SCALING SPEECH SYNTHESIS MODELS ON AWS

Niyathi Aiella
naiella@purdue.edu

Tvisha Goswami
tgoswami@purdue.edu

Nitin Jayan
njayan@purdue.edu

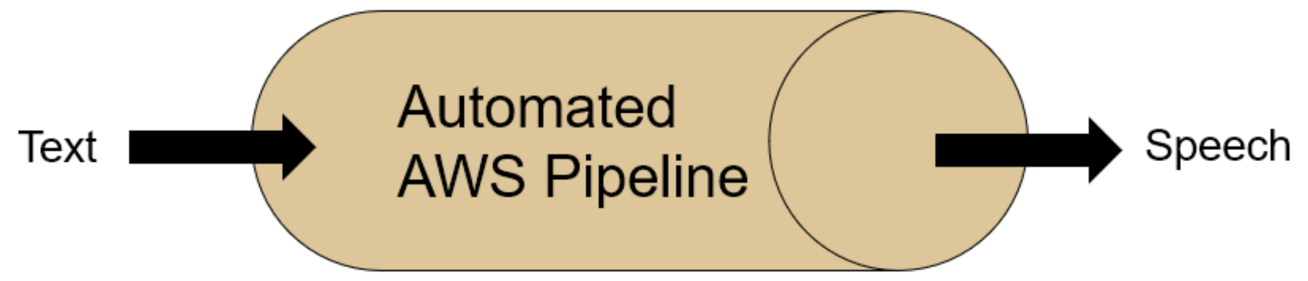
Abhishek Nambiar
