Scaling and Training of Speech Synthesis Models

Niyathi Aiella  
Krannert School of Management  
Purdue UniversityWest Lafayette, USA  
naiella@purdue.edu

Abhishek Nambiar  
Krannert School ofManagement  
Purdue UniversityWest Lafayette, USA  
nambiar3@purdue.eduTvisha Goswami  
Krannert School of Management  
Purdue UniversityWest Lafayette, USA  
tgoswami@purdue.edu

Amrit Singh  
Krannert School ofManagement  
Purdue UniversityWest Lafayette, USA  
singh992@purdue.eduNitin Jayan  
Krannert School of Management  
Purdue UniversityWest Lafayette, USA  
njayan@purdue.edu

Matthew A. Lanham   
Krannert School ofManagement  
Purdue UniversityWest Lafayette, USA  
lanhamm@purdue.edu

*Abstract*— A vast majority of the languages and consequently, their speakers are marginalized and displaced from the digital sphere, due to lack of support from existing speech synthesis technology service providers Moreover, the ubiquitous employment of speech synthesis technology, especially Text-To-Speech (TTS) for assistive technology services and customer service to name a few, renders this lack of coverage or support for various languages, an avenue that is ripe for a variety of opportunities.

This project aims to assist a global, faith-based nonprofit in developing a data pipeline for their TTS models that is both scalable and easily replicable. This pipeline will take audio and text data as inputs, and will generate trained and evaluated speech synthesis models as outputs. With this pipeline, the organization will have the capability to create speech synthesis models for a large number of new languages.

To develop the automated machine learning pipeline, we will be leveraging the services provided by AWS – chiefly AWS Step Function, S3, Lambda and Elastic Compute 2, with the architecture in line with the AWS well architected framework. Further we will be using the Montreal Forced Aligner to align the text with speech which is then used to train the model utilizing by Coqui TTS.

Keywords— Text-to-Speech, EC2, Step Function, Coqui TTS Montreal Forced Aligner, speech synthesis

# Introduction

There are a little over 7100 documented languages in the world, according to Busuu, the language learning application[1]. Of these, only a fraction are supported by the 3 major players in the speech synthesis space, namely Google Cloud which supports 100 languages, and Microsoft’s Azure and AWS’s Polly with support for 20 languages each. This leaves a lion’s share of mostly dialects and regional languages unsupported for digital integration. This ultimately means several sections of the present day society cannot take advantage of mobile or digital technology based services. According to the Scientific American, 700 million illiterate people across the globe are unable to receive the benefits of mobile technology due to this very reason[2].

While there has been considerable headway in integrating these languages to digital services by way of speech synthesis technology, it still leaves a lot to be desired. Moreover, this also presents organizations unparalleled opportunities to tap into this market as the global text-to-speech (TTS) market was valued at $2.8 billion in 2021, and is projected to reach $12.5 billion by 2031, growing at a CAGR of 16.3% from 2022 to 2031, per a report published by Allied Market Research [3].

One of the principal hindrances is the time and effort required to train and deploy a TTS model. Some of them are as follows,

1. Data pre-processing: Collecting and preparing large amounts of speech audio and corresponding transcripts for training can be a time-consuming and labour-intensive task.
2. Model training: Training a speech synthesis model requires a significant amount of computational resources and can take a long time.
3. Model evaluation: Evaluating a speech synthesis model can be difficult, as it requires listening to and assessing the quality of the synthesized speech.
4. Model deployment: Deploying a speech synthesis model in a production environment can be challenging, as it requires integrating the model with other systems and services.
5. Data security and governance: Difficulty in ensuring data security and compliance with regulations.

In this project, we aim to design and develop an automated machine learning pipeline that would help our client, a global faith based non-profit, tackle the above challenges.

The main components used in the architecture are.

