], [21]. Όταν ένα γλωσσικό μοντέλο καλείται να απαντήσει σε μια ερώτηση χωρίς εξωτερικό πλαίσιο, η απάντηση προκύπτει από την παραμετρική γνώση του η οποία μπορεί να είναι εσφαλμένη, ξεπερασμένη ή εντελώς απούσα για εξειδικευμένα θέματα. Στο εκπαιδευτικό πλαίσιο της παρούσας εργασίας, παραδείγματος χάρη, ένα γενικό γλωσσικό μοντέλο μπορεί να δώσει μια ορθή αλλά όχι εξειδικευμένη απάντηση σχετικά με μια έννοια του Αντικειμενοστραφούς Προγραμματισμού, η οποία όμως μπορεί να μη συμβαδίζει με την ακριβή ορολογία και τη σειρά παρουσίασης του μαθήματος. Με τη χρήση RAG, το μοντέλο καλείται να βασίσει την απάντησή του στις σημειώσεις του συγκεκριμένου μαθήματος, εξασφαλίζοντας συνέπεια με το εκπαιδευτικό υλικό το οποίο έχει δει ο φοιτητής. Η εργασία των Shuster et al. [5] έχει δείξει εμπειρικά ότι αυτή η επαύξηση μέσω ανάκτησης μειώνει σημαντικά το ποσοστό ψευδαισθήσεων σε εφαρμογές συνομιλίας, ενώ πιο πρόσφατες εργασίες [6] παρέχουν συστηματική επισκόπηση των διαφόρων τεχνικών μετριασμού που έχουν προταθεί.

Ένα δεύτερο σημαντικό πλεονέκτημα είναι η δυνατότητα επικαιροποίησης της γνώσης χωρίς επανεκπαίδευση του γλωσσικού μοντέλου. Σε ένα παραδοσιακό σύστημα βασισμένο αποκλειστικά σε LLM, η ενσωμάτωση νέας γνώσης απαιτεί είτε πλήρη επανεκπαίδευση είτε fine-tuning, διαδικασίες οι οποίες είναι υπολογιστικά ακριβές και τεχνικά σύνθετες. Σε ένα σύστημα RAG, αντιθέτως, η ενσωμάτωση νέου υλικού απαιτεί μόνο την προσθήκη του στη διανυσματική βάση, μια διαδικασία η οποία είναι ουσιαστικά στιγμιαία σε σύγκριση με την εκπαίδευση. Στο εκπαιδευτικό πλαίσιο, αυτό σημαίνει ότι η επέκταση του συστήματος σε επιπλέον μαθήματα ή η ενημέρωση του υπάρχοντος υλικού με νέες εκδοχές των σημειώσεων μπορεί να γίνει χωρίς καμία παρέμβαση στο ίδιο το γλωσσικό μοντέλο. Επιπλέον, η πηγή κάθε πληροφορίας παραμένει πλήρως ανιχνεύσιμη, καθώς κάθε

απάντηση μπορεί να συνοδευτεί από αναφορά στα συγκεκριμένα chunks από τα οποία προέκυψε, ένα χαρακτηριστικό το οποίο ενισχύει σημαντικά τη διαφάνεια και την εκπαιδευτική αξιοπιστία του συστήματος [8].

Τα τελευταία χρόνια, η εξέλιξη του πεδίου έχει οδηγήσει στη διάκριση μεταξύ τριών γενεών RAG συστημάτων, όπως περιγράφεται στη συστηματική επισκόπηση των Gao et al. [10]. Η πρώτη γενιά, το naive RAG, αφορά τη βασική αρχιτεκτονική που περιγράφηκε παραπάνω. Η δεύτερη γενιά, το advanced RAG, ενσωματώνει τις βελτιώσεις που επίσης αναφέρθηκαν, δηλαδή query rewriting, reranking, βελτιωμένο chunking και prompt engineering. Στην τρίτη γενιά, το modular RAG, αντιμετωπίζει το pipeline ως μια ευέλικτη δομή από ανεξάρτητα, αντικαταστάσιμα modules τα οποία μπορούν να διατάσσονται και να αντικαθίστανται δυναμικά ανάλογα με τις ανάγκες της εφαρμογής. Το σύστημα που σχεδιάστηκε και υλοποιήθηκε στην παρούσα εργασία εμπίπτει στη δεύτερη γενιά καθώς ενσωματώνει τις πλέον καθιερωμένες βέλτιστες πρακτικές query rewriting, dense retrieval, reranking, προσεκτικό prompt engineering και διαχείριση μνήμης συνομιλίας χωρίς να επεκτείνεται στις πιο σύνθετες αρχιτεκτονικές της τρίτης γενιάς.

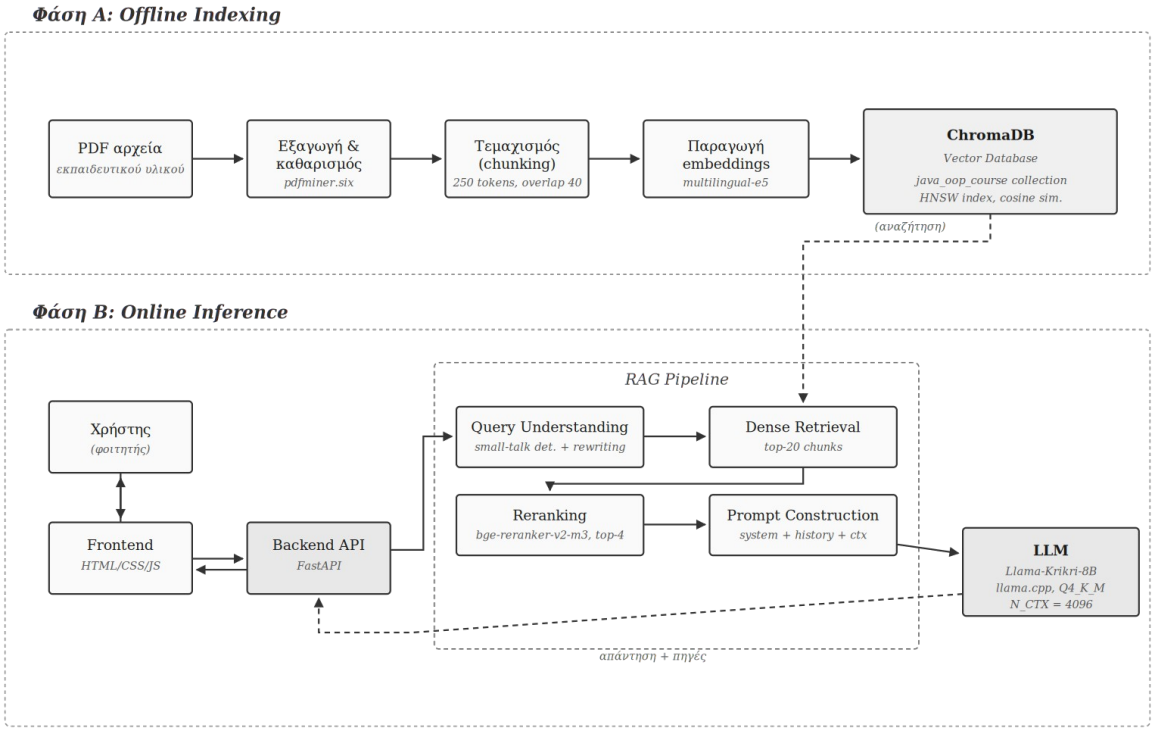
Κεφάλαιο 3ο: Σχεδίαση του Συστήματος

3.1 Αρχιτεκτονική Συστήματος

Στην αρχιτεκτονική του συστήματος εικονικού διδάσκοντα ακολουθεί το καθιερωμένο μοτίβο Retrieval-Augmented Generation (RAG), όπως αυτό αρχικά παρουσιάστηκε από τους Lewis et al. [9] και στη συνέχεια εξελίχθηκε σε μια ευρεία οικογένεια μοντέλων και εργαλείων [10]. Η βασική ιδέα του RAG βρίσκεται στον συνδυασμό δύο συμπληρωματικών μηχανισμών γνώσης: της παραμετρικής γνώσης που βρίσκεται αποθηκευμένη στα βάρη ενός προεκπαιδευμένου γλωσσικού μοντέλου, και της μη παραμετρικής γνώσης που αντλείται δυναμικά από μια εξωτερική βάση εγγράφων κατά τη στιγμή της απόκρισης. Αυτή η επιλογής της προσέγγισης, αντί της αποκλειστικής χρήσης ενός γλωσσικού μοντέλου, αιτιολογείται από την ανάγκη σύνδεσης των παραγόμενων απαντήσεων με συγκεκριμένο, επαληθεύσιμο εκπαιδευτικό υλικό, ένα χαρακτηριστικό το οποίο είναι εξαιρετικά σημαντικό στο εκπαιδευτικό πλαίσιο.

Το σύστημα υλοποιείται ως διαδικτυακή εφαρμογή (web-based) και αποτελείται από τέσσερα βασικά υποσυστήματα τα οποία επικοινωνούν μεταξύ τους μέσω σαφώς καθορισμένων διεπαφών. Το πρώτο υποσύστημα είναι το γραφικό περιβάλλον χρήστη (frontend), το οποίο παρέχει στον φοιτητή ένα φιλικό περιβάλλον συνομιλίας για την υποβολή ερωτημάτων και την ανάγνωση των απαντήσεων. Το δεύτερο είναι το backend API, το οποίο λειτουργεί ως οργανωτής των αιτημάτων: δέχεται τα ερωτήματα από το frontend, τα δρομολογεί στα επιμέρους στάδια επεξεργασίας και επιστρέφει το τελικό αποτέλεσμα. Το τρίτο υποσύστημα είναι το RAG pipeline, το οποίο υλοποιεί την κύρια λογική του συστήματος, δηλαδή την ανάκτηση σχετικών αποσπασμάτων από τη βάση γνώσης και την επανακατάταξή τους. Το τέταρτο και τελευταίο υποσύστημα είναι το ίδιο το Μεγάλο Γλωσσικό Μοντέλο (LLM), το οποίο εκτελείται τοπικά και είναι υπεύθυνο για την παραγωγή της τελικής απάντησης με βάση το ανακτημένο περιεχόμενο.

Η λειτουργία του συστήματος διακρίνεται σε δύο κύρια στάδια: τη φάση ευρετηρίασης (indexing phase), η οποία εκτελείται εκτός σύνδεσης (offline), και τη φάση απόκρισης (inference phase), η οποία εκτελείται σε πραγματικό χρόνο (online) κατά την αλληλεπίδραση με τον χρήστη. Κατά την offline φάση, το εκπαιδευτικό υλικό επεξεργάζεται μία φορά και αποθηκεύεται σε μόνιμη μορφή στη διανυσματική βάση δεδομένων, ενώ κατά την online φάση, το σύστημα δέχεται ερωτήματα χρηστών και παράγει απαντήσεις σε πραγματικό χρόνο. Ο διαχωρισμός αυτός είναι σημαντικός από πλευράς απόδοσης καθώς η επεξεργασία του εκπαιδευτικού υλικού είναι υπολογιστικά δαπανηρή, αλλά πραγματοποιείται μόνο μία φορά. Ενώ αντιθέτως, η απάντηση σε ερωτήματα πρέπει να είναι άμεση. Για την συνολική αρχιτεκτονική και η ροή δεδομένων μεταξύ των υποσυστημάτων απεικονίζονται στο Σχήμα 3.1.



Σχήμα 3.1: Συνολική αρχιτεκτονική του συστήματος

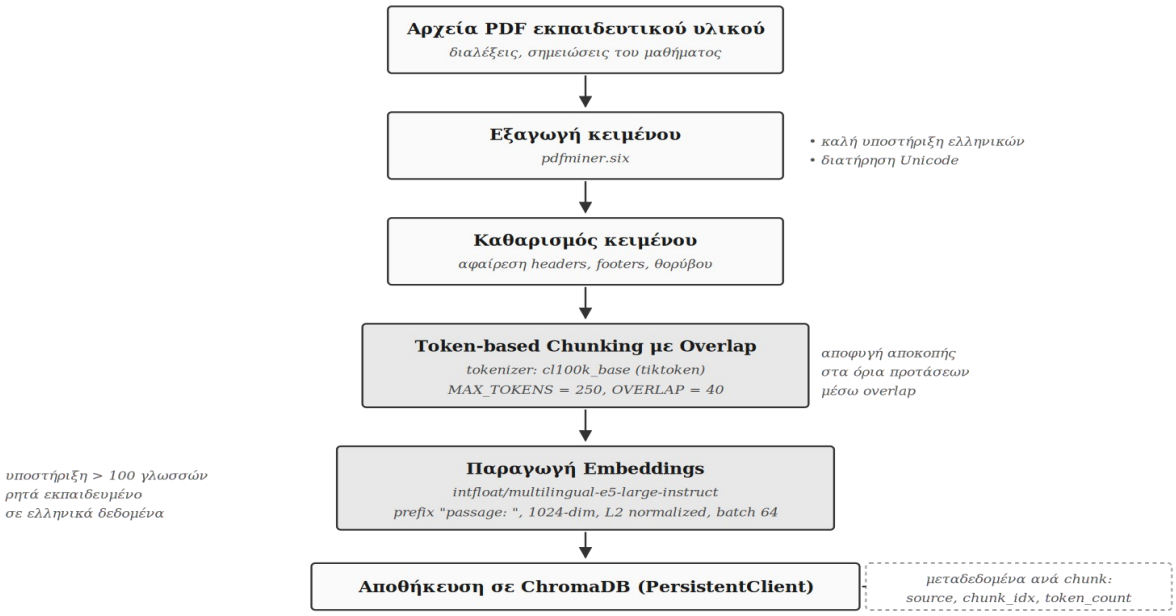
Η σχεδίαση του συστήματος βασίζεται σε τρεις βασικές αρχές. Η πρώτη είναι η τοπική εκτέλεση: ολόκληρη η εφαρμογή, συμπεριλαμβανομένου του γλωσσικού μοντέλου, εκτελείται αποκλειστικά σε τοπικό υπολογιστικό εξοπλισμό, χωρίς κλήσεις σε εξωτερικά εμπορικά APIs. Αυτή η επιλογή εξυπηρετεί την προστασία της ιδιωτικότητας των χρηστών, τον πλήρη έλεγχο της συμπεριφοράς του συστήματος και την ανεξαρτησία από εξωτερικούς παρόχους. Η δεύτερη είναι η αρθρωτή σχεδίαση (modular design): κάθε υποσύστημα μπορεί να αντικατασταθεί ανεξάρτητα από τα υπόλοιπα, διευκολύνοντας την εξέλιξη του συστήματος και την πειραματική αξιολόγηση διαφορετικών εκδοχών του. Η τρίτη είναι η γείωση των απαντήσεων στο παρεχόμενο εκπαιδευτικό υλικό: το γλωσσικό μοντέλο κατευθύνεται συστηματικά να βασίζει τις αποκρίσεις του αποκλειστικά στο ανακτημένο περιεχόμενο, ώστε να περιορίζεται η πιθανότητα εμφάνισης ψευδαίσθητων αποκρίσεων [9], [10].

Από άποψη ροής δεδομένων, η online φάση ξεκινά όταν ο χρήστης υποβάλει ένα ερώτημα μέσω του γραφικού περιβάλλοντος. Το ερώτημα αποστέλλεται στο backend API, το οποίο το δίνει στο στάδιο κατανόησης ερωτήματος (query understanding). Εκεί το ερώτημα αναλύεται, ελέγχεται για small talk και, αν χρειαστεί, αναδιατυπώνεται ώστε να αποτελεί αυτόνομο (standalone) ερώτημα. Στη συνέχεια, το αναδιατυπωμένο ερώτημα μετατρέπεται σε διανυσματική αναπαράσταση (embedding) και χρησιμοποιείται για την ανάκτηση ενός πρώτου συνόλου υποψήφιων αποσπασμάτων από τη βάση. Τα αποσπάσματα αυτά περνούν από επανακατάταξη (reranking) μέσω ενός εξειδικευμένου cross-encoder μοντέλου, και τα πιο σχετικά επιλέγονται ως πλαίσιο για το τελικό στάδιο παραγωγής απάντησης. Το γλωσσικό μοντέλο καλείται τελικά με το επιλεγμένο context, ένα κατάλληλα σχεδιασμένο system prompt και το ιστορικό της συνομιλίας, και παράγει την απάντηση η οποία επιστρέφεται στον χρήστη.

3.2 Στάδιο Indexing

Το στάδιο indexing αποτελεί τη θεμελιώδη offline διαδικασία η οποία μετατρέπει το ακατέργαστο εκπαιδευτικό υλικό σε μια μορφή αξιοποιήσιμη από τον RAG μηχανισμό. Ο σκοπός του σταδίου είναι η δημιουργία μιας διανυσματικής βάσης γνώσης, στην οποία το κείμενο των εγγράφων είναι σπασμένο σε μικρότερα τμήματα (chunks) και κάθε τμήμα αντιπροσωπεύεται από μια αριθμητική διανυσματική αναπαράσταση που κωδικοποιεί το σημασιολογικό του περιεχόμενο. Η ποιότητα του indexing επηρεάζει άμεσα την ποιότητα της ανάκτησης και, κατ' επέκταση, την ποιότητα των τελικών απαντήσεων του συστήματος. Για τον λόγο αυτό, η σχεδίαση του σταδίου αυτού δόθηκε ιδιαίτερη προσοχή [10].

Σε τρία διαδοχικά βήματα οργανώνεται η επεξεργασία του εκπαιδευτικού υλικού, όπως αποτυπώνεται σχηματικά στο Σχήμα 3.2. Το πρώτο βήμα αφορά την εξαγωγή κειμένου από τα αρχεία PDF του μαθήματος. Στην εξαγωγή αυτή δεν είναι απλή ανάγνωση: τα αρχεία PDF διατηρούν μόνο την οπτική διάταξη του κειμένου, όχι τη λογική του δομή, και συχνά περιέχουν τεχνικά στοιχεία όπως αλλαγές γραμμής μέσα σε λέξεις, επαναλαμβανόμενα headers και footers, αριθμούς σελίδων ενσωματωμένους στο σώμα του κειμένου, και διπλά κενά που προκύπτουν από το layout. Όλα αυτά τα στοιχεία πρέπει να αφαιρεθούν κατά τη φάση του καθαρισμού, καθώς διαφορετικά θα υποβάθμιζαν την ποιότητα των παραγόμενων embeddings εισάγοντας θόρυβο.



Σχήμα 3.2: Ροή indexing

Το δεύτερο βήμα αφορά τη διαίρεση του καθαρισμένου κειμένου σε μικρότερα τμήματα, διαδικασία γνωστή ως chunking. Η αναγκαιότητα του chunking πηγάζει από δύο πρακτικούς λόγους. Ο πρώτος είναι ότι τα γλωσσικά μοντέλα και τα μοντέλα embedding έχουν ένα μέγιστο μέγεθος εισόδου (context window), πέρα από το οποίο το κείμενο κόβεται ή αγνοείται. Ο δεύτερος και σημαντικότερος λόγος είναι ότι η σημασιολογική αναζήτηση λειτουργεί καλύτερα σε μικρότερα, εστιασμένα τμήματα κειμένου παρά σε ολόκληρα έγγραφα: ένα απόσπασμα 200-300 λέξεων περιέχει συνήθως ένα συνεκτικό νοηματικό σύνολο (π.χ. μια έννοια, ένα παράδειγμα, μια απόδειξη), ενώ ένα ολόκληρο

έγγραφο μπορεί να καλύπτει δεκάδες διαφορετικά θέματα, με αποτέλεσμα το embedding του να είναι υπερβολικά «διάχυτο» και άρα λιγότερο χρήσιμο για σημασιολογική αντιστοίχιση [10].

Στην παρούσα σχεδίαση υιοθετήθηκε token-based chunking, δηλαδή ο έλεγχος μεγέθους των τμημάτων βασίζεται στον αριθμό των tokens του γλωσσικού μοντέλου αντί για τον αριθμό χαρακτήρων ή λέξεων. Αυτή η προσέγγιση εξασφαλίζει ότι κάθε chunk χωράει εντός του context window του LLM ανεξάρτητα από τη γλώσσα ή τη δομή του κειμένου. Αυτό είναι ένα κρίσιμο ζήτημα για γλώσσες όπως τα ελληνικά, όπου η αντιστοιχία λέξης-token διαφέρει σημαντικά από τις αγγλικές ρυθμίσεις που χρησιμοποιούνται συνήθως ως προεπιλογή. Πέρα από τη σταθεροποίηση του μεγέθους, εφαρμόζεται και επικάλυψη (overlap) μεταξύ διαδοχικών chunks, κατά την οποία οι τελευταίες δεκάδες tokens κάθε chunk επαναλαμβάνονται στην αρχή του επόμενου. Αυτή η τεχνική διασφαλίζει ότι προτάσεις οι οποίες τυχαία κόβονται στα όρια των chunks δεν χάνουν το νόημά τους, μειώνοντας την πιθανότητα απώλειας συνάφειας στη σημασιολογική αναζήτηση.

Το τρίτο και τελευταίο βήμα αφορά την παραγωγή διανυσματικών αναπαραστάσεων (embeddings) για κάθε chunk. Τα embeddings είναι πυκνά (dense) διανύσματα χαμηλής διάστασης τα οποία κωδικοποιούν το σημασιολογικό περιεχόμενο του κειμένου με τρόπο τέτοιον, ώστε σημασιολογικά παρόμοια κείμενα να βρίσκονται κοντά στον διανυσματικό χώρο [11], [12]. Με την αποδοτική σημασιολογική αναζήτηση βασίζεται σε αυτή την ιδιότητα: δοθέντος ενός ερωτήματος, αρκεί να βρεθούν τα chunks των οποίων τα διανύσματα έχουν τη μεγαλύτερη ομοιότητα με το διάνυσμα του ερωτήματος. Για τη συγκεκριμένη εργασία, η επιλογή του μοντέλου embedding αποτέλεσε κρίσιμη σχεδιαστική απόφαση, καθώς το εκπαιδευτικό υλικό είναι στα ελληνικά. Επιλέχθηκε ένα πολυγλωσσικό μοντέλο το οποίο έχει εκπαιδευτεί σε κείμενα δεκάδων γλωσσών και υποστηρίζει ρητά την ελληνική [12], οι λεπτομέρειες της επιλογής αυτής παρουσιάζονται στο Κεφάλαιο 4.

Κάθε chunk, μετά την παραγωγή του embedding, αποθηκεύεται σε μια διανυσματική βάση δεδομένων μαζί με τα σχετικά μεταδεδομένα του: το όνομα του αρχείου προέλευσης, τον αύξοντα αριθμό του chunk εντός του αρχείου, και τον αριθμό των tokens που περιέχει. Τα μεταδεδομένα αυτά εξυπηρετούν δύο σκοπούς: πρώτον, επιτρέπουν την αναφορά της πηγής στην τελική απάντηση προς τον χρήστη, ενισχύοντας τη διαφάνεια και την εμπιστοσύνη στο σύστημα· δεύτερον, επιτρέπουν την παρακολούθηση του token budget κατά τη φάση του reranking, όπως θα αναλυθεί παρακάτω. Η διανυσματική βάση οργανώνει τα αποθηκευμένα διανύσματα με χρήση δομών ευρετηρίασης όπως ο αλγόριθμος HNSW (Hierarchical Navigable Small World), ο οποίος επιτρέπει προσεγγιστική αναζήτηση πλησιέστερων γειτόνων σε λογαριθμικό χρόνο [13], καθιστώντας αποδοτική την αναζήτηση ομοιότητας ακόμη και σε συλλογές εκατομμυρίων διανυσμάτων.

3.3 Στάδιο Query Understanding

Η άμεση χρήση του ερωτήματος του χρήστη ως εισόδου στο στάδιο της ανάκτησης, χωρίς ενδιάμεση επεξεργασία, αποτελεί μία από τις συνηθέστερες πηγές υποβαθμισμένης απόδοσης σε RAG συστήματα [10]. Οι φοιτητές τυπικά υποβάλλουν ερωτήματα σε συνομιλιακό ύφος, που συχνά είναι σύντομα, περιέχουν αναφορικές εκφράσεις (π.χ. «αυτό», «το προηγούμενο», «δώσε μου ένα παράδειγμα»), ή είναι καθαρά κοινωνικού χαρακτήρα (π.χ. «γεια», «ευχαριστώ», «ποιος είσαι»). Κάθε μία από αυτές τις περιπτώσεις απαιτεί διαφορετική αντιμετώπιση, και για τον λόγο αυτό στο σύστημα προστίθεται ένα στάδιο κατανόησης ερωτήματος (query understanding), το οποίο προηγείται της ανάκτησης.

Ο πρώτος έλεγχος αφορά την ανίχνευση small talk, δηλαδή ερωτημάτων που δεν απαιτούν αναζήτηση στη βάση γνώσης. Η ανίχνευση υλοποιείται μέσω ενός συνδυασμού κανόνων και λεξιλογικών προτύπων που αναγνωρίζουν τρεις κατηγορίες: χαιρετισμούς, γενικές ερωτήσεις σχετικά με την ταυτότητα ή τις δυνατότητες του συστήματος, και meta-ερωτήματα για την ίδια τη συνομιλία (π.χ. αιτήματα σύνοψης). Όταν ανιχνευθεί τέτοιο ερώτημα, το σύστημα παρακάμπτει το RAG pipeline και απαντά απευθείας μέσω του γλωσσικού μοντέλου με ένα διαφορετικό, γενικότερου σκοπού system prompt. Αυτή η προσέγγιση εξοικονομεί υπολογιστικούς πόρους, καθώς η αναζήτηση στη διανυσματική βάση και το στάδιο επανακατάταξης είναι σαφώς πιο ακριβά από μια απλή κλήση του LLM. Παράλληλα, αποφεύγει την ανάκτηση άσχετων αποσπασμάτων για ερωτήματα όπως ένα απλό «γεια», τα οποία δεν σχετίζονται με το αντικείμενο του μαθήματος.