nambiar3@purdue.edu

Amrit Singh
singh992@purdue.edu

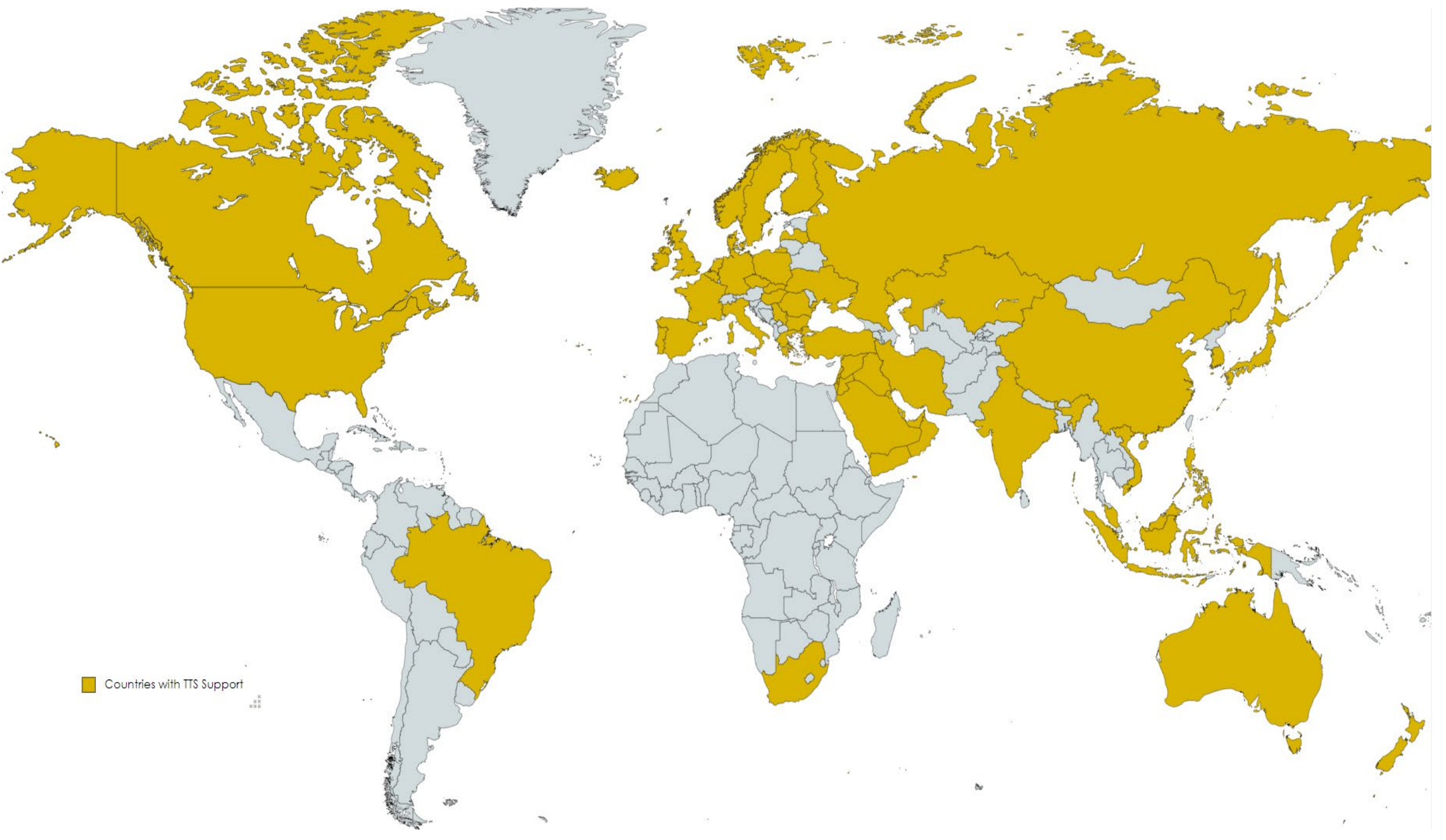
Mentor: Matthew A. Lanham; lanhamm@purdue.edu
Purdue University, Daniels School of Business

