Language Translation by Stand-Alone Voice Cloning: A Multispeaker Text-To-Speech Synthesis Approach based on Transfer Learning

Sakshi Bhajikhaye [2], Dr Sonali Ridhorkar [1], Vidhi Gautam [2], Mamta Soni [2], Mayank Badole [2], Adarsh Kant [2], Pranjali Rewatkar [2]

[1]HOD, Department of Computer Science, G.H. Raisoni Academy of Engineering and Technology, RTMNU, Maharashtra, India 440033.
[2]Student, Department of Computer Science, G.H. Raisoni Academy of Engineering and Technology RTMNU, Maharashtra, India 440033.

---***---

Abstract - *Stand Alone Language Translator is a speech to speech translation application for android mobile phone, which enables the translation of speech signals in a source language to the target language in the human voice, which is the same as the source voice. Stand Alone Language Translator includes three modules, Speech Recognition, Language Translation, and Speech Synthesis. The speech recognition module captures the voice or speech from the mobile user through Microphone, identifies then converts speech into text, and then the text is sent to Language Translation along with a sample voice for further process. Language Translation module does the process of translation, i.e., this module consists of a library for both languages, and when text is received by this module, it converts the text of one language to another selected by a user, and thus it sends the translated text to the last module. The speech Synthesis module acts as the text-to-speech translator, i.e., when it gets the translated text. This module processes translated text which converts it into speech and will provide the output to the user in the same voice as the source human voice. Thus, this language Translation application works by integrating all these three modules and gives the user the best output.*

Key Words: **— End-to-end speech-to-speech translation, Speech Recognition, Language Translation, Speech Synthesizer, Multilingual speech**

1.INTRODUCTION

Owing to this era, the global scenario adds to the demand for communication among speakers of different languages. Stand Alone Language Translator (SALT) enables the communication between people speaking in different languages. Stand Alone Language Translator being able to speak and have one's words translated automatically into the other person's language. This translator is used to convey the original tone and intent of a source language to the target language. Automatic speech to speech translation technology consists of three separate technologies: technology to recognize speech (speech recognition), technology to translate the recognized words (language translation), and technology to synthesize speech in the other person's language (speech synthesis). Speech to Speech Translation systems is often used in a specific situation which includes supporting conversations in non-native languages. The demand for trans-lingual conversations triggered by IT technologies has boosted research activities on Speech to Speech Translation technology. The work proposed for Speech to Speech Translation is a mobile application for an android platform that translates the real-time speech of one language into another required targeting language. A good speech-to-speech translation system can be characterized by its ability to keep intact the fluency and meaning of the original speech input.

2. REVIEW

CASMACAT is a modular, web-based translation workbench that offers advanced functionalities for computer-aided translation and the scientific study of human translation. MateCat is a tool whose objective is to improve the integration of machine translation and human translation within the so-called computer-aided translation framework. It provides translators with text editors that can manage several document formats and suitably arrange their content into text segments ready to be translated [3]. Curriculum learning (CL) might help avoid bad local minimums, hasten training convergence, and improve generalization. These advantages have been empirically demonstrated in various tasks, including shape recognition, object classification, and language modeling [1]. The advantage of SMT is that one does not require a deeper syntactic understanding of Source and Target languages [8]. Voice Translator is an android mobile application that helps the user to translate one language to another by using a Bluetooth environment which makes it possible to talk with every human being indifferent language [7]. The main goal of stand-alone voice conversion is to modify an utterance from the source speaker while keeping the linguistic contents unchanged in order to match the frequency of the target speaker [4].

3. PROBLEM DEFINITION

Let us consider a dataset of utterances grouped by their speaker where we denote the jth utterance of the ith speaker as uij. Utterances are indicated in the waveform domain. Then, we denote by xij -the log-mel spectrogram of the utterance uij. A log-mel spectrogram is a deterministic, noninvertible function that extracts the features of speech from waveform in order to handle speech in machine learning.

