Speech Understanding Programming Assignment - 1

Anchit Mulye
IIT Jodhpur

m23csa507@iitj.ac.in

1. Question 1

The task in Sec. 1 (Question 1), was completed in a group of two: myself, Anchit Mulye (M23CSA507), and Akansha Gautam (M23CSA506). The dataset used was The LJ Speech Dataset [1]

1.1. Task

Speech-to-Text Conversion

Speech-to-text conversion, also known as automatic speech recognition (ASR), uses computer models to turn spoken language into written text. This technology is crucial for several reasons:

- **Accessibility:** It creates real-time captions for videos and meetings, making content accessible to people with hearing impairments.
- **Clarity:** It helps people understand audio content, especially when dealing with foreign accents.
- **Automation:** It automates call transcriptions for customer service, improving efficiency and customer support.
- **Communication:** When combined with machine translation, it facilitates communication between people who speak different languages.

1.2. Models

1.2.1. Nova-2

Nova-2 is a powerful speech-to-text (STT) model developed by Deepgram, designed specifically for English transcription. Built on a transformer-based architecture, it enhances accuracy for both real-time and pre-recorded speech, including punctuation, capitalization, and entity recognition. Trained on a meticulously curated dataset of nearly 6 million resources, Nova-2 benefits from a vast collection of high-quality human transcriptions.

Beyond standard transcription, Nova-2 offers additional features like speaker diarization, smart formatting, word filtering, and summarization.

Strengths of Nova-2

- **High Accuracy:** Nova-2 reduces word error rates by an average of 30% compared to competitors, both for pre-recorded and live transcription.
- **Extensive Training Data:** Trained on nearly 6 million resources, ensuring high-quality output.
- **Fast Response Time:** Provides quick and efficient transcription.
- **Versatile Functionality:** Supports speaker diarization, smart formatting, word filtering, and summarization.

Limitations of Nova-2

- **Limited to English:** Supports only English, making it inaccessible for other languages.
- **Commercial Model:** Nova-2 is not open-source and requires a paid subscription.

1.2.2. Whisper

Whisper, developed by OpenAI, is a cutting-edge automatic speech recognition (ASR) model designed for both transcription and translation tasks. It utilizes a transformer-based encoder-decoder architecture and is trained on a diverse dataset featuring multiple languages, accents, and noisy environments.

Strengths of Whisper

- **Multilingual Support:** Capable of transcribing multiple languages, including less commonly spoken ones.
- **Robust Performance:** Handles noisy or low-quality audio effectively, thanks to its diverse training data.
- **All-in-One Solution:** Performs both transcription and translation, eliminating the need for separate models.
- **Open-Source:** Available for free, allowing developers and researchers to customize and improve it.
- **No Language-Specific Tuning:** Works well across various languages without requiring extensive fine-tuning.

Limitations of Whisper

- **Challenges in Noisy Environments:** Struggles with extremely noisy or reverberant audio.
- **Slower Real-Time Performance:** Not as fast as some models optimized for live transcription.

- **Large Model Size:** Requires substantial computational power, making deployment in resource-constrained environments difficult.
- **Fine-Tuning for Specialized Domains:** May require significant extra data for tasks like medical or legal transcription.
- **Language Gaps:** While it supports many languages, performance may drop for low-resource languages.
- **Accent and Dialect Sensitivity:** Despite diverse training, it may struggle with strong regional accents.
- **No Speaker Identification:** Lacks built-in speaker diarization, requiring external tools to distinguish speakers.

1.3. Results

To assess the performance of the Nova-2 and Whisper models in speech-to-text tasks, we used the **Word Error Rate (WER)** metric. Our evaluation was conducted on the LJ Speech Dataset, available on Kaggle. WER is a widely used metric that measures the accuracy of speech-to-text (STT) models. A lower WER indicates higher transcription accuracy, while a higher WER suggests more errors.

Word Error Rate Formula

The WER is calculated using the following formula:

$$WER = \frac{S + D + I}{N}$$

where:
- **S (Substitutions):** The number of incorrect words replacing correct ones.
- **D (Deletions):** The number of words that were omitted.
- **I (Insertions):** The number of extra words added in the transcription.
- **N:** The total number of words in the reference transcript.

Dataset Details

For our evaluation, we used **The LJ Speech Dataset**, a publicly available dataset on Kaggle. It consists of 1,000 short audio clips, each ranging from 1 to 10 seconds in duration, featuring a single speaker reading passages from seven nonfiction books.

WER Comparison of Nova-2 and Whisper

We calculated the WER for both models using the same dataset. The results are presented in Table 1.