* Amazon S3: Amazon Simple Storage Service (Amazon S3) is an object storage service offered by Amazon Web Services (AWS). It provides a scalable, secure, and durable way to store and access a large amount of data from anywhere on the web. With Amazon S3, you can store and retrieve any amount of data, at any time, from anywhere on the web
* Amazon Web Services (AWS) Lambda: It is a serverless computing platform provided by AWS. It allows you to run code in the cloud without having to manage infrastructure, making it easier to build, run, and scale applications.With AWS Lambda, you can run your code in response to events, such as changes to data in an Amazon S3 bucket. This makes it a useful tool for building event-driven and microservices-based architectures, as well as for running background tasks that don't require a long-lived infrastructure. Lambda automatically manages the underlying infrastructure, including provisioning and scaling resources as needed, so you can focus on writing and deploying your code. You only pay for the compute time you consume, which can help you reduce costs and increase efficiency compared to traditional infrastructure-based solutions.
* Amazon Elastic Compute Cloud (Amazon EC2): A web service provided by Amazon Web Services (AWS) that provides resizable compute capacity in the cloud. EC2 enables you to launch virtual machines (known as instances), with a variety of operating systems, including Linux and Windows, and configure network and storage resources as needed.
* Amazon SageMaker: A fully managed machine learning platform provided by Amazon Web Services (AWS). It provides tools and services to help developers and data scientists build, train, and deploy machine learning models at scale.

# DATA

The data for training the TTS model is sourced from Open Bible, a free, open – source website. SIL has leveraged Open Bible as it provides accurate, contemporary translations and formats around the world. Additionally, by releasing complete Biblical texts for free under Creative Commons licensing, the traditional Bible licensing, publishing, and distribution pipeline can be completely replaced with a much faster, more efficient vehicle. We will be using the audio files sourced from Open Bible in conjunction with the text translation for the bible provided in GitHub by the SIL team. SIL Team has built the code base for aligning the text and audio data. We are using Hausa, Hindi, Achi and Zapotec language to train TTS model.

The statistics from our dataset are displayed in the figure below. Each language has its own folder, which contains a train, validation, and test split. Additionally, each language has a metadata file attached to it, the contents of which are displayed in the table below.

|  |  |
| --- | --- |
| **Field** | **Description** |
| languageCode | code for language |
| translationId | Translation ID |
| languageName | Language |
| languageNameInEnglish | English Language name |
| dialect | Dialect of the language |
| homeDomain | domain from where the bible data is sourced |
| title | Bible title |
| description | Description of the bible |
| Redistributable | redistributable flag |
| Copyright | copyright information |
| UpdateDate | Date the corpus updated |
| publicationURL | Publication URL format of the bible |
| OTbooks | Number of books in the Old Testament |
| OTchapters | Number of chapters in the Old Testament |
| OTverses | Number of Verses in the Old Testament |
| NTbooks | Number of books in the New Testament |
| NTchapters | Number of books in the New Testament |
| NTverses | Number of books in the New Testament |
| FCBHID | ID of the project |
| Certified | is translation certified |
| inScript | translation script in Python |
| swordName |  |
| rodCode |  |
| textDirection | direction of text |
| downloadable | is the file downloadable |
| font | font of the translation |
| shortTitle |  |
| PODISBN |  |
| script | script of the text |

# Literature Review

The paper "BibleTTS: a large, high-fidelity, multilingual, and uniquely African speech" (Meyer et al., 2022) conducted a study on an open dataset for ten sub-Saharan African languages. The authors made use of high-quality audio from the BibleTTS corpus and divided the chapters into manageable recording segments of 30 seconds by breaking them down to the verse level. This approach will be adopted in our process. Additionally, we will be exploring a selection of acoustic models, vocoders, training strategies, and multi-speaker and multilingual generalizations for Indian languages in Neural TTS systems, as discussed in "Towards building text-to-speech systems for the next billion users" (Kumar et al., 2022).

In a recent study, Rusell et al. (2022) discussed the design, collection, and verification of a bilingual text-to-speech synthesis corpus for Welsh and English from which we have understood how to approach the problem from a perspective of a different language other than English. Furthermore, to understand the role of AWS in speech synthesis, we went through the paper which talked about VocBench, a framework for benchmarking neural vocoders on speech synthesis tasks (AlBadawy et al., 2021). The purpose of VocBench was to provide a standard and comprehensive evaluation approach for the speech community. The results of the study showed both objective and subjective differences between the tested vocoders and the evaluations were performed using AWS for reproducibility. These studies provided valuable insight into the field of speech synthesis and the use of AWS in speech synthesis evaluations.

