Nick Liccini
CS4635 Section A – Project 2 Reflection
10.30.2019

Introduction

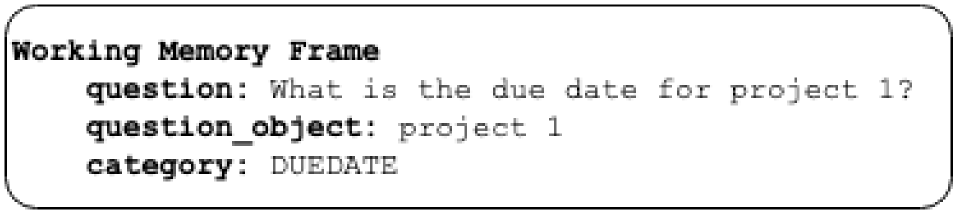
The objective of this project is to classify a user's question based on some domain knowledge. Project 1 focused on implementing a solution to the problem of understanding a question and representing that information in a way that the agent can easily analyze. Project 2 raises deeper questions such as, *"What is thinking?"*, *"How is memory represented and stored?"*, or *"How are perceptions and memory compared?"* It is first critical to understand how a human organizes memory and approaches classifying a sentence before the agent can be designed.

Human Thought Process

Humans have prior knowledge about many subjects. When classifying anything, the scope of the problem space must be constrained; let's call this the domain which has various topics, relevant information and experiences, and rules. Humans also store a lot of information, but it's not very organized so let's say this domain knowledge is stored as an unordered lump of information. Now that we understand how a human's knowledge might be organized, let's look at the task of classifying a sentence:

"What is the due date for project 1?"

First, we must understand what the question is about in the context of our domain. Since this is about CS4635, our topics are limited to DUEDATE, RELEASEDATE and others. We know keywords for each topic and this knowledge allows us to infer the topic by verifying which one has the most matching keywords, here it's DUEDATE. Next, our grammatical constraints help us understand that the question object is "project 1" by leveraging the constraints set up by verb phrases and prepositional phrases. We have now formed a working memory that we'll use to compare to our memory.



```
Working Memory Frame
    question: What is the due date for project 1?
    question_object: project 1
    category: DUEDATE
```

Figure 1. Working Memory represented as a frame.

Next, let's try to recall what we know about CS4635. We know a bunch of things because we've read the syllabus (information) and we've looked through Piazza (experience), so let's search through what we know and see how similar it is to our working memory. Here we may apply some sort of evaluation function for how similar our working memory is to each bit of memory; once we've found the best match, we can use that as the question intention.

This was my thought process when looking through the questions in the project description; it involved searching through my memory and ranking how similar my knowledge was with what the question was asking, if I found a good enough match then that's what I chose as the intention. But not every question is that simple, take for example:

"When is project 1?"

Even having constraints on the problem and using domain knowledge I have no idea what the intention of this sentence is. Since I've failed to understand the topic, I will fail to classify its intention. This situation arises when there is ambiguity, extraneous vocabulary, or a lack of useful features in the perception (e.g. keywords). My natural response to this question would be the reaction, "I don't know" to indicate that I've tried to classify it but nothing in my memory indicates I have experience or information about this question.

Human Thought Process and KBAI

The first task was understanding the question; here thematic roles were used to apply grammar rules and determine the object of the question. After parsing the sentence, its main features were extracted and put into a frame that represented my working memory. This working memory was then compared to cases in my episodic knowledge by following the rules of my procedural knowledge. If a case was evaluated as matching, then it was selected to be adapted and stored as more episodic knowledge.

The figure below depicts the combination of these concepts as they pertain to my classification process:

