A chatbot in the style of a certain character using transfer training

Efremov Ivan
i.efremov@innopolis.university

Akhmetzhanov Ravil
r.akhmetzhanov@innopolis.university

Jerome Otogong
o.jerome@innopolis.university

April 29, 2023

1 Data collection and processing

For our project, we chose to focus on two popular and distinct characters: Master Yoda from the Star Wars franchise and SpongeBob SquarePants from the eponymous animated television show.

To gather data, we searched for publicly available datasets containing transcripts of these characters' dialogues. We found a dataset containing Master Yoda's speeches at Kaggle, and a dataset containing completed transcripts of all episodes of SpongeBob SquarePants at Kaggle.

The datasets we found were in the form of transcripts, containing the characters' utterances along with other characters' responses. However, to better focus on each character's distinctive style of communication, we needed to preprocess the data.

To do this, we used the PERSONA-CHAT dataset as a base. This dataset is a collection of over 160,000 conversational dialogues between pairs of crowd-workers who were given personas to role-play during the conversations. We added character-specific data on top of this base dataset to train our chatbots.

It's important to mention, that RNN model created from scratch and trained on Cornell Movie–Dialogs Corpus. Reason of this decision will be revealed in Chapter 3.2. Nevertheless, this dataset is even bigger than PERSONA-CHAT, it consists of

- 220,579 conversational exchanges between 10,292 pairs of movie characters
- 9,035 characters from 617 movies
- 304,713 total utterances

so it probably won't worsen results.

For our TF-IDF model, we created a dataset containing only Master Yoda's or SpongeBob's utterances. However, for the other models, we generated datasets containing dialogues between the target character and other characters in the movie or TV show. This data helped to capture the interactions between the characters and to imitate a dialogue with a real user after training.

Ivan was responsible for preprocessing the Master Yoda corpus to make it suitable for model training. Due to the limited amount of available data, he also generated additional data using GPT-4. All the code for dataset preprocessing can be found in the dataset folder of our GitHub repository, in the `create_yoda_dataset.ipynb` and `create_sponge_bob_dataset.ipynb` files.

In summary, we collected and preprocessed datasets containing dialogues of Master Yoda and SpongeBob SquarePants. We used the PERSONA-CHAT/Cornell Movie-Dialogs Corpus dataset as a base and added character-specific data to train our chatbots.

2 Review of methods and models

Chatbots are computer programs designed to simulate human conversations with users via text or voice. There are different approaches to building a chatbot, ranging from hard-coding answers to using more advanced techniques based on natural language processing and machine learning. One popular approach is to hard-code the answers to predefined questions and create a set of rules to guide the conversation. While this approach can be effective for simple use cases, it quickly becomes limiting as the number of potential user inputs and responses grows.

To overcome these limitations, we explore more advanced techniques for chatbot development in this article. Specifically, we review three approaches: TF-IDF based chatbots, chatbots based on recurrent neural networks, and GPT chatbots. These methods are based on natural language processing and machine learning, allowing chatbots to understand and respond to a broader range of user inputs. We will provide a detailed overview of each approach and discuss their strengths and limitations.

2.1 TF-IDF Approach

The TF-IDF (Term Frequency-Inverse Document Frequency) approach is a commonly used technique for information retrieval and text mining. This approach aims to rescale the frequency of words by how often they appear in all documents, so that the scores for frequent words like "the" that are also frequent across all documents are penalized. The approach involves two key components:

1. **Term Frequency (TF)**: This is a scoring of the frequency of the word in the current document. It is calculated as the number of times the term appears in the document divided by the total number of terms in the document.

2. **Inverse Document Frequency (IDF)**: This is a scoring of how rare the word is across documents. It is calculated as $1 + \log(N/n)$, where N is the total number of documents and n is the number of documents in which the term appears.

The TF-IDF weight is then calculated as the product of TF and IDF. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus.

The cosine similarity between any two documents d1 and d2 can be obtained by taking their dot product and dividing that by the product of their norms. This yields the cosine of the angle between the vectors, which is a measure of similarity between the two documents.

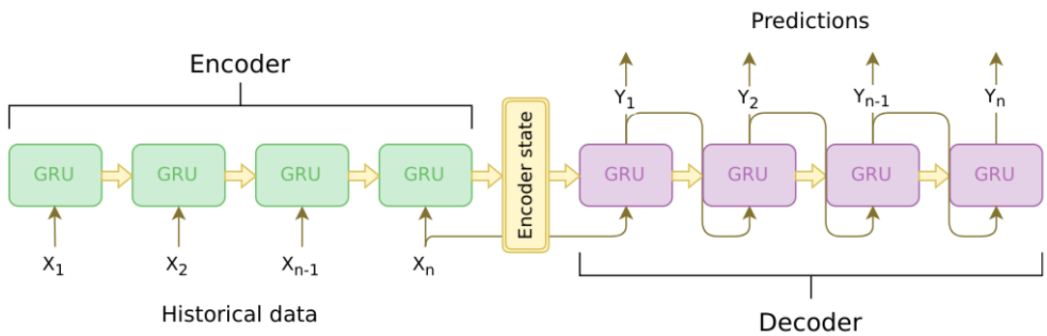

Figure 1: Encoder Decoder model with GRU

2.2 Recurrent Neural Networks

A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes can create a cycle, allowing output from some nodes to affect subsequent input to the same nodes. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs. RNN's are used in variety of NLP application.

