Voice Translation Chatbot

Name: Yuechen Jiang
CWID: 10453204

1. Introduction and Objective

In order to help people make conversation in their native language. And also can let people with visual impairment communicate with others. We built A Chatbot allows people to communicate using different languages via voice message. Including 62 different language libraries. To help people easily to communicate with others in different languages.

2. Methodology

- Convert one user1's the voice message to the text
- Confirm the result is correct or repeat the previous (Optional)
- Chatbot receives the text message from User 1
- Translate User1's text message to English
- Translate User1's English message to the Target language, which is the language User2 is using.
- Chatbot sends the text message to User2
- Convert the text message from the chatbot to voice
- Convert one user2's the voice message to the text
- Confirm the result is correct or repeat the previous (Optional)
- Chatbot receives the text message from User 2
- Translate User1's text message to English
- Translate User1's English message to the Target language, which is the language User1 is using.
- Chatbot sends the text message to User1
- Repeat the process

3. Data Source and Pre-view

This dataset contains English to 80 different languages.

And the data looks like English + TAB + The Other Language + TAB + Attribution.

This work isn't easy.　この仕事は簡単じゃない。　CC-BY 2.0 (France) Attribution: tatoeba.org #3737550 (CK) & #7977622 (Ninja)

Those are sunflowers.　それはひまわりです。 CC-BY 2.0 (France) Attribution: tatoeba.org #441940 (CK) & #205407 (arnab)

Tom bought a new car.　トムは新車を買った。 CC-BY 2.0 (France) Attribution: tatoeba.org #1026984 (CK) & #2733633 (tommy_san)

This watch is broken.　この時計は壊れている。　CC-BY 2.0 (France) Attribution: tatoeba.org #58929 (CK) & #221604 (bunbuku)

The attribution gets imported into Anki as a tag, by default This attribution contains the domain name of the source material, the sentences' ID numbers, and the sentence owners' usernames. You can basically ignore the attribution field if you are using this material for personal use and not redistributing these files. However, it's needed here to comply with the CC-BY license.

Let's take Spanish as an example, the pre-view shown in the figure below.

	Go.	Ve.	CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #4986655 (cueyayotl)
0	Go.	Vete.	CC-BY 2.0 (France) Attribution: tatoeba.org #2...
1	Go.	Vaya.	CC-BY 2.0 (France) Attribution: tatoeba.org #2...
2	Go.	Váyase.	CC-BY 2.0 (France) Attribution: tatoeba.org #2...
3	Hi.	Hola.	CC-BY 2.0 (France) Attribution: tatoeba.org #5...
4	Run!	¡Corre!	CC-BY 2.0 (France) Attribution: tatoeba.org #9...

The description of the data is shown in the figure below.

	Go.	Ve.	CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #4986655 (cueyayotl)
count	138436	138436	138436
unique	117884	130468	138436
top	You can put it there.	¡Órale!	CC-BY 2.0 (France) Attribution: tatoeba.org #2...
freq	68	10	1

There are 138436 data contained in the Spanish dataset. We can split it into 3 parts (train, validation, and test). And the first stage of the model output is shown in the figure below. The parameters need to tune in the further work.

4. Evaluate the Translate Model with BLEU Score Using the Test Set.

BLEU, or the Bilingual Evaluation Understudy, is a score for comparing a candidate's translation of the text to one or more reference translations. Although developed for translation, it can be used to evaluate text generated for a suite of natural language processing tasks. We will discover the BLEU score for evaluating and scoring candidate text.

Scores are calculated for individual translated segments—generally sentences—by comparing them with a set of good quality reference translations. Those scores are then averaged over the whole corpus to reach an estimate of the translation's overall quality. Intelligibility or grammatical correctness are not taken into account.

BLEU's output is always a number between 0 and 1. This value indicates how similar the candidate text is to the reference texts, with values closer to 1 representing more similar texts. Few human translations will attain a score of 1, since this would indicate that the candidate is identical to one of the reference translations. For this reason, it is not necessary to attain a score of 1. Because there are more opportunities to match, adding additional reference translations will increase the BLEU score. A

A reasonable BLEU score should be 0.1 ~ 0.5.
In this model, we get a BLEU score of 0.14822081469853618. Which is in the target range.