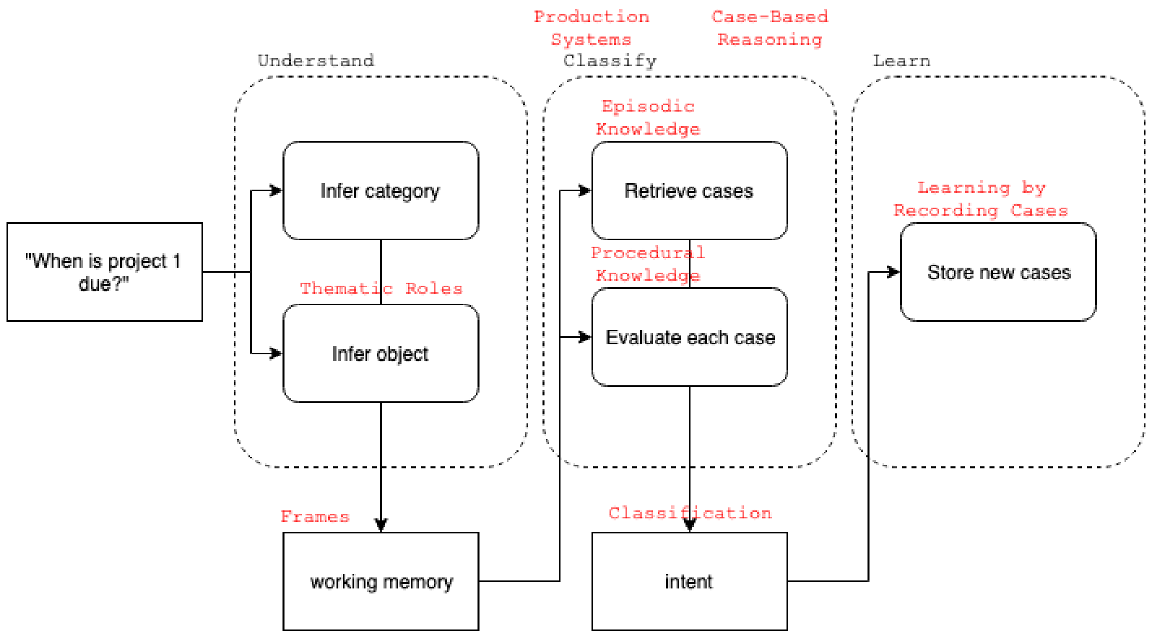


Figure 2. Human classification process highlighting KBAI methods.

Agent Design

The agent's knowledge consists of two components: Understanding and Cognition. Its understanding component leverages keywords, preferences, and grammatical constraints to infer a working memory frame from the input question; this component was built in Project 1. The cognition component is based on a production system cognitive architecture that uses the Soar model for its deliberation level. The Episodic Knowledge consists of an unordered set of cases which may be either an experience or some information, each case type has an implementation of a comparison function which follows a set of rules defined as the Procedural Knowledge. There is a Reaction level implementation that returns a default response (e.g. "I don't know") when the working memory cannot sufficiently be matched to any knowledge. The figure below walks through the agent's framework and classifies the question, *"What project is due on week 6?"*

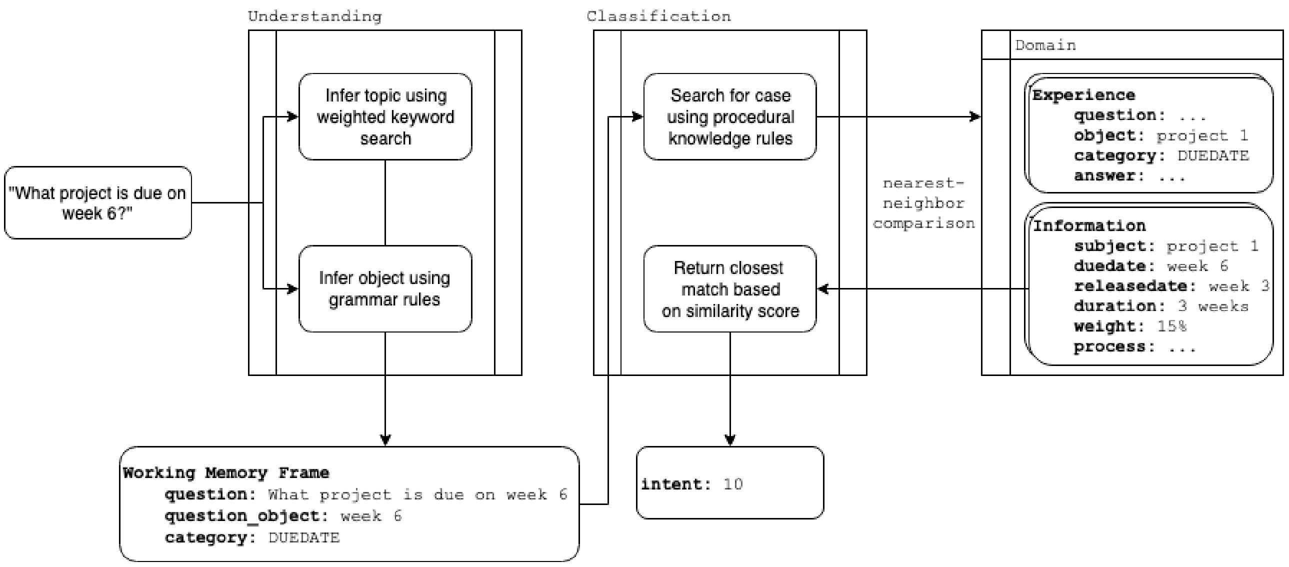


Figure 3. Agent architecture with example data.

