

**Chatbot Using Seq2Seq Approach**

Shruthi AK

University of San Diego

Natural Language Processing and GenAI AAI-520

Prof. Haisav Chokshi

21-10-2024

# Abstract

# This project explores the design and implementation of a generative chatbot capable of mimicking human-like conversations. Using a sample of 30,000 records from the Cornell Movie Dialogues dataset, the chatbot was trained to generate contextually relevant responses. Various models, including Seq2Seq, transformers, and large language models (LLMs), were tested, though limitations in computational resources and dataset size influenced the final choice. The chatbot was deployed locally via a custom web interface. This report outlines the project’s methodology, challenges, evaluation results, and future improvements, along with tables and figures that visualize key aspects of the process.

# Introduction

Chatbots have become indispensable tools across industries, enhancing customer service, personal assistants, and interactive platforms. While rule-based chatbots follow predefined patterns, generative chatbots rely on machine learning to generate responses dynamically based on input data. The challenge with building generative chatbots lies in their ability to understand context and deliver meaningful responses, even when presented with new, previously unseen queries.

This project aims to bridge the gap between rigid, rule-based systems and more flexible, conversational agents by developing a generative chatbot trained on movie dialogue exchanges. By leveraging modern natural language processing (NLP) techniques and deep learning architectures, this chatbot aspires to engage in conversational exchanges that feel fluid and spontaneous (Vaswani et al., 2017).

# Description of the Project

The project's primary goal was to create a generative chatbot capable of producing contextually relevant responses. This involved several steps, including data preprocessing, model selection, training, evaluation, and deployment. The model had to effectively learn conversational patterns from the dataset and generalize to generate suitable responses in real-time. A custom web interface was developed locally for user interaction with the chatbot. In the interface, users can input text, and the chatbot generates a response dynamically, reflecting the model's understanding of conversational structures.

Due to the scope of the project, various machine learning models were tested before settling on a Seq2Seq architecture (Hochreiter & Schmidhuber, 1997). Additionally, efforts were made to deploy the chatbot efficiently, ensuring that even with limited computational resources, the system performed within acceptable response times.

# Objectives

This project set out to achieve the following objectives:

1. Model Development: Design and implement a generative chatbot using advanced deep learning techniques, focusing on sequence-to-sequence (Seq2Seq) modelling with LSTM units.
2. Exploration of Alternatives: Experiment with more complex models, such as fine-tuned transformers and pre-trained large language models, to evaluate their suitability for chatbot generation.
3. Local Web Interface Deployment: Build a web-based interface to deploy the chatbot locally, allowing for user interaction and testing in a controlled environment.

# Dataset Description and Source

The **Cornell Movie Dialogues dataset** is an extensive collection of over 220,000 lines of dialogue from 617 different movies. For this project, a sample of 30,000 dialogues was extracted, providing a balance between conversational diversity and manageable computational load. Each dialogue consists of pairs of utterances between characters, allowing the model to learn how one speaker’s input is typically followed by another’s response (Refer Appendix A).

The dataset includes additional metadata like character information and movie names, though for the purpose of training the chatbot, only the raw dialogue pairs were used. The Cornell dataset is available for free on Kaggle, making it a popular resource for conversational AI projects.

# Why This Dataset Was Chosen

The Cornell Movie Dialogues dataset was selected for several reasons:

1. Conversational Richness: With dialogue spanning various genres and tones, the dataset provides rich, natural language interactions that are more complex than synthetic data or scripted dialogues.
2. Structured Format: The dataset includes clear pairings of conversational turns, making it suitable for training generative models that need to learn input-response relationships.

# Exploratory Data Analysis (EDA)

EDA played a crucial role in understanding the data before model training. The analysis uncovered various insights about the dataset's structure and content.

During EDA, the following key findings were made:

1. Most dialogues contain around 15 words per utterance, which indicates a relatively short conversation style typical of movie dialogues.
2. The word cloud analysis revealed that the most frequently occurring words are common pronouns and conversational markers like "I", "you", and "it". This suggests that conversations in the dataset often revolve around personal interactions and relationships.

In terms of vocabulary, the dataset has a moderate vocabulary size, which is ideal for training a generative chatbot. However, there are many rare words that the model may not learn effectively due to their infrequent appearance (Refer Appendix B for EDA and Appendix C for Preprocessing)

# Methodology and Model Selection

## Seq2Seq Architecture

The model used in this project is based on the Seq2Seq architecture with LSTM units. The Seq2Seq model is popular in NLP tasks involving input-output sequences, such as machine translation. The encoder-decoder structure is designed to map a sequence of words (the user's input) to a target sequence (the chatbot's response).

|  |  |
| --- | --- |
| **Layer Type** | **Details** |
| Embedding Layer | 50 - dimension embedding vector |
| Encoder LSTM | 400 Units |
| Decoder LSTM | 400 Units |
| Dense Layer | SoftMax activation for predictions |

The encoder processes the input text and condenses it into a vector representation (also known as the context vector). This representation is passed to the decoder, which generates a response word-by-word. The model was trained using the cleaned dataset and optimized with categorical cross-entropy loss (Refer Appendix D and E for Model, Appendix F for Chatbot responses, Appendix G for Front End Development).

# Experiment with Other Models

Before finalizing the Seq2Seq model, several other architectures were tested. Below is a summary of the models experimented with and the challenges encountered:

|  |  |  |
| --- | --- | --- |
| **Model** | **Result** | **Challenges** |
| GPT 2 | Underfitting | Limited data, Model complexity, very long training time |
| BERT Fine Tuning | Poor conversational flow | Long training time |
| Vanilla Seq2Seq | Moderate performance | Required extensive tuning, slow to converge |
| Transformer | Unsuccessful due to size | Insufficient resources for full training |

Although models like GPT-2 and transformers have shown state-of-the-art performance in various NLP tasks, they were too resource-intensive for this project given the computational limitations. The Seq2Seq architecture, while simpler, provided a good trade-off between performance and feasibility.

# Challenges Faced and Solutions Implemented

Several challenges arose during the development of the chatbot, particularly related to computational limitations and the complexity of the dataset. These challenges, and the solutions adopted, are outlined below:

1. **Data Processing**: The original dataset contained noisy data such as movie-specific jargon and incomplete dialogues.

**Solution**: The dataset was cleaned using text normalization techniques (lowercasing, removing punctuation, etc.), Stemming/ Lemmatization is not carried out as for the Chatbot model, the sequence and the exact meaning is required to generate the proper response.

1. **Computational Resources**: Training models like GPT-2 and transformers was computationally prohibitive. **Solution**: By opting for a Seq2Seq architecture, the training time and memory usage were significantly reduced, allowing the project to stay within resource constraints.
2. **Model Optimization**: Initial model results showed overfitting on the training data.

**Solution**: Hyperparameter tuning, including early stopping and dropout layers, was applied to generalize the model's learning to unseen data.

1. **Deployment**: Cloud deployment posed cost-related challenges.