5. Conclusion

This project can help people easily to communicate with others in different languages. Although, our BLEU score got an acceptable result. The data source we take is different from the original paper, and due to the memory limitation of Google Colab, we cannot load all the data for model training. Due to time constraints, the Seq2Seq model of this project is only trained in English and Spanish. In future work, more data and more languages need to be trained on GPUs with greater computing power, and the model network structure should be further adjusted and tuned parameters to obtain better model results and more accurate, more understandable translation. And in further work attention should be added to the model.

6. Problems and Short-come

The mutual conversion between speech and text is not accurate enough, and it has high requirements on the sound collection environment, and cannot be converted accurately in the

case of noise in the environment. Voice messages must be clearly pronounced, and the conversion rate of non-native speakers is significantly lower than that of native speakers. There is a higher failure rate in the conversion of slang and colloquial expressions. There are obvious defects in the translation model, and the model structure and parameters should be continued to be optimized. And add attention mechanism needs to be added to improve the accuracy of the translation.

6.1 Seq2Seq Main Idea

Seq2Seq models "encode and decode" sequence signals to generate new sequence signals, often used for tasks such as machine translation, speech recognition, and automated dialogue.

Seq2Seq is a network of Encoder-Decoder structure. Its input is a sequence, and its output is also a sequence. The Encoder converts a variable-length signal sequence into a fixed-length vector representation, and the Decoder converts this fixed-length vector into a variable-length target variable Signal sequence.

The Seq2Seq model is the model used when the output length is uncertain. This situation generally occurs in the task of machine translation. When translating a Spanish sentence to English, the length of the bold English sentence may be shorter than the Spanish sentence. May also be longer than Spanish, so the length of the output is indeterminate.

6.2 Optimization Objectives for Seq2Seq Models

In most supervised learning models, we will consider how to update the model parameters according to the loss function (or objective function), which is also the goal of model training, and the Seq2Seq model is no exception.

The goal of the seq2seq model is to maximize the probability of the target output sequence based on the information of the input sequence, similar to the idea of language models. For all training samples, there is a loss function of the form:

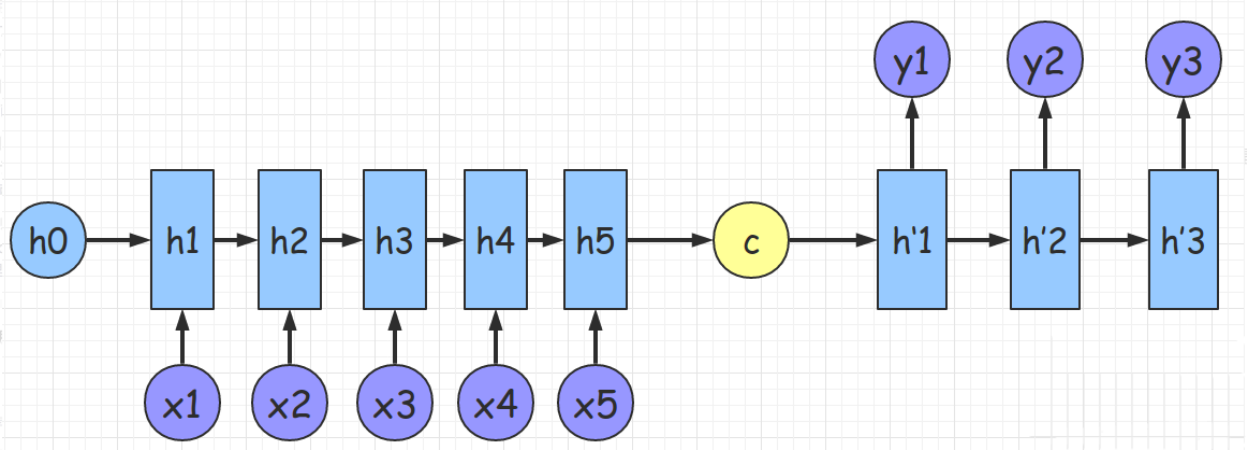
$$loss = -\frac{1}{N}\sum_{n=1}^{N} logp(Y_n|X_n,\theta)$$

where **N** is the number of training samples, X_n, Y_n is the input and output sequence corresponding to each sample, and θ is the parameter vector to be learned. Each $p(Y_n|x_n,\theta)$is generated by the Encoder-Decoder framework, which contains a large number of parameters in the neural network, which can be gradually optimized by gradient descent.

6.3 Disadvantages of the Seq2Seq model

6.3.1 The output vector **C** of the Encoder will lose information during the encoding process

It can be seen from the figure below that the only connection between Encoder and Decoder is semantic encoding C, that is, encoding the information of the entire input sequence into a fixed-size state vector and then decoding it, which is equivalent to "lossy compression" of the information.