BUSINESS PROBLEM FRAMING

- We have worked in collaboration with SIL in developing an AWS-based data pipeline that is both scalable and reproducible. The pipeline will take audio and text data as inputs and generate trained and evaluated speech synthesis models as outputs.



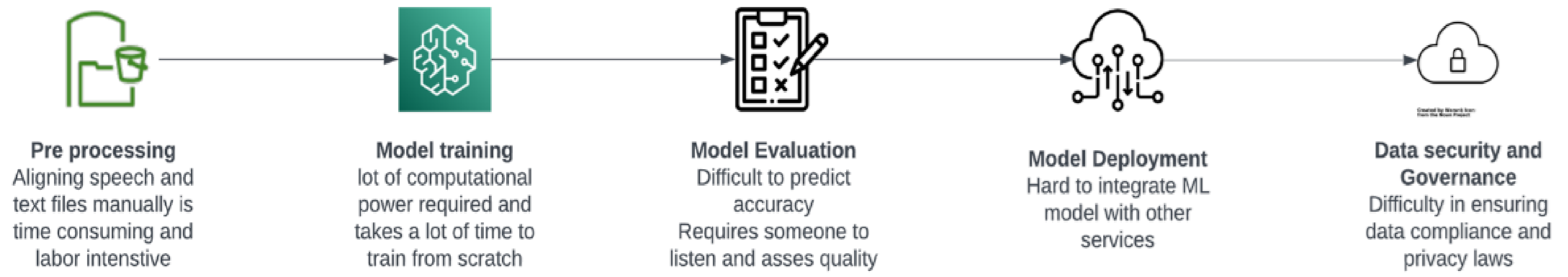
- By implementing this pipeline, SIL will be able to generate speech synthesis models in hundreds of additional languages.



PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

ANALYTICAL PROBLEM FRAMING

We have developed a scalable data pipeline for Text-To-Speech (TTS) models to increase language coverage for TTS technology and provide opportunities for marginalized language speakers. SIL has a vast database of over 3600 languages with audio books, stories, and bibles, and they want to automate the Text to Speech Recognition process using AWS services. This will make it easier for people worldwide to access the Bible in their native language.