The encoder E computes the embedding in the following equation given below:

eij = E (xij; wE)

that corresponding to the utterance uij,

where wE are the parameters of the encoder.

Additionally, the authors define a speaker embedding as the centroid of the embeddings of the speaker's utterances:

-------(1)

The synthesizer is indicated by S, parametrized by wS, approximate to xij

Given,

ci and tij are the transcripts of utterance uij.

We have ,

xˆij = S (ci, tij; wS).

In our implementation, we use an utterance embedding rather than the speaker embedding, giving

xˆij = S (uij, tij; wS).

In the end, the vocoder V, parametrized by wV, is tasked to approximate uij given xˆij.

We have, uˆij = V (xˆij; wV).

One could train this framework in an end-to-end fashion with the following objective function:

Min wE, wS, wV LV(uij , V(S(E(xij ; wE), tij ; wS); wV))

Where LV is a loss function in the waveform domain.

4. METHODOLOGY

SALT is mainly divided into three main modules- Speech recognition module (speech-to-text conversion), language translation module, Speech synthesis module (text-to-speech conversion) as shown in the following Figure [a]:

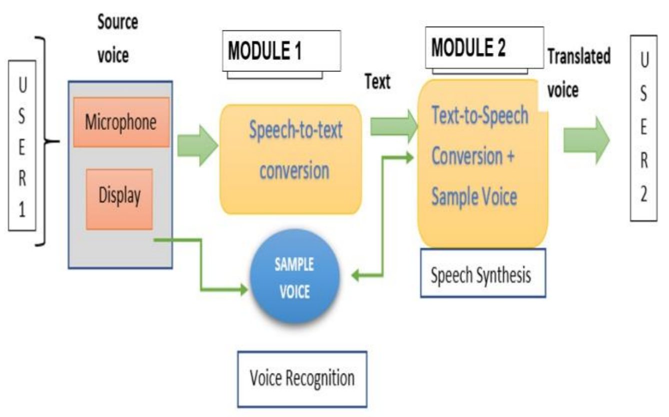


Fig [a]. Block Diagram of SALT framework

4.1. Speech-to-text conversion

The first module of our work is based on the Speech-to-text conversion, which is the Voice Recognition module. This also includes Speech Translation. Speech Translation is a function that instantly translates a spoken language into the selected foreign language. Here, we are using Google Translation API using a python script to record and convert the recognized Speech into Text format. A Speech-to-Text API is used to convert voice from the Microphone and generate English TEXT format from multiple languages.

This works on the Cloud platform, where it converts audio to text by applying powerful neural network models. In Speech-to-Text conversion, Google translates API's enables easy integration of Google Speech recognition technologies into developer applications. Cloud Translation can translate text between thousands of language pairs dynamically. Translation lets websites and programs integrate programmatically with the translation service. It also provides Natural Language Processing involves various technologies, such as sentiment analysis, entity recognition, entity sentiment analysis, and other text annotations, to developers. The speech-to-text conversion uses machine learning to reveal the structure and meaning of the text. The powerful, trained models of the Natural Language API empower developers to easily apply natural language understanding to their applications. It combines Natural Language with Speech-to-Text API to extract insights from audio conversations. During the initial stage of Cloud Translation, you need a project that has the Cloud Translation API enabled and credentials to make authenticated calls.

Google Neural Machine Translation (GNMT) translates a whole sentence at a time, rather than just letter by letter. It uses broad context to help it figure out the most relevant translation, which is then rearranged and adjusts with proper grammar. It uses deep learning techniques, in particular, long short-term memory networks, in order to translate a whole sentence at a time, which has been measured to be more accurate between English and French, German, Spanish, and Chinese. GNMT improves the quality of translations because it uses an example-based machine translation (EBMT) method. In this method, the system learns from millions of examples. It uses this broad context to help it figure out the most relevant translation. The GNMT network attempts interlingual machine translation. Most common words in English have at least two senses, which produces equal odds in the likely case that the target language uses different words for those different senses. The odds are similar in other languages to English.

