Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Telecommunications and Media Informatics

Project Lab Report (#1)

Title: Voice Conversion Technology and its Application with Emotional Speech

Written by: Shaima Al Balushi **Neptun#:** S3Y2FA

Field: Computer Engineering **Specialization:** Data Science
E-mail: shaima89albalushi@gmail.com

Supervisor: Al-Radhi Mohammed
E-mail: mohammed.alradhi@vik.bme.hu

Academic year: 2023/2024/1 Fall semester

Contents

Abstract

Speaking serves as the most commonly employed and instinctive mode of human communication.Voice conversion aims to alter nonlinguistic elements while preserving speech essence, a field studied extensively, especially in transforming speakers. The Voice Conversion Challenge (VCC) from 2016 and 2018 facilitated exploration of various conversion technologies. "Sprocket," a standout freeware in VCC 2018, encompasses diverse methods, from traditional vocoder-centric and Gaussian Mixture Models to innovative vocoder-free techniques.

Sprocket's adaptability stems from its multiple functions, which allow end-users to easily recreate transformed sounds using VCC datafiles. It also provides an environment for developing VC systems using various parallel speech databases. Precise duration alignment between source and targeted output features, GMM and Differential GMM training, synthesis of waveform, feature converting, and Acoustic extraction of feature are all critical core activities inherent within sprocket.

This project is important because it falls within the field of cybernated audio processing and has potential applications in enhancing user experiences, individualized digital assistants, and human-computer interaction. The study deeply explores voice conversion, focusing on refining speech properties for accurate voice imitation.

The study extensively explores voice conversion techniques, emphasizing altering speech properties for precise imitation. To ensure reliable results, it suggests thorough investigation, algorithm selection, testing, and validation. With an eye toward the future, the research aims to propel improvements in voice translation methods and provide the foundation for complex emotional expressions in synthetic speech.

In this report, the exploration and implementation of the "sprocket-vc" open-source software for voice conversion stands as a pivotal focus. The report outlines the study of step-by-step procedure and commands utilized for installing the "sprocket-vc" package, cloning the associated repository from GitHub, and subsequently downloading and initializing the VCC2016 datasets. The report delves into the technical details of initializing the software for distinct speakers, like: SF1 and TM. At a sampling rate of 16000Hz. Furthermore, it elucidates the comprehensive execution of the "run_sprocket.py" script, which encompasses various functionalities (-1 through -5) for voice conversion between the specified speakers using the "sprocket-vc" framework. This abstract captures the sequential steps undertaken in the utilization and application of the "sprocket-vc" software, laying the groundwork for a detailed exploration within the report's context. Added to that, voice conversion methods within sprocket that rely on Gaussian Mixture Models (GMM) were discussed. The GMM techniques were effectively applied. Alongside, F0 transformations has been executed.

The sequence commands assisted the successful installation and application of the "sprocket-vc" package, enabling voice conversion operations with the VCC2016 dataset involving the Same-gender Voice Conversion (SF1-SF2 & TM1-TM2) and the Cross-gender Voice Conversion (SF1-TM1). Those commands were python programming executed on GoogleColab via a browser. However, Throughout this study, several notable challenges emerged, posing significant complexities and hurdles to overcome be it technical or documentational.

1. Introduction

Technology and human communication interact through the field of speech processing, which is a complex interaction of verbal expression and emotional nuance [1]. This research delves further into the topic of Voice Conversion Technology (VCT) and how it may be used to transform speech from neutral into a wide range of emotions. The main goal of this project is to develop a rule-based voice conversion system that can be used to add a wide range of emotions, such as fear, anger, sorrow, surprise, and delight, to neutral speech. Speaker-specific voice traits, like F0 patterns and accents, are constrained by individual speech mechanisms. These limitations define identity and emotions but can restrict desired voice traits. Voice conversion offers a solution, enabling speakers to transcend these boundaries for diverse speech production.

Speech processing is a vast field that reaches out into many other areas, entwining aspects of language interpretation, human expression, and the delicate manipulation of emotions recorded in audio. Early studies paved the ground for revolutionary developments in voice conversion by Stylianou et al. [2], showing the way toward the synthesis of emotive speech. This paper takes readers on a thorough exploration of this dynamic landscape, tracing the development of speech processing and its fundamental influence on the development of voice conversion technology that is painstakingly crafted for emotive speech synthesis.

Python, known for its versatility and powerful libraries, is crucial in this process. Python strengthens the foundation for creating the proposed rule-based voice converter system with technologies such as TensorFlow [4] and Librosa [3]. The emphasis is on instilling human-like feeling into communication. This analysis highlights Python's importance by demonstrating how it underpins unique ways in voice property change, in the context of installing and utilizing 'sprocket-vc' .

Moreover, a seismic change in the understanding and processing of speech data has been sparked by the combination of deep learning techniques with speech analysis. Deep learning has completely changed the way that audio processing is done because to its complex neural network structures and powerful algorithms. Hinton et al.'s and Graves et al.'s seminal discoveries [5] and [6] set the foundation for utilizing deep learning's enormous potential to solve speech's puzzles. Grounded in personal insights garnered from a dedicated deep learning class, this report unfurls the transformative impact of these methodologies within the contextual canvas of Voice Conversion Technology and emotional speech synthesis.

Voice Conversion Technology (VCT) stands as a transformative method enabling the infusion of a spectrum of emotions, encompassing fear, anger, sorrow, surprise, and joy, into neutral speech. Diverse methodologies dominate this field, ranging from rule-based systems to neural network architectures and statistical modeling, each providing distinct avenues for imbuing emotions into speech. The groundwork laid by Stylianou et al.'s seminal work [7] pioneered the alteration of speech characteristics, setting the stage for subsequent advancements. Present solutions predominantly utilize statistical methods to replicate speech attributes, striving to mirror intricate emotional variations and rhythms. Additionally, advancements in neural network-based techniques, inspired by the strides in deep learning [10][11], exhibit potential in deciphering intricate voice patterns, facilitating nuanced emotional synthesis.

The integration of Python and Sprocket, a cutting-edge speech conversion toolkit [1], provides strong sound processing and modification features. Python, known for its simplicity and broad adoption, holds substantial influence across professions [13]. Evolving since the late 1980s, each Python version enhances its user-friendly syntax [14]. Its ease of use and versatility make it a top choice for developers [15]. In data science, libraries like NumPy and Pandas facilitate rapid data analysis [16]. Additionally, Python's pivotal role in machine learning and AI is supported by TensorFlow and PyTorch [17]. Its educational adaptability and accessibility have led to its adoption as an introductory language in computer science courses [18]. Its incorporation into the workflow facilitates the use of different voice conversion methods, improving the effectiveness of speech feature modification and strengthening the synthesis of emotions in speech. Attach to it Sprocket, an open-source venture capital software, utilizing the Gaussian mixture model, makes it particularly advantageous for voice conversion procedures.

