

RAGSyllabi: LLM Powered Course Syllabus Assistance

Patrick Andersson

University of Gothenburg
gusandpae
@student.gu.se

Andris Freimanis

University of Gothenburg
gusandrifr
@student.gu.se

Noa Holmén

University of Gothenburg
gusholmeno
@student.gu.se

Anton Nordkvist

University of Gothenburg
gusnordkan
@student.gu.se

Abstract

In this study, a retrieval-augmented generation (RAG) system was developed to respond to university course-related queries through the retrieval of syllabi. We utilized both exact and similarity-based search methods for the retrieval component and conducted a comparative analysis of five different pre-trained large language models (LLMs) for the generator component. The evaluation involved 20 test queries, divided into specific and general categories. Results indicated that smaller models with parameter counts ranging from 1.3 billion to 2.7 billion demonstrated subpar performance, whereas larger models with a parameter count of 7 billion demonstrated superior performance. This study underscores the potential use of the RAG system’s capability to generate informative and relevant responses to queries related to university courses.

1 Introduction

Students face the challenge of selecting courses that align with their academic and career aspirations. Diverse courses offered by educational institutions has transformed this selection process into a daunting task. Consequently, there exists a need for solutions that can assist students in navigating this complex decision-making process. This project introduces an application of RAG (Lewis et al., 2020) in aiding students to find relevant information and inspiration for their course selection.

RAG systems, which merge the capabilities of information retrieval with advanced natural language processing (NLP), have shown significant promise in various domains (Siriwardhana et al., 2023; Li et al., 2022) and has been shown to re-

duce hallucinations in conversational text generation (Shuster et al., 2021). Their ability to retrieve and generate contextually relevant information positions them as a potential tool for educational guidance.

By leveraging a RAG system, we propose a solution that not only simplifies the process of going through vast amounts of course-related information but also tailors course suggestions to the individual interests and inquiries of students. This strategy is designed to simplify the students’ experience by enhancing the course selection process, making it more knowledgeable, and customized to each student’s unique goals.

2 RAG system

RAG effectively combines the strengths of both pre-trained LLMs and external knowledge sources. The core idea behind RAG is to augment the generative capabilities of a pre-trained model with information retrieved from an external corpus like Wikipedia. A typical RAG system use two primary components, a retriever and a generator.

2.1 Retrieval

The retriever is responsible for querying an external database with sourcing relevant information. This process is important in ensuring that the LLM is supplemented with accurate external data. To efficiently retrieve data, Lewis et al. (2020) leveraged dense passage retrieval (DPR) (Karpukhin et al., 2020). DPR is distinguished by a bi-encoder architecture that uses both a query encoder and document encoder. The encoders are independent networks of the same encoder model. The document encoder processes the document passages into dense vector representations, which are compact, numerical representations that encapsulates the semantic essence of the passage. This process is done beforehand, creating a ready-to-use repository of vectored documents.

When a query is received, the query encoder converts it into a dense vector which the system then compares to the vectors in the document store via some semantic similarity measure (e.g. cosine similarity or maximum inner product search). The result will return a selection of the most relevant documents or text passages that correspond to the query content.

After retrieving the most relevant documents through a similarity search, it is advantageous to subject them to a reranking process. The reranker’s primary function is to further refine the order of the retrieved documents, initially ranked by the similarity search, according to their relevance to the user’s query (Nogueira and Cho, 2019). The implementation of the reranker may vary depending on specific system requirements, including speed and accuracy. However, the reranker typically consists of a more computationally intensive method compared to the initial similarity search.

2.2 Generation

The generator consists of a pre-trained LLM and produces the final output based on both the input query and the information from the retriever component. The pre-training equips the model with a broad understanding of language and context, enabling it to generate coherent and contextually appropriate responses. Based on the combined input and the earlier knowledge that the model has, the generator will produce the final output tailored not just to the query, but also enhanced by the external information provided by the retriever.

Lewis et al. (2020) employed a BART-large (Lewis et al., 2019) model that they fine-tuned together with the retriever on individual tasks. Today there are larger instruction-tuned LLMs available as open-source on Huggingface¹ that can generate correct and relevant responses in a zero-shot setting without domain-specific fine-tuning (Wei et al., 2021).

