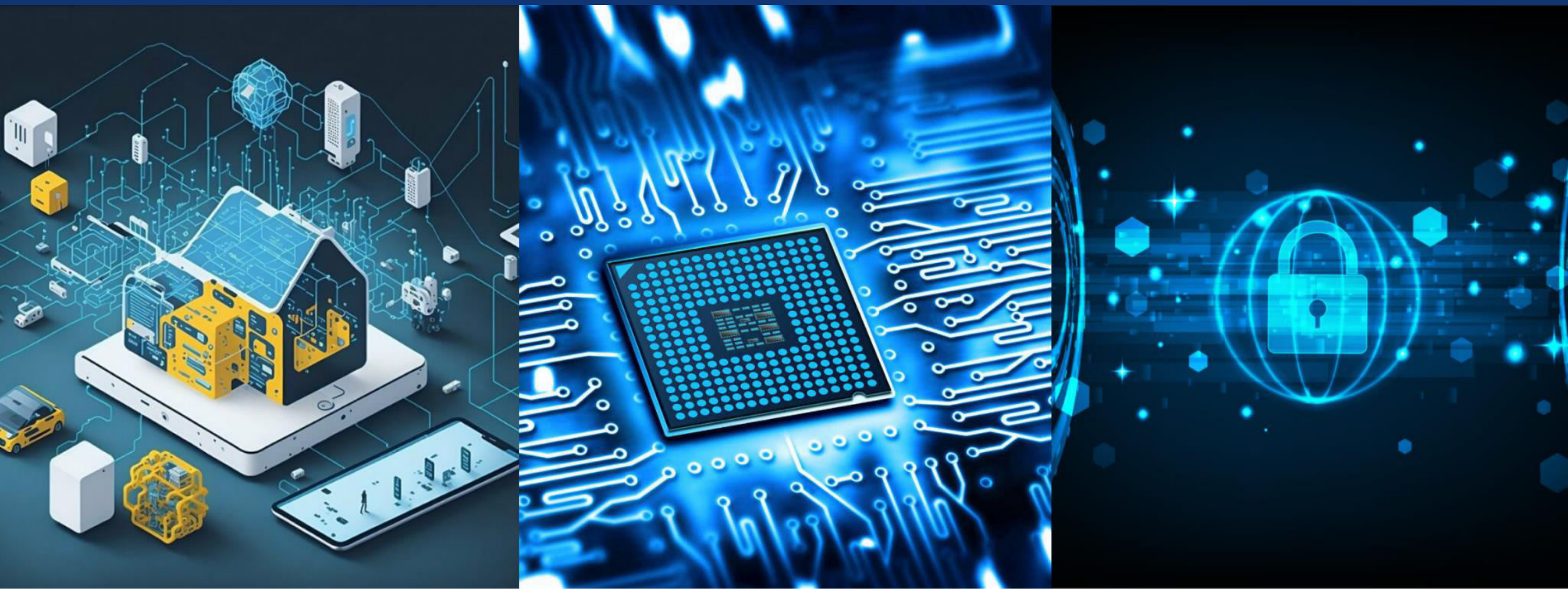
IJIRCCE

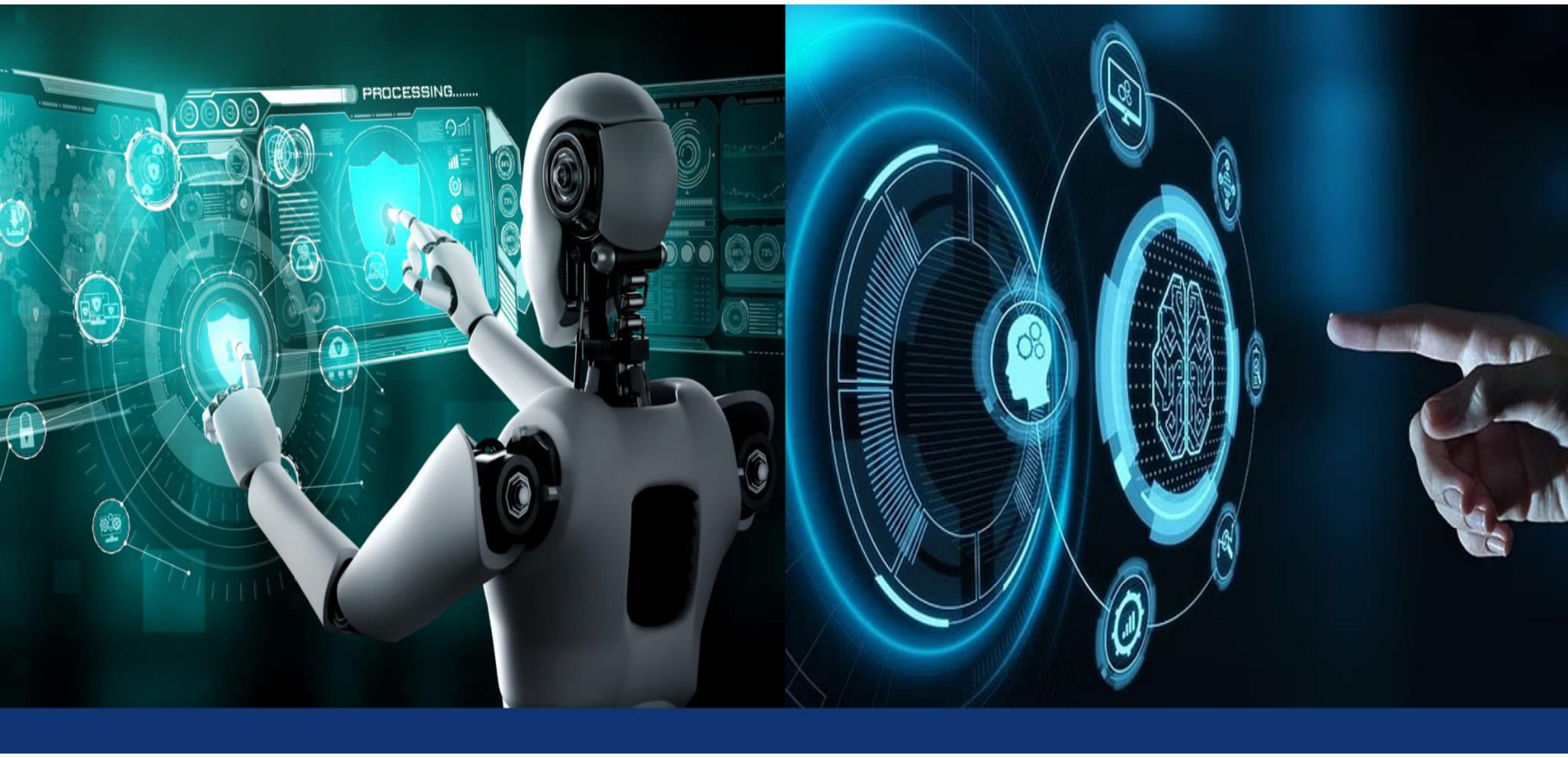
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.771

Volume 13, Issue 5, May 2025

www.ijircce.com ijircce@gmail.com +91-9940572462 +91 63819 07438

© 2025 IJIRCCE | Volume 13, Issue 5, May 2025| DOI:10.15680/IJIRCCE.2025.1305114

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

AI-Music Agent using Large Language Models

Prof. Waseem Khan[1], Bindu K[2], Gagan Kumar S V[3], Sanjana K R[4], Suhas S M[5]

Assistant Professor, Dept. of C.S., Bapuji Institute of Engineering and Technology, Davanagere, Karnataka, India[1]

UG Student, Dept. of C.S., Bapuji Institute of Engineering and Technology, Davanagere, Karnataka, India[2,3,4,5]

ABSTRACT: An advanced audio generation application built using Streamlit and Meta's Audiocraft framework. It integrates two state-of-the-art models, MusicGen and AudioGen, allowing users to generate both musical compositions and ambient soundscapes from text prompts. Users can refine the generated audio by adjusting parameters such as Top-k sampling and Temperature to control creativity and coherence. Additionally, the app supports optional features like reverb effects and stereo output conversion, enhancing the overall sound quality. The intuitive interface permits combining general music prompts with optional instrumental details, enabling dynamic and personalized audio production. This flexible system caters to a wide range of creative needs, from orchestral-inspired anthems to immersive environmental sounds, and demonstrates the potential of generative AI to democratize audio creation for both enthusiasts and professionals.

