FRI Natural language processing course 2024
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

WIP: Literary Conversational Agents

Žiga Trček, Matej Urbas, and Jan Vasiljević

Abstract

DRAFT: Engaging audiences deeply with literature is crucial for enhancing global literacy levels. In this study, we build upon the foundation of conversational agents, which have historically underperformed but have recently been revolutionized by advancements in large-scale language models. We fine-tune various foundational models and assess their enhanced capabilities using a series of standardized quizzes that we introduce. Our focus is on popular literary series, specifically ”A Song of Ice and Fire” and ”Harry Potter.” Additionally, we underscore the significance of employing In-Context Learning to effectively emulate the dialogue styles of key characters within these series, thereby enriching the interactive reading experience.

Keywords
DRAFT, conversational agents, large language models, in-context-learning

Advisors: Slavko Žitnik

Introduction

Literacy among young people is declining, as highlighted in [1]. Many young people have a disinterest in reading and seldom read for enjoyment. A potential strategy to encourage reading is to involve them in conversational interactions with digital pedagogical agents that imitate well-known literary figures. Although numerous studies, such as those [2, 3], discuss the advantages of pedagogical agents, detailed technical implementation aspects are often overlooked. Our work aims to explore various methods for creating pedagogical agents and to provide a comprehensive technical implementation for the method we choose.

Related work

In [4] they built a social bot, called Alana, which is able to engage in an open-domain conversation with their users over various popular topics. The key requirements for such a bot are to maintain the context, provide coherent responses, and be engaging and knowledgeable. The final bot is an ensemble of many different bots, each of which has a different purpose. A ranker is used to determine the best bot response for the given user input. For context maintaining, a state object with information from previous conversations is stored and accessible to every bot in the ensemble.

Retrieval augmented generation (RAG) is often used to correct factually inaccurate, outdated or halucinated Large Language Model (LLM) outputs. A survey on different RAG methods is conducted in [5]. RAGs can be seprated into three categories: pre-training, fine-tunning and inference. Nowdays inference RAGs are mostly used. [6] proposed FLARE, Forward-Looking Active REtrieval Augmented Generation, which re-prompts the language model with extra retrieved data about the subject when some tokens in LLM's output have a low probability.

A straightforward way to create agents is by training or fine-tuning LLM. In [7] the authors developed conversational agents that resemble historical figures like Beethoven, Cleopatra, and Caesar, with personalized profiles, experiences, and emotional states. They introduced three new methods for training specialized agents. Experience Reconstruction extracts scenes in the style of memory flashes, such as profiles, scenes, or interactions. Protective Experience aims at teaching the model to forget or ignore information not relevant to the character to prevent knowledge hallucinations. Experience Upload uses the previous two techniques to fine-tune an existing LLM. They fine-tuned a LLaMa 7B model [8] on a dataset of 750 000 words per character using eight A100 80GB GPUs for 1 hour per character. To assess the models, they used an interview process.

Obtaining data to train a character can be difficult. [9] proposes a novel data augmentation approach named PEDANT that helps train models that mimic human personality by generating large amounts of data with a GPT combined with domain expertise. The method first gathers unlabeled data from online resources and trains a generative language model with it. Then, this model is prompted with seed sentences that an expert created and is asked to complete them. Then, these completions are filtered and ranked based on an expert-defined scoring function. In the paper, PEDANT was implemented on an anti-social psychopatic personality disorder. A labeled corpus with this disorder does not exist, so this is a good showcase of the usefulness of the approach. The data to train the GPT comes from cinema, TV, and Reddit. The model was validated using a text classification task. They used the generated data to train a classifier and tested it on offensive-speech datasets. The results were very encouraging, but requires domain knowledge, which can be a big limiting factor and bottleneck in a larger process.

To avoid training a LLM, [10] suggests using prompt engineering, specifically Chain-of-Thought (COT), on an existing model to incorporate more contextual information. They recommend employing Information-Rich Prompts (IRP) that include the emotional state, the character's relationship with the interlocutor, and the character's memories. Memories are categorized into short-term, which are a limited number of the most recent conversations with the interlocutor, and long-term, which are recursively summarized memories of longer conversations from the character's perspective. Although not explicitly stated, implementing the Big Five personality model [11] could further refine the character's responses. This model would detail the character's Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism.