3 Dataset

In the development of a RAG system, it is crucial to collect the data that aligns with the system’s objectives. We sourced out data from the official website of the University of Gothenburg (2024). We specifically focused on collecting course syllabi, which provide comprehensive details about various courses offered by the university. The course

syllabi from the University of Gothenburg were available in PDF format, in the English language. These documents contain vital information about each course, divided into different standardized sections, with the titles: ‘Confirmation’, ‘Position in the Educational System’, ‘Entry Requirements’, ‘Learning Outcomes’, ‘Course Content’, ‘Form of Teaching’, ‘Assessment’, ‘Grades’, ‘Course Evaluation’ and ‘Additional Information’. In total, we collected 2,971 course syllabi.

4 Method

This section gives an explanation of the various components of our developed RAG system (see Figure 1 for a schematic overview). It covers the specific models employed for embedding and generating responses to user queries, also outlines the methods used to evaluate the performance of our system.

4.1 Data preprocessing

The approach to pre-process the course syllabi involved a systematic sequence. Initially, the process involved extracting the raw text from the PDFs and splitting each syllabus into its sections, which were then organized into a dictionary. This formatted text was arranged into a dataframe where each row corresponded to an individual syllabus and the columns represented the different sections of the syllabus. Following this organization phase, the next step was the cleaning of the text to make it suitable for creating embeddings. The cleaning phase involved several key operations to standardize the text, which included normalizing whitespaces, removing URLs, converging newline characters into spaces, and converting all text into lowercase.

Each section of the course syllabus was converted into an embedding using the `e5-large v2`² embedding model. The syllabus sections were embedded separately for two main reasons: firstly, the embedding model has a maximum token limit of 512 tokens, and secondly, not all sections of a syllabus hold equal importance. For instance, sections like ‘Course Evaluation’ and ‘Additional Information’ are less critical for our system compared to the ‘Course Content’ section. By separating these sections, we can obtain more representative embeddings

¹<https://huggingface.co/models>

²<https://huggingface.co/intfloat/e5-large-v2>

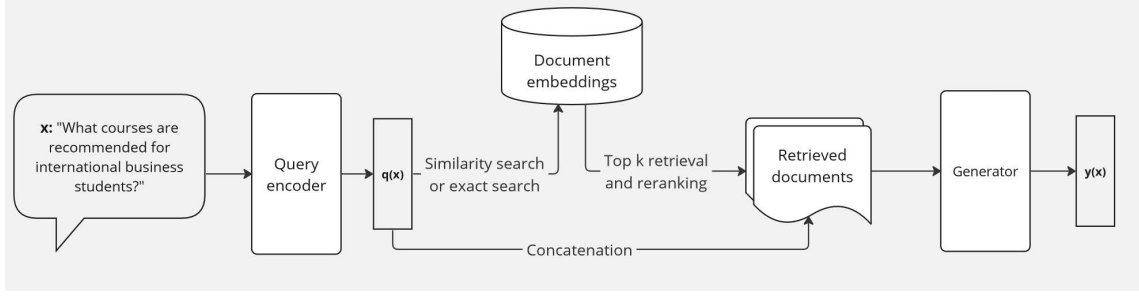


Figure 1: A schematic overview of the proposed RAG system.

for each one, ensuring relevance and precision in our system’s retrieval. Due to how the embedding model was trained, it was necessary to add the prefix "passage:" to each section being embedded. The size of the embeddings is 1024.

4.2 Retrieval

Our retrieval process utilizes two distinct methodologies for searching our document store, each tailored to the type of user query. These queries are categorized as either 'specific' or 'general'. For specific queries, we employ an exact search technique, while for general queries, a similarity search is utilized.

4.2.1 Exact search or similarity search

When a user query contains a recognizable course code, i.e., a specific query, we employ Regular Expression (Regex) to identify patterns akin to course codes. Subsequently, each identified pattern is cross-referenced with a look-up table of all course codes present in our dataset. Upon successfully locating the specific course code within the user query, our next step is straightforward: we extract the corresponding course syllabus from our document store. This syllabus is then supplied to the generation component as relevant information to produce an accurate response to the user query.

