Robotics Club IITKanpur

Summer Projects
Science and Technology Council
IIT Kanpur

Conversational Robot

Documentation

Date of Submission:

July 24, 2020

Contents

1 Acknowledgements

We feel obliged in taking the opportunity to sincerely thank Robotics Club, IIT Kanpur and Science & Technology Council, IIT Kanpur for furnishing us with the Summer Project in these unprecedented times. We thank our mentors Paras Mittal and Ashwin Shenai and the Co-ordinators of Robotics Club: Prakash Choudhary, Ramyata Pate, Suman Singha and Yatish Sharma for their invaluable guidance and readiness to help, which motivated us to conclude this project punctually.

2 Introduction

2.1 Aim

The aim is to develop a "talking" bot, one that can listen to the user, pay attention to his/her intent and make a meaningful reply, as if you are talking to a human being. The conversational bot would converse with a user keeping in mind factors of a conversation like intent, context and then generate corresponding response. The main goal is to create an end to end conversational machine learning based model.

2.1.1 Problem Statement

The primary objective remains to implement the most basic bot that can meet our expectations and understand how to use natural language processing for creating a satisfactory response as per users' intent.
When time permits, we may upgrade to a superior version of the talking bot.
The problem of talking bot implementation can be broken down into 4 major parts:

- Converting speech to machine-understandable text.
- Processing text to infer intent, emotion, nature of the response(positive, negative or neutral) or some other useful feature.
- Generating a reply based on the inferred features.
- Making an audio reply.

3 Plan of Action

Initially the 25 students in the project were further divided into groups of 5 in order to promote a healthy discussion environment amongst the members.
The approach to go about the whole project was quite straightforward: The project was mainly divided into sub tasks which formed the components that build up the final project with few modifications along the way and the last few days were focused more towards improving the chat bot pipeline and the integration of all these different components.
Throughout the project various reading assignments were provided which gave the students skills and tools required to complete the programming assignments with an in depth understanding of the core concepts and finally achieve the main goal of building a Conversational Robot.

4 Assignments

4.1 Reading Assignments

4.1.1 Coursera Courses

- Networks and Deep Learning
- Sequence Models
- Natural Language Processing in Tensorflow

4.1.2 Research Papers

- Deep Speech 2: End to End speech recognition.
- Topic Aware Neural Response Generation
- Neural Response Generation via GAN with an Approximate Embedding Layer

4.2 Programming Assignments

4.2.1 AIML ChatBot

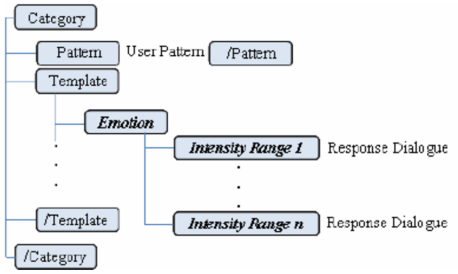


Figure 1: AIML bot logic tree

Implemented a basic customer care chatbot using AIML. AIML stands for Artificial Intelligence Modelling Language. AIML is an XML based markup language meant to create artificial intelligent applications. (1)
AIML makes it possible to create human interfaces while keeping the implementation simple to program, easy to understand and highly maintainable.
This was a peer-graded assignment done individually by the group members.

4.2.2 Word Embedding

We implemented a model (Glove or some version of Word2vec) to train word embeddings and used this trained embedding layer to train an IMDB review classifier with fewer data and iterations. (transfer learning) (2)

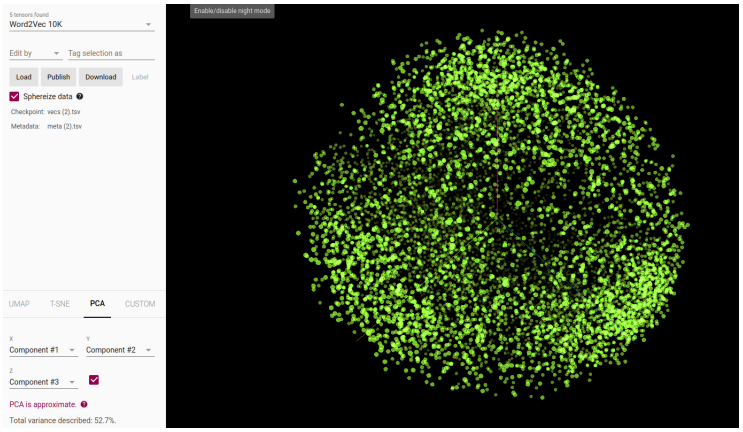


Figure 2: Tensorflow projector

4.2.3 Speech Recognition

Speech recognition is an interdisciplinary sub-field of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers.
It is also known as automatic speech recognition (ASR), computer speech recognition or speech to text (STT). It incorporates knowledge and research in the computer science, linguistics and computer engineering fields.