There are obviously two disadvantages to doing this:

- Intermediate semantic vectors cannot fully express the information of the entire input sequence.
- As the length of the input information increases, because the length of the vector is fixed, the previously encoded information will be covered by the later information, and a lot of information will be lost.

The Encoder understands the input sentence and converts it into a context vector, but if the input sentence is too long, the resulting context vector will lose information. No matter how it is operated, it essentially encodes the source language into a fixed-dimensional vector **C**. This vector has limited representation capabilities, and for long sentences, some information will definitely be lost. Even if LSTM is used and the input is reversed, the dependency length between the first word input into the Encoder and the last word output from the Decoder is too long, and the dependency is weak.

Solution: With the help of the alignment idea, when the Decoder translates the **t** word, the source language word related to the **t** target language word is found from the Encoder for the translation of the **t** word.

Solution: Attention mechanism

6.3.2 The decoder cannot align the encoder

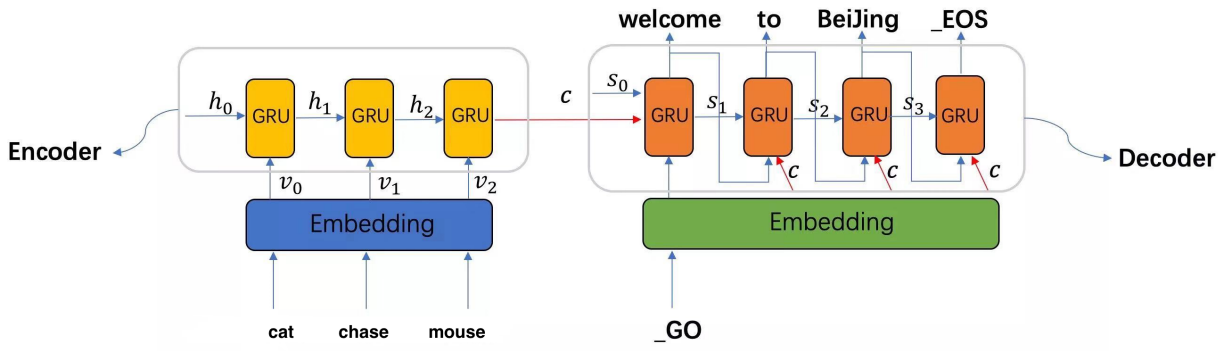
From the figure above, it can be clearly found that the contribution of the semantic code **C** to them is the same when generating $\mathbf{y}_1$, $\mathbf{y}_2$, and $\mathbf{y}_3$.

For example translation: **Cat chase mouse** , the model is generated verbatim: **'gato', 'persigue', 'al', 'ratón'.** When translating the word mouse, every English word contributes the same to "mouse". If the Attention model is introduced, then the influence of mouse on it should be the greatest.

6.3.3 The hidden vector $\mathbf{s}_i$ of the Decoder will gradually lose the information of the input vector **C**

During the decoding process by the Decoder, as the time step progresses, the hidden vector $\mathbf{s}_i$ of the Decoder will gradually lose the information of the original input vector $\mathbf{s}_0$ **(**$\mathbf{s}_0$ == **C** this semantic vector **C i**s the final semantic vector output by the encoder);

Solution: When Decoder decodes at each time step, the original input vector **C** is added again as an input parameter. As shown below:



- $\mathbf{h}_i$ represents the output Embedding Vector of each input data in the encoder module after passing through the RNN.
- $\mathbf{s}_i$ represents the output Embedding Vector of each output data in the decoder module after passing through the RNN. As the time step increases, the information of **c** contained in $\mathbf{s}_i$ will become less and less.
- Each time step in the decoder module uses the **c** vector, which avoids the loss of information caused by less and less information of **c** contained in $\mathbf{s}_i$**.**

$$Y_t = g(Y_{t-1}, s_t, C)$$

- Y_{t-1} represents the output of the Decoder at the previous time step $\mathbf{t-1}$ as one of the inputs of the Decoder at the current time step **t.**

- $\mathbf{s_t}$ is the hidden layer of the Decoder at time step **t.**
- **C** represents the semantic vector finally output by the previous Encoder.
- **g** can be a nonlinear multi-layer neural network to generate the probability that each word in the dictionary belongs to $\mathbf{Y_t}$**.**