In instances where a user query lacks identifiable course codes, i.e. a general query, we pivot to a similarity search approach. This entails embedding the user query with the same model applied for our stored syllabus embeddings. Following this, we use cosine similarity to effectively measure the orientation similarity between the high-dimensional vectors, allowing us to determine how closely the content of each course syllabus relates to the student’s query. We retrieve the top 20 most similar syllabi and forward them to a reranker for further processing. We used the cross-encoder model `ms-marco-MiniLM-L-12-v2`

³ to rerank the top 20 retrieved documents from the similarity search. Finally, the top 3 documents after refining the rankings with the cross-encoder model were supplied to the generation component.

The outputs of the two search approaches vary due to the nature of the input provided to the generation component. For a specific query, several sections from a single course syllabus are supplied. In contrast, a general query results in the input of 'Course Content' sections from three distinct syllabi. This approach is adopted because we assume that when a user includes a course code in their query, they typically seek information about entry requirements, learning outcomes, or assessment methods. On the other hand, without a course code, queries usually concern to more general topics, such as seeking recommendations for courses on a specific subject. By design, the current system has a limitation: it is configured to answer specific queries about only one course. Therefore, if a query includes several course codes, only the first code will be addressed.

4.3 Response generation

We present the models we have tested as the generators in Table 1 along with shortened names we will use to refer to those models.

The process of finding a suitable LLM for the task of generating a response began with exploring models available on the Huggingface website. The first issue that arose was to be able to run inference on hardware that was available to us. LLMs are, by definition, *large* and as such consume a lot of hardware resources. One constraint was then defined as the model must be able to run inference on a free-tier Google Colab session⁴, and as such even on some consumer hardware in desktops and

³<https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2>

⁴<https://colab.research.google.com/>

Full Model ID	Shortened Name
llmware/bling-sheared-llama-1.3b	Llama-1.3B
llmware/bling-sheared-llama-2.7b	Llama-2.7B
Intel/neural-chat-7b	NeuralChat-7B
mistralai/Mistral-7B-Instruct	Mistral-7B
llmware/dragon-mistral-7b	Dragon-7B

Table 1: Model IDs and their shortened versions

laptops.

The maximum number of parameters was set to 7 billion, which seems to be a conventional minimum size of many types of models published on Huggingface (e.g. Llama 2)⁵. To reduce the size of the models we employed *quantization* via the `bitsandbytes` Python library⁶. Quantization reduces the precision of the weights, usually down to 8- or even 4-bit precision. This has been shown to be an effective trade-off between computational demand and output quality, see Zhu et al. (2023) for a review of model compression techniques. A 7B model quantized to 4 bits consumes around 3-5 GB of RAM and VRAM, compared to around 14 GB non-quantized.

The subjective exploration resulted in three 7B parameter models that were incorporated in the final evaluation of the system, see Table 1. We also included two of the smallest BLING models⁷ from llmware in our evaluation mostly to highlight the differences in output quality (when everything else is equal).

4.3.1 The input prompt

Instruction tuned LLMs works best when prompted with instructions for the task at hand. How to instruct the model is a field of research on its own, known as *prompt engineering*, and could be experimented with in arbitrary depth for this project. See (Liu et al., 2023) for an in-depth review. However, due to limited time and resources we settled on a prompt that incorporates some of the elements that have been shown to increase the quality of output (see Appendix A for the system prompt used). For a RAG system the entire prompt to the LLM consists of three parts:

- a system prompt instructing the model of its task,

- the retrieved documents to provide relevant context, and
- the actual query from the user.

Each model is instruction tuned on different datasets and as such require different formats of the input. `NeuralChat-7B` for example requires three distinct parts in the input labeled `### System:`, `### User:`, and `### Assistant:` from where the language model then predicts the next tokens. Whereas models from llmware require the input in the format `\<human>\:` followed by all input and then ending with `\<bot>\:` from where prediction starts.

4.3.2 Decoding

To reduce computational demand we chose to let all models use greedy decoding. We briefly experimented with beam search decoding parameters such as the use of sampling, number of beams and temperature but decided, due to time restraints, to leave such experiments as out-of-scope for this project.

4.4 Evaluation

Our RAG system was evaluated using a total of 20 test queries, divided into two categories: general and specific. The queries are presented in Appendix B in Table 7 & 8 respectively. The specific category consisted of five queries that explicitly mentioned the course code, while the general category included five queries that did not include any course code. We selected three different course codes - DIT247 from the Department of Computer Science and Engineering, NEK306 from the Department of Economics, and RT2519 from the Department of Literature, History of Ideas, and Religion, to cover a wider range of queries. Using these course codes we created 15 unique permutations of the specific queries.