4.2.4 Response Generation

Generated a meaningful response to user input text using Topic Aware Seq2Seq model. At first we focused on the context aware attention model and then extraction of topics using the LDA Algorithm which were further combined to form the joint Attention model.

4.2.5 Integration

This marked as the last of the major programming assignments wherein we focused on bringing it all together, combine the speech recognition, reply generating, and voice output(using gTTS API) layers to complete the talking bot.

5 Implementation

We implement the final chat bot which would be able to take input from microphone and generate an audio response.

5.1 Speech Recognition

This is the first component of the final bot where we need to take audio input from the user and convert it into text which can be further fed into the model.

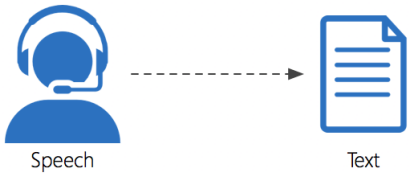


Figure 3: Speech Recognition

5.1.1 Ideation

The typical approach to a speech processing task is to use a deep learning component which is either a CNN or RNN followed by a time consistency component. Now one of the main issue is it's really hard to predict what's in a (say 10ms) frame without context and thereby the need arises for the second time consistency component

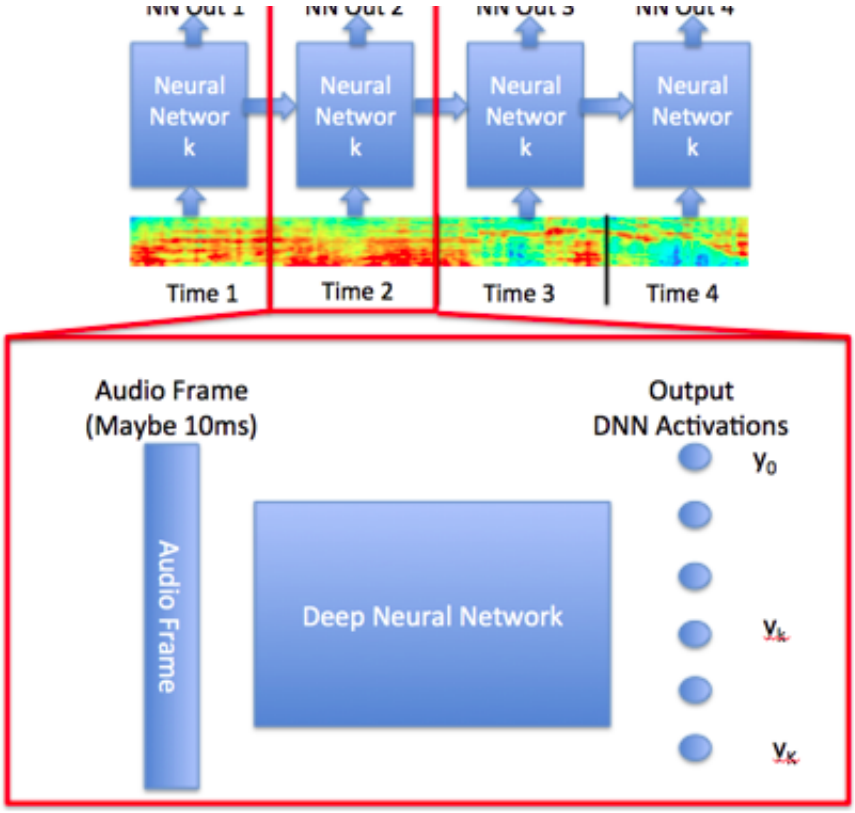


Figure 4: Frame by frame prediction

. **Need for Context**: In the diagram we see that at each time step, the neural network predicts an output (a vector y) with each dimension of y indicating the probability of the utterance of a phoneme. But the phoneme can be incorrect quite easily depending upon accents and manner of speaking certain words. This is where context plays an important role with the general idea being to predict the **sequence** of outputs that make the most sense instead of predicting each frame at a time. So we just need to predict the best path.