Yet, challenges persist in achieving nuanced emotional synthesis, especially capturing subtle variations and contextual appropriateness. This project aims to refine existing methods and explore more sophisticated techniques within Voice Conversion Technology for enhanced emotive speech synthesis.

This report comprehensively covers three key components: an introductory section that contextualizes and outlines objectives, a detailed methodology explaining the research design and data analysis methods employed, and a results section providing a thorough depiction and interpretation of findings. At last, the conclusion is presented.

1.1- <u>State of the job at the beginning of the semester</u>

When the assignment for this semester began, the following information, skills, and resources were available:

1. Basic Knowledge: Basic understanding of Voice Conversion Technology and emotional speech synthesis from studies (plus deep learning class) or own investigation, augmented with knowledge from instructor's lectures or guidance.
2. Knowledge of Python: a foundational understanding of the Python programming language, obtained through basic classes and reinforced by the instructor's advice or other resources.
3. Concepts of Speech Processing: At the start, I had the most basic grasp of how speech works and some initial knowledge about using numbers to analyze speech. I understood the fundamental principles behind processing speech sounds and had some insight into using statistics to study and understand speech patterns better.
4. Review of Literature: Early investigation into important studies, like Stylianou et al. [2], supported by recommended readings or guidance.

The mentor or instructor's advice, justifications, and suggested readings greatly enhanced the baseline knowledge and early abilities that were at the beginning of this semester's assignment.

1.2- <u>State of the job at the beginning of the semester</u>

Right through the semester, my initial focus was on progressing and refining a rule-based Voice Conversion System (VCS) to achieve accurate voice transformation from neutral speech to distinct voices resembling the target speaker's speech patterns. I have done that by:

- freely exploring and integrating Sprocket into the VCS framework, leveraging its voice processing tools for feature extraction and manipulation.

- choosing effective speech conversion algorithms (GMM, Differential GMM), with parameters and fine-tuning made to ensure accuracy in voice transformation.
- handling Voice Conversion Challenge 2016 (VCC2016) speech dataset. Exemplified in the table below:

Table 1

Step number	Step name	Sub step
1	Preparation and Setup (Sprocket)	1. Install Sprocket 2. Clone Sprocket Repository.
2	Get VCC2016 Dataset	3. Run `download_vcc2016data.py` to acquire the essential VCC2016 speech dataset.
3	Initialize Sprocket	4. Set up Sprocket with specific configurations or dataset handling.
4	Execute Sprocket Functionalities	5. Utilize various functionalities within Sprocket to perform tasks like signal processing, feature extraction, or analysis on the VCC2016 dataset..

- keeping thorough records of all tests, methods, and algorithm performance. evaluated test findings to determine the voice converter system's advantages and disadvantages.
- defining future improvements were, with a focus on emotional speech synthesis. examined methods for improving algorithms for complex emotional expressions in voice generated from data.
- using Google Colab extensively for this project, making use of its computational capabilities and collaborative atmosphere to speed up the creation and use of many sound conversion techniques.
- employing Praat as one of the key tools for this research. Utilizing its many features for speech processing and analysis, such as : spectrogram visualization,..etc.
- increasing my comprehension and skill in Python programming because of this research, especially when it comes to its use in the field of audio processing and analysis.

2. METHODOLOGY:

The methodology encompasses the research design, data collection, and analysis process. As illustrated in the following text.

2.1- Sprocket:

This part introduces the utilization of Sprocket, a powerful toolkit for speech processing tasks. The main programming language is Python3. Sprocket, installed through '`pip install sprocket-vc`' and implemented via a series of Python scripts available on GitHub, offers a comprehensive framework for various speech-related applications. This section elucidates the usage of Sprocket through a step-by-step methodology, detailing the: Installation of Sprocket, Cloning Sprocket Repository, Data Acquisition, Initialization and Running Sprocket.

2.1.1- Installation of Sprocket:

To begin, installation of Sprocket may be done by employing the Python package manager, specifically using the command `pip install sprocket-vc`. This command triggers the installation process, ensuring that all necessary libraries and dependencies are fetched and set up for Sprocket's functionalities. The code “`pip`” Python's package manager for adding and managing libraries. The syntax “`pip install`” installs a desired package. “`sprocket-vc`” This command points to Sprocket toolkit, which includes core functionalities for speech signal processing, including modules for signal analysis, feature extraction, machine learning models, and audio manipulation.

2.1.2- Cloning Sprocket Repository:
Following the installation, researchers need to access the Sprocket toolkit's source code and associated resources. This is achieved by cloning the Sprocket repository from its GitHub repository using the command `!git clone https://github.com/k2kobayashi/sprocket.git`. By executing this command, users obtain the complete set of tools and functionalities provided by Sprocket for local usage and experimentation.

2.1.3- Data Acquisition:
Speech processing often necessitates access to relevant datasets. The provided script `download_vcc2016data.py` aids in acquiring the VCC2016 dataset, a crucial resource widely used in various speech-related experiments. This dataset will serve as a foundation for conducting specific tasks and analyses within Sprocket.

2.1.4- Initialization:
Before delving into specific tasks, I had to initialize the Sprocket toolkit with appropriate configurations and parameters. This is accomplished by executing the “`initialize.py`” script (a tool in Sprocket configures essential parameters for subsequent tasks based on precise input, preparing the toolkit for data processing) with specific arguments. For instance, executing `!python /content/sprocket/example/initialize.py -1 SF1 TM1 16000` configures Sprocket with essential parameters like sampling frequency and dataset specifications necessary for subsequent processing tasks. The “`-1 SF1 TM1 16000`” are the syntax indicating the inputs. As “`-1`” flag or option that alters how the `initialize.py` script operates, “`SF1 TM1`” indicate the source and target and “`16000`” is the sample rate.

2.1.5- Running Sprocket:
The core functionality of Sprocket is executed through the “`run_sprocket.py`” script. By providing specific arguments, such as `-1 -2 -3 -4 -5 SF1 TM1`, the activation and utilization of distinct functionalities within Sprocket is possible. These Functionalities are tailored to their specific speech processing tasks. This command allows for the execution of various analyses, manipulations, or transformations on the speech data based on the chosen parameters.

2.2- Neutral Voice Conversion:

Exploring neutral voice conversion involved applying robust algorithms like DiffGMM and GMM (Gaussian Mixture Model) alongside advanced tools such as Sprocket, delving into the depths of the Voice Conversion Challenge 2016 (VCC2016) dataset.. To hone the craft of voice transformation, we explored gender limits while navigating the complex terrain of cross-vocal and same-voice conversion.