Previous methods that do not involve fine-tuning could be enhanced by using the OpenICL framework [12]. In-Context Learning (ICL) is an approach used with LLMs where the model learns a specific task without the need to update its weights. Instead, the model is shown examples of how the task should be performed. OpenICL offers the tools needed to construct ICL tasks, including key components like retrieval strategies and inference methods. For retrieval, it incorporates heuristic-based methods (such as BM25 and Top-K), random sampling, and model-based retrieval (using embeddings, RAG, Minimum Description Length (MDL), and entropy-based selection). For inference, OpenICL facilitates the integration of COT and other methods along with a prompt template.

It is important to consider teaching strategies while implementing an agent that serves an educational purpose. [13] carried out a detailed analysis of reading comprehension textbooks from the Netherlands, which is one of the nations with a low comprehensive literacy. The researchers analysed lessons withing the textbooks and then also analysed the utilisation of these textbooks by teachers, both by conducting interviews with teachers and attending live lessons. They found that the lessons are mostly focused on exercising and that there is no strong alignment between goals of the lessons, the theory behind them and the assignments that the students must carry out. Little actual knowledge about reading strategies was illustrated and there was no opportunity to choose and apply strategies yourself. The interviews showed that the teachers were aware of these problems, but there were very few who adapted the lessons to counteract them and improve the quality of their teaching. The knowledge that was observed in the textbooks was divided into:

- declarative knowledge - knowing something
- procedural knowledge - knowing how to do something
- conditional knowledge - knowing when to do something.

The textbooks were mostly just focused on the procedural part of the knowledge. To improve literacy, all three should be taught.

[14] describes the importance of setting and situational continuity while reading, which can have major implications in providing a good user experience. Three experiments were carried out on 27 psychology students that tested which aspects of a five-dimensional situational model are more important to our experience. They tested the impact of different aspects by measuring reading time while introducing discontinuities across different dimensions (time, space, causation, motivation, protagonist). The reading time increase is very noticeable in all but the spatial dimension. There, spatial discontinuities did not present a large increase in reading time unless the study participants memorized the map of the story space in advance. The study confirmed the "processing-load hypothesis" that predicts that the reading time goes up when there is more data to process. It's very likely that this information could be taken into account when constructing a model used for learning by keeping continuities along dimensions that are irrelevant for the learning experience and channeling the focus elsewhere.

BookNLP [15] is an NLP pipeline that supports the analysis of literary texts. It performs Part-of-Speech (POS) tagging, dependency parsing, entity recognition with co-reference resolution and clustering, event tagging, and more. Built on top of SpaCy [16], it uses BERT [17] for co-reference resolution. BookNLP provided an effective pipeline for extracting dialogues from the books and attributing them to characters.

Methods

Our main approach to creating conversational agents revolves around data extraction from literary works. Based on this data, we provide the models with additional context to improve their performance. We tried several methods throughout the entire pipeline. However, based on numerous criteria, we settled on dialogue extraction, tagging characters from books, summarization, retrieval-augmented generation, and in-context learning. We will describe each of these steps in more detail.

As our source material, we chose two popular book series: *A Song of Ice and Fire* (ASoIaF) [18] by George R. R. Martin and *Harry Potter* (HP) by J. K. Rowling [19]. ASoIaF comprises 5 books totaling 1 778 216 words, while HP consists of 7 books and 1 087 549 words. The two main reasons for this choice were the length of the books and the team's familiarity with the material. The books also differ significantly in style, providing a good test for our approaches.

