ChatRegex for Detective Novels

Aaron Daniels, Bryan Finlayson, Austin Rhodes, Alexis Tochiki

University of Tennessee at Knoxville

CS524 Natural Language Processing

I. Introduction

Formally, regular expressions define the set of strings of a regular language over a finite alphabet. In practice, they can be used as a tool to perform text analysis programmatically. Regular expressions are often used in natural language processing pipelines to match regular patterns in text, such as identifying specific names or syntactic constructs. By using regular expressions, operations such as input validation and string recognition are possible.

The goal of our project is to produce an interactive chatbot that can report various stats and analysis from three famous crime novels. All text analysis will be performed using regular expressions to parse both the chatbot input and the novel text itself. The project is implemented using Python 3.10 and using only the `re` package.

II. Implementation

The project consists of two major parts: the prompt processor and the pipeline.

The prompt processor is designed to act as an intent-based chatbot that matches the user's input to phrases with corresponding intents. Each intent triggers the building of the corresponding answer.

The pipeline parses a text file that contains a crime novel, breaking the file into chapters and sentences. To assist with parsing, we included dictionaries containing keywords in each novel. The pipeline's text parser is also responsible for finding certain occurrences of words and surrounding words. In addition, the pipeline is where the answers to questions are built. Each intent within the prompt processor calls its correlating method in the pipeline to create an answer.

III. Results/Findings

Overall, the project was successful. Most questions are correctly answered with the required details. While most details are correctly retrieved for each correctly asked question, there can sometimes be too many details returned for the question. A more granular question and answer chatbot would have been preferred.

Query	A Study In Scarlet	The Secret Adversary	The Sign of Four
When does the investigator occur for the first time?	Success	Success	Success
When is the crime first mentioned?	Success	Success	Success
When is the perpetrator first mentioned?	Success	Success	Success
What are the three words that occur around the perpetrator?	Success	Success	Success
When and how do the detective/perpetrators co-occur?	Failure	Failure	Failure
When are other suspects first introduced?	Success	Success	Success

Table 1. Our final results in answering the 6 queries provided

Analysis of the text led to some discoveries. Initially, we theorized that the investigator would always be the main character in the book. Counting the number of occurrences shows that this is mostly true. There is one counter example in "The Secret Adversary". Further investigation shows that this likely occurs because of the plot. It seems clear there is some dramatic betrayal in this book. The perpetrator is probably close to the main character.

Book	Count of Investigator	Count of Perpetrator	(investigator - perpetrator) / Count of both characters
A Study In Scarlet	48	35	0.16
The Secret Adversary	8	88	-0.83
The Sign of Four	33	17	0.32

Table 2. Comparison of occurrences of investigator vs perpetrator for each book.

Analysis of the most common words surrounding (3 words) the investigator and perpetrator revealed that most characters are portrayed with a neutral tone. This is likely due to the nature of crime novels. The enjoyable part of the book is the mystery. The reader should be surprised by the ending. Portraying any character too negatively may spoil the mystery and make the book much less fun to read.

	Investigator		Perpetrator	
Book	Name	Words	Name	Words
A Study In Scarlet	Sherlock Holmes	sprang know rose	Jefferson Hope	among led able
The Secret Adversary	'Tommy' Beresford	adventures great developments	'Mr. Brown'	julius inspector called
The Sign of Four	Sherlock Holmes	took watson see	Jonathan Small	get treasure associates

Table 3. Most common words surrounding investigator and perpetrator

IV. Discussion

Honorifics vs sentences

In the early stages of our design, we ran into the issue of sentence counts being influenced by punctuated honorifics or abbreviations. The period in "Dr. Watson" would be interpreted as a sentence terminator. Other situations include abbreviations ("LTD."), punctuated abbreviations ("U.S.A"), and initialized names ("John H. Watson"). A great deal of effort went into identifying and discerning these punctuations to improve sentence splitting accuracy.

Permutations of character's names

Though finding a specific character's name was not difficult, considering every permutation of a character's name complicated the regex searches considerably. Without properly setting up these permutation regexes, it is possible to double count occurrences of a character within the story. In some cases, there is ambiguity if two characters share the same first name, for example, so we erred on the side of stricter name recognition to improve quality over quantity of identifications.

What worked well?

Regex parsing is particularly well suited for dealing with formatted or annotation text. For instance, all of the Gutenberg metadata was easily removed by searching for the intentionally placed "*** *START OF THE PROJECT GUTENBERG EBOOK* ***" strings. Header/footer text, such as chapter identifiers were easily located for processing the text into chapter chunks. Splitting the text into sentences was mostly feasible with regex but nears the limit of regex's utility as discussed later in this section.

Regex is especially useful when the target of a search is known verbatim. First and last names, or specific strings are easily searched. Known string searches are easily implemented and performant.

What did not work well?

Regex by nature is not powerful enough to predict the story without significant preparation (hardcoding)

Initially, an effort was made to predict the results completely generically. For instance, rather than searching for the literal strings "Sherlock" or "Sherlock Holmes" to find when the investigator is first mentioned in the text, we searched for the most frequent proper noun (via capitalized words) with the assumption that the most frequently mentioned proper noun would be the main character, then found the first occurrence of that string. While this worked in some cases, it was not a suitable approach for more advanced queries such as "what was the crime?" as these tend not to generalize well. The generic method of finding a crime for Doyle's books would not be applicable to Christie's book, as her story explores crime in a markedly different way. Our final design's book-specific profile approach yielded vastly better results than the generic approach in all cases. We reason that regex is not powerful enough to conduct advanced text analysis generically in the way that other tools like ChatGPT are able to.

Some punctuation was very difficult to parse: Our sentence parsing system is liable to get hung up on awkward or unexpected punctuation. Already mentioned is the challenge we faced with honorifics or abbreviations, but in some instances the challenge was insurmountable. For example, in the following snippet from A Study in Scarlet:

"[…] and having cards in his pocket bearing the name of 'Enoch J. Drebber, Cleveland, Ohio, U.S.A.' There had been no robbery, nor […]"

Doyle is separating these two sentences with the last intra-quote period punctuation in the word "U.S.A." (underlined). It is exceedingly difficult to discern this as a sentence terminator in such a way that it does not also interpret other punctuated abbreviations as sentence terminators. The only solution presented is to assign a special case to this exact string.

V. Conclusions

Regex is well suited for finding common patterns in data but struggles when used alone. It would be much more effective if it were paired with some additional components to identify parts of speech and proper names.

If we were to approach this another time, we would not solely rely on regex to find answers. An LLM or regex in combination with a package like Spacey would be much better suited for programmatically understanding the text in the way the project requirements had requested.

The 3 books are completely separate entities. The book must be selected before the chat it initialized. I would have liked to have the book name parsed from each question.