KEYWORDS: Deep Learning, Audio Generation, MusicGen, AudioGen, Text-to-Music, AI-Generative Models, Large Language Models

I. INTRODUCTION

The rapid evolution of deep learning has paved the way for revolutionary advancements in audio generation, transforming how we create and experience sound. Harnessing the power of state-of-the-art models like MusicGen and AudioGen, this project leverages text-to-music synthesis to produce professional-grade audio outputs from simple textual descriptions. By integrating customizable parameters and an intuitive Streamlit interface, the platform streamlines the complex process of music creation, democratizing access for both seasoned musicians and enthusiasts with little to no prior experience.

Leveraging cutting-edge technologies in deep learning, this project redefines music and sound creation by enabling users to effortlessly transform textual prompts into rich, high-fidelity audio. By deploying sophisticated models such as MusicGen and AudioGen, it not only simplifies the creative process but also offers an unprecedented level of customization through adjustable parameters like sampling techniques and fade effects. This innovative platform is designed to serve as a bridge between complex audio production and everyday creativity, empowering individuals regardless of their musical background to generate ambient soundscapes and musical compositions with professional quality.

- **Large Language Models (LLM)**
  Large language models (LLMs) are deep neural networks—most commonly based on the Transformer architecture—that are pretrained on massive text corpora to learn the statistical patterns of human language. At their core, Transformers replace recurrence with multi-headed self-attention mechanisms, enabling each token in an input sequence to dynamically weigh and integrate information from all other tokens, which vastly improves the model's ability to capture long-range dependencies compared to earlier RNN-based approaches. The size of these models is measured by the number of trainable parameters—GPT-3, for example, has 175 billion, while smaller variants like Phi-1.5 have 1.3 billion—reflecting the model's capacity to store nuanced linguistic and factual knowledge. Pretraining LLMs involves self-supervised objectives: **masked language modeling** (MLM), where the model learns to fill in missing tokens, and **autoregressive prediction**, where it learns to predict the next token in a sequence. During inference, these models can adapt to new tasks without gradient updates by leveraging **in-context learning**, wherein a handful of examples or instructions included directly in the prompt guide the model's behavior. The **context window**—the maximum number of tokens the model can process at once—varies by architecture (e.g., 2 K tokens for GPT-3, up to 32 K for GPT-4) and dictates how much conversational or

document history the model can consider in a single query. Advanced prompting techniques such as **Chain-of-Thought** further enhance reasoning by encouraging the model to generate intermediate logical steps, boosting factual recall and problem-solving performance.

Despite their remarkable fluency and versatility, LLMs face critical challenges. They are prone to **hallucinations**, producing plausible but incorrect or fabricated information, which undermines trustworthiness in real-world applications. Because they learn from uncurated web-scale data, LLMs often **perpetuate social biases**, necessitating ongoing research into de-biasing methods and logical constraints to mitigate harmful stereotypes.

The immense computational and energy demands for pretraining and inference raise **environmental** and **access-equity** concerns, while security risks emerge when models inadvertently memorize sensitive training data. Finally, assuring the **alignment** and **safety** of LLMs—ensuring they act according to human values and remain robust against misuse—poses foundational scientific and sociotechnical challenges that the research community continues to unpack.

Advantages of Large Language Models

1. **Efficiency & Productivity:**
   By automating text-centric tasks—drafting emails, generating reports, or tagging content—LLMs dramatically reduce manual effort and accelerate workflows.
2. **Scalability & Reusability:**
   A single pretrained LLM can serve dozens of applications via fine-tuning or prompt-engineering, avoiding the cost of building separate models for each task.
3. **Adaptability & Rapid Prototyping:**
   In-context learning allows developers to tweak behavior on the fly with a handful of examples, speeding up experimentation and deployment cycles.
4. **Accessibility & Ease of Integration:**
   Exposed via simple APIs or no-code interfaces, LLMs unlock advanced NLP capabilities for nontechnical teams—marketing, legal, education—through plain-language prompts.
5. **Deep Contextual Understanding:**
   LLMs excel at modeling nuance, disambiguation, and long-range dependencies, yielding more coherent and contextually appropriate outputs than legacy NLP systems.
6. **Embedded Knowledge & Zero-Shot Reasoning:**
   Pretrained on a broad corpus, LLMs store factual information and can answer questions or perform tasks with little to no additional training, rivaling specialized knowledge bases.

