Logo

Description automatically generated with medium confidence

**Module Title:** Human-Robot Interaction

**Module Code:** EE4705

**Project No.:** 2

**Title of Lab Exercise:** Project 2 Natural Language Processing

**Module Group No.:** EE4705-6

**Student Contributed to This Assignment:**

| Admission Number | Name |
| --- | --- |
| A0184893L | Ng Wei Jie, Brandon |
| A220837H | Zheng Jiezhi |
| A0248558M | Zhao Hengyuan |

**Date of Lab Exercise:** 25th February 2022

**Module Lecturer:** Professor Shuzhi Sam Ge

| **Task** | **Name** |
| --- | --- |
| **i** | **Zheng Jiezhi** |
| **ii** | **Ng Wei Jie, Branodn** |
| **iii** | **Zhao Hengyuan** |
| **iv** | **All** |
| **v** | **All** |

**1. Look for references or web resources based on your interests and learn how to design your dialogue system involving natural language processing.**

**Background of Natural Language Processing**

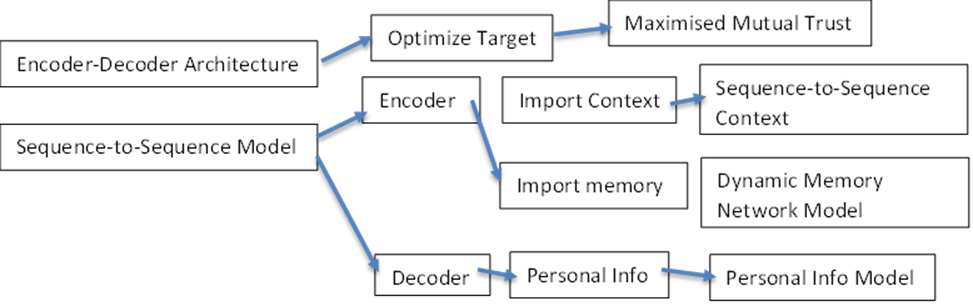
Natural Language Processing (NLP) is an important direction in the field of computer science and artificial intelligence. It studies various theories and methods that can realize effective communication between humans and computers using natural language. Natural language processing is a science that integrates linguistics, computer science, and mathematics. Therefore, research in this field will involve natural language, the language that people use on a daily basis, so it is closely related to the study of linguistics, but with important differences. Natural language processing is not the general study of natural language, but the development of computer systems that can effectively realize natural language communication, especially the software systems therein. Hence it is part of computer science.

**Generative Mode of Chatbot**

Our group decided to use a generative model for chatbots. The generative dialogue systems are usually done using a deep learning-based Encoder-Decoder architecture [1]. Technologies based on deep learning usually do not rely on a specific answer library or template, but conduct dialogues based on the language ability learned from a large number of corpora. The method of directly generating answers based on question content is defined as generating a Model. The Seq2Seq technology of deep learning can implement the framework of generative models very well. Its most attractive advantage is the end-to-end framework that can avoid artificial feature engineering, that is, using powerful computing and abstraction capabilities to automatically summarize and extract valuable knowledge and features for solving problems from massive data sources, making this process transparent to the problem solver, thereby avoiding the uncertainty and heavy workload brought by artificial feature engineering.

**Architecture of Generative Model of Chatbot**

RNN-based Seq2Seq model [2] completes the generative dialogue models of logic, correctness or contextual coherence with the improved methods for optimisation.



**Word-Level RNN Encoder-Decoder Model**

With the aid of neural machine translation (NMT) and neural responding conversation (NRC), it can easily translate, this model is widely applied to Sequence-to-Sequence tasks. Given a word-level source sequence (denoted as message) and a target sequence (denoted as response), an EncDec model maximizes the following conditional probability [2].

Encoder is supposed to create the embedding vector sequence. Afterwards Encoder employs a recurrent neural network (RNN) to summarize the input into a context vector. Then, the hidden state of decoder’s RNN at time step and the corresponding embedding vector generated at time step for the objective function.

EncDec is often being implemented with an Attention mechanism. The Attentive Decoder utilizes a dynamic context vector to estimate rather than a fixed. More specially, the outcome is a weighted average of all Encoder's hidden states..

DLC Encoder [2]

DLCEncoder directly operates at character-level without word segmentation. DLCEncoder first extracts the n-gram character features by a convolutional layer and four highway network layers, then DLCEncoder concatenates the input character embeddings and the extracted n-gram character features. Finally, the concatenated representation is fed to a bi-directional LSTM layer.

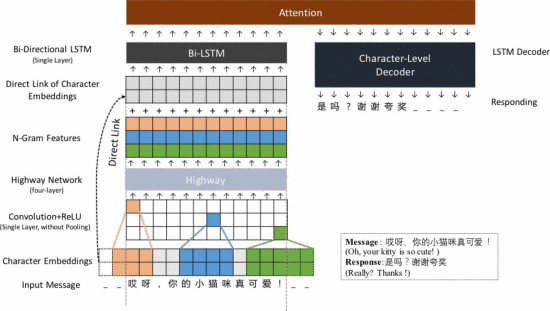
The character embeddings of an input are regarded as the fine-grained features, because each embedding only aligns one character itself. Correspondingly, the extracted n-gram features are denoted as coarse features because each extracted feature could be aligned to characters.

a) Character Embedding [2]

DLCEncoder maintains a vocabulary that records unique characters and a corresponding embedding matrix. Thus, a character sequence could be transformed into a character embedding sequence.

b) Convolution Without Pooling [2]

A convolutional layer is utilized to capture the n-gram character features. Specially, for a filter of width , the height is fixed as the dimension of a character embedding. The input is first padded to, then a narrow convolution is applied between and to obtain a feature map.



**2. Look for appropriate datasets and models (or pre-trained datasets, pre-trained models), depending on the topic of your dialogue system. Try playing around with the arguments in models or pre-trained models to obtain better samples.**

**Topic Of Dialogue System**

Given the ongoing Covid-19 pandemic, this project’s dialogue system targets to answer public questions on the current state of Covid-19 in Singapore. This Covid-19 chatbot will address common questions, reducing the workload on the call centers. Having been involved in building the Covid-19 chatbot for the government, below were the pain points with the process, and this project aims to explore solutions to address them.