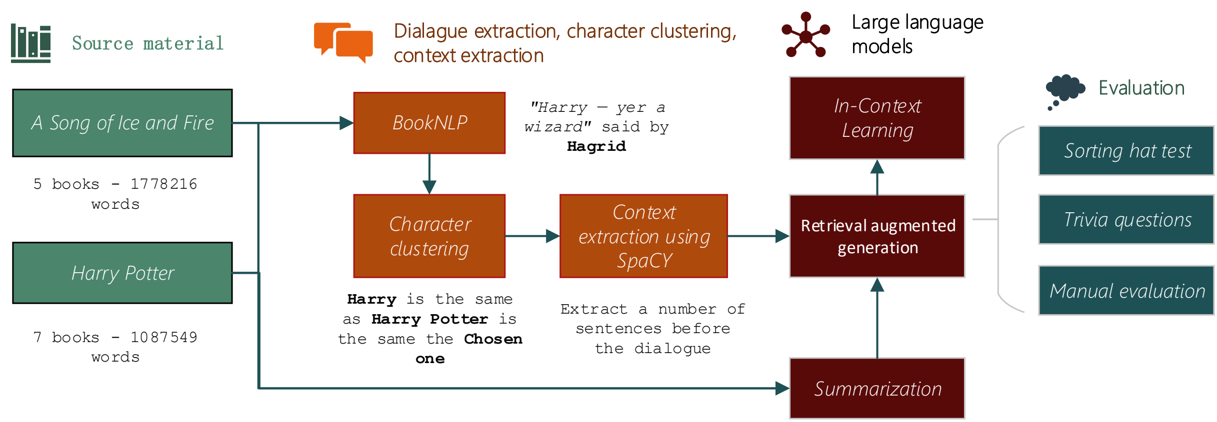

Figure 1. Our proposed pipeline. High-level overview of our pipeline and methods used in the final project.

Dialogue Extraction Using Instruct LLM

We extracted all the dialogue along with the pre and post-context (10 sentences before and 2 sentences after each dialogue) and used Phi3 and Llama8B to classify the dialogue by identifying the speaker. We ignored dialogues shorter than 16 characters (as they were not meaningful) and longer than 500 characters (to save VRAM consumption). With a batch size of 10 dialogues, we classified all 40 318 dialogues in 7 hours. We used a 2 and 4-shot prompt with examples of classification but did not achieve good results. There were three main reasons for this:

1. The models did not possess good enough reasoning capabilities to classify the dialogues.
2. Co-reference resolution was not adequate to classify the dialogues. When pronouns were used, the model did not know who was speaking most of the time.
3. Information leakage: The models had some prior knowledge about the books, as they sometimes classified the dialogue with characters who were not even present in the pre or post-context or the dialogue itself.

By validating the data by hand, we realized we would not achieve good enough results with this approach.

Dialogue Extraction Using BookNLP and Clustering

We used BookNLP to extract dialogues from both books. The `big` model provided by the authors of BookNLP was utilized, which required 5-8 minutes of processing time per book. We also attempted to merge the books before processing to improve co-reference clustering and resolution; however, this resulted in memory segmentation faults, even on a machine with 128GB of RAM (Arnes). Consequently, we processed each book individually, necessitating the correct correlation (clustering) of character names across the series. This process involved normalizing names and removing duplicates by extracting sub-tokens and taking the root with the highest occurrence. If two sub-tokens had the same occurrence, we joined them, indicating a name with a space in it. For example, the character *Hot Pie* from the series ASoIaF.

To validate the results, we manually reviewed randomly sampled dialogues. Based on this assessment, we assessed that approximately 90% of the dialogues were correctly classified. Additionally, we constructed two graphs that show the dialogue frequency by character per book in the series (fig. 2 and fig. 3). Based on our familiarity with the books, we can confirm that the results are accurate. In the ASoIaF series, the chapters are also told from the perspective of the characters, so we matched the frequencies with their respective chapters. In total, we gathered 36 946 dialogues from ASoIaF and 32 541 from HP, totaling 69 487 dialogues.

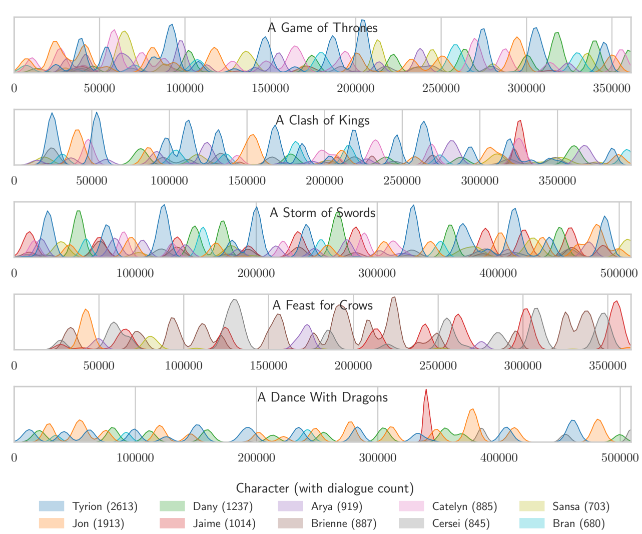

Figure 2. Dialogue from 10 most frequent characters in A Song of Ice and Fire. This shows the dialogue frequency by character per book in the series. The x-axis represents the token count when the character speaks, while the y-axis is the kernel density estimate of the dialogue frequency.