Ο δεύτερος έλεγχος αφορά την αναδιατύπωση (rewriting) εξαρτώμενων από context ερωτημάτων. Σε ένα μεγάλο σε έκταση διάλογο, ο χρήστης συχνά υποβάλλει ερωτήματα τα οποία δεν είναι αυτόνομα αλλά εξαρτώνται από το προηγούμενο πλαίσιο της συνομιλίας. Για παράδειγμα, αν ένας φοιτητής πρώτα ρωτήσει «τι είναι το interface στην Java;» και στη συνέχεια υποβάλει το ερώτημα «δώσε μου ένα παράδειγμα», το δεύτερο ερώτημα από μόνο του είναι ασαφές, δεν περιέχει δηλαδή καμία λεκτική ένδειξη του θέματος στο οποίο αναφέρεται. Αν το ερώτημα αυτό χρησιμοποιηθεί απευθείας για σημασιολογική αναζήτηση στη βάση, τα αποτελέσματα θα είναι τυχαία, καθώς το ίδιο το ερώτημα δεν φέρει επαρκή σημασιολογική πληροφορία για το θέμα.

Η λύση που υιοθετείται είναι η κλήση του γλωσσικού μοντέλου ως ενδιάμεσου επεξεργαστή. Το ίδιο το μοντέλο λαμβάνει ως είσοδο το τρέχον ερώτημα και το ιστορικό της συνομιλίας, και καλείται να παράγει μια αυτόνομη (standalone) εκδοχή του ερωτήματος που μπορεί να χρησιμοποιηθεί ανεξάρτητα από το ιστορικό. Στο παραπάνω παράδειγμα, το ερώτημα «δώσε μου ένα παράδειγμα» μετασχηματίζεται στο «δώσε μου ένα παράδειγμα interface στην Java». Το αναδιατυπωμένο ερώτημα χρησιμοποιείται στη συνέχεια ως είσοδος στο στάδιο ανάκτησης. Αυτή η τεχνική, γνωστή και ως query rewriting, αποτελεί μία από τις πλέον καθιερωμένες βελτιώσεις του «naive RAG» και έχει τεκμηριωθεί εκτενώς στη βιβλιογραφία ως αποτελεσματική μέθοδος για την αντιμετώπιση μεγάλων σεναρίων διαλόγου [10]. Πρέπει να επισημανθεί ότι η αναδιατύπωση επηρεάζει μόνο την ανάκτηση ενώ το αρχικό ερώτημα του χρήστη παραμένει αυτούσιο και χρησιμοποιείται στο τελικό στάδιο παραγωγής απάντησης, ώστε το σύστημα να απαντά στη συγκεκριμένη διατύπωση του χρήστη και όχι στην αναδιατυπωμένη εκδοχή.

Η διαδικασία αναδιατύπωσης ερωτήματος παρουσιάζει ιδιαίτερο ενδιαφέρον αν αναλυθεί σε συγκεκριμένα παραδείγματα. Ένα τυπικό σενάριο χρήσης είναι η συνέχιση μιας συνομιλίας όπου ο φοιτητής έχει μόλις ρωτήσει «τι είναι interface» και λαμβάνει μια εξήγηση. Αν η επόμενη ερώτηση του φοιτητή είναι «δώσε μου ένα παράδειγμα», το ερώτημα από μόνο του είναι αδιαφανές καθώς χωρίς το πλαίσιο της προηγούμενης συνομιλίας, δεν είναι δυνατή η ανάκτηση σχετικών chunks από τη διανυσματική βάση, καθώς η λέξη «παράδειγμα» είναι σημασιολογικά πολύ γενική. Το στάδιο της αναδιατύπωσης μετατρέπει το ερώτημα αυτό σε μια πλήρη, αυτόνομη διατύπωση του τύπου «δώσε μου ένα παράδειγμα υλοποίησης interface στην Java», η οποία περιέχει πλέον επαρκή σημασιολογική πληροφορία για αποτελεσματική ανάκτηση.

Η επιλογή του ίδιου γλωσσικού μοντέλου Llama-Krikri για τον ρόλο του αναδιατυπωτή αντί για μια εναλλακτική προσέγγιση (όπως rule-based μετασχηματισμοί ή εξειδικευμένος classifier) δικαιολογείται από τρεις λόγους. Πρώτον, το γλωσσικό μοντέλο διαθέτει ήδη την ικανότητα κατανόησης του ελληνικού συνομιλιακού πλαισίου, την οποία ένα rule-based σύστημα θα έπρεπε να αντιγράψει με μεγάλη δυσκολία. Δεύτερον, η ίδια υποδομή του μοντέλου που χρησιμοποιείται για τη

γένεση χρησιμοποιείται και για την αναδιατύπωση, χωρίς την ανάγκη φόρτωσης επιπλέον μοντέλων στη μνήμη. Τρίτον, η προσέγγιση αυτή είναι συμβατή με τη σύγχρονη βιβλιογραφία, η οποία προτείνει τη χρήση LLMs ως rewriters σε RAG συστήματα και έχει αποδείξει την αποτελεσματικότητα της προσέγγισης σε πολυγύρους διαλόγους [27].

Ο σχεδιασμός αυτός περιλαμβάνει ωστόσο ένα ουσιαστικό trade-off μεταξύ ποιότητας και ταχύτητας. Κάθε αναδιατύπωση απαιτεί μια επιπλέον κλήση στο γλωσσικό μοντέλο, η οποία προσθέτει ορατή καθυστέρηση στον συνολικό χρόνο απόκρισης. Για να μετριαστεί αυτό το κόστος, η αναδιατύπωση ενεργοποιείται επιλεκτικά. Το σύστημα κάνει πρώτα ένα σύντομο έλεγχο για το αν το ερώτημα είναι ήδη αυτόνομο (δηλαδή δεν περιέχει αναφορικές εκφράσεις όπως «αυτό», «αυτή», «εκείνο», «όπως είπες») και αν είναι αρκετά μακρύ για να φέρει από μόνο του επαρκή σημασιολογική πληροφορία. Μόνο αν το ερώτημα θεωρηθεί εξαρτώμενο από context, πραγματοποιείται η κλήση του rewriter. Σε οριακές περιπτώσεις, όπως ερωτήματα με δύο ή τρεις λέξεις (π.χ. «και γιατί;»), η αναδιατύπωση είναι πάντα απαραίτητη, ενώ σε ερωτήματα πλήρους διατύπωσης (π.χ. «τι είναι η κληρονομικότητα στην Java») παρακάμπτεται εντελώς.

3.4 Στάδιο Ανάκτησης (Retrieval)

Το στάδιο της ανάκτησης αποτελεί την καρδιά ενός RAG συστήματος: σκοπός του είναι, δοθέντος ενός ερωτήματος, να εντοπίσει στη διανυσματική βάση γνώσης τα πλέον σχετικά αποσπάσματα εκπαιδευτικού υλικού τα οποία θα χρησιμοποιηθούν στη συνέχεια ως πλαίσιο για την παραγωγή απάντησης. Η ποιότητα της ανάκτησης επηρεάζει άμεσα και καθοριστικά την ποιότητα της τελικής απάντησης: αν τα ανακτηθέντα αποσπάσματα δεν περιέχουν τη σχετική πληροφορία, ακόμη και το ισχυρότερο γλωσσικό μοντέλο δεν θα μπορέσει να δώσει σωστή απάντηση [9], [10].

Οι κλασικές προσεγγίσεις ανάκτησης πληροφορίας, όπως ο αλγόριθμος BM25, βασίζονται σε λεξικολογική αντιστοίχιση υπολογίζοντας τη σχετικότητα ανάμεσα σε ένα ερώτημα και ένα έγγραφο μετρώντας τη συχνότητα εμφάνισης των όρων του ερωτήματος στο έγγραφο, σταθμισμένη με την αντίστροφη συχνότητα εμφάνισής τους σε ολόκληρη τη συλλογή. Αυτή η προσέγγιση είναι ταχύτατη και συχνά εκπληκτικά αποδοτική, αλλά έχει ένα θεμελιώδες πρόβλημα, δεν αναγνωρίζει σημασιολογικές συσχετίσεις πέραν της ακριβούς λεξικολογικής ταύτισης. Αν ο χρήστης ρωτήσει για «κληρονομικότητα» και το έγγραφο χρησιμοποιεί τον όρο «παραγωγή κλάσης», η BM25 δεν θα τα συσχετίσει, παρότι σημασιολογικά περιγράφουν την ίδια έννοια. Το πρόβλημα αυτό είναι ιδιαίτερα έντονο στο εκπαιδευτικό πλαίσιο, όπου οι φοιτητές συχνά διατυπώνουν τα ερωτήματά τους με τρόπο διαφορετικό από αυτόν που χρησιμοποιεί το επίσημο εκπαιδευτικό υλικό.

Καθοριστικό βήμα για την αντιμετώπιση του προβλήματος αυτού υπήρξε η εμφάνιση της πυκνής ανάκτησης (dense retrieval), όπως διατυπώθηκε στην εργασία των Karpukhin et al. στο Dense Passage Retrieval [11]. Στην προσέγγιση αυτή, τόσο τα ερωτήματα όσο και τα έγγραφα κωδικοποιούνται σε πυκνά διανύσματα μέσω νευρωνικών κωδικοποιητών, οι οποίοι εκπαιδεύονται ώστε σημασιολογικά συγγενή κείμενα να βρίσκονται κοντά στον διανυσματικό χώρο. Στην ομοιότητα μεταξύ ερωτήματος και εγγράφου υπολογίζεται πλέον ως η ομοιότητα συνημιτόνου (cosine similarity) των αντίστοιχων διανυσμάτων τους, η οποία είναι ευαίσθητη στο νόημα και όχι απλώς στις ακριβείς λέξεις. Με την dense retrieval έχει πλέον καθιερωθεί ως η βάση των σύγχρονων RAG συστημάτων και έχει δείξει σταθερά καλύτερη απόδοση από τις καθαρά λεξικολογικές προσεγγίσεις σε ευρύ φάσμα εφαρμογών [11], [12].

Στο σύστημα αυτό, η dense retrieval υλοποιείται με bi-encoder αρχιτεκτονική που σημαίνει ότι το ερώτημα και τα chunks κωδικοποιούνται ανεξάρτητα το ένα από το άλλο μέσω του ίδιου προεκπαιδευμένου μοντέλου embedding. Στην offline φάση indexing, όλα τα chunks κωδικοποιούνται μία φορά και τα διανύσματά τους αποθηκεύονται στη διανυσματική βάση. Στην online φάση, το ερώτημα κωδικοποιείται σε πραγματικό χρόνο, και η αναζήτηση ομοιότητας γίνεται μεταξύ του διανύσματος του ερωτήματος και των προϋπολογισμένων διανυσμάτων των chunks. Η ασύμμετρη αυτή προσέγγιση είναι υπολογιστικά αποδοτική, καθώς το ακριβό στάδιο της κωδικοποίησης των εγγράφων εκτελείται μόνο μία φορά, ενώ η αναζήτηση σε πραγματικό χρόνο περιορίζεται στον υπολογισμό ενός και μόνο embedding ερωτήματος συν την αναζήτηση πλησιέστερων γειτόνων στον προϋπολογισμένο δείκτη.

Ένα σημαντικό σχεδιαστικό ζήτημα αφορά την επιλογή του μοντέλου embedding σε σχέση με τη γλώσσα του υλικού. Η πλειονότητα των state-of-the-art μοντέλων embedding έχει εκπαιδευτεί κυρίως σε αγγλικά κείμενα και η εφαρμογή τους σε ελληνικά δεδομένα οδηγεί σε σημαντική υποβάθμιση της ποιότητας της ανάκτησης. Η λύση στο πρόβλημα αυτό παρέχεται από πολυγλωσσικά μοντέλα όπως το multilingual E5 [12], τα οποία εκπαιδεύονται εξαρχής σε μεγάλα σύνολα πολυγλωσσικών κειμένων και επιτυγχάνουν υψηλή ποιότητα σε δεκάδες γλώσσες, συμπεριλαμβανομένων των ελληνικών.

Από πρακτικής πλευράς, κατά το στάδιο ανάκτησης το σύστημα επιστρέφει όχι μόνο το ένα «καλύτερο» chunk αλλά ένα ευρύτερο σύνολο υποψηφίων. Συγκεκριμένα τα 20 chunks με την υψηλότερη ομοιότητα συνημιτόνου ως προς το ερώτημα. Η επιλογή της τιμής αυτής δεν είναι τυχαία. Ένα πολύ μικρό σύνολο υποψηφίων (π.χ. 3-5) αυξάνει τον κίνδυνο να παραβλεφθεί ένα πραγματικά σχετικό απόσπασμα, ενώ ένα πολύ μεγάλο σύνολο επιβαρύνει το επόμενο στάδιο του reranking χωρίς ανάλογο όφελος. Η τιμή των 20 υποψηφίων αποτελεί μια κοινώς χρησιμοποιούμενη τιμή στη βιβλιογραφία και εξασφαλίζει επαρκή κάλυψη σχετικών αποσπασμάτων ακόμη και σε περιπτώσεις που η σημασιολογική αντιστοίχιση δεν είναι τέλεια. Το σύνολο αυτών των 20 υποψηφίων τροφοδοτείται στη συνέχεια στο στάδιο της επανακατάταξης.

3.5 Reranking

Παρά την αποτελεσματικότητα της πυκνής ανάκτησης, τα αποτελέσματα που επιστρέφει ένας bi-encoder δεν είναι πάντοτε τα βέλτιστα δυνατά. Ο λόγος βρίσκεται στην ίδια την αρχιτεκτονική του bi-encoder. Το ερώτημα και το κάθε chunk κωδικοποιούνται εντελώς ανεξάρτητα, χωρίς το μοντέλο να «βλέπει» ταυτόχρονα και τα δύο κατά την παραγωγή του embedding. Η ανεξαρτησία αυτή είναι που κάνει τον bi-encoder αποδοτικό, επιτρέπει δηλαδή την προϋπολογισμένη αποθήκευση όλων των document embeddings αλλά ταυτόχρονα τον περιορίζει, καθώς δεν μπορεί να αποτυπώσει λεπτομερείς αλληλεπιδράσεις ανάμεσα σε συγκεκριμένα στοιχεία του ερωτήματος και συγκεκριμένα στοιχεία του εγγράφου.

Η λύση στο πρόβλημα αυτό παρέχεται από τους cross-encoders, μια διαφορετική κατηγορία μοντέλων που εισήχθη στη βιβλιογραφία από τους Nogueira και Cho [14] στο πλαίσιο της επανακατάταξης αποτελεσμάτων. Ένας cross-encoder λαμβάνει ως είσοδο ένα ζεύγος (ερώτημα, έγγραφο) ταυτόχρονα και παράγει μια βαθμολογία σχετικότητας που λαμβάνει υπόψη τη σχέση τους. Με αυτόν τον τρόπο το μοντέλο μπορεί να αναγνωρίσει λεπτές σημασιολογικές αντιστοιχίες τις οποίες ένας bi-encoder θα έχανε. Το μειονέκτημα είναι ότι ο cross-encoder πρέπει να κληθεί μία φορά για κάθε ζεύγος ερωτήματος-εγγράφου, κάτι που τον καθιστά υπολογιστικά απαγορευτικό για αναζήτηση σε ολόκληρη τη βάση. Για τον λόγο αυτό, οι cross-encoders χρησιμοποιούνται χαρακτηριστικά ως

δεύτερο στάδιο επανακατάταξης μετά από ένα αρχικό στάδιο bi-encoder retrieval, μια αρχιτεκτονική γνωστή ως «retrieve-then-rerank», η οποία αποτελεί πλέον τυπική βέλτιστη πρακτική σε RAG pipelines [10], [14].

Υιοθετείται λοιπόν ακριβώς αυτή η δύο-σταδίων προσέγγιση. Τα 20 υποψήφια chunks που επέστρεψε το bi-encoder στάδιο της ανάκτησης περνούν από τον cross-encoder, ο οποίος υπολογίζει μια νέα, πιο ακριβή βαθμολογία σχετικότητας για κάθε ένα από αυτά. Τα chunks κατατάσσονται εκ νέου με βάση τη νέα βαθμολογία, και τα τέσσερα chunks με την υψηλότερη βαθμολογία επιλέγονται για την τελική τροφοδοσία του γλωσσικού μοντέλου. Η επιλογή του αριθμού τέσσερα δεν είναι αυθαίρετη αλλά αντιστοιχεί σε έναν συμβιβασμό μεταξύ επαρκούς κάλυψης πληροφορίας και περιορισμένου token budget που μπορεί να χωρέσει στο context window του γλωσσικού μοντέλου μαζί με το system prompt, το ιστορικό συνομιλίας και το ίδιο το ερώτημα.

Ένα σημαντικό ζήτημα ειδικά στο ελληνόγλωσσο πλαίσιο αφορά την επιλογή του cross-encoder μοντέλου. Η πλειονότητα των διαθέσιμων cross-encoders είναι εκπαιδευμένη σε αγγλικά δεδομένα και δεν αποδίδει επαρκώς σε άλλες γλώσσες. Για την αντιμετώπιση αυτού του προβλήματος, χρησιμοποιήθηκε το BGE M3-Embedding [15], μια σύγχρονη οικογένεια μοντέλων που υποστηρίζει περισσότερες από 100 γλώσσες, συμπεριλαμβανομένων των ελληνικών, και παρέχει τόσο δυνατότητες bi-encoder retrieval όσο και cross-encoder reranking.

Συγκριτικά με τη σκέτη cosine similarity του σταδίου ανάκτησης, η προστιθέμενη αξία του cross-encoder είναι μετρήσιμη και προέρχεται από έναν συγκεκριμένο αρχιτεκτονικό μηχανισμό. Στον bi-encoder, το ερώτημα και το chunk περνούν χωριστά από το ίδιο μοντέλο και παράγουν δύο ανεξάρτητα διανύσματα. Η σχετικότητά τους εκφράζεται εκ των υστέρων ως το συνημίτονο της γωνίας τους στον 1024-διάστατο χώρο. Αυτή η συμπίεση όλης της πληροφορίας ενός κειμένου σε ένα και μόνο διάνυσμα είναι αναγκαία για την κλιμάκωση της αναζήτησης σε χιλιάδες chunks, αλλά αναπόφευκτα χάνει λεπτές διαφοροποιήσεις. Ο cross-encoder, αντίθετα, τροφοδοτείται με το ζεύγος (ερώτημα, chunk) ως ενιαία ακολουθία εισόδου και εφαρμόζει τους μηχανισμούς self-attention της αρχιτεκτονικής Transformer πάνω και στα δύο ταυτόχρονα. Το αποτέλεσμα είναι ότι κάθε λέξη του ερωτήματος μπορεί να «δει» κάθε λέξη του chunk και αντίστροφα, εντοπίζοντας λεπτές αλληλεπιδράσεις δηλαδή αν μια συγκεκριμένη λέξη-κλειδί του ερωτήματος εμφανίζεται στο σωστό συντακτικό ή σημασιολογικό περιβάλλον μέσα στο chunk, οι οποίες είναι αδύνατο να αναπαρασταθούν στη γεωμετρική απόσταση δύο ανεξάρτητων διανυσμάτων. Το κόστος είναι υπολογιστικό. Ο cross-encoder απαιτεί μία πλήρη forward pass ανά ζεύγος αντί για μία προϋπολογισμένη αναζήτηση γείτονα, γι' αυτό και εφαρμόζεται μόνο στα 20 υποψήφια του πρώτου σταδίου και όχι σε ολόκληρη τη βάση. Η αρχιτεκτονική two-stage που προκύπτει είναι bi-encoder υψηλής ανάκλησης για τη συρρίκνωση του χώρου αναζήτησης και cross-encoder υψηλής ακρίβειας για την τελική κατάταξη. Αυτή η αρχιτεκτονική αποτελεί την πλέον καθιερωμένη πρακτική στη σύγχρονη ανάκτησης πληροφορίας [14], [15].

Πέρα από τη βασική επανακατάταξη, το στάδιο αυτό περιλαμβάνει και δύο ακόμη μηχανισμούς οι οποίοι είναι κρίσιμοι για την πρακτική αξιοπιστία του συστήματος. Ο πρώτος είναι ένας έλεγχος token budget: πριν τα επιλεγμένα τέσσερα chunks περάσουν στο στάδιο παραγωγής απάντησης, υπολογίζεται ο συνολικός αριθμός των tokens που θα καταλάβουν στο context window του γλωσσικού μοντέλου, και τυχόν chunks που υπερβαίνουν το διαθέσιμο όριο παραλείπονται. Αυτό διασφαλίζει ότι το context που τροφοδοτείται στο LLM δεν υπερβαίνει ποτέ το μέγιστο επιτρεπτό μέγεθος, αλλιώς θα προκαλούσε είτε αποκοπή πληροφοριών είτε σφάλματα εκτέλεσης.

Ο δεύτερος μηχανισμός είναι ένα κατώφλι σχετικότητας (relevance threshold). Εάν ακόμη και το κορυφαίο chunk στην κατάταξη επιστρέφει βαθμολογία σχετικότητας κάτω από ένα προκαθορισμένο όριο, το σύστημα θεωρεί ότι στη βάση δεν υπάρχει σχετικό υλικό. Σε αυτή την περίπτωση επιστρέφει στον χρήστη ένα τυποποιημένο μήνυμα που αναφέρει. Αυτή η συμπεριφορά είναι πολύ σημαντική για την αποφυγή ψευδαίσθητων αποκρίσεων: αντί το γλωσσικό μοντέλο να αναγκαστεί να παράγει μια απάντηση βασισμένη σε άσχετο ή ανύπαρκτο context, προτιμάται η διαφανής αναγνώριση των ορίων του συστήματος.

3.6 Generation

Το τελικό στάδιο του RAG pipeline αφορά την παραγωγή της απάντησης μέσω του γλωσσικού μοντέλου. Στο σημείο αυτό, τα δύο προηγούμενα στάδια έχουν ήδη εντοπίσει ένα μικρό, εστιασμένο σύνολο σχετικών αποσπασμάτων από το εκπαιδευτικό υλικό ενώ η πρόκληση τώρα είναι η σύνθεση μιας εκπαιδευτικά κατάλληλης απάντησης η οποία βασίζεται αυστηρά σε αυτό το περιεχόμενο και όχι στην παραμετρική γνώση του μοντέλου. Εδώ, η πρόκληση αυτή είναι κρίσιμη, καθώς ακόμη και μοντέλα που έχουν πρόσβαση σε σχετικό context μπορεί να αγνοήσουν το παρεχόμενο υλικό και να βασιστούν σε εσωτερικές, ενδεχομένως εσφαλμένες αναπαραστάσεις [6].

Το πρώτο βήμα του σταδίου αυτού της δημιουργίας απάντησης είναι η κατασκευή του context. Τα τέσσερα chunks που επιλέχθηκαν από το reranking συναρμολογούνται σε μία ενιαία ακολουθία κειμένου. Κάθε chunk συνοδεύεται από μια ετικέτα που αναφέρει ρητά την πηγή του (όνομα αρχείου και αριθμός chunk), ώστε να είναι ανιχνεύσιμη η προέλευση κάθε τμήματος πληροφορίας. Οι ετικέτες αυτές εξυπηρετούν δύο σκοπούς: αφενός καθοδηγούν το γλωσσικό μοντέλο να αντιμετωπίζει κάθε chunk ως διακριτή πηγή, και αφετέρου εμφανίζονται στο γραφικό περιβάλλον ως εμφανείς παραπομπές, επιτρέποντας στον φοιτητή να ανατρέξει στο πρωτότυπο εκπαιδευτικό υλικό για περαιτέρω μελέτη.

Το δεύτερο βήμα είναι η κατασκευή του πλήρους prompt που θα αποσταλεί στο γλωσσικό μοντέλο. Το prompt δομείται σε τέσσερα μέρη: (α) το system prompt, το οποίο ορίζει τον ρόλο και τη συμπεριφορά του μοντέλου ως εκπαιδευτικού βοηθού, (β) το ιστορικό της συνομιλίας, το οποίο επιτρέπει συνεκτικούς πολυγύρους διαλόγους, (γ) το ανακτημένο context με τα επιλεγμένα chunks, και (δ) το αρχικό ερώτημα του χρήστη στην αρχική του μορφή. Η σειρά και η δομή αυτών των στοιχείων έχει ιδιαίτερη σημασία, καθώς επηρεάζει άμεσα τη συμπεριφορά του μοντέλου [10].