Pre processing
Aligning speech and text files manually is time consuming and labor intensive

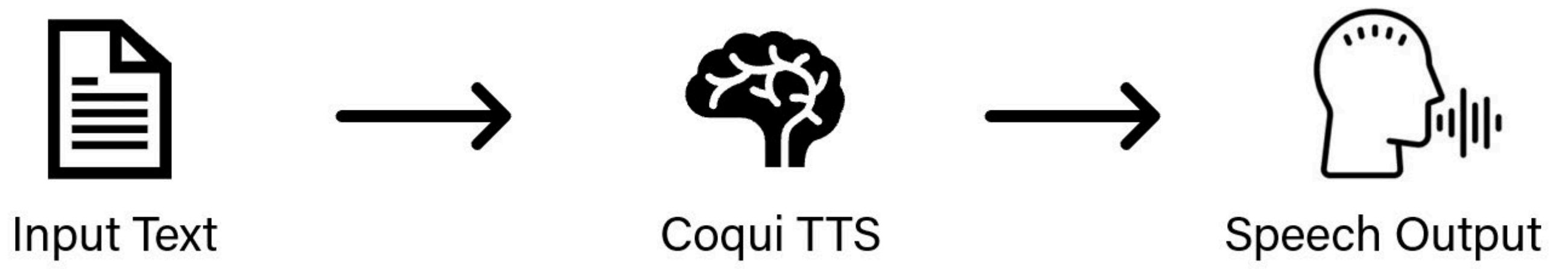
Model training
lot of computational power required and takes a lot of time to train from scratch

Model Evaluation
Difficult to predict accuracy
Requires someone to listen and asses quality

Model Deployment
Hard to integrate ML model with other services

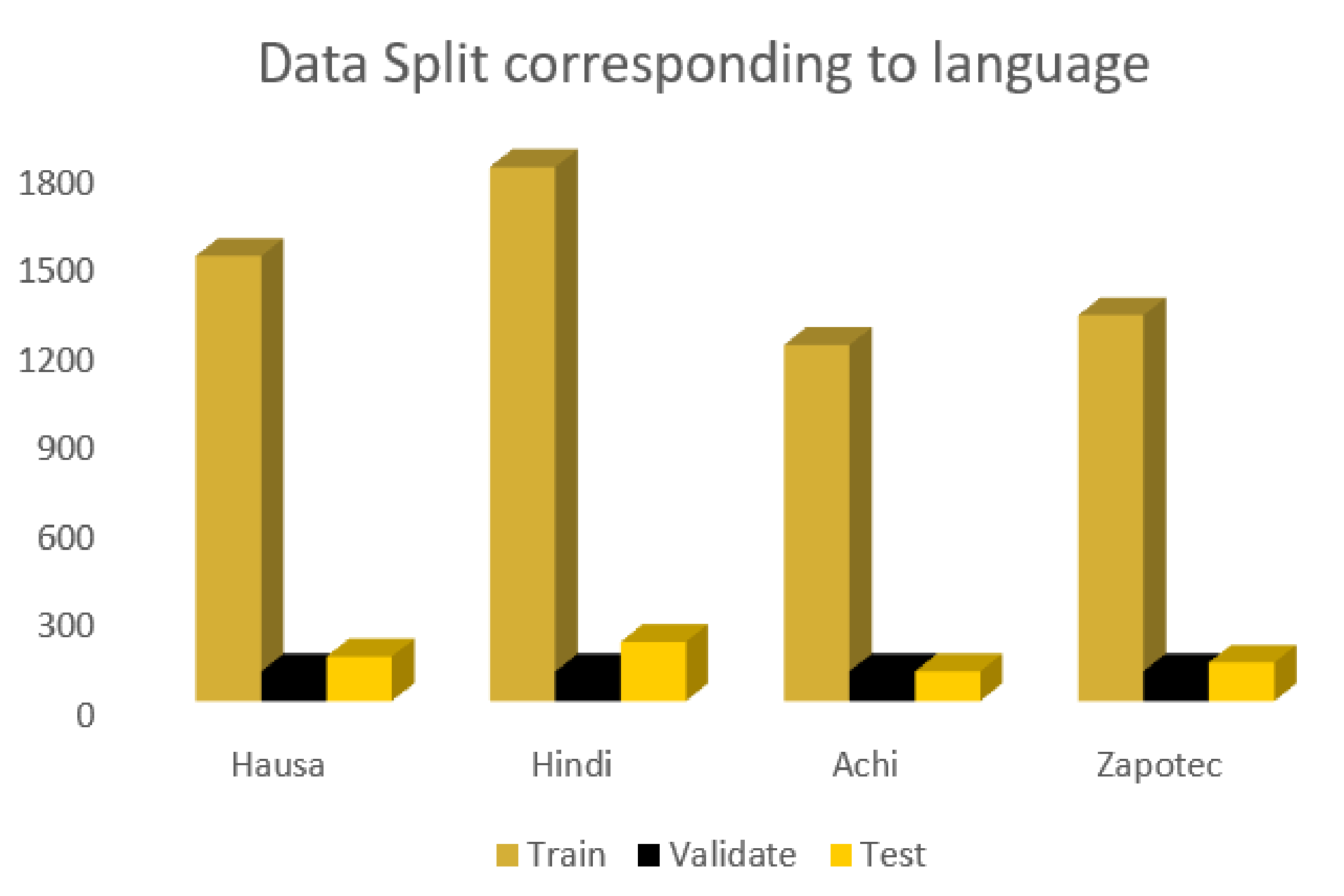
Data security and Governance
Difficulty in ensuring data compliance and privacy laws