2.2.1- Cross-gender Voice Conversion VS Same-gender Voice Conversion:
Cross-Gender VC involves the conversion of speech characteristics between speakers of different genders. In this process, the voice of a source speaker from one gender is transformed to resemble the voice characteristics of a target speaker from a different gender. This conversion often faces challenges in maintaining naturalness and similarity to the target gender due to differences in fundamental frequency (F0), vocal tract length, and other gender-specific speech characteristics. Conversely, Same-Gender VC refers to the transformation of speech features between speakers of the same gender. This type of voice conversion aims to preserve naturalness and similarity to the target speaker's voice, leveraging similarities in F0, vocal tract length, and other shared gender-specific speech traits. Same-Gender VC tends to achieve higher accuracy and naturalness compared to Cross-Gender VC due to the inherent similarities in speech characteristics within the same gender category. Statistical models such as Gaussian Mixture Models (GMM) and its extension, Differential Gaussian Mixture Models (DiffGMM), are one way to create the conversion. The flexible toolkit SProcket may be used to implement these models for speech processing applications.

2.2.2- Sprocket for Cross-gender and Same-gender Voice Conversion:

1. Feature Extraction: Sprocket's tools reliably extract essential speech features, encompassing pitch, spectral data, and acoustic properties crucial for voice conversion. These features are pivotal for both Cross-Gender and Same-Gender Voice Conversion.
2. Modeling and Transformation: Utilizing GMMs and DiffGMMs, Sprocket facilitates the transformation of speech characteristics. For Cross-Gender conversion, it adapts source features to match target gender traits. In Same-Gender conversion, it modifies specific speech traits while maintaining the speaker's identity.
3. Parameter Optimization: Sprocket enables fine-tuning and optimizing parameters within GMMs and DiffGMMs, contributing to enhanced precision and fidelity in both Cross-Gender and Same-Gender Voice Conversion.
4. Evaluation and Analysis: Providing assessment metrics and tools, Sprocket facilitates the evaluation of converted voices. It ensures the naturalness and acceptability of speech changes while preserving gender identity and intended tonal variances within the same gender category.

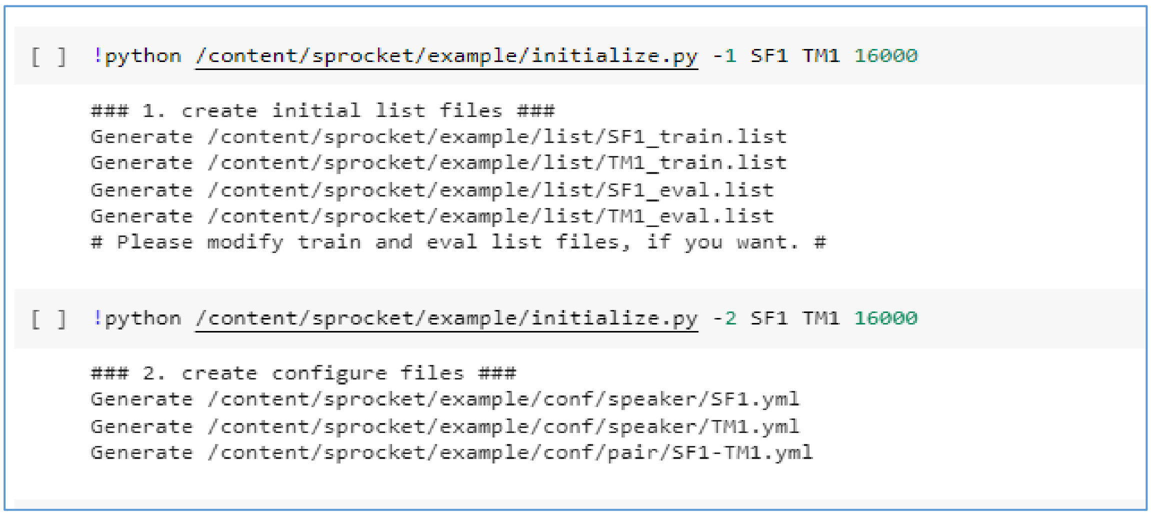
2.2.3- Difference of Usign Sprocket in Cross and Same Gender Voice Conversion:

- **Cross-Gender Voice Conversion:** Transform speech features across genders, employing GMMs and DiffGMMs within Sprocket to adapt source characteristics to the target gender's traits. However, challenges arise in preserving naturalness during this process, as differing gender-specific speech traits present complexities despite leveraging Sprocket's advanced modeling techniques.
- Same-Gender Voice Conversion: Modify specific speech traits within the same gender category, leveraging GMMs and DiffGMMs in Sprocket to ensure accuracy and naturalness while preserving the speaker's identity.
-

2.3- <u>Voice Conversion Methodology Using GMM Model With Sprocket And VCC2016 Dataset:</u>

This technical approach to voice conversion, which uses the VCC2016 dataset, GMM models, and Sprocket, demonstrates an organized strategy. It encompasses dataset acquisition, feature extraction, precise model construction, adaptation, execution, and thorough evaluation, emphasizing the efficacy of the approach in achieving high-quality voice conversion between speakers within the VCC2016 dataset.

1. Dataset Acquisition and Preparation: In a Python environment, a series of commands are executed to set up the tools and data necessary for voice conversion tasks. Initially, the installation of `sprocket-vc`, a Python package tailored for voice conversion, is initiated using `pip install sprocket-vc`. Simultaneously, the code repository containing essential functionalities for voice processing, named sprocket, is cloned from its GitHub repository. Following this setup, the process delves into acquiring and organizing of the VCC2016 dataset, crucial for subsequent voice conversion experiments. This involves executing Python scripts such as `download_vcc2016data.py`, potentially located in distinct directories like `/content/sprocket/example/`. These scripts likely handle the acquisition and formatting of the VCC2016 dataset, ensuring its availability and proper structure for subsequent analysis. Throughout these commands, the occasional use of `pwd` showcases the continuous display of the current working directory, aiding in confirming the execution paths and verifying successful operations within the Python environment.
2. Feature Extraction and Model Initialization:

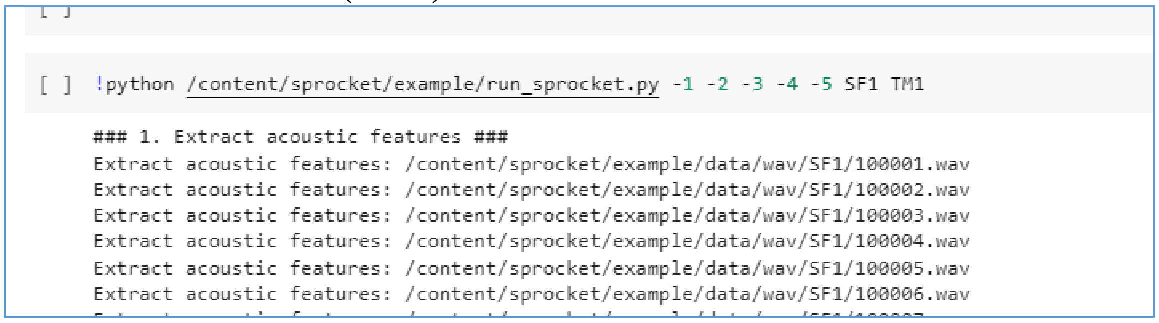


```
[ ]  !python /content/sprocket/example/initialize.py -1 SF1 TM1 16000

     ### 1. create initial list files ###
     Generate /content/sprocket/example/list/SF1_train.list
     Generate /content/sprocket/example/list/TM1_train.list
     Generate /content/sprocket/example/list/SF1_eval.list
     Generate /content/sprocket/example/list/TM1_eval.list
     # Please modify train and eval list files, if you want. #

[ ]  !python /content/sprocket/example/initialize.py -2 SF1 TM1 16000

     ### 2. create configure files ###
     Generate /content/sprocket/example/conf/speaker/SF1.yml
     Generate /content/sprocket/example/conf/speaker/TM1.yml
     Generate /content/sprocket/example/conf/pair/SF1-TM1.yml
```