1. Update intents and train model when government updates FAQs weekly is time-consuming
2. Add alternative ways the public would phrase the FAQ intent is troublesome and mundane
3. Create new intents for every information is time-consuming and reduces chatbots accuracy

**NLP Models**

To address the pain points, three different models are explored. The first model is a text classification model. To respond to the user correctly, the model needs to classify the utterance’s intent. Figure 2.1 shows that the text classification model uses BERT as the pretrained model to leverage on transfer learning to improve the chatbot performance. The output of the BERT module is a vector representation of the utterance and is fed to a linear layer to classify the intent. The vector representation encodes the semantic meaning of the utterance. [4] A softmax is applied to get the model confidence of the prediction.

self.bert = BertModel.from\_pretrained("bert-base-cased")

self.linear = nn.Linear(768, n\_classes)

self.sm = nn.Softmax(dim=1)

Figure 2.1 shows the model for text classification

Other methods of text classification such as training a model on one-hot representations of the utterances were attempted. However, this method did not perform as well when using BERT as the transformer. Besides, these methods require the model to train from scratch; hence it requires a lot of data and time. It is not feasible to update the chatbot and train when the dataset will be changed frequently.

Figure 2.2 shows how BERT tokenizes the text by first adding [CLS] to the start of the text by default, separating the sentence by [SEP], and splitting and adding hashes for words that cannot be found in BERT’s ~30k tokens. After the text is passed to the tokenizer, the output contains input\_ids that represent the indices of the words in the transformer dictionary and attention\_masks that has 1 and 0 to indicate if a word is present in the sentence or is padded. Passing the input\_ids and attention\_masks to the BERT module gives the embeddings of the utterance. [4]

# original text

text = "Here is the sentence I want embeddings for."

# tokenized text

['[CLS]', 'here', 'is', 'the', 'sentence', 'i', 'want', 'em', '##bed', '##ding', '##s', 'for', '.', '[SEP]']

Figure 2.2 shows the tokenize text of user utterances

The second model is a question-answer model to extract information from context. Figure 2.3 shows how the pretrained models trained on Squad dataset are used to fine tune for domain transfer to this project. [5, 9] Attempts to train the model from scratch using the Squad dataset were made but the process took 9hrs+. Therefore, the attempts were not continued. However, training the model on a smaller dataset (the covid dataset described in the next section with 2019 data points) did not perform as well. This demonstrates the usefulness of pretrained models to speed up the training process.

model\_name = "deepset/roberta-base-squad2"

model = AutoModelForQuestionAnswering.from\_pretrained(model\_name)

tokenizer = AutoTokenizer.from\_pretrained(model\_name)

Figure 2.3 shows the question-answer model trained on Squad

For the question-answer model, the pretrained model is RoBERTA instead of BERT. RoBERTA is trained on more data, larger batch size, higher learning rate, and without the next sentence prediction in BERT, and ultimately performs better than BERT. [6] The question-answer model takes in the embedding of the context and embedding of the question and outputs the softmax probability of the starting point of the answer across all the words in the context and the span of the answer. If the context is greater than the maximum length of the sentence to encode an embedding, the context is split into chunks. [4]

The third model is a generative model to generate response from the FAQs. The question is the utterance while the answer is the response. The model is trained using the FAQs to generate a sequence of text in response to the questions. A dictionary is used to store the words and the unique mapping between a word and a number. The model uses an embedding layer of the size of the dictionary. 2, 4, and 8 GRU layers were explored to encode the semantic meaning of the sentence. Model was also trained to explore sentences with the number and frequency of words occurring within some threshold. Attention layers were used to decode the features and dot product is used to predict the next word in the sentence. A greedy search decoder is used to determine the most likelihood of the next word. As the model outputs are numbers, the dictionary mapping is used to convert the number to words. [7] The generative model is hard to tune and requires massive amounts of data with specialized nlp techniques to produce a grammatically correct and sound sentence. Moreover, the model converges slowly when the sentences are long and contain low frequency words.

**Training Datasets**

This project explores two datasets to train or fine tune the models in this project. The first dataset is the Covid FAQs scrapped from the government API. Figure 2.4 shows a sample FAQ from the API. The full FAQs can be obtained by changing the params size to 178, the current number of FAQs on the website. The responses to the FAQs are meant to render on the website; hence there are html tags embedded within the response. [8]

<https://ask.gov.sg/api/v1/posts?sort=top&agencyId=5&topics=&page=1&size=10>

"title":"I have tested positive on my Antigen Rapid Test (ART) self-test. Do I need to report my results to MOH?",

Figure 2.5 shows a sample FAQ from the API

However, this dataset does not contain alternative phrasings to the original FAQ. The alternative phrasings can be manually generated; instead, online text augmentation was performed to introduce variations to the original sentence before feeding the model. Since the embedding encodes the semantic meaning of the sentence, this augmentation aims to help the model learn from minimal data while still capturing utterances’ intent. Augmentation includes dropping words and shuffling the sentence order.

The second dataset is a question answer dataset. It contains 2019 data points and is obtained from 147 scientific articles. The dataset is annotated by 15 volunteer biomedical experts. This dataset aims to build QA systems to help scientific researchers, public health authorities etc to find answers to common questions in these papers. [9] Figure 2.6 shows how the question answer dataset is loaded and split to train and valid dataset. As the dataset is small, the valid dataset is restricted to a small size. The dataset contains a lot of medical terms that will be useful to train the model to answer scientific or factual questions that are difficult to generate multiple intents for them. This is necessary since the public does ask all sorts of factual and scientific questions.

train\_dataset = load\_dataset("covid\_qa\_deepset", split='train[0:2000]')

valid\_dataset = load\_dataset("covid\_qa\_deepset", split='train[2000:]')

Figure 2.6 shows how to load the qa dataset

**Fine Tuning Models**

For the text classifier, the parameters of the model were modified but the model tends to overfit easily given that the dataset is small. The best method to train the model with a small dataset is to perform online NLP augmentation such as dropping words and shuffling the sentence order. Figure 2.7 shows that the augmentation helps the model to classify the intent correctly for this utterance.