**Solution**: A simple Flask-based web interface was developed to deploy the model locally for interactive testing.

# Evaluation Results

The evaluation was done based on qualitative measures (how coherent and relevant the generated responses were) and quantitative measures like accuracy and loss.

|  |  |
| --- | --- |
| **Model Performance Metrics** | **Value** |
| Accuracy | 85.04 |
| Loss | 0.64 |

The chatbot was able to produce reasonably coherent responses for short exchanges. However, in longer conversations, response quality degraded. This limitation is largely due to the constraints of the sample size and the simplified model architecture.

# Insights

1. **Limited Vocabulary**: The small vocabulary size of 3,334 words limited the variety of the chatbot’s responses. As a result, the model often produced repetitive or generic responses, especially in longer conversations.
2. **Contextual Understanding**: The model performed reasonably well for short, straightforward conversations but struggled with more complex interactions. This was expected, given the limitations of the Seq2Seq model, which does not inherently handle long-term context as well as transformer-based models.
3. **Training Efficiency**: Despite limited resources, the model was able to converge relatively quickly due to the reduced dataset size and simplified architecture.

# Future Improvements and Scalability Options

1. **Increasing Dataset Size**: In future iterations, incorporating a larger subset or the entire Cornell dataset could improve the chatbot’s conversational depth and variety. More training data would also allow the model to learn from less frequent words and phrases, increasing response diversity.
2. **Advanced Architectures**: Leveraging cloud resources would enable the use of more advanced architectures like transformers or LLMs (e.g., GPT-3). These models have been shown to produce more coherent and contextually aware responses, even in extended conversations.
3. **Interactive Learning**: Another improvement would be to implement a mechanism for real-time feedback, allowing the model to adjust and fine-tune its responses based on user interaction.
4. **Deployment on Cloud**: Moving the local web interface to a cloud-based platform would allow for wider accessibility and more scalable usage. Hosting the chatbot on platforms like AWS or Google Cloud could support larger user bases and more intensive real-time computations.

# Conclusion

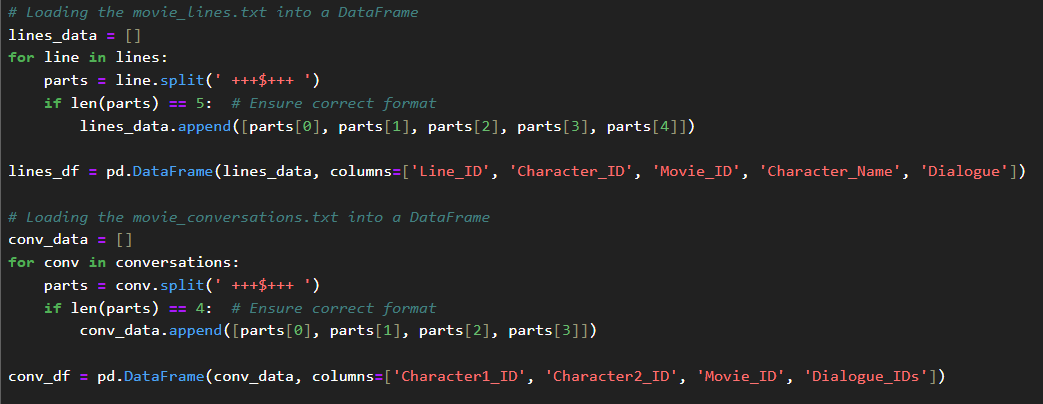
The development of a generative chatbot using a sample of the Cornell Movie Dialogues dataset proved successful within the given constraints. While the final model performs well for basic conversations, there are several areas where future improvements can be made, especially in terms of model complexity, resource allocation, and dataset size. The local web interface provides a working prototype, but expanding the system to handle larger-scale deployment and more complex conversations remains a key goal for future work.

# References And Links

* Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30, 5998-6008.
* Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
* [GitHub link for the project](https://github.com/ShruForAI/chatbot)
* [Project presentation video](https://www.youtube.com/watch?v=JreexdARQ90)