Another issue is to prevent contribution from phonemes appearing twice (say "/ii/" in "-h-eii-ii-ihh-") to the meaning of the phrase. This is becomes important in examples like —-bbb—eee– which in reality is should be predicted as just be. This is where the ***CTC Loss*** comes in. Instead of just optimizing for the best path, it also takes care of optimizing the labels.(3)

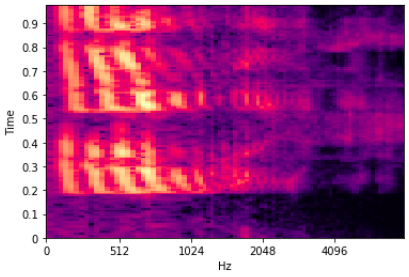
5.1.2 Data Processing

We explored the LibreSpeech dataset which had the utterances of single words. Data processing included converting the audio files into spectrograms and data augmentation to increase efficiency of our data-set and robustness of the system.

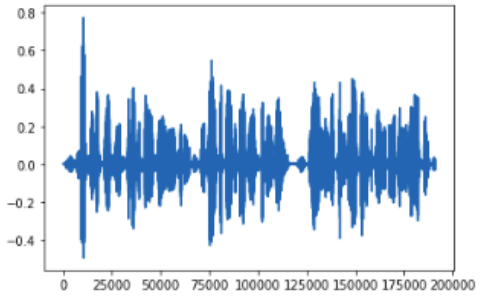
Spectrograms were formed with the help of librosa library from the input .wav file.

```
librosa.display.specshow(make_spec(keys[0]),
                         x_axis='mel',
                         fmax=8000,
                         y_axis='time',
                         sr = 16000,
                         hop_length = 128)
```

```
/usr/local/lib/python3.6/dist-packages/librosa/core/s
amplitude_to_db was called on complex input so phase
d. To suppress this warning, call amplitude_to_db(np.
  warnings.warn('amplitude_to_db was called on comple
<matplotlib.axes._subplots.AxesSubplot at 0x7fa9531b2
```



(a) Librosa spectrogram

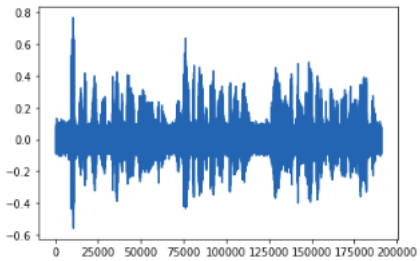


(b) Post noise addition

Figure 5: Spectrograms

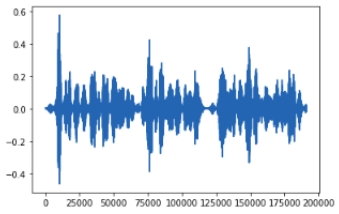
Data Augmentation was done by adding some noise and experimenting a bit with the pitch.(4)

```
def noise_inj(audio, noise_factor):      #introducing
    data, sr =sf.read(audio)
    noise = np.random.randn(len(data))
    aug_data=data + noise_factor * noise
    aug_data = aug_data.astype(type(data[0]))
#     sf.write("mod_aud.flac", aug_data, sr)
    return gen_audio(aug_data, sr)


noise_inj(keys[4], 0.03)
```



(a) Noise injection

```
def change_pitch(audio, pitch_factor):
    data,sr=sf.read(audio)
    return gen_audio(librosa.effects.pitch_shift(data, sr, pitch_factor),sr)

change_pitch(keys[4],0.5)
```



(b) Pitch change

Figure 6: Data Augmentation

Data Generator class was used to generate batches of data for training instead of feeding everything as a single batch because of the lack of memory to process data as whole. Now the overall architecture of our program after implementing the data generator becomes like the following figure.(5)

```python
import numpy as np

from keras.models import Sequential
from my_classes import DataGenerator

# Parameters
params = {'dim': (32,32,32),
          'batch_size': 64,
          'n_classes': 6,
          'n_channels': 1,
          'shuffle': True}

# Datasets
partition = # IDs
labels = # Labels

# Generators
training_generator = DataGenerator(partition['train'], labels, **params)
validation_generator = DataGenerator(partition['validation'], labels, **params)

# Design model
model = Sequential()
[...] # Architecture
model.compile()

# Train model on dataset
model.fit_generator(generator=training_generator,
                    validation_data=validation_generator,
                    use_multiprocessing=True,
                    workers=6)
```

Figure 7: Keras script

5.1.3 Model Architecture

In the core of DS2 system being used is a recurrent neural network (RNN) trained to take spectrograms as input and generate text transcriptions. The general architecture includes one or more convolutional layers, followed by one or more recurrent layers (bidirectional), and one or more fully connected layers.

