OptiBot:
Enhancing Chatbot Interactions through Clustering and LLM Support

Final Project

HSLU Hochschule Luzern

Dominik Bacher Suarez
Master of Science in Applied Information and Data Science

Spring, 2024

Generative AI Course
Prof. Dr. Marcel Blattner
Lucerne University of Applied Sciences

Abstract

This project is accessible via GitHub repository at https://github.com/dominik117/optibot and can be interacted with through a Streamlit application at https://optibot.streamlit.app/ . The Streamlit public app response times may vary.

For installation and use in local system for potentially better performance, instructions are available in the README file.

Development work and exploratory analysis are documented in the "notebooks" directory, which contains detailed notebooks explaining the code's functionality and the underlying concepts. The sequence of the notebooks is as follows:

1. Optibot_Topic_Modeling_Engineering.ipynb
2. Optibot_LLM_Topic_Labeling.ipynb
3. OptiBot_LLM_Satisfaction_Analysis.ipynb
4. OptiBot_LLM_Final_Insights.ipynb

For the core application code, refer to the "optibot" directory, which serves as the source, containing 769 lines of code.

Table of Contents

Challenge Description

This report presents Challenge 4, one of the final projects for the Generative AI course taught by Professor Dr. Marcel Blattner. The task is as follows:

> *"Build a system that is able to generate use case suggestionsbased on LLM conversations."*

This challenge makes use of the principles of Generative AI and Python application development. The project aims to integrate the tools and techniques learnt at the Generative AI class, as well as other knowledge from other Data Science modules, looking to demonstrate the completion of the Generative AI class, while also being a potential project for future development and release.

Project Overview

OptiBot is designed to suggest improvement areas for chatbot interactions across multiple sectors, such as healthcare, finance, customer service, among many others. Using NLP clustering techniques and LLM support, OptiBot identifies areas for chatbot improvement based on the historical chatbot conversation data from the end-user, providing insights to chatbot developers into which areas of their trained chatbot need further fine-tuning or incorporate better context.

Introduction

Chatbots are integral in various sectors for automated support, using AI and LLM; particularly in sectors like e-commerce, healthcare among others.. Their adoption faces challenges and many limitations despite their broad utilization (Caldarini, Jaf, & McGarry, 2022[1]). Some of the limitations can be absence of empathy or human-like responses. Other limitations might be inaccurately answering questions, or not even addressing the context of the user's question.

OptiBot, a system designed to enhance chatbot functionality by leveraging advanced topic modeling techniques from NLP and LLM. OptiBot is not just a tool but a platform open to users, aiming to assess chatbots across five key success metrics: relevance, accuracy, completeness, conciseness, and tone.

[1] Caldarini, G., Jaf, S., & McGarry, K. (2022). A Literature Survey of Recent Advances in Chatbots. Information, 13(1), 41. [https://www.mdpi.com/2078-2489/13/1/41]

Methodology

1. Data Gathering

OptiBot's initial design doesn't involve fine-tuning, with the chatbot data provided by the end-user. However, to evaluate OptiBot's initial performance and facilitate its engineering, it was needed to simulate end-user interactions. For this, open-source datasets from LLM chatbot interactions were needed for developing the necessary algorithms for clustering, labeling, and assessment.

These datasets were chosen for their relevance to the project's objectives and their potential to provide a comprehensive range of scenarios:

- **Bitext Dataset:** This dataset offers a wide array of customer service interactions across various fields. (Bitext Innovations, 2023)[2]
- **Medical AI Dataset:** Sourced from IBM WatsonAI, this dataset includes dialogues from doctor-patient consultations. (Ruslan Magana Vsevolodovna, 2024)[3]
- **ExpBot Dataset:** Containing 79 dialogues with an experimental customer service chatbot. (Gnewuch, Feine, Morana & Mädche, 2023)[4]

It's important to note that these datasets were solely used for testing and development purposes. They were referenced in creating the algorithms necessary for OptiBot's functionality but do not constitute part of the bot's final operational framework.

2. Clustering with Topic Modeling on End User Data

OptiBot first requires the user to upload their historical data of chatbot conversations. Initially, it requires users to identify the data columns corresponding to customer questions and chatbot responses. After identification, OptiBot concatenates these columns, prefacing each entry with "Question:" or "Answer:" to differentiate between them.