# Appendix

**Appendix A: Dataset**



Code 1. Dataset loading

**Appendix B: EDA**

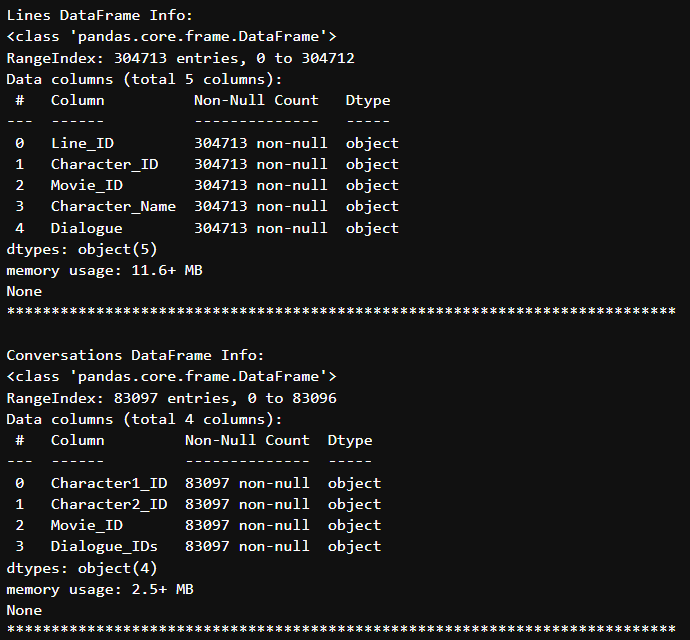


Fig 1. Info of the dataset movie\_lines.txt and movie\_conversation.txt

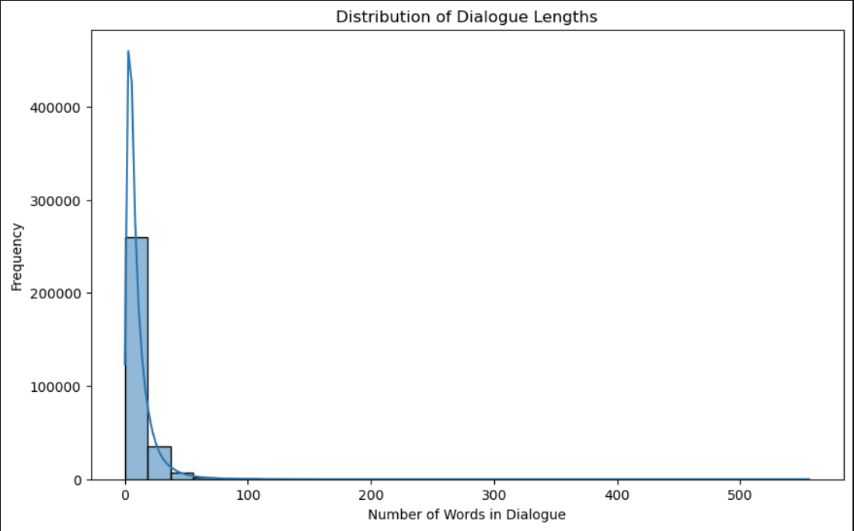


Fig 2. Distribution of the dialog lengths

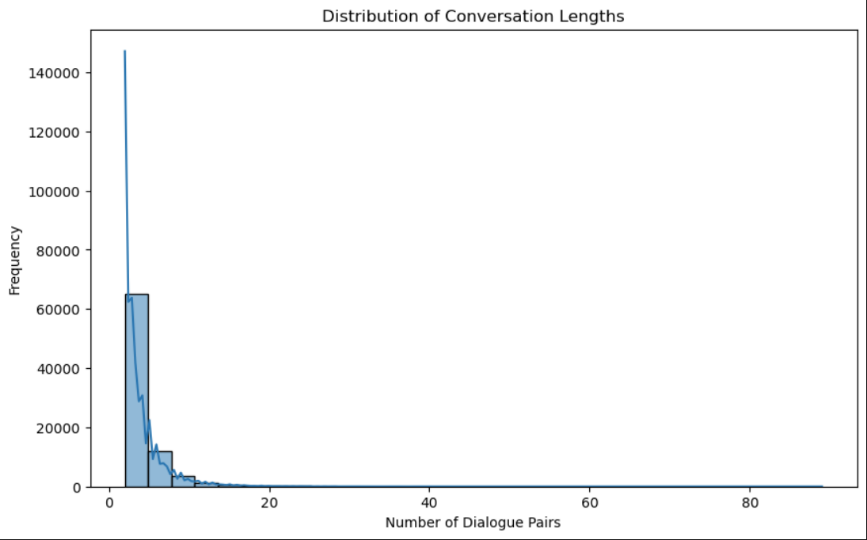


Fig 3. Distribution of the conversation lengths

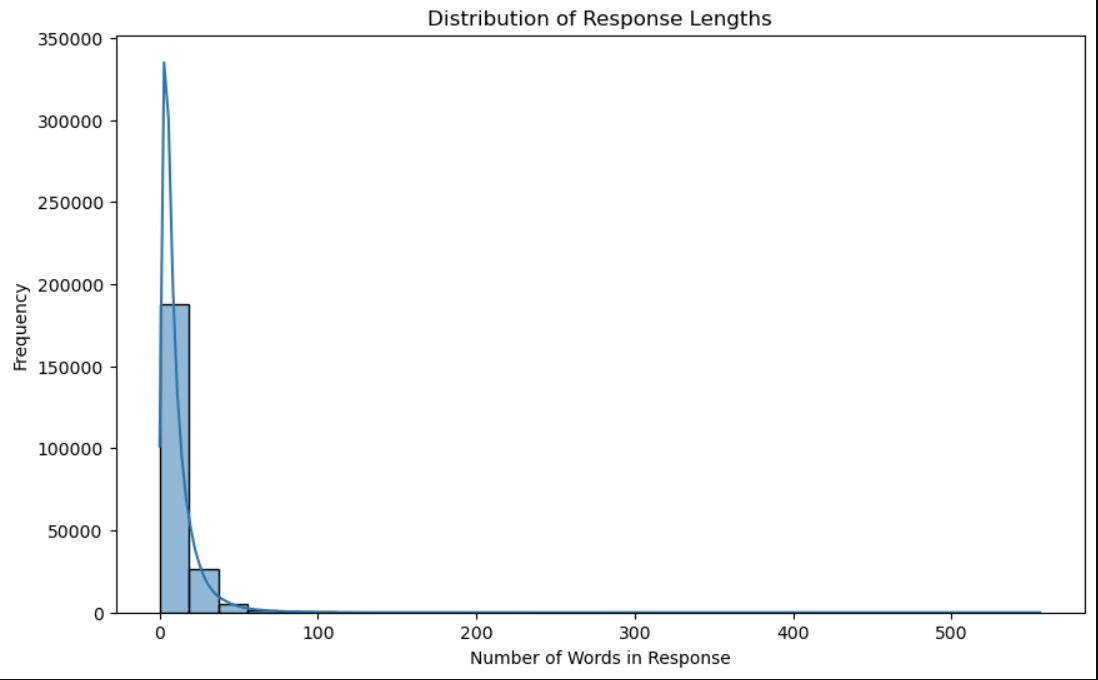