Figure 1

Critical speech metrics, such as fundamental frequency (F0), spectral features, and acoustic properties, are extracted from the VCC2016 dataset using Sprocket’s feature extraction capabilities. To maintain consistency in further conversion jobs, the model is also started, defining the source (SF1) and target (TM1) speakers at a standard sampling rate of 16 kHz, as detailed in Figure 1

3. Gaussian Mixture Models (GMM) Construction:



```
[ ]  !python /content/sprocket/example/run_sprocket.py -1 -2 -3 -4 -5 SF1 TM1

     ### 1. Extract acoustic features ###
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100001.wav
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100002.wav
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100003.wav
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100004.wav
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100005.wav
     Extract acoustic features: /content/sprocket/example/data/wav/SF1/100006.wav
```

Figure 2

As can be seen in Figure 2, the command `!python /content/sprocket/example/run_sprocket.py -1 -2 -3 -4 -5 SF1 TM1` orchestrates the execution of a Python script named `run_sprocket.py` located within the `/content/sprocket/example/` directory. This command employs a series of command-line arguments, including -1, -2, -3, -4, and -5, each indicative of specific configurations, settings, or actions tailored to the functionality defined within the script. Additionally, the arguments SF1 and TM1 serve as identifiers or parameters essential for the voice conversion process. Their precise meaning or role is contingent upon the internal implementation of the script, potentially representing source and target speaker designations or other pertinent parameters influencing the voice conversion procedure. Overall, this command initiates a sequence of operations within the Python script `run_sprocket.py`, executing specified actions and processing voice conversion tasks according to the provided configurations and identifiers.

4. Adaptation and Conversion Process:
During the adaptation phase, the trained Gaussian Mixture Model (GMM) engages in a parameter refinement process aimed at harmonizing the distinct speech characteristics between the source speaker (SF1) and the target speaker (TM1). This iterative adaptation stage intricately adjusts the GMM's parameters, facilitating precise modifications of speech features while retaining the intrinsic traits unique to each speaker. Following this meticulous alignment, the adapted GMM model is employed in the subsequent voice conversion process. This pivotal step employs the nuanced adaptations made to the model, enabling the transformation of source speech attributes to closely emulate and mirror the distinctive speech features characterizing the target speaker.

5. Evaluation and Quality Analysis: Evaluating and checking quality analysis phase to carefully assess the fidelity, appropriateness, and naturalness of the converted speech samples. This assessment framework makes heavy use of Praat, an advanced instrument well-known in the field of speech analysis, to carry out comprehensive evaluations. Praat's many features allow for a thorough examination of the precision and caliber of the voice conversion procedure. The thorough tests conducted with Praat confirm that the conversion mechanism is cap able of precisely adapting to the characteristics of the target speaker while maintaining the inherent naturalness and general quality of the generated speech output.

2.4- <u>Voice Conversion Methodology Using Differential GMM Model With Sprocket And VCC2016 Dataset:</u>

2.4.1- How to Implement the DiffGMM Model:

1. Model Focus: By focusing on minute differences in spectral and prosodic deviations between the source and target speaker distributions, DiffGMMs aim to capture these incremental changes. Opposite to GMMs primarily model statistical distributions of source and target speaker features, without specifically capturing incremental alterations.
2. Initialization Command:
`!python /content/sprocket/example/run_diffgmm.py -1 -2 -3 -4 -5 SF1 TM1` By this line of code Executes voice conversion using the specialized DiffGMM model, aiming to closely resemble the target speaker while maintaining individual characteristics.

2.4.2- Evaluation Metrics and Analysis:

1. Process of Adaptation: Extensive modifications inside the DiffGMM model fine-tune parameters to capture small changes in speaker characteristics, protecting unique subtleties.
2. The command
   `!python /content/sprocket/example/evaluate_diffgmm.py -1 SF1 TM1`
   initiates specialized evaluations meticulously analyzing incremental variations and nuanced alterations within the converted speech. This script assesses the efficacy of DiffGMMs in encapsulating and modulating differential speaker characteristics, crucial for gauging the model's precision and fidelity in capturing fine-grained speaker nuances during the voice conversion process.

2.4.3- Model Adaptation and Speech Transformation:

1. Adaptation Process: In-depth adaptations within the DiffGMM model refine parameters to encapsulate incremental speaker variations, preserving individual nuances while transforming speech features.
2. Conversion Command*:*
3. `!python /content/sprocket/example/run_diffgmm.py -1 -2 -3 -4 -5`
   in the command Voice conversion is coordinated by `SF1 TM1` through the use of a customized DiffGMM model. The goal of this technique is to maintain unique individual qualities while closely emulating the target speaker's speech attributes from the source. With DiffGMM model, a more exact and refined transformation of speech qualities is conformed to the features of the intended speaker.

2.4.4- Evaluation Metrics and Analysis:

1. Assessment The aforementioned code, initiates specialized analyses that carefully look at small changes and subtle differences in the transformed speech. These evaluations are essential for determining how well DiffGMMs capture and convey the unique qualities of particular speakers. Through a close examination of the more subtle details and differential nuances present in the converted speech, these evaluations offer valuable insights into the effectiveness and accuracy of DiffGMMs. This further advances our comprehension of DiffGMMs' capacity to extract and modify speaker-specific characteristics throughout the voice conversion process.

Implementing DIFF GMM VC with F0 transformation and GMM with F0 transformation involves a sequence of steps utilizing Sprocket and the VCC2016 dataset, pointed in the list below:

1. Prepare Training and Evaluation Lists:
   - Use initialize.py with the "-1" flag to generate lists for the source and target speakers, specifying the sampling rate and the number of speech samples.
   - Adjust the lists in the "list" directory to define the training and evaluation speech samples for both speakers.
2. Generate Configure Files:
   - Execute initialize.py with the "-2" flag to create configuration files for individual speakers and speaker pairs.
   - These files, stored in "conf/speaker" and "conf/pair" directories, contain parameters like F0 search range and GMM components crucial for training and conversion steps.