To further enrich our process, we also analyzed the paper "MaSS: A Large and Clean Multilingual Corpus of Sentence-aligned Spoken Utterances Extracted from the Bible" (Boito et al., 2020). This study focused on Text-to-Speech (TTS) system development using 20 hours of speech data in 8 different languages (Basque, English, Finnish, French, Hungarian, Romanian, Russian and Spanish). The authors applied both speech-to-text and speech-to-speech alignment processes to create the corpus. The speech-to-text alignment was performed using the Maus forced aligner, and the audio files were segmented into smaller units (verses) by aligning the TextGrid files with the written version of the Bible. This study provided valuable insights into the creation of large-scale TTS corpora and the alignment of speech and text data which we will be using for our analysis and build of our pipeline.

## Montreal Forced Aligner

Forced alignment is a technique to take an orthographic transcription of an audio file and generate a time-aligned version using a pronunciation dictionary to look up phones for words. The Montreal Forced Aligner uses a four-step training process. (McAuliffe et al., 2017):

1. The first step involves aligning monophone models, where each phone is modeled the same regardless of its context.
2. The second step uses triphone models, where the context on either side of a phone is considered for acoustic models.
3. The third step performs LDA+MLLT to learn a transform of the features that makes each phone’s features maximally different.
4. The final step enhances the triphone model by accounting for speaker differences and calculates a transformation of the mel frequency cepstrum coefficients (MFCC) features for each speaker.

## Coqui TTS

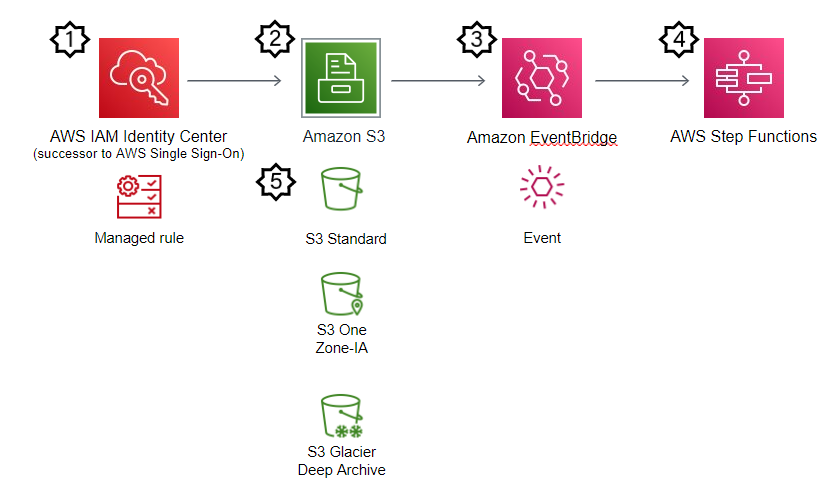
It leverages the power of neural text-to-speech (NTTS) technology, a form of machine learning that trains models using large datasets of speech audio and text pairs, resulting in more natural-sounding speech compared to traditional rule-based TTS systems.

The implementation of NTTS in Coqui TTS is based on the DeepMind Tacotron 2 architecture, a sequence-to-sequence model that transforms text into a spectrogram, a visual representation of sound frequency, and then generates waveform audio through a WaveNet-based neural network.