Applications of Large Language Models

- **Conversational AI & Virtual Assistants:**
  Deploying LLMs in chatbots enhances customer support by understanding intent, handling complex queries, and providing human-like dialogue 24/7.
- **Automated Content Generation:**
  LLMs draft blog posts, marketing copy, and social-media updates, freeing writers from repetitive tasks and sparking creative ideas.
- **Machine Translation & Localization:**
  Fine-tuned LLMs produce high-quality translations for multiple languages and domains, supporting global communication and publishing workflows.
- **Code Completion & Review:**
  Developer tools like GitHub Copilot leverage LLMs to suggest code snippets, refactor functions, and document APIs, boosting software engineering productivity.
- **Summarization & Information Extraction:**
  LLMs condense long documents into digestible summaries, extract key facts into structured formats, and accelerate data-analysis pipelines.
- **Research Assistance & Insights:**
  Academics use LLMs to survey literature, generate hypotheses, and draft manuscripts, streamlining the research lifecycle

- **Creative & Multimodal Systems:**
  Emerging applications blend text, vision, and audio (e.g., image captioning, music generation), enabling rich, interactive AI experiences across media

II. RELATED WORK

Yu et al. introduce MusicAgent, an LLM-powered system that integrates and orchestrates diverse music processing tools—from Hugging Face models to Web APIs—to automatically decompose user requests into sub-tasks and invoke the appropriate modules for tasks like composition, transcription, and retrieval; Li et al. present AI Choreographer, with the AIST++ dataset (5.2 hours of multi-view 3D dance motion paired with music) and a Full-Attention Cross-modal Transformer (FACT) that generates long, realistic 3D dance sequences tightly correlated to input music; Dong et al. propose MuseGAN, three GAN architectures (jamming, composer, hybrid) for multi-track symbolic music generation, capable of producing coherent four-bar rock-style piano-rolls and AI-assisted accompaniment for human-provided tracks; Dong and Yang develop BMuseGAN, appending a binary-neuron refiner network to a convolutional GAN to directly output binary piano-rolls—showing deterministic binary neurons yield better objective and subjective results than hard thresholding or sampling; Afchar et al. design the first AI-Generated Music Detector, training simple convolutional classifiers to distinguish real from synthesized audio with ~99.8 % accuracy, and discuss challenges of robustness to post-processing and generalization to new generative models; Tokui presents RhythmVAE, a VAE-based Ableton Live (Max for Live) plugin that musicians can train on arbitrary MIDI datasets via drag-and-drop and interactively explore a 2D latent space to generate and audition novel rhythms in real time; Zhang et al. propose Loop Copilot, an LLM “conductor” that interprets conversational user prompts to select and sequence specialized music-generation and editing models—maintaining coherence via a shared Global Attribute Table in multi-round iterative refinement of loops; and Bryan-Kinns et al. conduct a systematic study of Explainable VAEs for monophonic music generation, comparing MeasureVAE vs. AdversarialVAE, latent dimensionalities (4–256), and genre-specific datasets (Irish folk, Turkish folk, classical, pop) to identify configurations (e.g., 32–64 latent dims with MeasureVAE) that best balance reconstruction fidelity and semantically meaningful, user-controllable latent features.

III. PROPOSED ALGORITHM

1. **User Input Collection:**
   The user provides a primary text prompt describing the desired audio.
   Optionally, an instrumental prompt can be added to specify instruments, styles, or other musical elements.

2. **Model Selection and Loading:**
   Based on the user's choice, either the MusicGen or AudioGen model is selected.
   The selected model variant (e.g., "facebook/musicgen-small") is loaded using a caching mechanism to optimize performance.

3. **Text Encoding:**
   The combined text prompt is processed through a pre-trained text encoder (such as T5 or FLAN-T5) to generate hidden-state representations that capture the semantic meaning of the input.

4. **Audio Token Generation:**
   The model sets generation parameters, including:
   - **Duration**: Length of the audio to be generated.
   - **Top-k Sampling**: Controls the diversity by limiting the number of token options considered at each step.
   - **Temperature**: Adjusts the randomness of token selection; higher values yieldmore diverse outputs.
     Using these parameters, the model generates a sequence of audio tokens in an autoregressive manner, predicting each token based on the preceding ones.

5. **Audio Decoding:**
   The sequence of audio tokens is decoded back into a continuous audio waveform using an audio decoder like EnCodec.
   EnCodec utilizes multiple codebooks to reconstruct high-fidelity audio from the discrete tokens.

