

**IS6620 Group Project**

Final Report

# **The Harry Potter Expert**

**Team 4**

LU Wanshan 58876682

LI Jiayi 58965039

WEI Xiang 59157151

LIU Siliang 58894990

HUANG Yan 59037896

HUANG Xin 58932844

## Table of contents

Introduction.....	3
1. Project Idea .....	3
2. Key Components of the Solution .....	3
3. Main Findings and Deliverables .....	4
Methodology .....	5
1. Workflow.....	5
2. Data Sources and Preprocessing.....	5
3. Question Classification Module.....	6
4. Multi-Source Information Retrieval .....	6
5. RAG Response Module .....	6
6. Example.....	7
Key Challenges and Solutions.....	7
1. Lack of standard labeled data for classification.....	7
2. Hallucination .....	8
3. Retrieved information is wrong .....	8
4. Raw data is a mess .....	8
Project Results .....	8
Module/Package.....	8
Fine-Tuning.....	9
BookProcessor Class.....	9
TextClassifier Class.....	10
QueryProcessor Class .....	10
HPCharacterInfo Class .....	10
WebSearcher Class .....	11
Main Query Processing Function.....	11
Streamlit Application .....	11
Limitations and Future Work.....	12
Work Allocation .....	12
Appendices.....	13

# Introduction

## 1. Project Idea

The Harry Potter series created by J.K. Rowling has become a global phenomenon. By 2018, it had sold more than 500 million copies, covering movies, games and theme parks. The series has more than 750 characters and 2000 magical elements, poses an important navigation challenge. In order to help Harry Potter fans explore the magical world more conveniently and provide more accurate and context-related answers, we introduce "Harry Potter expert", a conversational artificial intelligence supported by Search Augmented Generation (RAG) technology. This tool ensures the consistency of accuracy among fans, scholars and creators, helps fan fiction authors to maintain the continuity of the world, helps film and game developers to combine new content with existing knowledge, and helps new readers to browse complex storylines and the rules of the Harry Potter world.

## 2. Key Components of the Solution

**Model Finetuning:** The question classification module uses BAAI/BGE-Ranker-V2-Gemma to fine-tune with Low-Rank Adaptation, and classify the problem into characters, plots or others. A customized data set was made for the training, covering all categories, and the questions generated by the template were supplemented to balance the data of different categories. Small, adjustable parts were added to transformer layers, and the AdamW optimizer with mixed-precision was utilized for efficiency over three training rounds.

**Question Classification:** User questions are sorted into "Character," "Plot," or "Other" categories. This classification is important for directing queries to the appropriate information sources. For example, questions about a character's name, wand, or actor fall under "Character," while inquiries about events or story actions are categorized as "Plot." Questions not fitting these categories are labeled "Other."

**Knowledge Retrieval from Books:** Information retrieval involves processing book texts into metadata-tagged chunks. An embedding model generates vector representations of these

chunks, stored in an in-memory array. When a query is made, the system converts it into a vector and calculates Euclidean distances against stored vectors using numpy. Relevant text chunks are retrieved based on similarity scores, enabling efficient matching and quick access to detailed information.

**Character Information Retrieval from API:** For character-specific questions, an external API provides detailed information including names, species, genders, houses, and other attributes. This allows users to look up specific characters by name or alias and receive comprehensive details, ensuring accurate and current character information.

**Web Search for External Knowledge:** For questions beyond the scope of the book dataset or character API, the system performs web searches. It extracts text content from relevant web pages, providing answers that require external knowledge, such as real-world inspirations behind Harry Potter elements.

**Chatbot Interface:** We designed a naughty-wizard style web chatbot that facilitates multi-round conversations. It processes user inputs, classifies questions, retrieves relevant information, and generates responses. Maintaining chat history ensures context continuity for subsequent interactions, making it easy for users to explore the Harry Potter universe.

