Comparative Analysis of "Friend" Style Chatbots Generated by ChatterBot and Seq2Seq Models

Zixi Zhou
23011611
04/12/2023

Introduction

In the field of Natural Language Processing (NLP) and Deep Learning (DL), chatbots have been a research direction of great interest. Based on the script dataset of the famous American drama "Friends", this project adopts the ChatterBot (a Python library dedicated to chatbot generation) engine and the Sequence-to-Sequence (Seq2Seq) model to respectively develop "Friends"-style chatbots, and compares the quality of generated dialogues of the two approaches on the same dataset, to evaluate the strengths and weaknesses of each of the two models for dialogue generation when the size of the training dataset is small.

Background

Chatbots are mostly designed for entertainment and casual conversation. To handle the complex structure inherent in human dialogue, machine learning has been used in the development process. Initially, these chatbots relied on retrieval methods due to the high computational requirements and large datasets needed to train the generative models. ChatterBot, a well-known dialogue generation engine, is about selecting the closest matching response by retrieving the known utterance that matches the input with the closest match [1]. As it is a library that comes with Python, it is very easy to implement.

However, the increased computational power of general purpose graphics processing units (GPGPU) has enabled the use of deep neural network (DNN) models to solve many AI problems. The training of generative NLP models has produced impressive results, generating text sequences that convincingly simulate human-computer dialogue [2]. Due to the need to process sequence data, chatbots rely on specific neural network architectures tailored for sequence processing, commonly referred to as Seq2Seq. Seq2Seq uses sequences as input and sequences as output. With the widespread use of DL techniques in NLP and the availability of a large number of pre-trained language models [3], Seq2Seq has become a standard architecture for processing a variety of NLP tasks, especially in the development of chatbots.

With the rapid development of artificial intelligence, chatbots are becoming more ubiquitous in people's daily lives. People want chatbots with distinctive personalities, which need to be built on datasets of different styles. However, such datasets tend to be small, such as the dialect of

a certain region, some specific movies and TV shows, the content of people's dialogue in a certain scenario, etc. Therefore, it is important to investigate the different performances of chatbots constructed based on the retrieval method (ChatterBot) and those constructed based on the deep neural network method (Seq2Seq) on actual small to medium-sized training datasets.

As one of the most popular television series, the unique linguistic style of “Friends” has become deeply embedded in popular culture, and its classic dialogues have become not only the common phrases used by Americans in their daily lives but also one of the most important references for many non-native English speakers around the world to learn English [4]. Therefore, in this project, the "Friends" script will be selected as the dataset and will be run and experimented with some pre-processing on two different models - ChatterBot and Seq2Seq models.

Method

Pre-processing of the dataset

The original dataset is a script with many unnecessary parts in the data. Therefore, the dataset needs to be pre-processed before it is used in training. This process is mainly based on the regular expression (regex) approach:

- Remove scene descriptions and voice-overs: Unnecessary scene descriptions and voice-overs exist in brackets, use regex to match the contents of brackets and replace them with empty strings.

```
cleaned_lines = []
for line in lines:
     pattern = r'\[[^]]*\]|\([^)]*\)'
     cleaned_line = re.sub(pattern, '', line)
     cleaned_lines.append(cleaned_line)
```

- Remove episode titles, writers' names and some scene-specific words: The titles of each episode are all capitalised, the format of the writer's name is the same for each episode, and there are three specific words 'Closing Credits', 'Commercial Break', and 'End' exist in each episode, use the string method to remove them.

```
for i in range(len(cleaned_lines)):
    if cleaned_lines[i].startswith('Written by:'):
        cleaned_lines[i] = ''
    elif cleaned_lines[i].isupper():
        cleaned_lines[i] = ''
    elif cleaned_lines[i].startswith('Closing Credits'):
        cleaned_lines[i] = ''
    elif cleaned_lines[i].startswith('Commercial Break'):
        cleaned_lines[i] = ''
    elif cleaned_lines[i].startswith('End'):
        cleaned_lines[i] = ''
```

- Remove character names: Character names exist at the beginning of each line, use the split ( ) method to split each line based on the colon, keeping only the part of the dialogue after the colon. Some of the lines in the dataset had two spaces following the colon, they need to be handled before.

```
for i in range(len(cleaned_lines)):
    if ': ' in cleaned_lines[i]:
        cleaned_lines[i] = cleaned_lines[i].split(': ', 1)[-1]
    elif ': ' in cleaned_lines[i]:
        cleaned_lines[i] = cleaned_lines[i].split(': ', 1)[-1]
```

