DAT410/DIT728

Group 54
Module 7: Dialogue Systems and Question Answering

Jonatan Hellgren, 960727-7010
Matematikprogrammet GU
gusheljot@student.gu.se

Ankita Rahavachari, 19960709-4605
MPDSC Chalmers University of Technology
ankita@student.chalmers.se

8th March 2021

We hereby declare that we have both actively participated in solving every exercise. All solutions are entirely our own work, without having taken part of other solutions.

Hours spent working

Jonatan	Ankita Rahavachari
15	15

1 Reading & Reflection

In the paper we got to read about GUS (Genial Understander System), which is a frame-based dialogue system that is able to book flights from a dialogue. How a frame-based dialogue system works is that it basically tries to fill out a form by directing questions to get the users to respond with the relevant information to be able to fill out the form. This is paired with natural language processing so that the system is able to respond accordingly to the user. The replies generated by the dialogue system is however pre-constructed sentences with blank-spaces, these blank-spaces gets filled in when the form gets filled in and thus the dialogue will be able to proceed until an eventual booking is possible.

Frames to direct the dialog: In order to conduct a dialog, an instance of the dialog frame is initially created. Each frame goes through the slots of instance and attempts to find fillers. Once the slot is filled by a new instance then slots of that instance are filled in the same way. GUS systematically completes the work on the given slot before moving to the next. It follows a depth-first recursive process. In this manner the initiative of the conversation is retained by GUS.

Problems of natural dialog and GUS handling it:

- The designers assure the participants a dominating position in the interaction. This skill often leads to the success of interactive computer systems. Mostly during natural conversations participants assume the initiatives from time to time. GUS tries not to spoil the flow of the conversation but retains the initiative. It is considered as a mixed initiative system.
- In the natural dialog the inference in the response can be complex that it is difficult to interpret indirect answers. In such cases GUS can handle the problem in simpler ways. It has very narrow expectations about the subject matter and the client's goal
- It is important that both of the participants are able to understand the given word or a phrase referring to the same event or object. GUS can associate the word to the previous occurrence and respond relative to it. This reference requires some complicated reasoning involving both the content and context of the conversation.

Conclusion:

This kind of dialogue systems are good at their specific tasks, they do however become very narrow since they are mainly rule-based and each sentence the system is able to use has to be written by hand. It essentially becomes an interactive experience with a database that is wrapped up in pre-constructed sentences. This could however be very useful in domains where the database is hard to interpret, for example if an less technical elder person would need to book a flight it might be easier for them to do it with a dialogue system rather then using the booking on a web-page.

2 Implementation

2.1 Outline of dialogue system

Our digital assistant is implemented with a database from stores, resturants and post offices located at Mariaplan in Gothenburg. It is implemented is such a way that the first task is to find what sort subject you need help with and later recommending a place for the user, in the last part it is possible to ask for information from the recommended place. Before the digital assistant fills in the current subject and place in the conversation validation questions are asked to make sure that it is correct. The general outline of a conversation looks like this:

- welcome message, to initialize dialogue,
- find current subject,

- validate subject,
- find current place,
- validate place,
- converse about current place.

The assistant tries to move through these steps with pre-constructed frame-based sentences with mostly recursive functions that only exits when a frame is filled or a validation is passed to the assistant.

To this triggers are added, these can only be triggered with certain pre-determined words, an example for this is if the user writes a greeting the system will respond with a greeting. Some triggers are only available when the current subject or place is found, thus making the conversation evolve a bit more naturally. In the last step where it is possible to converse with the assistant about the current place the user only interacts with triggers until and ending phrase it passed.

2.2 Data structure

Each subject has a name, tags and a list of places. The tags purpose is to make it possible for the digital assistant to pick up what subject the user needs help with, for example the tags for a restaurant are *drink, eat, resturant* and *food*, so if any of these words are written to the assistant it will be directed into the subject of restaurants. If there are multiple subjects with the same tags the assistant will add an extra step of verification before the validation step where it ask the user to be a bit more specific about what they need help with. Also if the assistant is not able to connect a response to a subject, a specification massage is printed.

The places are constructed in a similar way with name and tags, however added to this is there is some information included about the place currently price-range, opening and closing time.

Adding new subjects and places thus becomes almost trivial task with this design, one just need to look quickly at the previous inserts or the python code for the objects. It is also easy to insert additional information about places by adding variables to the object.

