Evaluation Event Final - *swift-comet*

1st Gianmarco Albano(*gianmarco.albano*)

2nd Abu Bakr Rahman Shaik (*abubakrrahman.shaik*)

December 9, 2024

1 Introduction

The agent is designed to answer questions related to movies based on a knowledge graph, multimedia, and crowdsource data. It extracts entities and predicates from user questions using NER and similarity-matching techniques, even if there are typos. It then generates an appropriate response and retrieves data from multimedia or crowd data, if requested.

Some key features of our agent are:

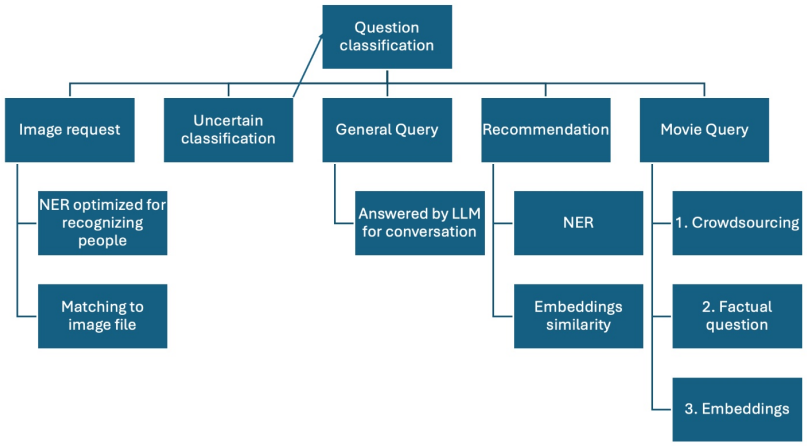
- Handling synonyms and potential misspellings using edit distance.
- Extracting relations and predicates from complex questions.
- A generalised content-based recommendation system using content derived from user questions.
- The ability to handle multiple questions within a single query.
- The ability to ask for clarifications if anything is unclear (such as, when there are multiple items with the same name, e.g., Movie, Game or TV show).
- Generating simple, human-like responses when an answer cannot be found.
- ability to have a dynamical conversation while mantaining some memory of previous questions

2 Capabilities

Below is an overview of our chatbot agent, depicting its main components, their interactions, and the flow of inputs and outputs:

2.1 Agent Overview Diagram

Below is an overview of our chatbot agent, depicting its main components, their interactions, and the flow of inputs and outputs:



- **Question Classifier:** Classifies incoming questions into five types using the zero-shot classification model `facebook/bart-large-mnli`.
- **Named Entity Recognition (NER):** Extracts entities and relations using `Babelscape/wikineural-multilingual-ner`.
- **Crowdsourcing Manager:** Handles validation and majority voting for crowdsourced answers.
- **Embedding Engine:** Computes entity and predicate embeddings using pre-trained knowledge graph embeddings.
- **Multimedia Manager:** Retrieves images from IMDb data.
- **Response Generator:** Synthesizes answers and generates paraphrased responses using `PegasusForConditionalGeneration`.

2.2 Factual Questions

Our agent answers factual questions in three steps:

1. **Question Classification:** We classify the question using the zero-shot classification pipeline.
2. **Entity and Predicate Extraction:** We extract relevant entities and predicates using NER and edit-distance matching.
3. **Knowledge Graph Querying:** We construct SPARQL queries to retrieve answers from the Wikidata-based knowledge graph.

2.3 Embedding Questions

Our agent resolves embedding-based questions through:

1. **Entity and Predicate Embedding Retrieval:** Extracts vector representations from pre-trained embeddings.
2. **Similarity Matching:** Computes cosine similarity between query embeddings and entities in the knowledge graph.
3. **Answer Retrieval:** Returns the most similar entities based on distance rankings.

2.4 Multimedia Questions

Multimedia queries are handled through:

1. **Entity Recognition:** Identifies named entities associated with movies or actors.
2. **IMDb ID Retrieval:** Finds the corresponding IMDb ID using the knowledge graph.
3. **Image Retrieval:** Searches a preloaded IMDb image dataset and returns the relevant image.

2.5 Recommendation Questions

The agent provides recommendations by:

1. **Entity Extraction:** Identifies liked movies and people using NER.
2. **Associated Movies Retrieval:** Gathers related movies based on people's careers.
3. **Recommendation Generation:** Uses embedding-based similarity scoring to recommend top movies.

2.6 Crowdsourcing Questions

For crowdsourced questions, the agent follows:

1. **Worker Validation:** Filters malicious workers based on task completion time and approval rates.
2. **Majority Voting:** Performs majority voting on crowdsourced answers.
3. **Confidence Scoring:** Uses Fleiss' Kappa to assess inter-rater reliability and returns validated answers.

3 Adopted Methods

In this section, we describe the key methods, techniques, tools, and packages employed in developing our chatbot agent, along with the rationale for their selection.

- **Zero-Shot Classification Model** We used the `facebook/bart-large-mnli`[1] model for zero-shot classification to determine the intent of the user's question. This model supports multi-label classification without task-specific fine-tuning, making it ideal for dynamic query classification.
- **Named Entity Recognition (NER) Model**
  We adopted the `Babelscape/wikineural-multilingual-ner`[2] model for multilingual entity recognition. It supports fine-grained entity extraction, enabling accurate identification of names, movies, and predicates.
- **Pegasus for Paraphrasing** [?]
  We used `PegasusForConditionalGeneration`[3] for paraphrasing generated responses. This improves user experience by providing varied and natural-sounding replies.
- **SpaCy**
  We used `SpaCy`[4] and `en core web md`[5] for advanced natural language processing tasks such as tokenization, stop-word removal, and similarity computation.