### 3. Main Findings and Deliverables

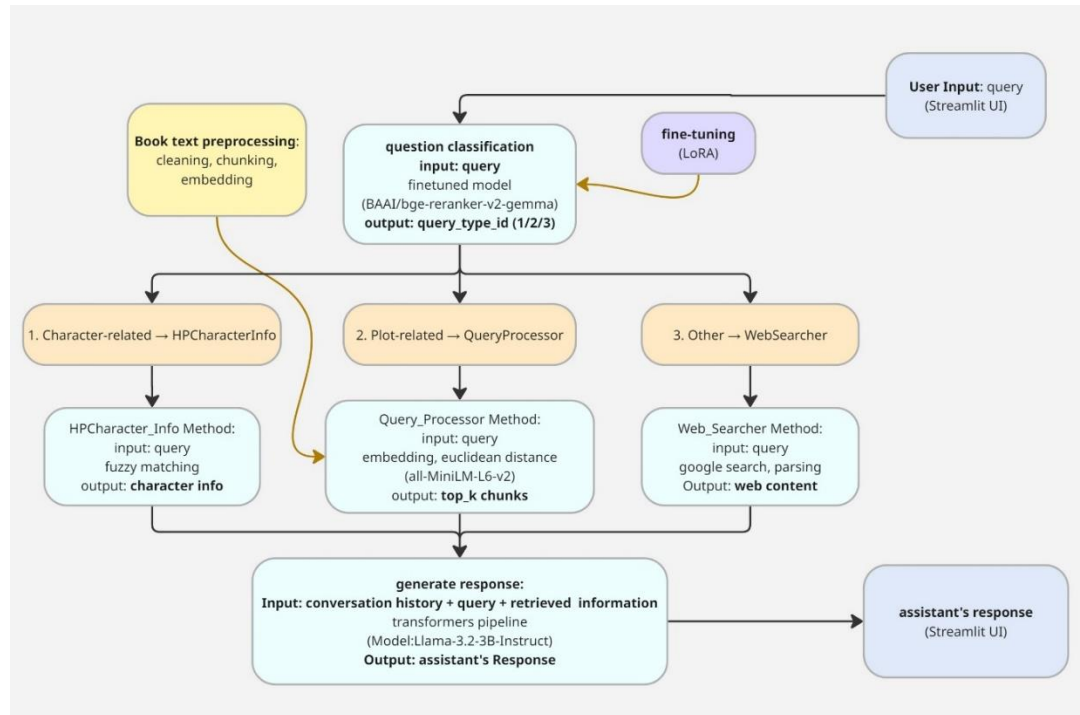
**Main Findings:** This system builds the Harry Potter Question answering system using RAG technology. First, the BookProcessor processes the books to generate the knowledge source. TextClassifier classifies problems. Different modules are called by category to retrieve information. Search results and the question are input into the large language model to generate responses, which not only improves the accuracy and timeliness of responses but also enriches the comprehensiveness of answers by integrating diverse knowledge sources, thus providing users with a more satisfactory and detailed query-answering experience.

**Deliverables:** A chatbot application that provides a user-friendly interface for multi-round conversations. It takes user input, classifies questions, retrieves information, and generates responses to streamline information retrieval for fans, scholars, and creators.

# Methodology

## 1. Workflow

The chart illustrates the workflow of the code.



## 2. Data Sources and Preprocessing

### 2.1 Datasets

- Structured Data: Character metadata (json.) such as name, species, gender, house, wand, and patronus from the public HP API (<https://hp-api.onrender.com/api/characters>)
- Unstructured Data: Harry Potter and the Sorcerer's Stone book (txt.). We selected the first book from [https://huggingface.co/datasets/WutYee/HarryPotter\\_books\\_1to7](https://huggingface.co/datasets/WutYee/HarryPotter_books_1to7)

### 2.2 Novel Text Processing and Embedding

To enable semantic search within unstructured text, we used the following workflow:

- Before chunking, we performed basic text cleaning such as lowercasing and whitespace removal to standardize the input.
- Chunking: Text was split into chunks using *RecursiveCharacterTextSplitter*.
- Embedding: We used the *all-MiniLM-L6-v2* model from *sentence-transformers* to encode each chunk into a vector and save into chunks.json file for efficient retrieval.

### 3. Question Classification Module

To ensure relevant retrieval logic is triggered for each query, we implemented a lightweight question classifier that routes questions into one of three categories.

#### 3.1 Model and Fine-Tuning Strategy

We used *AutoModelForSequenceClassification* as the base architecture, and loaded *LoRA-adapted weights* using *PeftModel* for parameter-efficient fine-tuning. The training set for finetuning was small in scale and constructed manually using sample queries and corresponding labels. The classifier was trained to assign one of three labels: "character", "plot", or "other", representing the main question categories in our system.