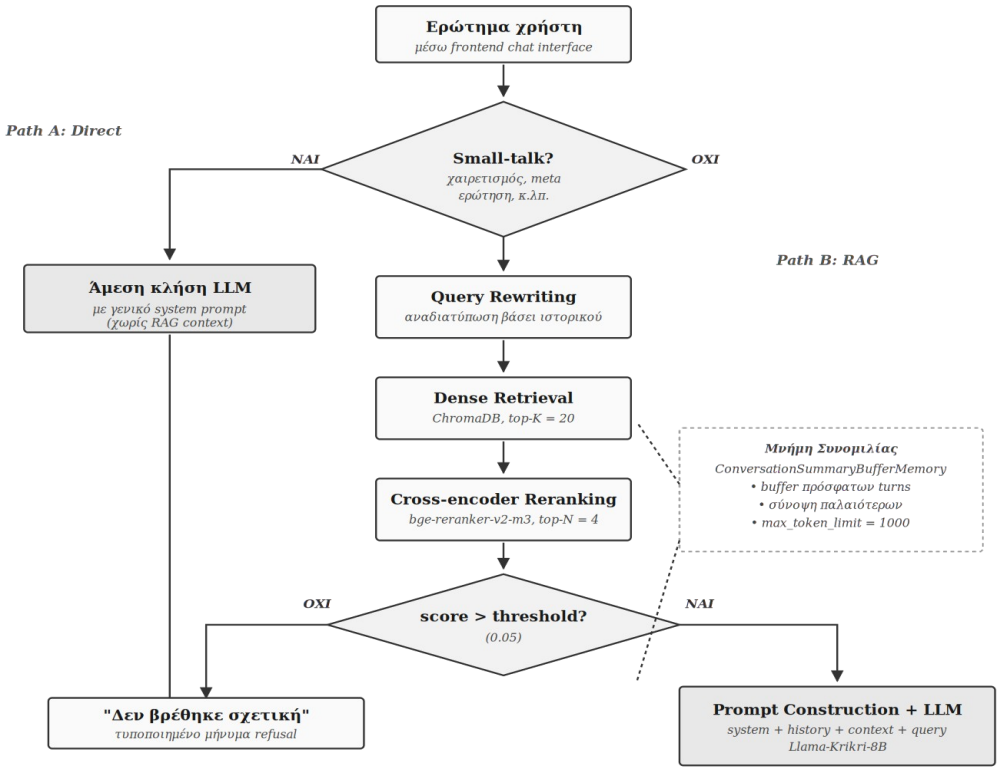
Το system prompt αποτελεί ίσως το πιο κρίσιμο σημείο της σχεδίασης. Σε αντίθεση με ένα «γυμνό» γλωσσικό μοντέλο, το οποίο απλώς προσπαθεί να συμπληρώσει το κείμενο που του δίνεται, το system prompt κατευθύνει ρητά το μοντέλο να υιοθετήσει έναν συγκεκριμένο ρόλο, με συγκεκριμένες οδηγίες συμπεριφοράς. Για το παρόν σύστημα σχεδιάστηκε ένα system prompt που καθορίζει το μοντέλο ως καθηγητή Αντικειμενοστραφούς Προγραμματισμού και περιλαμβάνει οδηγίες για το στυλ των απαντήσεων (συνοπτικές, εκπαιδευτικά δομημένες, χωρίς περιττούς χαρακτηρισμούς), για την αυστηρή χρήση αποκλειστικά των παρεχόμενων αποσπασμάτων (χωρίς εφεύρεση πληροφοριών), και για την εκπαιδευτική προσέγγιση (χρήση παραδειγμάτων, ερωτήσεις όπου χρειάζεται, εποικοδομητική διόρθωση λαθών). Η συστηματική αυτή μηχανική εντολών (prompt engineering) διασφαλίζει υψηλό βαθμό συμμόρφωσης με τις βέλτιστες πρακτικές των σύγχρονων εκπαιδευτικών συστημάτων που βασίζονται σε LLMs [3], [10].

Το τρίτο σημαντικό ζήτημα του σταδίου αυτού αφορά τη διαχείριση μνήμης σε συνομιλίες με πολλούς γύρους μηνυμάτων. Σε ένα ρεαλιστικό εκπαιδευτικό σενάριο, ο φοιτητής δεν θέτει ένα

μεμονωμένο ερώτημα και σταματά αλλά συχνά συνεχίζει με επεξηγηματικές ερωτήσεις, αιτήματα παραδειγμάτων, ή αλλαγές θέματος. Η διατήρηση του ιστορικού συνομιλίας είναι λοιπόν απαραίτητη για τη συνεκτική λειτουργία του συστήματος. Ωστόσο, το ιστορικό αυτό δεν μπορεί να αυξάνεται απεριόριστα, καθώς κάθε γλωσσικό μοντέλο έχει ένα πεπερασμένο context window. Αν το ιστορικό ξεπεράσει το όριο, τα παλαιότερα turns θα πρέπει είτε να αποκοπούν (με κίνδυνο απώλειας κρίσιμων πληροφοριών) είτε να συμπιεστούν.

Η προσέγγιση που υιοθετείται στο συγκεκριμένο σύστημα είναι μια δομημένη μνήμη με δύο επίπεδα: ένα buffer πρόσφατων turns το οποίο αποθηκεύεται ακέραιο, και μια αυτόματα παραγόμενη σύνοψη (summary) των παλαιότερων turns. Όταν το μέγεθος του buffer ξεπεράσει ένα προκαθορισμένο όριο tokens, τα παλαιότερα turns του buffer μεταφέρονται στη σύνοψη, η οποία ανασυντίθεται καλώντας το ίδιο το γλωσσικό μοντέλο ως συνοψιστή. Με αυτόν τον τρόπο το σύστημα διατηρεί τη συνεκτική εικόνα της συνομιλίας ακόμη και σε εκτενείς διαλόγους, χωρίς να εξαντλεί το context window του μοντέλου. Αυτή η προσέγγιση αντιστοιχεί σε αυτό που αναφέρεται ως summary buffer memory και αποτελεί μια πρακτικά αποτελεσματική λύση στο πρόβλημα της μακροχρόνιας μνήμης σε συνομιλιακά RAG συστήματα [4], [10].

Το σύνολο του σταδίου, σε συνδυασμό με τα προηγούμενα στάδια, συνθέτει μια πλήρη ροή από το αρχικό ερώτημα του χρήστη μέχρι την τελική απάντηση. Η ροή αυτή απεικονίζεται ολοκληρωμένα στο Σχήμα 3.3, όπου φαίνονται και οι δύο κύριες διαδρομές: η RAG διαδρομή για τεχνικά ερωτήματα, και η παρακαμπτήρια διαδρομή για small talk. Η υλοποίηση κάθε ενός από τα στάδια, με έμφαση στα συγκεκριμένα εργαλεία και τις συγκεκριμένες επιλογές παραμέτρων, παρουσιάζεται στο επόμενο κεφάλαιο.



Σχήμα 3.3: Ροή inference με δύο διαδρομές

Κεφάλαιο 4ο: Υλοποίηση

4.1 Τεχνολογίες και Εργαλεία

Η υλοποίηση του συστήματος βασίστηκε εξ ολοκλήρου σε εργαλεία ανοικτού κώδικα και γλώσσες προγραμματισμού ευρέως καθιερωμένες στην κοινότητα της Τεχνητής Νοημοσύνης και της μηχανικής λογισμικού. Κεντρική επιλογή αποτέλεσε η γλώσσα Python (έκδοση 3.11), η οποία προσφέρει ένα ώριμο οικοσύστημα βιβλιοθηκών για επεξεργασία φυσικής γλώσσας, μηχανική μάθηση και διαδικτυακές εφαρμογές. Η αποκλειστική χρήση Python σε όλο το backend επέτρεψε την αρθρωτή ανάπτυξη του συστήματος χωρίς διαγλωσσικές διεπαφές και διευκόλυνε την ενσωμάτωση των διαφόρων υποσυστημάτων σε ένα συνεκτικό σύνολο.

Για την υλοποίηση του backend API επιλέχθηκε το FastAPI, ένα σύγχρονο διαδικτυακό framework της Python το οποίο βασίζεται στα πρότυπα ASGI και υποστηρίζει φυσικά ασύγχρονη εκτέλεση (async/await). Η επιλογή του FastAPI έναντι εναλλακτικών όπως το Flask στηρίζεται σε τρεις λόγους. Πρώτον, παράγει αυτόματα διαδραστική τεκμηρίωση API μέσω Swagger UI (ενός web interface που εκθέτει όλα τα endpoints του συστήματος και επιτρέπει τον άμεσο έλεγχό τους μέσω browser, διευκολύνοντας σημαντικά τη φάση ανάπτυξης και αποσφαλμάτωσης). Δεύτερον, υποστηρίζει εγγενώς ασύγχρονα endpoints, στοιχείο κρίσιμο όταν τα αιτήματα περιλαμβάνουν ενέργειες εισόδου/εξόδου (I/O-bound), δηλαδή λειτουργίες όπου ο server αναμένει αποτέλεσμα από εξωτερικό πόρο, όπως κλήσεις στη διανυσματική βάση ή στο γλωσσικό μοντέλο. Τρίτον, προσφέρει ισχυρή τύπωση μέσω Pydantic models (κλάσεων Python που ορίζουν το σχήμα των δεδομένων εισόδου/εξόδου κάθε endpoint) επιτρέποντας την αυτόματη επικύρωσή τους σε χρόνο εκτέλεσης με ελάχιστη επιπλέον υλοποίηση.

Το API εκθέτει τρία endpoints στα οποία ανταποκρίνεται το frontend. Το πρώτο, POST /chat, αποτελεί τον κύριο μηχανισμό υποβολής ερωτημάτων: δέχεται ως είσοδο το μήνυμα του χρήστη και ένα αναγνωριστικό συνεδρίας (session_id), εκτελεί ολόκληρο το RAG pipeline και επιστρέφει την απάντηση μαζί με τις πηγές της. Το δεύτερο, POST /reset, χρησιμοποιείται για τον τερματισμό μιας συνεδρίας και τη διαγραφή του σχετικού ιστορικού συνομιλίας. Το τρίτο, GET /health, επιτρέπει τον έλεγχο της κατάστασης του συστήματος και επιβεβαιώνει ότι τα βαριά μοντέλα έχουν φορτωθεί ορθά στη μνήμη κατά την εκκίνηση του server. Η αρχικοποίηση των μοντέλων γίνεται μέσω του lifespan context manager του FastAPI, ώστε να φορτωθούν μία φορά κατά την εκκίνηση και να παραμείνουν στη μνήμη καθ' όλη τη διάρκεια λειτουργίας, αποφεύγοντας το επαναλαμβανόμενο κόστος φόρτωσης σε κάθε αίτημα.

Η επεξεργασία των αρχείων PDF γίνεται μέσω της βιβλιοθήκης pdfminer.six, η οποία επιλέχθηκε έπειτα από σύγκριση με εναλλακτικές όπως η PyPDF2 και η pdfplumber. Ο κύριος λόγος αυτής της επιλογής ήταν η καλύτερη συμπεριφορά της pdfminer.six στην εξαγωγή ελληνικών κειμένων: οι άλλες βιβλιοθήκες έτειναν να παράγουν αλλοιωμένους χαρακτήρες ή να αποτυγχάνουν εντελώς σε αρχεία με μη λατινικούς χαρακτήρες, πρόβλημα άμεσα συνδεδεμένο με τη χρήση εκπαιδευτικού υλικού στα ελληνικά. Για τη μετατροπή του κειμένου σε διανύσματα χρησιμοποιήθηκε η βιβλιοθήκη sentence-transformers, η οποία παρέχει μια ενιαία διεπαφή για τη φόρτωση και χρήση δεκάδων προεκπαιδευμένων μοντέλων embedding από το Hugging Face Hub. Για την αποθήκευση και ανάκτηση των διανυσμάτων επιλέχθηκε η ChromaDB, μια ελαφριά διανυσματική βάση δεδομένων σε Python η οποία προσφέρει μόνιμη αποθήκευση στον δίσκο και αναζήτηση πλησιέστερων γειτόνων με βάση τον αλγόριθμο HNSW [13].

Για τη διαχείριση της μνήμης συνομιλίας χρησιμοποιήθηκε η βιβλιοθήκη LangChain και συγκεκριμένα η κλάση ConversationSummaryBufferMemory, η οποία υλοποιεί μια δομή μνήμης με συνδυασμό ακέραιου buffer πρόσφατων turns και αυτόματα παραγόμενης σύνοψης παλαιότερων turns. Αυτή η επιλογή επιτρέπει τη διατήρηση συνεκτικών πολυγύρων διαλόγων χωρίς να υπερβαίνεται το πεπερασμένο μέγεθος του context window του γλωσσικού μοντέλου, όπως αναλύεται στην ενότητα 4.5. Τέλος, για τη μέτρηση του μήκους των κειμένων σε tokens κατά τη φάση του chunking χρησιμοποιήθηκε ο tokenizer cl100k_base της βιβλιοθήκης tiktoken, ο ίδιος tokenizer που χρησιμοποιούν τα μοντέλα της σειράς GPT, αποκλειστικά ως γρήγορος μηχανισμός εκτίμησης μήκους, καθώς η πραγματική μετατροπή σε tokens κατά την παραγωγή απάντησης γίνεται από τον εσωτερικό tokenizer του ίδιου του γλωσσικού μοντέλου.

Το γλωσσικό μοντέλο που χρησιμοποιήθηκε είναι το Llama-Krikri-8B-Instruct, ένα ελληνόφωνο μοντέλο το οποίο αναπτύχθηκε από το Ινστιτούτο Επεξεργασίας του Λόγου (ΙΕΛ) του Ερευνητικού Κέντρου Αθηνά [16]. Το Krikri είναι χτισμένο πάνω στο Llama-3.1-8B της εταιρείας Meta [17] και έχει εκπαιδευτεί επιπλέον (continual pretraining) σε ένα μεγάλο σώμα υψηλής ποιότητας ελληνικών κειμένων μεγέθους περίπου 57 δισεκατομμυρίων tokens, ώστε να βελτιωθούν σημαντικά οι ικανότητές του στην ελληνική γλώσσα σε σύγκριση με το γενικό Llama-3.1. Η αιτιολόγηση της συγκεκριμένης επιλογής, καθώς και η σύγκρισή της με εναλλακτικά μοντέλα που δοκιμάστηκαν κατά την ανάπτυξη, παρουσιάζονται αναλυτικά παρακάτω.

Η εκτέλεση του γλωσσικού μοντέλου γίνεται μέσω της βιβλιοθήκης llama-cpp-python, μιας διεπαφής Python για το llama.cpp, μιας βιβλιοθήκης C/C++ υψηλών επιδόσεων που επιτρέπει την εκτέλεση μοντέλων της οικογένειας Llama σε υπολογιστές με περιορισμένους πόρους, ακόμη και χωρίς εξειδικευμένη GPU. Το llama.cpp υποστηρίζει τη μορφή GGUF, μια συμπαγή δυαδική μορφή αρχείου η οποία αποθηκεύει τόσο τα βάρη του μοντέλου όσο και τα μεταδεδομένα που είναι αναγκαία για την εκτέλεσή του. Κρίσιμο χαρακτηριστικό της βιβλιοθήκης είναι η υποστήριξη ποσοτικοποίησης (quantization), δηλαδή της αναπαράστασης των βαρών του μοντέλου με μειωμένη ακρίβεια (για παράδειγμα 4 bits ανά βάρος αντί για τα συνήθη 16 ή 32) με σκοπό τη δραστική μείωση του μεγέθους και των απαιτήσεων μνήμης, με περιορισμένη υποβάθμιση της ποιότητας των απαντήσεων.

Το περιβάλλον εκτέλεσης του συστήματος τρέχει τοπικά σε επιτραπέζιο υπολογιστή με επεξεργαστή AMD Ryzen 7 και κάρτα γραφικών NVIDIA GeForce GTX 1060 με 3 GB μνήμης VRAM. Η συγκεκριμένη διάταξη επιβάλλει σημαντικούς περιορισμούς στις τεχνικές επιλογές. Το Llama-Krikri-8B στην πλήρη μη ποσοτικοποιημένη μορφή του (FP16) θα απαιτούσε περίπου 16 GB μνήμης, ποσότητα αδύνατη να χωρέσει στη διαθέσιμη VRAM. Ακόμη και μετά την ποσοτικοποίηση στα 4 bits με τη μέθοδο Q4_K_M, το μέγεθος του μοντέλου πέφτει στα ~5 GB, μικρότερο από την πλήρη FP16 μορφή, αλλά και πάλι μεγαλύτερο από τα 3 GB VRAM της κάρτας γραφικών.

Ως αποτέλεσμα, το llama.cpp διαμοιράζει το μοντέλο ανάμεσα στη μνήμη VRAM και στην κύρια μνήμη RAM, εκτελώντας ορισμένα από τα επίπεδα του νευρωνικού δικτύου στη GPU (όσα χωρούν) και τα υπόλοιπα στον επεξεργαστή. Η υβριδική αυτή εκτέλεση είναι σημαντικά πιο αργή από μια καθαρά GPU-based εκτέλεση, αλλά επιτρέπει την εκτέλεση του συστήματος σε τυπικό desktop περιβάλλον.

Η τελική επιλογή του Llama-Krikri-8B προέκυψε έπειτα από διαδοχικές δοκιμές εναλλακτικών μοντέλων τα οποία εγκαταλείφθηκαν λόγω ανεπαρκούς απόδοσης στα ελληνικά. Αρχικά δοκιμάστηκε το LLaMA 3.2 σε δύο μεγέθη, 1B και 3B παραμέτρων. Η επιλογή ήταν φυσική, καθώς τα μικρότερα μοντέλα θα μπορούσαν να εκτελεστούν ταχύτερα στον περιορισμένο διαθέσιμο εξοπλισμό. Το πρόβλημα ωστόσο ήταν η ανεπαρκής ποιότητα των απαντήσεων στα ελληνικά: το μοντέλο εμφάνιζε

τάση ανάμειξης γλωσσών, παρήγε συχνά σχεδόν ακατανόητες προτάσεις και χρησιμοποιούσε λανθασμένη ορολογία ακόμη και για θεμελιώδεις έννοιες του Αντικειμενοστραφούς Προγραμματισμού.

Στη συνέχεια δοκιμάστηκε το Mistral Instruct 7B, το οποίο παρήγε σαφώς καλύτερα αποτελέσματα από το LLaMA 3.2 αλλά εμφάνιζε παρόμοιες αδυναμίες στην ελληνική γλώσσα. Η ποιότητα των απαντήσεων παρέμενε χαμηλή ακόμη και όταν το RAG pipeline παρείχε καλής ποιότητας context, επιβεβαιώνοντας ότι το πρόβλημα εντοπιζόταν στη γλωσσική ικανότητα του ίδιου του μοντέλου και όχι στο σύστημα ανάκτησης. Το Llama-Krikri-8B επιλέχθηκε τελικά ως εξειδικευμένη επιλογή για την ελληνική γλώσσα και παρήγε σημαντικά καλύτερες απαντήσεις, με φυσικότερη ροή, ορθή χρήση της ορολογίας και πιο συνεκτική εκπαιδευτική συμπεριφορά.

Το γραφικό περιβάλλον αλληλεπίδρασης υλοποιήθηκε με καθαρή HTML, CSS και JavaScript, χωρίς τη χρήση κάποιου σύγχρονου frontend framework όπως το React ή το Vue. Η απόφαση αυτή λήφθηκε συνειδητά για λόγους απλότητας και ομαλής ενσωμάτωσης με το FastAPI backend: το frontend σερβίρεται απευθείας από τον server μέσω του μηχανισμού StaticFiles, εξαλείφοντας την ανάγκη για ξεχωριστή διαδικασία compilation, bundling ή hosting. Δεδομένης της σχετικά απλής φύσης της διεπαφής, ένα chat interface με υποστήριξη syntax highlighting και ελάχιστες διαδραστικές λειτουργίες, η χρήση πλήρους frontend framework θα έφερνε περιττή πολυπλοκότητα χωρίς ανάλογο όφελος. Αντιθέτως, η επιλογή vanilla JavaScript επιτρέπει σε έναν μελλοντικό φοιτητή που θα μελετήσει τον κώδικα να κατανοήσει άμεσα τη λειτουργία της διεπαφής, χωρίς να απαιτείται εξοικείωση με συγκεκριμένο framework.

Σε επίπεδο εκτέλεσης, το σύστημα ξεκινά μέσω του backend server ο οποίος εκκινείται με την εντολή uvicorn και εξυπηρετεί αιτήματα στην τοπική διεύθυνση localhost:8000. Κατά την εκκίνηση πραγματοποιείται μια φάση προετοιμασίας (cold start) κατά την οποία φορτώνονται στη μνήμη το embedding μοντέλο, ο reranker και το γλωσσικό μοντέλο Llama-Krikri-8B στην ποσοτικοποιημένη μορφή του. Η φάση αυτή είναι υπολογιστικά απαιτητική και διαρκεί τυπικά αρκετές δεκάδες δευτερόλεπτα έως ένα με δύο λεπτά ανάλογα με την κατάσταση της μνήμης και του δίσκου του συστήματος. Μετά την ολοκλήρωση της φάσης αυτής, το σύστημα διατηρεί όλα τα μοντέλα φορτωμένα στη μνήμη για την ελαχιστοποίηση του χρόνου απόκρισης των επόμενων αιτημάτων, με συνέπεια μια σημαντική αύξηση της κατανάλωσης RAM και VRAM. Η ChromaDB συνδέεται επίσης κατά την εκκίνηση, με τη συλλογή του εκπαιδευτικού υλικού να είναι άμεσα διαθέσιμη για ερωτήματα χωρίς επιπλέον επεξεργασία.

Πριν την οριστικοποίηση των παραπάνω επιλογών, η διαδικασία ανάπτυξης περιέλαβε πολλαπλές δοκιμές εναλλακτικών μοντέλων οι οποίες αποδείχθηκαν ανεπαρκείς και οδήγησαν στις τελικές επιλογές μέσω εμπειρικής σύγκρισης. Η τεκμηρίωση αυτών των αποτυχημένων δοκιμών έχει αυτοτελή αξία, καθώς διασώζει την τεχνική γνώση που προέκυψε από αυτές και αιτιολογεί με εμπειρικά στοιχεία τις τελικές σχεδιαστικές αποφάσεις.

Σε ένα πρώιμο στάδιο της ανάπτυξης δοκιμάστηκαν δύο μικρότερα μοντέλα της οικογένειας LLaMA 3.2. Συγκεκριμένα δοκιμάστηκαν οι εκδοχές με 1 και 3 δισεκατομμύρια παραμέτρους με σκοπό να εξεταστεί αν ένα μικρότερο μοντέλο θα μπορούσε να προσφέρει ικανοποιητική απόδοση με πολύ ταχύτερους χρόνους εκτέλεσης. Ένας άλλος λόγος ήταν και η χαμηλή υπολογιστική ισχύς που υπήρχε στην διαθέσιμη. Τα μοντέλα αυτά εμφάνισαν δύο σοβαρά προβλήματα: πρώτον, σημαντικά υποβαθμισμένη ποιότητα ελληνικής παραγωγής, με συχνά συντακτικά λάθη και αφύσικες διατυπώσεις και δεύτερον, αδυναμία κατανόησης μακροσκελών ελληνικών ερωτημάτων και ανακρίβειες στη σημασιολογική ερμηνεία των όρων του μαθήματος. Τα προβλήματα αυτά είναι

συμβατά με τη γενικότερη παρατήρηση ότι τα μικρότερα μοντέλα γενικού σκοπού, παρά τις εντυπωσιακές ικανότητές τους στην αγγλική, υποφέρουν σημαντικά σε γλώσσες χαμηλότερων πόρων όπως η ελληνική.

Δοκιμάστηκε επιπλέον και το Mistral 7B Instruct, ένα δημοφιλές ανοικτό μοντέλο μεσαίου μεγέθους με καλή φήμη σε αγγλικές εργασίες. Αν και το Mistral επέδειξε καλύτερη ποιότητα από τα μικρότερα LLaMA 3.2, υπέφερε και αυτό από τον ίδιο θεμελιώδη περιορισμό: η εκπαίδευσή του σε κυρίως αγγλικά δεδομένα καθιστούσε τις απαντήσεις του στα ελληνικά αφύσικες και τυπικά λιγότερο ακριβείς σε τεχνικούς όρους. Η τελική επιλογή του Llama-Krikri-8B βασίστηκε στην εμπειρική διαπίστωση ότι ένα μοντέλο ρητά εκπαιδευμένο σε ελληνικά δεδομένα, ακόμη κι αν είναι ελαφρώς μεγαλύτερο από το Mistral, προσφέρει δραματικά καλύτερη εμπειρία χρήσης στο ελληνικό εκπαιδευτικό πλαίσιο. Αντίστοιχες αποτυχημένες δοκιμές έγιναν και με εναλλακτικά μοντέλα embedding, συμπεριλαμβανομένου του native embedder του LLaMA και του προεπιλεγμένου embedder της ChromaDB, τα οποία απορρίφθηκαν για παρόμοιους λόγους υποβαθμισμένης απόδοσης σε ελληνικό περιεχόμενο, οδηγώντας στην επιλογή του multilingual-e5-large-instruct που περιγράφεται αναλυτικά στην επόμενη ενότητα.

Καταλυτική παράμετρος του pipeline indexing είναι η στρατηγική κατάτμησης του εκπαιδευτικού υλικού σε chunks. Η ρύθμιση 250 tokens ανά chunk με επικάλυψη 40 tokens επιλέχθηκε ως ισορροπία ανάμεσα σε δύο ανταγωνιστικές απαιτήσεις.

Από τη μία πλευρά, το context window του Llama-Krikri-8B (N_CTX=4096) πρέπει να χωρέσει το system prompt (~300 tokens), τη σύνοψη ιστορικού συνομιλίας (έως ~1.000 tokens), τα ανακτημένα chunks (TOP_N=4) και την παραγόμενη απάντηση. Με μέγιστο budget 2.500 tokens για το ανακτημένο πλαίσιο, τέσσερα chunks των 250 tokens χωρούν άνετα, αφήνοντας περιθώριο ασφαλείας.

Από την άλλη, τα chunks πρέπει να είναι αρκετά μεγάλα ώστε να φέρουν συνεκτικό σημασιολογικό περιεχόμενο. Μια έννοια του Αντικειμενοστραφούς Προγραμματισμού, όπως ο πολυμορφισμός ή η κληρονομικότητα, τυπικά απαιτεί 150 έως 250 tokens για να εκτεθεί με ορισμό, παράδειγμα και πλαίσιο εφαρμογής. Μικρότερα chunks (50–100 tokens) θα κατακερμάτιζαν τέτοιες έννοιες σε αποσπάσματα στερούμενα συμφραζομένων, ενώ μεγαλύτερα (>400 tokens) θα μείωναν τον αριθμό διακριτών αποσπασμάτων και θα αύξαναν την πιθανότητα συμπερίληψης άσχετου περιεχομένου.

Η προσθήκη επικάλυψης 40 tokens (περίπου 16% του chunk) λειτουργεί ως δίχτυ ασφαλείας για προτάσεις που τέμνονται στα όρια δύο διαδοχικών chunks, εξασφαλίζοντας ότι κρίσιμη πληροφορία δεν θα χαθεί. Η ορθότητα της επιλογής επιβεβαιώνεται εμπειρικά στο Κεφάλαιο 5, όπου η εκδοχή RAG 250/Overlap αναδείχθηκε ως η βέλτιστη μεταξύ των τεσσάρων εξεταζόμενων διαμορφώσεων.