Fig 4. Distribution of the response lengths

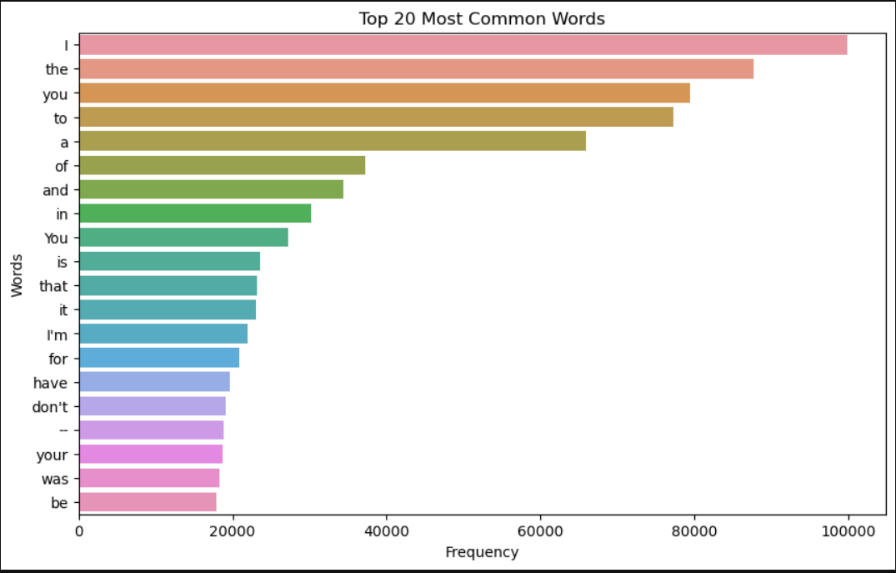


Fig 5. Top 20 common words

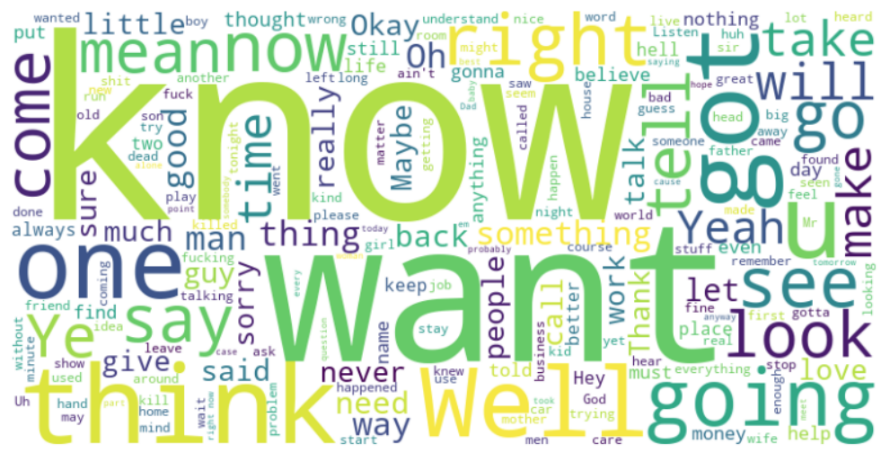


Fig 6. Word Cloud representation

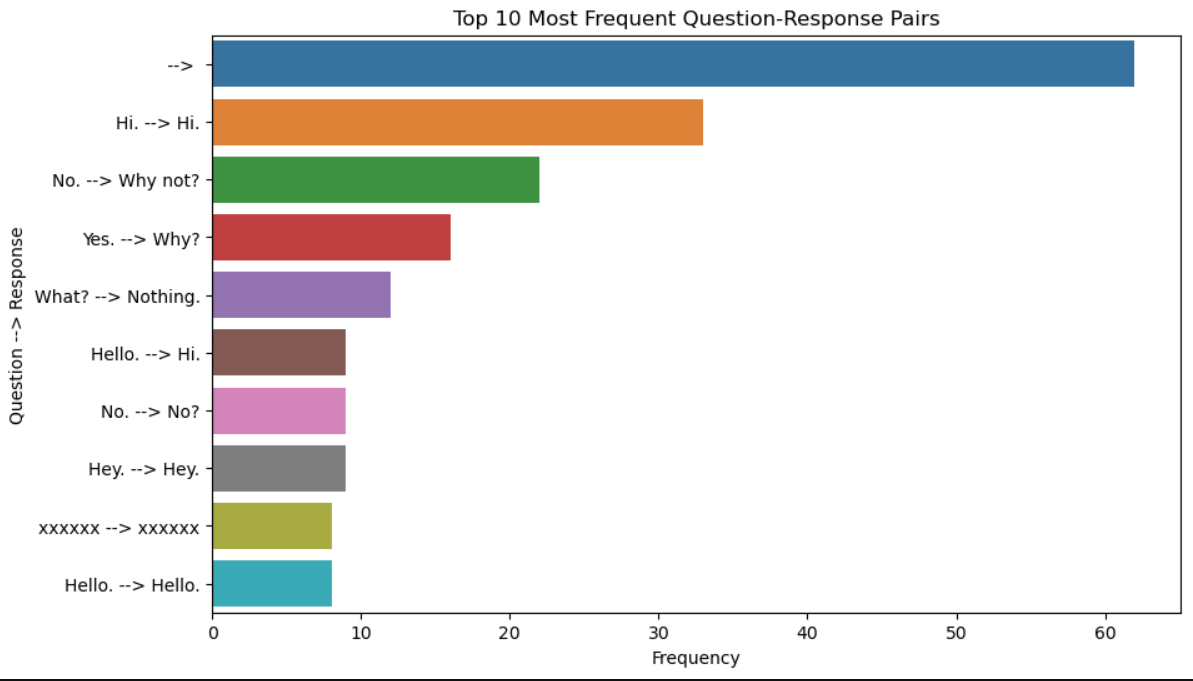


Fig 7. Top 10 question - response pairs

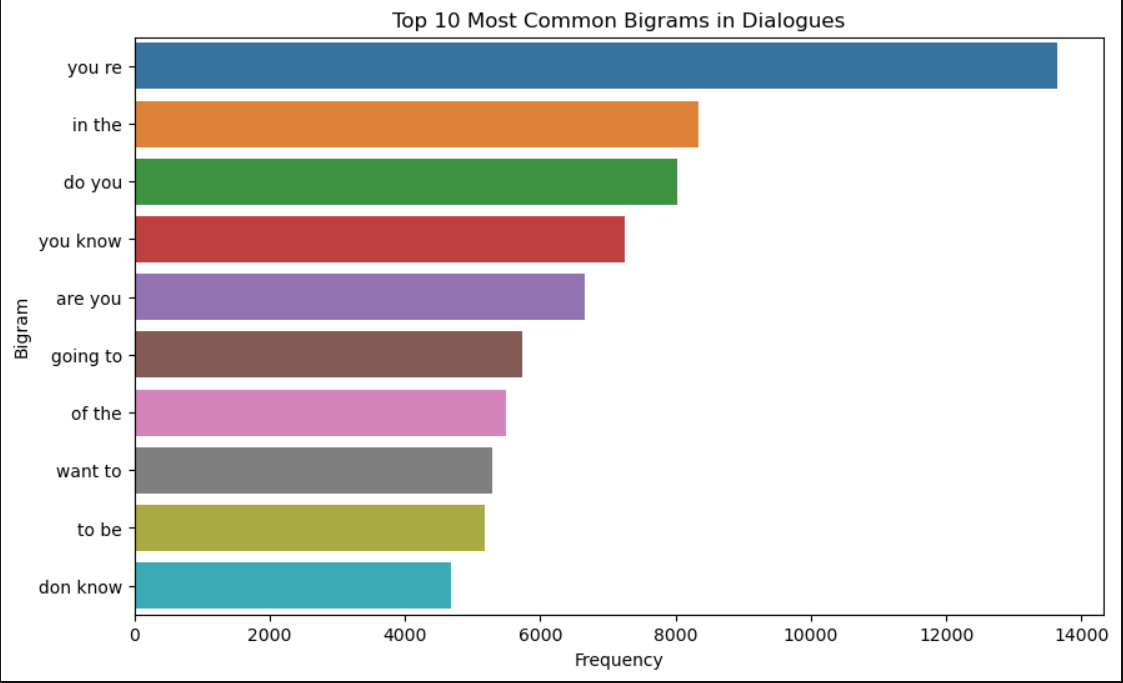


Fig 8. Top 10 common bigrams in dialogues

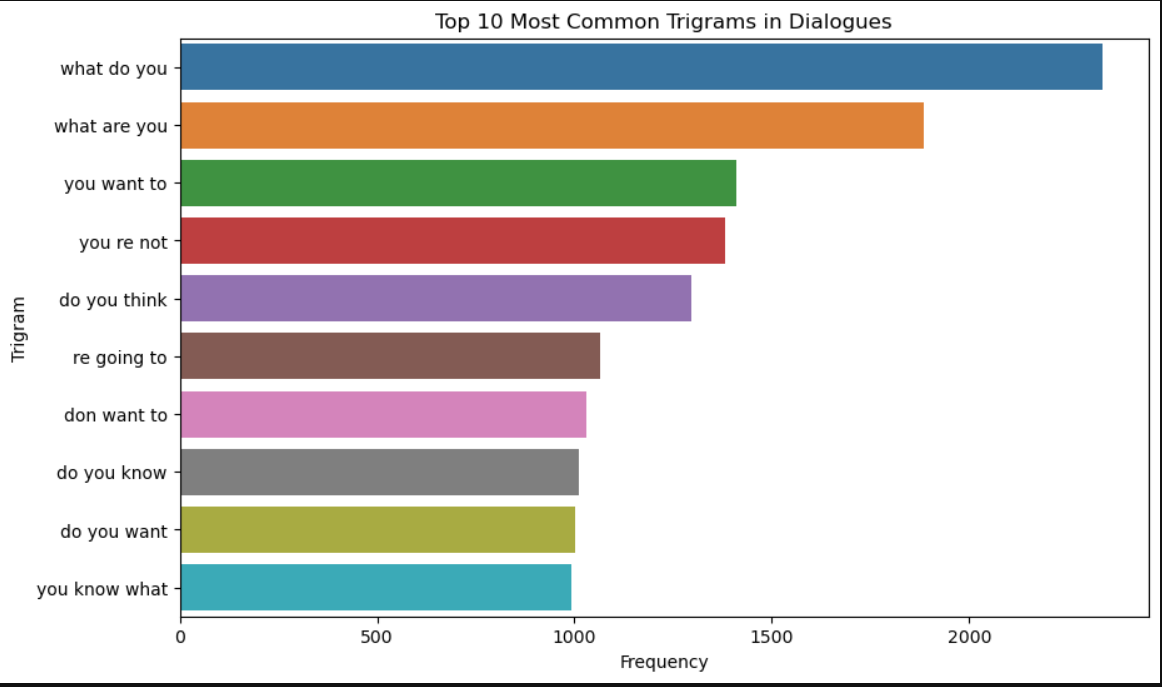
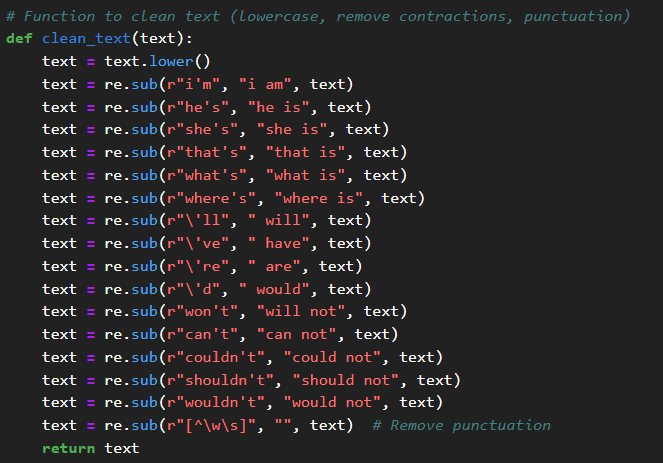