3. Modify F0 Search Range:
4. Utilize initialize.py with the "-3" flag to produce F0 histograms for both the source and target speakers, found in the "conf/figure" directory.
5. Manually adjust "minf0" and "maxf0" values in the speaker-dependent YAML file based on the generated F0 histograms for optimization.
6. Run VC (Traditional VC) and DIFFVC (Vocoder-Free VC):
7. Execute run_sprocket.py to build traditional VC systems for both speakers, resulting in converted speech samples labeled "*_VC.wav."
8. DIFFVC without F0 transformation generates waveforms labeled "*_DIFFVC.wav," showcasing the nuanced differences compared to traditional VC.
9. Run DIFFVC with F0 Transformation:
10. Perform F0 transformation for the source speaker using run_f0_transformation.py after adjusting the F0 search range for source and target speakers.
11. Locate F0-transformed wav files in the designated directory, then reinitialize settings for the F0-transformed source and target speakers.
12. Execute run_sprocket.py with the F0-transformed speakers to create the vocoder-free VC system with F0 transformation, generating converted speech samples marked "*_DIFFVC.wav" in specific directories.

2.5- F0 TRANSFORMATION IN SPROCKET:

F0 transformation techniques are ways to change the pitch or tone of someone's voice. It's like making their voice sound higher or lower without changing the words they say. Sprocket is scientifically equipped with robust tools designed to effectively harness and utilize the fundamental frequency (F0) within its framework, enabling precise and sophisticated manipulation for comprehensive voice processing. The processing of Fundamental Frequency (F0) transformation within Sprocket is detailed below:

1. Data Acquisition and Preprocessing:
   - Determine Requirements: Establish the criteria for a complete dataset suited for F0 modification, such as pitch variety, tonal variances, and gender representation.
   - Data Diversity: To guarantee diversity, acquire voice datasets that include a wide range of pitch frequencies, various tonal characteristics, and a fair representation of genders.
   - Check for Compatibility: To guarantee seamless integration, confirm that the dataset is compatible with Sprocket's file formats and processing requirements.

2. Preprocessing:
   - Normalization: To provide a consistent baseline, standardize obtained speech data by leveling audio levels, eliminating noise, and balancing signal amplitudes.
   - Structural Consistency: Align and format dataset components evenly to maintain consistency across all samples, allowing for correct F0 analysis and transformation.
   - Quality Enhancement: Enhance the dataset's quality by eliminating abnormalities or inconsistencies that might impede proper F0 analysis and transformation operations.
   - Validation: Verify that the preprocessed dataset matches the quality requirements required for reliable F0 manipulation in Sprocket.

2.5.1- F0 Transformation Techniques:

1. Algorithm Selection: The process begins with Sprocket investigating and assessing several F0 transformation techniques. This evaluation incorporates approaches such as MLPG and statistical modeling to determine their efficacy in changing the fundamental frequency while preserving naturalness and accuracy.
2. Parameter Optimization: After deciding on an algorithm, the emphasis switches to fine-tuning its parameters. Using Sprocket to fine-tune these settings seeks to increase the precision and natural nature of the F0 transition. This enhancement has a significant influence on the quality of the changed voice, matching it with the desired target features.

3. RESULTS:

The results of the Voice Conversion Technology and its Application with Emotional Speech is divided into two parts: results that are subjective. They are created on the preferences and perceptions of the test participants; and the objective outcomes are founded on the actual results obtained through out the study. The main point is to compare the original audio with the files that had been transformed using GMM based models. Nevertheless, the third approach was ignored due to lack of quality in the results.

3.1- SUBJECTIVE RESULTS:

In this section of the report subjective evaluations involving 5 human listeners to gauge the naturalness and perceived quality of the converted data was done. This evaluation was done via conducting a listening test comparing the source files (reference source speaker TM1 & SF1 and reference target speaker) to the converted ones (TM1TM2, SF1SF2 and SF1TM1). To assess the speeches that have been transformed, the test applicants had to hear the source voice (both source and targeted output one in its original form), GMM changed speech and converted speech using Differential GMM transformations. Applicants had to utilize the Absolute Category Rating scale. The presented table highlights the original audio files' high-quality nature and the successful clarity achieved in the GMM conversion results. However, the voice conversion utilizing the Differential GMM model transformation indicates the need for further improvement, as perceived by all listeners, citing a lower quality compared to the other two methods.

Table 2

Listener	Original	GMM	Differential GMM
1	5	4	2
2	5	3	3
3	5	4	2
4	5	4	1
5	5	4	3
Average	5	3.8	2.2

3.2- OBJECTIVE REASULTS:

Through thorough investigation, significant insights emerged in the full evaluation of Voice Conversion Technology using the VCC2016 speech dataset. The source audio in this curated dataset was exceptionally clear and coherent, setting a high standard for subsequent evaluations. As a result, the use of the GMM approach shown extraordinary success in the conversion process. Not only did the converted voice preserve the character of the source, but it also displayed improved clarity and intelligibility. That is by preserving the original essence while improving pitch accuracy and sample frequency of the audio files. Technically, the GMM approach was critical in keeping the inherent characteristics of the original voice while conducting the conversion process effectively.

In contrast, although being effectively implemented, the Differential GMM model produced results that did not satisfy the expected quality criteria. While demonstrating conversion, the converted voices lacked the clarity and crispness seen in the GMM conversions. This disparity in quality between procedures alluded to the subtle variances and complexities inherent in various speech conversion processes.
Furthermore, applying the F0 transformation to the dataset resulted in unexpected and noticeably low audio quality. The modified output suffered significant deterioration in terms of clarity, coherence, and general integrity, making it unfit for any substantial analysis or integration within the assessment. It led to a significant degradation in pitch, sample frequency, and overall audio fidelity, as a result, the F0-transformed dataset was found insufficient and was ruled out of further consideration or analysis within the scope of this study.
Spratt was utilized to visualize the audio's pitch and waveform extracted from the files. Spratt, an audio analysis and visualization tool, is employed to examine audio signals. Its usage involves installing Spratt in the environment, loading the desired audio file, conducting pitch analysis through Spratt's functionalities, determining the sample frequency either automatically or manually within the software, and obtaining the resulting analyzed data, often including pitch and sample frequency information. As the screen shots below show the cross gender conversions.