The combined text passes through normalization processes including conversion to lowercase, removal of special or single characters, exclusion of stopwords, and lemmatization, followed by tokenization of the entire conversation corpus.

OptiBot then identifies significant bigrams within the conversation data. Using Gensim, a bigram is recognized if a specific word pair, such as "customer" and "support," appears together in a minimum number of conversations (e.g. 20). Consequently, "customer_support" becomes a new token in the dataset's bag of words, among other meaningful bigrams found in the data.

A bag of words (BoW) is created, which is a collection where each unique word in the text corresponds to an index. Transforming text into numerical representations that the clustering algorithm can deal with.

[2] Bitext Innovations (2023). Bitext - Customer Service Tagged Training Dataset for LLM-based Virtual Assistants. [https://github.com/bitext/customer-support-llm-chatbot-training-dataset]
[3] Ruslan Magana Vsevolodovna (2024). Doctor Consultation with Artificial Intelligence. [https://github.com/ruslanmv/ai-medical-chatbot]
[4] Gnewuch, U., Feine, J., Morana, S. & Mädche, A., (2023). Sie sind hier:ExpBot - A dataset of 79 dialogs with an experimental customer service chatbot
Datenpaket: ExpBot - A dataset of 79 dialogs with an experimental customer service chatbot. [https://www.radar-service.eu/radar/de/dataset/FdJmclKpjHzLfExE]

Finally, OptiBot applies these tokens to a Latent Dirichlet Allocation (LDA) model. LDA is a type of topic model that identifies themes from a collection of texts. OptiBot iterates over a range of topic numbers, selecting the optimal count based on the highest coherence Cv score, which is the best metric assessing the model's quality (Kapadia, S., 2019)[5]. Each chat conversation is then assigned to the topic with the highest likelihood, which clustering the data based on their thematic content.

3. Labeling with LLM

After clustering topics using LDA, OptiBot makes use of an LLM, for the first deployment of the system it uses a "gpt-3.5-turbo" model from OpenAI through API calls, which labels the topics. The LLM is prompted to generate labels based on the top 20 keywords from each topic, which are ranked by their LDA clustering weights. The process is two-fold:

1. First Pass: The initial prompt to the LLM provides context about the chatbot conversations and instructs the model to label each topic based on its keywords. The prompt specifies that keywords are listed in order of significance, with the first being most important. Here is the prompt used:

> *“You are designed to generate concise labels for topics. These topics are derived from chatbot conversations about {context} using LDA Topic Modeling. You will be provided with keywords which are the most representative words from the topic. The first keywords in the list are way more significant for topic assignment of the chat conversation, while the latter ones decrementally reduce their importance and should be used to provide additional context. Existing generated labels for other topics are: {existing}. Your task is to provide a single, pertinent label for each set of keywords representing a topic, ensuring the label accurately reflects the {context} context of the conversation.”*

2. Second Pass (Review/Validation): After initial labels are generated, a second LLM prompt conducts a review, considering all topics and their labels collectively. This holistic review ensures all each label is distinct and accurately represents its topic within the context of the conversation data. Here is the prompt used:

> *"You are designed to review generated labels for topics. These topics are derived from chatbot conversations about {context} using LDA Topic Modeling. The LDA modeling gave the most impactful keywords for topic assignment of the chat conversation, The first keywords in the list of are way more significant for the topic assignment, while the latter ones decrementally reduce their importance and should be used to provide additional context only. The AI generated topics are as follows: {iteration over topic labels}. Holistically considering all these previously generated labels and their keywords, for each topic suggest an improved label or keep the current label if it's optimal. Only answer with the new suggested label name or the initial label name if no improvement is needed."*

The process aims to refine topic labels, making them representative and concise.

[5] Kapadia, S. (2019). Evaluate Topic Models: Latent Dirichlet Allocation (LDA) [https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0]

4. Assessing The Conversations

In this phase, OptiBot prompts the LLM to evaluate the relevance and quality of the question-answer pairs, guided by the topic labels assigned earlier. The evaluation is based on a set of five criteria, with the process starting by presenting a one-shot learning example to the LLM, showcasing how to apply the scoring system.

