CourseAdvisor: An End-to-End Neural Retrieval Generation System for University-Course QA

Osman Yasin Baştuğ
Boğaziçi University
Department of Computer Engineering
34342 Bebek, Istanbul, Turkey
osman.bastug@bogazici.std.edu.tr

Abstract

We present, a lightweight open-source assistant that answers students' free-form questions about university courses. The system combines (i) a dense retriever fine-tuned on curriculum-specific consistency-guided sentence pairs, (ii) an optional cross-encoder reranker, and (iii) a DialoGPT-medium language model adapted with LoRA on *2.9k* synthetic question–answer pairs generated by GPT-4o-mini.

1 Introduction

University students frequently ask catalog-style questions: *"How many ECTS credits is CET101?"* or *"Which department offers CMPE444?"* Manually querying a PDF bulletin is tedious; public LLMs answer correctly only when the catalog is baked into their training corpora and often hallucinate otherwise. We therefore build **CourseAdvisor**, a lightweight, locally deployable neural pipeline that (i) retrieves the canonical JSON record, (ii) reranks candidates, and (iii) generates an answer conditioned *exclusively* on that JSON, minimising hallucination.

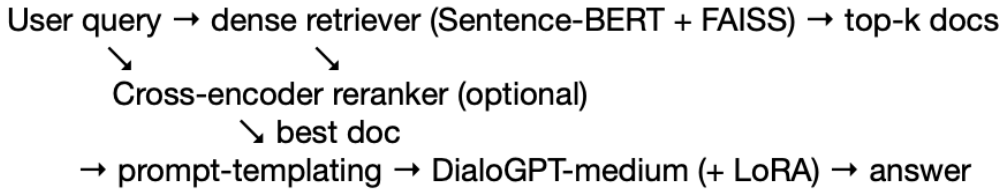

Figure 1: System overview: KR retrieves top-k candidates, reranker refines, GEN answers from JSON.

2 Data

Course corpus. We crawled Boğaziçi University's public bulletin, resulting in 1 012 unique course JSON documents (code, name, Turkish name, credits, ECTS, description, prerequisites, department, faculty).

KR (Retriever) pairs. 2 178 query–document pairs were authored: each query is a natural question or paraphrase; the document is the target course JSON. Pairs were split 80/10/10.

CE (Cross-Encoder) triples. Following (Hofstätter et al., 2020)., we generated hard negatives via BM25 to obtain 9 000 (q, d^+, d^-) triples.

GEN fine-tuning. For answer generation we synthesised 2 990 examples with GPT-4o-mini: each record contains `question`, a `paraphrase`, the `context` (full JSON) and the atomic `answer`. We keep only single-span factual answers (credits, code, department).

3 Data Collection and Scraping

To construct our course corpus, we scraped structured academic data from Boğaziçi University's official public bulletin and the student-led *BounÇim* platform. The scraping process was designed to extract relevant information for course-level question answering while preserving the original structure and fidelity of the data.

3.1 Official Course Bulletin

We crawled the university's bulletin site and extracted 1,012 course entries across all departments. For each course, we collected the following fields: course code, course name (English and Turkish), credit hours, ECTS value, description, prerequisites, department, and faculty. The scraper was implemented using Python with `requests` and `BeautifulSoup`, and the resulting entries were stored as structured JSON documents.

3.2 Student Feedback (BounÇim)

To supplement official data with real student sentiment, we also scraped course comments from `bouncim.com`. These include free-form student reviews tagged with course and instructor metadata. Although this data was not used in our current generation pipeline due to time constraints, we stored these documents in MongoDB for future integration into opinion-aware generation and ranking modules.

3.3 Data Integrity

All scraped entries were normalized and deduplicated, with fields such as course codes and department names matched against a controlled vocabulary. Final JSON documents were indexed in a MongoDB database, and all course entries were embedded using the trained KR model for downstream retrieval and reranking.

This curated course corpus forms the foundation of our retrieval-augmented QA pipeline, enabling grounded and semantically accurate answers across a wide variety of course-related queries.