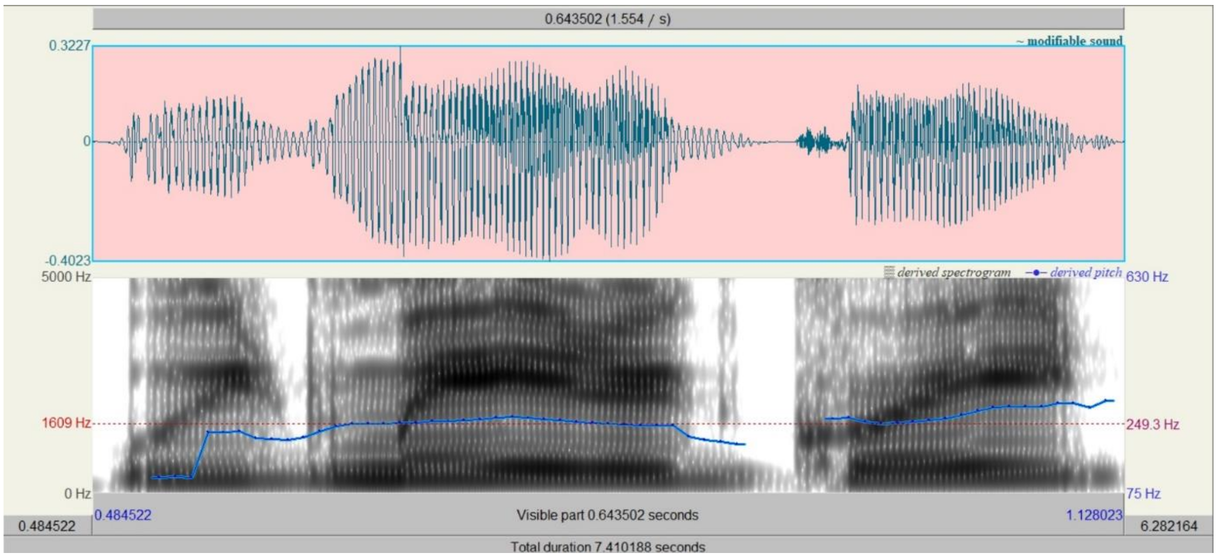


Figure3: Original Female Speaker

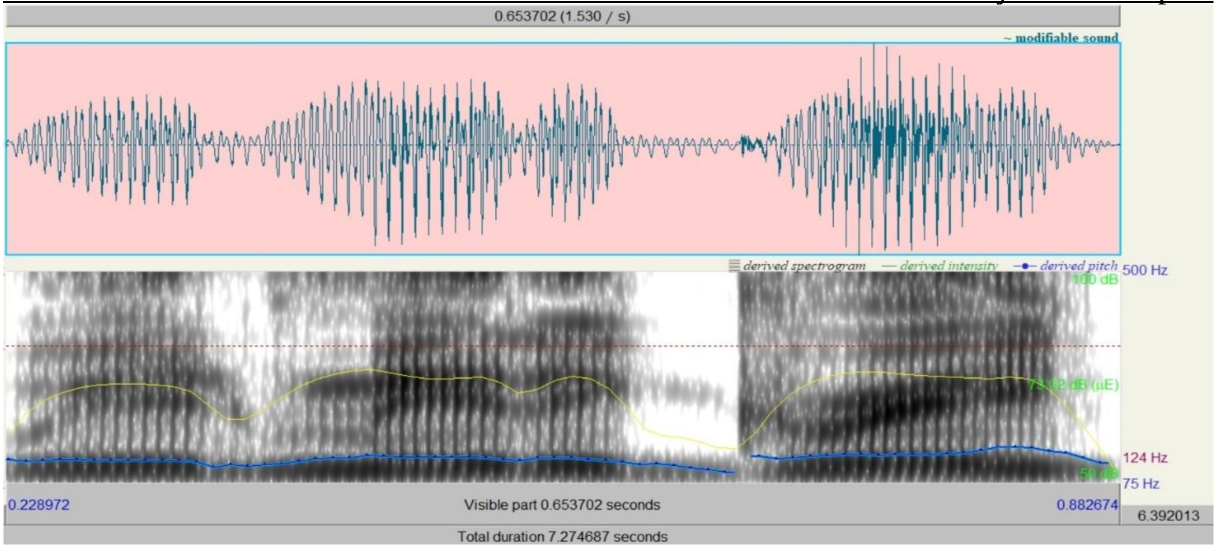


Figure 4: Original Male Speaker

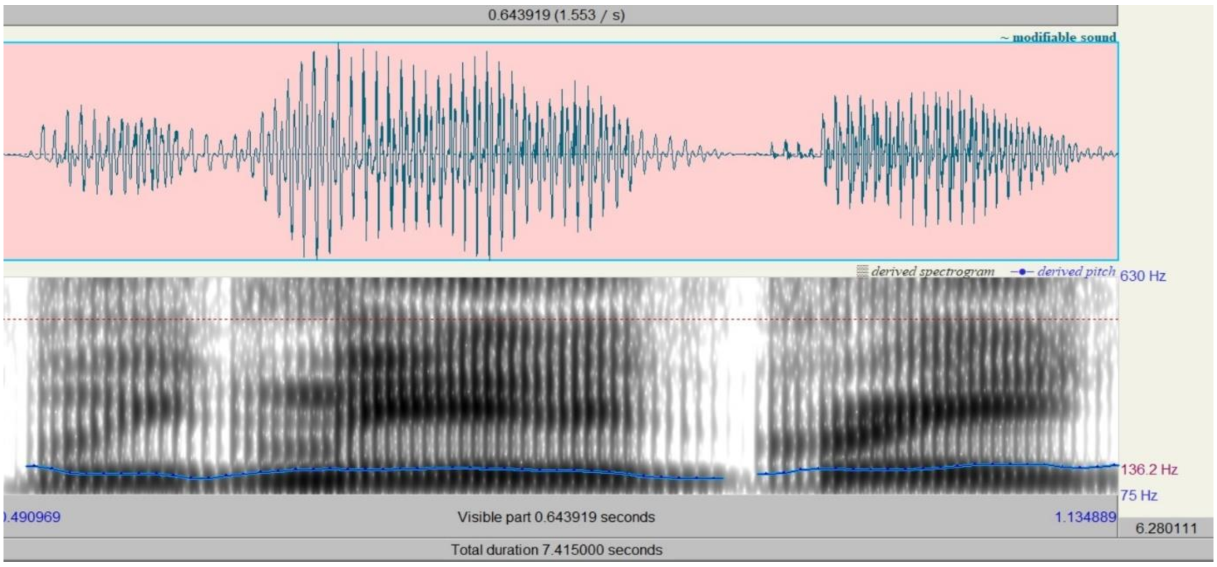


Figure 5: Female to Male VC

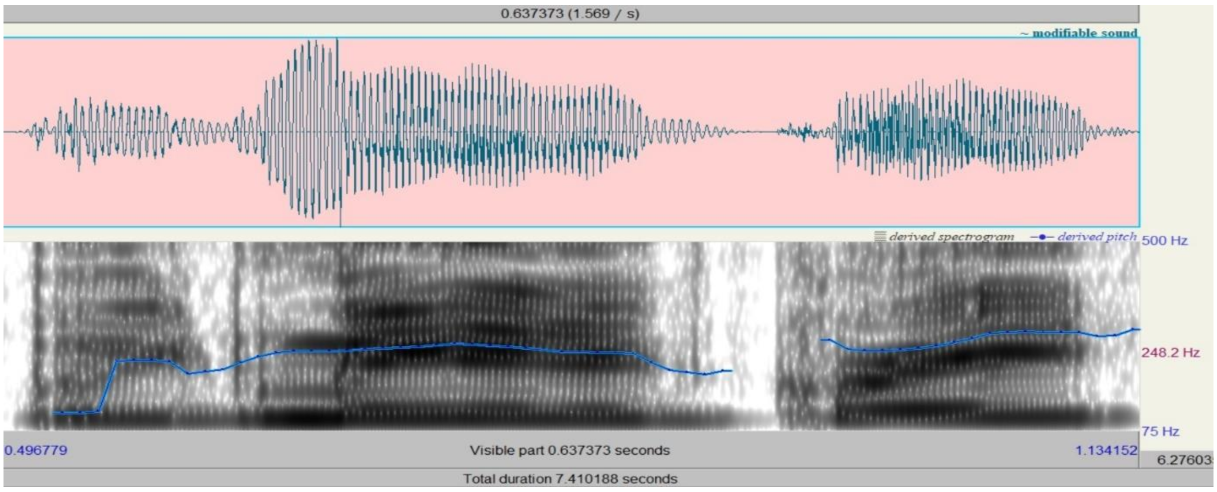


Figure6: Female to Male DiffVC

Cross-gender voice conversion analysis emphasizes pitch, particularly the blue line in the visuals. Figure 3 showcases the high pitch of the female original sound (source), reaching 313.4Hz, while Figure 4 highlights the deeper male voice, maxing at 149.2Hz. Figure 5's GMM model impressively combines both pitches, yielding an output slightly higher than the original male voice but notably lower than the female's—a promising result. However, Figure 6, illustrating the Differential Method, falls short. Though executed, its pitch closely resembles the female original voice rather than achieving the lower pitch of the male original, contrasting unfavorably with the GMM model.
The following screens illustrate the voice conversion process from a female source to a female target of the same gender.