4.2. Text to speech conversion

The second module of our work is based on Text-to-speech (TTS), which is the Speech Synthesis module, the process of synthesizing an artificial speech from a text prompt. Most of the research focus has been since gathered around making these deep models more efficient, sound more natural, or training them hastily. The current Text To Speech tools and functionalities provided by Google, such as the Google assistant6 or the Google cloud services, make use of these same models. The complete framework is a three-stage pipeline.

The framework is divided into three stages are as follows:

• A speaker encoder is the stage that derives an embedding from the short utterance of a single speaker. The embedding represents a meaningful representation of the voice of the speaker, such that similar voices are closed in latent space.

• A synthesizer that controls the embedding of a speaker and generates a spectrogram from the text.

• A vocoder is the stage that infers an audio waveform from the spectrograms generated by the synthesizer. It generates an embedding which is used to control the synthesizer, and a text is processed as a phoneme sequence is given as input to the synthesizer. The vocoder then takes the output of the synthesizer to generate the speech waveform. The following Figure [b] is illustrated below:

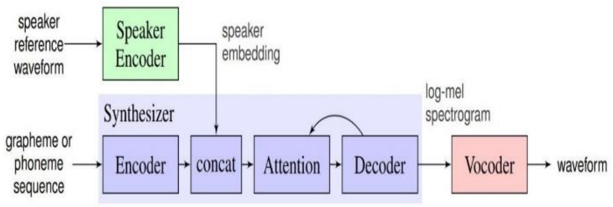


Fig b. Block diagram of TTS

The blue blocks in the above Figure represent a high-level view of the Tacotron architecture modified to allow conditioning on a voice. The above Figure is extracted from (Jia et al., 2018). This model can be trained separately and on distinct datasets. For the encoder, one looks for a model that is robust to noise and able to capture many characteristics of the human voice. A large corpus of many speakers would be preferred to train the encoder for the datasets of the synthesizer and the vocoder. The Transcripts are required, and the quality of the generated audio can be equally good as that of the data. Higher quality and annotated datasets are required. This shows that they are smaller in size.

Speaker Encoder: The speaker encoder needs to generalize well enough to produce meaningful embeddings on the dataset of the synthesizer, and even when trained on a common dataset, it still has to be able to operate in a zero-shot setting at inference time. We reproduced this model with a PyTorch implementation of our own. We synthesize the parts that are pertinent to SV2TTS as well as our choices of implementation.

A template is created for a person by deriving their speaker embedding from a few utterances. This process is called enrolment. The Generalized End to End loss simulates this process to optimize the model. To avoid segments that are mostly silent when sampling partial utterances from complete utterances, we use the webrtcvad8 python package to perform Voice Activity. Detection (VAD).

A last pre-processing step applied to the audio waveforms is normalization to make up for the varying volume of the speakers in the dataset. Figure [d] shows the steps to silence removal with VAD, from top to bottom. The orange line is the binary voice flag, where the upper value means that the segment is voiced and unvoiced when lower. In all our tests, the UMAP projections perfectly separate utterances from the test set of each of the three datasets, with large inter-cluster distances and small intra-cluster variance. The following Figure [c]UMAP projections of utterance embeddings from randomly selected batches from the train set at different iterations of our model. Utterances from the same speaker are represented by a dot of the same color.

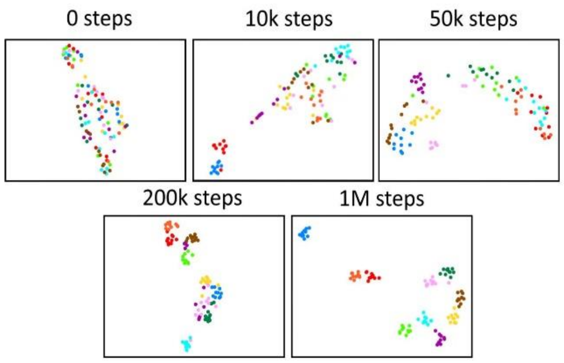


Figure [c]. UMAP projections of utterance embeddings from randomly selected batches from the train set at different iterations of our model.

