Journal of Science and Technology
ISSN: 2456-5660 Volume 5, Issue 3, May-June 2020

www.jst.org.in *DOI: https://doi.org/10.46243/jst.2020.v5.i3.pp192-194*

Language to language Translation using GRU method

Shwetha A[1], Ravikala[2], Vijetha[3]

[1](Computer Science, Srinivas Institute of Technology, India)
[2](Computer Science, Srinivas Institute of Technology, India)
[3](Computer Science, Srinivas Institute of Technology, India)
[1]Corresponding Author: shwetha20598@gmail.com

To Cite this Article
Shwetha A, Ravikala and Vijetha, "Language to language Translation using GRU method", Journal of Science and Technology, Vol. 05, Issue 03, May-June 2020, pp192-194

Article Info
Received: 14-02-2020 *Revised: 08-05-2020* *Accepted: 15-05-2020* *Published: 25-05-2020*

Abstract: *Current state of the art translation systems for speech to speech rely heavily on a text representation for the interpretation. By transcoding speech to text we lose important information about the characteristics of the voice like the emotion, pitch and accent. The thesis examine the likelihood of using an GRU neural network model to translate speech to speech without the requirement of a text representation that's by translating using the raw audio data directly so as to persevere the characteristics of the voice that otherwise stray within the text transcoding a part of the interpretation process. As a part of the research we create an information set of phrases suitable for speech to speech translation tasks. The thesis leads to a signal of concept system which requires scaling the underlying deep neural network so as to figure better.*

Key Word*: GRU, speech translation, Skype translation, Tensorflow, Google translation.*

I. Introduction

This thesis takes on the matter of speech-to-speech translation with persevered voice characteristics found within the source voice throughout the interpretation. Supported recent advancements within the scientific computing area combined with the leading edge technology of deep neural networks we believe that a translation system based solely on the human voice is ready to require computational linguistics to the following level. Today state-of-the-art translation systems rely heavily on the text representation so as to translate. These systems are getting a divorce into three distinct steps. The primary step is that the speech-to-text part. The core component of this step is typically remarked because the speech recognition component. The subsequent step is that the actual translation step. The interpretation is completed within the text domain. The ultimate step is that the text-to-speech step. The core component of the ultimate step is that the speech synthesizer. With the advancements of neural networks and deep learning each of those components has independently become better the previous couple of years. Previous research shows that deep recurrent neural networks have a formidable effectiveness on previously difficult tasks where dependency over time between consecutive examples is very important.

II. Literature survey

Google translation system from Google mainly on text, but do have built-in functionality to require the input from a microphone then output the translated audio through the speakers. As previously described Google translate use the classical speech-to-speech translation style with the employment of a speech recognizer for speech to text, then translating the text and eventually a speech synthesizer system for generating the audio associated with the text. It's worth to say that the Google translate service could be a tough candidate to drill terms of correct translations thanks to its well engineered implementation with enormous quantities of computer file from many different sources. Skype translator The VoIP service from Skype [2] has currently an open beta where it's possible to create near real-time speech translation during calls with their service using deep learning [1]. The inspiration of their translation feature, called Skype Translator [3], is constructed on the identical setting as Google's translation service with one addition [2, 6, 7, 4, and 8]. Skype also use an acoustic model for mapping the source voice to a translated voice just like the source voice. At the purpose of inscribing this report the acoustic model more or less

use two translator voices, one male and one female, and take a look at to map the source voice to the closest acoustic model. This in theory will yield a more dynamic translated voice in terms of voice characteristics. However the present results rather bulky and looks like a robot with small fragments of human elements [5]. Google's translation service provide a more human like speech output although they are doing not use an acoustic model for his or her output.

III. Proposed System

In project to build these networks we got an open source library released by Google. The Tensorflow library contains functions to create and run neural networks as smooth and seamless as possible. Since it's well documented and have an honest set of examples to create from, including a sequence to sequence example, it became an honest choice for this project. Tensorflow is overall well built but do indeed still have some bugs thanks to its short lifetime. However the Tensorflow team is consistently updating and improving the library and its current state makes it a robust candidate for creating neural networks. One main advantage with Tensorflow is that the easy assigning the network computations to GPUs and as described within the background the employment of GPU processing is important for networks of this scale. The long GRU network we use in our system is constructed of two layers with 800 neurons in each layer at the most. The inner workings of the GRU network has been described within the background and therefore the Tensorflow library provide call Basic GRU Layer to make the layer. Since our data representation is larger than the neuron size of the network we'd like to define the input dimension when creating the primary GRU layer. When defining this input dimension Tensorflow build an input projection so as to travel from the information representation size to the GRU size. When testing the network we'd like to produce a loop function. This loop function is employed at output of every decoder step so as to predict the input to the subsequent time step. The loop function used for this network is that the same function because the output projection described below. When training the system skip the loop function and therefore the network instead insert the proper prediction as input for every next time step. This can be done to permit the network to stay learning throughout the full sequence even when the primary outputs are bad. In the model we map each phrase into its matching bucket. We used six different bucket sizes, 50, 100, 150, 250, 350, and 500. Tensorflow supply a call to form a bucket model, which we built our bucket model on. The difference between our model and therefore the supplied bucket model was that we wanted to stay the loss function break away the model and thus we had to implement it without the loss function. The bucket modeling function creates a model matching the longest bucket. Then for the smaller buckets it slices the encoder and decoder length to match the smaller bucket. By doing this we get a model that make use of the identical training between different bucket models for encoder and decoder steps that are shared between two buckets.

