APPLICATION OF A LARGE LANGUAGE MODEL IN QUESTION ANSWERING

Fernando González Sanz
Diego Rodríguez de Roa
International Students, Course 23-24
Knowledge Engineering

INDEX:

1. INTRODUCTION

This project consists of the application of a large language model to answer questions based on a given ontology. In general terms, an ontology is a formal representation of a set of concepts within a domain and their relationships. On the other hand, a large language model is an AI model designed to understand and generate natural text in human language.

The ontology is created by using Chat GPT 4. Later, we used a verbalizer (just a Python file where we transformed from .owl to text). With this ontology in text format, we have used it as an input to a large language model (llama) for answering questions related to the knowledge base.

2. GOALS

The goal of this project is to connect our ontology to a large language model (LLM) to answer simple questions formulated in natural language related to our ontology.

3. ONTOLOGY CREATION

To create our ontology, we have used the tool chat GPT 4. We decided to make the ontology about music genres and artists.

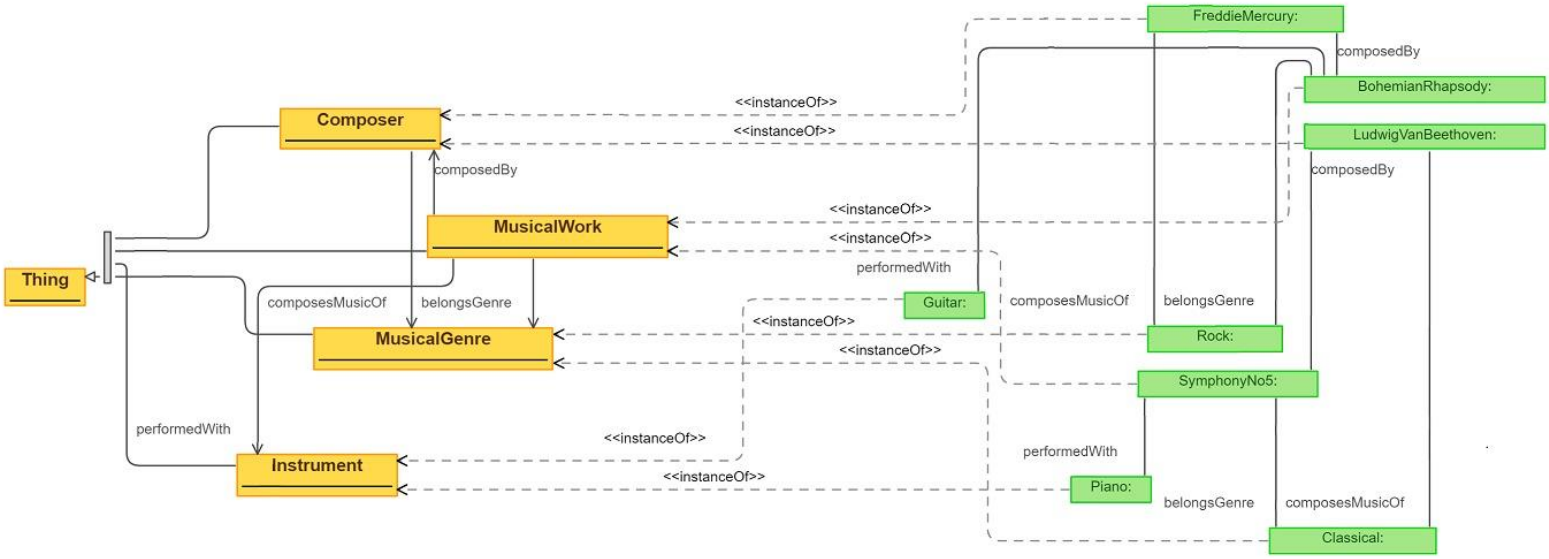
Our first decision was to make the ontology about the parts of a song, developing classes like melody or chord. However, when we tried to instantiate some instances for these classes and relationships, it became an ontology that was hard to understand.

At this point, we decided to try an ontology based on another classes, also related to music, but at a higher level. We had the idea to include the following classes: composer, instrument, work, and genre. The idea was to develop relationships of the type:

- Composer composes music of a genre of music.
- Musical work belongs to a genre of music.
- Musical work is composed by a composer.
- Musical work is performed by an instrument.

With this information, we used chat GPT as a tool to develop our ontology. The result was a .owl file (attached in the GitHub project)

We used the tool http://owlgred.lumii.lv/ to visualize our ontology in an RDF schema. The result was the following:



Regarding the schema, we can see that this ontology is quite simple, but useful for our goals.