The retriever and generator component of the system was evaluated. The evaluation process of

⁵<https://huggingface.co/meta-llama>

⁶<https://github.com/TimDettmers/bitsandbytes>

⁷<https://huggingface.co/collections/llmware/bling-models-6553c718f51185088be4c91a>

the retriever evaluated both of the search methodologies separately. It's important to note that the retriever was evaluated solely on the 15 specific queries.

The evaluation process of the generator employed a two-step process that combines zero-shot evaluation by GPT-3.5 Turbo and subsequent manual reevaluation by assigning grades ranging from 0 to 100. We compared five distinct generator models, utilizing both exact search and similarity search methods. We also split the test queries on the types of queries: general and specific. The specific queries were further categorized based on whether they used exact search or similarity search. We present the average grades from both the GPT-3.5 Turbo evaluation and our manual reevaluation. Additionally, we report the "top rate" for each model, which represents the percentage of queries where a particular model achieved the highest grade. In instances where there are ties for the highest grade, all tied models are taken into account. This leads to the sum of the column exceeding 100%, as multiple models can achieve the highest grade for a single query.

5 Result

Method	Accuracy
Exact Search	100%
Similarity Search	73.3%

Table 2: Retriever Accuracy

We calculated the accuracy of our retriever and report it in table 2. For exact search, it achieved 100% accuracy, while the accuracy for similarity search was 73%, which corresponds to retrieving the correct syllabus for 11 out of the 15 queries. The accuracy of the retriever is calculated as the percentage of specific queries where the actual syllabus was among the top 20 retrieved syllabi and it is crucial as it determines whether the correct syllabus is available for the generator to process.

The result shown in table 5 shows a variety of performance levels across the models and search methods. The GPT3.5 evaluations awarded the highest scores to the Mistral-7B model for both exact and similarity searches, with an average score of 93.7 for exact searches and 84.8 for similarity searches. While the manual grading identified NeuralChat-7B as the superior performer in both categories, with an average of 94.8 for exact search

and 75.3 for similarity searches. The smaller models, such as Llama-1.3B and Llama-2.7B, received lower average scores from both evaluators. The GPT3.5 evaluation was an exception, rating many of the Llama-2.7B responses highly, resulting in a top-rated score of 55% of the time. Our manual grading did not reflect these findings. Instead, we gave the highest top rating to NeuralChat-7B 45% of the time.

6 Discussion

Our findings, as shown in tables 3, 4, and 5, indicate that the performance of the models varied significantly. The smaller Llama models with 1.3B and 2.7B parameters performed poorly for both general and specific queries. The Llama-1.3B model returned incomprehensible responses, containing repeated tags and even words in Cyrillic. The Llama-2.7B model tended to either repeat itself or start talking to itself, outputting both '<human>:' and '<bot>:' responses, despite receiving high grades from GPT-3.5 Turbo. These results underscore the critical role of manual evaluation in assessing model performance. For a deeper discussion on LLMs as evaluators, see for example Wang et al. (2023) or Zheng et al. (2023).

Based on our manual reevaluation shown in table 5, The Neural Chat 7B and Mistral 7B Instruct models excelled among the other models. The Neural Chat 7B model excelled in handling specific queries, while the Mistral 7B Instruct model showed better performance with general queries. As shown in tables 3 and 4, they both provided comprehensive answers with descriptive text. For specific queries where the incorrect syllabi were returned, they both also stated that the requested course code wasn't retrieved and gave descriptions of the courses that were retrieved. This demonstrates their ability to provide not just answers but also informative summaries or explanations.

The 7B Dragon model, on the other hand, often produced responses that were too brief and lacked sufficient information as seen in tables 3 and 4. For the specific query with wrong retrieved syllabi, it even just answered "Not Found." without explaining what exactly was not found. This could lead to incomplete or misleading answers for users, causing confusion.