3.1 Factual Questions

The agent answers factual questions related to movies using pre-defined patterns and SPARQL queries. It can identify entities and relationships mentioned in the questions which are structured as one of the patterns, and retrieve relevant information from the knowledge graph.

3.2 Embedding Questions

For questions with implicit or unknown relationships, The agent uses the pre-trained embeddings. It converts entities and predicates into vectors, combines them to create a query embedding and searches for the most semantically similar entities in the embeddings dataset to find potential answers.

[1] `https://huggingface.co/facebook/bart-large-mnli`
[2] `https://huggingface.co/Babelscape/wikineural-multilingual-ner`
[3] `https://huggingface.co/tuner007/pegasus_paraphrase`
[4] `https://spacy.io`
[5] `https://huggingface.co/spacy/en_core_web_md`

3.3 Recommendation Questions

The agent uses the cosine similarity of the embeddings of the extracted entities for the recommendations. Using NER, the entities in the questions are extracted, and it then suggests movies based on similarity to liked entities (movies or people).

3.4 Crowd Source Questions

The agent first filters the malicious workers using the 'LifetimeApprovalRate' threshold of 50 and 'WorkTimeInSeconds' threshold of 35. Then the merge_crowdsourced_data function integrates this crowd data into the KG, then the answers are searched from the responses of valid_workers, using majority_vote to resolve conflicting responses. Inter-rater agreement is measured using Fleiss' kappa score and displayed with the answers retrieved from KG and crowd_data, along with total valid votes.

4 Examples

4.1 Factual

Question: Who is the director of Good Will Hunting?
Answer: Gus Van Sant

4.2 Embeddings

Question: Who is the screenwriter of The Masked Gang: Cyprus?
Answer: I think that the screenwriter of The Masked Gang: Cyprus might be Cengiz Küçükayvaz, Murat Aslan or maybe Melih Ekener. Not sure which one tbh

4.3 Recommendation

Question: What films should I watch if I enjoyed The Godfather, Scarface, and Pulp Fiction?
Answer: Recommended Movies: The Godfather Part II, The Godfather Part III, Goodfellas, Vigilante, Apocalypse Now

4.4 CrowdSource

Question: Who is the executive producer of X-Men: First Class?
Answer: The answer is Sheryl Lee Ralph. [Crowd, inter-rater agreement: 0.199, The answer distribution for this task was 2 support votes, 1 reject votes]

4.5 Multimedia

Question: Show me a picture of Halle Berry
Answer: I found an Image for Halle Berry: image:0353/rm3257480192

5 Additional Features

Our chatbot agent incorporates advanced features aimed at creating a dynamic, human-like interaction experience by leveraging various language models (LLMs) and implementing a contextual conversation flow.

5.1 Dynamic Conversation Management

To maintain a fluid conversation, our agent supports dynamic question handling through a memory mechanism. When the intent classification yields uncertain results, the agent prompts the user for clarification while storing context for follow-up interactions. This ensures that even ambiguous inputs lead to meaningful exchanges. For example:

```
question = "Given that I liked Tom Cruise in Mission: Impossible and Top Gun, what similar movies should I watch?"
print(bot.answer_question(question))
```

```
I'm not entirely sure, but are you asking about movie query or recommendation query? Sorry but i get confused if i need to handle multiple st
uff at once hahaha
```

```
question = "i think it is more of a recommendation query"
bot.answer_question(question)
```

```
'Recommended Movies: Jack Reacher, Mission: Impossible 2, Lions for Lambs, The Mummy, Mission: Impossible – Ghost Protocol'
```

5.2 Multi-Model Integration for Human-Like Responses

We integrated different LLMs tailored for specific tasks:

- **BART for Intent Classification:** We used the BART model to classify user intents with high accuracy due to its strong contextual understanding capabilities.
- **Wikineural NER for Entity Recognition:** Named Entity Recognition (NER) from Wikineural ensures robust entity extraction for factual and embedding-based queries.
- **Pegasus for Response Paraphrasing:** To generate diverse and human-like responses, we integrated the Pegasus model, which creates paraphrased answers dynamically.

5.3 Crowdsourcing for Reliable Answers

We developed a reliable crowdsourcing integration. Workers are filtered based on approval rate and task completion time. The majority vote mechanism ensures answer reliability, complemented by Fleiss' Kappa for inter-rater agreement evaluation.

5.4 Human-Like Error Handling

When the chatbot encounters an unknown or complex query, it gracefully acknowledges its limitations with emotionally aware responses, making conversations feel natural and less mechanical.

By combining multi-model integration, dynamic memory management, and thoughtful user interaction design, our chatbot effectively balances technical accuracy and engaging human-like communication.

6 Conclusions

This agent demonstrates the ability to answer various questions related to movies using data from knowledge graph, embeddings, and crowd_data. The agent employs a multi-faceted approach: using SPARQL queries for factual questions, pre-trained embeddings for semantic understanding, and querying image databases for multimedia responses. Upon implementing all the functionalities utilizing various packages and libraries such as BERT Classification Model, wikineural-NER, SpaCy, PegasusForConditionalGeneration3, the agent has demonstrated good performance by generating quality answers which actually make sense. The work was divided among us equally, and if time was not a constraint, we would have also implemented an 'explanation generation module' to explain the reasoning behind answers which would have helped the evaluating student to make sense of the answers