Me: i am living with someone with covid

# Without Augmentation

Bot: What happens if I do not have a room to self-isolate, or live ...

# With Augmentation

Bot: I live with somebody who has tested positive for COVID-19, what should ...

Figure 2.7 shows the NLP augmentation impact on the text classifier

For the question answer model, the parameters of the model, such as the truncation method and flag to return overflowing tokens. The model used was not pretrained on the Squad dataset. However, the dataset is small, and the model is not able to precisely locate the answer to the question. Figure 2.8 shows an example of the answer that the bot is able to locate, but provided more information than is needed. When the pretrained model on Squad is used and fine tuned on the Covid dataset for domain transfer, the model can precisely locate the answer.

Me: Where was covid first identified?

# Without Squad

Bot: from an outbreak in the Chinese city of Wuhan in December 2019

# With Squad

Bot: Wuhan

Figure 2.8 shows the performance improvement using pretrained model on Squad

For the generative chatbot, the parameters of the model, such as the number of layers, maximum length of the sentence, and the minimum frequency of words are experimented. However, the model replies only from what it learns from the dataset. Besides giving gibberish responses, the model has the tendency to produce grammatically incorrect sentences and unnecessary long sentences if the maximum length is not set properly. Figure 2.9 shows an example of the poor response of the generative model.

Me: i am lonely

Bot: you re not going to be a hero ? you ? me . me . me

Figure 2.9 shows the response from the generative model for the given utterance

Refer to Covid\*Chatbot.ipynb files on how the models are trained and chatbot\_\*\_model.py files on how to test the models. Refer to the README.md for instructions on how to set up the environment and place the models in the correct path directory. The models can be downloaded from the Google Drive link below.

https://drive.google.com/drive/folders/1cBHe68B-fJ\_H\_tvBthbfJ6udvRbPofmi?usp=sharing

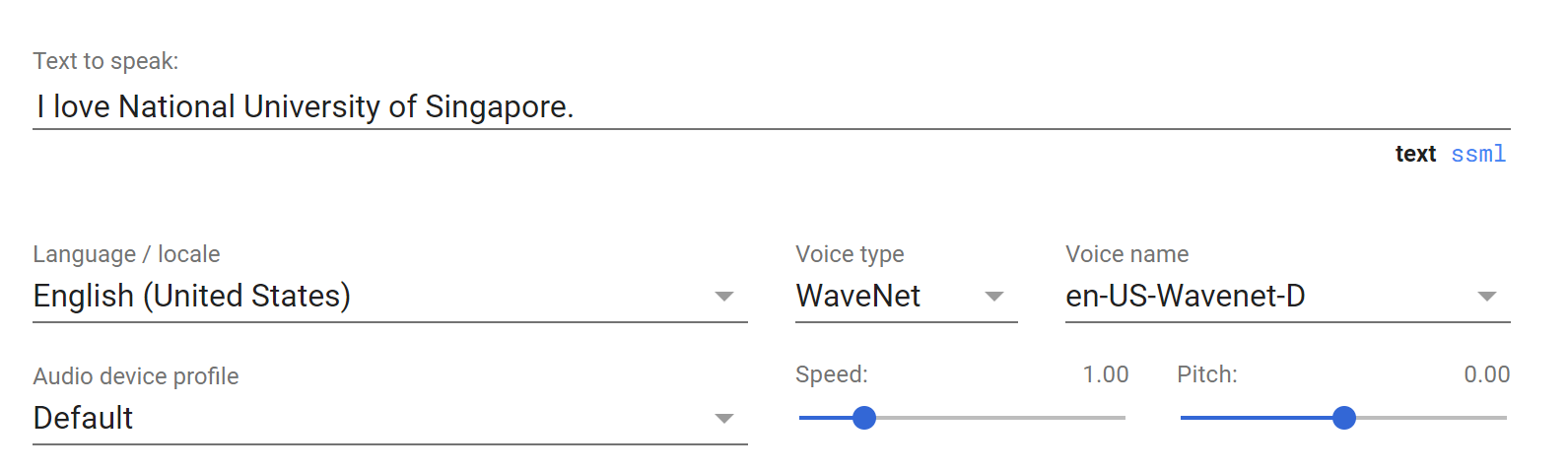
**3. Look for some APIs available to convert text to speech in Python. Try adjusting the arguments in APIs to obtain better samples.**

**API 1: Google Cloud Text-to-Speech API**

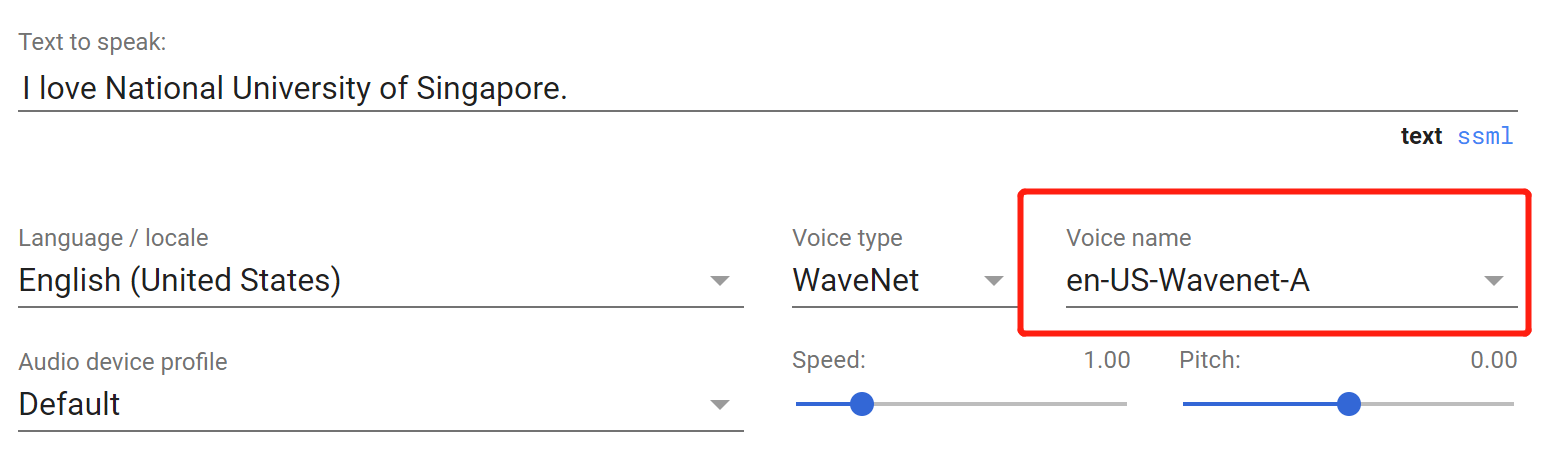
This API uses the pretrained model of WaveNet [3] to translate the text data to speech data. This paper introduces WaveNet, a deep neural network for generating raw audio waveforms. The model is fully probabilistic and autoregressive, with the predictive distribution for each audio sample conditioned on all previous ones; nonetheless it shows that it can be efficiently trained on data with tens of thousands of samples per second of audio. When applied to text-to-speech, it yields promising performance, with human listeners rating it as significantly more natural sounding than the best parametric and concatenative systems for both English and Chinese. A single WaveNet can capture the characteristics of many different speakers with equal fidelity, and can switch between them by conditioning on the speaker identity.