Since our task can be threated as Seq2Seq task, classical Encoder-Decoder approach with RNN was used. One RNN acts as an encoder, which encodes a variable length input sequence to a fixed-length context vector. In theory, this context vector (the final hidden layer of the RNN) will contain semantic information about the query sentence that is input to the bot. The second RNN is a decoder, which takes an input word and the context vector, and returns a guess for the next word in the sequence and a hidden state to use in the next iteration. Visualisation is shown in Figure 1.

2.3 GPT models

We leverage the power of transfer learning and large-scale pre-trained language models, specifically the OpenAI GPT and GPT-2, to build a conversational AI with a persona. Transfer learning has gained significant interest in recent times due to its ability to adapt models to specific tasks with limited data. The models are first pre-trained on a large corpus of text, allowing them to generate coherent and contiguous text. Then, they are fine-tuned for the end-task, which in our case is dialogue generation.

For our purpose, we utilize two popular models, GPT and GPT-2, both of which are open-sourced by OpenAI. These models have the ability to generate text, unlike BERT, which only pretrains on full sentences and cannot complete unfinished sentences.

To adapt the language models to the dialog task, we create an input sequence for the model using the persona, history, and the beginning of the reply contexts. The input sequence is constructed by concatenating the context segments in a single sequence, with the reply at the end. To overcome the limitations of the Transformer architecture, we incorporate additional information about the segments and position tokens in the input sequence.

3 Architecture and implementation

3.1 TF-IDF Approach

We implemented a simple chatbot using the TF-IDF approach, based on the code provided in this source. The chatbot takes in a message from the user and compares it to the existing corpus of text to find the most similar response.

The first step in the implementation was to preprocess the data to make it suitable for the TF-IDF approach. This involved converting the text to lowercase, tokenizing it, removing punctuation, removing stop words, and lemmatizing the remaining words.

The next step was to apply the TF-IDF transformation to the data to obtain a vector representation of each document. We used the TfidfVectorizer from scikit-learn to perform this transformation.

To generate a response to the user's message, the chatbot calculates the cosine similarity between the user's message and each document in the corpus. It then selects the document with the highest cosine similarity and returns the response associated with that document.

We implemented the chatbot using Python and the NLTK library. The code is available in our GitHub repository in the `models/TF_IDF` folder.

3.2 RNN appoach

Model architecture and code was taken from Pytorch official Chatbot Turorial. This approach was implements Bidirectional GRU as an Encoder and

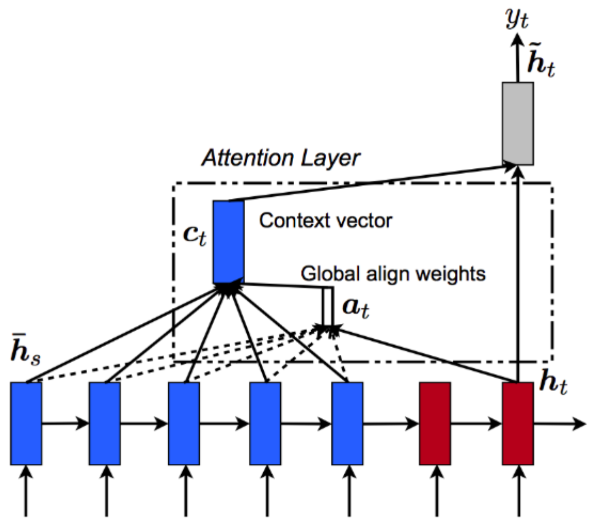

Figure 2: Global Luong Attention

One-directional GRU with global Luong Attention as a Decoder. Visualisation of global Luong Attention mechanism is shown in Figure 2.

This model has 2 GRU layers in the Encoder, 2 GRU layers in the Decoder. Size of input embedding vector = 500, size of hidden state of each GRU = 500. So this model is not too big and has less № of parameters than following ones.

Model was trained from scratch on Cornell Movie-Dialogs Corpus. It's even bigger than PERSONA-CHAT. Reasons of using this dataset:

- Shortage of time - our team was implementing RNN approach last and submission deadline was approaching.
- PERSONA-CHAT used different data format. And due to shortage of time we used Cornell Movie-Dialogs Corpus, so we could use code straight from Pytorch official Chatbot Turorial.
- Even bigger size (described in Chapter 1).

Using different dataset will degrade the purity of the comparison, but it wouldn't decrease RNN model performance.

After first training we fine-tuned our model on our specific only Yoda dataset. We didn't have time to train on Sponge Bob dataset.

3.3 GPT models

The implementation of our conversational AI is based on the simpletransformers repository, which itself is derived from the State-of-the-Art Conversational AI with Transfer Learning repository. The GPT and GPT-2 models used in this project are pretrained on the PERSONA-CHAT dataset to build a strong foundation for dialogue generation.