- Remove blank lines and generate line-by-line text: There are blank lines with spaces and placeholders in the text, use strip ( ) to remove them.

```
for i in range(len(cleaned_lines)):
   if cleaned_lines[i].startswith(' '):
      cleaned_lines[i] = ''

final = []
for line in cleaned_lines:
   # code reference source:
    if line == '\n':
    line = line.strip('\n')
    final.append(line)
```

- Generate sentence pairs based on line-by-line text: Use a for loop to generate sentence pairs. There would be lots of space without strip ( ), the improvement comes from ChatGPT.

```
pairs = []
for i in range(len(lines) - 1):
    line1 = lines[i].strip()
    line2 = lines[i + 1].strip()
    pairs.append((line1, line2))
```

After the pre-processing above, the line-by-line text 'Friends_lines' is generated for ChatterBot training, and the sentence-pair text 'Friends_ pairs' is generated for Seq2Seq model training.

Training of chatbots

ChatterBot training model is sourced from ChatterBot: Build a Chatbot With Python (Martin Breuss) [5].

Seq2Seq training model is sourced from NLP-23-24/Week-9-Chatbots and LLMs [6]. I fine-tuned parameters such as hidden layer dimensions and the number of encoder and decoder layers to optimise the resulting performance.

Conception of experiments

ChatterBot is a machine learning algorithm and the Seq2Seq model is a sequence-to-sequence algorithm based on Recurrent Neural Networks (RNN). ChatterBot learns based on line-by-line text, while the Seq2Seq model learns based on pair of sentences.

When training the Seq2Seq model, the number of encoder and decoder layers was adjusted from the original 2 to 3 with 500 hidden layers, and 4000 training sessions were performed in each setting. This adjustment aimed to explore the impact of different layer numbers [7] on the model's performance.

During the experiment, six iconic lines from "Friends" [8] were selected as prompts (Table 1). These prompts were used to query seven different conditions (Table 2): the ChatterBot model, the Seq2Seq model with 2 layers in the encoder and decoder trained for 500, 1500, and 4000 epochs, and the Seq2Seq model with 3 layers in the encoder and decoder trained for 500, 1500, and 4000 epochs, respectively. Each question was posed three times, and the responses were recorded for analysis.

Prompts	Contents
1	How you doing?
2	Do you have a plan?
3	We were on a break!
4	I had a very long, hard day.
5	Could I be wearing any more clothes?
6	Well, maybe I don’t need your money, wait, wait, I said maybe!

Table 1: iconic lines from "Friends"

Experiments	Conditions
A	ChatterBot.
B	Seq2Seq: 2 layers in the encoder and decoder trained for 500 epochs.
C	Seq2Seq: 2 layers in the encoder and decoder trained for 1500 epochs.
D	Seq2Seq: 2 layers in the encoder and decoder trained for 4000 epochs.
E	Seq2Seq: 3 layers in the encoder and decoder trained for 500 epochs.
F	Seq2Seq: 3 layers in the encoder and decoder trained for 1500 epochs.
G	Seq2Seq: 3 layers in the encoder and decoder trained for 4000 epochs.

Table 2: Experiment plans

Based on the experimental results, compare and evaluate the characteristics of the two models in terms of answer relevance (whether the response relates to the question) and answer diversity (whether different responses are generated for the same question).

Results

The original screenshots of the experimental results are saved in the GitHub project folder.

Answer relevance

Rate the answers from each of the 7 models to each question on a five-point scale (Table 3).

	Prompt 1	Prompt 2	Prompt 3	Prompt 4	Prompt 5	Prompt 6	Average
A	4	5	5	5	3	2	4
B	2	2	3	2	2	2	2.2
C	4	3	3	2	2	3	2.8
D	4	4	4	2	3	3	3.3
E	2	3	3	3	3	2	2.6
F	3	3	2	4	2	4	3
G	4	4	5	4	2	3	3.6

Table 3: Answer relevance rate

Answer diversity

Record the times when each model answered differently to the same question, counting all three times the same as 1, once different and twice the same as 2, and all three times different as 3 (Table 4).

	Prompt 1	Prompt 2	Prompt 3	Prompt 4	Prompt 5	Prompt 6	Average
A	1	1	1	1	3	3	1.6
B	1	1	3	1	1	1	1.3
C	3	1	3	1	1	2	1.8
D	1	3	3	1	2	3	2.2
E	1	1	1	2	2	1	1.3
F	3	2	1	2	1	3	2
G	2	3	1	3	1	3	2.2

Table 4: Answer diversity statistic

Discussion

ChatterBot performs best in terms of answer relevance, but falls short in terms of answer diversity, reflecting the limitations of machine learning based on search methods. It can accurately answer known questions, but when questions are slightly modified, such as in prompts 5 and 6, where the original questions are shortened, ChatterBot tends to give similar answers. On the other hand, the Seq2Seq model shows relatively better overall performance in terms of answer diversity. As the number of training iterations increases, both answer relevance and diversity improve. Changing the number of encoder and decoder layers affects answer relevance to some extent, but has less impact on answer diversity.

