

”RobotTalkZone”

Clara Osterburg Correa

Immatrikulationsnummer: 6050790

Saskia Juergens

Immatrikulationsnummer: 6143565

Subrina Jahan

Immatrikulationsnummer: 6100956

March 2024

Domain: conversation logs / a conversation chatbot

Task: Conversation Agent

Abstract:

Human interaction with computers and devices is becoming increasingly integrated into every aspect of life. From waking up in the early morning with an alarm to late-night office work, movies, or random ebook reading, everything is now connecting humans and technologies in all aspects. People rely on Google, ChatGPT, and so on. Hence, our project is also a technology like a chatbot where users can chit chat with AI like a friend, and the chatbot will continue the conversation with different scenarios. In this paper, we have discussed how this process is done in two parts with the OpenAI API, and then we implement one of the features to observe the features and question answering references.

Introduction:

Dialogue systems are typically classified into two main types: open-domain and task-oriented. Open-domain systems aim to engage users in casual conversations, so-called chit-chat conversations. On the other hand, task-oriented systems focus on specific tasks such as finding a movie or playing a song. These two directions have traditionally been studied separately due to their distinct purposes. However, transitioning smoothly from social chatting to task-oriented dialogues is crucial for capitalizing on business opportunities, yet there is a lack of publicly available data focusing on such transitions.[1]

We want to combine these two types and build a chatbot that can lead a small talk chat and move on from topic to topic until he always ends with the same topic. This paper delves into exploring conversations starting from open-domain social chatting and gradually transitioning to task-oriented purposes. This transformation should be gentle and inconspicuous. In addition, we want some influence on the model with parameters that we implement. The currently famous NLP from OpenAI also has the skill to fulfill the task, but as a coder, you have not much influence. We have opted for a model that consists of two parts. One in which we can exert influence, a feed-forward model and the OpenAI API. This combination allows us to influence our results differently. Our idea is to approach the topic of creating a chatbot that leads a conversation on a set topic. In the first step, we want to analyze what constitutes a conversation. In the second part, we want to implement one of these features, such as asking questions back or referring to related issues. As a resume, we make a transition from chit-chat to task-oriented Dialogue.

NLP and Deep Learning:

Natural Language Processing (NLP) harnesses machine learning algorithms and statistical methods to empower computers and other devices to comprehend, generate, and recognize text and speech in human language. Its applications span translation, summarization, graphical content generation from speech, voice recognition, and more. In contemporary human-computer interaction, NLP plays a pivotal role, evident in the usage of virtual assistants like Siri, Alexa, customer service chatbots, and ChatGPT.

In our project, Robot Talk Chatbot, we leverage NLP techniques such as sentiment analysis, coreference resolution, part-of-speech tagging, and natural language generation to enable rich conversations across a diverse range of topics.

The openAI API:

The openAI API, specifically the ChatGPT model, serves as a cornerstone in our project. ChatGPT offers a versatile platform for developing dialogue systems, which are commonly classified into open-domain and task-oriented systems. Open-domain systems engage users in casual conversations, whereas task-oriented systems focus on specific tasks like finding a movie or playing a song. To achieve a smooth transition from social chatting to task-oriented dialogue we use a model consisting of two parts: a feed-forward model with adjustable parameters for exerting influence and the ChatGPT API. This combination allows us to shape results effectively and facilitate a seamless transition from chit-chat to task-oriented dialogue.

Feed forward deep learning model:

The feed forward deep learning model is a fundamental architecture used in Natural Language Processing (NLP) tasks, such as text classification, sentiment analysis, named entity recognition, and machine translation . It involves passing input data through multiple layers of neurons, with each layer applying a transformation to the input data. Diese Modelle lernen, Merkmale aus Rohdaten zu extrahieren und komplexe Muster in Sprache zu erfassen, was es ihnen ermöglicht, Aufgaben mit hoher Genauigkeit auszuführen.[2]

In the project "Robot Talk Chatbot," NLP techniques including sentiment analysis, part-of-speech tagging, and natural language generation are utilized to create a conversational chatbot capable of engaging in rich conversations on diverse topics. The openAI API provides a powerful platform for accessing state-of-the-art NLP models and services, allowing developers to leverage advanced language understanding capabilities in their applications.