The instances generated are also useful. We have two “groups” of instances. Number one is Freddy Mercury, who composed “Bohemian Rhapsody”, and whose musical genre is rock. “Bohemian Rhapsody” is performed by the guitar. The second group of instances is Ludwig van Beethoven, who composed “Symphony nº5”, and whose musical genre is classical. “Symphony nº5” is performed by the piano.

With the ontology created we started thinking about how to connect it to a large language model.

4. VERBALIZATION OF THE ONTOLOGY

The first thing to think about is that large language models take text as input, not an .owl. So, we need to transform the ontology from an .owl file to plain text that the model can take as input.

For this task, we created a function called “verbalizer”. This function takes as input the rdf ontology as a graph and starts making string characters for every class, property (relationship) and instance. The result is a variable of type string character that contains our ontology verbalized.

After loading the ontology, we define a prefix dictionary to translate from RDF URI prefix into short names, such as “rdf” or “owl”. Later, we define a function called “local name” that replaces the full prefix with the local names named before. After this preparation, we are ready to transform the classes, properties, and instances into readable lists. The result is the ontology described in natural language:

```
The ontology contains the following classes:
- :MusicalWork
- :Composer
- :Instrument
- :MusicalGenre

The ontology contains the following properties:
- :composedBy with domain :MusicalWork and range :Composer
- :performedWith with domain :MusicalWork and range :Instrument
- :belongsGenre with domain :MusicalWork and range :MusicalGenre
- :composesMusicOf with domain :Composer and range :MusicalGenre

The ontology contains the following instances:
- :composesMusicOf is an instance of owl:ObjectProperty with the following relationships: rdf:type owl:ObjectProperty, rdfs:domain :Composer, rdfs:range :MusicalGenre
- :SymphonyNo5 is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :LudwigVanBeethoven, :belongsGenre :Classical, :performedWith :Piano
- :BohemianRhapsody is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :FreddieMercury, :belongsGenre :Rock, :performedWith :Guitar
- :Piano is an instance of :Instrument with the following relationships: rdf:type :Instrument
- :belongsGenre is an instance of owl:ObjectProperty with the following relationships: rdf:type owl:ObjectProperty, rdfs:domain :MusicalWork, rdfs:range :MusicalGenre
- :SymphonyNo5 is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :LudwigVanBeethoven, :belongsGenre :Classical, :performedWith :Piano
- :BohemianRhapsody is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :FreddieMercury, :belongsGenre :Rock, :performedWith :Guitar
- :BohemianRhapsody is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :FreddieMercury, :belongsGenre :Rock, :performedWith :Guitar
- :performedWith is an instance of owl:ObjectProperty with the following relationships: rdf:type owl:ObjectProperty, rdfs:domain :MusicalWork, rdfs:range :Instrument
- None is an instance of owl:Ontology with the following relationships: rdf:type owl:Ontology
- :FreddieMercury is an instance of :Composer with the following relationships: rdf:type :Composer, :composesMusicOf :Rock
- :composesMusicOf is an instance of owl:ObjectProperty with the following relationships: rdf:type owl:ObjectProperty, rdfs:domain :Composer, rdfs:range :MusicalGenre
...
- :BohemianRhapsody is an instance of :MusicalWork with the following relationships: rdf:type :MusicalWork, :composedBy :FreddieMercury, :belongsGenre :Rock, :performedWith :Guitar
- :LudwigVanBeethoven is an instance of :Composer with the following relationships: rdf:type :Composer, :composesMusicOf :Classical
- :Composer is an instance of owl:Class with the following relationships: rdf:type owl:Class

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

5. LARGE LANGUAGE MODEL SELECTION

After testing some models, we decided to use “Meta-Llama-3-8B-Instruct”.

This model is a large language model developed by meta and designed for processing natural language. It stands out for its capacity to generate text from relevant entries.

This model contains approximately 8.000 million parameters, this elevated number helps us generate text with very high precision. However, this precision has a price, Llama requires a high computational capacity. Furthermore, it requires permission from the authors of this repository to use the model.

6. CONNECTION WITH THE ONTOLOGY

As we have said before, “llama” requires permission from the authors. This permission is given to an account of “hugging face”. “Hugging face” is an online repository that offers several types of large language models, such as question answering or text generation. These models are pre-trained, so their use is very simple. To use the model, we need to log in with our access token generated by hugging face.

The code developed by hugging face (and added to our project) uses the library “transformers” from hugging face to generate text with the language model.

First of all, we initialize the pipeline using the function “transformers.pipeline” of Hugging Face. We use the model assigned before to the variable model_id: “meta-llama/Meta-Llama-3-8B-Instruct"

Later, we define a list of messages that we want to pass to the model. Messages simulate a conversation, with roles like “system” and “user”. The initial instruction is defined in “verb” (containing the verbalized ontology) and the question we want to ask.

The variable output contains the text generated, with parameters like the maximum number of tokens (max_num_tokens) to generate, or if we want to generate the response in a random way or deterministic way (do_sample), and the temperature parameter (temperature) that allow us to adjust the randomness of the text generation process.

Finally, we show the output on the screen with the print function.

7. RESULTS OF THE LARGE LANGUAGE MODEL

Our first question to the model was the following “Who composed bohemian Rhapsody?”

We decided to test the model with a competency question developed by us to check if the model worked. After waiting a long time, we received this:

```
⋯  Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained
   Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
   {'role': 'assistant', 'content': 'According to the given ontology, Freddie Mercury composed Bohemian Rhapsody.'}
```

Observing the ontology, we can see that the response is correct.

Once we had checked that the model worked, we decided to ask for competency questions, so we asked the model the following: “tell me 3 competency questions about this ontology”.

The output received was the following:

Here are three competency questions about this ontology:

- Who composed the musical work "Symphony No. 5"?
  This competency question is related to the property: composedBy, which connects a: MusicalWork to a: Composer.
  The answer to this question would be: LudwigVanBeethoven.
- What instrument is typically used to perform the musical work "Bohemian Rhapsody"?
  This competency question is related to the property: performedWith, which connects a: MusicalWork to an: Instrument.
  The answer to this question would be Guitar
- What genre of music does the composer: FreddieMercury typically compose?
  This competency question is related to the property: composesMusicOf, which connects a: Composer to a: MusicalGenre.
  The answer to this question would be Rock.

With these competency questions generated we tested them in the model by asking for them. For asking for them we have developed a function that contains the connection between the ontology and the large language model. Also, we have put the questions in different .txt files, for easier comprehension.

Here are the code and the outputs for the different questions:

Question 1:

```
with open('competency-questions/CQ1.txt','r',encoding='utf-8') as file:
    question1=file.read()
response=questionAnswering(verb,question1)
print(question1)
print(response)
```

```
'According to the ontology, the musical work "Symphony No. 5" was composed by Ludwig Van Beethoven.'}
```

Question 2:

```
with open('competency-questions/CQ2.txt','r',encoding='utf-8') as file:
    question2=file.read()
response=questionAnswering(verb,question2)
print(question2)
print(response)
```

```
'According to the ontology, the musical work "Bohemian Rhapsody" is typically performed with a "Guitar".'}
```

Question 3:

```
with open('competency-questions/CQ3.txt','r',encoding='utf-8') as file:
    question3=file.read()
response=questionAnswering(verb,question3)
print(question3)
print(response)
```

```
:FreddieMercury typically composes music of the genre :Rock.
```

As shown by the answers to the competency questions, the responses match our expectations. So, our model performs well on these example competency questions.

8. PROBLEMS WE HAVE DEALED WITH

Our first problem was the model selection. We tried with some models until we found “llama”.

One model we tried with was “dynamic tinybert” (https://huggingface.co/Intel/dynamic_tinybert) this model was not useful for us because it didn´t accept the ontology because it was very large.

Another model we tried with was “Qwen2-72B-Instruct” (https://huggingface.co/Qwen/Qwen2-72B-Instruct) this model didn´t work for us because it required “CUDA”, a platform needed for using the GPU. We were not able to execute this model. We tried to install NVIDIA drivers, and even with this installation it still didn´t work.

Once we had decided to use “llama” it required us to install “git” and add it to the environmental path. We tried to do it on the control panel of windows, adding this directory to the path, but it didn´t work (it appears something different to Files\Git\bin, what we wanted). Finally, we fixed it adding it manually on python, writing the directory we would like to add.

9. CONCLUSSIONS

Our ontology is not large. We could have done one larger, but when we started executing the models and see how much time it took, we decided to keep it like this.

This model is so slow, it takes more than 150 minutes to solve the questions (the first time to execute it, it required the download of the files, and it takes 500 minutes). The explanation of the long time it takes to execute, could be for the low computational capacity of our computers. One solution to this problem, could be using gpu instead of cpu, but as we have said before, it required “CUDA” and we unable to use it in the good way.

Finally, we would like to expose that this project has been a challenge for us, since we have not previously studied anything related to knowledge representation. We hope that our project fits the requirements of the project, despite the fact that it is not perfect.