4.2 Embedding Model: multilingual-e5-large-instruct

Η επιλογή του μοντέλου embedding αποτέλεσε μία από τις πιο κρίσιμες τεχνικές αποφάσεις της εργασίας, καθώς η ποιότητα των παραγόμενων διανυσμάτων επηρεάζει άμεσα την απόδοση όλου του RAG pipeline. Το μοντέλο που τελικά υιοθετήθηκε είναι το intfloat/multilingual-e5-large-instruct [12], ένα σύγχρονο instruction-tuned πολυγλωσσικό μοντέλο embedding της οικογένειας E5, το οποίο αναπτύχθηκε από τη Microsoft και κυκλοφόρησε ως τεχνική αναφορά το 2024. Το μοντέλο αρχικοποιείται από το XLM-RoBERTa-large και εκπαιδεύεται σε δύο στάδια: μια φάση ασθενώς επιβλεπόμενης contrastive pre-training σε περίπου 1 δισεκατομμύριο πολυγλωσσικά ζεύγη κειμένου,

και μια επακόλουθη φάση επιβλεπόμενου fine-tuning σε επιμελημένα σύνολα δεδομένων συμπεριλαμβανομένης της χρήσης instruction templates [12].

Οι τεχνικές προδιαγραφές του μοντέλου είναι οι εξής: 24 στρώματα transformer, διάσταση διανύσματος 1024, υποστήριξη για περισσότερες από 100 γλώσσες συμπεριλαμβανομένης της ελληνικής, και μέγιστο μήκος εισόδου 512 tokens, το οποίο ακριβώς καθορίζει το άνω όριο για το μέγεθος ενός chunk κατά τη φάση indexing. Η επιλογή αυτού του μοντέλου στηρίχθηκε σε τρεις βασικούς λόγους. Πρώτον, η ρητή υποστήριξη της ελληνικής γλώσσας στο σύνολο εκπαίδευσης εξασφαλίζει αξιόπιστη σημασιολογική αναπαράσταση ελληνικών κειμένων, κάτι που δεν ισχύει για τα περισσότερα μοντέλα embedding τα οποία είναι εκπαιδευμένα αποκλειστικά σε αγγλικά δεδομένα. Δεύτερον, η χρήση instruction-based προθεμάτων βελτιώνει αισθητά την ποιότητα της ανάκτησης σε σχέση με παλαιότερα μοντέλα της ίδιας οικογένειας. Τρίτον, το μοντέλο εμφανίζει υψηλή απόδοση σε benchmarks σημασιολογικής αναζήτησης όπως το MTEB, αποτέλεσμα το οποίο επαληθεύεται στην τεχνική αναφορά του [12].

Ένα ιδιαίτερο χαρακτηριστικό του multilingual-e5-large-instruct είναι η ασύμμετρη χρήση προθεμάτων ανάλογα με το είδος του κειμένου που κωδικοποιείται. Συγκεκριμένα, τα αποσπάσματα του εκπαιδευτικού υλικού κωδικοποιούνται με το πρόθεμα «passage: » πριν από το κείμενο του chunk, ενώ τα ερωτήματα των φοιτητών κωδικοποιούνται με το πρόθεμα «query: ». Η διάκριση αυτή δεν είναι απλώς συμβατική αλλά ενσωματωμένη στη ροή εκπαίδευσης του μοντέλου: κατά την contrastive pre-training, το μοντέλο έχει μάθει να τοποθετεί ζεύγη (query, passage) κοντά μεταξύ τους στον διανυσματικό χώρο όταν είναι σχετικά, χρησιμοποιώντας ακριβώς αυτά τα προθέματα ως ενδείξεις του ρόλου του κειμένου. Η τήρηση της σύμβασης αυτής είναι κρίσιμη: η αγνόησή της οδηγεί σε αισθητά υποβαθμισμένη ποιότητα ανάκτησης, καθώς το μοντέλο ουσιαστικά λειτουργεί έξω από το περιβάλλον για το οποίο έχει εκπαιδευτεί.

Ένα δεύτερο τεχνικό σημείο αφορά την κανονικοποίηση των διανυσμάτων. Κατά την παραγωγή embeddings, κάθε διάνυσμα κανονικοποιείται σε μοναδιαίο μήκος (L2-normalization) μέσω της παραμέτρου normalize_embeddings=True της sentence-transformers. Η κανονικοποίηση αυτή είναι απαραίτητη για τη σωστή λειτουργία της cosine similarity στη ChromaDB. Όταν τα διανύσματα είναι κανονικοποιημένα, η cosine similarity μεταξύ δύο διανυσμάτων ισούται απλώς με το εσωτερικό τους γινόμενο, και η αναζήτηση πλησιέστερων γειτόνων αντιστοιχεί σε αναζήτηση μέγιστου εσωτερικού γινομένου, μια λειτουργία που είναι εξαιρετικά αποδοτική και καλά υποστηριζόμενη από τη βάση. Για λόγους αποδοτικότητας, η παραγωγή embeddings κατά τη φάση indexing γίνεται σε παρτίδες (batches) μεγέθους 64, ώστε να αξιοποιείται η παραλληλοποίηση των υπολογισμών στη CPU και τη GPU.

Η επιλογή του multilingual-e5-large-instruct προέκυψε μετά από μια σειρά αποτυχημένων προσπαθειών με εναλλακτικές προσεγγίσεις. Σε πρώτη φάση επιχειρήθηκε η απευθείας χρήση του ίδιου του Llama μοντέλου για την παραγωγή embeddings, μέσω των σχετικών δυνατοτήτων που παρέχει η llama-cpp-python. Αυτή η προσέγγιση είναι κατ' αρχήν ελκυστική καθώς εξαλείφει την ανάγκη για ξεχωριστό embedding μοντέλο, μειώνοντας έτσι τον αριθμό των φορτωμένων μοντέλων στη μνήμη. Στην πράξη ωστόσο παρουσιάστηκε πρόβλημα συμβατότητας με τη ChromaDB: τα διανύσματα που παρήγαγε το Llama είτε είχαν διαφορετικές διαστάσεις είτε αποθηκεύονταν σε μη αναμενόμενη μορφή, με αποτέλεσμα η collection της ChromaDB να παραμένει κενή παρά τις επανειλημμένες προσπάθειες εισαγωγής. Σε δεύτερη φάση δοκιμάστηκε ο προεπιλεγμένος embedder της ChromaDB, ο οποίος λειτούργησε τεχνικά αλλά απέδιδε ανεπαρκώς σε ελληνικά κείμενα, παράγοντας ανάκτηση πολύ χαμηλής ποιότητας. Η τελική επιλογή του multilingual-e5-large-instruct μέσω sentence-transformers επέλυσε και τα δύο προβλήματα: εξασφάλισε τεχνική συμβατότητα με τη ChromaDB και σημαντική βελτίωση της ποιότητας της ανάκτησης στα ελληνικά.

4.3 Vector Database Configuration

Για την αποθήκευση των διανυσματικών αναπαραστάσεων και την αποδοτική αναζήτηση ομοιότητας επιλέχθηκε η ChromaDB, μια διανυσματική βάση δεδομένων ανοικτού κώδικα γραμμένη σε Python και ειδικά σχεδιασμένη για εφαρμογές RAG και αναζήτησης ομοιότητας κειμένου. Η επιλογή της ChromaDB έναντι άλλων εναλλακτικών, όπως η FAISS, το Milvus ή η Qdrant, έγινε για τρεις βασικούς λόγους. Πρώτον, η ChromaDB είναι εγγενώς Python-based και ενσωματώνεται απρόσκοπτα σε μια Python εφαρμογή χωρίς την ανάγκη ξεχωριστού διακομιστή ή διαδικασίας. Δεύτερον, υποστηρίζει μόνιμη αποθήκευση στον δίσκο μέσω του PersistentClient, άρα η ευρετηριοποίηση του εκπαιδευτικού υλικού γίνεται μόνο μία φορά και το αποτέλεσμα παραμένει διαθέσιμο μεταξύ επανεκκινήσεων. Τρίτον, η ChromaDB παρέχει μια απλή και σαφή διεπαφή προγραμματισμού η οποία αποκρύπτει την πολυπλοκότητα του υποκείμενου αλγορίθμου ευρετηρίασης, επιτρέποντας στον προγραμματιστή να εστιάσει στη λογική της εφαρμογής αντί στις λεπτομέρειες της υλοποίησης της αναζήτησης.

Στο παρασκήνιο, η ChromaDB χρησιμοποιεί τον αλγόριθμο HNSW (Hierarchical Navigable Small World) για την ευρετηρίαση και αναζήτηση των διανυσμάτων [13]. Ο αλγόριθμος αυτός αποτελεί μια από τις πλέον αποδοτικές προσεγγίσεις για approximate nearest neighbor search σε χώρους υψηλής διάστασης: αντί για εξαντλητική σύγκριση του ερωτήματος με κάθε αποθηκευμένο διάνυσμα, διαδικασία γραμμικής πολυπλοκότητας που γίνεται απαγορευτική καθώς η συλλογή μεγαλώνει. Ο HNSW χτίζει μια ιεραρχική δομή γραφήματος στην οποία η αναζήτηση εκτελείται σε λογαριθμικό χρόνο ως προς το πλήθος των διανυσμάτων. Για την παρούσα εργασία, όπου η συλλογή των chunks είναι σχετικά μικρή (λίγες χιλιάδες), η απόδοση της αναζήτησης είναι πρακτικά στιγμιαία, αλλά η επιλογή του HNSW εξασφαλίζει ότι το σύστημα θα παραμείνει αποδοτικό ακόμη και σε περιπτώσεις επέκτασης του εκπαιδευτικού υλικού σε σημαντικά μεγαλύτερη κλίμακα.

Ως όνομα της συλλογής (collection) που φιλοξενεί τα chunks του εκπαιδευτικού υλικού του μαθήματος επιλέχθηκε το «java_oop_course». Η ονομασία αυτή αντικατοπτρίζει τη μελέτη περίπτωσης της εργασίας, το μάθημα του Αντικειμενοστραφούς Προγραμματισμού σε Java και αφήνει ανοιχτή τη δυνατότητα επέκτασης σε πρόσθετα μαθήματα μέσω επιπλέον collections. Κατά τη δημιουργία της συλλογής, ρυθμίζεται ρητά ως μετρική ομοιότητας η cosine similarity μέσω της παραμέτρου hnsw:space="cosine". Αυτή η επιλογή συμβαδίζει με την κανονικοποίηση των διανυσμάτων που περιγράφηκε στην προηγούμενη ενότητα και εξασφαλίζει ότι η αναζήτηση ομοιότητας πραγματοποιείται με τη σωστή μετρική.

Κάθε εγγραφή στη συλλογή περιέχει τέσσερα στοιχεία: το ίδιο το διάνυσμα (1024 διαστάσεων), το πρωτότυπο κείμενο του chunk, ένα μοναδικό αναγνωριστικό, και ένα σύνολο μεταδεδομένων. Τα μεταδεδομένα περιλαμβάνουν τρία πεδία: το όνομα του αρχείου PDF από το οποίο προέρχεται το chunk (source), τον αύξοντα αριθμό του chunk εντός αυτού του αρχείου (chunk_idx), και το μέγεθός του σε tokens (token_count). Η ύπαρξη των μεταδεδομένων αυτών είναι κρίσιμη για δύο λόγους. Πρώτον, επιτρέπουν την ιχνηλάτηση της προέλευσης κάθε πληροφορίας που εμφανίζεται στις απαντήσεις του συστήματος προς τους φοιτητές, ενισχύοντας τη διαφάνεια και επιτρέποντας σε αυτούς να ανατρέξουν στο πρωτότυπο υλικό για περαιτέρω μελέτη. Δεύτερον, το πεδίο token_count αξιοποιείται κατά το στάδιο του reranking για τον υπολογισμό του token budget: πριν ένα chunk προωθηθεί στο γλωσσικό μοντέλο, ο αριθμός των tokens του προστίθεται στον συνολικό λογαριασμό ώστε να διασφαλιστεί ότι το context που θα τροφοδοτηθεί στο LLM δεν υπερβαίνει τα όρια του context window του.

Η συλλογή αποθηκεύεται μόνιμα σε έναν φάκελο του τοπικού συστήματος αρχείων (chroma_db/) μέσω του PersistentClient αντί του εφήμερου client. Η συγκεκριμένη επιλογή έχει πρακτική σημασία: η φόρτωση του embedding μοντέλου και η κωδικοποίηση των χιλιάδων chunks του εκπαιδευτικού υλικού είναι χρονοβόρες διαδικασίες που διαρκούν αρκετά λεπτά. Θα ήταν εξαιρετικά ανεπαρκές αν έπρεπε να επαναλαμβάνονται σε κάθε εκκίνηση της εφαρμογής. Με τη μόνιμη αποθήκευση, το indexing εκτελείται μία μόνο φορά κατά την αρχική προετοιμασία του συστήματος· στη συνέχεια ο server φορτώνει την προϋπολογισμένη βάση στη μνήμη σε δευτερόλεπτα κατά την εκκίνηση.

4.4 Reranker Integration

Για το στάδιο της επανακατάταξης επιλέχθηκε το μοντέλο BAAI/bge-reranker-v2-m3 [15], ένα cross-encoder reranker της οικογένειας BGE M3 που αναπτύχθηκε από την Beijing Academy of Artificial Intelligence [15]. Η οικογένεια αυτή περιγράφεται αναλυτικά στην εργασία των Chen et al. και χαρακτηρίζεται από τρεις κεντρικές ιδιότητες που είναι γνωστές ως «τα τρία M»: Multi-Linguality (υποστήριξη περισσότερων από 100 γλωσσών), Multi-Functionality (ικανότητα εκτέλεσης dense retrieval, sparse retrieval και multi-vector retrieval) και Multi-Granularity (επεξεργασία εισόδων διαφορετικής κλίμακας, από σύντομες προτάσεις έως μακρά έγγραφα μήκους έως 8.192 tokens). Η συγκεκριμένη παραλλαγή bge-reranker-v2-m3 είναι ένα εξειδικευμένο cross-encoder το οποίο εκπαιδεύτηκε με τεχνική self-knowledge distillation πάνω στη βασική M3 αρχιτεκτονική με στόχο τη μέγιστη δυνατή ακρίβεια στη βαθμολόγηση ζευγών (ερώτηση, απόσπασμα) [15].

Η επιλογή του bge-reranker-v2-m3 έγινε για δύο κύριους λόγους. Πρώτον, η ρητή πολυγλωσσική εκπαίδευσή του, η οποία περιλαμβάνει και ελληνικά κείμενα, το καθιστά κατάλληλο για το πλαίσιο της παρούσας εργασίας σε αντίθεση με τα περισσότερα cross-encoders της βιβλιογραφίας τα οποία είναι εκπαιδευμένα αποκλειστικά σε αγγλικά δεδομένα [14]. Δεύτερον, η οικογένεια BGE αποτελεί μία από τις πλέον αποδοτικές λύσεις στον τομέα της πυκνής ανάκτησης και επανακατάταξης, επιτυγχάνοντας state-of-the-art επιδόσεις σε multilingual benchmarks όπως το MIRACL [15]. Η ενσωμάτωσή του στο pipeline γίνεται μέσω της κλάσης CrossEncoder της βιβλιοθήκης sentence-transformers, η οποία παρέχει μια απλή διεπαφή για τη φόρτωση και χρήση του μοντέλου.

Η λειτουργία του reranker στο pipeline έχει ως εξής: τα 20 υποψήφια chunks που επέστρεψε το στάδιο dense retrieval τροφοδοτούνται στο cross-encoder ως ζεύγη (ερώτημα, chunk), και για κάθε ζεύγος παράγεται μια αριθμητική βαθμολογία σχετικότητας. Σε αντίθεση με τον bi-encoder, που κωδικοποιεί χωριστά το ερώτημα και το chunk πριν υπολογίσει την ομοιότητά τους, ο cross-encoder επεξεργάζεται ταυτόχρονα τα δύο κείμενα μέσω ενός κοινού transformer. Αυτό επιτρέπει στον μηχανισμό attention να εντοπίσει λεπτές αλληλεπιδράσεις ανάμεσα σε συγκεκριμένες λέξεις του ερωτήματος και του chunk. Έτσι, ο cross-encoder μπορεί να αναγνωρίσει σχετικότητα που ο bi-encoder θα έχανε, με κόστος όμως μία κλήση μοντέλου ανά ζεύγος. Για τον λόγο αυτό χρησιμοποιείται μόνο στα 20 υποψήφια chunks του πρώτου σταδίου και όχι σε ολόκληρη τη βάση [14], [15].

Μετά την παραγωγή βαθμολογιών, τα chunks ταξινομούνται φθίνουσα με βάση αυτές και επιλέγονται τα τέσσερα κορυφαία (TOP_N = 4). Πριν όμως τα chunks αυτά προωθηθούν στο στάδιο παραγωγής απάντησης, εφαρμόζονται δύο επιπλέον έλεγχοι οι οποίοι διασφαλίζουν την πρακτική αξιοπιστία του συστήματος. Ο πρώτος είναι ένας έλεγχος προϋπολογισμού tokens: υπολογίζεται το συνολικό μέγεθος των επιλεγμένων chunks σε tokens και, αν υπερβαίνει το επιτρεπτό μέγιστο των 2.500 tokens (παράμετρος MAX_CONTEXT_TOKENS), τα chunks που δεν χωρούν παραλείπονται ξεκινώντας από τα λιγότερο σχετικά. Η τιμή των 2.500 tokens επιλέχθηκε ως εύλογο άνω όριο λαμβάνοντας

υπόψη ότι το context window του γλωσσικού μοντέλου (4.096 tokens στη συγκεκριμένη ρύθμιση) πρέπει να χωρέσει επίσης το system prompt, το ιστορικό συνομιλίας, το ερώτημα και την ίδια την απάντηση. Με αυτόν τον τρόπο αποφεύγεται ο κίνδυνος υπερχείλισης του context window, κάτι που θα προκαλούσε είτε αποκοπή πληροφοριών είτε σφάλματα εκτέλεσης.

Ο δεύτερος έλεγχος είναι το κατώφλι σχετικότητας (RELEVANCE_THRESHOLD). Εάν ακόμη και το κορυφαίο chunk στην κατάταξη επιστρέφει βαθμολογία reranker χαμηλότερη από 0.05, το σύστημα ερμηνεύει αυτό ως ένδειξη ότι κανένα από τα ανακτηθέντα chunks δεν είναι ουσιαστικά σχετικό με το ερώτημα του χρήστη. Σε αυτή την περίπτωση, αντί να προωθηθεί το άσχετο context στο γλωσσικό μοντέλο, κίνηση που θα μπορούσε να οδηγήσει σε ψευδαίσθητη απάντηση (hallucination). Στην περίπτωση αυτή, αυτό που επιστρέφεται προς τον χρήστη είναι ένα τυποποιημένο μήνυμα που αναφέρει ρητά ότι δεν βρέθηκε σχετική πληροφορία στο εκπαιδευτικό υλικό του μαθήματος. Η συγκεκριμένη τιμή κατωφλιού επιλέχθηκε έπειτα από εμπειρική βαθμονόμηση: τιμές πολύ κοντά στο μηδέν απέτυχαν να απορρίψουν πραγματικά άσχετα αποτελέσματα, ενώ υψηλότερες τιμές απέρριπταν και ερωτήματα για τα οποία υπήρχε πραγματική σχετική πληροφορία στη βάση.

4.5 Διαχείριση Context Window

Η σωστή διαχείριση του context window του γλωσσικού μοντέλου αποτελεί έναν από τους πιο κρίσιμους τεχνικούς περιορισμούς στη σχεδίαση ενός RAG συστήματος με πολυγυρική συνομιλία. Ως context window ορίζεται το μέγιστο μήκος κειμένου, σε tokens, που δέχεται το μοντέλο σε μία κλήση ως είσοδο.

Στη συγκεκριμένη υλοποίηση, το context window του Llama-Krikri-8B έχει ρυθμιστεί μέσω της παραμέτρου N_CTX=4096. Η τιμή αυτή αποτελεί συμβιβασμό ανάμεσα σε δύο αντικρουόμενες απαιτήσεις: αρκετά μεγάλη ώστε να χωρά ολόκληρο το prompt (system prompt, ιστορικό, context, ερώτημα) και την παραγόμενη απάντηση, αλλά αρκετά μικρή ώστε η κατανάλωση μνήμης να παραμένει εντός των ορίων του διαθέσιμου εξοπλισμού. Παρότι το Krikri υποστηρίζει θεωρητικά context window έως 128.000 tokens [16], η τιμή αυτή δεν είναι πρακτικά αξιοποιήσιμη σε εξοπλισμό με περιορισμένη μνήμη.

Για κάθε κλήση του γλωσσικού μοντέλου, το prompt που αποστέλλεται αποτελείται από τέσσερα λογικά μέρη τα οποία τοποθετούνται σειριακά εντός του context window. Το πρώτο μέρος είναι το system prompt, που ορίζει τον ρόλο του μοντέλου ως εκπαιδευτικού βοηθού και καθορίζει τον τόνο των απαντήσεων. Περιλαμβάνει επίσης ρητή οδηγία να βασίζει τις απαντήσεις αποκλειστικά στο παρεχόμενο πλαίσιο, αρνούμενο να απαντήσει όταν αυτό δεν περιέχει σχετική πληροφορία, μια οδηγία η οποία συμπληρώνει τον αριθμητικό έλεγχο κατωφλιού σχετικότητας που περιγράφηκε στην ενότητα 4.4 και αναλύεται περαιτέρω στην ενότητα 5.7. Το system prompt καταλαμβάνει περίπου 300 tokens.

Η μεγαλύτερη τεχνική πρόκληση σε ένα πολυγυρικό RAG σύστημα είναι η διαχείριση του ιστορικού συνομιλίας. Σε ένα ρεαλιστικό σενάριο εκπαιδευτικής χρήσης, ο φοιτητής μπορεί να θέσει δεκάδες διαδοχικές ερωτήσεις στο ίδιο session, συχνά με αναφορές σε προηγούμενες ανταλλαγές («όπως είπες πριν», «θυμάσαι το προηγούμενο παράδειγμα;»). Η απλή συσσώρευση ολόκληρου του ιστορικού είναι αδύνατη: μετά από μερικούς γύρους η συνολική κατανάλωση tokens θα ξεπεράσει το context window. Η απλή αποκοπή των παλαιότερων γύρων, από την άλλη, θα οδηγούσε σε απώλεια κρίσιμου πλαισίου και σε ασυνεπείς απαντήσεις.

Η λύση που υιοθετήθηκε είναι η δομή μνήμης ConversationSummaryBufferMemory της βιβλιοθήκης LangChain. Η δομή αυτή συνδυάζει δύο μηχανισμούς: ένα buffer πρόσφατων turns το οποίο διατηρείται ακέραιο, και μια αυτόματα παραγόμενη σύνοψη παλαιότερων turns. Η λογική λειτουργίας είναι η εξής: όσο το συνολικό μέγεθος των πρόσφατων turns του buffer παραμένει μικρότερο από ένα προκαθορισμένο όριο (π.χ. max_token_limit = 1.000 tokens), όλα τα turns αποθηκεύονται αυτούσια και ενσωματώνονται απευθείας στο prompt. Μόλις το όριο αυτό ξεπεραστεί, οι παλαιότερες ανταλλαγές του buffer μεταφέρονται στη σύνοψη: το ίδιο το γλωσσικό μοντέλο καλείται ως συνοψιστής, δέχεται ως είσοδο τα υπό συμπίεση turns και παράγει μια πυκνή αφήγηση των βασικών σημείων της συνομιλίας έως εκείνο το σημείο. Η σύνοψη αυτή αντικαθιστά τα αντίστοιχα turns στο εξής, ενώ τα πιο πρόσφατα turns παραμένουν ακέραια στο buffer. Με αυτόν τον τρόπο, κάθε κλήση του μοντέλου λαμβάνει ως είσοδο μια δομή της μορφής [σύνοψη παλαιών turns] + [πρόσφατα turns] + [retrieved chunks] + [ερώτημα], που εξασφαλίζει συνεκτική συνομιλία με σταθερό κόστος tokens ανεξαρτήτως του πραγματικού μήκους του ιστορικού.

Η ενσωμάτωση της ConversationSummaryBufferMemory του LangChain με το llama-cpp-python, το οποίο δεν αποτελεί επίσημα υποστηριζόμενο backend της LangChain, απαιτούσε την ανάπτυξη μιας προσαρμοσμένης κλάσης-περιτύλιγμα (wrapper class), την LlamaLangChainWrapper. Η κλάση αυτή τυλίγει το instance του llama-cpp-python και εκθέτει τη διεπαφή που απαιτεί το LangChain για να το χρησιμοποιήσει ως γλωσσικό μοντέλο, ικανοποιώντας τις μεθόδους που αναμένει το framework από οποιοδήποτε backend LLM. Με τον μηχανισμό αυτό, η ίδια η παραγωγή της σύνοψης παλαιότερων turns γίνεται μέσω του ίδιου Llama-Krikri-8B που χρησιμοποιείται και για την παραγωγή των τελικών απαντήσεων, εξασφαλίζοντας συνέπεια ύφους και ορολογίας σε ολόκληρο το σύστημα.