We only have a training dataset as we only train a language using a particular Audio Bible. We can define metrics based on how fast the pipeline is working compared to manual processing of the Audio Files. We can define this project as a success when we can decrease the timing by at least 25%.

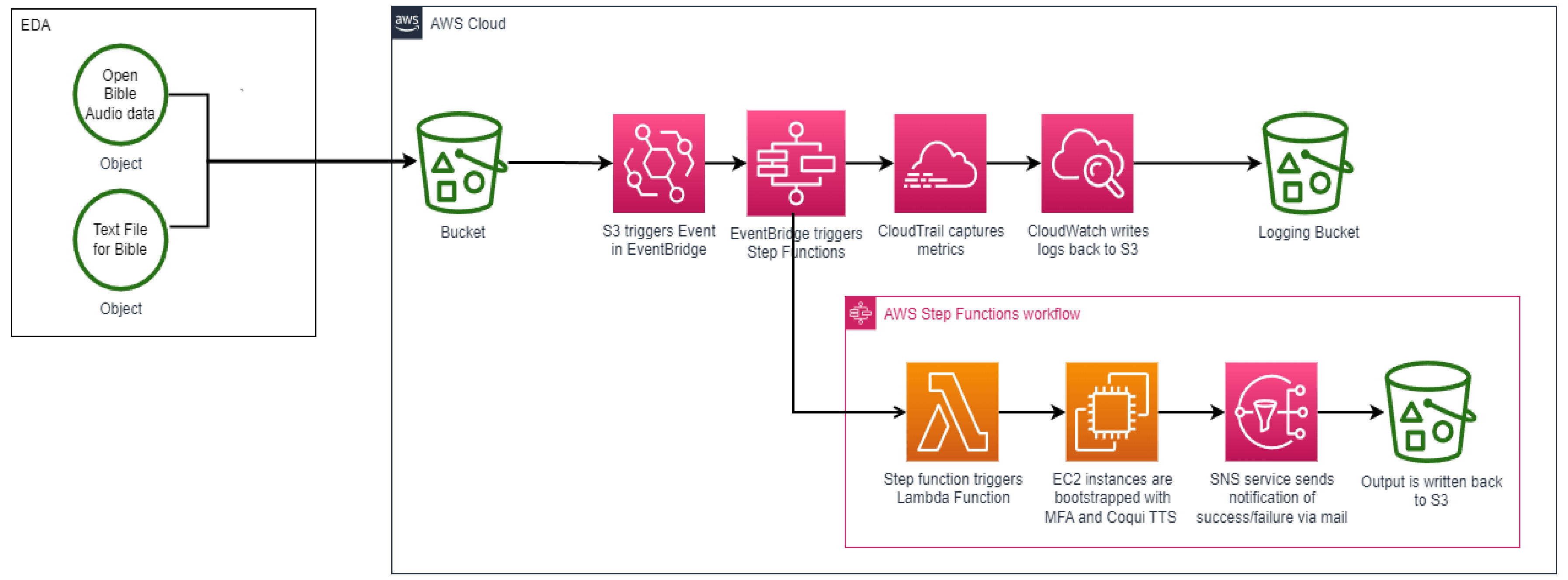


DATA

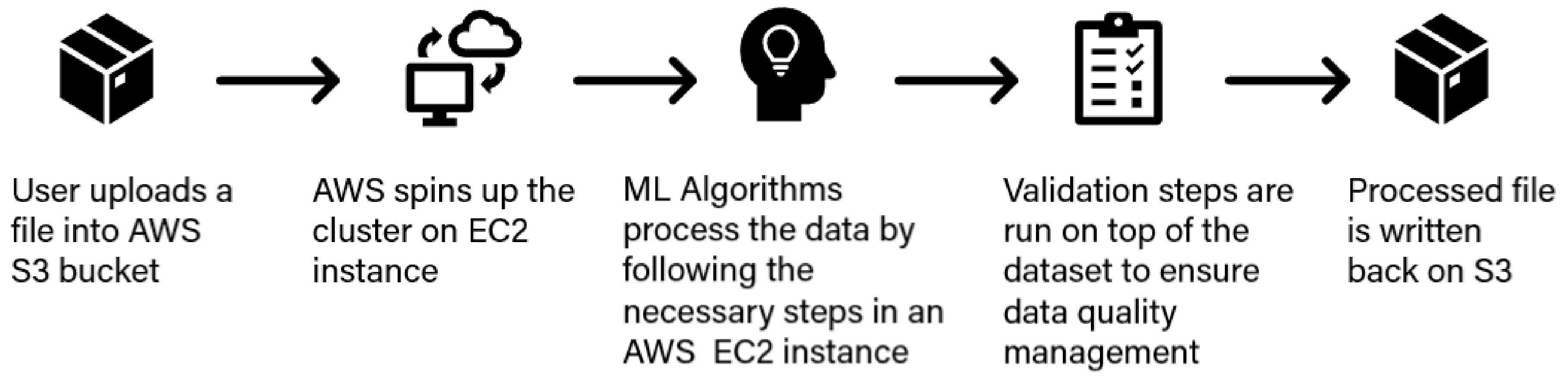
- The TTS model is trained using data from Open Bible (https://www.openbible.info/), which provides accurate and contemporary translations.
- Audio files from Open Bible are used in conjunction with text translations provided by the SIL team
- The TTS (https://github.com/coqui-ai/TTS) model is trained using Hausa, Hindi, Achi, and Zapotec languages, and each language has its own folder containing train, validation, and test splits.