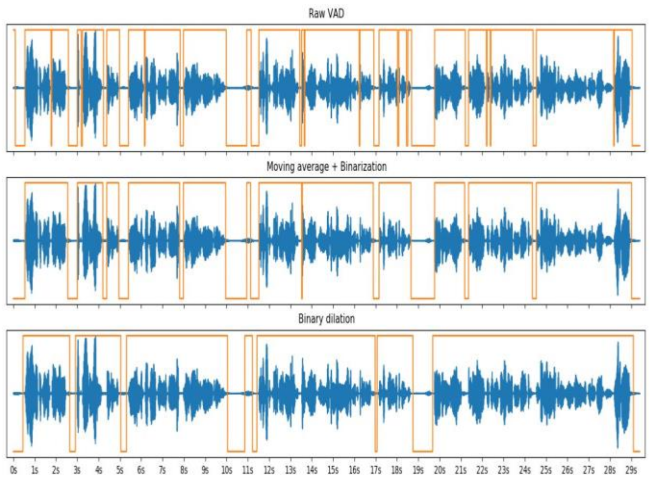
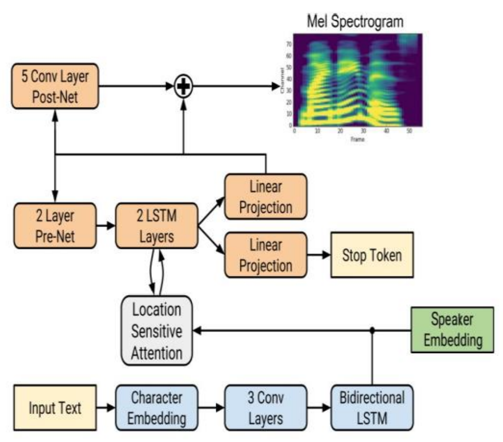


Figure [d]. Silence removal with VAD, from top to bottom

1) *Synthesizer:* The synthesizer is Tacotron without Wavenet. Tacotron is a recurrent sequence-to-sequence model. It predicts a mel spectrogram from the text. Tacotron features an encoder-decoder structure (not to be mistaken with the speaker encoder of TTS) that is bridged by a location-sensitive attention mechanism.

The entire sequence of frames goes through a residual post-net before it becomes the mel spectrogram. This architecture is represented in Figure [e]. We have used the automatic speech recognition (ASR) model to force-align the LibriSpeech transcripts to text. Tacotron usually operates faster than in real-time. We use a python implementation. We adapt this implementation to profile the noise and to clean the speech.



Figure[e]. Architecture of Synthesizer

2) *Vocoder:* The vocoder model which we use is an open-source PyTorch implementation that is based on WaveRNN.

Figure [f], which is given below, depicts the WaveRNN architecture. At each training step, a mel spectrogram and its corresponding waveform are divided into an equal number of segments. The spectrogram segment and the waveform segment t − 1 are given as input to the model. The mel spectrogram goes through an up-sampling network to match the length of the target waveform. As a result, the number of mel channels remains unchanged. A Resnet-like model uses the spectrogram as input to generate features that will control the layers throughout the transformation of the mel spectrogram to a waveform. The resulting vector is repeated to match the length of the waveform segment. This conditioning vector is then split equally four ways along the channel dimension, and the first part is concatenated with the up sampled spectrogram and with the waveform segment of the previous timestep. As a result, the vector goes through a diverse transformation with skip connections which include the first two GRU layers and a dense layer. The conditioning vector is concatenated between each step with the intermediate waveform. In the end, two dense layers produce a distribution over discrete values that correspond to a 9-bit encoding of mu-law commanded audio.