6. **Post-Processing**
   The raw audio waveform undergoes normalization to ensure consistent volume levels and prevent clipping. Fade-in and fade-out effects are applied to the beginning and end of the audio to enhance smoothness and eliminate abrupt transitions.

7. **Output Delivery**
   The processed audio is saved in a specified directory (e.g., "audio_output/") in WAV format. Download links are generated using base64 encoding, allowing users to easily download the generated audio files.

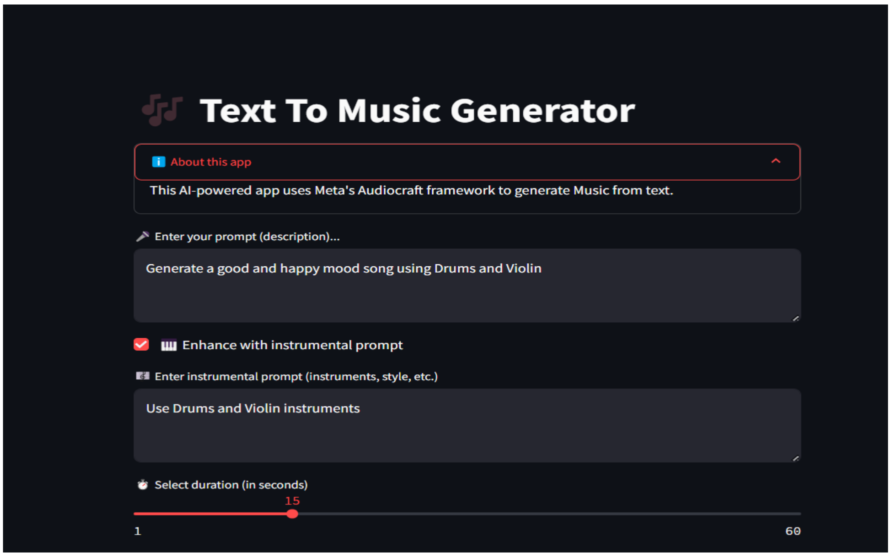
8. **User Interface Integration**
   The entire process is encapsulated within a Streamlit application, providing an interactive interface where users can:
   - Input text and instrumental prompts.
   - Select model types and variants.
   - Adjust advanced parameters like top-k and temperature.
   - Initiate audio generation and access the results directly within the app.

IV. PSEUDO CODE

```
Step 1: Load the Pre-trained Model
        Select and load the appropriate model (MusicGen or AudioGen) along with the desired variant (e.g., 'facebook/musicgen-small').
Step 2: Gather Input Data
        Obtain the text prompt describing the desired audio.
        Optionally, include an instrumental prompt to specify musical elements.
Step 3: Preprocess the Input
        Combine the text and instrumental prompts if both are provided.
        Clean the text by removing special characters and unnecessary whitespace.
Step 4: Encode the Text Prompt
        Use a pre-trained text encoder (e.g., T5 or FLAN-T5) to convert the text prompt into a latent representation.
Step 5: Generate Audio Tokens
        Set generation parameters such as duration, top-k sampling, and temperature. Feed the encoded text into the model to generate a sequence of audio tokens.
Step 6: Decode Audio Tokens to Waveform
        Use an audio decoder (e.g., EnCodec) to convert the audio tokens into a continuous audio waveform.
Step 7: Post-process the Audio
        Normalize the audio to ensure consistent volume levels. Apply fade-in and fade-out effects to smooth the beginning and end of the audio.
Step 8: Output the Final Audio
        Save the processed audio as a .wav file.
        Provide options for playback and download through the user interface.
Step 9: Repeat for New Inputs
        Allow users to input new text prompts and repeat the process for generating additional audio samples.
```

V. SIMULATION RESULTS

In our simulation, we tested the text-to-audio generation platform using various prompts to evaluate the capabilities of Meta's MusicGen and AudioGen models. For instance, when given a prompt describing a grand orchestral arrangement with thunderous percussion and soaring strings, the system produced a rich, cinematic audio piece that matched the description. Similarly, prompts like "classic reggae track with an electronic guitar solo" and "drum and bass beat with intense percussions" resulted in genre-appropriate musical outputs, showcasing the model's versatility in handling different musical styles. On the environmental sound front, using AudioGen, prompts such as "sound of a dog barking" and "cars honking in a busy street" generated realistic audio clips that accurately represented the described sounds. These results demonstrate the platform's effectiveness in translating textual descriptions into high-quality audio, making it a valuable tool for users seeking to create music or ambient sounds without prior audio production experience.