2.3 Example conversations

Now we are going to demonstrate a possible conversations with our digital assistant. The bold text is the digital assistant and the italic text is the user input:

- **Hi my name is mArIa!**
- *Hi mArIa, I would like to find someplace to eat*
- **Nice to meet you!**
- **So you are looking for Restaurants?**
- *Yeah*
- **What kind of Restaurant are you looking for?**
- *A place where I can drink a beer*
- **The Red Lion is a beer and burger place**
- **Does that sound good?**
- *no*

- **What are you looking for besides beer?**
- *I would like to have some fish maybe*
- **Tullen is a fish and meat place**
- **Does that sound good?**
- *Yes*
- **Do you have any more questions about Tullen?**
- *Yes, is it expensive? And when does it close?*
- **Tullen is in the moderate priceclass**
- **Tullen is open from 16 until 1**
- **Do you have any more questions about Tullen?**
- *No thanks, goodbye*
- **Have a nice day, thanks for the chat**

Here we can see the greeting triggering when the user input's "Hi", where the system replied with "Nice to meet you!". Afterwards we can see that a tag was found for the subject of Restaurants and thus a validation question is executed to confirm the current conversation. After the current subject is confirmed the assistant moves on to finding a place, here the user asks for beer and the Red Lion is recommended but the user is not satisfied with the recommendation and thus the systems ask what the user is looking for more then beer to get a new tag from the user. Later Tullen is suggested and accepted by the user, after this the user asks some questions about the current place which triggers the assistant to print out the relevant information asked for.

As we can see the conversation is fairly basic and most of the words that are used isn't even necessary to write since the tags are the only things that matters, thus this conversation generates the same response:

- **Hi my name is mArIa!**
- *Hi eat*
- **Nice to meet you!**
- **So you are looking for Restaurants?**
- *Yeah*
- **What kind of Restaurant are you looking for?**
- *beer*
- **The Red Lion is a beer and burger place**
- **Does that sound good?**
- *no*
- **What are you looking for besides beer?**
- *fish*
- **Tullen is a fish and meat place**

- **Does that sound good?**
- *Yes*
- **Do you have any more questions about Tullen?**
- *expensive close*
- **Tullen is in the moderate priceclass**
- **Tullen is open from 16 until 1**
- **Do you have any more questions about Tullen?**
- *No*
- **Have a nice day, thanks for the chat**

Here only the tags are directly written to the assistant, this does however require insight knowledge about the system.

If there is no tags to be found in a reply the system executes the last function recursively until an tag is found.

- **Hi my name is mArIa!**
- *Good morning!*
- **Could you try that again and be a bit more specific, what can I help you with?**
- *Well sorry, I'm looking for a nice place to shop clothes*
- **So you are looking for Shops?**
- *Hell yeah!*
- **Sorry I didn't get that**
- **So you are looking for Shops?**
- *yes*
- **Filur is a kid and clothes place**
- **Does that sound good?**
- *Hmmm, maybe*
- **Sorry I didn't get that**
- **Filur is a kid and clothes place**
- **Does that sound good?**
- *Well ok then*
- **Do you have any questions about Filur?**
- *No, bye*
- **Have a nice day, thanks for the chat**

As we can see in the conversation above when a tag is not found in an reply the assistant prints that it didn't understand and the repeats the question, it is implemented in this way so that one who uses the assistant can retry their reply until the assistant understands it.

Another thing we can see in this conversation is that since the tag *clothes* was written earlier in the conversation the assistant skipped asking the user for what kind of shop is being searched for, this is due to the fact that the assistant stores all previous replies and looks through them in reverse chronological order when looking for a place to suggest, thus the previously written tag will be used.

2.4 Notes on improvement

Some things that would be nice to implement for improvement is more subjects and places off course, but also more triggers at different stages of the conversation. For example if you are on the subject of restaurants it would be nice to be able to ask what places serves beer and to that printed list be able to choose one by referring to the first or second one. Another nice trigger to have would be to ask for a restaurant that is open until later or opens earlier. So basically adding more triggers to interact with the information we have already implemented.

Another option would be to extend the functionality of the assistant by for example adding a booking system for restaurants, this could be done by adding a boolean variable to the object subject which indicates whether or not it is bookable, if it is, then a whole new set of framed sentences would be possible to trigger.

