SUpport: Deep Learning Infused Virtual Orientation Assistant for Essential University Services

Alize Sevgi Yalçınkaya
alizesevgi@sabanciuniv.edu
Data Mining, Development

Faraz Badali Naghadeh
faraz.badali@sabanciuniv.edu
Evaluation, Development

Bilgehan Çağıltay
bcagiltay@sabanciuniv.edu
Evaluation, Development

I. Problem Definition and Motivation

Sabanci University hosts numerous websites for student use; however, accessing up-to-date and reliable information remains a challenge. This issue, widely observed and felt by the student body, stems from numerous dead links and the scattering of related topics across multiple platforms, complicating the university's already complex system. In various industries, chatbots have demonstrated significant effectiveness as a tool for streamlining access to information [1], [2], [3], [4], suggesting that a similar solution could substantially enhance the student experience at Sabanci University. This proposal intersects Human-Computer Interaction (HCI) and Natural Language Processing (NLP), aiming to alleviate the often tedious and sometimes detrimental effects of navigating the bureaucratic and fragmented information landscape of the university.

The proposed chatbot, named SUpport, is designed to answer any question related to Sabanci University, ranging from credit systems and available classes for the semester to faculty expertise and current research projects. SUpport aims to be user-friendly, offering comfortable interactions and providing links and sources when necessary. The successful implementation of this project is expected to benefit most students, particularly newcomers, by significantly improving their access to essential information.

Drawing inspiration from the widely used and recognized student-led project SUchedule [5], which addressed a critical shortfall in one of the university's key IT systems, SUpport aspires to achieve a similar level of utility and acceptance among Sabanci University students. This initiative not only aims to enhance the student experience by simplifying access to information but also sets a precedent for leveraging technology to overcome institutional information barriers.

II. Related Work

Chatbots have become increasingly integral in various domains such as education and healthcare, serving as sophisticated virtual assistants. The design and development of chatbots have significantly evolved, transitioning from basic rule-based models to more advanced versions employing deep learning (DL) and natural language processing techniques.

Retrieval-Based Chatbots, developed using technologies such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and bidirectional LSTMs, process and predict language sequences effectively. Notably, some retrieval-based systems also employ simpler methods like keyword matching and decision trees. Primarily, these models select responses from a predetermined repository, ensuring consistency and accuracy in specific scenarios where response precision is paramount [6].

In contrast, generative models typically utilize an encoder-decoder architecture. While GPT (Generative Pre-trained Transformer) exemplifies this approach, producing contextually relevant responses, BERT (Bidirectional Encoder Representations from Transformers) is more aligned with tasks like classification, entity recognition and understanding context of a query. The strength of generative models lies in their ability to grasp and generate nuanced, human-like language, making them suitable for a diverse range of conversational applications [6].

While GPT based models have become very popular in recent years with the success of openAI's chatGPT, BERT still continues to be a popular pick for DL due to its ability to capture context and details from the surrounding text. In particular, the excellent guide by Briggs [7] shows the many strengths of BERTopic. Due to its very detailed and informative explanation and utilization of BERTopic, the work of Briggs was invaluable to us. In addition to the work of Briggs, we found the works of Julien [8], Su et. al. [9], and Tang [10] to be incredibly helpful.

III. Methods

For the SUpport project at Sabanci University, the development of a hybrid chatbot that leverages both generative and retrieval-based deep learning models is proposed. This hybrid approach will enable the chatbot to provide direct links and cite sources where necessary (retrieval-based) while also allowing for natural conversational capabilities and the synthesis of information from various sites (generative).

To achieve this, we first scrape the HTML structure of the websites under sabanciuniv.edu domain. We process these structures to isolate the main body of the text and separate it into descriptive fields and metadata. The descriptive fields contain long form sentences and paragraphs that need to be semantically analyzed, while the metadata contains short pieces of information such as courses or research. After being cleaned of noise, the texts are separated into their individual sentences so that they can be analyzed for named entity recognition (NER) and intent classification. Due to issues detailed in section 2, these have not been utilized yet but are envisioned to be used with BERTopic for a more reliable detection of the purpose of the text.