METHODOLOGY



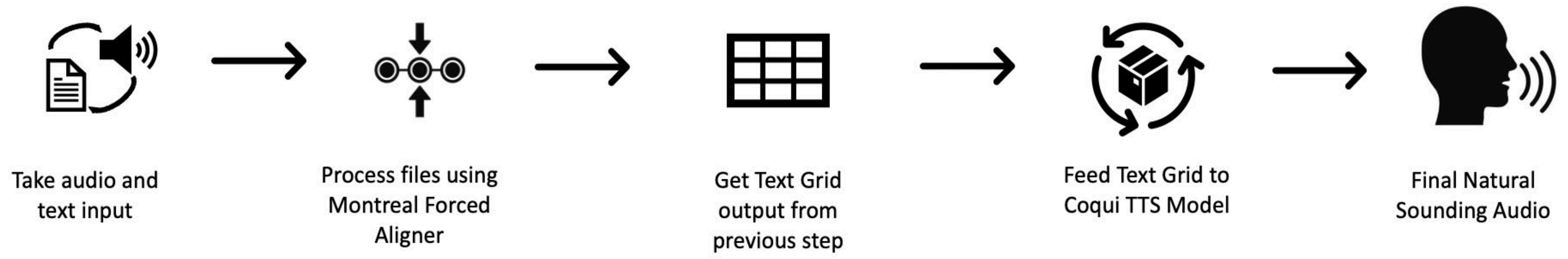
EDA
Open Bible Audio data
Object
Text File for Bible
Object
AWS Cloud
Bucket
S3 triggers Event in EventBridge
EventBridge triggers Step Functions
CloudTrail captures metrics
CloudWatch writes logs back to S3
Logging Bucket
AWS Step Functions workflow
Step function triggers Lambda Function
EC2 instances are bootstrapped with MFA and Coqui TTS
SNS service sends notification of success/failure via mail
Output is written back to S3



1. User uploads a file into AWS S3 bucket
2. AWS spins up the cluster on EC2 instance
3. ML Algorithms process the data by following the necessary steps in an AWS EC2 instance
4. Validation steps are run on top of the dataset to ensure data quality management
5. Processed file is written back on S3

MODEL BUILDING

- Using Audio and Text files as input to Montreal Forced Aligner (https://montreal-forced-aligner.readthedocs.io/en/latest/index.html), we train MFA to get a Text Grid output
- This text grid output has audio aligned to the respective words along with a timestamp at which the word appears
- This text grid is fed to Coqui Text-To-Speech model that generates a natural sounding
- The final audio files are added to an output S3 bucket