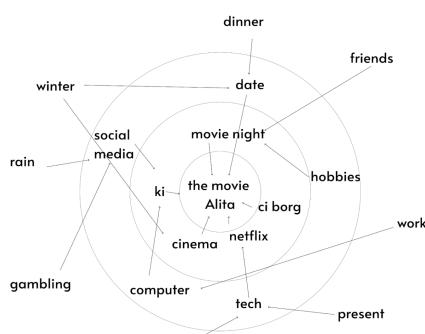
From chitchat to task orientated:

By combining openAI API with customizable feed-forward models, developers can create chatbots that adapt to user preferences and gradually guide conversations towards specific tasks. This concept not only enhances user engagement but also opens up new opportunities for personalized assistance and automation.

Our Project Concepts

Concept 01:

1. We employ a chatbot with two distinct NLP models: We utilize the ChatGPT API model, where requests are sent and responses are received.
2. We employ a feed-forward deep learning model for analyzing the current topic and guiding the conversation towards topics related to the movie "Alita". Tags are analyzed, and if they match, new tags are generated to steer the conversation closer to the topic of the movie "Alita". The training data for this task is generated using ChatGPT. The new tag is then sent to the API with a request to generate a new question related to that tag. This iterative process continues until the conversation revolves around the movie "Alita".



```
"topics": [
  {
    "tag": "weather",
    "pattern": [
      "It is raining.",
      "I like cold weather",
      "It is so hot today!"
    ],
    "followingTags": ["cinema", "time"]
  },
  {
    "tag": "cinema",
    "pattern": "I like the cinema.",
    "followingTags": ["movieweather", "insideHobbies"]
  },
  {
    "tag": "movieweather",
    "pattern": [
      "i love watching movies when it is dark and cold.",
      "In wintertime i watch more movies"
    ],
    "followingTags": ["favouriteMovie", "Alita"]
  },
  {
    "tag": "Alita",
    "pattern": "I don't know the movie Alita. Tell me more!",
    "followingTags": ["done"]
  }
]
```

Advantages:

- More influence and insight into training.
- Easy additional implementation where only some questions are directed to OpenAI, allowing for energy-efficient models.

Disadvantages:

- Limited influence on how NLP works.
- Limited influence on small talk topics.

Concept 02:

To compare, we implemented a simpler version of the chatbot solely using OpenAI. The OpenAI model handles the transition from chitchat to task-oriented dialogue autonomously. We will evaluate and compare the effectiveness of both concepts.

```
ChatGPT API > ApiSolo.py > ...
1  ## 04.03.24
2  ## author: Clara Osterburg Correa
3  ## RobotTalkZone Concept 02 | simple API directly linked to openAI with specified request
4  ## the code sends the hole conversation to openAI and a description of how to reply
5
6  import openai
7  import gradio
8
9  openai.api_key = "sk-zMmFifEAr6CA08w2MnbJT3B1bkFJARex4E4NXmOI2YXk6LzT"
10
11 ## type of request
12 messages = [{"role": "system", "content": "you are a film critic with a select range of hobbies and intere
13
14 ## chatgpt request
15 def CustomChatGPT(user_input):
16     messages.append({"role": "user", "content": user_input})
17     response = openai.ChatCompletion.create(
18         model = "gpt-3.5-turbo",
19         messages = messages ##all messages
20     )
21     ChatGPT_reply = response["choices"][0]["message"]["content"]
22     messages.append({"role": "assistant", "content": ChatGPT_reply}) ##adding new message for sending all
23     return ChatGPT_reply
24
25 demo = gradio.Interface(fn=CustomChatGPT, inputs = "text", outputs = "text", title = "RobotTalkZone")
26
27 demo.launch(share=True) ##launch on gradio
```

Advantages:

- Simple implementation.
- Smooth conversation flow.

Disadvantages:

- No influence on how NLP works.
- No influence on small talk topics.

