Audio Generation Using Bark Model

1. Introduction

This report covers the usage of the **Bark model** to convert a Hindi text prompt into audio. The code utilizes small models from the Bark framework, making it suitable for systems with limited computational resources. The output is saved as a `.wav` file, which can be played and evaluated.

2. Dataset Description

In this scenario, the dataset consists of a single text prompt written in Hindi:

गाँवों में लोग सामुदायिक जीवन जीते हैं, जहाँ पड़ोसी एक-दूसरे के सुख-दुख में भागीदार होते हैं। यहाँ परंपराएँ और रीति-रिवाज पीढ़ियों से चली आ रही हैं। लोग एक साथ त्योहार मनाते।

The prompt describes the communal lifestyle in Indian villages, emphasizing traditions, celebrations, and the interdependence of neighbors. The text serves as the input for the Bark model to generate an audio file.

3. Code Breakdown

1. **Importing and Environment Setup:**
    - The `os` module is used to set the environment variable `SUNO_USE_SMALL_MODELS=True`. This instructs the Bark framework to load smaller, lightweight models, which are faster and more resource-efficient but may provide slightly lower fidelity in audio synthesis.
2. **Preloading the Model:**
    - The function `preload_models()` is called to download and load all necessary pre-trained models. This prepares the framework for text-to-speech generation.
3. **Generating Audio:**
    - The core function `generate_audio(text_prompt)` takes the Hindi text as input and generates an audio array that represents the spoken version of the prompt.
4. **Saving Audio:**
    - The resulting audio array is saved as a `.wav` file using `scipy.io.wavfile.write()` with a sample rate (`SAMPLE_RATE`) and the generated audio data.
5. **Playing Audio:**
    - Finally, the code utilizes `IPython.display.Audio()` to play the generated audio in the notebook environment.

4. Training Details

No training was conducted in this scenario because the Bark model used is **pre-trained**. Bark models have been trained on large datasets consisting of multilingual text and

corresponding speech samples. These models are capable of converting text from various languages into speech.

Since the small model variant is used here, the training logs or further fine-tuning on this particular text are absent. However, the model's underlying training data includes a mix of text and speech from various languages, ensuring it can handle non-English input such as Hindi.

5. Performance Evaluation

A. Audio Quality

- **Clarity:** The generated audio output reflects the natural intonation and rhythm typically associated with Hindi speech. However, using the smaller model version may result in reduced nuance or detail in the pronunciation and prosody.
- **Pronunciation:** It is crucial to verify if the model accurately pronounces Hindi words and sentences. Bark models, being pre-trained on multilingual datasets, typically perform well in this regard, but there might be occasional inaccuracies, especially with uncommon or dialect-specific words.
- **Naturalness:** Smaller models tend to sacrifice some naturalness and fluidity in the speech to improve speed and resource efficiency. This trade-off might lead to a somewhat robotic or mechanical sound in the generated speech.

B. Efficiency

- **Inference Time:** One of the key advantages of using small models is faster inference. Loading the models and generating audio in a relatively short time is crucial when computational resources are limited. This script runs efficiently on most standard hardware setups.
- **Memory Usage:** By using the environment variable to load small models, the script consumes less memory compared to the full-size version. This allows it to be deployed in lower-resource environments.

C. File Size

The audio is saved in `.wav` format, an uncompressed audio format. The size of the resulting file (`bark_generation.wav`) will depend on both the duration of the generated audio and the sample rate (which is typically 24kHz or 48kHz). Higher sample rates will lead to better audio fidelity at the cost of larger file size.

D. Text Coverage

The Bark model is able to convert the entire text prompt into speech. Evaluating how well it captures the meaning and structure of the original Hindi text is vital. The output audio should convey the intended message in a manner that is both clear and accurate.

6. Potential Issues

- **Accent and Pronunciation:** Since Hindi may not be the dominant language in the pre-training dataset, there could be slight issues with regional accents or

pronunciation that differ from native speakers. An evaluation by a fluent Hindi speaker is recommended to assess this aspect.
- **Naturalness of Voice:** The small model sacrifices some natural intonations for efficiency. While the text-to-speech model works well, it may not sound as natural as a full-sized model or human-recorded voice.
- **Emotion and Expressiveness:** Given the description of communal life, a human speaker would likely use expressive tones to convey warmth and celebration. The model might not fully capture these emotional nuances.

7. Future Improvements

- **Switch to Full Models:** Running the same text through the full Bark models (by removing the `SUNO_USE_SMALL_MODELS=True` setting) will generate higher-quality audio, albeit at the cost of longer processing times and higher resource consumption.
- **Fine-Tuning for Hindi:** The audio quality, especially in terms of pronunciation and emotional expressiveness, can be improved through fine-tuning the model on more Hindi-specific text-to-speech datasets.
- **Multi-lingual Model Evaluation:** Since the model supports multiple languages, future experiments could explore cross-lingual scenarios or additional language prompts to assess versatility.

8. Conclusion

The provided code successfully converts a Hindi text prompt into audio using the Bark model with small model configurations. While this approach results in efficient and relatively fast text-to-audio generation, there are potential compromises in terms of naturalness, emotional tone, and pronunciation accuracy. Further experimentation with larger models or fine-tuning for specific languages like Hindi can enhance the performance.

9. Performance Metrics Summary

- **Inference Time:** Fast (due to small models)
- **Memory Usage:** Low
- **Audio Quality:** Decent, though somewhat robotic (due to model size)
- **Pronunciation Accuracy:** Acceptable but may need human verification
- **File Size:** Moderate (depending on sample rate)

This approach provides a balance between speed and quality, making it suitable for quick audio generation tasks where high-fidelity output is not the primary concern.