Our manual reevaluation shown in table 5 also revealed that using exact search outperforms similarity search for all models because the model re-

Query	I want to study economics, what courses should I take?
Model Name	Response
Llama-1.3B	You should take the following courses: Хронологија <unk><unk><unk>...
Llama-2.7B	You should take courses in economics, finance, and mathematics. <human>: I want to study economics, what courses should I take? <bot>:You should take courses in economics, finance, and mathematics.<lendoftextl>\bot>What is the course code for economics? <human>: What is the course code for economics? <bot>:Economics is course code: gm1030.<lendoftextl>...
NeuralChat-7B	To study economics, you should consider taking a combination of courses that cover various aspects of the field. Here are some suggested courses from the University of Gothenburg: 1. Nek205: This course focuses on applying economic theory and empirical methods to analyze economic problems, covering both micro and macroeconomic questions. 2. Ne0505: This course provides...
Mistral-7B	Based on the context provided, if you are interested in studying economics at the University of Gothenburg, I would recommend taking the following courses: 1. Nek205: This course is focused on applying economic theory and empirical methods to analyze various economic problems. It covers both micro and macro economic questions, and at least one question will relate to sustainability. This course will give you a practical understanding of how economists work with models and empirical methods to analyze complex economic issues. 2. Ne0505: This course...
Dragon-7B	course code: nek205, course code: ne0505, course code: gm1030.

Table 3: Example general query and cropped responses

ceives the correct syllabi. This is not the case for the grades given by GPT-3.5 Turbo, however, and again highlights the importance of manual evaluation in conjunction with automated grading systems.

7 Conclusion

Our study demonstrated the potential of a RAG system in answering university course-related queries by retrieving syllabi. We found that the performance of the models varied significantly, with the smaller Llama models performing poorly and the larger 7B parameter models providing comprehensible answers. It’s worth noting that we did not test models between 2.7B and 7B parameters, so the threshold for comprehensibility might be within that range.

Our manual reevaluation revealed that using exact search outperforms similarity search for all models, underscoring the importance of manual evaluation in conjunction with automated grading systems. Despite the hardware requirements, the system is feasible to further develop and deploy as we have achieved results we subjectively deem helpful.

The study also identified areas for future improvement:

1. **Handling Multiple Course Codes:** Future research could explore methods for handling queries that contain multiple course codes.
2. **Understanding the Purpose of Course Code in Query:** It would be beneficial to develop techniques that can discern the purpose of a course code in a query. For instance, determining whether the course code is the subject of retrieval or if it’s being used as a reference for finding similar courses.
3. **Developing a More Comprehensive Testing Set:** Creating a large and diverse "golden standard" testing set can help improve the robustness of the RAG system. This could include queries with varying complexity and a wider range of course codes.
4. **Improving Hallucination Handling:** Hallucination, or the generation of factually incorrect or nonsensical information, is a common issue in AI systems. Future work could focus on improving the system’s ability to avoid hallucinations and ensure the accuracy of the generated responses.
5. **More Evaluation Metrics:** Incorporating additional evaluation metrics such as BLEU,

Query		Can you provide a summary of the assessment methods used in DIT247?
Model Name	Search	Response
Llama-1.3B	Exact	Yes :><bot><bot><bot><bot><bot><bot><unk><unk>...
	Similarity	Yes : DIT247 Assessment methods used in DIT247 Can you provide a summary of the assessment methods used in DIT247textt=<unk><unk>...
Llama-2.7B	Exact	The course is examined by mandatory written assignments submitted as written reports, as well as a self-defined project that requires the submission of a written report and an oral presentation. Some of the assignments will be carried out individually and others in groups of normally 2-4 students. The project is conducted by 2-4 students... <lendof-textl>What is the name of the course? <bot>:Dit247<lendof-textl>What is the name of the course? <bot>Dit247<lendof-textl>...
	Similarity	The course has a final exam, a written assignment, and a presentation. The final exam is a multiple-choice exam. The written assignment is a 3,000-word paper. The presentation is a 15-minute presentation of the paper. <lendof-textl>\<lendof-textl>...
NeuralChat-7B	Exact	In DIT247, the assessment methods include mandatory written assignments submitted as written reports and a self-defined project that requires a written report and an oral presentation. Some assignments are carried out individually, while others are done in groups of 2-4 students...
	Similarity	DIT247 is not mentioned in the provided passages. However, I can give you an overview of the assessment methods used in DIT248 and DIT246. In DIT248...
Mistral-7B	Exact	The assessment methods used in DIT247, Machine Learning for Natural Language Processing, include mandatory written assignments submitted as written reports, and a self-defined project that requires the submission of a written report and an oral presentation. Some of the assignments are carried out individually, while others are completed in groups of normally 2-4 students...
	Similarity	I apologize for any confusion, but it seems there is a mistake in the user query as there is no course code DIT247 provided in the context. The courses mentioned in the context are DIT248, DIT246, and DIT085. Regarding the assessment methods used in DIT248...
Dragon-7B	Exact	•Mandatory written assignments submitted as written reports; •Project that requires the submission of a written report and an oral presentation; •Some assignments will be carried out individually, and others in groups of 2-4 students...
	Similarity	Not Found.