IV. Result and Discussion

When running our network performance experiments we started with the network size experiment where we ran 8 different network sizes for 20 hours then plotted the loss for every network. What we see here could be a clear decrease within the loss function betting on the scale of the network where a bigger network lead to a smaller training and evaluation error. The actual decrease within the loss function is very small over the scale increase of the network. However the decrease is consistent and happens between all size increases except size 400 and 600 where it actually did decrease, just not enough to be caught within the plot. What we are able to conclude from this experiment is that we might probably have the benefit of having the ability to extend the network size more so as to urge better results. The second network performance experiment was the one where we alternated the training size of the biggest possible network, the one with 2 layers and 800 nodes. It shows that when employing a large training set we are ready to generalize over the test set in a very rather similar way on the test set since the difference between the evaluation error and training error decrease because the training size increase. Further we see that once we use a really small set of coaching points we are ready to yield superb results on the training data, however we perform very bad on the test set for obvious reasons. This tells us that the network is really ready to translate previously seen phrases as long because the training set is little. Finally it shows that the rise in training error is very fast within the early increments of the training size, which again indicates that we'd like a bigger network so as to urge better results. Both networks used for the interpretation experiments were trained for 20 hours and every network trained a complete of roughly 90000 phrases during that point. The complex network trained using all 11000 phrases could only output noise after these 20 hours. Although the regularization parts of the network should indeed yield a bigger error from the network we thought it'd perform better than simply plain noise both for training data and evaluation data. The results for the smaller network trained on 40 phrases were

better. When allowing the network to correct the input for every time step the network was able to perfectly reconstruct the output phrase. When not assisting the input for every time step the network managed to stay the output with a transparent voice and distinct characteristics of the speech for about 10 out of 100 segments of the input sequence before turning into noise. These first segments resulted in a very similar output voice and also the words said were even as easily identified as within the actual output. One difference were that the background changed its pitch slightly although the characteristics of the background remained the identical. When evaluating the test data attack the tiny network we found some interesting elements. When only predicting only once step at a time the test set resulted in total noise. No human like voice element were present. However when allowing the network to predict all the time steps by itself it actually produced some human like voice output for about 15 out of 100 segments. We believe that the explanation for the only time step prediction to perform worse than the complete prediction is that the only time step keeps dragging the prediction far away from the learned sweet spot of the network which successively leads to noise. Compared to the complete prediction where it finds the closest sweet spot of the network and tries to predict from there but since the input sequence is way far away from previously known inputs it leads to human-like nonsense. The characteristics of this nonsense were tough to define, but it had been clearly far from the characteristics of the right output. It had been closer to the input phrases that the network had trained on in terms of speed and pitch. Another interesting feature was that the complete prediction output managed to spot some simple and distinct elements of the background, sort of a car driving the way, when no voices were present.

V. Conclusion

Looking at the results of the network performance experiments combined with hearing the interpretation output from the trained network we are able to draw some conclusions. First of all, there's little question that the GRU specification is complex enough to be told this sort of sequences. Supported the error rate of coaching phrases with a tiny low data set we are able to translate the scene phrases almost perfectly. The problem we discover here is that the network isn't large enough to handle a more general setting with much larger data set, that's the network is under fitting the information. We are able to also see that because the training set increases the test error moves closer to the training error. This means that there are similarities among the phrases which will be identified by the network if we are able to proportion. We managed to make a knowledge set that suited all of our needs which had a awfully good distribution among the lengths of the phrases. However the time step added to the beginning and end of every phrase should be determined by a scientific search so as to cut back the bad cuts. Trying to eliminate the non voice phrases by identifying non voice subtitles could reduce the quantity of bad phrases within the data set. The pipeline setup for creating this data set also allows us to proportion the information set fast if need be.

References

[1] Skype translator. https://www.skype.com/en/features/skype-translator/. [Ac-cessed: 2016-05-16].
[2] Skype voip software. https://www.skype.com/. [Accessed: 2016-05-16].
[3] Martín Abadi, Ashish Agarwal, and Paul Barham et al. Tensorflow*: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.
[4] Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. *A probabilistic approach to syntax-based reordering for statistical machine translation*. In Annual Meeting-association for Computational Linguistics, vol-ume 45, page 720, 2007.
[5] Microsoft. Skype translator presentation. https://www.youtube.com/watch? v=rek3jjbYRLo. [Accessed: 2016-04-30].
[6] Microsoft. How technology can bridge language gaps. http://research. microsoft.com/en-us/research/stories/speech-to-speech.aspx, 2015. [Accessed: 2016-05-14].
[7] Y. Qian, Y. Fan, W. Hu, and F. K. Soong. On the training aspects of deep neural network (dnn) for parametric tts synthesis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3829– 3833, May 2014.