Text To Music Generator

About this app

This AI-powered app uses Meta's Audiocraft framework to generate Music from text.

Enter your prompt (description)...

Generate a good and happy mood song using Drums and Violin

☑ Enhance with instrumental prompt

Enter instrumental prompt (instruments, style, etc.)

Use Drums and Violin instruments

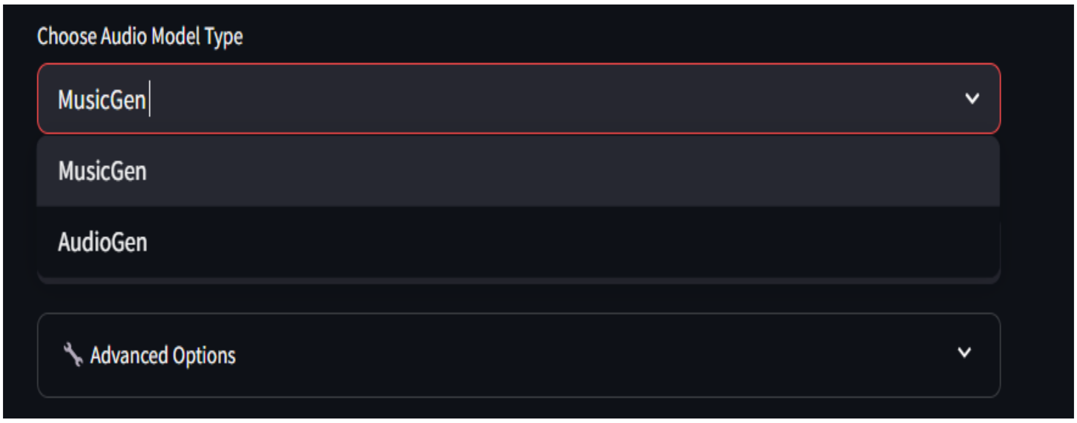
Select duration (in seconds)

15

1 60

Fig. 4.1 Main Text To Music Generator interface

Fig. 4.1 Shows the main “Text To Music Generator” interface. At the top is the app title with a collapsible “About this app” panel explaining that it uses Meta’s Audiocraft framework. Below, there’s a large text field where you type your main prompt—here, “Generate a good and happy mood song using Drums and Violin.” A checkbox lets you turn on an “Enhance with instrumental prompt” option, and a second box appears for specifying instruments or style details (“Use Drums and Violin instruments”). Finally, a red slider at the bottom lets you choose the desired clip duration from 1 to 60 seconds.

