CS7637: Knowledge-Based AI:

Mini-Project 3

Yaxin Yang

yyang894@gatech.edu

Abstract— This paper provides a comprehensive analysis of an agent designed for question-answering tasks. The agent employs fundamental Natural Language Processing (NLP) techniques such as tokenization, part-of-speech tagging, and shallow parsing. It relies on heuristic-based approaches, including keyword matching and simple conditional logic, to extract answers from sentences. However, its performance is constrained by its limited context understanding, especially when compared to human cognition. While humans possess deep linguistic understanding, flexibility, and the ability to infer and reason, the agent operates strictly within its predefined rules, lacking the nuance and adaptability inherent in human language processing. The agent's efficiency is linear with respect to the input size, but it may not scale well with increased sentence complexity. Despite its constraints, the agent showcases practical applications of foundational NLP concepts in a question-answering system.

1. HOW DOES YOUR AGENT WORK? DOES IT USE SOME CONCEPTS COVERED IN OUR COURSE? OR SOME OTHER APPROACH?

1.1 Tokenization and Text Preprocessing

remove_punctuation: Removes punctuation from text, which is a common preprocessing step in NLP. process_text: Tokenizes the text into words and converts them to lowercase. Tokenization is a fundamental step in NLP that divides text into words, phrases, symbols, or other meaningful elements called tokens.

1.2 Part-of-Speech Tagging

get_word_dict: This function suggests that linguistic_data contains part-of-speech (POS) tags for words, and word_dict is created to group words by their POS tags. POS tagging is a common NLP task that labels words with their corresponding part of speech (e.g., noun, verb, adjective).

1.3 Dependency on External Linguistic Analysis

In the __init__ method, an instance of LinguisticAnalysis is created and some linguistic data is loaded. This suggests the agent relies on external linguistic databases.

1.4 Question Answering

The solve method is the core of the question-answering functionality. It processes both the sentence and the question and based on certain keywords in the question (e.g., "who", "when", "where", "what", "how"), it tries to extract and return relevant information from the sentence.

1.5 Keyword Matching and Simple Heuristics

The agent employs keyword matching and simple heuristics to identify and extract relevant information. For example, if the question contains the word "who", the agent looks for proper nouns or nouns in the sentence.

1.6 Shallow Parsing

The agent does some form of shallow parsing by identifying keywords and their potential positions in sentences to extract answers, especially in the "where" and "what" question cases. Shallow parsing (or chunking) is a process of extracting phrases from text without diving deep into grammatical structures.

1.7 Iterative and Conditional Logic

The agent employs iterative and conditional logic to traverse through words and conditions to find a suitable answer. This is seen in multiple if-else conditions and loops. These techniques and approaches are heuristic-based this agent provides a practical example of how these concepts can be implemented in a question-answering system.

2 HOW WELL DOES YOUR AGENT PERFORM? DOES IT STRUGGLE ON ANY PARTICULAR CASES?

The performance of this agent is highly dependent on the simplicity and structure of the input sentence and question, as well as the accuracy of the linguistic analysis it relies on. After submitting to Gradescope, my agent only got one incorrect.

Sentence: "Give us all your money."

Question: "Who should you give your money to?"

The ideal answer should be "us," reflecting the entity to whom the money should be given according to the sentence. Analyzing the code, the solve method processes both the sentence and the question to generate word dictionaries and raw word lists. The method then checks for the keyword "who" in the question, which triggers the relevant block of code to look for a proper noun ('PROPN') or noun ('NOUN') in the sentence to answer the "who" question. However, the code as it stands might not provide the correct answer because:

2.1 Limited Context Understanding

The agent doesn't understand context or semantics, it merely looks for pattern matches based on pre-defined rules. Hence, it lacks the ability to understand the implied context within the question and the sentence.

Given these limitations, the agent might struggle with this particular example, and similar cases where understanding the grammatical relationships between words or handling pronouns is essential for providing a correct answer. Improving the agent to handle such cases might require integrating more advanced NLP techniques such as dependency parsing and pronoun resolution, or utilizing more sophisticated NLP models that have a better understanding of language semantics and context.

3 HOW EFFICIENT IS YOUR AGENT? HOW DOES ITS PERFORMANCE CHANGE AS THE SENTENCE COMPLEXITY GROWS?

3.1 Efficiency

Time Complexity: The time complexity of the agent's operations largely depends on the implementation of the underlying LinguisticAnalysis class and the linguistic_data it uses. However, the explicit operations in the provided methods like remove_punctuation, get_word_dict, and the various conditional checks within solve are generally linear (O(n)) with respect to the number of words or characters in the input text.

Space Complexity: The space complexity is also linear (O(n)) as the agent creates data structures to store words, their corresponding part-of-speech tags, and other processed information.

3.2 Performance with Increasing Complexity

3.2.1 Lack of Scalability

The agent's heuristic and keyword-based approach does not scale well with increasing sentence complexity. More complex or longer sentences may contain more potential answer candidates, more distracting information, and more complex linguistic structures that the agent's simple heuristics are not equipped to handle.

3.2.2 No Handling of Multi-Sentence Texts

The agent is designed to work on single sentences. If the input text contains multiple sentences, the agent’s simplistic approach will struggle to identify relationships across sentences, making it ill-suited for handling multi-sentence texts.

4. DOES YOUR AGENT DO ANYTHING PARTICULARLY CLEVER TO TRY TO ARRIVE AT AN ANSWER MORE EFFICIENTLY?

4.1 Basic Text Preprocessing

The agent performs some basic text preprocessing like removing punctuation and converting text to lowercase which can help in standardizing the text and potentially reducing the complexity of the matching process.

4.2 Part-of-Speech Grouping

It groups words by their part-of-speech tags using the get_word_dict method. This can potentially speed up the process of finding relevant words by reducing the search space when looking for specific types of words (e.g., nouns, verbs, etc.).

4.3 Early Exit

In some cases, the agent uses an early exit strategy where it returns an answer as soon as a matching word or condition is found, such as in the case of the "when" and "where" keyword checks. This can potentially improve efficiency by avoiding unnecessary processing.

5 HOW DOES YOUR AGENT COMPARE TO A HUMAN? DO YOU FEEL PEOPLE INTERPRET THE QUESTIONS SIMILARLY?

5.1 Understanding and Context

Human: Humans possess a deep understanding of language, context, and world knowledge which allows them to interpret questions and sentences in a nuanced and sophisticated manner. They can understand implied meanings, emotions, and the broader context in which a sentence or question is posed.

Agent: The agent lacks understanding and can only match patterns have based on predefined rules. It does not grasp context or the broader meanings that might be obvious to a human reader.

5.2 Flexibility and Adaptability

Human: Humans can adapt to new language patterns, slang, or novel sentence structures easily. They can also understand misspellings, grammatical errors, and informal language.

Agent: The agent is rigid in its interpretation and can easily fail if the input deviates from the expected format. It is not equipped to handle variations in language expression.

5.3 Semantic Understanding

Human: Humans can understand synonyms, antonyms, and the semantic relationships between words. They can also infer meanings based on the semantic context.

Agent: It lacks semantic understanding and is unable to recognize synonyms or understand semantic relationships between words.

5.5 Learning and Improvement

Human: Humans learn from their mistakes and from feedback, improving their understanding and interpretations over time.

Agent: The agent does not have a learning mechanism to improve its performance over time based on feedback or new data.

5.6 Inference and Reasoning

Human: Humans can perform reasoning and inference to derive answers even when the information is not explicitly stated.

Agent: The agent can only operate on explicit information provided in the input and lacks the ability to infer.