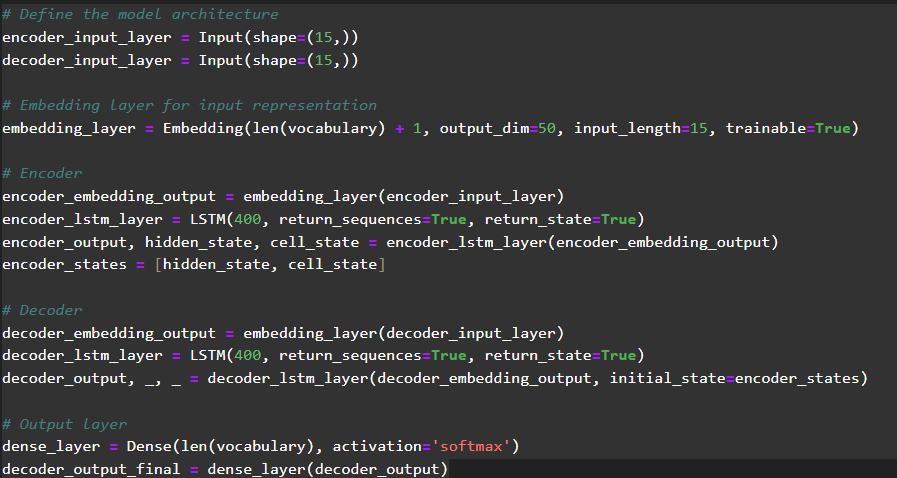
Fig 9. Top 10 common trigrams in dialogues

**Appendix C: Preprocessing**



Code 2. Function to clean text (Lowercase, Remove Contractions, Punctuations)