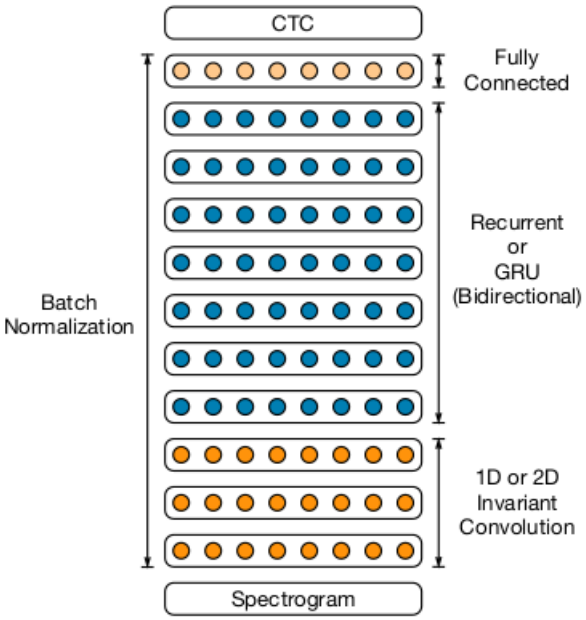


Figure 8: Frame by frame prediction

This model is trained using the CTC loss function.(6) Given the input pair *(x,y* and the parameters of the network (theta) , we compute the loss function *L(x,y;theta* and using it's derivative we update the parameters through back-propagation.

5.2 Topic Modelling

Topic Modelling is a precursor step to generating topic aware responses. In this, we use statistical modelling to generate topics from a corpus. It becomes far more important when we work with very large datasets as it helps the model generalise quicker by adding in topic information.(7)

5.2.1 LDA

We used Latent Dirichlet Allocation for Topic Modelling. LDA assumes documents are composed of topics, which in turn are composed of words. These words which make up a topic are generated using their probability distribution. A popular Natural Language Processing library Gensim had been used to train our own LDA Model.

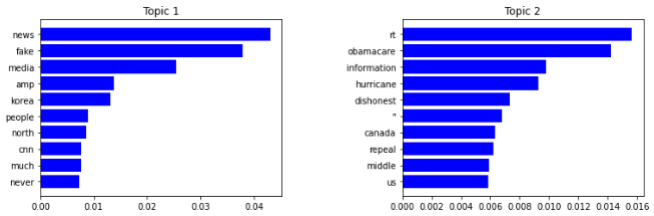


Figure 9: Some topics and their topic words

In our project, we use an LDA Model trained on Twitter Corpus.(8) This allows us to identify the topic of an input to the response generator. In essence, this makes the Response Generator more 'aware' of the input it has received.

5.3 Response Generation

The second step in our pipeline is generating conversational responses after we have recognised input speech content. We tried two distinct response generation models trained on a subset of OpenSubtitles Dataset.

5.3.1 Seq2Seq with Message Attention

This works like a machine translation model which is trained on input, response pairs. The main components of this are an Encoder Model, an Attention Mechanism, and a Decoder Model.
Encoder reads X word by word and represents it as a context vector c through a recurrent neural network, and then the decoder estimates the generation probability of Y with c as input. By adding in attention, each word in Y corresponds to a context vector c , and c is a weighted average of all hidden states of the encoder.
Lastly, a Beam Search Decoder is used to find optimal responses for inference.

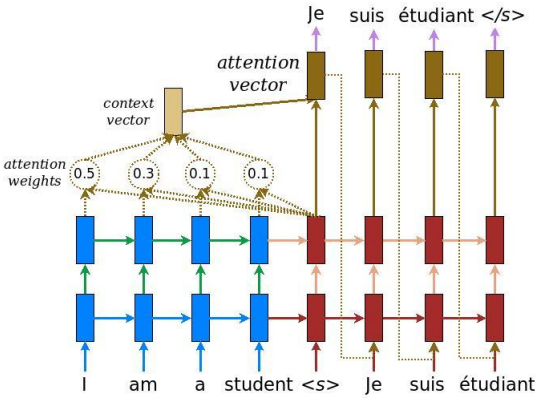


Figure 10: Structure of Seq2Seq with Attention

Encoder: This is composed of an embedding layer followed by a GRU layer. The embedding layer uses a GloVe embedding. This generates a context vector for input X which can then be passed to Decoder.
Attention: The traditional Seq2Seq model assumes that every word is generated from the same context vector. In practice, however, different words in Y could be semantically related to different parts of X, hence we introduce an Attention Mechanism.
Decoder: This is composed of an embedding layer followed by a GRU layer. The embedding layer uses a GloVe embedding. Using the Attention Weights and Context vector, it generates candidate words from which Sparse Categorical Cross Entropy loss is calculated.

