Technical Disclosure Commons

Defensive Publications Series

February 2024

Augmenting Large Language Models with Audio Generation Capabilities

Ágoston Weisz

Timo Denk

Mauricio Zuluaga

Alessandro Agostini

Christian Frank

See next page for additional authors

Follow this and additional works at: https://www.tdcommons.org/dpubs_series

Recommended Citation

Weisz, Ágoston; Denk, Timo; Zuluaga, Mauricio; Agostini, Alessandro; Frank, Christian; and Siegenthaler, Olivier, "Augmenting Large Language Models with Audio Generation Capabilities", Technical Disclosure Commons, (February 08, 2024)
https://www.tdcommons.org/dpubs_series/6668

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

Inventor(s)
Ágoston Weisz, Timo Denk, Mauricio Zuluaga, Alessandro Agostini, Christian Frank, and Olivier Siegenthaler

This article is available at Technical Disclosure Commons: https://www.tdcommons.org/dpubs_series/6668

Augmenting Large Language Models with Audio Generation Capabilities

ABSTRACT

Chatbots or conversational agent interfaces utilize large language models (LLMs) to provide text responses to user queries. However, such chatbots are not capable of receiving audio input and providing generated audio as a response. This disclosure describes techniques to augment a LLM with an interface to an audio generation model. The LLM is fine-tuned to train it to leverage an API to access the audio generation model when input queries request query response in audio form. The trained LLM performs reasoning tasks and generates prompts for the audio generation model. The user-provided audio input and the LLM-generated prompts are fed to the audio generation model which generates audio. The output audio is analyzed to determine attributes as textual description. The LLM can perform multiple rounds of reasoning, prompt generation, and calling the audio generation model based on previously generated audio and associated textual descriptions. The ultimate audio output as generated by the audio generation model is provided as a response to the user query.

KEYWORDS

- Generative audio
- Large language model (LLM)
- Conditioned audio generation
- Multimodal generation
- Melody conditioning
- Automatic music tagging
- Chain of thought
- Chatbot
- Multimodal response
- Conversational agent
- Speech recognition
- Reasoning model

BACKGROUND

Chatbots or conversational agents are used in various domains and can provide an enhanced user experience and improve operational efficiency. Current chatbots based on large language models (LLMs) can only process and generate responses for text inputs. LLMs can only process a limited set of tokens or inputs provided in sequence. Current chatbots are not capable of receiving audio input and providing generated audio as a response.

```
User: could you add a guitar solo to this tune? <audio>
Observation: the user wants me to add a guitar solo to a tune.
Thought: I don't know how to add a guitar solo to a tune.
Action: reply to the user:
System: I don't know which tune you are referring to. I cannot
generate audio.
```

Table 1: Chatbot based on LLM cannot interpret or respond to multimodal queries

Table 1 illustrates an example of a user query that includes text and audio. In the query, the user requests the chatbot to “add a guitar solo” to a tune, with the input including audio of the tune. The internal states of the model are also illustrated - under observation, the query interpretation indicates that the query is interpreted correctly (but the audio is discarded); under thought, it is reasoned that the user request cannot be fulfilled by the model; and under action, an appropriate reply is generated. The reply indicates that the model cannot determine the “tune you are referring to” and that it “cannot generate audio.”

DESCRIPTION

Users can benefit from being able to interact with chatbots or conversational agents via audio and receive responses as audio. To enable such interaction, the chatbot needs the capability to analyze audio input, reason about it, and provide a response with new generated audio samples along the flow of the conversation. Audio input from the user can include spoken language or other audio, potentially with a large number of tokens. The audio output needs to be contextually relevant, coherent, and responsive to the user input.