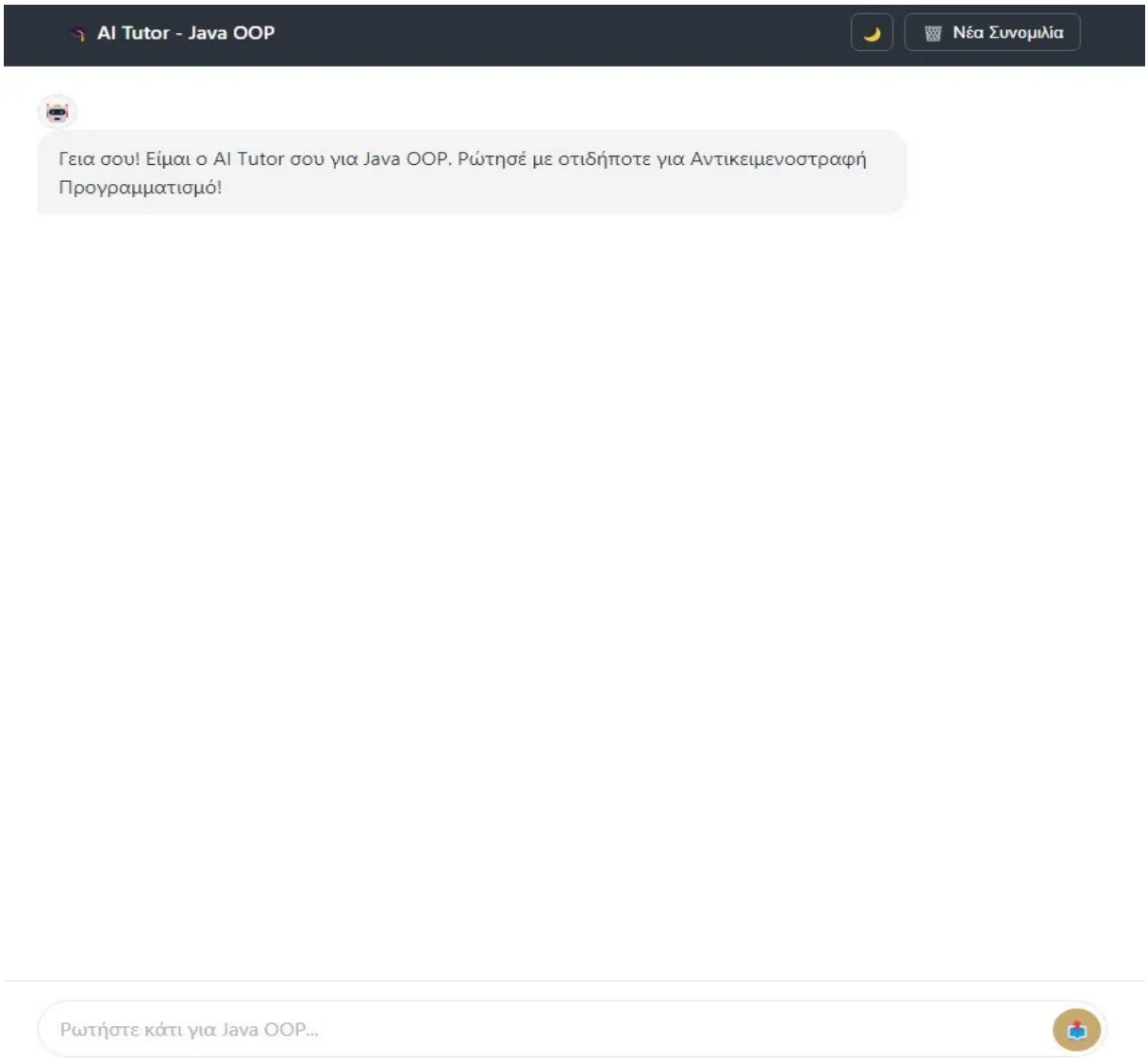
Η διαχείριση των sessions γίνεται μέσω μιας κλάσης ConversationManager η οποία υλοποιεί το μοτίβο singleton στο επίπεδο του backend. Ο ConversationManager διατηρεί ένα λεξικό από ενεργά sessions, όπου κάθε session αναγνωρίζεται από ένα μοναδικό UUID και σχετίζεται με το δικό του, ανεξάρτητο αντικείμενο ConversationSummaryBufferMemory. Όταν φτάνει ένα αίτημα στο /chat endpoint με ένα session_id το οποίο δεν υπάρχει στο λεξικό, δημιουργείται αυτόματα νέο session· όταν φτάνει αίτημα στο /reset endpoint, το αντίστοιχο session διαγράφεται μαζί με όλο το ιστορικό του. Η επιλογή του singleton pattern διασφαλίζει ότι όλα τα sessions μοιράζονται το ίδιο instance του γλωσσικού μοντέλου στη μνήμη, αποφεύγοντας έτσι την απαγορευτικά ακριβή επαναλαμβανόμενη φόρτωση, ενώ ταυτόχρονα κάθε session διατηρεί τη δική του ανεξάρτητη κατάσταση για την συνομιλία του με τον χρήστη.

4.6 Frontend

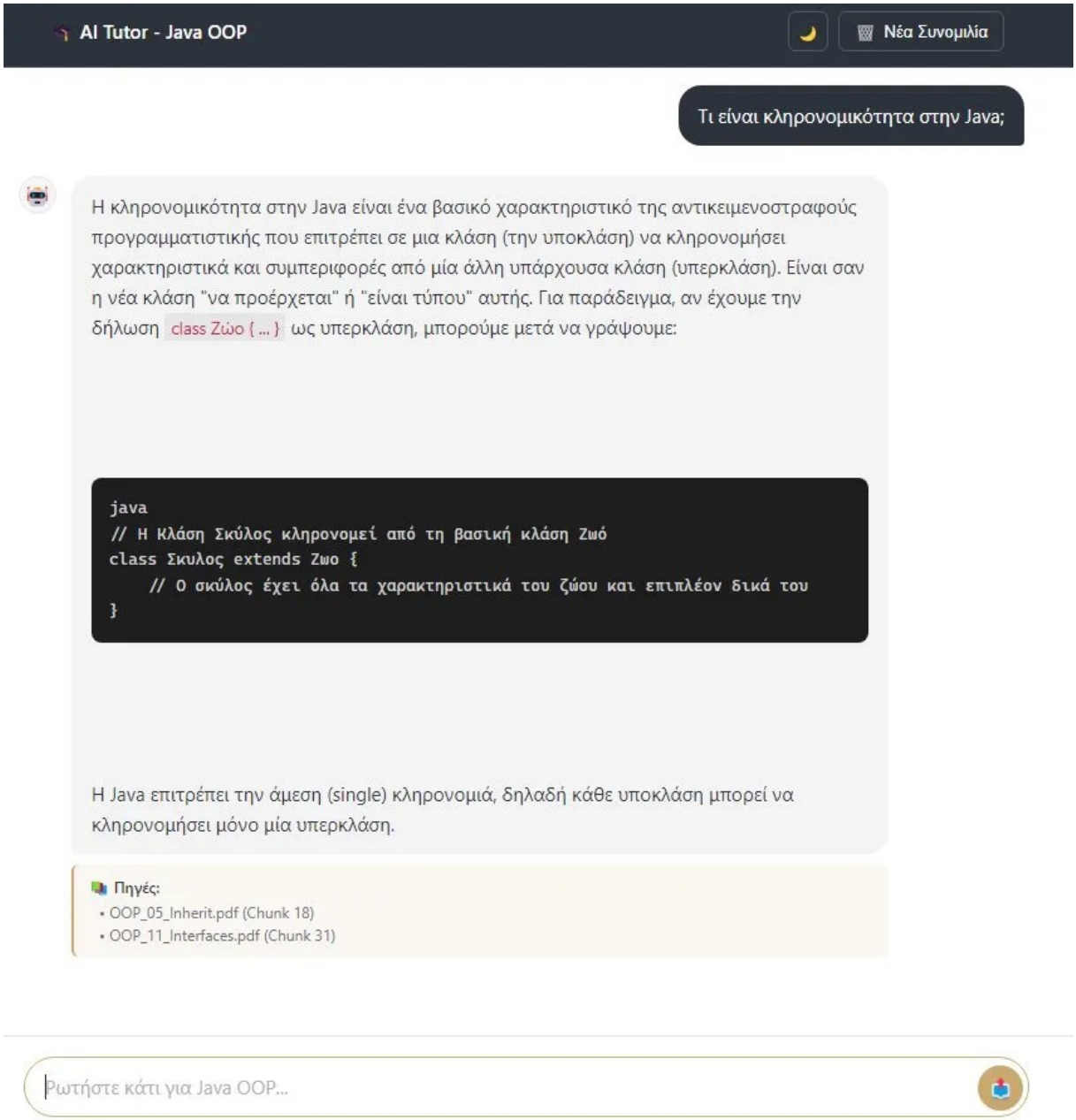
Το γραφικό περιβάλλον χρήστη σχεδιάστηκε με γνώμονα την απλότητα και την άμεση χρηστικότητα, ακολουθώντας το οικείο μοτίβο ενός σύγχρονου chat interface. Η υλοποίηση έγινε με χρήση αποκλειστικά πρότυπων τεχνολογιών web. Χρησιμποιήθηκε HTML5 για τη δομή, CSS3 για την εμφάνιση και vanilla JavaScript (ES6+) για τη λογική. Δεν έγινε η χρήση κάποιου frontend framework όπως το React, το Vue ή το Angular. Αυτή η επιλογή ελήφθη συνειδητά, για λόγους απλότητας, μείωσης εξαρτήσεων και άμεσης ενσωμάτωσης με το FastAPI backend μέσω του μηχανισμού StaticFiles. Ολόκληρος ο κώδικας του frontend είναι οργανωμένος σε τρία αρχεία: το index.html που ορίζει τη δομή της σελίδας, το styles.css που καθορίζει την οπτική εμφάνιση, και το script.js που υλοποιεί τη διαδραστική λογική.

Το βασικό στοιχείο της διεπαφής είναι η περιοχή συνομιλίας, στην οποία τα μηνύματα του χρήστη και του συστήματος εμφανίζονται σε διαφορετικές οπτικές «φυσαλίδες» με διακριτή τοποθέτηση και χρωματισμό. Κάτω από κάθε απάντηση του συστήματος εμφανίζεται το σύνολο των πηγών από τις οποίες αντλήθηκε η πληροφορία, δηλαδή το όνομα του αρχείου PDF και ο αύξων αριθμός του chunk, σε μια μικρότερη, διακριτική γραμματοσειρά. Η εμφάνιση των πηγών ενισχύει τη διαφάνεια του συστήματος και επιτρέπει στον φοιτητή να ανατρέξει απευθείας στο αντίστοιχο σημείο του εκπαιδευτικού υλικού για περαιτέρω μελέτη. Στο Σχήμα 4.2 παρουσιάζεται ενδεικτικό στιγμιότυπο απάντησης με τις πηγές της.

Αυτή η σχεδιαστική επιλογή με την μόνιμη και ορατή παράθεση των πηγών κάτω από κάθε απάντηση εξυπηρετεί έναν διπλό σκοπό που υπερβαίνει την απλή λειτουργικότητα. Σε επίπεδο εμπειρίας χρήστη (UX), δίνει στον φοιτητή τη δυνατότητα να επαληθεύσει άμεσα την προέλευση κάθε πληροφορίας, μετριάζοντας το φαινόμενο της «τυφλής εμπιστοσύνης» που παρατηρείται συχνά στις αλληλεπιδράσεις των χρηστών με συστήματα γενικής συνομιλίας [21]. Σε εκπαιδευτικό επίπεδο, λειτουργεί ως μηχανισμός μεταγνωστικής υποστήριξης: ο φοιτητής δεν δέχεται την απάντηση ως αδιαμφισβήτητη γνώση αλλά ενθαρρύνεται να ανατρέξει στο πρωτότυπο εκπαιδευτικό υλικό για επιβεβαίωση και εμβάθυνση. Με τον τρόπο αυτό, το σύστημα δεν λειτουργεί ως υποκατάστατο της μελέτης αλλά ως οργανωμένος δείκτης πρόσβασης σε αυτή. Η προσέγγιση συμβαδίζει με σύγχρονες παιδαγωγικές αρχές που τονίζουν την καλλιέργεια κριτικής σκέψης αντί της παθητικής κατανάλωσης πληροφορίας [1], [2]. Ένα δευτερεύον αλλά πρακτικά σημαντικό όφελος είναι ότι η ορατότητα των πηγών κατευθύνει εμμέσως και τη συμπεριφορά του ίδιου του φοιτητή κατά τη διατύπωση των ερωτημάτων του: όταν αντιλαμβάνεται ότι η ποιότητα της απάντησης συνδέεται με συγκεκριμένα σημεία του υλικού, τείνει να θέτει πιο εστιασμένα και ευθύβολα ερωτήματα.



Σχήμα 4.1: Στιγμιότυπο της αρχικής οθόνης

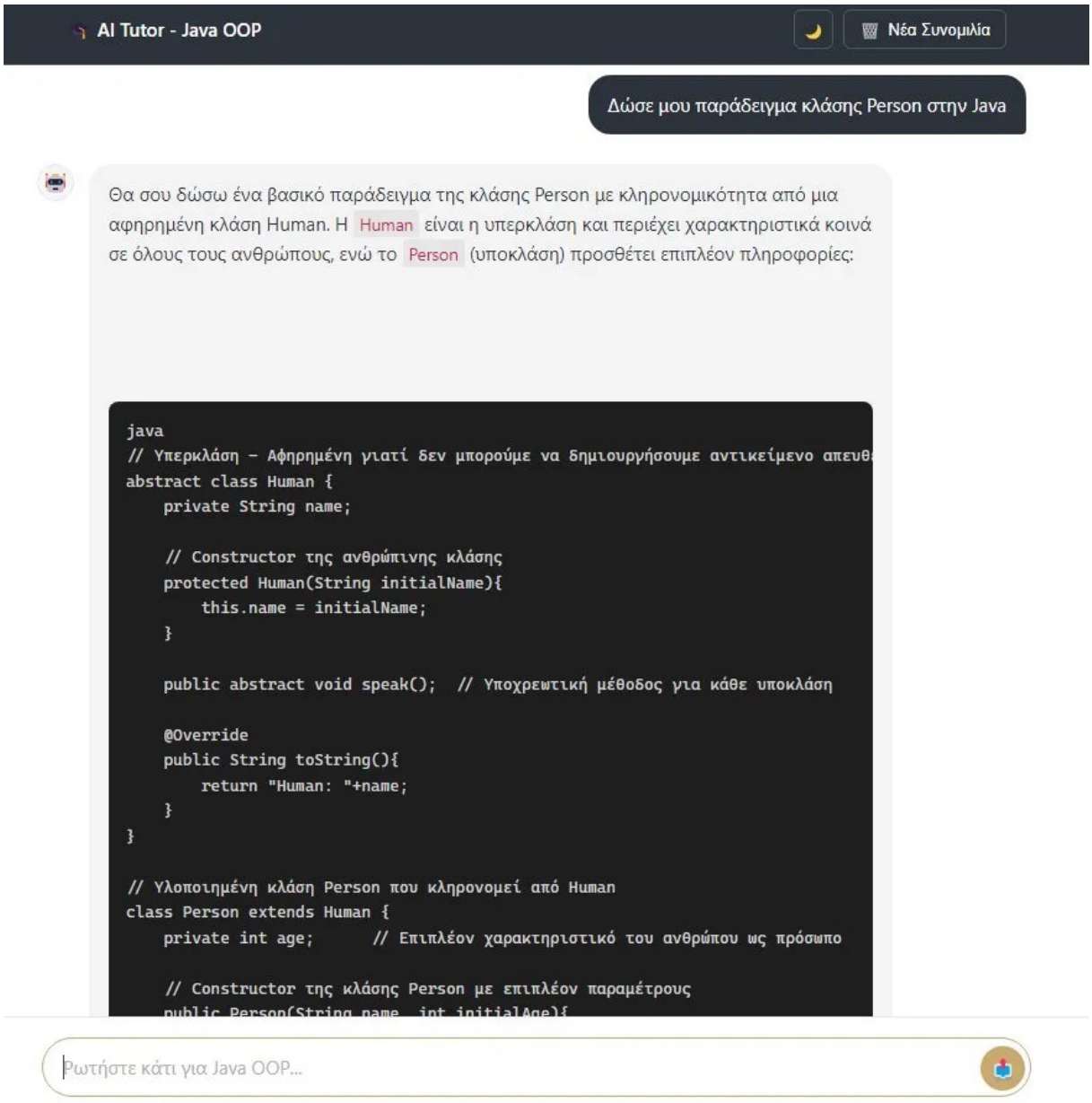


Σχήμα 4.2: Στιγμιότυπο απάντησης με πηγές

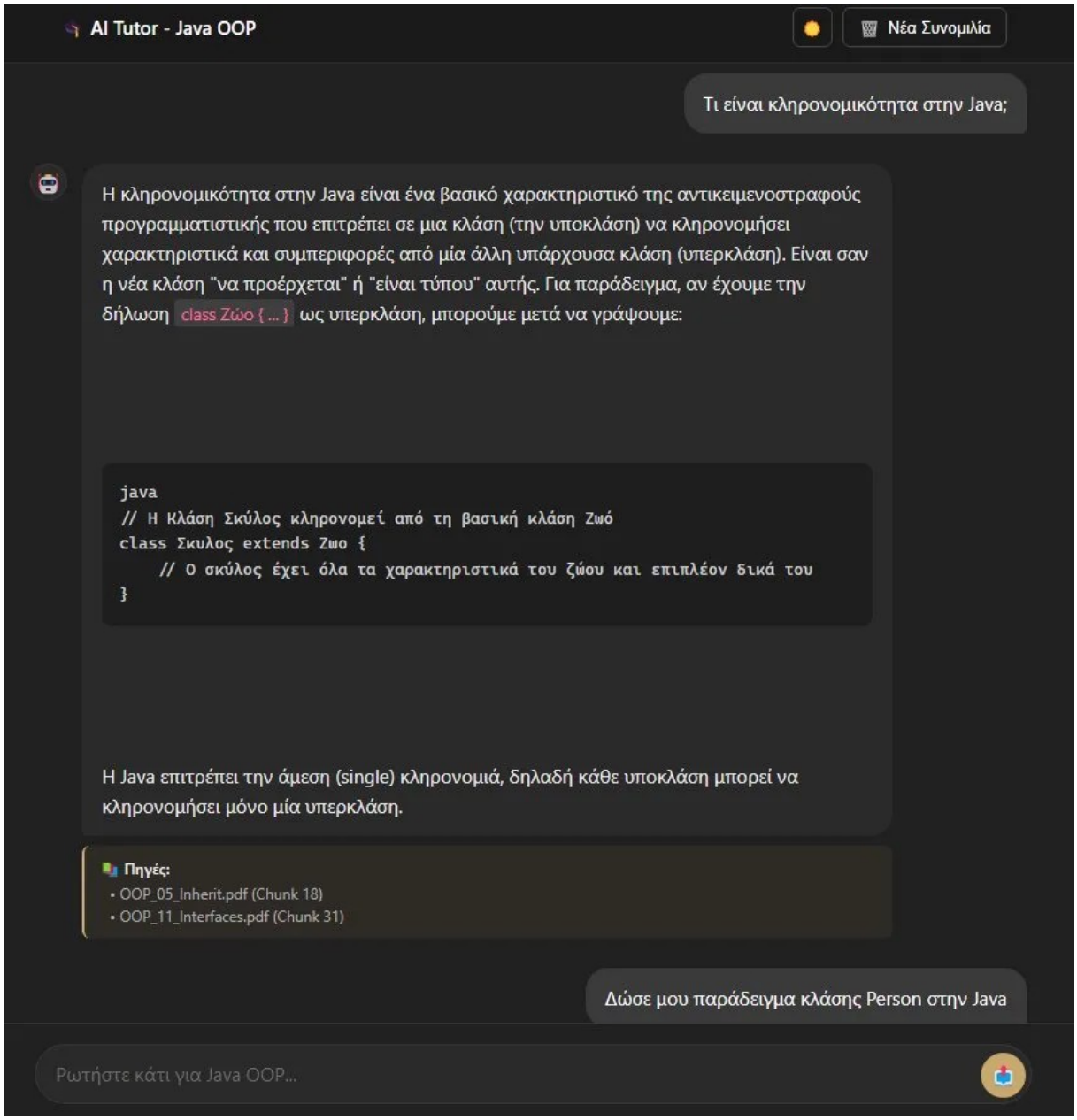
Για την εμφάνιση κώδικα εντός των απαντήσεων, μια συχνή ανάγκη σε μάθημα προγραμματισμού, το frontend υποστηρίζει syntax highlighting τόσο για inline code (μέσα σε backticks) όσο και για block code (τριπλά backticks). Κάθε code block εμφανίζεται σε διακριτή περιοχή με σκούρο φόντο, μονοδιάστατη γραμματοσειρά και χρωματισμό των λέξεων-κλειδιών της εκάστοτε γλώσσας. Κάθε code block συνοδεύεται επίσης από κουμπί αντιγραφής (copy button) στο πάνω δεξί μέρος, που επιτρέπει τη γρήγορη αντιγραφή στο πρόχειρο με ένα κλικ. Η λειτουργικότητα αυτή διευκολύνει τον φοιτητή να δοκιμάσει άμεσα τον παρεχόμενο κώδικα στο δικό του περιβάλλον ανάπτυξης.

Επιπλέον λειτουργίες που υλοποιήθηκαν στο frontend περιλαμβάνουν: έναν typing indicator με κινούμενες τελείες ο οποίος εμφανίζεται κατά την αναμονή της απάντησης του συστήματος, παρέχοντας οπτική ανατροφοδότηση στον χρήστη ότι το αίτημά του επεξεργάζεται. Υπάρχει και υποστήριξη dark mode με δυνατότητα εναλλαγής μέσω ενός κουμπιού, με την προτίμηση του χρήστη

να διατηρείται στο localStorage του περιηγητή ώστε να εφαρμόζεται αυτόματα σε μελλοντικές επισκέψεις. Επίσης παρέχεται αυτόματη προσαρμογή μεγέθους (auto-resize) του πεδίου εισαγωγής μηνύματος καθώς ο χρήστης πληκτρολογεί κείμενο πολλαπλών γραμμών και αποστολή μηνύματος με το πάτημα του πλήκτρου Enter, με υποστήριξη Shift+Enter για εισαγωγή νέας γραμμής χωρίς αποστολή. Υπάρχει κουμπί «Νέα Συνομιλία» το οποίο καλεί το /reset endpoint και ξεκινά μια εντελώς νέα συνεδρία καθώς και responsive σχεδίαση μέσω media queries ώστε η διεπαφή να παραμένει λειτουργική και σε κινητές συσκευές.



Σχήμα 4.3: Στιγμιότυπο σε dark mode



Σχήμα 4.4: Στιγμιότυπο με code highlighting

Η επικοινωνία frontend–backend γίνεται μέσω του native fetch API του περιηγητή, με POST αιτήματα σε JSON μορφή προς το /chat endpoint. Κάθε αίτημα περιλαμβάνει το μήνυμα του χρήστη και το session_id, όπου το τελευταίο δημιουργείται αυτόματα από το backend στο πρώτο μήνυμα κάθε συνεδρίας και αποθηκεύεται τοπικά στο JavaScript context της σελίδας για χρήση στα επόμενα αιτήματα. Κατά την ανάπτυξη και τη δοκιμή του συστήματος, το CORS middleware του FastAPI ρυθμίστηκε ώστε να δέχεται αιτήματα από οποιαδήποτε προέλευση (allow_origins=["*"]), ρύθμιση η οποία διευκολύνει τον έλεγχο σε τοπικό περιβάλλον αλλά θα έπρεπε να περιοριστεί σε συγκεκριμένα domains σε ένα παραγωγικό περιβάλλον.

Κεφάλαιο 5ο: Πειραματική Αξιολόγηση

5.1 Στόχος της Αξιολόγησης

Η πειραματική αξιολόγηση του συστήματος εικονικού διδάσκοντα έχει ως κεντρικό στόχο την ποσοτικοποίηση της επίδρασης της στρατηγικής κατάτμησης (chunking) του εκπαιδευτικού υλικού στην ποιότητα των απαντήσεων του πλήρους RAG συστήματος. Σε ένα σύστημα RAG, η κατάτμηση καθορίζει σε μεγάλο βαθμό τι «βλέπει» το γλωσσικό μοντέλο τη στιγμή της παραγωγής της απάντησης, και επομένως αποτελεί μία από τις πιο θεμελιώδεις σχεδιαστικές παραμέτρους του συστήματος [10].

Πριν από την περιγραφή του πειραματικού πρωτοκόλλου είναι απαραίτητος ο σαφής ορισμός του πειραματικού πλαισίου (experimental setup), δηλαδή του συνόλου των παραμέτρων που παραμένουν σταθερές σε όλες τις εκδοχές που συγκρίνονται. Η ύπαρξη αυτού του πλαισίου είναι απαραίτητη προϋπόθεση για την πειραματική εγκυρότητα καθώς χωρίς αυτό, οι παρατηρούμενες διαφορές μεταξύ εκδοχών δεν μπορούν να αποδοθούν με ασφάλεια στις μεταβαλλόμενες παραμέτρους. Στη συγκεκριμένη αξιολόγηση, το πειραματικό πλαίσιο ορίζεται από τα ακόλουθα σταθερά στοιχεία: το επίσημο εκπαιδευτικό υλικό του μαθήματος Αντικειμενοστραφούς Προγραμματισμού σε Java ως πηγή γνώσης, το γλωσσικό μοντέλο Llama-Krikri-8B-Instruct σε συμπιεσμένη μορφή Q4_K_M, το embedding model multilingual-e5-large-instruct, οι παράμετροι ανάκτησης TOP_K=20 και TOP_N=4, καθώς και οι παράμετροι παραγωγής (temperature=0, max_tokens=500). Η μόνη παράμετρος που μεταβάλλεται μεταξύ των εκδοχών RAG είναι το μέγεθος chunk και η παρουσία ή απουσία επικάλυψης (overlap).

Ο πρώτος ειδικότερος στόχος της αξιολόγησης είναι η σύγκριση του συστήματος με μια γυμνή εκδοχή γλωσσικού μοντέλου χωρίς κανένα RAG μηχανισμό. Αυτή η σύγκριση απαντά στο θεμελιώδες ερώτημα αν ένα ελληνόφωνο γλωσσικό μοντέλο γενικού σκοπού μπορεί από μόνο του, βασιζόμενο μόνο στην παραμετρική του γνώση, να ανταποκριθεί επαρκώς σε ερωτήσεις εξειδικευμένου εκπαιδευτικού περιεχομένου [16]. Έτσι έχουμε μία καλή βάση για να συμπεράνουμε αν και κατά πόσο η πρακτική που ακολουθήσαμε αποδίδει στην πράξη.