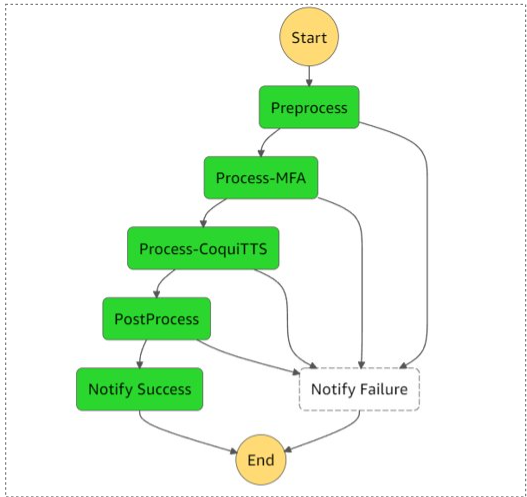
## Data Pipeline build

As numerous languages need to pass through MFA and Coqui TTS, it gets difficult to monitor the progress manually. Hence, SIL has chosen to build an automated pipeline in AWS in which they will drop the text files in S3 buckets and get the processed audio output in the end.

AWS Services for pipeline:



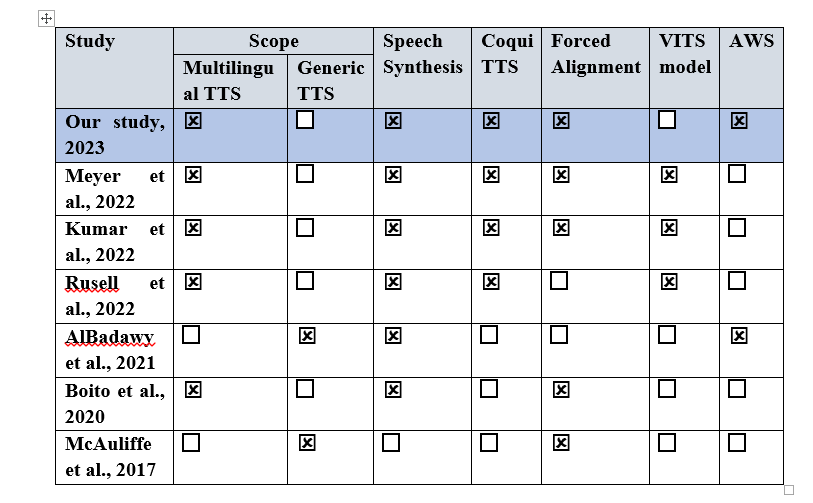
1. File storage in S3: amazon Simple Storage Service (Amazon S3) is an object storage service offered by Amazon Web Services (AWS). It provides a scalable, secure, and durable way to store and access a large amount of data from anywhere on the web. all the input and output files would be stored in S3 buckets as S3 storage
2. Leverage Eventbridge: Amazon EventBridge Event Bus is a serverless event bus that helps you receive, filter, transform, route, and deliver events. Eventbridge is a trigger which has an end point as step function in our pipeline. It will trigger as soon as something is dropped in the input files and reach the end point for further implementation of steps in the pipeline.
3. Step Function: AWS Step Functions is a visual workflow service that helps developers use Amazon Web Services (AWS) to build distributed applications, automate processes, orchestrate microservices, and create data and machine learning (ML) pipelines. This service contains steps inside it which will each have a service embedded in it according to our architecture. They will be implemented one after the other in a pipeline fashion for the process to succeed.



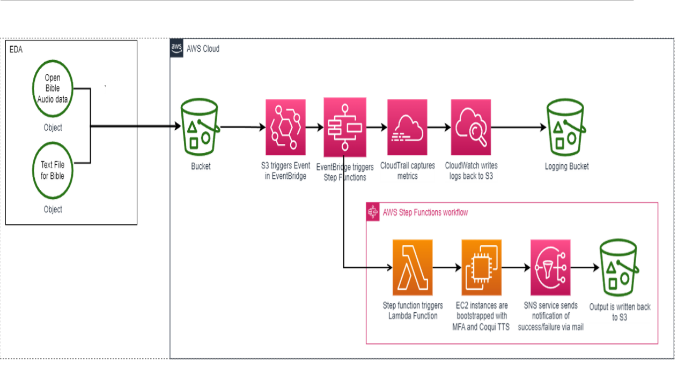
1. EC2 Instance: A web service provided by Amazon Web Services (AWS) that provides resizable compute capacity in the cloud. EC2 enables you to launch virtual machines (known as instances), with a variety of operating systems, including Linux and Windows, and configure network and storage resources as needed. This is our main Linux Machine where all data pre-processing would happen. Coqui TTS Model would run on this machine and the final output would be sent to an S3 Bucket.
2. Lambda Function: It is a serverless computing platform provided by AWS. It allows you to run code in the cloud without having to manage infrastructure, making it easier to build, run, and scale applications. With AWS Lambda, you can run your code in response to events, such as changes to data in an Amazon S3 bucket. This makes it a useful tool for building event-driven and microservices-based architectures, as well as for running background tasks that don't require a long-lived infrastructure. We will have python code in our lambda function which will remove the files in Input S3 bucket and drop them in Archive S3 bucket for ease of identification after the pipeline process has started implementation.
3. SNS or Simple Notification Service: This is the final step of the step function which will be triggered after the entire process has been finished. It will send out emails to selected people about the successful/failed implementation of the process. This step will mark the end of our process and automated pipeline.