1. Data Collection and Preprocessing

The data for this project was sourced from website pages within the sabanciuniv.edu domain, as our chatbot is designed to provide users with information related to their university queries. Initially, we faced a significant challenge due to the absence of readily available data. Our first step involved requesting an API or a sitemap from the university's IT department; however, this request was denied. To address this limitation, we opted to scrape data directly from the HTML files of the websites.

We implemented our initial data collection method by writing a crawler script using Scrapy [11]. This script was designed to visit every page within the specified domain and retrieve text information and URLs from each. After aggregating the data from all available websites, we encountered numerous dead links, which led to empty pages or irrelevant data. Further complicating our efforts, our review of the HTML, conducted using Soup [12], revealed that inconsistent naming of classes and the lack of a uniform design across the websites rendered much of the collected data unusable.

To address the initial data collection challenges, we revised our approach in the second stage, focusing on extracting data from departmental websites within each faculty. These sites were selected based on the likelihood of having common elements and similar layouts in their HTML files. The Faculty of Engineering and Natural Sciences (FENS) was chosen as the starting point. After a thorough inspection of the HTML files, we iteratively developed a script specifically designed to capture the necessary data. However, this process was iterative, with challenges encountered at various stages.

We utilized Soup [12] to systematically collect data from each department and save it as a JSON file. This included the URL of the page being visited and scraped, the title of the page, breadcrumbs, all links on the page, and a descriptive text field about the page (main text). Merely saving the text proved insufficient as, although some pages contained comprehensive information, the data was often too disorganized for our model training purposes, and additional metadata was required. To overcome this, we modified the script to differentiate between sets of URLs and to scrape data based on the structural layout of each page. This adjustment not only facilitated the capture of textual data from each page but also enabled the collection of detailed information about faculty members, their research areas, labs, research groups, and publications, as well as courses offered for both graduate and undergraduate programs. The newly acquired data was stored in a dictionary format inside the metadata field in the JSON file, serving as valuable metadata for subsequent processes in our project.

This methodology was subsequently applied to the websites of other departments within the Faculty of Engineering and Natural Sciences (FENS). While effective for some departments, it failed to capture data from others. We observed that although the layouts were nearly identical, slight differences in the naming of elements and structures necessitated adaptations to our script to enable data capture. Writing multiple scripts for each page was deemed impractical. With these adjustments, we successfully gathered the required information and incorporated code to cleanse our data during the scraping process, ensuring it was saved in the correct format.

One specific task involved converting Turkish characters in names to English. This conversion was initially effective for some websites but problematic for others, particularly because many Turkish words or sentences persisted even on English-version pages. Consequently, altering the characters proved redundant, prompting us to revert these changes. To address this, we utilized the "google translate" library for Python to translate the text into English. Despite these efforts, some words and sentences remained unchanged. Therefore, we manually reviewed the JSON files to identify and correct specific Turkish words after analyzing the data in our data frame.

The final task we completed for data preprocessing was for sentence segmentation. The plain texts needed to be separated into their individual sentences. This way, the processing for NER and intent classification would be easier. To achieve this, we utilized Spacy [13] and its english language pack. While the small language pack was able to do successful segmentations, we eventually chose the medium language pack, as explained in the following feature extraction section.

2. *Feature Extraction*

To process the main text data further, we researched reliable and expandable text classification models as we suspected we would have issues with the size of our data. First, to do NER, we again made use of Spacy and its english language pack. In its non-tuned form, this model is able to recognize course codes, field names, and Sabanci University as organizations. As we were unable to get meaningful performance differences between the large and medium language packs, we opted to pick the smaller but performant "en_core_web_md.' To get more specific and usable information, we added four new labels to the existing tags; 'COURSES,' 'FACULTY,' 'MAJOR,' and 'RESEARCH _TOPIC.' We provided the NER pipeline with example uses of these labels within some sentences but were not able to check its performance on the plain text of our scraped data due to time constraints. We plan to add more training sentences to train the model further and evaluate it against its baseline counterpart in the future.

Following our work with NER, we collected and expanded intents databases from online sources [14], [15], [16] in order to train an intent classification model. For this, we utilized the base uncased BERT model from Transformers [17]. In our first attempt, we focused on university related intent databases for domain specificity. We merged intents of similar purposes to minimize the number of classes and maximize the prompts available per class. By doing this, we ended up with 40 classes (labels) and 518 label-pattern pairs in total, resulting in 12 patterns per class on average. We split this database into training, test, and validation sets, used Adam optimization, and trained for seven epochs.