### 4. Multi-Source Information Retrieval

Depending on the classification result, the system fetches information from one of three sources: HP API, novel text, or web pages. These retrievals are modular and return string results to feed into the generation module.

#### 4.1 HP API character data retrieval (Fuzzy Matching)

If a question is classified as character-related (id:1), it will call the function and fetch character data from the API. It uses fuzzy matching to identify character names from the question. If the character name is found, it retrieves detailed information about that character.

#### 4.2 Plot Content Retrieval (Embeddings + Euclidean Distance)

If a question is classified as plot-related (id:2), it will call the function and the query is embedded using *all-MiniLM-L6-v2*. Similarity is calculated using *Euclidean distance* against all precomputed chunk embeddings. Finally, top-5 chunks are selected.

#### 4.3 Open Question Handling (Web Search)

For general questions beyond the above scope, the question is classified as other (id:3). It runs a *Google search* and fetches the top 3 URLs. Each page is parsed with *BeautifulSoup* to extract text from them.

### 5. RAG Response Module

After retrieving relevant information, the system uses a RAG approach to generate responses

based on both the query and the retrieved content.

### 5.1 Prompt Construction and Context Injection

The retrieved content is inserted into the system prompt alongside the user query:

- A recent history of user and assistant messages is retained.
- Retrieved passages are inserted as "system" messages to ground the model, once it is used in the current dialogue, it will be popped to save memory.
- A standard chat template is used to format the prompt.

This design helps the model focus on relevant content and avoid hallucinations.

### 5.2 Generation Model and Deployment

We used the *transformers.pipeline* interface to load *meta-llama/Llama-3.2-3B-Instruct*, which handles tokenization and generation internally, significantly simplifying the implementation.

- Loaded via Hugging Face Transformers.
- Automatically detects GPU for acceleration. If not, then CPU.
- Hosted on a Streamlit interface.

## 6. Example

This is a quick example of how this application runs:

- Query: "When did Harry first meet Ron?"
- Classified as a plot-related question and top similar chunks are retrieved.
- Retrieved chunks are appended to messages.
- The model generates a reply: " Harry first met Ron Weasley on the Hogwarts Express on the way to their first year at Hogwarts."
- The queries and answers are shown on web interface.

## Key Challenges and Solutions

### 1. Lack of standard labeled data for classification

We did not have access to a Harry Potter-specific question classification dataset. So we

manually created a small training set and used LoRA to fine-tune a base model with minimal resource requirements.

## 2. Hallucination

Preventing hallucinated answers from the language model LLMs tend to generate false or misleading content if not grounded in real context. We adopted a RAG approach that injects retrieved passages into the prompt and limited the max token output to reduce off-topic generation.

## 3. Retrieved information is wrong

We adjusted various parameter settings, such as chunk size and overlap size. Additionally, we ensured that each task used the most suitable model: different models were employed for text classification, text embedding, and dialogue tasks.

## 4. Raw data is a mess

All the Harry Potter book txt document have different structure and processing all the data is time consuming. Also, it is impossible to use only one function to handle all books. Since data preprocessing is not the purpose of this course, we simply used the first Harry Potter series book as our text resource, which giving us more time to improve the data quality and improve the accuracy of the model in later steps.

# Project Results

## Module/Package

We utilized the following module or packages in our code.

Module/Package	Components/Classes	Description/Purpose
os	-	file and directory operations, such as creating directories, joining paths, checking file existence, etc.
json	-	handling JSON data, such as reading and writing JSON files.
numpy	-	numerical computations and matrix operations, such as calculating Euclidean distances, saving and loading embedding vectors.