**Appendix D: Model**



Code 3. Model Architecture

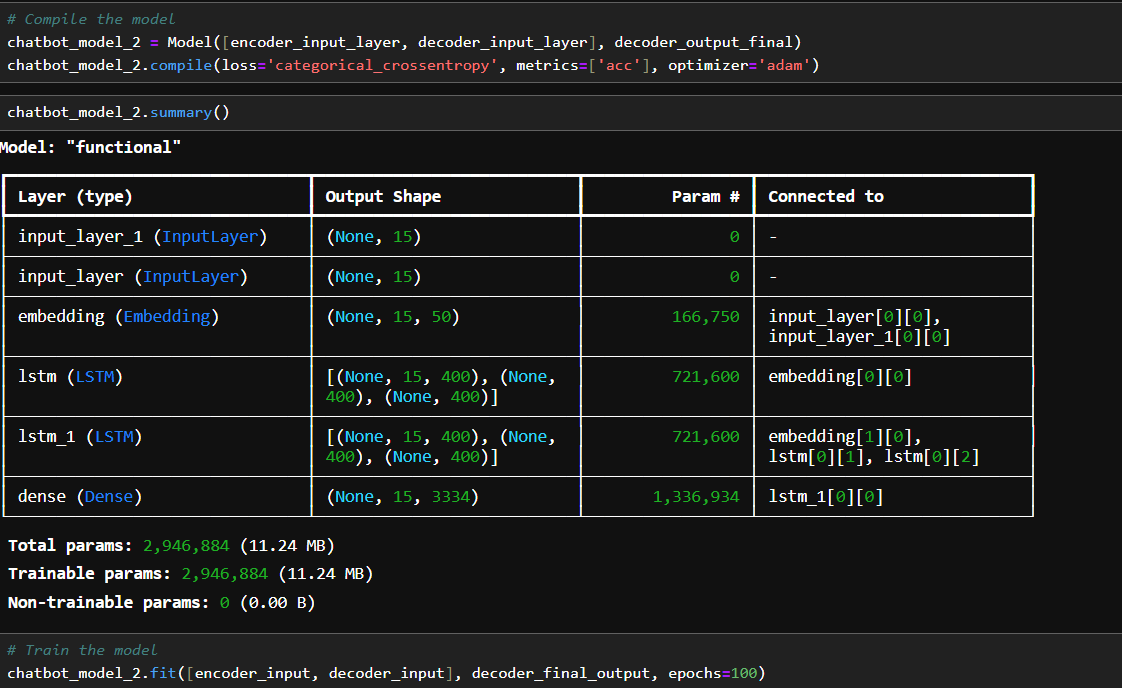


Fig 10. Model summary

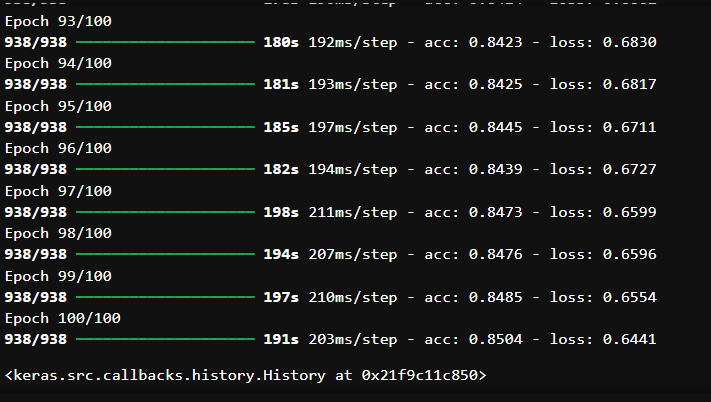
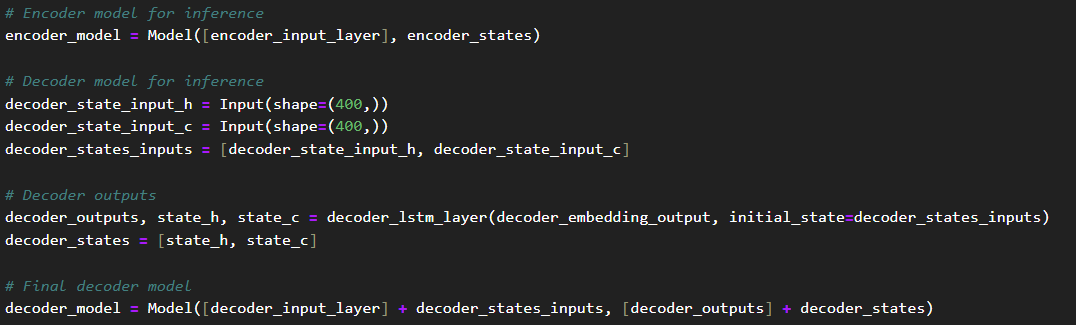