Example1: I love National University of Singapore.

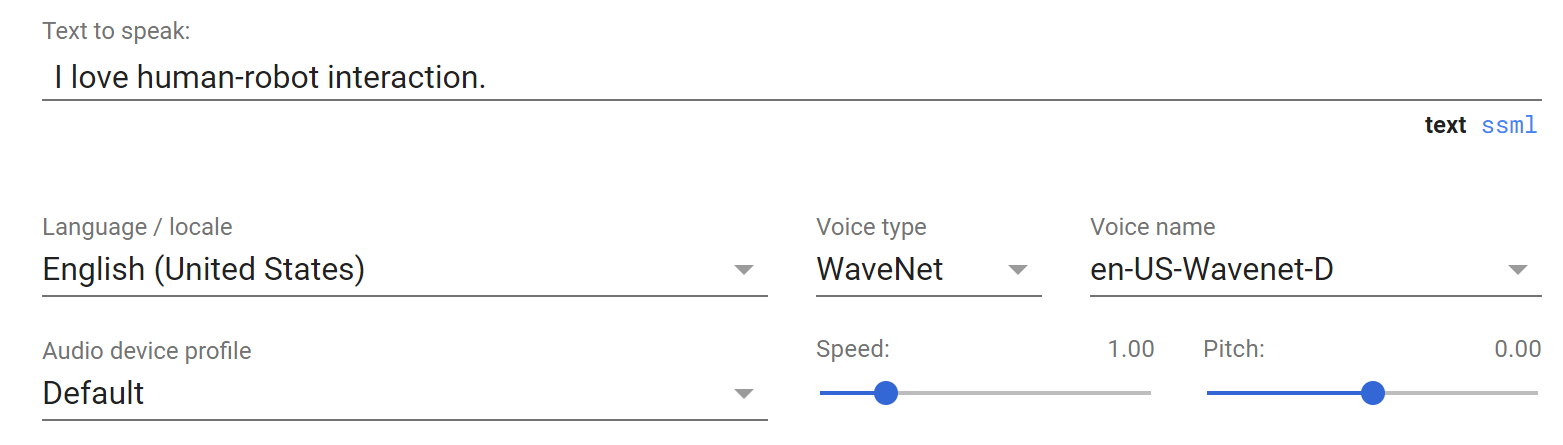
I set the default setting of the translation.



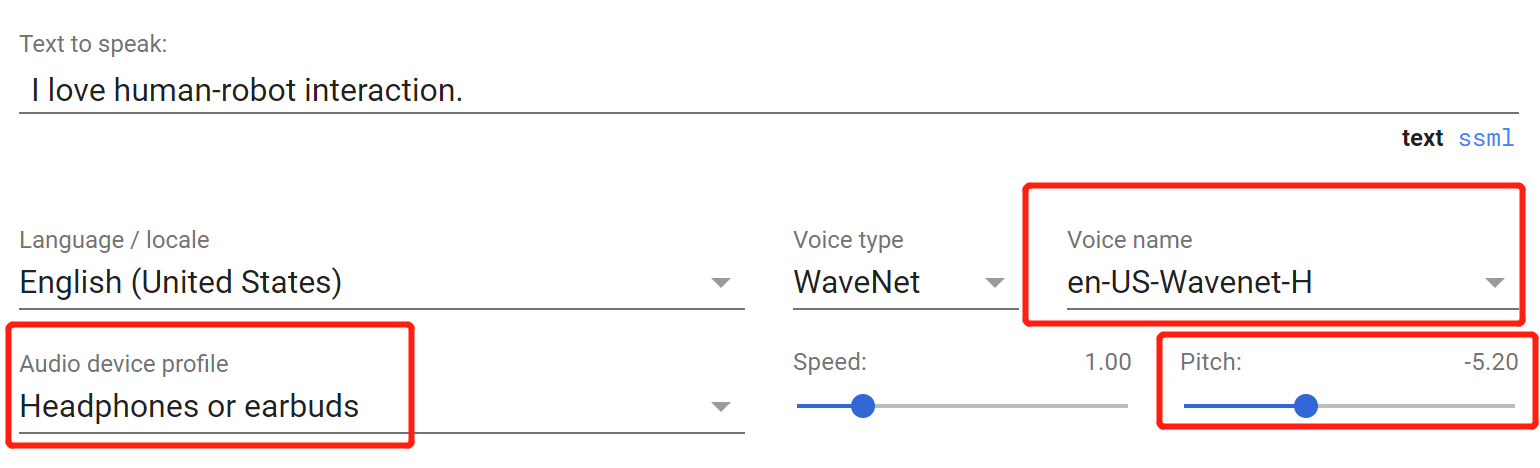
To test the function of conditioned on different voice types, I changed the voice name of the speech translation.



Example 2: I love human-robot interaction.



I changed the audio device profile, voice name and pitch of the translation.