Ο δεύτερος στόχος είναι η εμπειρική διερεύνηση της επίδρασης των παραμέτρων κατάτμησης στην τελική ποιότητα των απαντήσεων. Συγκεκριμένα, εξετάζονται δύο τιμές για το μέγεθος chunk (150 και 250 tokens) σε συνδυασμό με δύο επιλογές για την επικάλυψη (με overlap και χωρίς overlap), δίνοντας τέσσερις διαφορετικές διαμορφώσεις RAG. Αυτή η σύγκριση στοχεύει στον εντοπισμό των βέλτιστων παραμέτρων για το συγκεκριμένο κομμάτι του συστήματος και στο κατά πόσο επηρεάζει την τελική παραγωγή της απάντησης που θα δώσει το σύστημα στον χρήστη [10].

Πρωταρχικός στόχος της παρούσας πειραματικής μελέτης ήταν η βελτιστοποίηση των παραμέτρων κατάτμησης, ως πιο θεμελιώδες βήμα. Για την ποσοτική σύγκριση των επιμέρους μηχανισμών reranking, query rewriting και διαχείρισης μνήμης συνομιλίας, που περιγράφονται στα Κεφάλαια 3 και 4 ως μέρος του πλήρους σχεδιασμού, αποτελεί φυσική επέκταση της παρούσας εργασίας και αναφέρεται στις μελλοντικές κατευθύνσεις της Ενότητας 6.3.

Πριν από την περιγραφή του πειραματικού πρωτοκόλλου, είναι απαραίτητος ο ρητός προσδιορισμός των κριτηρίων επιτυχίας (success criteria) για κάθε στόχο [10]. Ο πρώτος στόχος θεωρείται επιτυχής αν το σύστημα RAG, σε τουλάχιστον μία από τις τέσσερις εξεταζόμενες διαμορφώσεις, επιδεικνύει μετρήσιμη βελτίωση στον μέσο όρο βαθμολογίας σε σχέση με τη γυμνή εκδοχή του γλωσσικού μοντέλου σε όλο το σύνολο των εξεταζόμενων ερωτήσεων. Ο δεύτερος στόχος θεωρείται επιτυχής αν

προκύπτουν διακρίσιμες διαφορές μεταξύ των τεσσάρων διαμορφώσεων κατάτμησης, που επιτρέπουν τη διατύπωση τεκμηριωμένων συστάσεων για τη βέλτιστη επιλογή παραμέτρων.

5.2 Πειραματικό Πρωτόκολλο

Για την πειραματική αξιολόγηση σχεδιάστηκε ένα σύνολο είκοσι δοκιμαστικών ερωτήσεων οι οποίες καλύπτουν ένα μεγάλο και γενικό εύρος της διδακτέας ύλης του μαθήματος του Αντικειμενοστραφούς Προγραμματισμού. Το σύνολο αυτό περιλαμβάνει ερωτήσεις διαφορετικών τύπων, ώστε να αξιολογηθεί η συμπεριφορά του συστήματος σε ποικίλα γνωστικά επίπεδα. Το πλήρες σύνολο των είκοσι ερωτήσεων παρατίθεται στο Παράρτημα Α.

Οι ερωτήσεις κατηγοριοποιούνται σε τρεις βασικές ομάδες, κάθε μία από τις οποίες εξετάζει μια διαφορετική πτυχή της λειτουργικότητας του συστήματος. Η πρώτη ομάδα περιλαμβάνει δέκα ερωτήσεις ορισμού βασικών εννοιών, όπως «τι είναι κληρονομικότητα» ή «τι είναι το final keyword» κ.α. Οι ερωτήσεις αυτές αξιολογούν την ικανότητα του συστήματος να ανακαλεί και να διατυπώνει κεντρικούς ορισμούς της ύλης. Η δεύτερη ομάδα περιλαμβάνει τρεις πιο πρακτικές ερωτήσεις και ερωτήσεις παραδείγματος κώδικα, όπως «πώς δημιουργώ και χρησιμοποιώ arrays στην Java» ή «πώς χειριζόμαστε exceptions». Οι ερωτήσεις αυτές απαιτούν από το σύστημα να συνδυάσει θεωρητική γνώση με την παραγωγή ορθών παραδειγμάτων κώδικα. Η τρίτη ομάδα περιλαμβάνει επτά εννοιολογικές και συγκριτικές ερωτήσεις, όπως «ποια η διαφορά μεταξύ overriding και overloading» ή «πώς λειτουργεί ο αλγόριθμος Merge Sort». Οι ερωτήσεις αυτές απαιτούν βαθύτερη κατανόηση και ικανότητα σύνθεσης πληροφοριών από διαφορετικά μέρη της ύλης.

Η αξιολόγηση της ποιότητας των απαντήσεων γίνεται βάσει μιας συνολικής μετρικής ορθότητας [10], σε κλίμακα τριών επιπέδων: 0 για λανθασμένη ή άσχετη απάντηση, 1 για μερικώς σωστή απάντηση και 2 για πλήρη και σωστή απάντηση. Η μετρική αυτή ενσωματώνει στην κρίση του αξιολογητή τόσο την πραγματολογική ορθότητα του περιεχομένου σε σχέση με την ύλη του μαθήματος όσο και την επάρκεια της κάλυψης του ερωτήματος [21]. Η επιλογή απλής κλίμακας τριών βαθμίδων αντί για πιο λεπτομερή πολυδιάστατη αξιολόγηση έγινε σε συμφωνία με το πειραματικό πλαίσιο που κρίθηκε κατάλληλο για την παρούσα εργασία. Πολυδιάστατες αξιολογήσεις (correctness, relevance, completeness, source accuracy) με ευρύτερες κλίμακες αποτελούν φυσική επέκταση που αναφέρεται στις μελλοντικές κατευθύνσεις.

Η βαθμολόγηση των απαντήσεων πραγματοποιήθηκε από τον επιβλέποντα καθηγητή [10], του οποίου η εξειδίκευση στο γνωστικό αντικείμενο και η ανεξαρτησία από τη σχεδίαση και υλοποίηση του συστήματος ενισχύουν την εγκυρότητα της αξιολόγησης. Ο αξιολογητής γνώριζε ποια απάντηση προέρχεται από ποια εκδοχή του συστήματος, καθώς το πειραματικό σχήμα δεν περιλάμβανε διαδικασία τυφλής αξιολόγησης. Η μη τυφλή φύση της αξιολόγησης αναγνωρίζεται ως μεθοδολογικός περιορισμός που συζητείται στην Ενότητα 5.8. Η χρήση πολλαπλών ανεξάρτητων αξιολογητών με διαδικασία διπλά τυφλής αξιολόγησης και υπολογισμό συμφωνίας μεταξύ τους (inter-annotator agreement) [21] θα αποτελούσε περαιτέρω ενίσχυση της μεθοδολογίας και αναφέρεται στις μελλοντικές κατευθύνσεις.

5.3 Διαμόρφωση των Συστημάτων που Συγκρίνονται

Για την αξιολόγηση συγκρίνονται πέντε διαφορετικές εκδοχές του συστήματος. Η πρώτη εκδοχή λειτουργεί ως βάση χωρίς κανέναν μηχανισμό ανάκτησης, ενώ οι υπόλοιπες τέσσερις εκδοχές αντιστοιχούν σε διαφορετικούς συνδυασμούς δύο παραμέτρων κατάτμησης: του μεγέθους του chunk (150 ή 250 tokens) και της παρουσίας ή απουσίας επικάλυψης (overlap) μεταξύ διαδοχικών chunks. Αυτή η προσέγγιση επιτρέπει την απομόνωση της επίδρασης της στρατηγικής κατάτμησης στη συνολική απόδοση του συστήματος [10], κρατώντας σταθερά όλα τα υπόλοιπα στοιχεία της πειραματικής δομής που ορίστηκε.

Εκδοχή	LLM	RAG	Chunk Size	Overlap
(1) LLM Only	Krikri-8B	—	—	—
(2) RAG 250/Overlap	Krikri-8B	✓	250 tokens	✓
(3) RAG 250/NoOverlap	Krikri-8B	✓	250 tokens	—
(4) RAG 150/Overlap	Krikri-8B	✓	150 tokens	✓
(5) RAG 150/NoOverlap	Krikri-8B	✓	150 tokens	—

Πίνακας 5.1: Οι πέντε εκδοχές του συστήματος

Η πρώτη εκδοχή, που αναφέρεται ως LLM Only, αποτελεί την απλούστερη δυνατή διαμόρφωση και χρησιμοποιείται ως βάση χωρίς ανάκτηση. Το γλωσσικό μοντέλο Krikri-8B καλείται απευθείας με το ερώτημα του χρήστη, χωρίς κανέναν μηχανισμό ανάκτησης πληροφορίας από εξωτερική πηγή. Το μοντέλο βασίζεται αποκλειστικά στην παραμετρική του γνώση, δηλαδή σε όσα έμαθε κατά τη διάρκεια της προεκπαίδευσής του, για να παράγει την απάντηση. Η σύγκριση των υπόλοιπων εκδοχών με αυτήν τη γυμνή διαμόρφωση επιτρέπει να εκτιμηθεί η συνολική προστιθέμενη αξία της αρχιτεκτονικής RAG στο συγκεκριμένο εκπαιδευτικό domain.

Η δεύτερη εκδοχή, RAG 250/Overlap, ενεργοποιεί τον πλήρη μηχανισμό RAG με κατάτμηση του εκπαιδευτικού υλικού σε chunks μεγέθους 250 tokens και με επικάλυψη 40 tokens μεταξύ των διαδοχικών τμημάτων. Το ερώτημα του χρήστη μετατρέπεται σε διάνυσμα μέσω του multilingual-e5-large-instruct [12], αναζητούνται τα είκοσι πιο όμοια chunks στη διανυσματιή βάση ChromaDB, επανακατατάσσονται από τον bge-reranker-v2-m3 [14], [15] και τα τέσσερα κορυφαία παρέχονται ως context στο γλωσσικό μοντέλο. Η συγκεκριμένη διαμόρφωση αντιπροσωπεύει τη καθιερωμένη προσέγγιση της βιβλιογραφίας με σχετικά μεγάλο μέγεθος chunk και επικάλυψη που εξασφαλίζει ότι σημαντικές ιδέες δεν διασπώνται σε διαφορετικά chunks.

Η τρίτη εκδοχή, RAG 250/NoOverlap, διατηρεί το μέγεθος των chunks στα 250 tokens αλλά αφαιρεί την επικάλυψη μεταξύ διαδοχικών chunks. Η σύγκρισή της με τη δεύτερη εκδοχή απομονώνει την επίδραση της επικάλυψης διατηρώντας το μέγεθος σταθερό και επιτρέπει να εκτιμηθεί αν η συνεχής ροή πληροφορίας μέσω overlap δικαιολογεί το αυξημένο πλήθος chunks και τον επιπλέον υπολογιστικό φόρτο.

Στην τέταρτη εκδοχή, RAG 150/Overlap, μειώνει το μέγεθος των chunk σε 150 tokens διατηρώντας την επικάλυψη ενεργοποιημένη. Η σύγκρισή της με τη δεύτερη εκδοχή απομονώνει την επίδραση του

μεγέθους του chunk με σταθερή την επικάλυψη. Μικρότερα chunks παράγουν πιο εστιασμένη ανάκτηση αλλά κινδυνεύουν να αποσπάσουν τις βασικές ιδέες από το εννοιολογικό τους πλαίσιο.

Στην πέμπτη εκδοχή, RAG 150/NoOverlap, συνδυάζει το μικρότερο μέγεθος chunk με την απουσία επικάλυψης και αντιπροσωπεύει την πιο μινιμαλιστική RAG διαμόρφωση. Αυτή η εκδοχή χρησιμεύει ως ένα δεύτερο σημείο αναφοράς στο διάγραμμα παραμέτρων και επιτρέπει να εξεταστεί αν η συνδυαστική επίδραση μικρού μεγέθους και απουσίας επικάλυψης οδηγεί σε πτώση απόδοσης ή αντίθετα διατηρεί ανταγωνιστική ποιότητα με μικρότερο υπολογιστικό κόστος.

Όλες οι εκδοχές RAG εκτελούνται στον ίδιο εξοπλισμό (AMD Ryzen 7 με NVIDIA GTX 1060 3GB) και χρησιμοποιούν το ίδιο υποκείμενο γλωσσικό μοντέλο (Llama-Krikri-8B Q4_K_M [16], [17]) με τις ίδιες παραμέτρους εκτέλεσης (temperature=0, max_tokens=500, N_CTX=4096). Η μόνη μεταβλητή που διαφοροποιεί τις τέσσερις RAG εκδοχές είναι ο συνδυασμός μεγέθους chunk και επικάλυψης. Αυτή η επιλογή εξασφαλίζει ότι οι παρατηρούμενες διαφορές στην απόδοση μπορούν να αποδοθούν αποκλειστικά στη στρατηγική που ακολουθείται και όχι σε άλλους παράγοντες, ενισχύοντας την πειραματική εγκυρότητα της σύγκρισης.

5.4 Διαδικασία Εκτέλεσης των Πειραμάτων

Σχεδιαστικά, η εκτέλεση των πειραμάτων οργανώθηκε γύρω από δύο άξονες: την αναπαραγωγιμότητα και τη μεθοδολογική καθαρότητα. Σε ένα προηγούμενο βήμα, καθεμία από τις τέσσερις RAG εκδοχές υποβλήθηκε σε ξεχωριστή φάση indexing με τις αντίστοιχες παραμέτρους κατάτμησης (chunk 250 ή 150 tokens, με ή χωρίς επικάλυψη 40 tokens). Προέκυψαν έτσι τέσσερις διακριτές μόνιμες ChromaDB συλλογές, αποθηκευμένες σε ανεξάρτητους φακέλους και ενεργοποιούμενες αναλόγως κατά την εκτέλεση των ερωτημάτων. Η εκδοχή LLM Only δεν απαίτησε ευρετηρίαση, καθώς λειτουργεί χωρίς μηχανισμό ανάκτησης. Με τη σταθεροποίηση αυτή εξασφαλίζεται ότι, εντός κάθε εκδοχής, οι είκοσι ερωτήσεις απαντώνται από το ίδιο indexed corpus. Τυχόν παρατηρούμενες διαφορές αποδίδονται έτσι αποκλειστικά στη συμπεριφορά του pipeline και όχι σε διακύμανση της ευρετηρίασης.

Στην κυρίως φάση εκτέλεσης, οι είκοσι ερωτήσεις του Παραρτήματος Α υποβλήθηκαν διαδοχικά σε καθεμία από τις πέντε εκδοχές. Για κάθε υποβολή καταγράφηκαν σε δομημένη μορφή JSON τέσσερα στοιχεία: το πλήρες κείμενο της απάντησης, οι πηγές που εμφανίζονται στο φοιτητικό περιβάλλον (όνομα PDF και αύξων αριθμός chunk), ο μετρούμενος χρόνος απόκρισης και τυχόν προειδοποιήσεις ή σφάλματα κατά την εκτέλεση. Αυτή η δομημένη καταγραφή επέτρεψε την εκ των υστέρων αντιπαραβολή και αξιολόγηση των αποτελεσμάτων χωρίς εξάρτηση από την ίδια τη συνεδρία εκτέλεσης. Κάθε εκδοχή εκτελέστηκε σε ανεξάρτητη συνεδρία του backend, με προηγούμενη επανεκκίνηση της διεργασίας ώστε να αποκλειστούν επιδράσεις προηγούμενης κατάστασης μνήμης ή cache στο αποτέλεσμα.

Η αξιολόγηση των εκατό παραγόμενων απαντήσεων (είκοσι ερωτήσεις επί πέντε εκδοχές) ανατέθηκε στον επιβλέποντα καθηγητή, του οποίου η εξειδίκευση στο γνωστικό αντικείμενο και η ανεξαρτησία από τη σχεδίαση και υλοποίηση του συστήματος ενισχύουν την εγκυρότητα της κρίσης. Για κάθε απάντηση αποδόθηκε μία βαθμολογία ορθότητας στην κλίμακα 0–2 που ορίστηκε στην Ενότητα 5.2: 0 για λανθασμένη ή άσχετη απάντηση, 1 για μερικώς σωστή και 2 για πλήρη και σωστή. Επισημαίνεται ότι η διαδικασία δεν ήταν τυφλή: ο αξιολογητής γνώριζε την αντιστοίχιση κάθε απάντησης με την εκδοχή του συστήματος που την παρήγαγε, καθώς το πειραματικό σχήμα δεν περιλάμβανε ανωνυμοποίηση ή τυχαιοποίηση των πηγών. Αυτή η επιλογή υπαγορεύτηκε από τους πρακτικούς περιορισμούς εκπόνησης μιας προπτυχιακής εργασίας και αναγνωρίζεται ως

μεθοδολογικός περιορισμός που συζητείται αναλυτικά στην Ενότητα 5.8. Πιο αυστηρές προσεγγίσεις, όπως διπλά τυφλή διαδικασία και χρήση πολλαπλών ανεξάρτητων αξιολογητών με υπολογισμό συμφωνίας μέσω μετρικών inter-annotator agreement, αναφέρονται ως κατευθύνσεις μελλοντικής έρευνας στην Ενότητα 6.3 [21].

Για τη διασφάλιση της αναπαραγωγιμότητας λήφθηκαν συγκεκριμένα μέτρα κατά την εκτέλεση. Σε επίπεδο μοντέλου, η παράμετρος temperature ορίστηκε σε μηδέν για όλες τις εκδοχές, εξασφαλίζοντας ντετερμινιστική παραγωγή κειμένου: δοθείσας ταυτόσημης εισόδου, το μοντέλο επιστρέφει την ίδια ακολουθία εξόδου σε κάθε κλήση [17]. Οι τέσσερις ChromaDB collections χρησιμοποιήθηκαν ως μόνιμες αποθήκες χωρίς επανα-ευρετηρίαση μεταξύ διαφορετικών συνεδριών αξιολόγησης, ώστε να αποκλειστεί τυχόν διακύμανση οφειλόμενη σε μη ντετερμινιστικά στοιχεία της κατασκευής του HNSW δείκτη. Όλα τα πειράματα εκτελέστηκαν στο ίδιο φυσικό μηχάνημα, με το ίδιο λειτουργικό περιβάλλον και τις ίδιες εκδόσεις βιβλιοθηκών, σε διαδοχικές συνεδρίες. Οι χρόνοι απόκρισης μετρήθηκαν εσωτερικά από το backend με χρήση timestamps, αποφεύγοντας έτσι την επίδραση δικτυακών καθυστερήσεων ή χρόνου rendering του frontend στις καταγεγραμμένες τιμές.

5.5 Αποτελέσματα Αξιολόγησης

Η αξιολόγηση των πέντε εκδοχών στο σύνολο των είκοσι ερωτήσεων παρήγαγε εκατό βαθμολογίες. Κάθε απάντηση βαθμολογήθηκε από τον επιβλέποντα καθηγητή στην κλίμακα που αναφέρθηκε. Τα συγκεντρωτικά αποτελέσματα παρουσιάζονται στον Πίνακα 5.2 για το σύνολο των ερωτήσεων και στον Πίνακα 5.3 ανά κατηγορία ερωτήσεων. Η ανάλυση και η ερμηνεία τους ακολουθούν στην Ενότητα 5.6.

Εκδοχή	**Μέσος Όρος (0-2)**
(1) LLM Only	1.35
(2) RAG 250/Overlap	1.65
(3) RAG 250/NoOverlap	1.55
(4) RAG 150/Overlap	1.55
(5) RAG 150/NoOverlap	1.20

Πίνακας 5.2: Μέσοι όροι βαθμολογίας ανά εκδοχή

Εκδοχή	Ορισμοί (10)	Διαδικαστικές (3)	Εννοιολογικές (7)
(1) LLM Only	1.40	1.33	1.29
(2) RAG 250/Overlap	1.60	1.67	1.71
(3) RAG 250/NoOverlap	1.40	1.67	1.71
(4) RAG 150/Overlap	1.50	1.67	1.57
(5) RAG 150/NoOverlap	1.20	0.67	1.43

Πίνακας 5.3: Μέσοι όροι βαθμολογίας ανά εκδοχή και ανά κατηγορία ερωτήσεων

5.6 Ανάλυση και Συζήτηση Αποτελεσμάτων

Η συνολική χρησιμότητα του RAG επιβεβαιώνεται άμεσα από τον Πίνακα 5.2: τρεις από τις τέσσερις εξεταζόμενες διαμορφώσεις RAG επιτυγχάνουν μέσο όρο βαθμολογίας υψηλότερο από τη γυμνή εκδοχή του γλωσσικού μοντέλου. Συγκεκριμένα, η εκδοχή RAG 250/Overlap παρουσιάζει τη μέγιστη απόδοση με μέσο όρο 1.65, υπερτερώντας κατά 0.30 σε σχέση με την εκδοχή LLM Only (1.35). Παρόμοια εικόνα παρατηρείται στις εκδοχές RAG 250/NoOverlap και RAG 150/Overlap, οι οποίες ισοβαθμούν στο 1.55. Μοναδική διαμόρφωση RAG που υπολείπεται της γυμνής εκδοχής είναι η RAG 150/NoOverlap, με μέσο όρο 1.20. Τα ευρήματα απαντούν θετικά στη δεύτερη ερευνητική ερώτηση: η ενσωμάτωση μηχανισμού RAG στο Krikri-8B επιφέρει μετρήσιμη βελτίωση στην ποιότητα των απαντήσεων, υπό την προϋπόθεση ότι οι παράμετροι κατάτμησης επιλέγονται κατάλληλα [9], [10].

Ο Πίνακας 5.2 αποκαλύπτει συστηματική επίδραση της επικάλυψης μεταξύ διαδοχικών chunks. Σε σταθερό μέγεθος chunk 250 tokens, η ενεργοποίηση του overlap αυξάνει τον μέσο όρο από 1.55 σε 1.65, βελτίωση μικρή αλλά συνεπής με τη θεωρητικά αναμενόμενη συμπεριφορά. Στα 150 tokens, η επίδραση γίνεται δραματικά εντονότερη: η απουσία overlap οδηγεί σε πτώση από 1.55 σε 1.20, δηλαδή 0.35 μονάδες. Αυτή η ασυμμετρία υποδεικνύει ότι η επικάλυψη λειτουργεί ως μηχανισμός προστασίας έναντι της απώλειας πληροφορίας στα όρια των chunks. Σε μικρά chunks η πιθανότητα να διασπαστεί κρίσιμη πληροφορία στο όριο δύο γειτονικών μονάδων είναι μεγαλύτερη, και η απουσία επικάλυψης καθιστά αυτή την απώλεια μη ανακτήσιμη.

Η επίδραση του μεγέθους chunk, εξεταζόμενη με σταθερή την επικάλυψη, παρουσιάζει πιο σύνθετη εικόνα. Στον συνολικό μέσο όρο του Πίνακα 5.2, η μετάβαση από 250 σε 150 tokens με overlap προκαλεί μέτρια μείωση από 1.65 σε 1.55. Όταν όμως διασπάται η ανάλυση ανά κατηγορία ερωτήσεων στον Πίνακα 5.3, φαίνεται ότι η μείωση συγκεντρώνεται κυρίως στις εννοιολογικές ερωτήσεις (από 1.71 σε 1.57), ενώ στις ερωτήσεις ορισμού η εκδοχή 150/Overlap υπολείπεται μόνο οριακά της 250/Overlap (1.50 έναντι 1.60) και στις διαδικαστικές οι δύο εκδοχές ισοβαθμούν στο 1.67. Η ερμηνεία της κατανομής σχετίζεται με τη φύση κάθε κατηγορίας: οι εννοιολογικές και συγκριτικές ερωτήσεις απαιτούν τη σύνθεση πληροφοριών από εκτενέστερα τμήματα της ύλης, κάτι που ευνοεί τα μεγαλύτερα chunks που διατηρούν περισσότερο εννοιολογικό πλαίσιο.

Κατάρρευση της εκδοχής RAG 150/NoOverlap στις διαδικαστικές ερωτήσεις. Το πιο εντυπωσιακό εύρημα του Πίνακα 5.3 αφορά την εκδοχή RAG 150/NoOverlap, η οποία στις διαδικαστικές ερωτήσεις παρουσιάζει μέσο όρο μόλις 0.67, αποτέλεσμα σημαντικά χαμηλότερο τόσο από τη γυμνή εκδοχή (1.33) όσο και από όλες τις υπόλοιπες RAG διαμορφώσεις (1.67 και στις τρεις). Η συνδυαστική επίδραση μικρού μεγέθους chunk και απουσίας επικάλυψης αποδεικνύεται καταστροφική για τις ερωτήσεις που απαιτούν παραγωγή ή κατανόηση παραδειγμάτων κώδικα. Ένα παράδειγμα κώδικα στη Java, μια δήλωση array, ένα try-catch block, μια δομή κλάσης, συχνά υπερβαίνει τα 150 tokens ή απαιτεί εκτεταμένο σχόλιο για να γίνει κατανοητό. Όταν ο κώδικας διασπάται σε διαδοχικά μη επικαλυπτόμενα chunks των 150 tokens, ο reranker επιστρέφει αποσπασματικά τμήματα που στερούνται τη συνεκτική δομή που χρειάζεται το γλωσσικό μοντέλο για να αναπαράγει σωστή σύνταξη. Το εύρημα αυτό συνιστά πρακτική σύσταση: σε εκπαιδευτικά domains με σημαντική παρουσία κώδικα, η συντηρητική επιλογή μεγαλύτερου chunk και ενεργοποίησης overlap είναι ασφαλέστερη.