Agent vs Human Thought Processes

In code there are limitations and code is a more formal representation than human language. Fig. 2 and Fig. 3 both represent the same workflow for classifying a question, yet the nuances are quite distinct. First let's examine the structure of knowledge then the implementation details of the classification process.

When designing the agent, first I thought, "How is my memory organized?" Then I thought, maybe it isn't organized at all! One way to represent memory is as a pile of random information and experiences. However, I acknowledged that I always know to pull out the most relevant information in conversation without much thought. What sort of data structure allows such quick access to so many pieces of information by so many different indexing methods? This is called the indexing problem that surrounds case-based reasoning techniques (Kolodner, 1992, p. 5). As the indexing problem is very complex, a design choice was made for my agent: it stores information as an unordered set that gets iterated through entirely for every new input. An unordered set provides some degree of probability that every case is equally distant from the desired case which partially addresses the indexing problem by uniformly distributing the cases. This may seem inefficient but for small, related groups of information the inefficiencies are negligible, and they reflect the human mind in that there is a lot of information available in a semi-random, quickly accessed manner.

Looking at the classification process, humans use a variety of heuristics to quickly come up with answers that are often correct. Some keywords hint at the likelihood that some cases are more relevant, or some ambiguity in the question hints that experience won't help. Since defining all of these heuristics and indexing methods would be very complex and possibly intractable, the chosen design simply iterates through all of the known cases and computes a similarity score to find the most relevant case. Here, Procedural Knowledge rules are used to evaluate how similar two cases are. For example, a human might remember one previous experience was about the due date for project 1 being week 6, and if the question was interpreted to ask if a project is due on week 6, the case would return a high score and thus the case about project 1 is the best match. The figures below provide pseudocode for determining the similarity of a case to an input:

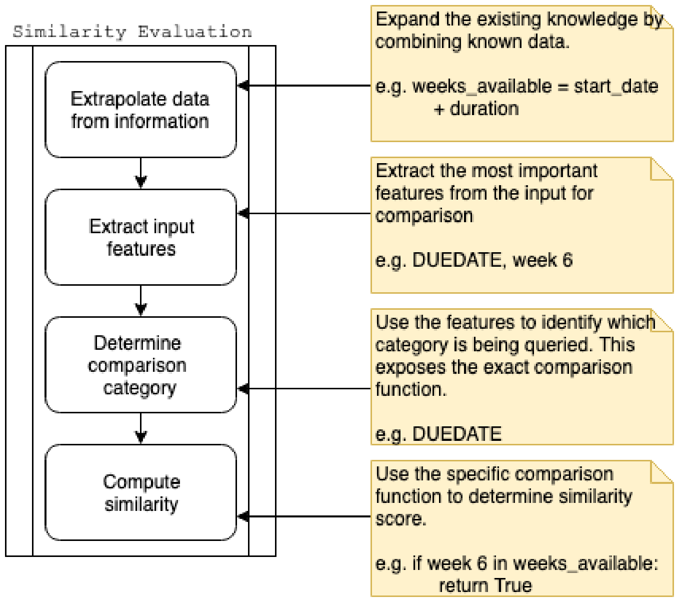
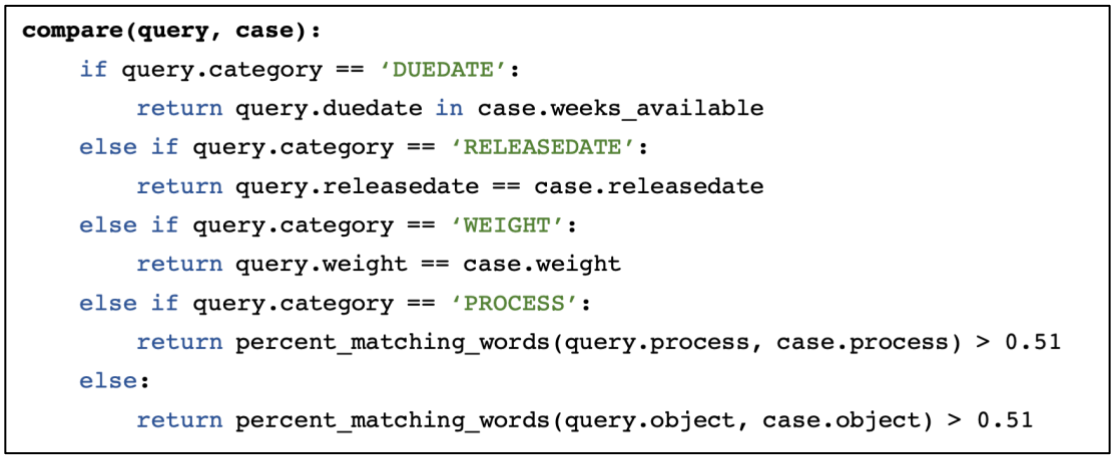