sentence_transformers	SentenceTransformer	load pre-trained sentence embedding models (e.g., all-MiniLM-L6-v2) and generate text embeddings.
langchain.text_splitter	RecursiveCharacterTextSplitter	split long texts into smaller chunks for further processing (e.g., generating embeddings).
transformers	AutoModelForSequenceClassification	load pre-trained text classification models.
	AutoTokenizer	tokenizing and encoding text to fit model inputs.
	AutoModelForCausalLM	load causal language models (e.g., generative models).
	TextIteratorStreamer	streaming text generation.
	pipeline	quickly build NLP pipelines (e.g., text generation).
	Trainer	a high-level training interface used to simplify the model training process.
	TrainingArguments	set configuration for training parameters (e.g., batch size, number of training epochs, logging directory, etc.).
peft	PeftModel	load and process LoRA (Low-Rank Adaptation) fine-tuned models, optimizing inference speed.
torch	-	loading deep learning models, inference, and tensor operations, with GPU acceleration support.
fuzzywuzzy	process	fuzzy string matching (e.g., extracting character names from questions).
googlesearch	search	perform Google searches and retrieve URLs of relevant web pages.
bs4 (BeautifulSoup)	BeautifulSoup	parse HTML documents and extract web content.
requests	-	send HTTP requests (e.g., calling APIs or fetching web content).
streamlit	-	build interactive web applications, providing user interfaces and session management.

## Fine-Tuning

In order to improve the accuracy of the question classification, we fine-tuned the text classification model based on a set of Harry Potter related questions with corresponding labels.

The steps are as follows:

- Use the *BAAI/bge-reranker-v2-gemma* as base model for three types of text classification tasks: character, plot, and other.
- Use *LoRA (Low-Rank Adaptation)* to fine-tune the model by adjusting only a small number of parameters while freezing most of the pre-trained model's parameters.
- Train the model using Hugging Face's Trainer. Save the fine-tuned model and tokenizer.

## BookProcessor Class

The *BookProcessor* class handles book text processing, including preprocessing, chunking, and generating embeddings. Key methods include:

- **preprocess\_text**: Cleans the text by removing extra spaces and converting it to lowercase.

- **load\_and\_preprocess\_book**: Loads book content, preprocesses it, and saves the cleaned text to a file.
- **chunk\_and\_embed**: Divides the text into chunks using *RecursiveCharacterTextSplitter* and generates embeddings for each chunk with the *SentenceTransformer* model. Chunk size is 1000, overlap size is set to 100.
- **save\_intermediate\_results**: Saves the text chunks and their embeddings in JSON and .npy formats respectively for future use.
- **process**: Orchestrates the entire workflow

### TextClassifier Class

The *TextClassifier* class is designed to classify user queries into predefined categories (e.g., character-related, plot-related, or other). Key methods include:

- **\_\_init\_\_**: Loads the classification model *AutoModelForSequenceClassification* and fine-tuned weights (*LoRA*) and initializes the classification labels.
- **classify\_text**: Classifies the input text and returns the predicted label along with its confidence score.
- **classify\_question**: Maps the classification result to a category id: 1, 2 or 3.

### QueryProcessor Class

The *QueryProcessor* class handles user queries by identifying the most relevant book content based on similarity. Key method includes:

- **generate\_query\_embedding**: Generates an embedding vector for the user query using the loaded embedding model *all-MiniLM-L6-v2*.
- **find\_most\_similar\_chunks**: Identifies the most similar text chunks (top 5) to the query based on *Euclidean distance* between embeddings.
- **process\_query**: Integrates all steps, from loading embeddings to returning chunks ([see appendix 1](#)).

### HPCharacterInfo Class

The *HPCharacterInfo* class retrieves detailed information about characters from the Harry

Potter series using an external API and answers character-related questions. Key methods include:

- **fetch\_characters\_data**: Fetches character data from an external API using *requests*.
- **extract\_character\_name**: Uses fuzzy matching (*fuzzywuzzy*, score > 70) to extract the character's name from the user's question.
- **get\_character\_info**: Retrieves detailed information about a specific character based on their name.
- **answer\_question**: Integrates the above steps to extract the character name, fetch their details, and return the information as a response ([see appendix 2](#)).

## WebSearcher Class

The *WebSearcher* class performs web searches to retrieve relevant content for user queries.

- **perform\_search**: Executes a *Google search* and extracts relevant webpage content. Each page is parsed with *BeautifulSoup* to extract text from them ([see appendix 3](#)).

## Main Query Processing Function

The `process_query_and_get_info` function determines the appropriate processing logic based on the user's query:

- **Query Classification**: Use `TextClassifier` to classify the query into one of three categories and call the corresponding function.
- **Response Generation**: Depending on the classification, retrieves the relevant information and generates a response.