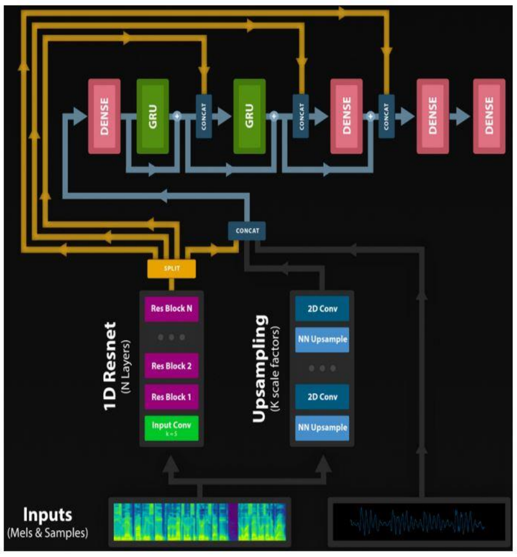
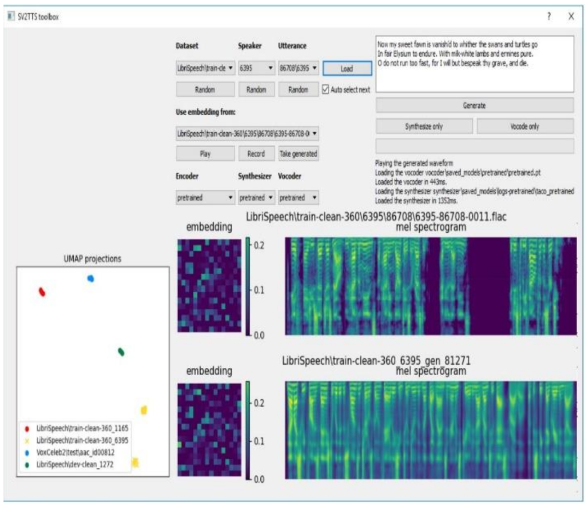


Figure [f]. WaveRNN architecture

3) *Text to speech tool box interface:* The interface of the toolbox can be seen in Figure [g]. The WaveRNN architecture is written in Python with the Qt4 graphical interface and is, therefore, a cross-platform. The image is best viewed on digital

support. A user begins the process by selecting an utterance audio file from any of the datasets available on their disk. The toolbox handles many popular speech datasets and can be customized to add up-to-date ones. Further, the user can also record utterances so as to clone one's own voice. Once an utterance is loaded, embedding will be computed, and the Uniform Manifold Approximate Projections will be updated reflexively. The mel spectrogram of the utterance is pulled out, but it is only for reference. We draw an embedding vector with a heatmap plot. This embedding is unidimensional vectors. The embeddings are drawn give us visual cues as to how two embeddings differ from each other. When an embedding has been enumerated, it can be used to generate a spectrogram. The user can write any arbitrary text in order to be synthesized. Note that the punctuation is not supported by our model and will be discarded. The user has to insert line breaks between parts that should be synthesized individually to tune the prosody of the generate utterance. The complete spectrogram is the concatenation of those parts. After synthesizing, the spectrogram will be displayed on the bottom right of the interface. While synthesizing the same sentences multiple times will yield contrasting outputs.

In the end, the user can generate the segment corresponding to the synthesized spectrogram with the vocoder. The progress of the generation is displayed by a loading bar. After the progress is displayed, the embedding of the synthesized utterance is generated on the left of the synthesized spectrogram. This embedding will also be projected with Uniform Manifold Approximate Projections. The user is unbound to take that embedding as a reference for further generation. The following Figure [g] depicts the toolbox interface.