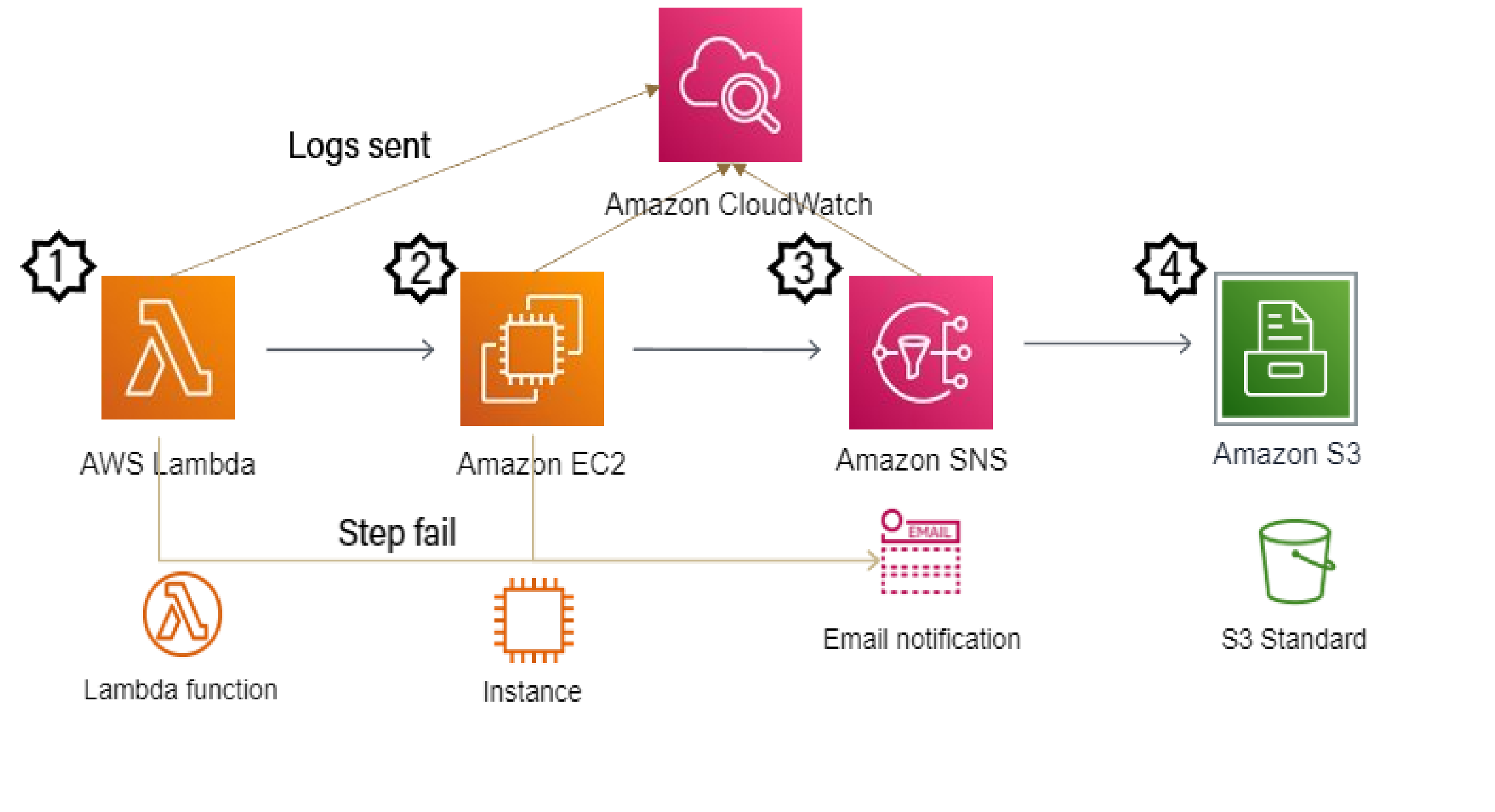


1. Take audio and text input
2. Process files using Montreal Forced Aligner
3. Get Text Grid output from previous step
4. Feed Text Grid to Coqui TTS Model
5. Final Natural Sounding Audio

DEPLOYMENT AND LIFECYCLE MANAGEMENT

Deployment of our Text-To-Speech model happens in an automated fashion on AWS. We are making use of an Event bridge trigger which triggers the Step Function every time there's an object drop in S3 bucket.

- The Data creation and storage is done through S3 input bucket
- Usage of data occurs in EC2 instance where Coqui TTS runs the Text-To-Speech model
- Data is archived into another S3 folder and all the logs are stored in Amazon CloudWatch logs, and an SNS Email notification is sent for monitoring purposes



1. Step function triggers Lambda function to launch instances in EC2
2. Scripts are installed onto EC2 instances and executed
3. SNS sends out email notification to specified list for success/step failure
4. Output files are written back on S3, can create a folder for logs

CONCLUSIONS

- This project has created a scalable and reproducible AWS-based data pipeline to aid SIL in generating speech synthesis models in numerous languages by taking audio and text data as inputs and producing trained and evaluated speech synthesis models as outputs.
- To accomplish this, Coqui TTS, Montreal Forced Aligner, GitHub, and various AWS services were utilized. The success of the project will enable SIL to create speech synthesis models in hundreds of new languages, enhancing their digital content accessibility.

ACKNOWLEDGEMENTS

We would like to thank Professor Matthew Lanham and our industry partner-SIL International for this opportunity, their guidance, and support on this project.

informs BUSINESS ANALYTICS CONFERENCE