# *Summary Table*

The following is a summary of the literature review conducted on text-to-speech synthesis. Our study stands out for its comprehensive approach, as it thoroughly examines previous research papers to identify areas for improvement and develop a cutting-edge solution.



# methodology



This diagram represents the methodology we followed, and the steps we have taken for the AWS pipeline to be executed. There will be many steps involved in building a pipeline, and ours requires special attention as it takes into consideration the training of a speech synthesis model on top of an EC2 instance.

The entire pipeline has been built on top of AWS and is completely automated hence requires no manual intervention at any point after dropping the required input files in S3.

##### References

1. <https://blog.busuu.com/most-spoken-languages-in-the-world/#:~:text=Well%2C%20more%20than%207%2C100%20languages,spoken%20by%20only%20eight%20people>.
2. <https://www.scientificamerican.com/article/why-ai-needs-to-be-able-to-understand-all-the-worlds-languages/>
3. <https://www.alliedmarketresearch.com/text-to-speech-tts-market-A31559#:~:text=The%20global%20text%2Dto%2Dspeech,16.3%25%20from%202022%20to%202031>.
4. Meyer, J., Adelani, D.I., Casanova, E., Öktem, A., Whitenack, D., Weber, J., Kabongo, S., Salesky,
5. E., Orife, I., Leong, C., Ogayo, P., Emezue, C., Mukiibi, J., Osei, S., Agbolo, A., Akinode, V., Opoku, B., Olanrewaju, S., Alabi, J., & Muhammad, S. (2022). BibleTTS: a large, high-fidelity, multilingual, and uniquely African speech. <https://arxiv.org/pdf/2207.03546.pdf>
6. Kumar, G.K., Praveen, S.V., Kumar, P., Khapra, M.M., Nandakumar, K. (2022). Towards building
7. text-to-speech systems for the next billion users. <https://arxiv.org/pdf/2211.09536.pdf>
8. Russell, S.J., Jones, D.B., Prys, D. (2022). BU-TTS: An Open-Source, Bilingual Welsh-English,
9. Text-to-Speech Corpus. <https://aclanthology.org/2022.cltw-1.15.pdf>
10. AlBadawy, E.A., Gibiansky, A., He, Q., Wu, J., Chang, M.C., Lyu, S. (2021). Vocbench: A neural
11. vocoder benchmark for speech synthesis. <https://arxiv.org/pdf/2112.03099.pdf>
12. Boito, M.Z., Havard, W.N., Garnerin, M., Ferrand, E.L., Besacier, L. (2020). MaSS: A Large and
13. Clean Multilingual Corpus of Sentence-aligned Spoken Utterances Extracted from the Bible. <https://arxiv.org/pdf/1907.12895.pdf>
14. McAuliffe, M., Socolof, M., Mihuc, M., Wagner, M., & Sonderegger, M. (2017). Montreal Forced
15. Aligner: trainable text-speech alignment using Kaldi. In Proceedings of the 18th Conference of the International Speech Communication Association.
16. https://montreal-forced-aligner.readthedocs.io/en/latest/\_downloads/998b0c31eadaf048e8e3de805b9ef8e6/MFA\_paper\_Interspeech2017.pdf