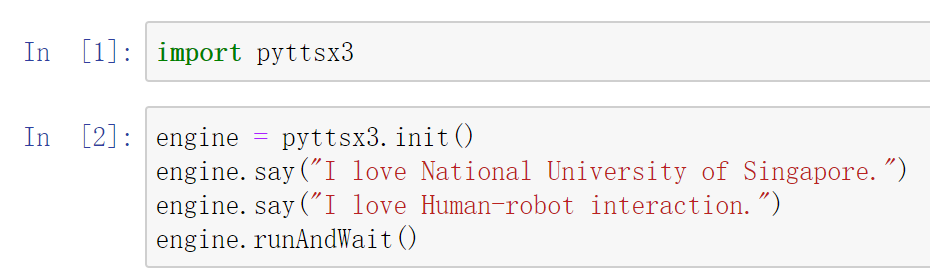


**API2: Pyttsx3**

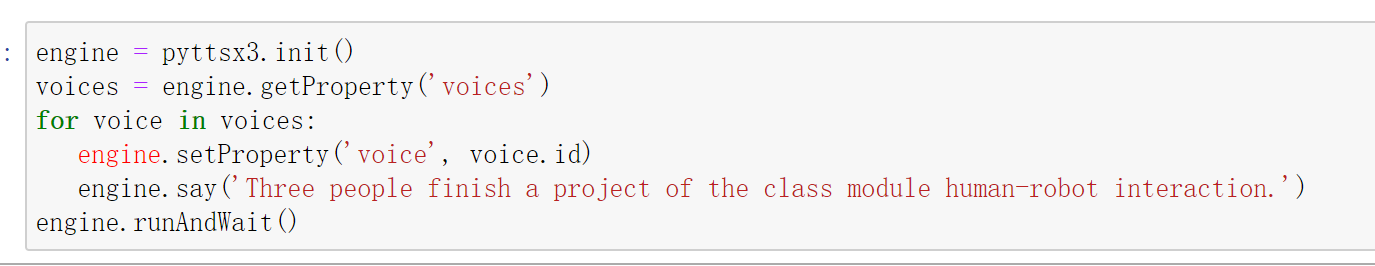
pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to pyttsx3. Engine instance. It is a very easy to use tool which converts the entered text into speech.

Example 1:

This is a simple and fast try of the pyttsx3 module.

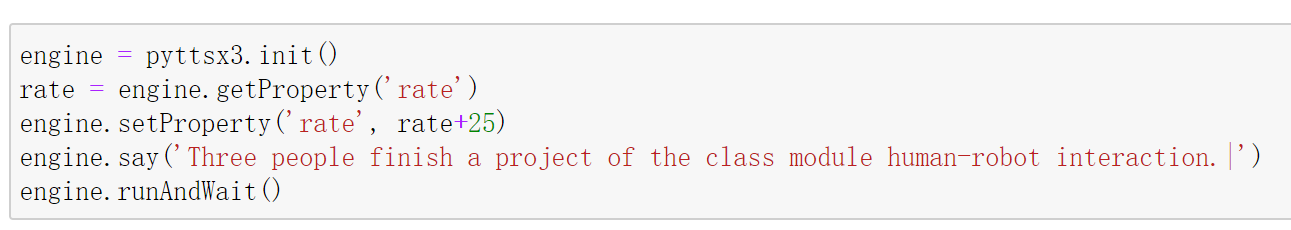


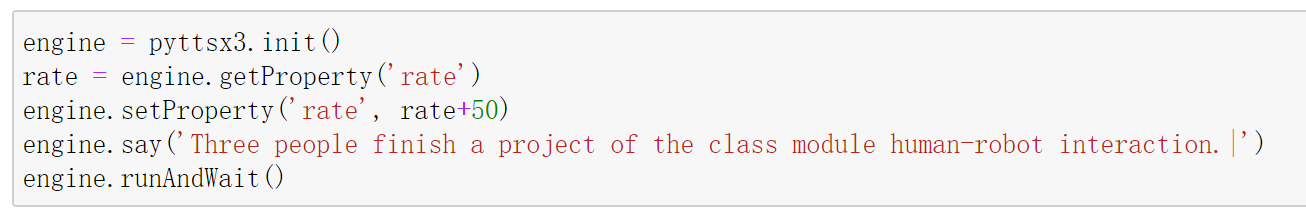
Example 2:

This example is to change the voice type of the translation with the pretrained model. 

Example 3:

Changing the speech rate to 25 and 50.





Afterall, this is a simple tool to use to help us feel the interest of translating the text-to-speech.

**4. Develop the dialogue systems with speech output according to your aspiration/interest, such as Receptionist for Shops, Restaurant, Clinics.**

The Covid-19 dialogue system uses the text classification model and question-answer model. The dialogue system is in the context of government chatbots such as AskJamie. [10] Figure 4.1 shows the flowchart of the dialogue system.

The goal of the text classification model is to respond to the utterance if it is similar to the FAQs. The chatbot is trained to recognize the intents from the utterance and will provide an official response to the corresponding intent.

The goal of the question-answer model is to answer factual questions that can be found in newspapers and reports without having to create new intents and classes to retrain the text classification models. The question-answer model addresses questions such as the number of cases in Singapore and where Covid was first identified.

The generative model is not used because the response of the chatbot is often gibberish. In the context of a pandemic, the generative model has a higher chance of giving false information or gibberish that will confuse the public.