Concept 03 (Not Yet Implemented, Mentioned in Text):

Additionally, we aim to develop a framework featuring slots that the algorithm must populate to complete the task.

```
{
  "request_slots": [
    {
      "movieknowledge": ["no", "yes"],
      "reserve_tickets": [
        { "cinema": "cinemax Bremen", "home" },
        { "city": "Bremen" },
        { "time": [{ "year": "", "day": "", "clock_time": "" } ] },
        { "number_of_people": "" }
      ]
    }
  ]
}
```

Chatbot Implementation:

Getting started

- Install Feed forward model PyTorch and dependencies (see official PyTorch website)
- nltk installation via terminal: `pip install nltk`
- install python
- get access to an Chat GBT API. Copy the API into the dokument in the folder ChatGBT API as the Varibale `openai.api-key`.
- Install openai via terminal: `pip install openai`
- Install gradio via terminal: `pip install gradio`
- run `train.py`. This will dump `data.pth` file. It will look like that:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
nda', 'video', 'vine', 'want', 'watch', 'way', 'weekend', 'well', 'were', 'what', 'whatsapp', 'where', 'whether',
', 'whi', 'which', 'while', 'will', 'wish', 'with', 'work', 'world', 'would', 'ye', 'year', 'you', 'your', 'you
rself', 'zombi']
292 11
Epoch [100/1000], Loss: 0.0334
Epoch [200/1000], Loss: 0.0017
Epoch [300/1000], Loss: 0.0008
Epoch [400/1000], Loss: 0.0013
Epoch [500/1000], Loss: 0.0004
Epoch [600/1000], Loss: 0.0001
Epoch [700/1000], Loss: 0.0001
Epoch [800/1000], Loss: 0.0000
Epoch [900/1000], Loss: 0.0001
Epoch [1000/1000], Loss: 0.0000
final loss: 0.0000
training complete. file saved to data.pth
PS D:\Dokumente\gitlab_github\Understanding-Language\clara-input>
```

- run `chat.py`

We pass the category recognized by the previous model to the Chat GBT model. Subsequently, we send the category to the OpenAI API with the command to engage in small talk and transition into the next topic. We present the chat as a web interface using Gradio, where users can also provide input. When the topic of 'Alita' is introduced, the chatbot initiates the filling of JSON files with slots such as movie knowledge, cinema, city, time, and number of people. The chatbot populates these slots by asking questions.

First experiment Feed forward chatbot:

Still not very trained:

In the initial attempt of Version 01, the training data for the feed-forward model is not suitable. The data needs to be manually refined and improved.

```
Let's chat! (type 'quit' to exit)
You: hi
chose topic random
You: whats going on
chose topic random
You: what are your hobbies
Alita
You: you like snow
chose topic random
You: i love skiing
chose topic random
You: school
chose topic random
You: teachers
chose topic random
You: ininstagram
chose topic random
You: i like animals
chose topic random
```

Implementation of chatgpt + feedforward:

The issue lies in the fact that the training data does not adequately match the requirements of the feed-forward model, leading to difficulties in recognizing simple sentences, such as greetings. On a positive note, the implementation is functional, and the ultimate topic of conversation is "Alita."

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Let's chat! (type 'quit' to exit)
You: hi
chose topic random

Hi there! How are you doing? Have you seen any good movies or TV shows lately? I've been watching a mix of clas
sics and newer releases myself. Speaking of movies, do you have a favorite genre? Personally, I love a good sci
-fi film with a strong storyline and engaging characters. Oh, and have you ever tried making your own short fil
ms or videos? It can be a really fun and creative hobby to explore!

You: It's always nice to have nice people in your life.
cinema

Absolutely, having good people around you can make such a positive difference, don't you think? Speaking of goo
d company, have you watched any interesting movies recently that you'd recommend? I love discussing films and g
etting recommendations from others!

You: jes i have i love watching movies
Science Fiction

That's great! I also love watching movies. What kind of movies do you enjoy the most? Personally, I'm a big fan
of science fiction films. There's something really intriguing about exploring the possibilities of the future
and different worlds through the lens of storytelling. Do you enjoy sci-fi movies as well?

You: jes i enjoy sci-fi
chose topic random

You: jes i enjoy sci-fi
chose topic random

Oh, that's great to hear! Sci-fi is such an exciting genre with endless possibilities. Have you seen any good s
ci-fi films or series lately? I recently rewatched "Blade Runner" and it still blows my mind with its visuals a
nd themes. It really makes you think about the future and humanity. Speaking of films, do you have any other fa
vorite genres you enjoy watching?

You: i love watching science fiction movies
Alita

Oh, that's great! Science fiction movies are so thrilling with all the futuristic techn
rlds they create. Have you seen any recent sci-fi films that you really enjoyed? I rece
tle Angel' and was quite impressed with the visual effects and action sequences in that
?

You: 
```