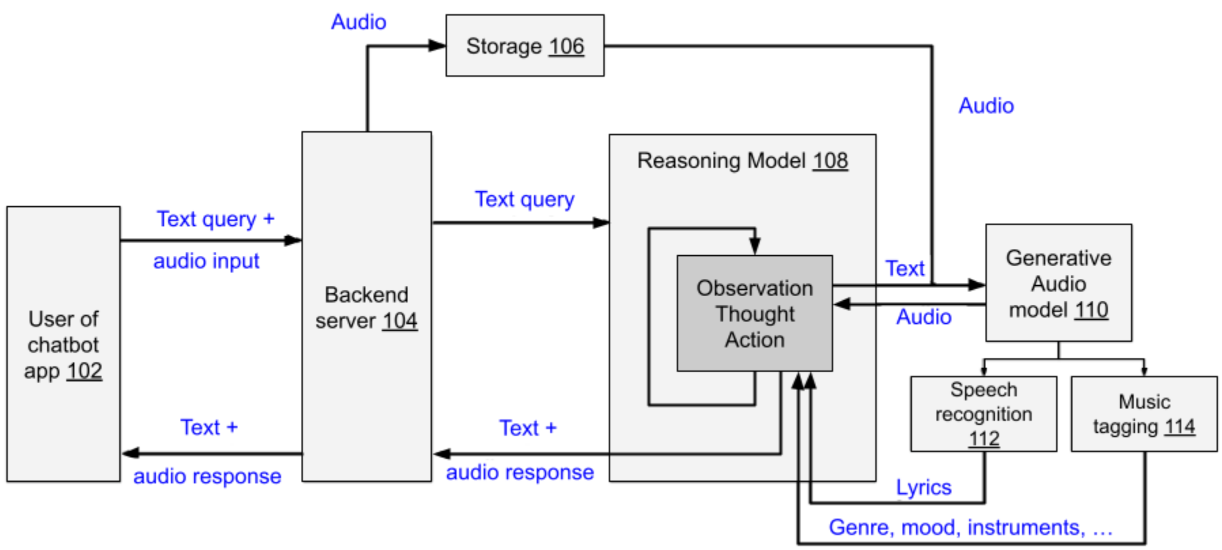

Fig. 1: Illustration of a large language model augmented with a generative audio model

Fig. 1 illustrates an example of leveraging a large language model (LLM) for audio processing by augmenting the LLM with a generative audio model. When a user (102) of a chatbot or conversational agent provides a query that includes text and optionally, the user-provided inputs are sent to a backend server (104). If audio is present in the user input query, it is stored (106). The text input is provided to a reasoning model (108), e.g., a LLM, for query processing.

The reasoning model analyzes the text query via a series of observation, thought, and action sequences. Based on the identified actions, the reasoning model generates text prompts for a generative audio model (110) which are provided to the model along with the stored audio. The audio output generated by the audio model is analyzed by additional models such as speech recognition (112), e.g., to detect text part of the generated output, e.g., lyrics; music tagging (114) to detect attributes such as genre, mood, instruments playing in the audio; etc.

Textual descriptions obtained from such models and the generated audio from the generative audio model are fed back to the reasoning model for additional executions of the LLM reasoning loop. In such executions, a piece of generated audio may be used as part of the subsequent inputs to the generative audio model. Providing portions of generated audio and corresponding textual descriptions to the reasoning model can ensure that the overall conversation with the user is consistent and relevant to the user input query. The ultimate output, including text and audio response generated by the reasoning model augmented by the generative audio model is provided to the user via the backend server.

```
Context: audio sample provided: <pointer-to-the-audio-bytes>
User: can you add a guitar solo to this tune?
Observation: the user wants me to add a guitar solo and they
uploaded a music file.
Thought: I could call an audio generation model
Action: call audio generation model with: (<audio bytes>, "add a
guitar solo")
Observation: I received generated audio + {
             lyrics: "I got rhythm, I got music"
             genre: "jazz standard"
             instruments: "vocals, guitar, bass, drums"
              }
Thought: the user wanted to add a guitar solo to a tune,
and I have an audio file with a guitar solo added.
Action: reply to the user:
System: Sure, here is a new version of this famous jazz standard:
<audio player>
```

Table 2: LLM augmented with audio generation model responds to multimodal queries

Table 2 illustrates interaction between a user and a LLM augmented with an audio generation model as described herein, where the model output includes text and audio response. The user provides a multimodal query, with the input text "add a guitar solo to this tune" along with an audio sample. The internal states of the model are illustrated. As seen in Table 2, the observation state indicates that the model interprets the query as a request to add a guitar solo and that the input includes a music file. The thought state of the model indicates that an audio generation model can be called to augment query processing. At the action state, the LLM calls the audio generation model, providing the user input audio and an instruction ("add a guitar solo") in the prompt provided to the audio generation model.