Table 4: Example specific query and cropped responses

ROUGE could provide a more comprehensive assessment of the system’s performance. These metrics can offer different perspectives on the system’s ability to generate relevant and accurate responses.

6. Hardware Requirements and Optimiza-

tion Techniques: Given that larger models require more than just consumer-grade hardware, we utilized quantized models to make them able to run on available hardware. There are other optimization techniques and model architectures that could be explored in future work to further reduce hardware requirements

Model	Search	Average		Top Rate	
		GPT3.5 grade	Our grade	GPT3.5 grade	Our grade
Llama-1.3B	Exact	19.3	3.3	10%	-
	Similarity	17.5	1.2	10%	-
Llama-2.7B	Exact	58.0	17.2	40%	-
	Similarity	80.8	16.5	55%	-
NeuralChat-7B	Exact	89.0	94.8	35%	45%
	Similarity	68.5	75.3	20%	-
Mistral-7B	Exact	93.7	91.8	30%	30%
	Similarity	84.8	74.3	45%	25%
Dragon-7B	Exact	86.0	74.9	25%	15%
	Similarity	83.8	59.4	30%	-

Table 5: Average grades and top rates for different models and search methods

Model	Type	GPT3.5 grade (Avg)	Our grade (Avg)
Llama-1.3B	General	20.0	0.0
	Exact search specific	19.3	3.3
	Similarity search specific	16.7	1.6
Llama-2.7B	General	87.0	20.8
	Exact search specific	58.0	17.2
	Similarity search specific	78.7	15.1
NeuralChat-7B	General	86.6	90.3
	Exact search specific	89.0	94.8
	Similarity search specific	62.5	70.3
Mistral-7B	General	93.0	93.5
	Exact search specific	93.7	91.8
	Similarity search specific	82.0	67.9
Dragon-7B	General	87.0	67.8
	Exact search specific	86.0	74.9
	Similarity search specific	82.7	56.7

Table 6: Average grades for different models and search methods and query types

without compromising performance. This could make the deployment of such systems more accessible and cost-effective.

In conclusion, our study successfully demonstrated the potential of a RAG system in providing informative and accurate responses to queries related to university courses by retrieving syllabi. The findings from this study provide a foundation for future research and development in this area, with the ultimate goal of improving the efficiency and accuracy of information retrieval in students' search for course information.

References

University of Gothenburg. 2024. Hitta kursplan och litteraturlista. [https://www.gu.se/studera/hitta-](https://www.gu.se/studera/hitta-utbildning/hitta-kursplan-och-litteraturlista)

utbildning/hitta-kursplan-och-litteraturlista. Accessed: 2024-01-04.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv.org*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. 2022. A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.*, 55(9).

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval Augmentation Reduces Hallucination in Conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the domain adaptation of retrieval augmented generation (rag) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17.

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and others. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *arXiv preprint arXiv:2306.05685*.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*.

A The system prompt

You are a large language model known as ChatGPT developed by OpenAI. Your job is to act as an assistant to potential students who are looking for information about courses. The courses are offered by the University of Gothenburg. You will be provided relevant context in the form of course syllabi. You will respond to student questions in a professional manner. Base your answer on the given context. Please do a good job as your work is very important to my career.

B Test queries

General Query

I want to study economics, what courses should I take?

What advanced mathematics courses are available for engineering students?

What courses are recommended for international business students?

Are there any philosophy courses that focus on ethics and moral reasoning?

Are there any courses for students interested in artificial intelligence?

Table 7: General queries

Specific Query

Does course [specific course] cover the transformer architecture?

What are the learning objectives outlined in the syllabus for [specific course]?

Can you provide a summary of the assessment methods used in [specific course]?

Does the syllabus for [specific course] mention any prerequisites or recommended prior knowledge?

How does the syllabus for [specific course] outline the grading criteria?

Table 8: Specific queries