Παρατηρείται διαφοροποιημένη συνεισφορά του RAG ανά κατηγορία ερωτήσεων. Η κατά κατηγορία ανάλυση του Πίνακα 5.3 αποκαλύπτει ότι το όφελος του RAG δεν κατανέμεται ομοιόμορφα στα διαφορετικά είδη ερωτήσεων. Η μέγιστη υπεροχή των τριών καλύτερων RAG εκδοχών έναντι της LLM Only εμφανίζεται στις εννοιολογικές ερωτήσεις (1.71 έναντι 1.29, διαφορά 0.42 μονάδες) και στις διαδικαστικές (1.67 έναντι 1.33, διαφορά 0.34). Στις ερωτήσεις ορισμού η διαφορά είναι μικρότερη (1.60 της καλύτερης RAG εκδοχής έναντι 1.40 της LLM Only, διαφορά 0.20). Αυτή η εικόνα είναι ερμηνεύσιμη: οι ορισμοί βασικών εννοιών όπως «τι είναι κληρονομικότητα» ή «τι είναι interface», αποτελούν μέρος της ευρέως διαθέσιμης γνώσης για την οποία το γενικού σκοπού γλωσσικό μοντέλο διαθέτει επαρκή παραμετρική κάλυψη ακόμη και χωρίς εξωτερικό context. Αντίθετα, οι διαδικαστικές και ιδίως οι εννοιολογικές-συγκριτικές ερωτήσεις απαιτούν συγκεκριμένη ακρίβεια ή σύνθεση πολλαπλών εννοιών, για τα οποία η ανάκτηση από το επίσημο εκπαιδευτικό υλικό προσφέρει ουσιαστική προστιθέμενη αξία [9].

Συνοψίζοντας, η πειραματική αξιολόγηση παρέχει συνεπή ποσοτική τεκμηρίωση τόσο για τη χρησιμότητα του RAG σε αυτό το εκπαιδευτικό domain όσο και για τη σημασία της προσεκτικής επιλογής παραμέτρων κατάτμησης. Το βασικό μεθοδολογικό συμπέρασμα είναι ότι η απλή προσθήκη μηχανισμού ανάκτησης δεν αρκεί από μόνη της: όταν τα chunks είναι πολύ μικρά και χωρίς overlap, το σύστημα ανακτά αποσπάσματα με ανεπαρκές context, οδηγώντας σε λιγότερο ολοκληρωμένες απαντήσεις από αυτές που θα παρείχε ακόμη και το γυμνό γλωσσικό μοντέλο. Αντίθετα, τα 250 tokens με overlap επιτυγχάνουν καλύτερη ισορροπία μεταξύ σημασιολογικής ακρίβειας και επάρκειας συμφραζομένων. Συνιστώμενη διαμόρφωση για το συγκεκριμένο σύστημα αναδεικνύεται η RAG 250/Overlap, ενώ η εκδοχή RAG 150/NoOverlap λειτουργεί ως αντι-υπόδειγμα και δεν συνιστάται για παρόμοιες εφαρμογές, ιδιαίτερα σε domains με σημαντική παρουσία κώδικα.

5.7 Έλεγχος του Μηχανισμού Refusal

Πέρα από τη συγκριτική μελέτη των πέντε εκδοχών, η αξιολόγηση περιλαμβάνει έναν ποιοτικό έλεγχο ενός κρίσιμου σχεδιαστικού χαρακτηριστικού: του μηχανισμού refusal. Πρόκειται για την ικανότητα του συστήματος να αναγνωρίζει πότε μια ερώτηση βρίσκεται εκτός του εκπαιδευτικού του σκοπού και να απαντά αναλόγως, χωρίς να επινοεί πληροφορίες. Ο έλεγχος αυτός παρουσιάζεται σε ξεχωριστή ενότητα γιατί δεν αποτελεί ποσοτική σύγκριση μεταξύ εναλλακτικών υλοποιήσεων, αλλά λειτουργική επαλήθευση (functional verification) ενός σχεδιαστικού feature που είναι παρόν στο τελικό σύστημα.

Κρίσιμη για την αξιοπιστία του συστήματος σε εκπαιδευτικό πλαίσιο είναι η ικανότητά του να αναγνωρίζει ερωτήματα που βρίσκονται εκτός του γνωστικού του πεδίου και να αποφεύγει την παραγωγή μη τεκμηριωμένων απαντήσεων. Ένας εκπαιδευτικός βοηθός που επιχειρεί να απαντήσει σε κάθε ερώτηση είναι επικίνδυνος. Ακόμη και όταν δεν υπάρχει σχετική πληροφορία στο υλικό, μπορεί να παραπλανήσει τον φοιτητή με λανθασμένες απαντήσεις που αντλούνται από την παραμετρική γνώση του μοντέλου, χωρίς κανέναν μηχανισμό επαλήθευσης. Αντιθέτως, ένα σύστημα το οποίο αναγνωρίζει τα όρια της γνώσης του και παραδέχεται τίμια όταν μια ερώτηση δεν καλύπτεται από το υλικό του, ενισχύει την εμπιστοσύνη του χρήστη και αποτρέπει το φαινόμενο των ψευδαισθήσεων [5], [21]. Η σχεδίαση αυτή συμβαδίζει άλλωστε με τη βασική παιδαγωγική αρχή ότι η αναγνώριση των ορίων της γνώσης είναι σημαντικότερη από την προσποίηση γνώσης.

Ο μηχανισμός refusal στο παρόν σύστημα υλοποιείται σε δύο διακριτά, συμπληρωματικά επίπεδα τα οποία λειτουργούν ως διπλή άμυνα απέναντι σε ερωτήματα εκτός ύλης. Το πρώτο επίπεδο, το οποίο έχει ήδη περιγραφεί αναλυτικά στην ενότητα 4.4, είναι ο αριθμητικός έλεγχος κατωφλιού σχετικότητας στο στάδιο του reranking. Αν η υψηλότερη βαθμολογία σχετικότητας που επιστρέφει ο cross-encoder reranker για τα υποψήφια chunks είναι χαμηλότερη από την παράμετρο RELEVANCE_THRESHOLD (που έχει οριστεί στην τιμή 0.05), το σύστημα ερμηνεύει αυτό ως ισχυρή ένδειξη ότι κανένα από τα chunks στο εκπαιδευτικό υλικό δεν είναι πραγματικά σχετικό με το ερώτημα. Σε αυτή την περίπτωση, το σύστημα δεν προχωρά καν σε κλήση του γλωσσικού μοντέλου και επιστρέφει απευθείας ένα τυποποιημένο μήνυμα το οποίο ενημερώνει τον χρήστη ότι δεν βρέθηκε σχετική πληροφορία στις σημειώσεις του μαθήματος.

Το δεύτερο επίπεδο της άμυνας λειτουργεί στο επίπεδο του ίδιου του γλωσσικού μοντέλου, μέσω του system prompt το οποίο τροφοδοτείται στο Llama-Krikri-8B σε κάθε κλήση. Στο system prompt αυτό έχει ενσωματωθεί μια ρητή οδηγία η οποία καθοδηγεί το μοντέλο να απαντά αποκλειστικά βασισμένο στο παρεχόμενο εκπαιδευτικό πλαίσιο και να δηλώνει ρητά την αδυναμία του όταν η ερώτηση δεν καλύπτεται από αυτό, αντί να παράγει απάντηση από τη γενική του γνώση. Η οδηγία αυτή είναι ουσιώδης για δύο λόγους. Πρώτον, ενεργοποιείται σε περιπτώσεις όπου ο αριθμητικός έλεγχος του reranker περνά αλλά παρ' όλα αυτά το περιεχόμενο είναι επιφανειακά σχετικό χωρίς να περιέχει πραγματική απάντηση στο ερώτημα του χρήστη. Δεύτερον, λειτουργεί ως στοχοθέτηση του γλωσσικού μοντέλου προς την επιθυμητή εκπαιδευτική συμπεριφορά η οποία ακόμη κι αν το μοντέλο θα μπορούσε θεωρητικά να παράγει μια «λογική» απάντηση από την παραμετρική του γνώση, ενθαρρύνεται ρητά να μην το κάνει.

Ο συνδυασμός των δύο αυτών μηχανισμών δημιουργεί ένα συνολικά πιο ανθεκτικό σύστημα. Ο αριθμητικός έλεγχος κατωφλιού στο reranking είναι ένας γρήγορος, φθηνός και πλήρως προκαθορισμένος μηχανισμός ο οποίος απορρίπτει προφανώς άσχετα ερωτήματα πριν καν γίνει κλήση στο γλωσσικό μοντέλο, εξοικονομώντας υπολογιστικούς πόρους. Η prompt-based οδηγία στο επίπεδο του LLM, από την άλλη, είναι πιο λεπτή και μπορεί να διαχειριστεί οριακές περιπτώσεις όπου το ανακτημένο πλαίσιο περιέχει μεν κάποιες σχετικές λέξεις αλλά όχι ουσιαστική απάντηση στο συγκεκριμένο ερώτημα. Κανένας από τους δύο μηχανισμούς από μόνος του δεν θα ήταν επαρκής. Ο λόγος είναι ότι ο πρώτος είναι αρκετά αυστηρός και μπορεί να αστοχήσει σε λεπτές περιπτώσεις, ενώ ο δεύτερος βασίζεται στη συμμόρφωση του γλωσσικού μοντέλου με τις οδηγίες του prompt, συμμόρφωση η οποία δεν είναι ποτέ απόλυτα εγγυημένη.

Ο έλεγχος λειτουργικότητας του συνολικού μηχανισμού refusal πραγματοποιήθηκε με ποιοτικό τρόπο, μέσω χειροκίνητης δοκιμής σε ένα μικρό αλλά αντιπροσωπευτικό σύνολο ερωτημάτων εκτός εκπαιδευτικού πλαισίου. Δόθηκαν απλές κοινωνικές προσφωνήσεις και φράσεις μικρού μήκους

συζήτησης, ερωτήματα για άλλες γλώσσες προγραμματισμού πέραν της Java, ερωτήματα γενικής κουλτούρας άσχετα με τον αντικειμενοστραφή προγραμματισμό, καθώς και ερωτήματα που αφορούν άλλα μαθήματα του προγράμματος σπουδών. Για κάθε τέτοιο ερώτημα, το ζητούμενο ήταν να επιστρέφει το σύστημα είτε κατάλληλη συνομιλιακή απόκριση small talk είτε τυποποιημένο μήνυμα παραπομπής στο γνωστικό αντικείμενο του μαθήματος, χωρίς να επιχειρεί απάντηση από την παραμετρική γνώση του γλωσσικού μοντέλου. Σε όλες τις εξεταζόμενες περιπτώσεις, το σύστημα ανταποκρίθηκε σύμφωνα με την προδιαγραφή, επιβεβαιώνοντας τη λειτουργικότητα και των δύο επιπέδων άμυνας.

Ο έλεγχος συνιστά λειτουργική επαλήθευση και όχι ποσοτικό benchmark. Στόχος δεν ήταν η εξαγωγή αριθμητικού δείκτη ακρίβειας, αλλά η επιβεβαίωση ότι οι δύο μηχανισμοί λειτουργούν όπως αναμένεται σε πρακτικές συνθήκες: ο πρώτος αποτρέπει τα προφανώς άσχετα ερωτήματα πριν φτάσουν στο γλωσσικό μοντέλο, ενώ ο δεύτερος αναλαμβάνει τις οριακές περιπτώσεις όπου το ανακτημένο πλαίσιο περνά τον πρώτο φραγμό. Η μετατροπή σε ποσοτικό benchmark, με ευρύτερο σύνολο ερωτημάτων εκτός ύλης και συγκριτική ανάλυση εναλλακτικών μηχανισμών άρνησης, αναφέρεται ως κατεύθυνση μελλοντικής έρευνας στην Ενότητα 6.3.

5.8 Περιορισμοί της Μελέτης

Πριν από τη μετάβαση στα Συμπεράσματα της εργασίας, είναι σημαντικό να αναγνωριστούν ρητά οι περιορισμοί της παρούσας μελέτης, τόσο σε επίπεδο υλοποίησης όσο και σε επίπεδο μεθοδολογίας αξιολόγησης. Η κριτική αυτή αυτοαξιολόγηση δεν υποβαθμίζει τα ευρήματα αλλά τα πλαισιώνει κατάλληλα, αναδεικνύοντας ταυτόχρονα κατευθύνσεις για μελλοντική εργασία οι οποίες θα μπορούσαν να αντιμετωπίσουν τις αδυναμίες αυτές.

Ο πρώτος και σημαντικότερος περιορισμός αφορά την κλίμακα του συνόλου αξιολόγησης. Οι είκοσι ερωτήσεις που χρησιμοποιήθηκαν αντιπροσωπεύουν ένα διαχειρίσιμο αλλά σχετικά μικρό δείγμα συγκριτικά με τα μεγέθη που χρησιμοποιούνται σε εμπορικά RAG benchmarks. Ενώ η κλίμακα αυτή επαρκεί για την παρούσα μελέτη η οποία επικεντρώνεται στη συγκριτική ανάλυση αρχιτεκτονικών επιλογών και όχι στη δημιουργία ενός επίσημου benchmark, δεν επιτρέπει εξαγωγή στατιστικά σταθερών συμπερασμάτων με υψηλή εμπιστοσύνη. Μια μελλοντική μελέτη με πολλαπλάσιο αριθμό ερωτημάτων, κατηγοριοποιημένων σε περισσότερες και λεπτότερες υποκατηγορίες, θα μπορούσε να παρέχει πιο λεπτομερή και στατιστικά ισχυρή εικόνα.

Ο δεύτερος περιορισμός αφορά την εξάρτηση από την ανθρώπινη αξιολόγηση. Η ανάγκη χειροκίνητης βαθμολόγησης από αξιολογητές καθιστά την αξιολόγηση υποκειμενική σε έναν βαθμό και δυσκολοκλιμάκωτη. Πρόκειται ωστόσο για συνειδητή επιλογή: οι αυτοματοποιημένες μετρικές τύπου BLEU ή ROUGE δεν είναι κατάλληλες για αξιολόγηση εκπαιδευτικών απαντήσεων ανοικτού τύπου, ενώ πιο σύγχρονες LLM-based μετρικές όπως το RAGAS δεν ήταν διαθέσιμες ή ώριμες για ελληνικό περιεχόμενο κατά την εκπόνηση της εργασίας. Η ενσωμάτωση τέτοιων αυτοματοποιημένων μετρικών ως συμπλήρωμα της ανθρώπινης αξιολόγησης αποτελεί σαφή κατεύθυνση για μελλοντική εργασία. Ο τρίτος περιορισμός είναι μεθοδολογικός και αφορά τη διαδικασία αξιολόγησης. Η αξιολόγηση πραγματοποιήθηκε από έναν αξιολογητή, ο οποίος γνώριζε την προέλευση κάθε απάντησης. Το γεγονός αυτό αποτελεί περιορισμό, καθώς θα μπορούσε να εισάγει υποκειμενικότητα στη βαθμολόγηση. Σε μελλοντική επέκταση, η χρήση πολλαπλών ανεξάρτητων αξιολογητών και τυφλής διαδικασίας —με υπολογισμό συμφωνίας μέσω μετρικών όπως το Cohen's κ— θα επέτρεπε πιο αξιόπιστη εκτίμηση της ποιότητας των απαντήσεων [21]. Η συγκεκριμένη μεθοδολογική

κατεύθυνση αναφέρεται και ως σημείο μελλοντικής έρευνας στην Ενότητα 6.3. Η συνδυαστική επιλογή ενός αξιολογητή και μη τυφλής διαδικασίας οφείλεται στους πρακτικούς περιορισμούς της εκπόνησης μιας προπτυχιακής πτυχιακής εργασίας. Ο τέταρτος περιορισμός αφορά τη στενότητα του εκπαιδευτικού πεδίου. Το σύστημα έχει σχεδιαστεί και αξιολογηθεί αποκλειστικά για το μάθημα του Αντικειμενοστραφούς Προγραμματισμού σε Java. Αν και η αρχιτεκτονική είναι γενική και μπορεί να μεταφερθεί σε άλλα μαθήματα χωρίς αλλαγές, η πραγματική απόδοση σε διαφορετικά γνωστικά πεδία όπως για παράδειγμα σε μαθηματικά, όπου η σωστή απάντηση απαιτεί συμβολικό συλλογισμό και όχι μόνο γλωσσική σύνθεση, δεν έχει επαληθευτεί εμπειρικά. Τα ευρήματα της παρούσας εργασίας δεν θα πρέπει επομένως να γενικευτούν άκριτα σε άλλα εκπαιδευτικά πλαίσια.

Ο πέμπτος περιορισμός αφορά τους χρόνους απόκρισης. Η εκτέλεση του συστήματος σε τυπικό desktop εξοπλισμό οδηγεί σε χρόνους απόκρισης της τάξης των δεκάδων δευτερολέπτων ανά ερώτημα, περιορίζοντας τη χρηστικότητά του σε σενάρια υψηλής διαδραστικότητας. Αν και η απόδοση αυτή είναι αποδεκτή για σενάρια προσεκτικής μελέτης όπου ο φοιτητής αναμένει δομημένη απάντηση, δεν προσομοιώνει την εμπειρία ενός εμπορικού chatbot με σχεδόν άμεση απόκριση. Η βελτίωση αυτής της πτυχής απαιτεί είτε περισσότερους υπολογιστικούς πόρους είτε αλλαγές στην αρχιτεκτονική εκτέλεσης.

Τέλος, ο έκτος και τελευταίος περιορισμός αφορά την απουσία αξιολόγησης σε πραγματικές συνθήκες χρήσης από φοιτητές. Η παρούσα μελέτη αξιολογεί τεχνικά την ποιότητα των απαντήσεων του συστήματος, αλλά δεν εξετάζει τη μακροχρόνια επίδρασή του στη μαθησιακή διαδικασία. Ερωτήματα όπως το αν το σύστημα πράγματι βοηθά τη μάθηση, αν οδηγεί σε επιθυμητές αλλαγές στη μελετητική συμπεριφορά των φοιτητών, ή αν εγκυμονεί κινδύνους υπερβολικής εξάρτησης, παραμένουν ανοιχτά προς μελέτη καθώς έχουν ενδιαφέρον και ταυτόχρονα είναι σημαντικά δεδομένα για ένα λογισμικό που είναι φτιαγμένο να βοηθήσει την διαδικασία της μάθησης και μελέτης.

Κεφάλαιο 6ο: Συμπεράσματα

6.1 Κύρια Ευρήματα

Στην παρούσα εργασία αξιολογήθηκε εμπειρικά η επίδραση της στρατηγικής κατάτμησης σε ένα ελληνόφωνο RAG σύστημα εικονικού διδάσκοντα για το μάθημα του Αντικειμενοστραφούς Προγραμματισμού σε Java. Από τη συστηματική σύγκριση πέντε εκδοχών στο σύνολο είκοσι ερωτήσεων προκύπτουν τέσσερα κύρια ευρήματα.

Το πρώτο είναι η ποσοτική επιβεβαίωση της προστιθέμενης αξίας του RAG. Τρεις από τις τέσσερις εξεταζόμενες RAG διαμορφώσεις ξεπέρασαν την απλή εκδοχή του γλωσσικού μοντέλου (1.35), με την βέλτιστη εκδοχή RAG 250/Overlap να φτάνει σε μέσο όρο 1.65 με βελτίωση 0.30 μονάδων στην κλίμακα αξιολόγησης που χρησιμοποιήθηκε. Το εύρημα αυτό απαντά θετικά στο θεμελιώδες ερευνητικό ερώτημα της εργασίας: η ενσωμάτωση μηχανισμού ανάκτησης σε ένα ελληνόφωνο γενικού σκοπού γλωσσικό μοντέλο μπορεί να βελτιώσει μετρήσιμα την ποιότητα των απαντήσεων σε εξειδικευμένο εκπαιδευτικό περιεχόμενο [9].

Το δεύτερο εύρημα είναι η αναγνώριση της παραμέτρου επικάλυψης (overlap) ως κρίσιμης σχεδιαστικής επιλογής. Η εφαρμογή της επικάλυψης βελτίωσε την απόδοση και στα δύο εξεταζόμενα μεγέθη chunk, με τη συνεισφορά της να γίνεται δραματικά εντονότερη στα μικρότερα chunks. Στην εκδοχή 150 tokens, η απουσία overlap προκάλεσε πτώση 0.35 μονάδων και τη χαμηλότερη συνολική επίδοση όλων των εκδοχών (1.20). Αυτή η ασυμμετρία υποδεικνύει ότι η επικάλυψη πρέπει να θεωρείται υποχρεωτική σχεδιαστική επιλογή σε υλοποιήσεις με μικρά chunks.

Το τρίτο εύρημα είναι η εξάρτηση της βέλτιστης διαμόρφωσης από τη φύση της ερώτησης. Οι μεγαλύτερες RAG εκδοχές (250 tokens) υπερτερούν συστηματικά στις εννοιολογικές και συγκριτικές ερωτήσεις, για τις οποίες η σύνθεση πληροφοριών από εκτενέστερα τμήματα της ύλης είναι κρίσιμη (μέγιστο 1.71 έναντι 1.29 της LLM Only). Στις διαδικαστικές ερωτήσεις και ερωτήσεις παραδείγματος κώδικα, η εκδοχή RAG 150/NoOverlap έπεσε στο 0.67, σημαντικά χαμηλότερα από την απλή εκδοχή. Το εύρημα αυτό υποδεικνύει μια πρακτική σύσταση που υπερβαίνει τα όρια της παρούσας εργασίας: σε εκπαιδευτικά domains με σημαντική παρουσία κώδικα ή δομημένων διαδικασιών, η συντηρητική επιλογή μεγαλύτερων chunks με ενεργή επικάλυψη είναι ασφαλέστερη.

Το τέταρτο εύρημα είναι η διαφοροποιημένη συνεισφορά του RAG ανά γνωστικό επίπεδο της ερώτησης. Το όφελος του RAG εμφανίζεται μικρότερο στις ερωτήσεις ορισμού βασικών εννοιών (διαφορά 0.20 μονάδες), όπου το γλωσσικό μοντέλο έχει επαρκή παραμετρική κάλυψη ακόμη και χωρίς εξωτερικό context, και μεγαλύτερο στις πιο απαιτητικές κατηγορίες (διαφορά 0.34 και 0.42 μονάδες). Αυτή η εικόνα επιβεβαιώνει τη θεωρητική πρόβλεψη ότι η συνεισφορά του RAG μεγιστοποιείται όταν απαιτείται ακρίβεια ή σύνθεση πληροφοριών που δεν υπάρχουν συστηματικά στην προεκπαίδευση του μοντέλου.

Συνολικά, η εργασία τεκμηριώνει εμπειρικά ότι η επιλογή των παραμέτρων κατάτμησης δεν είναι ουδέτερη σχεδιαστική απόφαση αλλά παράγοντας με μετρήσιμη επίδραση στην ποιότητα του παραγόμενου εκπαιδευτικού περιεχομένου, και αναδεικνύει τη διαμόρφωση RAG 250/Overlap ως τη συνιστώμενη επιλογή για το συγκεκριμένο σύστημα.

6.2 Συνεισφορά της Εργασίας

Η συνεισφορά της παρούσας εργασίας εντοπίζεται σε τρία κύρια επίπεδα. Σε τεχνικό επίπεδο, παρουσιάζεται μια ολοκληρωμένη, από άκρο σε άκρο υλοποίηση ενός ελληνόγλωσσου RAG συστήματος [16] για εκπαιδευτικούς σκοπούς, με πλήρη τεκμηρίωση των τεχνικών επιλογών, των παραμέτρων και των εμπειρικά τεκμηριωμένων αποφάσεων. Όλος ο κώδικας και οι σχεδιαστικές αποφάσεις τεκμηριώνονται με τέτοιο τρόπο ώστε η υλοποίηση να μπορεί να επαναχρησιμοποιηθεί ως αφετηρία για παρόμοια συστήματα σε άλλα μαθήματα, σε άλλες γλώσσες με περιορισμένους πόρους, ή σε διαφορετικά εκπαιδευτικά περιβάλλοντα. Η επιλογή εργαλείων ανοικτού κώδικα σε κάθε στάδιο του pipeline εξασφαλίζει ότι το σύστημα μπορεί να αναπτυχθεί και να επεκταθεί χωρίς εξαρτήσεις από εμπορικούς παρόχους.

Σε μεθοδολογικό επίπεδο, η εργασία προσφέρει μια συστηματική πειραματική σύγκριση πέντε διαμορφώσεων του συστήματος, η οποία ποσοτικοποιεί τη συνεισφορά της προσέγγισης RAG και της στρατηγικής τεμαχισμού στην ποιότητα των απαντήσεων. Η μελέτη αυτή αναδεικνύει ότι η απλή προσθήκη μηχανισμού ανάκτησης δεν αρκεί από μόνη της: η επιλογή κατάλληλου μεγέθους chunk και η ύπαρξη επικάλυψης (overlap) αποτελούν κρίσιμους παράγοντες που μπορούν να καθορίσουν αν ένα RAG σύστημα θα ξεπεράσει ή θα υπολειφθεί από ένα γυμνό LLM. Για μελλοντικούς σχεδιαστές εκπαιδευτικών RAG συστημάτων, η εμπειρική αυτή τεκμηρίωση προσφέρει χρήσιμες κατευθυντήριες γραμμές για την επιλογή στρατηγικής τεμαχισμού σε αντίστοιχα ελληνόγλωσσα εκπαιδευτικά πλαίσια.

Σε πρακτικό εκπαιδευτικό επίπεδο, η εργασία αναδεικνύει τη δυνατότητα αξιοποίησης σύγχρονων τεχνολογιών Τεχνητής Νοημοσύνης σε ρεαλιστικό πλαίσιο ενός ελληνικού πανεπιστημιακού μαθήματος. Η τοπική εκτέλεση του συστήματος χωρίς εξαρτήσεις από εμπορικά APIs είναι ιδιαίτερα σημαντική σε δύο επίπεδα: πρώτον, εξασφαλίζει πλήρη προστασία της ιδιωτικότητας των φοιτητών, καθώς καμία ερώτηση δεν μεταδίδεται σε τρίτους παρόχους. Δεύτερον, αποκλείει τη δυνατότητα αποσύνδεσης από εξωτερικές υπηρεσίες, κάτι που θα καθιστούσε το σύστημα ευάλωτο σε αλλαγές πολιτικής ή κόστους των παρόχων αυτών. Το σύστημα μπορεί να χρησιμοποιηθεί από φοιτητές ως συμπληρωματικό εργαλείο μελέτης, προσφέροντας άμεση πρόσβαση σε πληροφορίες από τις σημειώσεις του μαθήματος χωρίς την ανάγκη χειροκίνητης αναζήτησης σε δεκάδες αρχεία PDF.