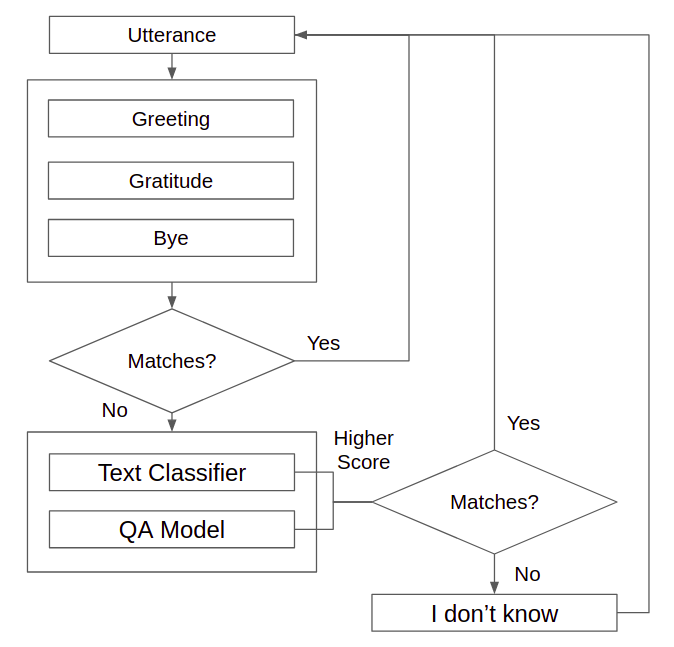


Figure 4.1 shows the flowchart of the dialogue system

When the dialogue system receives the utterance, the chatbot first checks if the utterance intends to greet, show gratitude, or bid farewell. This part of the dialogue system is separated from the text classifier to avoid having to retrain the model and separate the functions of the dialogue system into distinct modules.

For the first module that checks if the utterance intends to greet, show gratitude or bid farewell, levenshtein distance is used to compare the similarity. Figure 4.2 shows an example of the text similarity. Levenshtein distance measures the number of edits to transform one word to another. If the utterance has more than the word “hi”, the utterance might have an intent. This leads to a low levenshtein score and triggers the text classifier and the qa model to respond instead.

>>> 1 - distance.nlevenshtein("hello.", "hello", method=2)

0.8333333333333334

Figure 4.2 shows the levenshtein similarity between the two text

If the text classifier and qa model is triggered to respond to the utterance, both models provide their response and confidence of their response. The response with the higher confidence will be returned to the user. However, if the confidences are below a threshold, both models are uncertain if their response is correct, the dialogue system will respond that it does not know the answer.

Figure 4.3 shows the conversation with the dialogue system. Part of the response is redacted since it is long. The bot starts by introducing itself. The next 3 questions are meant to trigger the text classifier response and the next 3 questions are meant to trigger the qa model. Even though the questions are phrased differently from the FAQs or the context, the dialogue system responds correctly.

Bot: Hi, I am Ask Jamie. I can answer queries about the Covid situation in Singapore. How can I help you

Me: i am living with someone with covid what to do

Bot: If a household member has tested positive for COVID-19, ...

Me: where can i collect art kits to test myself for covid

Bot: For Residents: You can collect the ART kits from vending machines ...

Me: if i recover from covid when can i exit isolation

Bot: If you are well and test positive, or your condition has been assessed by a doctor to be mild, ...

Me: Where was covid first identified?

Bot: Wuhan

Me: when did the world health organization declare the pandemic

Bot: 11 March 2020

Me: how many cases and deaths

Bot: more than 423 million cases and 5.88 million

Me: thanks

Bot: No problem!

Me: bye

Bot: Bye!

Figure 4.3 shows the conversation with the dialogue system

**5. Analyse the performance of your dialogue systems. Show all your results and findings in the report and discuss and comment on them.**

Figure 5.1 shows the distribution of the length of the sentences for the utterance and response. The length of the utterance is below 40 and is smaller than the BERT maximum sentence length of convert to embeddings. For the text classification model, the transformer “bert-based-cased” is used. There are very few preprocessing steps needed, since the FAQs do not contain html or url etc.

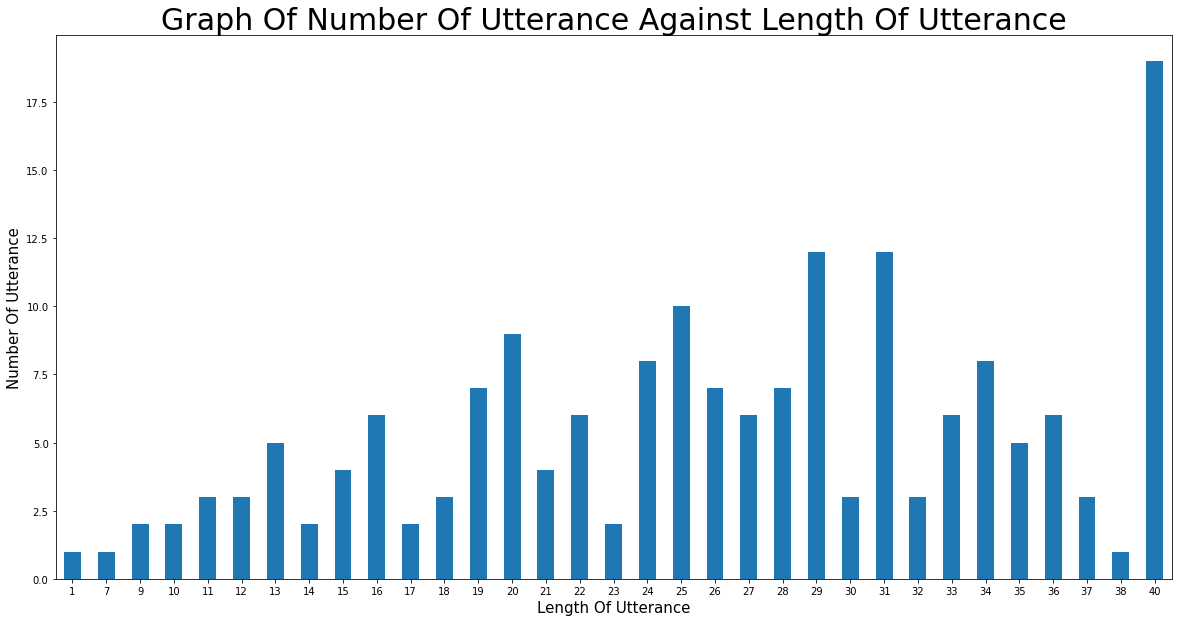
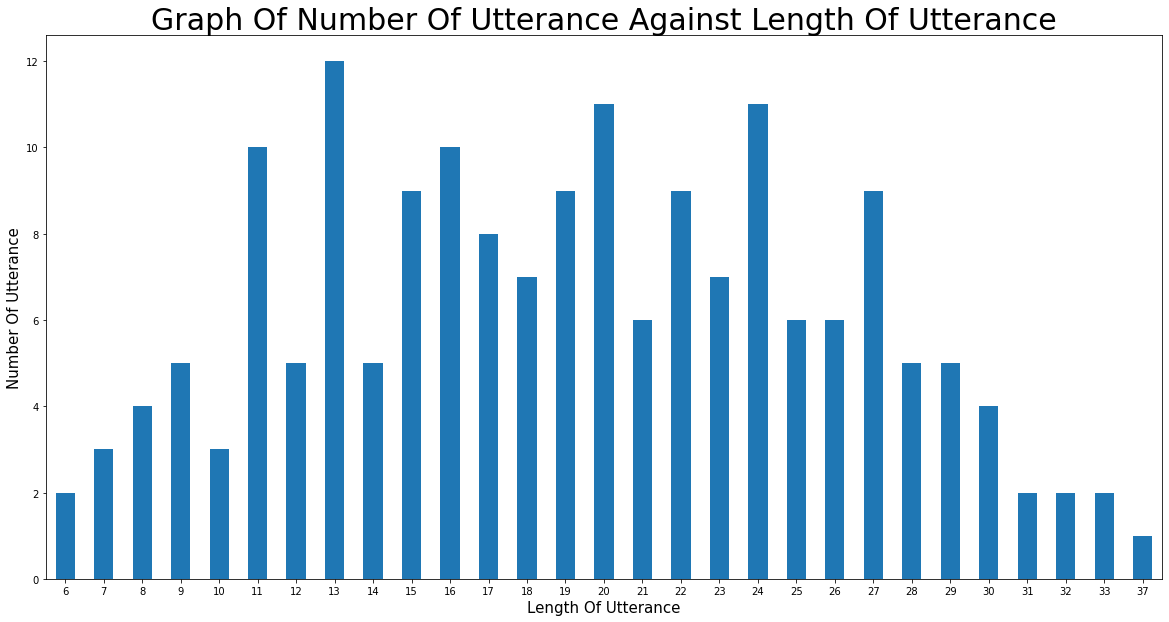