## Streamlit Application

The application provides a simple web interface ([see appendix 4](#)) for users to input queries and receive answers.

- **process\_book\_once function**: Preprocesses the book content (e.g., chunking and embedding) only **once** during initialization to avoid redundant computations.
- **User Interaction Logic**: When a user submits a query, firstly, call *process\_query\_and\_get\_info* to determine the appropriate response. Then use a language model (*meta-llama/Llama-3.2-3B-Instruct*) to refine and generate natural-sounding reply.

## Limitations and Future Work

The Harry Potter Expert system faces several operational constraints. Presently, it **only analyzes content from the first book** and relies exclusively on **English-language datasets**, limiting its capacity to process non-English queries. This significantly reduces accessibility for global audiences. Furthermore, the **three-category classification framework struggles with nuanced overlaps inherent to the Harry Potter universe**. For example, questions regarding magical artifacts tied to character histories or events spanning multiple timelines often receive incomplete responses due to the system's rigid categorization structure. The system additionally **lacks mechanisms for analyzing causal relationships**, leading to oversimplified answers for questions like "How would removing an event affect a character's choices?"

To improve these aspects, future developments should prioritize expanding the system's coverage to include **all books** in the series while integrating multilingual support through **cross-language models** and region-specific datasets. **Enhancing the classification system** with subcategories such as "Magical Objects," "Historical Contexts," and "Character Motivations," validated through user testing, would better address complex inquiries. Implementing advanced frameworks for **understanding cause-effect relationships**, such as time-sensitive logic systems or event-based knowledge networks would strengthen analytical capabilities. These adjustments would create a more adaptable system capable of deeper exploration into the Harry Potter universe and similar fictional domains.

## Work Allocation

Final Reviewer: Lu Wanshan

Coding: Lu Wanshan, Liu Siliang

Report: Li Jiayi, Huang Xin, Jiang Xinling

PPT slides: Huang Yan, Wei Xiang

# Appendices

## Appendix 1

Retrieved information from book chunks.

```
71 # Example query
72 query = "what did harry potter receive on his birthday?"
73
```

问题 输出 调试控制台 终端 端口 评论

oss the table. behind the wild beard and eyebrows he wore a very kind smile. "don' you worry,

Distance: 0.9615

Chunk: hp 1 - harry potter and the sorcerer's stone chapter three letters from no one t he escape of the brazilian boa constrictor earned harry his longest-ever punishment. by the time he was allowed ou t of his cupboard again, the summer holidays had started and dudley had already broken his new video camera, crashed his remote control airplane, and, first time out on his racing bike, knocked down old mrs. figg as she crossed privet drive on her crutches. harry was glad school was over, but there was no escaping dudley's gang, who visited the house every single day. piers, dennis, malcolm, and gordo n were all big and stupid, but as dudley was the biggest and stupidest of the lot, he was the leader. the rest of them were all quite happy to join in dudley's favorite sport: harry hunting. this was wh y harry spent as much time as possible out of the house, wandering around and thinking about the end of the holidays, where he could see a tiny ray of hope. when september came he would be going off

Distance: 0.9700

Chunk: out. harry felt in the pocket of his robes and pulled out a chocolate frog, the very last one from the box hermione had given him for christmas. he gave it to neville, who looked as though he mig ht cry. "you're worth twelve of malfoy," harry said. "the sorting hat chose you for gryffindor, didn' t it? and where's malfoy? in stinking slytherin." neville's lips twitched in a weak smile as he unwra pped the frog. "thanks, harry...i think i'll go to bed....d'you want the card, you collect them, don't yo u?" as neville walked away, harry looked at the famous wizard card. "dumbledore again," he said, "he was the first one i ever -" he gasped. he stared at the back of the card. then he looked up at ron an d hermione. "i've found him!" he whispered. "i've found flamel! i told you i'd read the name somewher e before, i read it on the train coming here - listen to this: 'dumbledore is particularly famous for his defeat of the dark wizard grindelwald in 1945, for the discovery of the twelve uses of