Now my sweet fawn is vanish'd to whither the swans and turtles go
In fair Elysium to endure. With milk-white lambs and ermines pure.
O do not run too fast, for I will but bespeak thy grave, and die.

```
Playing the generated waveform
Loading the vocoder vocoder\saved_models\pretrained\pretrained.pt
Loaded the vocoder in 443ms.
Loading the synthesizer synthesizer\saved_models\logs-pretrained\taco_pretrained
Loaded the synthesizer in 1352ms.
```

Figure[g]. TTS Toolbox Interface

4.3. User Interface

We are using Flutter technology to develop our user interface. Flutter is an open-source UI software development kit created by Google. The major components of Flutter include the Dart platform, Flutter engine, Foundation library, Design-specific widgets, Flutter Dev Tools.

In Python, we have used a microwave framework. In order to connect the Flutter with Python, we have installed pip and flask. We then worked on two inputs- flask and just sonify. We have created a flask instance and assigned it to the app variable, and run it in debug mode. We have installed the HTTP (dart) package, which would be used in the later stage. We have used the setup empty string variable assigned to display the data. We have also framed a button with a final response which, on clicking it, will get us back to our python script to fetch the data and display it on the screen. We had done this by sending an HTTP get request to get the request from the data server. In return, we will get an URL from the flask application. In the flask, we have framed the routes and route path and used methods. We have defined a function that is executed after we hit that specific route and then return to sonify and execute it. Once we get the data from the python script, we decode it back to key-value format and store it in response and use the set state. We receive the data in the Flutter sent by the python script.

The above modules can be summarized using a web API which is developed on the cloud computer to reduce the cost-effectiveness of the whole project.

5. CONCLUSIONS

A stand-alone language translator is a device that bridges the gaps of language barriers with the intelligent system in real-time by translating the source language to the target language. At present, we need a real-time system that can translate multilingual speech with the matched frequency of the human voice with proper fluency and accuracy.

This system fulfills all the requirements. This device works on three main modules-speech recognition, language translation, and speech synthesis.

This is an effective device which is easily accessible to any of the Bluetooth device anywhere without the use of internet with minimum cost and is very easy to use. Therefore, in this situation, the system proposed will suffice the purpose reasonably well and minimize the communication inefficiencies.

REFERENCES

[1] Takatomo Kano, "End-to-End Speech Translation With Transcoding by Multi-Task Learning for Distant Language Pairs," IEEE/ACM Transactions on Audio, Speech, and Language Processing (Volume: 28), Year:2020

[2] Ye Jia, Yu Zhang, Ron J. Weiss," Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis," Google Inc., 2019

[3] Rashid Ahmad, Priyanka Gupta, Nagaraju Vuppala," Transzaar: Empowers Human Translators, "18th International Conference on Computational Science and Applications (2018)

[4] M. Teduh Uliniansyah, Hammam Riza, Agung Santosa, Gunarso, Made Gunawan, Elvira Nurfadhilah," Development of Text and Speech corpus for an Indonesian Speech to Speech Translation System," Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Technique, South Korea (2017)

[5] Mrinalini K and Vijayalakshmi P," Hindi-English Speech-to-Speech Translation System/or Travel Expressions," International conference on communication. Year:2015

[6] M.D. Faizullaha Ansari, R.J. Shaji," Multilingual speech to speech translation system in Bluetooth environment," International Conference on Control, Communication and Computational technology (2014)

[7] Akshay Suresh Deshpande, Keshav Sheshrao Ambulgekar, Kedar Raghunath Joshi," Voice to Voice Language Translation System," International Journal of Engineering Research & Technology (10, October-2014)

[8] Karunesh Arora, Mukund Kumar Roy," Speech to Speech Translation: a communication boon," CSIT (7 June 2013)

[9] Yuxuan Wang, RJ Skerry-Ryan. Tacotron: Towards End-to-End Speech Synthesis, Google, Inc-2017.

[10] Leland McInnes and John Healy. UMAP: Uniform manifold approximation and projection for dimension reduction. 02 ,2018.