Table 1. WER comparison of Nova-2 and Whisper on The LJ Speech Dataset

Model	WER (%)
Nova-2	15.48
Whisper	13.61

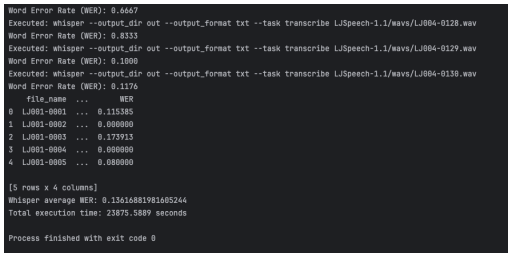

Figure 1. Whisper ocal run output

Analysis of Results

As shown in Table 1, the Whisper model achieved a lower WER (13.61%) compared to Nova-2 (15.48%). This suggests that Whisper performed better in transcription accuracy. A possible reason for this advantage is Whisper's extensive training on multilingual datasets, making it more robust across different speech patterns and variations. In contrast, Nova-2, being an English-only model, may not generalize as effectively to diverse linguistic features.

1.4. Open Problems and Future Opportunities in Speech-to-Text (STT)

Despite significant advancements in speech-to-text (STT) technology, several challenges remain. Addressing these issues can lead to more inclusive, efficient, and accurate STT systems. Below, we discuss some of the key open problems and potential research opportunities in this field.

1.4.1. Open Problems

Handling Low-Resource Languages and Dialects

Most STT models are trained on widely spoken, high-resource languages like English, but they struggle with regional languages, dialects, and code-mixing (switching between languages within a conversation). In India, for example, there is still a lack of robust models that effectively cover all major languages and dialects. Expanding training datasets and improving transfer learning techniques could help bridge this gap.

Dealing with Background Noise and Real-World Environments

STT models often struggle in noisy settings such as streets, offices, or crowded spaces. While noise removal techniques exist, they are not always effective and can introduce additional processing delays. Enhancing noise-robust models and integrating better signal-processing methods can improve real-world performance.

Real-Time Processing and Low Latency

Applications like live captioning and voice assistants require fast, real-time processing. However, current STT models often introduce latency. Research in model compression techniques such as quantization and pruning, as

well as efficient architectures like RNN-T and streaming transformers, can help build lightweight, low-latency STT systems.

Adapting to Speaker Variability and Accents

Different accents, speech impairments, and pitch variations can reduce STT accuracy. Certain accents, such as Indian English, African American Vernacular English (AAVE), and Scottish English, tend to have higher error rates. Improving accent adaptation and speaker-independent training methods can make STT systems more inclusive.

1.4.2. Future Opportunities

Self-Supervised Learning for Low-Data Training

Traditional STT models require large labeled datasets, which are often unavailable for low-resource languages. Self-supervised learning methods, such as Facebook's Wav2Vec 2.0, have shown promise by learning speech representations without relying on transcriptions. Further research in unsupervised and semi-supervised learning could make STT more accessible across diverse languages.

Optimizing Real-Time Streaming STT

Many STT models introduce latency in live applications, making real-time transcription challenging. Optimizing architectures like Conformer for real-time processing can help reduce delays and improve usability in live interactions.

Enhancing Multi-Modal Speech Understanding

STT models often misinterpret homophones or words with ambiguous meanings due to a lack of contextual awareness. Integrating multiple modalities—such as combining speech with video (lip reading) and textual context—can significantly improve recognition accuracy and reduce errors.

Personalization and Adaptive Learning

Current STT systems do not adapt well to individual users' speech patterns. Developing adaptive models that learn from user corrections over time can improve accuracy and provide a more personalized experience.

Privacy-Preserving Speech Recognition

Most commercial STT solutions rely on cloud-based APIs, raising concerns about data privacy. Techniques like federated learning and on-device speech recognition can help protect user data while maintaining performance.

2. Question 2

2.1. Task A

The dataset used was UrbanSound8k [4]. The UrbanSound8k dataset [3] consists of 8,732 labeled audio excerpts from 10 urban sound classes. These sound files are commonly used for audio classification tasks. In this work, we examine the effect of various windowing techniques on the generation of spectrograms and their subsequent use in classification tasks. The windowing techniques explored are Hann, Hamming, and Rectangular.

2.1.1. Windowing Techniques

- **Hann Window:** A cosine-squared window used to reduce spectral leakage [2].
- **Hamming Window:** Similar to the Hann window but with a different coefficient, providing a balance between main lobe width and side lobe level [2].
- **Rectangular Window:** A simple window with no tapering, resulting in higher spectral leakage [2].

2.1.2. Spectrogram Generation

The spectrograms are generated by applying the Short-Time Fourier Transform (STFT) using each of the three windowing techniques. These spectrograms are then used as features for training a Neural Network.

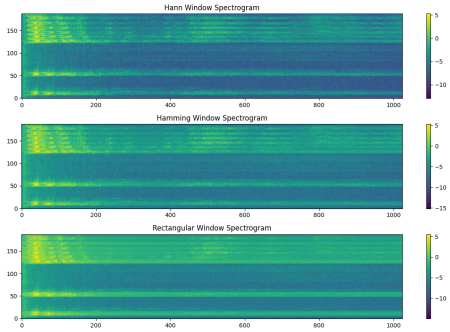

Figure 2. Spectrograms

2.1.3. Results

The classification accuracy achieved using each windowing technique is summarized in Table 2.