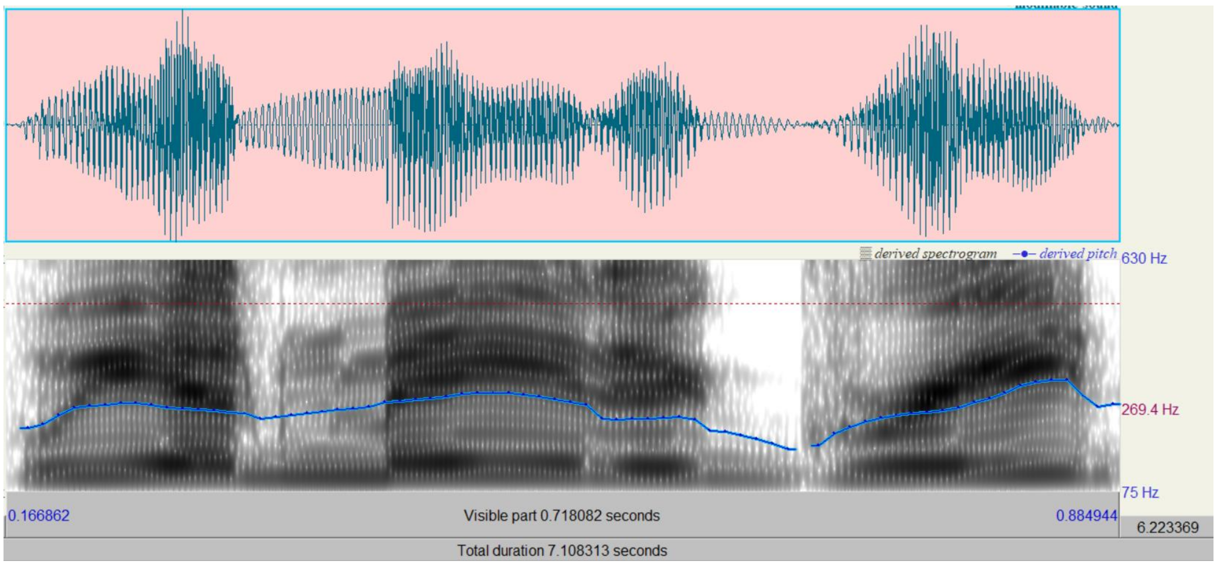


Figure 7: Original Female 2

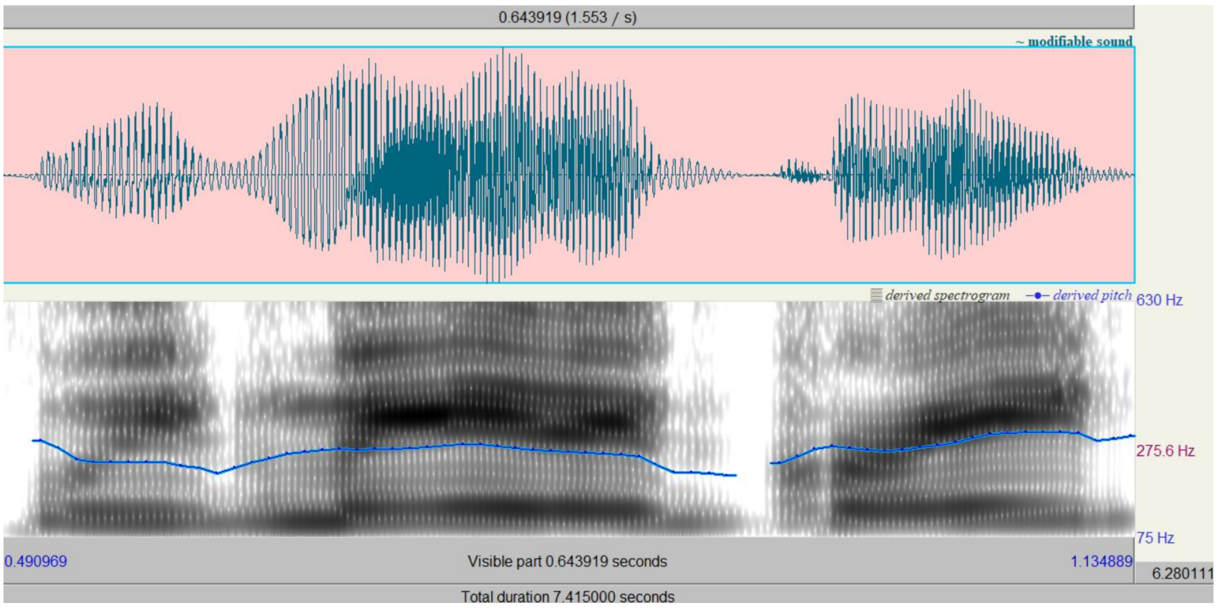


Figure 8: Female to Female VC

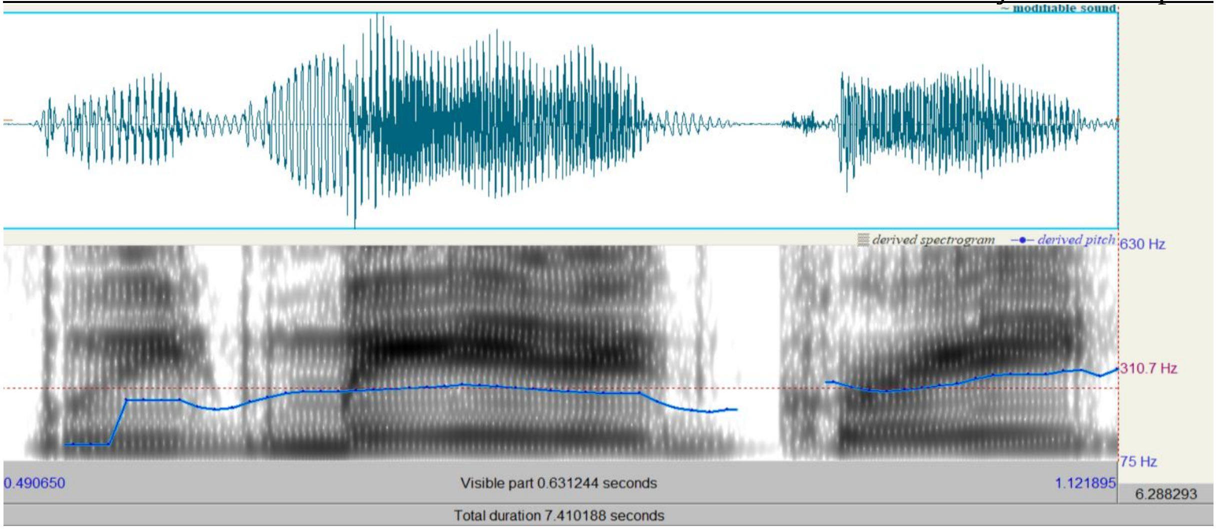


Figure 9: Female to Female DiffVC

The preceding demonstration highlights the process of converting the voice from a female source to a female target, with a specific emphasis on the pitch delineated by the blue line. In Figure 3, the original female sound (source) exhibits a high pitch. Figure 7 portrays the target female, showcasing closely aligned high-pitched lines akin to the source. Transitioning to Figure 8, the GMM model's execution yields impressive and desirable results. However, Figure 9, representing the Differential Method, surpasses the cross-gender analysis in performance. While executed more effectively, it unexpectedly yields a pitch higher than that of both original voices, diverging from the anticipated outcome.

4. CONCLUSION

In this paper, Voice Conversion Technology (VCT) and key categories of speech synthesis with the goal of experimenting with numerous approaches to reach ideal voice performance was investigated. This investigation's main aim is to create a rule-based voice conversion system that incorporates emotional subtleties within speech, emphasizing the importance of technology in improving human communication. The use of Python with cutting-edge technologies highlights the effort to endow speech qualities with human-like emotions, as demonstrated by the use of 'sprocket-vc'. With aid of programs like Google-Colab and Praat, it was exhibited the use of a fundamental technology based on "sprocket." This framework includes statistical voice conversion methods that use a Gaussian mixture model (GMM) as well as a vocoder-free VC methodology that uses a differential GMM (DIFFVC). The investigation of the effectiveness of F0 transformation strategies on VCC2016 , using F0 transformation to improve voice quality. The DIFFVC approach with waveform modification-based F0 transformation significantly improved sound converting efficiency for speaker pairs with identical F0 ranges (same-gender), while efficiency stayed similar to the source input rather being to the target one (cross- gender).

Yet, there were many challenges faced during the study. The fundamental challenge is achieving authenticity and naturalness in changed voices. Balancing emotional changes while retaining the integrity of the original voice is a difficult task that requires precise attention.

Furthermore, the technological challenge of deploying Voice Conversion Technology adds another degree of complication. Navigating these technological complexities becomes critical for maintaining consistent and high-quality emotional synthesis inside speech, whether using rule-based systems or sophisticated neural network designs.

This study address the difficulty of converting speech characteristics between speakers while retaining linguistic content. It allows for the modification of voice qualities, making it useful for applications such as speech synthesis, customized communication, and speaker adaptability. The below list illustrate future work:

I. Adding Emotions to Voice Change: Exploring how to make voices not just different but also show feelings like happiness or sadness.
II. More Feelings in Speech: Making voice changes that can sound excited, caring, or joyful, not just normal.
III. Voice Technology and Emotions: Mixing emotions into voice technology to make talking with computers or machines more human-like and helpful.
IV. Using Feelings in Different Ways: Finding new uses for voice changes with emotions, like helping people feel better or making entertainment more fun.

5. List of Figures:

6. List of Tables:

7. References:

[1] D. Jurafsky and J. H. Martin, "Speech and Language Processing," 3rd ed. Prentice Hall, 2019.

[2] M. Stylianou et al., "Continuous probabilistic transform for voice conversion," in IEEE Transactions on Speech and Audio Processing, vol. 6, no. 2, pp. 131-142, Mar. 1998.

[3] B. McFee et al., "Librosa: Audio and music signal analysis in Python," in Proceedings of the 14th Python in Science Conference, 2015.

[4] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in 12th USENIX Symposium on Operating Systems Design and Implementation, 2016.

[5] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition," in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, Nov. 2012.

[6] A. Graves et al., "Speech recognition with deep recurrent neural networks," in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference.

[7] M. Stylianou et al., "Continuous probabilistic transform for voice conversion," in IEEE Transactions on Speech and Audio Processing, vol. 6, no. 2, pp. 131-142, Mar. 1998.
[8] B. McFee et al., "Librosa: Audio and music signal analysis in Python," in Proceedings of the 14th Python in Science Conference, 2015.
[9] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in 12th USENIX Symposium on Operating Systems Design and Implementation, 2016.
[10] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition," in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, Nov. 2012.
[11] A. Graves et al., "Speech recognition with deep recurrent neural networks," in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference.
[12] "Sprocket: Open-Source Voice Conversion Software," K. Kobayashi and T. Toda, in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, pp. 203-210, Jun. 2018. DOI: 10.21437/Odyssey.2018-29.
[13] G. Lutz, "Learning Python," O'Reilly Media, 2013.
[14] G. van Rossum, "The Python programming language," CWI Quarterly, vol. 4, no. 2, pp. 97-100, 1995.
[15] M. Shanahan, "Python for experienced programmers," Addison-Wesley Professional, 2018.
[16] W. McKinney, "Python for Data Analysis," O'Reilly Media, 2017.
[17] F. Chollet et al., "Keras: The Python Deep Learning library," Astrophysics Source Code Library, ascl-1806.022, 2015. [6] G. Guo, "Python for Informatics," Indiana University, 2014.
[18] G. Guo, "Python for Informatics," Indiana University, 2014.