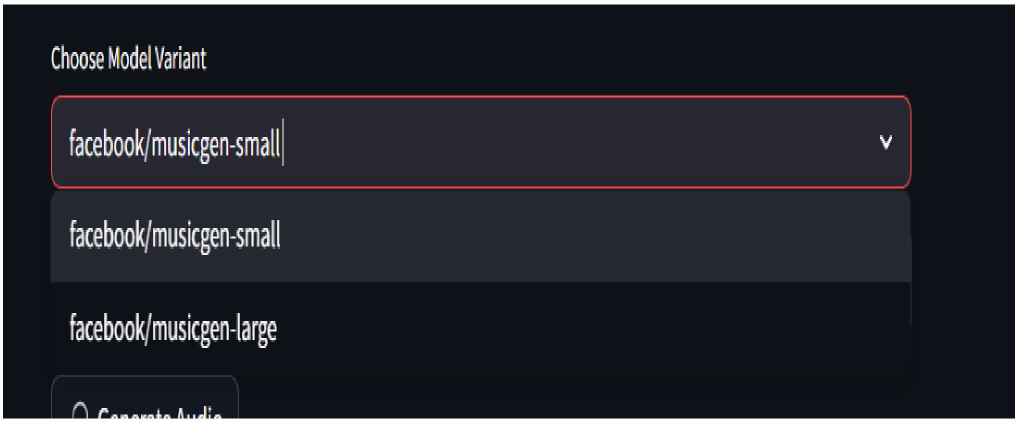


Choose Audio Model Type

MusicGen

Fig. 4.2 Choose Audio Model Type to Generate Music

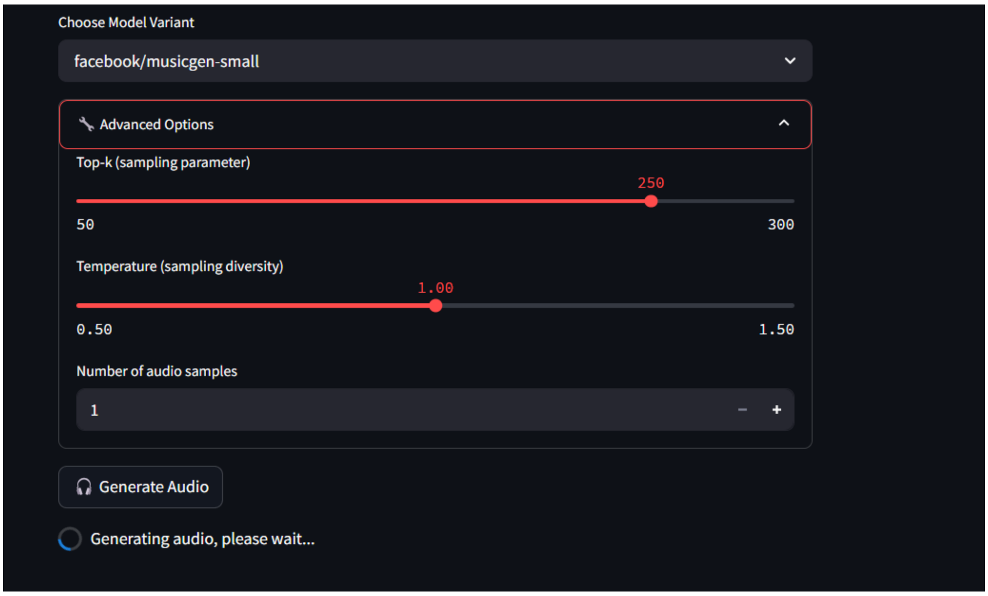
Fig. 4.2 Shows a dropdown labeled “Choose Audio Model Type.” The menu is open and shows two options: **MusicGen** (intended for full musical pieces) and **AudioGen** (better suited for shorter sound effects or audio snippets). It’s a simple way to switch between different underlying generation engines.


Choose Model Variant

facebook/musicgen-small

Fig. 4.3 Choosing of Model Variant based on required Music Quality

Fig. 4.4 Shows the “Choose Model Variant” dropdown in action. Once you’ve picked MusicGen or AudioGen, this menu lets you select a specific model checkpoint. The example shows “facebook/musicgen-small” highlighted, with “facebook/musicgen-large” also available—smaller models run faster with fewer resources, while larger ones produce richer results.


Choose Model Variant: facebook/musicgen-small

Advanced Options

Top-k (sampling parameter): 250 (50 – 300)

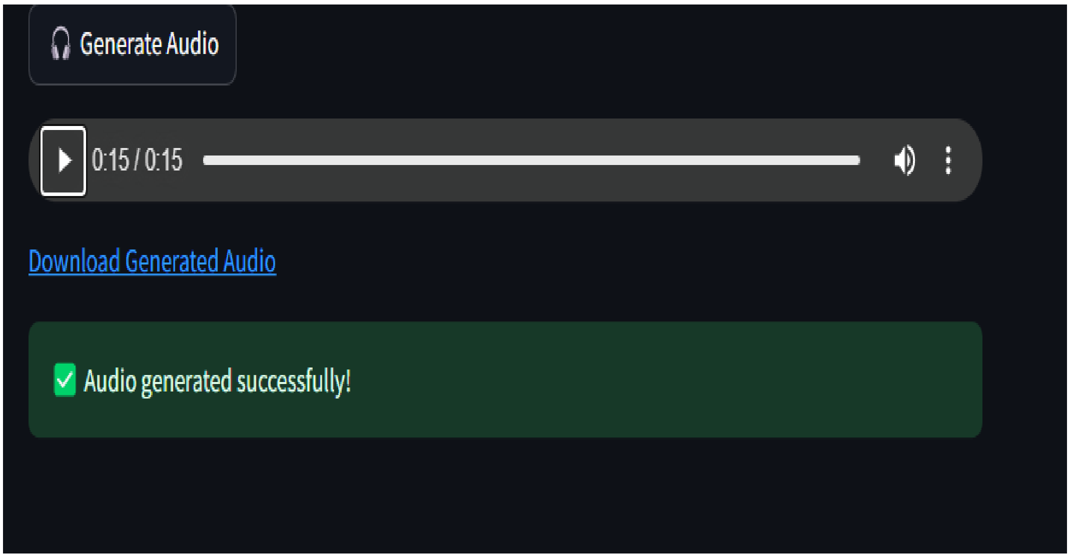
Temperature (sampling diversity): 1.00 (0.50 – 1.50)

Number of audio samples: 1

Generating audio, please wait...