While the domain of these models were small, an issue with them was the lack of variation in the data as well as the size of prompts per label. This resulted in the model not having enough data to learn from, especially after the train-test-validation split. Due to this, the final validation accuracy was 1.82% (Figure 1). In order to address this we first attempted to expand the patterns of each label in order to help with recognition. By doing so, the model would have more data to learn from per label and (hopefully) increase its validation accuracy. However, this method failed to produce satisfactory results. As an alternative, we opted to search for a method that would utilize a pre-trained model that we could enhance, similar to our utilization of Spacy for NER.

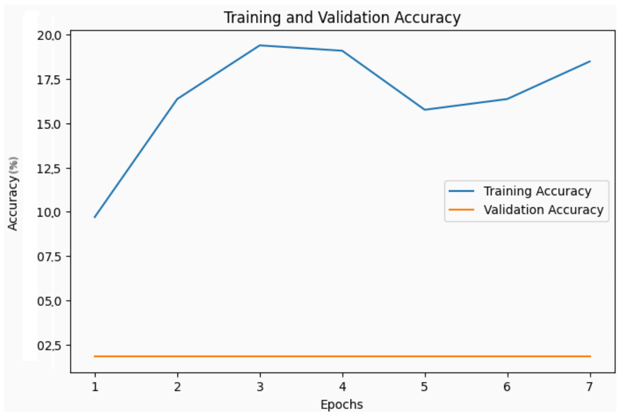

Figure 1: Base intent classification

For the alternative, pre-trained models, we attempted to work with Llama and GPT-3 models. Due to memory limitations with Llama and GPT, we were unable to benchmark the performances of these models. We also tried GPT-2 due to the smaller size but the results were below our expectations.

While we were researching and working on these alternative models, we were also attempting to modify and improve the intent classification pipeline. To allow for better prediction through a smaller and more recognizable tokenization sequence, the intent classification pipeline was merged with the NER model. This was done through entity abstraction. The sentence that was used to train the intent classification model is first passed through the NER model. As a result, the entities of the sentence are detected and returned with their entity keyword counterparts. These sections are then replaced with their entity counterparts, which anonymizes these entities. This is applied to the whole intents dataset, which is in turn used to train the intent classification model. While the idea is simple and should be creating a better score, the final validation score of the model was 1.61% (figure 2).

As the final step of the feature extraction and evaluation step, we processed each sentence that was isolated during the preprocessing step for their topics and clusterings. To achieve this,

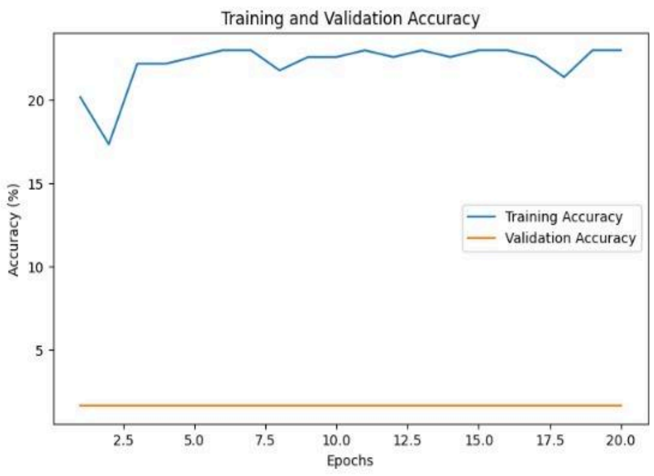

Figure 2: Modified intent classification

BERTopic [7] was utilized due to its context capturing capabilities by including BERT models which captures deep semantic meanings of words in context thus allowing more meaningful topics to be captured. To get a baseline for the model, we first checked its performance with the individual sentences in our dataset and no further modifications other than mandatory cleaning required because of how the data was acquired. The performance of the model as a result of this was below our expectations and needed to be tuned further to be usable for this project. We believe that the biggest reason for this is that we lack variation and amount in our data. Another limitation we are encountering is the amount of noise in our dataset even after the initial cleaning process. Even though BERTopic does not normally require cleaning of the dataset for noise, we wanted to see the difference between sentences that are untouched and sentences that are processed for noise as much as possible. So, to understand this process and how to treat our data better, in the end we ended up trying 3 BERTopic models: Unprocessed data without tuning, Processed data with hyperparameter tuning attempts, and Processed data without tuning.