As described with reference to Fig. 1, the audio generated by the model and associated text are fed back into the LLM which is the subsequent observation. The next thought state indicates that the audio generated by the audio generation model is responsive to the user query. Accordingly, the model generates a response that includes the audio and associated text ("famous jazz standard"). Note that table 2 presents a simplified view; in practice, there may be multiple loops between the LLM and the generative audio model to obtain audio responsive to the user query, with multiple associated LLM internal states.

Model training

The addition of a new capability to the reasoning model can be accomplished by training the LLM that performs the reasoning task to understand how to use the audio generation model which can be accessed via an application programming interface (API). This can be accomplished by fine-tuning the LLM as follows:

To perform the training, golden tuples of {user query, context, groundtruth LLM response} can be collected manually or automatically by filtering automatic responses that match quality criteria. Once a training set of sufficient size is collected, the LLM can be fine-tuned, e.g., by recalibrating its weights to make it more likely to output API calls to leverage the audio generation model. Table 3 illustrates the difference in query processing by a fine-tuned LLM from a LLM without such fine-tuning.

LLM without fine-tuning	LLM with fine-tuning and API access to audio generation model

`context: uploaded audio file: audio_1.mp3`
`query: add a guitar solo to this tune`
`system: I don't know how to add a guitar solo to a tune.`
	`context: uploaded audio file: audio_1.mp3`
	`query: add a guitar solo to this tune`
	`system: API call: generate_music("add a guitar solo", "audio_1.mp3")`

Table 3: LLM fine-tuning to enable use of audio generation model

In this manner, the described techniques utilize audio processing and language modeling technologies such as conditioned audio generation, melody conditioning, speech recognition, automatic music tagging, etc. to generate and analyze audio based on input audio and an LLM-generated prompt. The LLM can be fine-tuned, and the reasoning performed by the LLM can be decomposed into a series of intermediate reasoning steps to enable the use of an audio generation model to generate a response that includes output audio responsive to the multimodal input query.

The described techniques can be implemented in a chatbot or conversational agent, or any other interface that enables users to provide queries including audio and request responses in audio format. While the foregoing description refers to an input query with text and audio, the chatbot can also interpret spoken queries supplied along with audio.

Further to the descriptions above, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs or features described herein may enable collection of user information (e.g., information about a user's input queries and audio, a user's preferences), and if the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user.

CONCLUSION

This disclosure describes techniques to augment a LLM with an interface to an audio generation model. The LLM is fine-tuned to train it to leverage an API to access the audio generation model when input queries request query response in audio form. The trained LLM performs reasoning tasks and generates prompts for the audio generation model. The user-provided audio input and the LLM-generated prompts are fed to the audio generation model which generates audio. The output audio is analyzed to determine attributes as textual description. The LLM can perform multiple rounds of reasoning, prompt generation, and calling the audio generation model based on previously generated audio and associated textual descriptions. The ultimate audio output as generated by the audio generation model is provided as a response to the user query.

REFERENCES

1. "Discover How AI Turns Your Text Prompt to Music," available online at https://contentatscale.ai/blog/prompt-to-music/ accessed Dec 21, 2023
2. "Fine-tune MusicGen to generate music in any style," available online at https://replicate.com/blog/fine-tune-musicgen accessed Dec 21, 2023
3. "WavTool: The World's First Text-to-Music AI DAW Uses GPT-4," available online at https://www.audiocipher.com/post/ai-daw accessed Dec 21, 2023
4. "AI Music Generator," available online at https://www.veed.io/tools/ai-music-generator accessed Dec 21, 2023