Fig 11. Results after 100 Epoch

**Appendix E: Inference Model**



Code 4. Inference Model

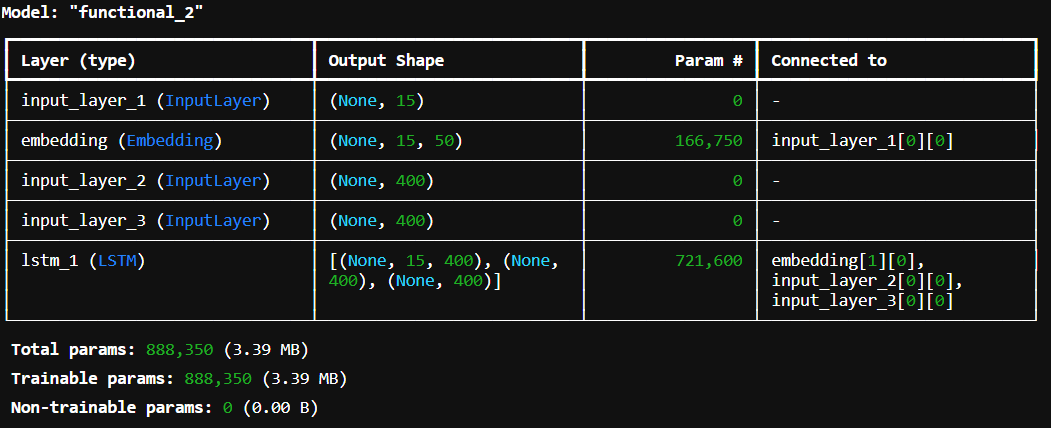


Fig 12. Inference Model Summary

**Appendix F: Conversation with Chatbot**

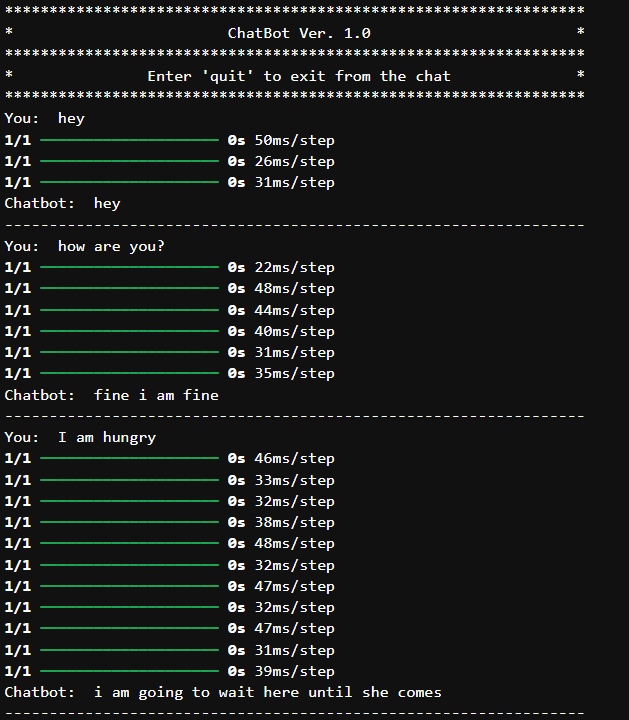


Fig 13.1. Conversation with chatbot

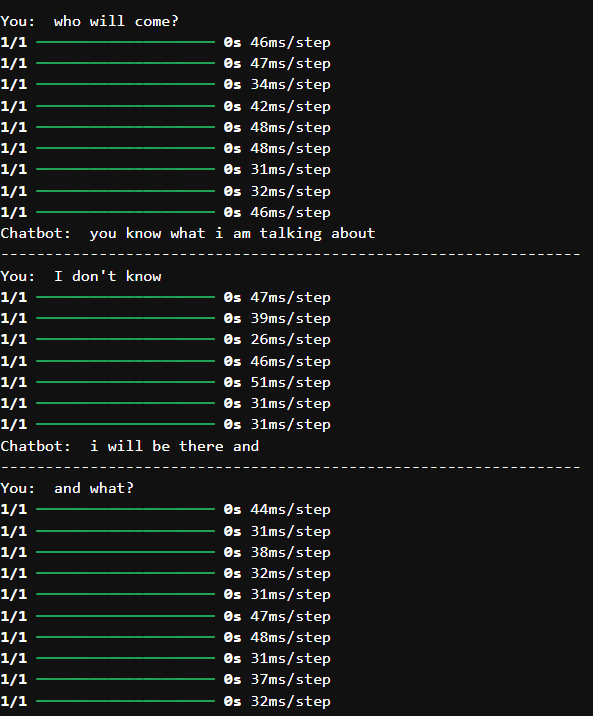


Fig 13.2. Conversation with chatbot continuation part 2

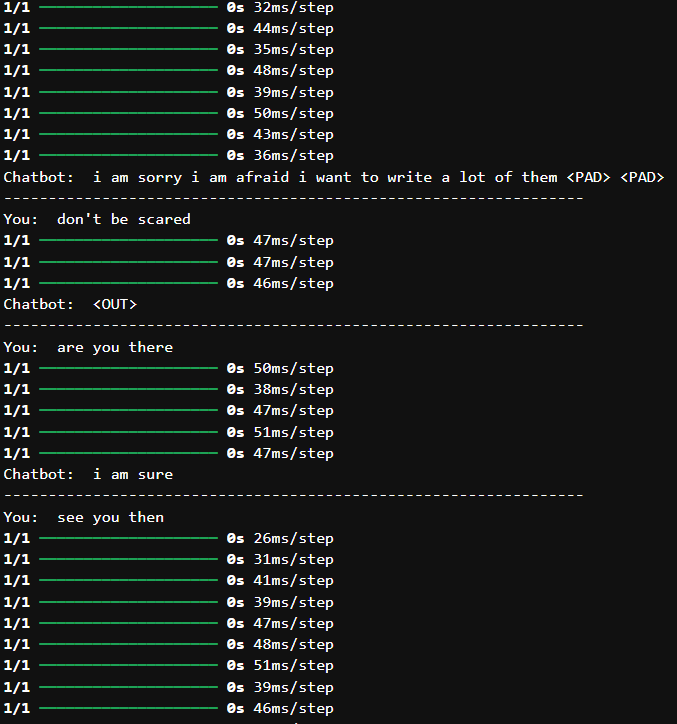


Fig 13.3. Conversation with chatbot continuation part 3

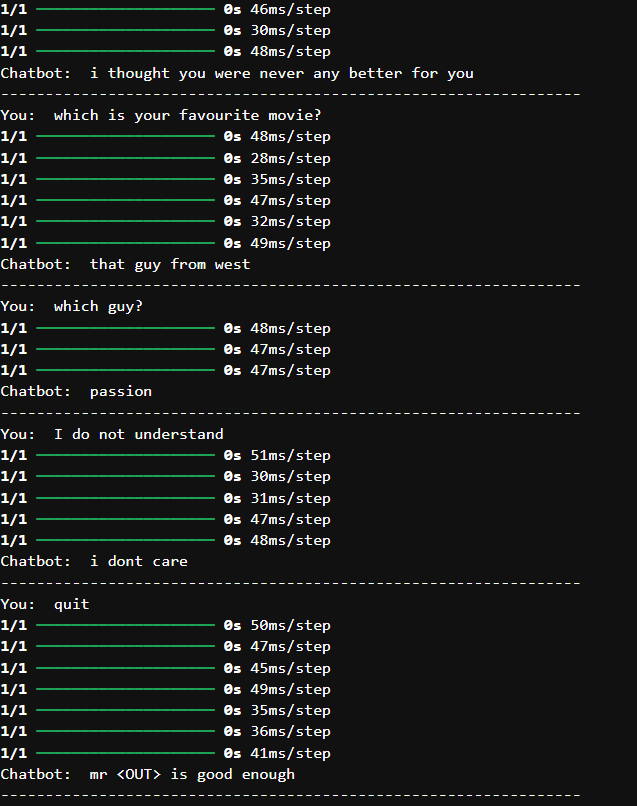
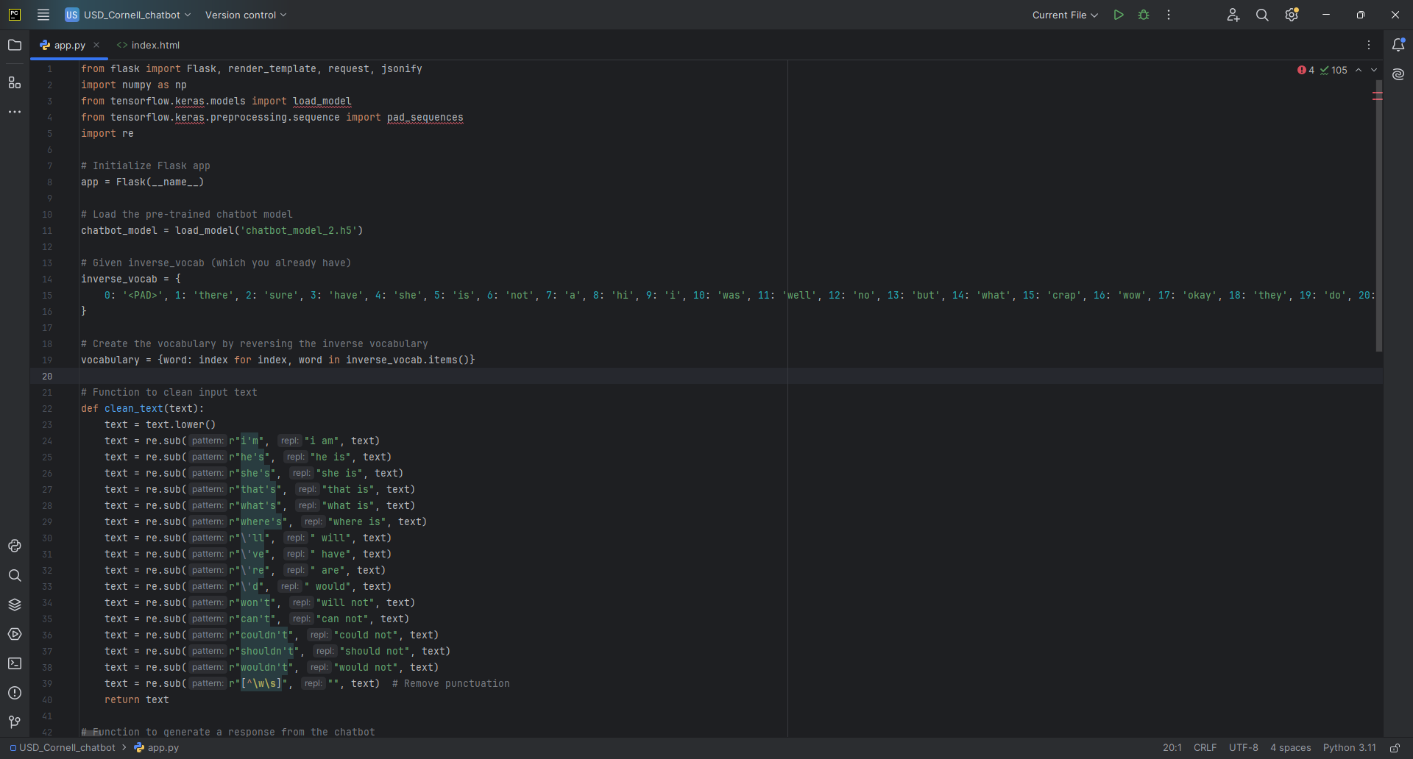