The scoring criteria include:

1. **Relevance:** Determines if the answer addresses the question effectively.
2. **Accuracy:** Assesses the correctness of the information, performing a fact-check backed by the LLM knowledge.
3. **Completeness:** Checks if the response covers all necessary aspects of the question.
4. **Conciseness:** Evaluates the clarity of the response, suggesting using a measure like the Flesch-Kincaid readability score.
5. **Tone:** Examines whether the response is engaging and maintains an appropriate tone.

Each criteria is rated on a scale from 1 to 5, followed by a brief and precise assessment made by the LLM prompt response. These scores are then applied to the entire corpus of labeled Q&A pairs. The use of prompt engineering is crucial in this stage, which proved to move the LLM to accurately understand and apply the evaluation criteria.

5. Final Insights

In the final stage, OptiBot analyzes the satisfaction assessment results to provide insights and recommendations for improving chatbot interactions, focusing on the three topics with lowest assessment scores. The process is the following:

1. **Aggregating Scores:** OptiBot computes the average score across the five criteria for each topic.
2. **Identifying Key Areas:** The system pinpoints the top three topics with the lowest performance scores, using these topics and their earlier assessment for the final prompt to the LLM.
3. **Generating Recommendations:** OptiBot consolidates the textual assessments produced during the assessment phase. It then makes another prompt to summarize these findings, summarizing the main issues identified with the lowest-scoring topics and proposing specific improvements.

6. Streamit User Interface

OptiBot is integrated into a Python-based Poetry project, with modularization, OOP and user interaction; deployed on a Streamlit app UI. Here's a breakdown of the user interface and processing:

1. **OptiBotModeling Class:** The core functionality is in a class named “OptiBotModeling”. When users upload their conversational data to the system, this class is instantiated, and the “fit” method is called to initiate all the functions needed to process OptiBot’s methodology, including clustering, labeling, assessing satisfaction, and making recommendations.

2. **Dataframe Processing:** Users upload their chatbot conversation data as a dataframe through the Streamlit application. They are prompted to indicate the columns belonging to the questions and answers, and give a brief context on the data, used by the LLM prompting.
3. **Analysis and Results Presentation:** After the user uploads the data, OptiBot performs the analysis, which may take between 1 to 2 hours, depending on the complexity of the data. Once completed, the app displays various results, statistics, and insights, focusing particularly on the three worst-performing topics as identified by the assessment analysis.
4. **Insights and Recommendations:** The final output presented to the user includes detailed insights and recommendations to improve the chatbot's performance.
5. **Privacy:** The app is designed with user privacy in mind; it does not store the uploaded data, not the analysis performed and displayed.

Results and Overview

OptiBot's implementation shows the practical integration of Generative AI and machine learning to seek to improve LLM chatbot communications. Here's a summary of the outcomes:

1. **Topic Modeling:** Gensim LDA was selected for its efficiency in categorizing chatbot conversations, demonstrating a balance in processing speed and result accuracy, while also choosing the best number of topics depending on the data, according to evaluation metrics.
2. **Large Language Model Use:** ChatGPT's integration highlighted its capability in generating relevant labels and assessments with ease, but at an API cost.
3. **Detailed Evaluations:** The LLM assessment framework provided comprehensive insights into chatbot conversations with the use of evaluation metrics.
4. **Practical Recommendations:** OptiBot translated its analytical findings into specific, actionable steps for chatbot improvement.
5. **AI Field Contribution:** The project demonstrates how Generative AI, in specific with the use of LLMs, can be combined with data science and software development to aid in the functionality of interactive AI systems, like AI powered chatbots, providing insights that could influence future developments.

Implementation and Challenges

The implementation of OptiBot involved a lot og testing, engineering and overcoming challenges, particularly in the choice and optimization of the topic modeling and language model components. Here's an overview of the implementation road:

- **Topic Modeling Selection:**
  - **Gensim LDA:** Chosen for its efficiency, Gensim LDA provided a good balance between speed and good coherence score.
  - **Scikit-learn LDA:** Found to be slower with slightly lower coherence scores than Gensim, making it less ideal.
  - **BERTopic:** Although it offered a 30% increase in coherence scores, BERTopic's training time and dependency requirements were considerably less suitable for deployment.