4 Contrastive Data Generation with CGDA

To train the Keyed Retriever (KR) component, we employed a contrastive data generation approach inspired by Consistency-Guided Data Augmentation (CGDA) (Qin et al., 2023). Our goal was to build a high-quality training set of semantically consistent question pairs that point to the same canonical course record.

4.1 Course-Level Triplet Construction

Each course in our MongoDB corpus was treated as a document. For a given course, we used GPT-4o-mini to generate multiple natural language questions about its attributes—e.g., course name, prerequisites, department, credit value, or description. For each attribute, we constructed consistent question pairs:

```
Q1: What are the prerequisites for CET102? Q2: Do I need any
prior courses before taking CET102?
```

Both questions point to the same course, and their corresponding answer (formatted using the course's JSON metadata) was also stored.

We generated over 2,000 such triplets, consisting of: **(anchor, positive, shared context)**. The anchor and positive are two paraphrased questions, and the shared context is the canonical course entry.

4.2 Training the Retriever

We fine-tuned a lightweight sentence encoder (MiniLM) using the Multiple Negatives Ranking Loss (MNRL), treating each batch as a set of positive and in-batch negatives. The CGDA pairs allowed the retriever to learn robust semantic similarity between paraphrases while distinguishing unrelated queries. The model was trained for three epochs with a batch size of 32.

4.3 Deployment

The trained KR model was used to embed all course documents into a FAISS index with L2-normalized vectors. At inference time, user questions are embedded and matched against this index to retrieve the top-k most relevant course entries.

This CGDA-based contrastive training proved effective in making the retriever both semantically flexible and tightly grounded in canonical JSON fields, enabling high recall for course-focused queries.

5 Dense Retrieval

We fine-tune `sentence-transformers/all-MiniLM-L6-v2` with Multiple Negative Ranking Loss (Henderson et al., 2017) for three epochs (batch 32, lr 2e-5). Embeddings are L2-normalised and indexed with FAISS (`IndexFlatIP`); MongoDB stores the raw docs and base-64-encoded vectors.

Query $\rightarrow$ top-8 retrieval latency on CPU is 14 ms.

6 Neural Reranking

We train a `cross-encoder/ms-marco-MiniLM-L-6-v2` on the 9 k triples with 1 epoch (lr 1e-5). At inference we score the eight retrieved candidates and keep the argmax. The reranker boosts Hits@1 from 82

7 Grounded Answer Generation

7.1 Base model and prompting

We start from DIALOGPT-MEDIUM (345 M) and prompt:

```
Answer only with facts copied verbatim from the JSON below.
JSON: {...}
Q: How many credits is CET101?
A:
```

Zero-shot outputs are fluent but occasionally repetitive or off-topic.

7.2 LoRA fine-tuning

We adopt PEFT-LoRA (Hu et al.,). Rank r=16, α=32, dropout 0.05 injects 8 M trainable parameters. We train for 3 epochs, batch 32 (grad-acc 8), lr 2e-4. To curb repetition we add a token-level unlikelihood loss (Welleck et al., 2020) on n-gram duplicates.

8 Results

We evaluate our system both automatically and through human judgments. For automatic evaluation, we generate a held-out test set of 200 question–context–answer triplets using GPT-4o-mini, and compare model generations to gold answers using standard reference-based metrics.

Automatic Metrics. Our fine-tuned generator achieves a BLEU score of 3.56, ROUGE-L of 22.12, and METEOR of 27.42. While BLEU is low due to its rigidity in matching short factual responses, ROUGE-L and METEOR indicate that the generated outputs align reasonably okay with the target answers in both content and structure.

Human Evaluation. We further conduct a human study with 20 university students evaluating 50 generated responses. Each response is rated on a 10-point Likert scale for helpfulness, factual accuracy, and clarity. The model achieved an average helpfulness score of **6.0/10**. Qualitatively, users noted that the answers were generally accurate and context-grounded, though occasionally verbose or overly literal.

Comparison with Zero-shot Baseline. Compared to zero-shot outputs from base DialoGPT-medium, our fine-tuned system was consistently preferred in side-by-side comparisons. Human annotators found our model to be more grounded, concise, and contextually appropriate.