Fig 13.4. Conversation with chatbot continuation part 4

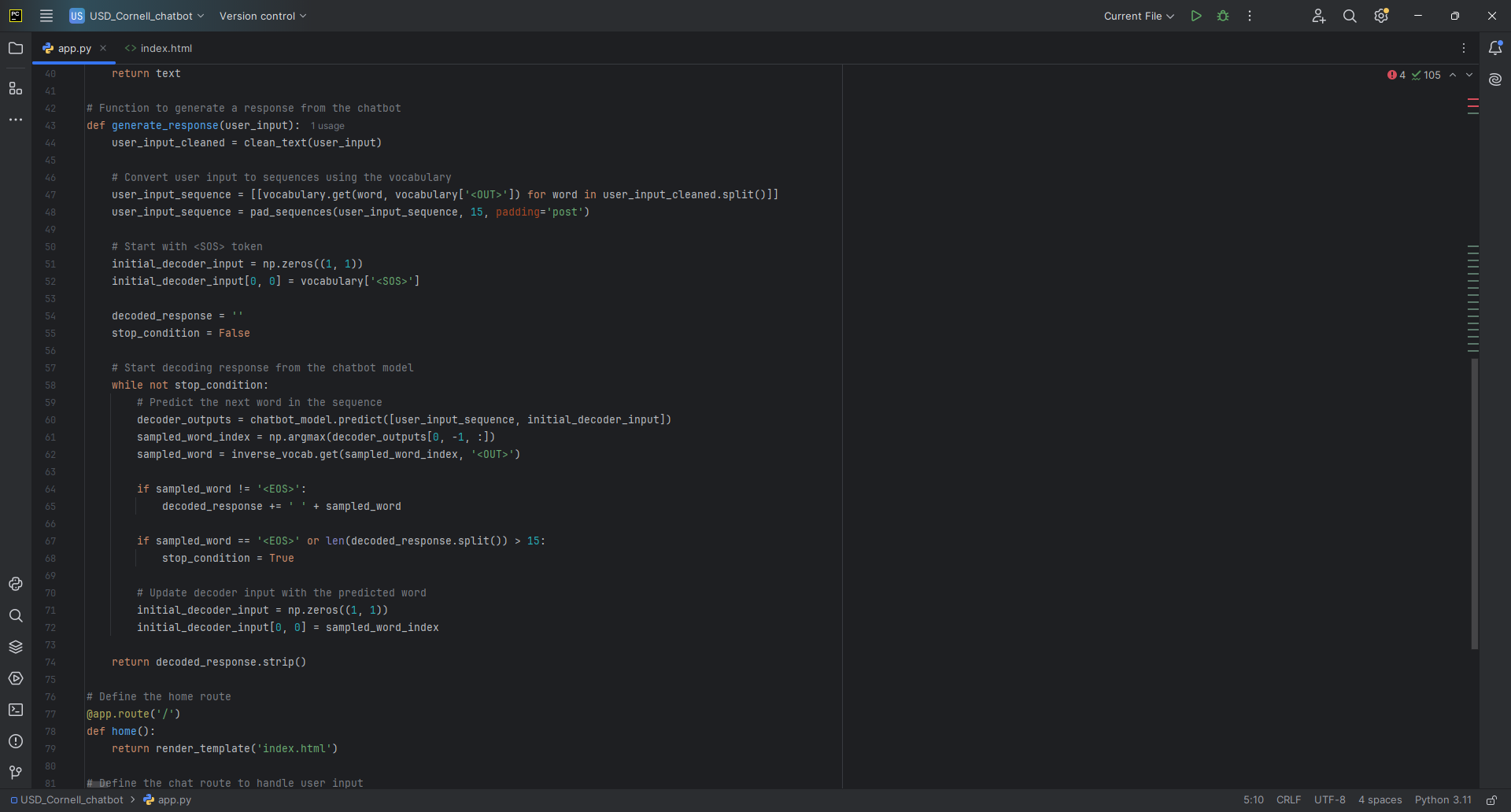
**Appendix G: Front End Using Html and Flask Application**



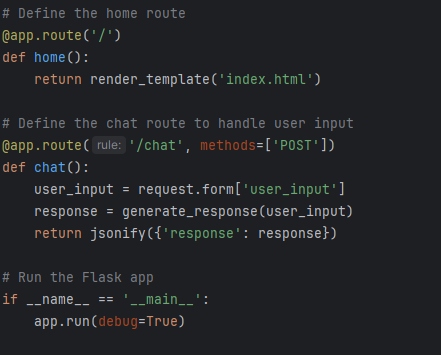
Code 5. Front end development



Code 6.1 Flask application part 1



Code 6.2 Flask application part 2



Code 6.3 Flask application part 3

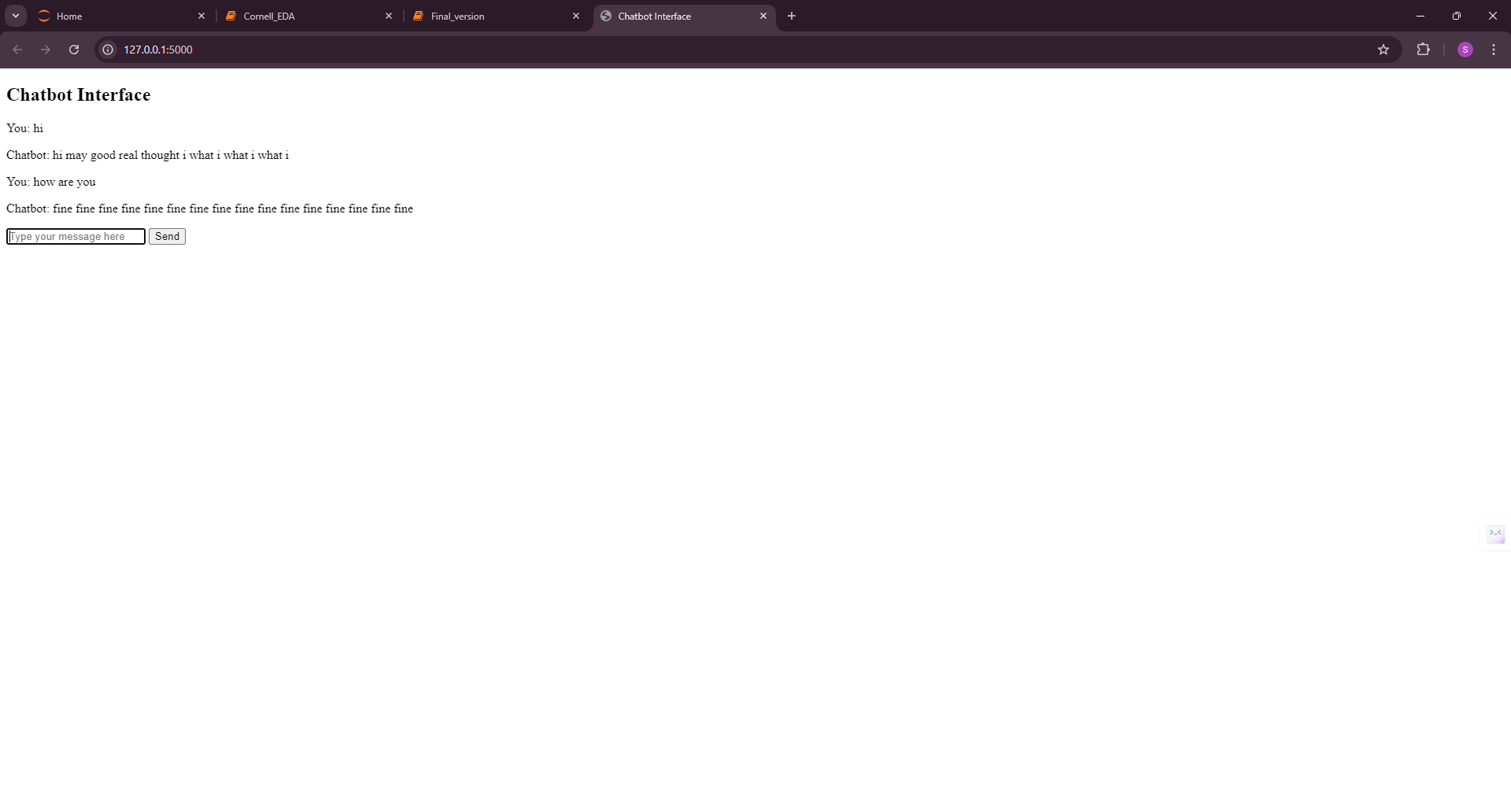


Fig 14. Chatbot in Front End Through Local Host (Additionally)