Similarity Evaluation
Extrapolate data from information
Extract input features
Determine comparison category
Compute similarity
Expand the existing knowledge by combining known data.
e.g. weeks_available = start_date + duration
Extract the most important features from the input for comparison
e.g. DUEDATE, week 6
Use the features to identify which category is being queried. This exposes the exact comparison function.
e.g. DUEDATE
Use the specific comparison function to determine similarity score.
e.g. if week 6 in weeks_available: return True

Figure 4. Algorithm design for the nearest neighbor similarity evaluation function.



```
compare(query, case):
    if query.category == 'DUEDATE':
        return query.duedate in case.weeks_available
    else if query.category == 'RELEASEDATE':
        return query.releasedate == case.releasedate
    else if query.category == 'WEIGHT':
        return query.weight == case.weight
    else if query.category == 'PROCESS':
        return percent_matching_words(query.process, case.process) > 0.51
    else:
        return percent_matching_words(query.object, case.object) > 0.51
```

Figure 5. Pseudocode for the nearest neighbor similarity function.

Testing

Test questions were generated based on the constraints provided in the instructions and using information that was available in the provided domain (tables and vocabulary). Some questions used extraneous vocabulary, ambiguous topics, or roundabout ways of asking for certain pieces of information. Questions that were simple, straightforward, had useful features (keywords), and directly related to the domain knowledge were more confidently classified.

Number	Topic	Object	Intent	Question	Inferred Topic	Inferred Object	Inferred Intent	Understood?	Classified?
1	DUEDATE	project 1	10	what is the due date for project 1	DUEDATE	project 1	10	YES	YES
2	DUEDATE	week 6	10	what is due on week 6	DUEDATE	week 6	10	YES	YES
3	WEIGHT	final	32	how much of my grade is the final worth	WEIGHT	final	32	YES	YES
4	WEIGHT	20%	32	what is worth 20%	WEIGHT	20%	32	YES	YES
5	PROCESS	syllabus	0	where is the syllabus	PROCESS	syllabus	0	YES	YES
6	WEIGHT	15%	28	what is worth 15%	WEIGHT	15%	26	YES	NO
7	WEIGHT	week 6	26	what contributes to my grade on week 6	WEIGHT	week 6	0	YES	NO
8	PROCESS	code	34	how do i submit my code	PROCESS	code	0	YES	NO
9	WEIGHT	project 1	26	what speculation can i make about my grade for project 1	UNKNOWN	UNKNOWN	0	NO	NO
10	DUEDATE	week 3	9	what is required by week 3	DUEDATE	week 3	0	YES	NO

Incorrectly Classifying

The latter 5 questions were mostly understood correctly but not classified correctly because they were complex, used vocabulary outside of the agent's knowledge, had ambiguities, and lacked useful features. The agent has knowledge about multiple assignments worth 15%, so it was unable to identify what the user was asking about since the question lacked useful features. The agent does not know what a "speculation" is, so it reacted with a default response. Thus, it is most important for the input to have many useful features so that it can be compared effectively to the agent's domain knowledge to properly classify it.

Metacognition

Reflecting on the design of this agent has revealed that humans implicitly apply many different evaluation functions similar to the comparison functions used by the agent. These functions can be rudimentary like a heuristic or they can be nuanced to closely analyze details. Humans also tend to respond with their first matching case rather than the closest matching one because it is easier to use trial and error than it is to carefully analyze each solution (e.g. To answer "When is the project", one might say "It's due on week 6" and if that is not the desired answer, one might try again saying "It's available during week 5").

Another interesting aspect of human cognition that arose during this project was the indexing problem. How is it that humans store and access cases from memory so quickly? The simple unordered set used in this agent does not give justice to the incredibly complex human mind. Perhaps the agent could be designed to constantly sort its knowledge based on the environment and perceptions, but there is no easy way to access such diverse information as quickly as humans can.

References

Kolodner, J. L., (1992). *Artificial Intelligence Review: An Introduction to Case-Based Reasoning*. Retrieved from http://alumni.media.mit.edu/~jorkin/generals/papers/Kolodner_case_based_reasoning.pdf

Weber, R. O. (2005). *Textual Case-Based Reasoning.* Retrieved from https://www.researchgate.net/profile/Rosina_Weber/publication/225070215_Textual_Case-Based_Reasoning/links/5759ad5308aed884620b2481.pdf

Bruninghaus, S. and Ashley, K. D (2005). *Reasoning with Textual Cases*. Retrieved from https://www.semanticscholar.org/paper/Reasoning-with-Textual-Cases-Brüninghaus-Ashley/2d224a9b8762cb3903067015400f2245063a0830