1st Model: No Preprocessing & No Tuning

Because of our data's condition, how small it is and how poorly formatted, even with a SentenseTransformer and vectorizer, BERTopic kept finding very similar words for each topic, and produced a lot of (170) outliers. This was highly descriptive of our data's condition. We decided to clean our data thoroughly without losing meaning, so we did not use techniques like stemming but we did utilize lemmatization and some custom functions.

2nd Model: Preprocessing & Tuning Efforts

BERTopic utilizes a transformer embedding model, UMAP dimensionality reduction, HDBSCAN clustering and cluster tagging using class-TF-IDF. Due to the nature of our data, UMAP and HDBscan were not able to yield good results.

To do hypertuning on BERTopic's components, we first utilized the breadcrumbs collected from each website, as the contents of the website should be related to the domain and the major specified in this data. The text within the breadcrumbs contained information in the format of `FENS\Molecular Biology Genetics and Bioengineering.` The first part of the breadcrumbs are taken as the domain, while the rest is designated as the major of the page. The content of the page, or the sentences within the page, are then set as the embedding data. With this, we attempt to do the most optimal clustering for hdb scan and dimensionality reduction for UMAP over this clustering.

We attempted to do this such that the global as well as local information can be utilized within BERTopic. However, the issue of a small dataset with large amounts of noise still haunted us at this step. While we attempted to process the sentences to reduce noise as much as possible, all steps of this process suffered due to the issues with our data.

3rd Model: Preprocessing & No Tuning

Due to the difficulties of Model 2, we still wanted to observe how well the preprocessing was performing. We yielded fewer outliers (22) and we were able to receive topics that made more sense and were not repeating as much as Model 1 such as [biology, molecular, biological, research, plant, genetics, plants, bioengineering, cell, protein]

3. *Model Selection*

Retrieval-based Model: Utilize a model like BERT (Bidirectional Encoder Representations from Transformers) for its strong understanding of language context and its ability to extract relevant information from a large dataset. This model will handle queries that require direct answers or links to sources.

Generative Model: Deploy a generative model like GPT (Generative Pretrained Transformer) for generating human-like responses to queries that need synthesis of information or when a direct answer is not available in the dataset. This model will be responsible for the chatbot's conversational aspects.

In the final step, the model and methods were chosen based on our prior experiments and the limitations that we faced. For this step the NER data was used. Certain rigid rules were implemented to tag certain entities such as course name, professor name and research field. The entities were derived from data using custom scripts to catch data patterns. Additionally, these data were checked using LabStudio software and the outliers were caught and fixed by tagging manually.

These data were then used to train the RAG model [18], [19], [20], [21], [22]. The reason for using the RAG model was the retrieval based nature of the model that makes it more accurate in comparison to the generative models. It is able to retrieve data from the knowledge base. In this model we first trained only using NER data without intent classification and then tested it with four prompts.

The four prompts are: "Who is the instructor for CS201?" "Who is Albert Levi?" "What MAJORS are there?" and "hi". The responses by the above model for prompts were "Software Engineering," "Albert Levi," "Industrial Engineering," and "Hello" respectively.

In the second iteration intent classification was integrated into the pipeline by utilizing BERT and comparing it against the same four prompts. We also attempted to augment the database so as to create a more extensive and reliably trainable dataset.

In this iteration, before data augmentation, the responses to prompts can be found in appendix 1. The answers to the given prompts after data augmentation can be found in appendix 2.

4. *Model Training and Fine-Tuning*

Retrieval-based Model Training: Train the BERT model on the collected and preprocessed university data, ensuring it can accurately retrieve information and links based on query context.

Generative Model Training: Train the GPT model on a dataset of conversations and university-related information to ensure it can generate coherent and contextually appropriate responses.

Fine-Tuning: Fine-tune both models on a specific subset of data that includes common student inquiries and official university responses to improve accuracy and relevance.