Επιπλέον, η εμπειρική τεκμηρίωση του αποτυχημένου πειραματισμού με εναλλακτικά γλωσσικά μοντέλα (LLaMA 3.2 1B/3B, Mistral 7B) και εναλλακτικά μοντέλα embedding αποτελεί από μόνη της μια συνεισφορά: διασώζει την τεχνική γνώση που προέκυψε από αυτές τις αποτυχίες και αποτρέπει μελλοντικούς ερευνητές και προγραμματιστές από το να επαναλάβουν τις ίδιες δοκιμές χωρίς αιτιολόγηση. Το εύρημα ότι τα γενικά γλωσσικά μοντέλα μικρού ή μεσαίου μεγέθους αποδίδουν ανεπαρκώς στα ελληνικά είναι ιδιαιτέρως σημαντικό για την ελληνόφωνη κοινότητα και αναδεικνύει την αξία εξειδικευμένων μοντέλων όπως το Krikri [16].

Σε σύγκριση με τις περισσότερες αντίστοιχες εργασίες της βιβλιογραφίας, η παρούσα προσέγγιση διαφοροποιείται σε δύο σημεία. Η συντριπτική πλειονότητα των RAG συστημάτων στη διεθνή βιβλιογραφία έχει αναπτυχθεί με αγγλικό περιεχόμενο ως στόχο, αξιοποιώντας εμπορικά μοντέλα όπως τα GPT-3.5 και GPT-4 μέσω APIs. Η παρούσα εργασία διαφοροποιείται σε δύο ουσιαστικά σημεία από αυτή τη συνηθισμένη προσέγγιση. Πρώτον, εστιάζει αποκλειστικά στην ελληνική γλώσσα, απαιτώντας εξειδικευμένες επιλογές σε κάθε στάδιο του pipeline, από το embedding μοντέλο μέχρι το γλωσσικό μοντέλο και το prompt engineering, οι οποίες δεν είναι άμεσα μεταφέρσιμες από την αγγλική βιβλιογραφία. Δεύτερον, επιμένει στην πλήρη τοπική εκτέλεση χωρίς εξαρτήσεις από εμπορικά APIs, επιλογή η οποία οδηγεί σε σημαντικές τεχνικές προκλήσεις (ποσοτικοποίηση,

περιορισμοί υλικού, διαχείριση μνήμης) αλλά προσφέρει αντίστοιχα πλεονεκτήματα ανεξαρτησίας και ιδιωτικότητας.

Πέρα από τις συνεισφορές, είναι εξίσου σημαντικό να ορίζονται ρητά και τα όρια αυτής της συνεισφοράς. Η παρούσα εργασία δεν συνιστά νέα αρχιτεκτονική καινοτομία στο πεδίο του RAG αλλά μάλλον μια προσεκτική, εμπειρικά τεκμηριωμένη σύνθεση υπαρχόντων τεχνικών σε ένα συγκεκριμένο εφαρμοσμένο πλαίσιο. Η συνεισφορά δεν έγκειται στην επινόηση νέων αλγορίθμων αλλά στην ενοποίηση, προσαρμογή και αξιολόγηση ενός συνόλου βέλτιστων πρακτικών σε ένα περιβάλλον (ελληνική γλώσσα, τοπική εκτέλεση, εκπαιδευτικός σκοπός) το οποίο έχει λάβει ελάχιστη ερευνητική προσοχή έως σήμερα. Ο χαρακτηρισμός αυτός είναι ειλικρινής για μια προπτυχιακή πτυχιακή εργασία και αναγνωρίζει τόσο τη θέση της στο ευρύτερο ερευνητικό τοπίο όσο και τον πραγματικά χρήσιμο ρόλο που μπορεί να διαδραματίσει ως αναπαραγώγιμο σημείο αναφοράς για μελλοντικές ελληνόφωνες εφαρμογές.

6.3 Μελλοντικές Επεκτάσεις

Παρά τη λειτουργικότητα και την πρακτική του χρησιμότητα, το συγκεκριμένο σύστημα αφήνει ανοικτές αρκετές κατευθύνσεις για μελλοντική έρευνα και βελτίωση. Οι σημαντικότερες οργανώνονται σε τέσσερις άξονες: τεχνικές βελτιώσεις του pipeline, εκπαιδευτικές επεκτάσεις σε ευρύτερα γνωστικά αντικείμενα, μεθοδολογικές ενισχύσεις του πειραματικού σχεδιασμού, και ευρύτερα ερευνητικά ερωτήματα σχετικά με την παιδαγωγική επίδραση τέτοιων συστημάτων. Στις επόμενες παραγράφους παρουσιάζεται μια συνοπτική επισκόπηση των πιο ώριμων από αυτές τις προοπτικές.

Από τεχνική πλευρά, μια πρώτη προφανής επέκταση αφορά την αντικατάσταση της αποκλειστικά πυκνής ανάκτησης από μια υβριδική προσέγγιση η οποία θα συνδυάζει dense retrieval με κλασική αναζήτηση BM25 [24]. Η υβριδική προσέγγιση έχει δείξει συχνά καλύτερα αποτελέσματα στη βιβλιογραφία, ιδιαίτερα σε περιπτώσεις όπου το ερώτημα περιέχει συγκεκριμένους όρους ή ονόματα τα οποία είναι καλύτερο να ανακτηθούν μέσω λεξικολογικής αντιστοίχισης παρά μέσω σημασιολογικής ομοιότητας. Η ενσωμάτωση BM25 στο υπάρχον pipeline είναι σχετικά απλή και αποτελεί μία από τις πρώτες βελτιώσεις που θα μπορούσαν να δοκιμαστούν.

Μια δεύτερη τεχνική κατεύθυνση είναι η μετάβαση σε μεγαλύτερο γλωσσικό μοντέλο [17] ή σε υλικό hardware με περισσότερη μνήμη VRAM, ώστε να εκτελείται ολόκληρο το μοντέλο απευθείας στη GPU χωρίς υβριδική εκτέλεση. Η βελτίωση αυτή θα μείωνε σημαντικά τον χρόνο απόκρισης και θα επέτρεπε τη χρήση μεγαλύτερων εκδοχών του Krikri (αν αυτές κυκλοφορήσουν στο μέλλον) ή τη χρήση υψηλότερης ανάλυσης ποσοτικοποίησης (Q6 ή Q8 αντί για Q4), με αναμενόμενη βελτίωση στην ποιότητα των απαντήσεων. Μια εναλλακτική κατεύθυνση στον ίδιο άξονα είναι η χρήση συστημάτων batching ή caching τα οποία θα μείωναν το πρακτικό κόστος της εκτέλεσης σε πολλαπλούς ταυτόχρονους χρήστες, αναγκαίο βήμα για μια ενδεχόμενη ανάπτυξη σε πλήρες παραγωγικό περιβάλλον.

Μια τρίτη τεχνική κατεύθυνση αφορά τη σχεδίαση πιο εξελιγμένης στρατηγικής chunking. Η παρούσα υλοποίηση χρησιμοποιεί token-based chunking σταθερού μήκους με overlap, μια προσέγγιση απλή και αποτελεσματική αλλά όχι βέλτιστη. Εναλλακτικές στρατηγικές περιλαμβάνουν semantic chunking (όπου τα chunks διαχωρίζονται στα φυσικά σημεία αλλαγής νοήματος), hierarchical chunking (με πολλαπλά επίπεδα ανάλυσης, πρόταση, παράγραφος, ενότητα) και recursive chunking (όπου μεγαλύτερα chunks αποδομούνται σε μικρότερα όταν χρειάζεται). Η πειραματική σύγκριση αυτών

των στρατηγικών σε ελληνικό εκπαιδευτικό υλικό θα αποτελούσε μια ενδιαφέρουσα ερευνητική επέκταση της παρούσας εργασίας.

Σε επίπεδο πειραματικής μεθοδολογίας, η εργασία προσφέρει αρκετές κατευθύνσεις φυσικής επέκτασης. Πρώτη και πιο άμεση είναι η ποσοτική σύγκριση των επιμέρους μηχανισμών του pipeline που περιγράφονται στα Κεφάλαια 3 και 4 αλλά δεν αξιολογήθηκαν απομονωμένα στην παρούσα μελέτη: επανακατάταξη με cross-encoder, αναδιατύπωση ερωτήματος και διαχείριση μνήμης συνομιλίας. Μια συστηματική αξιολόγηση των τριών αυτών συνιστωσών θα ποσοτικοποιούσε τη συνεισφορά κάθε μίας στη συνολική ποιότητα και θα παρείχε εμπειρική βάση για επόμενες αποφάσεις βελτιστοποίησης. Δεύτερη κατεύθυνση είναι η μετάβαση σε πολυδιάστατη μέτρηση: αντικατάσταση της ενιαίας μετρικής αξιολόγησης με πολλαπλές ανεξάρτητες μετρικές (ορθότητα, συνάφεια, πληρότητα, ακρίβεια πηγών), σε συνδυασμό με πολλαπλούς ανεξάρτητους αξιολογητές υπό διπλά τυφλή διαδικασία και υπολογισμό συμφωνίας μέσω μετρικών inter-annotator agreement [21]. Τρίτη κατεύθυνση είναι η επέκταση του συνόλου αξιολόγησης σε μεγαλύτερο πλήθος ερωτημάτων και η ενσωμάτωση εκτός ύλης ερωτημάτων ως ξεχωριστής κατηγορίας για την ποσοτική αξιολόγηση του μηχανισμού άρνησης.

Από εκπαιδευτική πλευρά, μια ενδιαφέρουσα κατεύθυνση είναι η επέκταση του συστήματος σε περισσότερα μαθήματα [15] του προγράμματος σπουδών. Η υπάρχουσα αρχιτεκτονική υποστηρίζει εγγενώς αυτή την επέκταση: αρκεί η δημιουργία επιπλέον ChromaDB collections, ένα για κάθε μάθημα, με κατάλληλο μηχανισμό δρομολόγησης των ερωτημάτων στη σωστή collection. Πιο σωστά, θα μπορούσε να αναπτυχθεί ένας μηχανισμός αυτόματης κατηγοριοποίησης ερωτημάτων ο οποίος να δρομολογεί το ερώτημα στο σωστό μάθημα, επιτρέποντας στον φοιτητή να απευθύνει ερωτήσεις σε ολόκληρο το πρόγραμμα σπουδών χωρίς να χρειάζεται να επιλέγει μάθημα. Μια τέτοια επέκταση θα καθιστούσε το σύστημα έναν πραγματικό πανεπιστημιακό βοηθό αντί για βοηθό ενός και μόνο μαθήματος.

Μια δεύτερη εκπαιδευτική κατεύθυνση αφορά την ενσωμάτωση περισσότερων αλληλεπιδραστικών δυνατοτήτων για εκπαιδευτική υποστήριξη. Παραδείγματος χάρη, το σύστημα θα μπορούσε να υποστηρίζει τη δημιουργία ερωτήσεων εξάσκησης πάνω στην ύλη, τη δημιουργία quiz με αυτόματη διόρθωση, την παροχή ανάδρασης σε κώδικα που υποβάλλει ο φοιτητής, ή την προσαρμογή του στυλ απάντησης στο επίπεδο εξοικείωσης του χρήστη (αρχάριος, μεσαίος, προχωρημένος). Όλες αυτές οι επεκτάσεις προϋποθέτουν πιο εξελιγμένη χρήση των ικανοτήτων του γλωσσικού μοντέλου αλλά μπορούν να υλοποιηθούν πάνω στην υπάρχουσα αρχιτεκτονική χωρίς ριζικές αλλαγές.

Η εργασία ανοίγει αρκετά ερευνητικά ζητήματα που αξίζουν περαιτέρω διερεύνηση. Πρώτον, δεν έχει μελετηθεί συστηματικά η επίδραση της ποιότητας του εκπαιδευτικού υλικού στη συνολική απόδοση του συστήματος: σε ποιο βαθμό επηρεάζονται τα αποτελέσματα από τη δομή, τη σαφήνεια, την πληρότητα και τη μορφή των πηγαίων PDF. Δεύτερον, χρήζει εξέτασης η επίδραση του γλωσσικού ύφους των ερωτήσεων (επίσημο, καθημερινό, φοιτητικό slang) στην ποιότητα της ανάκτησης. Τρίτον, η σύγκριση της ανθρώπινης αξιολόγησης με αυτοματοποιημένες LLM-based μετρικές θα επέτρεπε την ανάπτυξη πιο κλιμακώσιμων μεθόδων αξιολόγησης εκπαιδευτικών chatbots.

Τέλος, μια ευρύτερη κατεύθυνση μελλοντικής έρευνας αφορά τη διερεύνηση της παιδαγωγικής επίδρασης τέτοιων συστημάτων στη μαθησιακή διαδικασία. Η παρούσα εργασία εστίασε στη σχεδίαση, υλοποίηση και τεχνική αξιολόγηση του συστήματος, αλλά δεν εξέτασε τη μακροχρόνια επίδρασή του στη μάθηση των φοιτητών. Μια ελεγχόμενη παιδαγωγική μελέτη, με δύο ομάδες φοιτητών. Μία ομάδα με πρόσβαση στο σύστημα και μία χωρίς, θα μπορούσε να δώσει απαντήσεις σε κρίσιμα ερωτήματα. Το πρώτο ερώτημα αναφέρεται στο αν βοηθά πραγματικά ένα τέτοιο σύστημα τη

μάθηση ή κρύβει κινδύνους υπερβολικής εξάρτησης ενώ το δεύτερο, στο σημείο της μαθησιακής διαδικασίας θα είναι πιο ωφέλιμο. Οι απαντήσεις σε τέτοια ερωτήματα θα ήταν καθοριστικές για τη σωστή αξιοποίηση τέτοιων τεχνολογιών στο ελληνικό τριτοβάθμιο εκπαιδευτικό σύστημα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] S. Wang et al., "Large Language Models for Education: A Survey and Outlook," IEEE Signal Process. Mag., vol. 42, no. 6, pp. 51–63, Nov. 2025, doi: 10.1109/MSP.2025.3594309.

[2] B. D. Nye, D. Mee, and M. G. Core, "Generative Large Language Models for Dialog-Based Tutoring: An Early Consideration of Opportunities and Concerns," in Proc. Workshop Empowering Education with LLMs, AIED, Tokyo, Japan, Jul. 2023.

[3] D. Venugopalan, Z. Yan, C. Borchers, J. Lin, and V. Aleven, "Combining Large Language Models with Tutoring System Intelligence: A Case Study in Caregiver Homework Support," in Proc. 15th Int. Learn. Analytics Knowl. Conf. (LAK 2025), Dublin, Ireland, Mar. 2025, pp. 1–16, doi: 10.1145/3706468.3706516.

[4] Y. Chen, N. Ding, H.-T. Zheng, Z. Liu, M. Sun, and B. Zhou, "Empowering Private Tutoring by Chaining Large Language Models," in Proc. 33rd ACM Int. Conf. Inf. Knowl. Manage. (CIKM 2024), Boise, ID, USA, Oct. 2024, pp. 354–364, doi: 10.1145/3627673.3679665.

[5] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval Augmentation Reduces Hallucination in Conversation," in Findings of the Assoc. Comput. Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, Nov. 2021, pp. 3784–3803, doi: 10.18653/v1/2021.findings-emnlp.320.

[6] W. Zhang and J. Zhang, "Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review," Mathematics, vol. 13, no. 5, p. 856, Mar. 2025, doi: 10.3390/math13050856.

[7] E. Kasneci et al., "ChatGPT for Good? On Opportunities and Challenges of Large Language Models for Education," Learn. Individ. Differ., vol. 103, Art. no. 102274, Apr. 2023, doi: 10.1016/j.lindif.2023.102274.

[8] P. Béchard and O. M. Ayala, "Reducing Hallucination in Structured Outputs via Retrieval-Augmented Generation," in Proc. 2024 Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technol. (NAACL 2024), Mexico City, Mexico, Jun. 2024, pp. 228–238, doi: 10.18653/v1/2024.naacl-industry.19.

[9] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, Vancouver, Canada, Dec. 2020, pp. 9459–9474.

[10] Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," arXiv preprint arXiv:2312.10997, Mar. 2024. [Online]. Available: https://arxiv.org/abs/2312.10997

[11] V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain Question Answering," in Proc. 2020 Conf. Empirical Methods Natural Lang. Process. (EMNLP), Online, Nov. 2020, pp. 6769–6781, doi: 10.18653/v1/2020.emnlp-main.550.

[12] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei, "Multilingual E5 Text Embeddings: A Technical Report," arXiv preprint arXiv:2402.05672, Feb. 2024. [Online]. Available: https://arxiv.org/abs/2402.05672

[13] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 4, pp. 824–836, Apr. 2020, doi: 10.1109/TPAMI.2018.2889473.

[14] R. Nogueira and K. Cho, "Passage Re-ranking with BERT," arXiv preprint arXiv:1901.04085, Apr. 2020. [Online]. Available: https://arxiv.org/abs/1901.04085

[15] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation," in Findings Assoc. Comput. Linguistics: ACL 2024, Bangkok, Thailand, Aug. 2024, pp. 2318–2335, doi: 10.18653/v1/2024.findings-acl.137.

[16] D. Roussis et al., "Krikri: Advancing Open Large Language Models for Greek," in Findings Assoc. Comput. Linguistics: EMNLP 2025, Suzhou, China, Nov. 2025, pp. 5012–5033. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.268/

[17] A. Grattafiori et al., "The Llama 3 Herd of Models," arXiv preprint arXiv:2407.21783, Nov. 2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[18] A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, Long Beach, CA, USA, Dec. 2017, pp. 5998–6008.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. 2019 Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technol. (NAACL-HLT), Minneapolis, MN, USA, Jun. 2019, pp. 4171–4186, doi: 10.18653/v1/N19-1423.

[20] T. B. Brown et al., "Language Models are Few-Shot Learners," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, Online, Dec. 2020, pp. 1877–1901.

[21] Z. Ji et al., "Survey of Hallucination in Natural Language Generation," ACM Comput. Surv., vol. 55, no. 12, Art. no. 248, Mar. 2023, doi: 10.1145/3571730.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in Proc. 1st Int. Conf. Learn. Representations (ICLR Workshop), Scottsdale, AZ, USA, May 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[23] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in Proc. 2019 Conf. Empirical Methods Natural Lang. Process. and 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), Hong Kong, China, Nov. 2019, pp. 3982–3992, doi: 10.18653/v1/D19-1410.

[24] S. Robertson and H. Zaragoza, "The Probabilistic Relevance Framework: BM25 and Beyond," Foundations and Trends in Information Retrieval, vol. 3, no. 4, pp. 333–389, Apr. 2009, doi: 10.1561/1500000019.

[25] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in Proc. 2014 Conf. Empirical Methods Natural Lang. Process. (EMNLP), Doha, Qatar, Oct. 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[26] J. Johnson, M. Douze, and H. Jégou, "Billion-Scale Similarity Search with GPUs," IEEE Trans. Big Data, vol. 7, no. 3, pp. 535–547, Jul. 2021, doi: 10.1109/TBDATA.2019.2921572.

[27] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query Rewriting in Retrieval-Augmented Large Language Models," in Proc. 2023 Conf. Empirical Methods Natural Lang. Process. (EMNLP), Singapore, Dec. 2023, pp. 5303–5315, doi: 10.18653/v1/2023.emnlp-main.322.

ΠΑΡΑΡΤΗΜΑ Α : Σύνολο Ερωτήσεων Αξιολόγησης

Το παρόν παράρτημα παρουσιάζει το σύνολο των ερωτήσεων που χρησιμοποιήθηκαν για την πειραματική αξιολόγηση του συστήματος εικονικού διδάσκοντα, όπως περιγράφεται στο Κεφάλαιο 5. Οι ερωτήσεις καλύπτουν ένα μεγάλο θεματικό εύρος του μαθήματος του Αντικειμενοστραφούς Προγραμματισμού σε Java, όπως διδάσκεται στο τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου Ελλάδος. Η διαμόρφωσή τους έγινε με γνώμονα την αναπαράσταση ρεαλιστικών ερωτημάτων φοιτητών, τόσο ως προς τη διατύπωση όσο και ως προς το επίπεδο δυσκολίας.

Για τους σκοπούς της αξιολόγησης, οι ερωτήσεις οργανώνονται σε τρεις βασικές κατηγορίες οι οποίες αντιστοιχούν σε διαφορετικά είδη γνωστικών εργασιών που καλείται να επιτελέσει το σύστημα. Κάθε ερώτηση υποβλήθηκε σε όλες τις πέντε εκδοχές του συστήματος που περιγράφονται στην ενότητα 5.3, ώστε να είναι εφικτή η άμεση σύγκριση των απαντήσεών τους. Οι ερωτήσεις παρουσιάζονται στη συνέχεια ακριβώς όπως δόθηκαν στο σύστημα, χωρίς καμία μετατροπή ή προετοιμασία του κειμένου τους.

Η πρώτη κατηγορία περιλαμβάνει ερωτήσεις που ζητούν τον ορισμό ή την εξήγηση μιας θεμελιώδους έννοιας της ύλης. Πρόκειται για το απλούστερο είδος ερωτήματος από τη σκοπιά του RAG pipeline, καθώς η απάντηση τυπικά βρίσκεται συγκεντρωμένη σε ένα ή δύο chunks του εκπαιδευτικού υλικού, και η κύρια πρόκληση είναι η ακριβής ανάκτηση του σχετικού αποσπάσματος και η παρουσίασή του σε κατανοητή μορφή για τον φοιτητή.

1. Τι είναι αντικειμενοστραφής προγραμματισμός και ποια τα βασικά χαρακτηριστικά του;

2. Τι είναι κληρονομικότητα στην Java και πώς τη δηλώνουμε;

3. Τι είναι οι Wrapper classes και γιατί τις χρησιμοποιούμε;

4. Τι είναι το final keyword και πού εφαρμόζεται;

5. Τι είναι abstract class και πότε τη χρησιμοποιούμε;

6. Τι είναι πολυμορφισμός και πώς υλοποιείται στην Java;

7. Τι είναι interface και ποια η διαφορά του από abstract class;

8. Τι είναι inner class και πότε τη χρησιμοποιούμε;

9. Τι είναι generics και γιατί τα χρησιμοποιούμε;

10. Τι είναι αναδρομή και πώς ορίζουμε αναδρομική συνάρτηση;

Η δεύτερη κατηγορία περιλαμβάνει ερωτήσεις διαδικαστικού χαρακτήρα οι οποίες ζητούν από το σύστημα να εξηγήσει τον τρόπο χρήσης ή υλοποίησης ενός συγκεκριμένου μηχανισμού της γλώσσας, συχνά με παροχή παραδείγματος κώδικα. Οι ερωτήσεις αυτής της κατηγορίας θέτουν ιδιαίτερη πρόκληση στο σύστημα καθώς απαιτούν την παραγωγή συντακτικά ορθού κώδικα Java ο οποίος να συμβαδίζει με τις συμβάσεις και τη στυλιστική προσέγγιση του εκπαιδευτικού υλικού.

1. Πώς δημιουργώ και χρησιμοποιώ arrays στην Java;

2. Πώς δουλεύουν τα Strings στην Java και ποιες είναι οι βασικές μέθοδοί τους;

3. Πώς χειριζόμαστε exceptions στην Java;

Η τρίτη κατηγορία περιλαμβάνει τις πιο απαιτητικές ερωτήσεις της αξιολόγησης. Πρόκειται για εννοιολογικές ερωτήσεις και ερωτήσεις σύγκρισης οι οποίες ζητούν από το σύστημα να εξηγήσει τη λειτουργία ενός αλγορίθμου ή να αντιπαραβάλει δύο σχετικές αλλά διακριτές έννοιες. Οι ερωτήσεις αυτής της κατηγορίας απαιτούν τυπικά τη σύνθεση πληροφορίας από περισσότερα του ενός chunks του εκπαιδευτικού υλικού και την παραγωγή απάντησης η οποία δεν εντοπίζεται αυτούσια σε κανένα μεμονωμένο σημείο των σημειώσεων. Αποτελούν επομένως το πιο αυστηρό τεστ της ικανότητας του συστήματος να πραγματοποιεί πραγματική σημασιολογική σύνθεση πάνω στο εκπαιδευτικό υλικό.

1. Πώς λειτουργεί η γραμμική αναζήτηση και πότε χρησιμοποιείται;
2. Πώς λειτουργεί το binary search και ποια η διαφορά του από τη γραμμική αναζήτηση;
3. Εξήγησε τους αλγόριθμους ταξινόμησης Bubble Sort και Selection Sort;
4. Πώς λειτουργεί ο αλγόριθμος Merge Sort;
5. Τι είναι το method overriding και πώς διαφέρει από το overloading;
6. Τι είναι upcasting και downcasting στην Java;
7. Τι είναι binding στην Java και ποια η διαφορά static και dynamic binding;

Σύνολο ερωτήσεων αξιολόγησης: 20. Η κατανομή ανά κατηγορία είναι η ακόλουθη: 10 ερωτήσεις ορισμού, 3 διαδικαστικές ερωτήσεις και 7 εννοιολογικές και συγκριτικές ερωτήσεις. Η ανισομερής αυτή κατανομή αντικατοπτρίζει την ίδια τη δομή του εκπαιδευτικού υλικού, στο οποίο η πλειονότητα των εννοιών εισάγεται μέσω ορισμού πριν συνοδευτεί από παραδείγματα εφαρμογής και εμβάθυνση σε σχέσεις μεταξύ εννοιών. Στην ανάλυση της ενότητας 5.5 τα αποτελέσματα παρουσιάζονται ως μέσοι όροι ανά κατηγορία ώστε η τυχόν ανισομέρεια στον αριθμό των ερωτήσεων να μην επηρεάζει τη συγκριτική αξιολόγηση των εκδοχών του συστήματος.