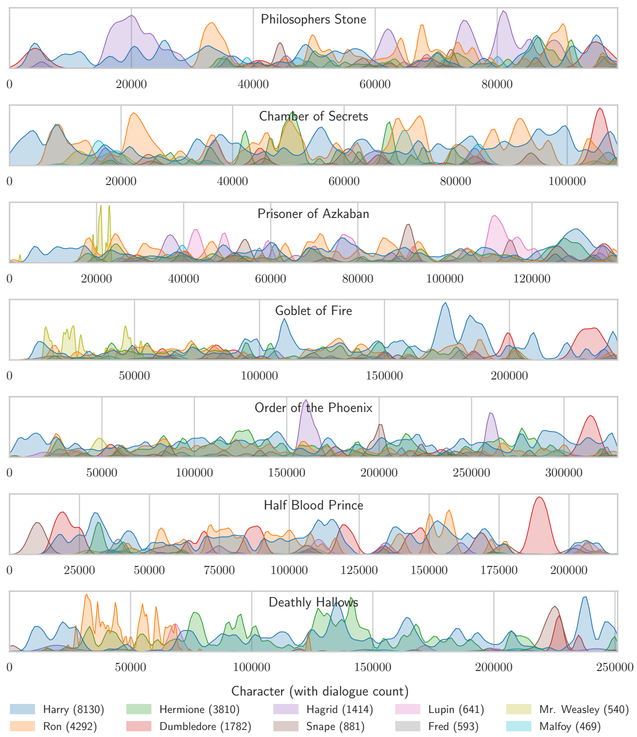

Figure 3. Dialogue from 10 most frequent characters in the Harry Potter series. This shows the dialogue frequency by character per book in the series. The x-axis represents the token count when the character speaks, while the y-axis is the kernel density estimate of the dialogue frequency.

Context extraction

Before each dialogue, we extracted 3 chunks of 3 sentences each. We used the *SpaCy* sentence tokenizer to split the text into sentences. With the constructed dataset, we gained the ability to selectively choose how much context to provide to the large language model (LLM) down the pipeline.

Other unsuscsessful attempts

After extracting dialogues, we attempted several other methods to extract data from the books. The goal was to enhance the conversational model by providing it with more context from the books. These methods included:

1. Extracting factual information from the books by reusing named entity recognition (NER) entities related to the characters. We extracted subject-verb-object triples from the books to gain more information about the characters. However, due to the complexity of the language used in the books, the extraction did not yield meaningful results.

2. Recursively summarizing the character behaviors to achieve a better understanding of their personalities and relationships. We used Phi3 with a 128k context window to recursively summarize sections of the books that included a particular character of interest. This approach, however, did not succeed. It was computationally expensive (even after splitting into 20k chunks, the model used upwards of 60GB of graphics memory) and the results were inadequate. Despite claims that the model can handle tasks within a long context, the results showed that the model completely forgot the instructions after utilizing only one-eighth of its theoretical context window.

3. Using DistilBART for question answering. Our final attempt was to extract key information about characters from the books (such as character locations, ages, etc.) and use DistilBART to answer questions. This was intended to serve as in-context learning for our conversational agents. However, this approach also failed due to the complex language used in the books.

4. While not entirely related to dialogue extraction, our first attempt at the problem included fine-tuning Phi 3 and Llama 8B on the *Harry Potter* series. We used a chunk size of 512 characters with an overlap of 64 characters. The models were trained on overlapping chunks of text to ensure that they learned the context of the conversation. However, it was evident by the end of the training that the models did not learn much. The source material was already included in the pretraining data of the models, so the fine-tuning did not provide any significant improvements.

Book summaries

The characters' dialogues can give the language model an idea of how a specific character speaks; however, it can still use more context to formulate better answers. Furthermore, much of the important contextual information cannot be extracted from speech alone. By providing the language model with extra content from the books, we hoped to increase its performance in some evaluation tasks.

Original texts are quite long. ASoIaF consists of around 1.7 million words, while the HP series is a bit shorter, with 1.1 million words. Therefore, we split the books into smaller chunks and summarized each of them. We tried a few different summarization language models from HuggingFace. These included `bart-large-cnn` [20], `google-t5/t5-large` [21], `google/pegasus-xsum` [22] and `Falconsai/text_summarization`.