Figure 5.1 shows the number of utterances (left) and response (right) of certain length

Figure 5.2 shows the train and validation plots per epoch. With and without augmentation, the accuracy plots look similar. The model can generalize when the text is augmented. Moreover, the model can perform decently when the utterance is phrased differently. The model can be trained under 100 epochs, taking about 30 mins on Google Collab.

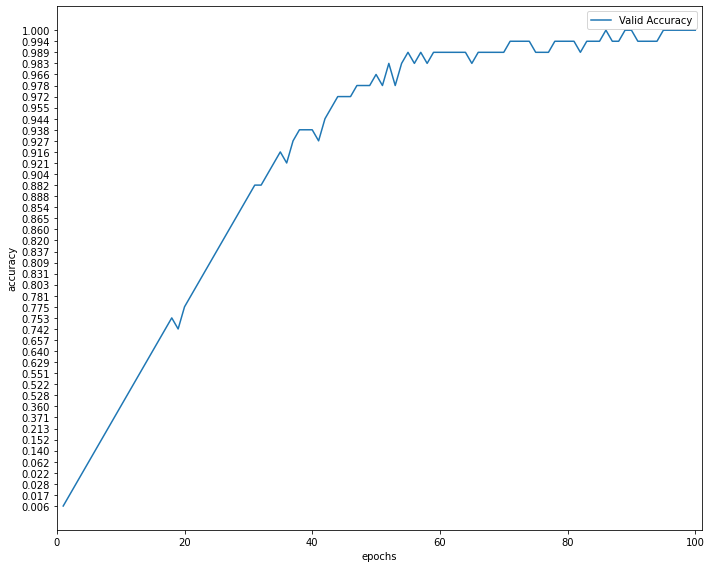
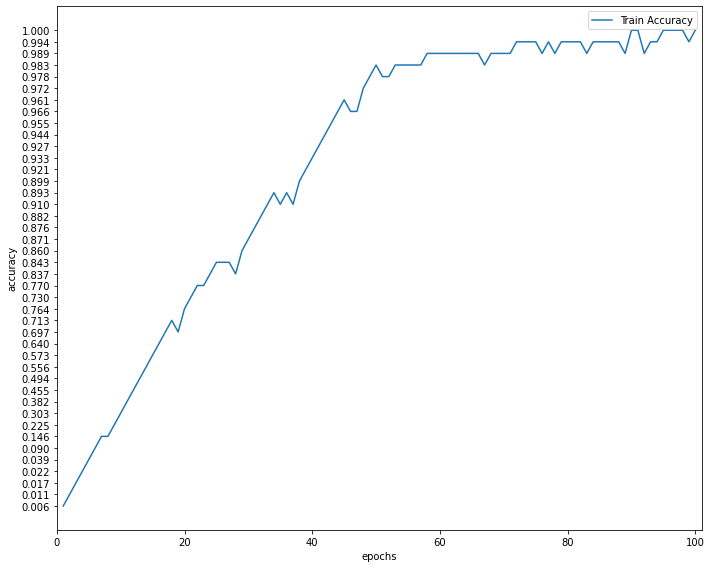


Figure 5.2 shows the train (left) and validation (right) accuracy with augmentation

From the example given in Figure 2.7, the model performs better when there is online text augmentation. The augmentation forces the model to generalize and learn from the context of the words. This addresses pain point 1 and 2 of having to frequently update the dataset and generative alternative phrasing given how fast policies of Covid-19 changes. This saves time and effort to build a decent chatbot.