In some specific cases, such as this project to create an entertaining Friends-style chatbot, it was necessary to retain some of the original dialogue from the series, as ChatterBot is excellent at providing the most accurate and expected responses and is much easier to implement. However, users do not always follow scripts when entering content. The Seq2Seq model can be trained to generate coherent and reasonably high-quality responses, although sometimes the content may not make complete sense. When building a chatbot, the use cases

and costs of both can be considered, and the best results can be achieved by combining the benefits of each.

Overall, the experimental results effectively show the characteristics of the two models. However, some problems were encountered during the experiments. Despite certain pre-processing steps performed on the original dataset, there were still names of people and places that only existed in the "Friends" series. As a result, instances of these names appeared in the actual answer of the Seq2Seq model (in Experiment G), indicating some degree of overfitting in the current model. The Seq2Seq model itself is very complex and its performance is affected by several parameters such as the number of hidden layers and the learning rate. The current experimental structure is relatively simple and therefore there is much room for improvement after gaining a deeper understanding of neural network principles. Finally, due to time and resource limitations, the model evaluation method lacks scientific rigour and the treatment of experimental data is more subjective. If more user tests can be included and the experimental results can be processed using statistical methods, it will provide more support for the experiment.

Conclusion

This project compares the performance of the ChatterBot and Seq2Seq models trained on a small dataset of “Friends” scripts. ChatterBot excels in answer relevance, while the Seq2Seq model performs better in answer diversity. When building entertainment chatbots, ChatterBot's ease of installation and low cost give it a huge advantage in specific cases, while the Seq2Seq model is more stable and applicable after sufficient training. Choosing the right development method based on actual needs can maximise the benefits of both retrieval-based and neural network-based dialogue generation models.

Ethical considerations

This project strictly adheres to the ACM Code of Ethics [9] in computing. The dataset used comes from the script of the American TV show "Friends" and is being used for academic research purposes only.

LLM disclaimer

A very small number of code blocks from ChatGPT were used in this project with appropriate comments in the source code. ChatGPT was used for debugging and troubleshooting during the project. Throughout the writing of this document, DeepL, ChatGPT and Grammarly were used to optimise the Chinese-to-English translation work.

Bibliography

[1] Python Software Foundation (2020) *ChatterBot PyPI*. Available at: https://pypi.org/project/ChatterBot/ (Accessed: 3/12/2023).

[2] Scotti, V., Sbattella, L., & Tedesco, R. (2023) ‘A Primer on Seq2Seq Models for Generative Chatbots’, *ACM Computing Surveys*, 56(3), pp. 1–58. Available at: https://doi.org/10.1145/3604281 (Accessed: 3/12/2023).

[3] Wang, H., Li, J., Wu, H., Hovy, E., & Sun, Y. (2023) ‘Pre-Trained Language Models and Their Applications’, *Engineering,* 25, pp. 51-65. Available at: https://doi.org/10.1016/j.eng.2022.04.024 (Accessed: 3/12/2023).

[4] 'Friends' (2023) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Friends (Accessed: 3/12/2023).

[5] Breuss, M. (2022) *ChatterBot: Build a Chatbot With Python.* Available at: https://realpython.com/build-a-chatbot-python-chatterbot/ (Accessed: 3/12/2023).

[6] Broad, T. (2023) *NLP-23-24/Week-9-Chatbots and LLMs.* Available at: https://git.arts.ac.uk/tbroad/NLP-23-24/tree/main/Week-9-Chatbots%20and%20LLMs (Accessed: 3/12/2023).

[7] Liu, X., Duh, K., Liu, L., & Gao, J. (2020). 'Very Deep Transformers for Neural Machine Translation', arXiv: 2008.07772v2 [cs.CL]. Available at: https://doi.org/10.48550/arXiv.2008.07772 (Accessed: 3/12/2023).

[8] Parade (2023) *Pivot Your Attention to These 67 Unforgettable Quotes From 'Friends'*. Available at: https://parade.com/1213691/alexandra-hurtado/friends-tv-show-quotes/ (Accessed: 3/12/2023).

[9] Association for Computing Machinery (2018) *ACM Code of Ethics and Professional Conduct.* Available at: https://www.acm.org/code-of-ethics (Accessed: 3/12/2023).