By examining the outputs, we concluded that Falconsai's model provided the best summarizations. It has a context window of 512 tokens; therefore, we split the books into chunks smaller than 512 tokens. Splitting was done only between sentences, so they were never cut in half, which resulted in chunks being smaller than 512 tokens.

This gave us 6 275 chunks from ASoIaF and 4 716 chunks from the HP series. Summarizing all 10 991 chunks took about 6 hours and resulted in approximately a six-fold reduction in word count. The summarized ASoIaF consists of around 300k words and the summarized HP series consists of around 200k words.

Retrieval-Augmented Generation

We stored the selected characters' dialogues, with or without surrounding context, in a FAISS vector database. During inference, the database was queried by the user's question and returned between 10 and 30 promising character lines, which were then used to create the model prompt. The same process was applied to book summaries to provide the model with additional context.

In-Context Learning

The closest retrieved dialogues, summaries, and other contexts were included in the prompt to provide the model with more context. This was done to improve the model's performance on the quiz questions and to make it more engaging in dialogue.

Results

We evaluated our fine-tuned models on a dataset of 108 multiple choice quiz questions about the Harry Potter series. The questions were designed to test the models' understanding of the characters and their relationships. The models achieved an average accuracy of 72% before and after fine-tuning.

We also evaluated the models with a sorting hat quiz, where the model had to solve a quiz as a character from a given house (Gryffindor, Hufflepuff, Ravenclaw, or Slytherin). The models were always classified into Ravenclaw.

We plan to evaluate the models on a dataset of quiz questions about the A Song of Ice and Fire series.

We will also evaluate our models on state of the art benchmarks, as in [23]. Use the results section to present the final results of your work. Present the results in a objective and scientific fashion. Use visualisations to convey your results in a clear and efficient manner. When comparing results between various techniques use appropriate statistical methodology.

Here starts the "final" evaluation

Most of the evaluation was manual, therefore slightly subjective. It was done independently by all three team members and averaged into the final conclusion.

Character evaluation

We looked at the language model's ability to speak as a selected character. For this test we picked 6 characters, 3 from both series. From the Harry Potter franchise we picked Harry Potter himself, headmaster Dumbledore and antagonist Voldemort. For A Song of Ice and Fire we picked mother of dragons Daenerys, lord commander John Snow and the best door holder Hodor.

We created a list of questions for every character. There are 8 non-specific questions (for all 6 characters), 9 Harry Potter specific questions, 14 ASOIF specific questions and for each character an additional 4-8 questions. In total there are 30 different questions for Harry Potter and 40 questions for ASOIF characters.

The biggest evaluation problem was quite unexpected. It was very difficult to find an appropriate large language model for this task. Bigger language models, such as Llama-3-8B [24] or Mistral-7B [25] were already so familiarized with both series that the model worked good without any modifications. Smaller language models, such as TinyLlama-1.1B [26], GPT2 [27] or Google Gemma-2B [28], were either unable to act as a given character (saying that they are a langauge model and unable to answer the question) or their answers were very random, and not connected to the books in any way, despite trying several different prompts and RAG strategies. In the end we decided to use Microsoft's Phi-3 [29] and Phi-2 models, pretrained for instruction tasks. These two models weren't as familiar with the characters but were still able to somewhat act as the desired character.

For each character we tested three different prompting strategies:

- Prompt contained only the name of the character and the book series and some instructions how to act. This was done so that we can get a rough baseline of what the model knows and how it behaves out of the box.
- RAG reveal: This prompt extended the previous one with dialogues and other context retrieved from the vector database.
- RAG hidden: This prompt didn't state the character name or the book series. Only data that the model recieved was retireved from the vector database.

Each of the two models answered 150 questions across all characters using all three prompting strategies, which resulted in 900 total answers. For every member in the group we randomly sampled 60 answers from Harry Potter and 60 from A Song of Ice and Fire. Half of the questions were answered by Phi-2 and the other half with Phi-3. Answers were randomly mixed, so that we didn't know which prompting strategy resulted in which answer. We then ranked the answers from best to worst based on the overall structure, answer quality and information correctness. In total we checked 360 questions with 1080 answers. For each prompting strategy we summed its final ranks and sorted them accordingly. Final rankings are shown in Figure 4.