- **Optimization:** Fine-tuning Gensim LDA required tunning hyperparameters to balance coherence and training time, this was performed through a very time consuming iterativ process.
- **Deployment and User Interface:** Originally, an AWS Dockerized application was considered and tried. But Streamlit proved to be an acceptable choice. Especially with the use of Poetry for the environment and dependencies management.
- **Language Model Integration:** LLaMA 2 was initially considered for the LLM component. However because of the urgency of deployment and the complexities to integrate LLaMA into the deployment, ChatGPT was chosen for its ease of use through API calls.
- **LLM Prompt Engineering:** A series of prompts were tested with ChatGPT to refine the topic labeling and the assessment process, with a lot human validation, to find aspects that could be isolated and algorithmically programmed.

System Architecture

- **Programming Language and Version:** Python 3.12.
- **Dependency Management:** Poetry.
- **Version Control:** Git, repository located in GitHub. [https://github.com/dominik117/optibot]
- **User Interface:** Streamlit. Deployed to the public. [https://optibot.streamlit.app/]
- **Natural Language Processing (NLP):**
    - Gensim[6]: Used for LDA topic modeling and bigram detection.
    - NLTK: tokenization, stemming, tagging, and parsing, general text preprocessing.
- **LLM:** OpenAI's Chat-GPT API.
- **Data Science Toolkit:**
    - Pandas
    - NumPy
    - Matplotlib
    - Plotly

Recommendations and Future Work

- **User Interface Improvements:** Implement a feature allowing users to preview examples from selected Q&A columns, aiding in more accurate data input.
- **Optimization of Topic Selection:** Explore methods like random or Bayesian search to optimize the selection of the number of topics. Consider asynchronous programming to improve the efficiency of the topic modeling process.
- **Model Fine-Tuning:** Further refine the k-topic model through more testing across diverse datasets to enhance model accuracy and reliability.
- **Deployment Enhancements:** Integrate a progress bar in the Streamlit app to provide real-time feedback on the progress.

[6] Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks (pp. 45–50). [http://is.muni.cz/publication/884893/en.]

- **Web Container:** Transition the deployment from Streamlit to a more robust web container, using a Dockerized environment hosted on AWS.
- **Data Visualization:** Implement t-SNE or PCA transformations to visualize topic clusters.
- **Local Model Integration:** Evaluate the feasibility of using a local model like LLaMA, using frameworks like langchain, Hugging Face, or ONNX.
- **Sample Selection Algorithm:** Develop an algorithm to identify the most coherent conversation samples for analysis, ensuring that clustering is performed on the most representative data.
- **Enhanced Assessment Analysis Metrics:**
    - For relevance assessment, compute semantic similarity scores.
    - For conciseness evaluation, consider implementing Flesch-Kincaid Readability or ROUGE scores directly without the use of the LLM.
- **User Customization:** Allow users to select the number of topics they wish to explore in-depth, rather than limiting the insight to the top three.
- **Monetization:** Consider introducing a paywall for access to OptiBot.

References

❖ Caldarini, G., Jaf, S., & McGarry, K. (2022). A Literature Survey of Recent Advances in Chatbots. Information, 13(1), 41. [https://www.mdpi.com/2078-2489/13/1/41]

❖ Bitext Innovations (2023). Bitext - Customer Service Tagged Training Dataset for LLM-based Virtual Assistants. [https://github.com/bitext/customer-support-llm-chatbot-training-dataset]

❖ Ruslan Magana Vsevolodovna (2024). Doctor Consultation with Artificial Intelligence. [https://github.com/ruslanmv/ai-medical-chatbot]

❖ Gnewuch, U., Feine, J., Morana, S. & Mädche, A., (2023). Sie sind hier:ExpBot - A dataset of 79 dialogs with an experimental customer service chatbot

❖ Datenpaket: ExpBot - A dataset of 79 dialogs with an experimental customer service chatbot. [https://www.radar-service.eu/radar/de/dataset/FdJmclKpjHzLfExE]

❖ Kapadia, S. (2019). Evaluate Topic Models: Latent Dirichlet Allocation (LDA) [https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0]

❖ Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks (pp. 45–50). [http://is.muni.cz/publication/884893/en.]