Figure 5.3 shows the training and validation loss when the QA model is trained on the Covid-19 dataset. The first two rows use the model “deepset/roberta-base-squad2” which was pretrained on the version 2 of the Squad dataset. [11] The last two rows use the “distilbert-based-cased” which was not pretrained on version 2 of the Squad dataset. [12] The train and validation losses for the non-pretrained model on Squad model is higher. Maybe the number of iterations is not sufficient. The performance difference can be seen earlier in Figure 2.8 where the pretrained on Squad model precisely locates the answer while fine tuning on 10 epochs only. The transfer learning is very powerful to improve a new QA model for a new domain given minimal data in a short period of time.

| Epoch | 5 | 6 | 7 | 8 | 9 | 10 |
| --- | --- | --- | --- | --- | --- | --- |
| Train Loss  (Squad) | 0.277100 | 0.277100 | 0.277100 | 0.064000 | 0.064000 | 0.0640000 |
| Val Loss  (Squad) | 0.010870 | 0.088372 | 0.026505 | 0.003444 | 0.049011 | 0.007765 |
| Train Loss  (No Squad) | 0.818100 | 0.818100 | 0.818100 | 0.420300 | 0.420300 | 0.420300 |
| Val Loss  (No Squad) | 0.030290 | 0.043201 | 0.044506 | 0.088298 | 0.054231 | 0.040333 |

Figure 5.3 shows the train and valid loss when model is pretrained on Squad dataset

For the generative model, Figure 5.4 shows the conversation with the model for greeting. For this example, the model is trained on Cornell Movie Dialogue dataset. [13] Even though the model seems to respond with some sense, the response is highly dependent on what conversations were present in the dataset. Moreover, the response shows that the punctuation is bad and needs advanced NLP techniques to fix.

Me: hi

Bot: hi . a minute . . .

Me: what you doing

Bot: i m going to go . dance .

Figure 5.4 shows the conversation with the generative model for greeting

Figure 5.5 shows the conversation with the generative model for mental illness, in particular lack of companionship. For this example, the model is trained on Cornell Movie Dialogue dataset. [13] Initially, the project idea was to build a dialogue system for addressing the lack of companionship in an aging society. However, training a generative model proves to be difficult. When the model is trained on conversation with sentences of maximum length 10 and 20, the response of the chatbot is entirely different. For the latter model, it is observed that the model did not stop after the question mark but repeated “me.” several times.

# Max Length = 10

Me: i am lonely

Bot: you re not a gambling man .

# Max Length = 20

Me: i am lonely

Bot: you re not going to be a hero ? you ? me . me . me

Figure 5.5 shows the conversation with the generative model for mental illness

Figure 5.6 shows the conversation with the generative model trained on the Covid FAQ scrapped from the government website. The model can respond sensibly to some extent. However, the sentence is badly phrased with lots of spelling errors. The errors are most likely a result of the preprocessing steps. Nevertheless, generative models are hard to train to produce responses that are mostly free from grammar and spelling errors given the limited time of this project.

# Query 1

query: i live with someone with covid

response: if you are to be quarant at a designated facility the quarant order qo agent will be able to self isolate in your room preferably with an attached bathroom covid . as a close contact of a confirmed

# Query 2

query: Can I go out to buy food, see a doctor, or go to work or school?

response: you should either order in food del ies and or seek the help of friends relat as you are allowed to leave home to collect the art test kits from the vending mach . however you should minim social contact

Figure 5.6 shows the conversation with the generative model trained on the Covid FAQ dataset

Figure 4.1 shows how the final version of the chatbot is constructed using the text classifier and qa model, since the generative model is difficult to produce good response. In this Covid situation, the government would prioritize having a dialogue system that replies utterance with correct response. Even though it is possible to solely have a text classifier, the model accuracy decreases as more classes are added to the dataset. The qa model can be relied on to answer factual and scientific questions instead to improve the text classifier accuracy. One issue with the current model is that the utterances are limited to English only.

In summary, the team built a dialogue system to address public queries on Covid-19 situation in Singapore. The main part of the dialogue system consists of the text classifier and the qa model. The text classifier is trained on the FAQ scrapped from the government website. The text is augmented during training. The qa model, previously trained on Squad dataset, is trained on a covid qa dataset for domain transfer. The greetings, bidding farewell, and showing gratitude part of the dialogue system is handled by computing the levenshtein similarity. A functioning dialogue system is demonstrated, showing the pain points can be innovatively addressed using these techniques.

**Appendix**

[1] S. A. Sheikh, V. Tiwari and S. Singhal, "Generative model chatbot for Human Resource using Deep Learning," 2019 International Conference on Data Science and Engineering (ICDSE), 2019, pp. 126-132, doi: 10.1109/ICDSE47409.2019.8971795.

[2] S. Wu, Y. Li, X. Zhang and Z. Wu, "DLCEncDec : A Fully Character-Level Encoder-Decoder Model for Neural Responding Conversation," *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, 2018, pp. 516-521, doi: 10.1109/COMPSAC.2018.00079.

[3] Van Den Oord, Aäron, et al. "WaveNet: A generative model for raw audio." *SSW* 125 (2016): 2.

[4] Jacob Devlin and Ming-Wei Chang and Kenton Lee and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019. arXiv. 1810.04805.

[5] Rajpurkar, Pranav and Zhang, Jian and Lopyrev, Konstantin and Liang, Percy. SQuAD: 100,000+

[6] Yinhan Liu and Myle Ott and Naman Goyal and Jingfei Du and Mandar Joshi and Danqi Chen and Omer Levy and Mike Lewis and Luke Zettlemoyer and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019. arXiv. 1907.11692

[7] Matthew Inkawhich. Chatbot Tutorial - PyTorch. Published on. May 26, 2021. Accessed on. Feb 24, 2022. <https://pytorch.org/tutorials/beginner/chatbot_tutorial.html>

[8] Singapore Government Agency Website. Answers from Ministry of Health. Accessed on. Feb 24, 2022. <https://ask.gov.sg/agency/moh>

[9] Moller, Timo and Reina, Anthony and Jayakumar, Raghavan and Pietsch, Malte. COVID-QA: A Question Answering Dataset for COVID-19. Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020. 2020.

[10] GovTech. 'Ask Jamie' Virtual Assistant. Accessed on. Feb 24, 2022.

<https://www.tech.gov.sg/products-and-services/ask-jamie/>

[11] Branden Chan, Timo Möller, Malte Pietsch. Deepset. Roberta-Base For QA. Accessed on. Feb 24, 2022. <https://huggingface.co/deepset/roberta-base-squad2>

[12] Victor Sanh and Lysandre Debut and Julien Chaumond and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. 2019. ArXiv. abs/1910.01108

[13] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. 2011. Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011