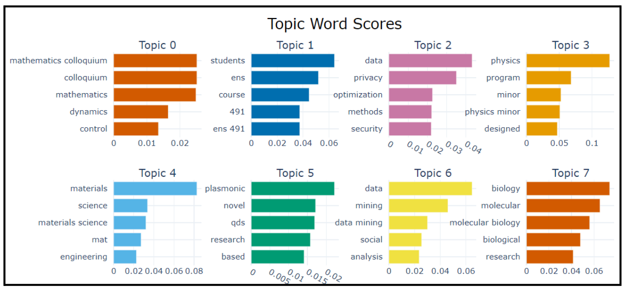

Figure 1. BERTopic with no tuning

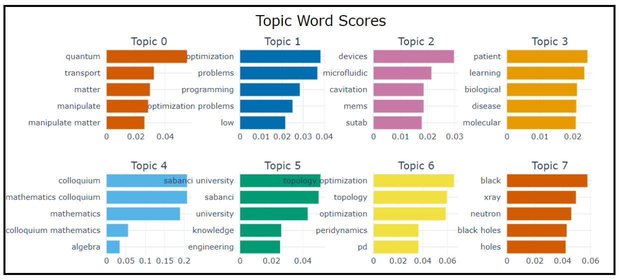

Figure 2. BERTopic with Preprocessed data and Tuning

IV. Future Work

1. *Integration and Workflow*

Develop an integration layer that decides which model to invoke based on the nature of the query. If the query is straightforward and matches the knowledge base directly, use the retrieval-based model to provide a precise answer or link. For more complex queries or when a synthesized response is required, switch to the generative model. Ensure seamless transition between models within the same conversation thread if needed.

2. *User Interface and Experience*

Design a user-friendly interface that allows students to easily interact with SUpport through text input. Implement feedback mechanisms for users to rate responses, allowing continuous improvement of the models based on user satisfaction.

3. *Continuous Learning and Updates*

Set up a system for continuous learning where the chatbot can learn from new data and student interactions to improve its accuracy and relevance over time. Regularly update the dataset with the latest university information and policies to keep the chatbot's responses current.

4. *Evaluation and Iteration*

Conduct testing with real users to evaluate the chatbot's performance, focusing on accuracy, relevance, and user satisfaction. Iterate on the models and user interface based on feedback and performance metrics to enhance the chatbot's capabilities and user experience.

References

[1] S. A. Abdul-Kader and J. Woods, "Question answer system for online feedable new born Chatbot," in *2017 Intelligent Systems Conference (IntelliSys)*, Sep. 2017, pp. 863–869. doi: 10.1109/IntelliSys.2017.8324231.

[2] L. Zhou, J. Gao, D. Li, and H.-Y. Shum, "The Design and Implementation of XiaoIce, an Empathetic Social Chatbot," *Computational Linguistics*, vol. 46, no. 1, pp. 53–93, Mar. 2020, doi: 10.1162/coli_a_00368.

[3] B. A. Eren, "Determinants of customer satisfaction in chatbot use: evidence from a banking application in Turkey," *International Journal of Bank Marketing*, vol. 39, no. 2, pp. 294–311, Jan. 2021, doi: 10.1108/IJBM-02-2020-0056.

[4] Q. Chen, Y. Gong, Y. Lu, and J. Tang, "Classifying and measuring the service quality of AI chatbot in frontline service," *Journal of Business Research*, vol. 145, pp. 552–568, Jun. 2022, doi: 10.1016/j.jbusres.2022.02.088.

[5] B. Ayaz, "aburakayaz/suchedule." Feb. 11, 2024. Accessed: Mar. 07, 2024. [Online]. Available: https://github.com/aburakayaz/suchedule

[6] S. Pandey and S. Sharma, "A comparative study of retrieval-based and generative-based chatbots using Deep Learning and Machine Learning," *Healthcare Analytics*, vol. 3, p. 100198, Nov. 2023, doi: 10.1016/j.health.2023.100198.