The best-performing model is Phi-3 with the RAG reveal strategy. Other models and strategies are, however, more interesting to analyze. If we exclude the best-performing model, we can observe that the strategies behave differently based on the chosen book series. Performance of bots that emulate ASoIaF's characters increases when we first add RAG and then increases again when we remove the character name and series from the prompt. However, with Harry Potter's characters, we can see a slight decrease in performance when adding RAG and then another decrease when removing the name and series information. There is a chance that this happened

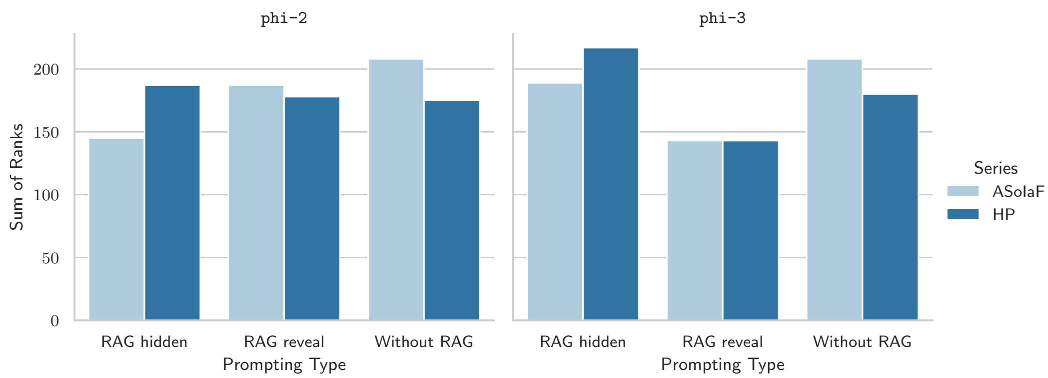

Figure 4. Evaluation: Sum of ranks for both models and all three prompting strategies. Lower is better.

because of our manual, subjective evaluation. However, when we examined the dialogues from both series, we found that the dialogues in ASoIaF are much longer and contain more useful information. Dialogues from Harry Potter are shorter and often lack meaningful information.

Sorting hat test

We thought it would be interesting to evaluate our Harry Potter characters with the sorting hat test. This evaluation was done out of curiosity. The rules and questions of the sorting test are not really stated inside the books. We scraped a random online sorting hat quiz which looked promising and easy to use. We used the quiz on 8 different characters, 2 from each house:

- Gryffindor: Harry and Dumbledore,
- Slytherin: Snape and Draco,
- Ravenclaw: Cho and Luna,
- Hufflepuff: Cedric and Tonks.

Each character was tested with all three prompting strategies, with Phi-3 as the language model. We ran the test 10 times for every character. The test results are shown in Table 1. Correct house predictions are in bold, number in parantheses is the number of times this house was selected among all 10 tries.

Table 1. Sorting hat results

Character	Without RAG	RAG hidden	RAG reveal
Harry	Raven (5)	**Gryff (9)**	**Gryff (6)**
Dumbledore	Raven (7)	**Gryff (7)**	Raven (7)
Snape	**Slyth (10)**	Gryff (7)	Gryff (7)
Draco	**Slyth (10)**	Gryff (7)	**Slyth (10)**
Cho	**Raven (7)**	Gryff (6)	**Raven (9)**
Luna	**Raven (6)**	Gryff (9)	**Raven (8)**
Cedric	Slyth (10)	Raven (5)	Raven (6)
Tonks	Gryff (9)	Gryff (5)	Gryff (10)

As expected, this test didn't bring any meaningful results. When the character is hidden from the model, it almost always predicts Gryffindor, which makes the correct guess for Harry and Dumbledore meaningless. When character is revealed to the model, it correctly sorts half of the characters. RAG doesn't really effect the outcome of the sort. We can therefore assume, that the quotes and context from RAG are not really useful for this test. We tried to run the test on Llama-3-8B, which should have deeper understanding of characters out of the box to see how it performs. Results are shown in Table 2.