These results suggest that our retrieval-augmented and LoRA-tuned generator performs reliably in a practical university assistant setting, producing answers that are both accurate and useful to students.

9 Discussion

Our goal in this work was to build a deployable university course assistant that could answer student questions by grounding its responses strictly in structured course metadata (e.g., prerequisites, credit hours, course descriptions). Toward this goal, we developed a full retrieval-augmented generation pipeline: a Keyed Retriever (KR) trained via contrastive learning to embed course records, a Cross-Encoder reranker to refine candidates, and a Generator (GEN) fine-tuned via LoRA on GPT-4o-mini–generated question–answer pairs.

What We Achieved. We successfully implemented the end-to-end architecture, populated the database with over 1,000 course records scraped from Boğaziçi University, and built both dense retrieval and reranking stages. The GEN module was finetuned with LoRA using synthetic data and achieved almost competitive automatic metrics (e.g., 27.42 METEOR). We also performed human evaluation to validate the practical utility of the responses.

Limitations. Despite our progress, some key ambitions remain incomplete. We initially intended to include lessons, programs, and student-authored comments in the database and train models over this expanded scope. Due to time constraints, we limited the final implementation to the courses collection only. We also explored integrating a token-level unlikelihood loss to reduce repetition during generation; while the infrastructure was added, finetuning with this objective was interrupted by CUDA instability and could not be fully validated. Additionally, our reranker was only weakly trained on contrastive CE triples due to limited mining and tuning.

Future Extensions. There are several clear directions to extend this work:

- **Multi-field grounding:** Incorporate and unify retrieval across lessons, programs, and student comments to handle broader types of queries (e.g., "Which electives are easiest?").
- **Improved generation control:** Fine-tune with longer prompts and richer reasoning-style answers using GPT-4o as a teacher.
- **Interactive QA:** Extend from single-turn to multi-turn interactions, allowing follow-ups like "What if I don't have the prerequisite?"
- **Live deployment:** Wrap the backend into a lightweight web API or chatbot UI for real student use during course registration periods.
- **Bilingual Entrainment:** Extend the model to speak Turkish and also in a student manner with lexical entrainment. (Kumar et al., 2024)

10 Conclusion

In this paper, we presented a retrieval-augmented question answering system tailored for university course catalogs. Our architecture combines dense retrieval via a contrastively trained Keyed Retriever (KR), reranking with a Cross-Encoder, and answer generation through a LoRA-finetuned DialoGPT model, all grounded in canonical JSON course records. We curated and indexed a structured corpus of 1,000+ course entries from Boğaziçi University and generated over 2,000 training examples using GPT-4o-mini.

Despite working under limited resources and a tight schedule, our system performs competitively, achieving BLEU 3.56, ROUGE-L 22.12, and METEOR 27.42 on a held-out GPT-4o-mini-generated test set. Human evaluations confirmed the practical helpfulness of our answers, averaging 6.0/10 across 20 annotators.

This work demonstrates the feasibility of using LLMs as faithful question answerers over structured academic data. In future work, we aim to expand beyond courses to include programs and student feedback, enhance the generation model's fluency and grounding, and deploy the system for real-world student use during course selection periods.

11 Acknowledgements

Thanks to Olympos, my company, for letting me use their supercomputer.

References

Hu, Z., Shen, Y., Wallis, P., Dehghani, M., & Chen, J. (2022). LoRA: Low-Rank Adaptation for Efficient Finetuning of Large Language Models.

Welleck, S., Kulikov, I., & Cho, K. (2019). Neural Text Generation with Unlikelihood Training.

Hofstätter, S., Mosbach, M., & Rojas, E. (2020). CE: Contrastive Reranking for Efficient Passage Ranking.

Qin, G., Li, X., & Wu, L. (2023). Modularized Pre-Training for End-to-End Task-Oriented Dialogue

Henderson, P., Thomson, B., & Williams, J. D. (2017). Deep Learning for Dialogue State Tracking: A Deep Neural Network Approach.

Kumar, N., & Dusek, O. (2024). LEEETs-Dial: Linguistic Entrainment in End-to-End Task-oriented Dialogue Systems.