[7] J. Briggs, "Advanced Topic Modeling with BERTopic | Pinecone." Accessed: Apr. 28, 2024. [Online]. Available: https://www.pinecone.io/learn/bertopic/

[8] J. Blanchard, "BERTopic, or how to combine transformers and TF-IDF for topic modelling," Julien's data blog. Accessed: Apr. 28, 2024. [Online]. Available: https://blanchardjulien.com/posts/bertopic/

[9] H. Su *et al.*, "One Embedder, Any Task: Instruction-Finetuned Text Embeddings," 2022. [Online]. Available: https://arxiv.org/abs/2212.09741

[10] Y. Tang, "Python Topic Modeling With a BERT Model," Deepgram. Accessed: Apr. 28, 2024. [Online]. Available: https://deepgram.com/learn/python-topic-modeling-with-a-bert-model

[11] "scrapy/scrapy." Scrapy project, Apr. 28, 2024. Accessed: Apr. 28, 2024. [Online]. Available: https://github.com/scrapy/scrapy

[12] L. Richardson, "Beautiful Soup." Accessed: Apr. 28, 2024. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/

[13] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020, doi: 10.5281/zenodo.1212303.

[14] rituparna-glitch, "rituparna-glitch/University-QnA-Chatbot." Jun. 21, 2022. Accessed: Apr. 28, 2024. [Online]. Available: https://github.com/rituparna-glitch/University-QnA-Chatbot

[15] T. Paul, "University Chatbot Dataset." Accessed: Apr. 28, 2024. [Online]. Available: https://www.kaggle.com/datasets/tusharpaul2001/university-chatbot-dataset

[16] M. Rashed, "mmohamedrashed/Python-University-Chatbot." Accessed: Apr. 28, 2024. [Online]. Available: https://github.com/mmohamedrashed/Python-University-Chatbot/tree/main

[17] T. Wolf *et al.*, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[18] T. Waitzenberg, “Mastering RAG Chatbots: Building Advanced RAG as a Conversational AI Tool with LangChain,” Medium. Accessed: Jun. 05, 2024. [Online]. Available: https://medium.com/@talon8080/mastering-rag-chatbots-building-advanced-rag-as-a-conversational-ai-tool-with-langchain-d740493ff328

[19] V. Lyashenko, “Data Augmentation in Python: Everything You Need to Know,” neptune.ai. Accessed: Jun. 05, 2024. [Online]. Available: https://neptune.ai/blog/data-augmentation-in-python

[20] Chituyi, “Spacy for Named Entity Recognition and LLMs for Text Summarization and Headline Generation.,” Medium. Accessed: Jun. 05, 2024. [Online]. Available: https://medium.com/@cwakhusama/spacy-for-named-entity-recognition-and-llms-for-text-summarization-and-headline-generation-74f35b15b35e

[21] “Retrieval Augmented Generation - Type 1.” Accessed: Jun. 05, 2024. [Online]. Available: https://kaggle.com/code/reichenbch/retrieval-augmented-generation-type-1

[22] N. V. Otten, “Retrieval-Augmented Generation (RAG) Made Simple & 2 How To Tutorials,” Spot Intelligence. Accessed: Jun. 05, 2024. [Online]. Available: https://spotintelligence.com/2023/10/19/retrieval-augmented-generation-rag/

Appendix

Appendix 1: Responses to prompts without data augmentation for RAG model with intent classification.

```
Question: Who is the instructor of CS 201?
Predicted Intent: request_person_info
Intent: request_person_info
Question: Who is the instructor of CS 201?
Response: I'm not sure how to respond to that.
Question: Hi
Predicted Intent: greeting
Intent: greeting
Question: Hi
Response: Hello! How can I assist you today?
Question: Who is Albert Levi?
Predicted Intent: request_person_info
Intent: request_person_info
Question: Who is Albert Levi?
Response: I'm not sure how to respond to that.
Question: What MAJORS are there
Predicted Intent: COURSE
Intent: COURSE
Question: What MAJORS are there
Response: industrial engineering
```

Appendix 2: Responses to prompts with data augmentation for RAG model with intent classification

```
Question: Who is the instructor of CS 201?
Predicted Intent: request_person_info
Intent: request_person_info
Question: Who is the instructor of CS 201?
Response: Programming Fundamentals.
Question: Hi
Predicted Intent: greeting
Intent: greeting
Question: Hi
Response: Hello! How can I assist you today?
Question: Who is Albert Levi?
Predicted Intent: request_person_info
Intent: request_person_info
Question: Who is Albert Levi?
Response: Albert Levi.
Question: What MAJORS are there
Predicted Intent: COURSE
Intent: COURSE
Question: What MAJORS are there
Response: industrial engineering
```