Table 2. LLama-3 sorting hat results

Character	Without RAG	RAG hidden	RAG reveal
Harry	**Gryff**	**Gryff**	**Gryff**
Dumbledore	Raven	Raven	Raven
Snape	Raven	Gryff	Gryff
Draco	Raven	Gryff	Gryff
Cho	Gryff	**Raven**	**Raven**
Luna	**Raven**	**Raven**	**Raven**
Cedric	Raven	Gryf	Raven
Tonks	Gryff	Raven	Raven

These are even worse than with Phi-3. Here it only sorted the characters into Gryffindor and Ravenclaw. We can conclude that this test is flawed in several ways: sorting questions are not stated in the books, we can't really verify the validty of the used test and questions in the quiz are often of philosophical/abstract nature which is not really interpretable by large langauge models.

Discussion

Our fine-tuning attempts have not been successful so far. The baseline models perform well on the quiz questions and simple fine-tuning does not seem to improve the performance. We will try to extract more context from the books using more sophisticated approaches and use it to fine-tune the models.

The dialogue generation is also very good out of the box, especially with smart prompting. We will try to improve the dialogue generation by using context databases.

After the 1st phase

Use the Discussion section to objectively evaluate your work, do not just put praise on everything you did, be critical and exposes flaws and weaknesses of your solution. You can also explain what you would do differently if you would be able to start again and what upgrades could be done on the project in the future.

Acknowledgments

After the 1st phase

Here you can thank other persons (advisors, colleagues ...) that contributed to the successful completion of your project.

References

[1] Jane Murray. Literacy is inadequate: young children need literacies, 2021.

[2] Thijs MJ Nielen, Glenn G Smith, Maria T Sikkema-de Jong, Jack Drobisz, Bill van Horne, and Adriana G Bus. Digital guidance for susceptible readers: Effects on fifth graders' reading motivation and incidental vocabulary learning. *Journal of Educational Computing Research*, 56(1):48–73, 2018.

[3] Mehdi Alaimi, Edith Law, Kevin Daniel Pantasdo, Pierre-Yves Oudeyer, and Hélène Sauzeon. Pedagogical agents for fostering question-asking skills in children. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[4] Ioannis Papaioannou et al. *Designing coherent and engaging open-domain conversational AI systems*. PhD thesis, Heriot-Watt University, 2022.

[5] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[6] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

[7] Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. Character-llm: A trainable agent for role-playing, 2023.

[8] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[9] Yair Neuman, Vladyslav Kozhukhov, and Dan Vilenchik. Data augmentation for modeling human personality: The dexter machine, 2023.

[10] Seokhoon Jeong and Assentay Makhmud. Chatbot is not all you need: Information-rich prompting for more realistic responses, 2023.

[11] L R Goldberg. An alternative "description of personality": the big-five factor structure. *J Pers Soc Psychol*, 59(6):1216–1229, December 1990.

[12] Zhenyu Wu, YaoXiang Wang, Jiacheng Ye, Jiangtao Feng, Jingjing Xu, Yu Qiao, and Zhiyong Wu. Openicl: An open-source framework for in-context learning, 2023.

[13] Suzanne TM Bogaerds-Hazenberg, Jacqueline Evers-Vermeul, and Huub van den Bergh. What textbooks offer and what teachers teach: an analysis of the dutch reading comprehension curriculum. *Reading and writing*, 35(7):1497–1523, 2022.

[14] Amy E. Hilliard Rolf A. Zwaan, Gabriel A. Radvansky and Jacqueline M. Curiel. Constructing multidimensional situation models during reading. *Scientific Studies of Reading*, 2(3):199–220, 1998.

[15] David Bamman. Booknlp/booknlp: Booknlp, a natural language processing pipeline for books.

[16] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[17] Mandar Joshi, Omer Levy, Daniel S. Weld, and Luke Zettlemoyer. Bert for coreference resolution: Baselines and analysis, 2019.

[18] George R. R. Martin. *A Game of Thrones*, volume 1 of *A Song of Ice and Fire*. Bantam Books, New York, 1996.

[19] J. K. Rowling. *Harry Potter and the Philosopher's Stone*, volume 1. Bloomsbury Publishing, London, 1 edition, June 1997.

[20] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[21] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[22] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, 2019.

[23] Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, Deyi Xiong, et al. Evaluating large language models: A comprehensive survey. *arXiv preprint arXiv:2310.19736*, 2023.

[24] AI@Meta. Llama 3 model card, 2024.

[25] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las

Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[26] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model, 2024.

[27] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[28] Thomas Mesnard Gemma Team, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, and et al. Gemma. 2024.

[29] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.