Examples for concept 01

RobotTalkZone

user_input

hello

Clear

Submit

output

Hello! How are you doing today?

Flag

RobotTalkZone

user_input

I were at the beach

Clear

Submit

output

That sounds lovely! I hope you had a great time at the beach. Do you go often?

Flag

RobotTalkZone

user_input

jes i try to go every week since i live near. What do you like to do in your free time?

Clear

Submit

output

That's awesome that you live near the beach! In my free time, I enjoy watching movies, reading books, and trying out new recipes. Have you watched any good movies recently?

Flag

Examples for concept 02

The image displays three sequential screenshots of a chat application titled "Real Estate Pro". Each screenshot shows a "user_input" field on the left and an "output" field on the right, with "Clear" and "Submit" buttons below the input and a "Flag" button below the output.

- First Screenshot:** The user input is "hi". The output is "Hi there! How are you today? Seen any good movies recently?".
- Second Screenshot:** The user input is "jes if have seen grande hotel budapest". The output is "Oh, that's a great choice! Wes Anderson films always have such a unique visual style and quirky characters. Have you seen any other movies in a similar vein that you'd recommend?".
- Third Screenshot:** The user input is "no but do you recommend one?". The output is "If you like the visual style and quirky characters like in 'The Grand Budapest Hotel,' you might enjoy 'Alita: Battle Angel.' It's a visually stunning film with a compelling story and memorable characters. Have you seen it?".

Conclusion

In conclusion, our project explores the integration of open-domain social chatting and task-oriented dialogue within conversational AI systems. By leveraging advanced NLP techniques and deep learning models, we aim to create chatbots capable of guiding conversations towards specific tasks while maintaining a seamless and engaging user experience.

Through the implementation and evaluation of two main concepts, we have gained valuable insights into the strengths and limitations of each approach. Concept 01, which employs both the ChatGPT API model and a feed-forward deep learning model, offers more control and insight into training. However, it comes with limitations regarding its influence on NLP functioning and small talk topics. On the other hand, Concept 02, utilizing solely the OpenAI model, provides a simpler implementation with a smooth conversation flow but lacks control over NLP functioning and small talk topics.

Our experiments have highlighted the importance of balancing control and simplicity in designing conversational AI systems. While Concept 01 offers more flexibility in shaping dialogue outcomes, Concept 02 demonstrates the efficiency of relying solely on pre-trained models for task-oriented dialogue. The comparison between these concepts underscores the trade-offs between control and simplicity, emphasizing the need for tailored approaches based on specific use cases and user preferences.

Moving forward, we envision further refinement of our models and exploration of additional concepts, such as the development of frameworks featuring task-oriented slots for dialogue completion. By continually innovating and refining our approaches, we aim to contribute to the advancement of conversational AI and its integration into various domains, ultimately enhancing user experiences and facilitating seamless human-computer interaction.

References

- [1] Ssu Chiu et al. “SalesBot: Transitioning from Chit-Chat to Task-Oriented Dialogues”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*. May 2022, p. 425. URL: <https://www.aclweb.org/anthology/2022.acl-long.425.pdf>.
- [2] gk. “Contextual Chatbots with Tensorflow”. In: *Chatbots Magazine* (May 2017). URL: <https://chatbotsmagazine.com/contextual-chat-bots-with-tensor-flow-4391749d0077>.