Fig. 4.4 Adjustment of Top-k, Temperature and Number of samples required

Fig. 4.4 Shows the “Advanced Options” section expanded under the model-variant selector. Three controls appear: a **Top-k** slider (adjusting how many of the highest-probability next-note choices the model considers), a **Temperature** slider (tuning creativity vs. predictability), and a spinner for “Number of audio samples” to generate at once. Below is a big “Generate Audio” button, and, once clicked, a small loading spinner appears with the label “Generating audio, please wait…” to indicate the process is running.



✓ Audio generated successfully!

Fig. 4.5 Final Generated Music and ready to Download

Fig. 4.5 Shows once generation finishes, this view shows an embedded audio player bar with play/pause controls, a timeline indicating “0:15 / 0:15,” and a volume button, allowing you to preview the clip right in the browser. Below it is a “Download Generated Audio” link so you can save the file locally, and a green success banner reading “Audio generated successfully!” to confirm that your request completed without errors.

VI. CONCLUSION AND FUTURE WORK

The development of our text-to-audio generation platform utilizing Meta's MusicGen and AudioGen models has successfully demonstrated the capability to transform textual descriptions into high-quality musical compositions and realistic ambient sounds. This innovation democratizes audio content creation, enabling users without formal musical training to produce professional-grade audio outputs. The platform's user-friendly interface and efficient processing have shown promise in various applications, from creative arts to accessibility tools.

Looking ahead, future work will focus on enhancing the system's versatility and user engagement. This includes expanding the range of supported languages and musical styles to cater to a broader audience. Incorporating user feedback mechanisms will allow for more personalized audio generation, aligning outputs more closely with user expectations. Additionally, integrating advanced features such as emotion recognition and adaptive learning could further refine the quality and relevance of the generated audio. Addressing challenges related to real-time processing and ensuring ethical use of AI-generated content will also be pivotal in the platform's ongoing development and deployment.

REFERENCES

[1]. D. Yu, K. Song, P. Lu, T. He, X. Tan, W. Ye, S. Zhang, and J. Bian, “MusicAgent: An AI Agent for Music Understanding and Generation with Large Language Models,” arXiv preprint arXiv:2310.11954v2, Oct. 2023.

[2]. R. Li, S. Yang, D. A. Ross, and A. Kanazawa, “AI Choreographer: Music Conditioned 3D Dance Generation with AIST++,” arXiv preprint arXiv:2101.08779v3, Jul. 2021.

[3]. H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, “MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment,” arXiv preprint arXiv:1709.06298v2, Nov. 2017.

[4]. H.-W. Dong and Y.-H. Yang, “Convolutional Generative Adversarial Networks with Binary Neurons for Polyphonic Music Generation,” arXiv preprint arXiv:1804.09399v3, Oct. 2018.

[5]. D. Afchar, G. Meseguer-Brocal, and R. Hennequin, “AI-Generated Music Detection and its Challenges,” arXiv preprint arXiv:2501.10111v1, Jan. 2025.

[6]. N. Tokui, "Towards Democratizing Music Production with AI—Design of Variational Autoencoder-based Rhythm Generator as a DAW Plugin," arXiv preprint arXiv:2004.01525v1, Apr. 2020.
[7]. Chundru, Swathi & Whig, Pawan. (2024). Future of Emotional Intelligence in Technology: Trends and Innovations. 10.4018/979-8-3693-7011-7.ch024.
[8]. Y. Zhang, A. Maezawa, G. Xia, K. Yamamoto, and S. Dixon, "Loop Copilot: Conducting AI Ensembles for Music Generation and Iterative Editing," arXiv preprint arXiv:2310.12404v2, Aug. 2024.
[9]. N. Bryan Kinns, B. Zhang, S. Zhao, and B. Banar, "Exploring Variational Auto Encoder Architectures, Configurations, and Datasets for Generative Music Explainable AI," arXiv preprint arXiv:2311.08336v1, Nov. 2023.