CinemaBot

A virtual assistant to improve your cinema experience


Jakub Balicki


Michał Ilski

Wrocław University of Science and Technology


Jan Pawłowski


Patryk Rygiel

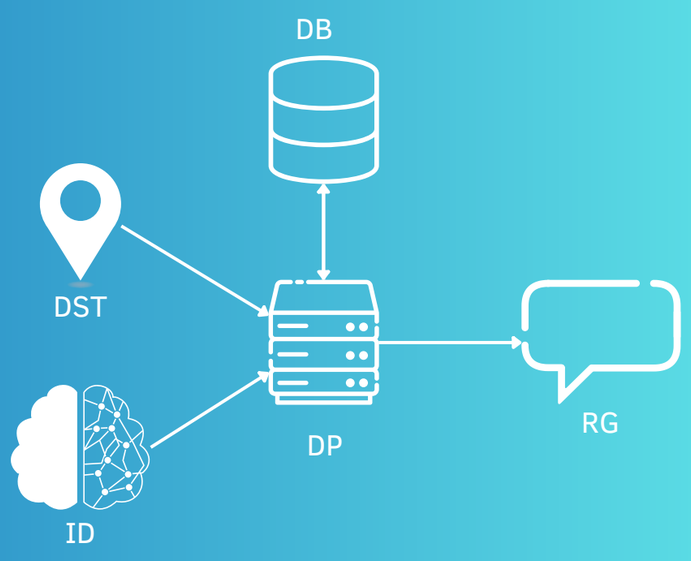
Katedra Sztucznej Inteligencji

AI Tech

Koordynator - prof. Maciej Piasecki

System architecture

CinemaBot is a modular task-oriented dialogue system, which consists of four primary components - intent detection (ID), dialogue state tracking (DST), dialogue policy (DP) and response generation (RG). ID is a Roberta model, DST and RG are T5-based language models, prepared both for English and Polish languages*.



* See Initial training and alpha release in development steps description.

Why CinemaBot?

I want to buy two tickets for Spiderman movie.

ChatGPT

Sure! I can help you with that. However, please note that as an AI text-based model, **I don't have the capability to make actual purchases** for you...

CinemaBot

Of course. When would you like to visit the cinema?

Intent detection

The ID module is vital for identifying the user's intent in a task-oriented dialogue system. It operates in every turn, following a classification approach where predefined intents determine the recognized slots and guide the processing of the user's request.

Dialgoue State Tracking

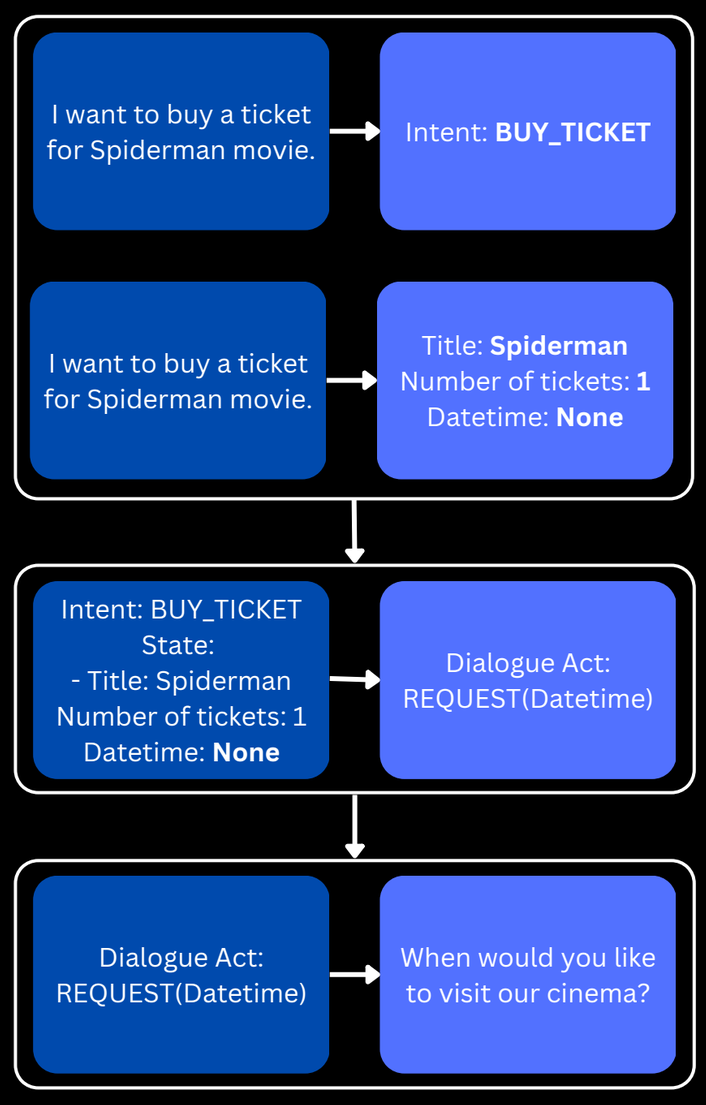
The DST module's main role is to identify and update the dialogue state by incorporating the current utterance with previous turns. Unlike intent detection, this module functions as a sequence-to-sequence task using the T5 model to predict slot values based on the dialogue context and slot name.

Dialogue Policy

This module is unique in our project as it relies on classic algorithms without any machine learning models, making it fully deterministic, due to safety considerations, especially when working with databases. Consequently, the system's naturalness and flexibility depend on its capacity to handle various scenarios and facilitate seamless task switching.

Response Generation

The Response Generation module is the final component in the task-oriented dialogue system's processing pipeline. Its role is to transform structured data, represented by the dialogue act, into natural language responses. By using response templates and the dialogue act, the Response Generation module generates human-like responses that are easily understandable to users.



Development steps

of ongoing process, in which we prepare a **task-oriented dialogue system.**

Datasets

In our research, we started from large conversational datasets like Twitter Corpus. However, extensive annotations, including intents, dialogue state, and response schemas, were required for ID, DST, and NLG. This led us to two suitable datasets: MultiWOZ and Schema Guided Dialogue (SGD). We chose SGD as our primary dataset due to its larger size, naturalness, and sophistication compared to MultiWOZ. To address limited intents in SGD, we incorporated additional datasets like NLU++, Banking77, and Clinic150. Since Polish data was unavailable, we used automatic translation tools (Google Translate, DeepL, PyLingva) with context-based translation for preserving slot values' intended meaning.

System design

After gaining a basic understanding of the pipeline in task-oriented dialogue systems, we determined the specific processing flow and output format for each module to ensure compatibility and integration. Additionally, we made a conscious decision to design the architecture with modularity in mind, allowing for easy replacement of modules when required.

Initial training and alpha release

We commenced our work with the software development and training the ID, DST, and RG modules. Despite utilizing the SGD dataset, our models encountered challenges in effectively generalizing to unfamiliar domains. As a result, we introduced the alpha version of CinemaBot, employing the zero-shot GPT-3.5 model for each module. This approach enables us to attain satisfying outcomes by employing designed prompts and an in-context learning approach. Moreover, it facilitates the collection of valuable data for future fine-tuning of our custom models. The left panel provides a description of the iterative development process.

Deployment and retraining

In the final phase, we employ an iterative approach to gather more data and retrain the models in order to enhance their performance. Recognizing that the current models may not meet our desired standards, we leverage other general-purpose large language models, such as GPT-3.5, to both accomplish the task and collect annotated data through this model.

Stable release roadmap