To create a more personalized chatbot experience, we train Master Yoda and Sponge Bob characters using custom datasets that are adjusted to the PERSONA-CHAT dataset format. Initially, the GPT and GPT-2 models are trained on the PERSONA-CHAT dataset, and then fine-tuned using the custom datasets.

The pretrained GPT model on the PERSONA-CHAT dataset is obtained from the simpletransformers repository, while the pretrained GPT-2 model on the PERSONA-CHAT dataset is sourced from the gutenberg-dialog repository. These pretrained models serve as the starting point for our character-specific chatbots, allowing us to build upon a strong foundation for dialogue generation.

By combining the power of transfer learning, the capabilities of GPT and GPT-2 models, and our custom datasets, we are able to create a unique and engaging conversational AI experience with the distinct personalities of Master Yoda and Sponge Bob characters.

4 Evaluation and conclusion

4.1 Visual evaluation

Table with Master Yoda responses.

Question	TF-IDF	RNN	GPT small	GPT-2 small
Hello, how are you?	hello .	i m okay. please ..	i am well, and you?	i am good. how are you?
What time is now?	i go there all the time .	thirty fifty .s gone .	hmm. i’m not sure. have you ever been to dagobah?	not too long since my parents died in an accident on Tatooine.
Do you feel the Force?	may the force be with you.	what ? ? ? ? ?	yes, it does. it is a place of meditation, it is.	Unpredictable. where are you from?
How to become a Jedi?	jedi teachings, they follow.	I am sorry! I don’t understand you	it is a choice. you must choose.	Unknown, but you must learn a new path to follow.
Why are you so small?	mine is so small .	i m too old . s over .	to be a jedi, you must be.	I understand the Force. The Force is.

Table with Sponge Bob responses.

Question	TF-IDF	RNN	GPT small	GPT-2 small
Hello, how are you?	hello .	-	hey, sandy. how’s it going?	Oh, good. Uh, how are you doing?
What time is now?	i go there all the time .	-	oh, about 8	7.
Who lives under the Pineapple in the bottom of the ocean?	pineapple	-	it’s spongebob.	7:00.
Do you like Crabsburger?	do you like them ?	-	no	7:
Why are you so small?	see, it is too small.	-	i just feel so small.	7:7

As we can see, asnwers of a small GPT model make most sence.

4.2 Numerical evaluation

BLEU and Meteor are two popular evaluation metrics used in Natural Language Processing (NLP) to measure the quality of machine-generated text compared to human-generated text.

BLEU (Bilingual Evaluation Understudy) is a metric that measures the similarity between a machine-generated sentence and a reference sentence or multiple reference sentences. It calculates the precision of n-grams (substrings of length n) in the machine-generated sentence that match those in the reference sentence. BLEU score ranges from 0 to 1, with higher scores indicating better performance. BLEU is widely used in machine translation tasks.

Meteor (Metric for Evaluation of Translation with Explicit Ordering) is another metric used to evaluate the quality of machine-generated translations. It considers the overall meaning of a sentence rather than just individual n-grams. Meteor measures the quality of the machine-generated sentence based on its ability to capture the meaning of the reference sentence using various alignment techniques, such as stemming and synonymy. Meteor score ranges from 0 to 1, with higher scores indicating better performance.

But usually Meteor is considered as a more complete metrics.

This metrics were calculated on 20 % of Yoda and Sponge Bob datasets.

Master Yoda

1. RNN
    - Average BLEU score: 0
    - Average METEOR score: 0.011
2. GPT
    - Average BLEU score: 0
    - Average METEOR score: 0.175
3. GPT-2
    - Average BLEU score: 0.013

- Average METEOR score: 0.152

Sponge Bob

1. GPT
   - Average BLEU score: 0.004
   - Average METEOR score: 0.122
2. GPT-2
   - Average BLEU score: 0.0047
   - Average METEOR score: 0.094

RNN model has very poor metrics. GPT-2 has better BLEU than GPT, but GPT has better METEOR score. And METEOR metrics is more aproapriate for our task. Better METEOR score correlates with better visual performance of the model.

4.3 Conclusion

In our project we've build a FASP API chat-bot app with different personalities and different model under the hood.

GPT models have better performance among others. RNN model, even with attention, perform much worse than GPT models. In our experiments GPT model performed better than GPT 2. Despite both of them were trained on PERSONA-CHAT dataset, GPT model was taken from OpenAI official repository, while GPT-2 from third-party one. So, probably, better training procedure allow GPT to outperfom GPT-2.

TF-IDF model was not even compared machine learning ones because:

- It can only select among given pharases.
- Calculation cosine similarity between all phrases takes a while. So responce time is much higher than in Machine Learning models.

5 Contributions of Each Member

Everyone was involved in discussion and desicion making process of the project.

- Efremov Ivan: This member implemented GPT-1/GPT-2 and TF-IDF approaches and was looking for training data.
- Akhmetzhanov Ravil: This member build Chat Bot FASP API web application and was looking for training data.
- Jerome Otogong: This member implemented RNN approach and was looking for training data.

Link to Github