5.3.2 Topic Aware Seq2Seq with Message Attention

The key addition in this over the previous model is the join attention mechanism. This uses both the Topic Information and the Message Attention while deciding candidate words. To accomplish this, we predict topic words for each input message using our trained LDA Model. We lookup embedding for topic words and then linearly combine them to form a Topic Attention vector. This is used with the Message Attention, to make the responses not only relevant to the message but also to the topic of conversation.

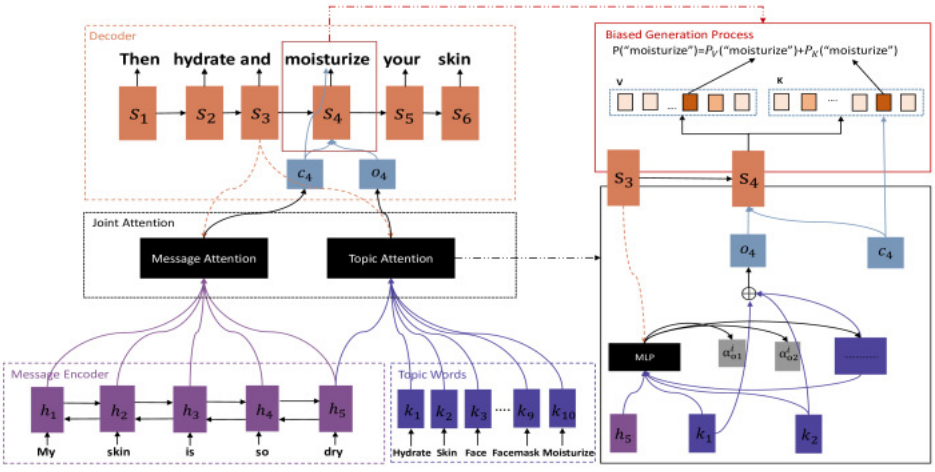


Figure 11: Structure of TA-Seq2Seq

5.4 Audio Response

To make the bot more human like we wanted it to respond using audio and so we converted the generated response to an audio response using the Google-text-to-Speech (**gTTS**) python library. The sample code is as follows

```
from gtts import gTTS

# This module is imported so that we can
# play the converted audio
import os

# The text that you want to convert to audio
mytext = 'Welcome to geeksforgeeks!'

# Language in which you want to convert
language = 'en'

# Passing the text and language to the engine,
# here we have marked slow=False. Which tells
# the module that the converted audio should
# have a high speed
myobj = gTTS(text=mytext, lang=language, slow=

# Saving the converted audio in a mp3 file nam
# welcome
myobj.save("welcome.mp3")

# Playing the converted file
os.system("mpg321 welcome.mp3")
```

Figure 12: Text to Audio

5.5 Issues Faced

Issues faced in implementation are listed below:

- **Small dataset:** Due to limited internet resources, training on very large dataset was not possible. Due to this, the Response Generation deep model could not properly inherit the properties of the full dataset. Same goes for the Speech Recognition model.
- **Time constraint on training:** To train large datasets, Google colab was used which has time constraints. Therefore it was very difficult to train models which take large time to train, like the speech-to-text model.
- **Topic awareness:** The incorporation of topic awareness in the Response Generation model using Latent Dirichlet Allocation did not improve the responses without using topic biasing.