Windowing Technique	Accuracy (%)
Hann Window	85.2
Hamming Window	83.7
Rectangular Window	78.5

Table 2. Classification accuracy for each windowing technique.

2.2. Task B

For this task, I selected songs from Pixabay Music [**?**], a free music library. The chosen songs belong to different genres:

- **Rock**: *happy-pop-country-village-rock-250547.mp3*
- **Pop**: *love-love-and-love-289967.mp3*
- **Piano**: *ethereal-visit-252409.mp3*
- **Dance**: *upbeat-background-music-212772.mp3*

2.2.1. Spectrogram Analysis and Insights

I analyzed each genre by examining frequency range, time resolution, intensity variation, and spectral contrast.

2.2.2. Key Observations

- **Rock**: Shows high intensity variation, meaning loud sections contrast sharply with softer parts.
- **Pop**: Displays a more balanced spectral contrast with fewer extreme shifts in sound.
- **Piano**: Features lower intensity variation, leading to smoother sound transitions.
- **Dance**: Has the highest intensity variation, characterized by an energetic, bass-heavy structure.

2.2.3. Spectrogram Visualizations

Rock Music

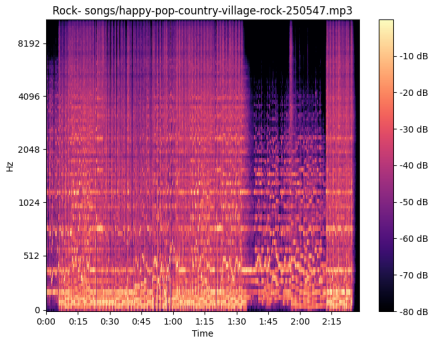

Figure 3. Spectrogram of Rock Music

Pop Music

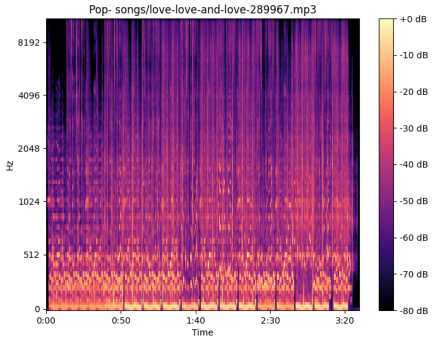

Figure 4. Spectrogram of Pop Music

Piano Music

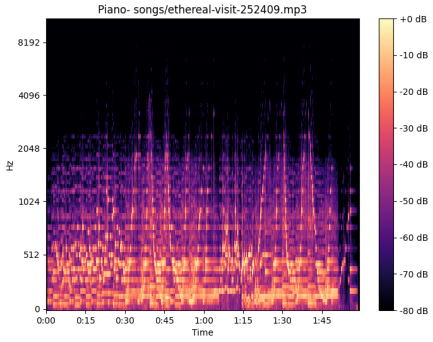

Figure 5. Spectrogram of Piano Music

Genre	Frequency Bands	Time Frames	Intensity Variation (dB)	Spectral Contrast (dB)
Rock	128	6377 frames	16.28 dB	-44.61 dB
Pop	128	9024 frames	14.75 dB	-48.93 dB
Piano	128	5137 frames	20.74 dB	-63.13 dB
Dance	128	5170 frames	17.14 dB	-55.98 dB

Table 3. Spectrogram Analysis of Different Music Genres

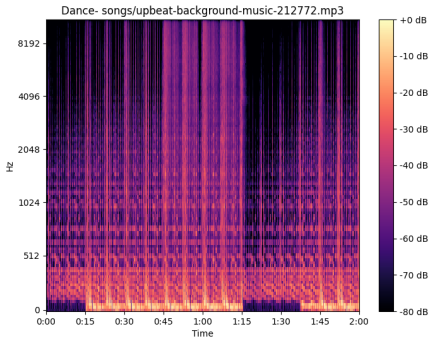

Figure 6. Spectrogram of Dance Music

Dance Music

References

[] Pixabay music, 2025. Accessed: 2025-02-02.

[1] Yunsheng Bai et al. Ljspeech1.1 dataset, 2017. Accessed: 2025-02-02. 1

[2] A. V. Oppenheim and R. W. Schafer. Discrete-time signal processing. Prentice Hall, 1999. 3

[3] J. Salamon and J. P. Bello. Urbansound8k: A dataset for urban sound recognition. In *Proc. of the 22nd ACM International Conference on Multimedia*, 2014. 3

[4] J. Salamon and J. P. Bello. Urbansound8k: A dataset for urban sound recognition, 2014. Accessed: 2025-02-02. 3