Apart from the above options, we can also consider implementing graphical user interface for much more user friendly experience and to add personalized messages in the interface. This way it becomes easier to read the responses of FAQ answering bots and this might in turn attract sales into the business. Furthermore, developing an advanced bot involving AI and NLP models allows the agent to learn over time. The accuracy of these models will be high as they will learn to respond like humans do.

3 Refelection & Summary

3.1 Jonatan

3.1.1 Summary of the Lecture: Dialogue systems and question answering

- There are two main types of dialogue systems, chat-bots and digital assistants. Chat-bots are for general conversations and digital assistants are more for specific tasks.
- We got to know a bit of the foundation of speech, this is necessary so that we know how to model it.
- Later we got to learn a bit of the history of dialogue systems, they started as many other AI-applications in the 1960s with rule-based system. ELIZA a therapist was the first one discussed.
- Corpus based chat-bots are trained on real conversation data instead, instead of rules it respons with the most likely response based on the corpus.
- In more recent times autoencoders has been used to generate dialogue.
- Natural language processing can be added to a dialogue system to make it more generally applicable.
- Then we talked a bit more about frame-based systems that we covered in this week assignment.

3.1.2 Summary of the Follow-up Lecture: Game Playing Systems

- Talked how the domain changes when moving into a more complicated domain (GO), where a brute force solution is not a valid option.
- We where given a recap in how MCTS and policies work.
- To find a policy in the more complicated domain it is better to train on historical pro player data. Later they also improved the policy by reinforcement learning, where the AI plays against itself to fine tune the weights.
- There is also a value network included in the alphaGo system, it is a convulotional neural network that inputs the board state and returns the value of the state i.e. how preferable it is for the player.
- Another game playing system discussed about where OpenAI FIVE, beat human players in DotA, it is a way harder problem.

3.1.3 Reflection on the previous module: Game Playing Systems

In the previous module we learned about game playing systems, it was a very interesting module. We got to know some history of gameplaying systems and later read more about Alpha Go that beat the South Korean Go champion. Alpha Go was a real data engineering masterpiece that combined Monte-Carlo Tree Search with Neural Networks and reinforcement learning to create a game playing system above human capability. Laten in the implementation part we got to try out to implement the Monte-Carlo Tree Search algorithm on tic-tac-toe, it was quite a simple game so implementing it wasn't to hard and it performed very well even with few iterations. However it was quite boring to play against it since it is so easy that the game ends up in a draw when playing tic-tac-toe.

3.2 Ankita Rahavachari

3.2.1 Summary of the Lecture: Dialogue systems and question answering

- Chatbots: They are used for arbitrary conversations, Digital Assistance: They are used for tasks. The modern systems uses a combination of rules and ML algorithms while the earlier systems where rule-based models
- Speech acts: Constatives: examples - answering, claiming & confirming ; Directives: examples - advising, asking, & forbidding ; Commissives: examples - promising, planning & opposing ; Acknowledgments: examples - thanking, accepting, apologizing
- Types of Initiative in a conversation: User, system and mixed initiative
- In the keyword based chatbots, there is higher rank for specific keyword and prefers response based on most specific keyword
- These chatbots include model of mental state that influences the conversation
- Corpus based chatbots: It is based on a very large set of real conversation data. Responses are generated based on user's last turn
- IR- based chatbots: These chatbots return response to most similar user turn
- Architecture of Dialogue system: Speech Recognition (statistical/NN-based), Natural Language Understanding (rule based or ML based), Dialogue Manager (rule based), Task Manager, Natural Language generation(template based), Text-to-Speech Synthesis
- NLU involves classifying the domain, determining the intent and extracting the relevant information
- Dialogue Management is done to avoid user inputting all the information in one statement

- Frame based dialogue management refers to a particular common situation. The words are understood in a relation to the mental frame of reference. Most of the task based systems uses frames
- Dialogue management with a frame based and with a dialog state: Keeps different aspects separate and there can be multiple frames that the system moves between
- Dialogue Policies determine next action based on entire previous dialogue
- Rule based models are useful for DM and NLG while the ML based can be useful in ASR and NLU

3.2.2 Reflection on the previous module: Game Playing Systems

In the reading part, the paper gave a brief overview of the significant improvements made in Alpha Go. It was interesting to know that combination of supervised learning and reinforcement learning yielded an improvised output. Implementing MCTS algorithm for the X & O game was insightful as well as exciting. The four stages select, expand, simulate and backpropagate clearly defined the working of the system.