6 Results

Table 1: Speech to Text

	Latency
Group A	2.5s
Group B	3.3s
Group C	3s
Group D	4s

Table 2: Response Generation

Inputs	Group A	Group B	Group C	Group D
Are you a good person?	I am a men, Don Sanchez	no, sorry	i don't know	its unlikely
It is going to be hard.	I thought so	you don't say	oh	the mail is war
But i guess that is how life is.	I know man	we have our old man	i understand	wait
Where are you going?	I don't own a proposal	Where am on duty at the hospital	What say	I'm going on duty
Do you like cats or dogs?	i like you	I'm sorry, no late supper	Maybe if you say	very profound

Table 3: Overall Latency of end-to-end process

	Latency (end-to-end process)
Group A	4s
Group B	4.279 s
Group C	7s
Group D	8s

The group repositories can be found here:

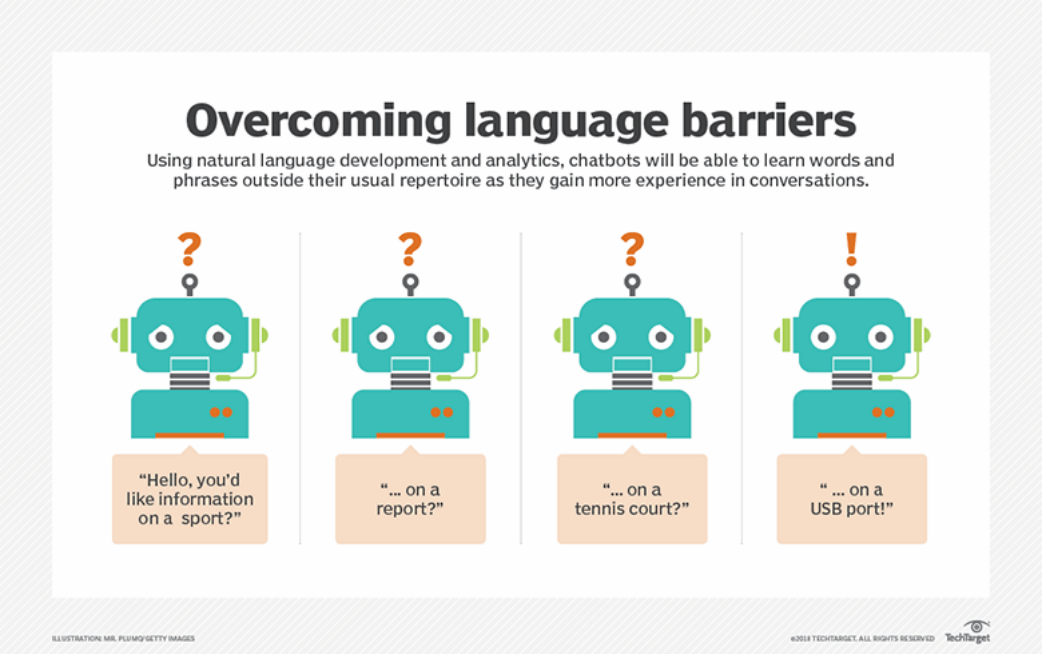
- Group A
- Group B
- Group C
- Group D

7 Future Scope

As per the Global Market insights, the overall market size of chatbots worldwide would be over 1.3 billion dollars by 2024 hence it is inevitable that chatbots would become more intelligent,

more human like and become the driving force of business communications and the face of interactions between a user and company services.

The future scope of chatbots begins by addresing the current flaws like removing language barriers and more proper procurement of data to help the machine understand the relationships between words, phrases and enable it to mimic human-like natural language. Also there is a lot of scope of improvement in their conversational ability. The chatbot must be able to (till some extent) exhibit qualities like patience, intelligence and flexibility as in a human service agent.



Some of the areas where chatbots are gaining popularity :

- Use in contact centers
- Messaging platforms
- Voice bots
- Payment automation

On a personal level after a bit of discussion amongst ourselves we would try to implement this for personal use in which the task is to feed in our say facebook messages or whatsapp texts as the training data so that the chatbot can in some way speak like us and then make it reply to our conversations and have some fun ofcourse!

Bibliography

[1] AIML bot. Website. [Online] https://www.tutorialspoint.com/aiml/index.htm.

[2] Embedding projector. Website. [Online] http://projector.tensorflow.org/.

[3] Baidu Research – Silicon Valley AI Lab. Deep speech 2: End-to-end speech recognition in english and mandarin. 2015.

[4] Data-augmentation. Website. [Online] https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6.

[5] Data-generator. Website. [Online] https://stanford.edu/~shervine/blog/keras-how-to-generate-data-on-the-fly.

[6] CTC-loss. Website. [Online] https://stackoverflow.com/questions/57292896/understanding-ctc-loss-for-speech-recognition-in-keras.

[7] Yu Wu Jie Liu Yalou Huang Ming Zhou Wei-Ying Ma Chen Xing, Wei Wu. Topic aware neural response generation. 2016.

[8] LDA model. Website. [Online] https://arxiv.org/pdf/1608.02519.pdf.