Distance: 0.9845

Chunk: corner of the room. uncle vernon made another funny noise, like a mouse being trodden on. "any way - harry," said the giant, turning his back on the dursleys, "a very happy birthday to yeh. got su mmat fer yeh here - i mighta sat on it at some point, but it'll taste all right." from an inside pock et of his black overcoat he pulled a slightly squashed box. harry opened it with trembling fingers. i nside was a large, sticky chocolate cake with happy birthday harry written on it in green icing. harr y looked up at the giant. he meant to say thank you, but the words got lost on the way to his mouth, and what he said instead was, "who are you?" the giant chuckled. "true, i haven't introduced meself. rubeus hagrid, keeper of keys and grounds at hogwarts." he held out an enormous hand and shook harry' s whole arm. "what about that tea then, eh?" he said, rubbing his hands together. "i'd not say no ter summat stronger if yeh've got it, mind." his eyes fell on the empty grate with the shriveled

## Appendix 2

Retrieved information from HP API.

```
58 question = "ron weasley's birthday"
59 print(answer_question(question))
```

问题 1 输出 调试控制台 终端 端口 评论

Information about Ron Weasley:

- Id: c3b1f9a5-b87b-48bf-b00d-95b093ea6390
- Name: Ron Weasley
- Alternate\_names: ['Dragomir Despard', 'Ronald', 'Ickle Ronniekins', 'Ronnie', 'Wheezy', 'Won-Won', 'Roonil Wazlib']
- Species: human
- Gender: male
- House: Gryffindor
- Dateofbirth: 01-03-1980
- Yearofbirth: 1980
- Wizard: True
- Ancestry: pure-blood
- Eyecolour: blue
- Haircolour: red
- Wand: {'wood': 'willow', 'core': 'unicorn tail hair', 'length': 14}
- Patronus: Jack Russell terrier
- Hogwartsstudent: True
- Hogwartsstaff: False
- Actor: Rupert Grint
- Alternate\_actors: []
- Alive: True
- Image: <https://ik.imagekit.io/hpapi/ron.jpg>

## Appendix 3

Retrieved information from web pages.

```
46 if __name__ == "__main__":
47     query = "Who is the actor that plays Harry Potter?"
48     search_results = web_search(query)
49
```

问题 1 输出 调试控制台 终端 端口 评论

Search Results:

Result 1:  
URL: <https://medium.com/pantheon-of-film/worst-to-best-harry-potter-movies-75173d6a2c9f>  
Content Preview: Worst to Best Harry Potter Movies | by Kayla Witman | Pantheon of Film | Medium Open in app Sign up Sign in Write Sign up Sign in Pantheon of Film · A brand new publication showcasing the best and brightest of film enthusiasts. Well-written essays, discussions, and reviews are welcome within the Pantheon. Email me at rjbrow2021@gmail.com to make yourself known! Member-only story Worst to Best Harry Potter Movies Kayla Witman · Follow Published in Pantheon of Film · 3 min read · Aug 8, 2023 -- 7 ...

Result 2:  
URL: [https://en.wikipedia.org/wiki/Daniel\\_Radcliffe](https://en.wikipedia.org/wiki/Daniel_Radcliffe)  
Content Preview: **Daniel Radcliffe** - Wikipedia Jump to content Main menu Main menu move to sidebar hide Navigation Ma in page Contents Current events Random article About Wikipedia Contact us Contribute Help Learn to edit Community po rtal Recent changes Upload file Special pages Search Search Appearance Donate Create account Log in Personal tools D onate Create account Log in Pages for logged out editors learn more Contributions Talk Contents move to sidebar hide (Top) 1 Early life and education 2 Career Toggle Car...

PS C:\Users\卢飞飞>

## Appendix 4

Web interface with conversation samples.


# Harry Potter Wizard Land

 You: tell me how many pages are there in the first harry potter book?

 Assistant: *mischievous grin*  
The first Harry Potter book, 'Harry Potter and the Philosopher's Stone' (published as 'Harry Potter and the Sorcerer's Stone' in the United States), has 320 pages in its British edition.

 You: when did harry first meet ron?

Query classified as: Plot-related (Category ID: 2)

 Assistant: *grin*  
Harry first met Ron Weasley on the Hogwarts Express on the way to their first year at Hogwarts.

Ah, curious soul, what secrets do ye wish to uncover? 