AlbumGen: Image-to-Music Generation with Textual Intermediaries

AU, Cheuk Sau
csauac@connect.ust.hk
21014200

LAI, Chun Yu
cylaiay@connect.ust.hk
20972118

Abstract

This project explores the intersection of computer vision and music generation by utilizing image captioning models on album cover art and employing text-to-music generation models, such as MusicGen, to create music based on these captions. By training a model to generate descriptive captions from album covers, we aim to develop a system that automatically produces music aligned with the visual themes of album artwork. This innovative approach opens new avenues for creative AI-assisted music production.

1 Introduction

Music is prevalent in our culture and even dates back to times of ancient history. Today, when artists create music, they consider not only the many structures of music, such as tone, timbre, rhythm, pitch, and key, but more broadly, how a user might experience their music. Album covers are an integral part of an artist's style, hinting at the album's genre and mood and evoking a broader multi-sensory appreciation of music [Ioannou et al.]. However, connecting the visual design of the album cover with the music itself hasn't been explored much. With new technology in image captioning and text-to-music generation, we can potentially automate music creation based on how the album cover looks. This project aims to use image captioning to describe what's shown on album covers and then use models like MusicGen to turn these descriptions into music. This could open a creative way for artists and producers to explore the link between what we see and hear in music.

2 Related Work

Developments of various architectures and applications of large language models have grown since the introduction of self-attention mechanisms [Vaswani et al., 2017]. More recently, innovation has extended to developing multi-modal large language models such as GPT4 [Adler et al.]. The rise of generative models has also led to innovation in other modalities, such as its application into music generation, with the first convincing result starting with models such as WaveNet [Oord et al., 2016]. Since then, model architecture and performance have greatly accelerated, notably the development of encoder-decoder architecture models such as the EnCodec model [Défossez et al., 2022]. In the EnCodec model, a music file is fed to a streaming, convolutional architecture encoder, which encodes the music clip to a latent space. The results are then quantized and reconstructed to the decoder, and the model is trained via a reconstruction and adversarial loss.

3 Pretrained Models and Datasets

Our project aims to explore the development of an image-to-music pipeline with LLMs. Rather than delving into philosophical interpretations of how music *should* be conditioned on image inputs, we strive to ground our pipeline on explainable intermediaries in the generation process through text. We plan on incorporating image-to-text, text-to-music, and prompt engineering into our pipeline.

3.1 Pretrained Models

Image Captioning: We utilize CLIP (Contrastive Language-Image Pretraining) to generate captions from album cover images. These models are trained on extensive image-text datasets and can be fine-tuned for captioning album art.

Text-to-Music Generation: To convert the generated captions into music, we employ Meta's MusicGen model. This model is designed to produce music based on text prompts, making it well-suited for transforming captions into musical compositions.

3.2 Dataset

The MusicOSet Dataset: The MusicOSet dataset [Rocha et al., 2019] serves as a comprehensive resource for music data mining, providing enriched metadata on music, artists, and albums. This dataset includes an annotated collection of over 20,000 songs, over 26,000 albums, and 11,000 artists. The data is structured in a relational database format (SQL), featuring a complex schema encompassing various attributes essential for our project.

- **Artists Information:** Each artist entry contains detailed information such as:
  - Popularity Score: Rated from 0 to 100, indicating the artist's overall popularity.
  - Type: Classification of the artist (e.g., solo singer, band, duo, rapper).
  - Genres: Associated musical genres that define the artist's style.
- **Albums Data:** The dataset provides extensive details on albums, including:
  - Popularity Ratings: Reflecting the album's reception in the music industry.
  - Total Number of Tracks: The count of individual tracks within each album.
  - Album Type: Categorization into full albums, singles, or compilations.
  - Image URLs: Links to album cover images that visually represent the music.
- **Tracks Metadata:** Each track entry is rich with metadata, including:
  - Popularity Score: A score reflecting the track's popularity similar to that of artists.
  - Explicit Content Indicator: A flag indicating whether the track contains explicit content.
  - Type of Track: Differentiation between solo and collaborative tracks.
  - Musical Attributes: Detailed features such as key and mode (major/minor), time signature, energy levels, danceability, and various acoustic characteristics.

Musical Attributes: The dataset encompasses numerous musical characteristics critical for analysis:

- **Acousticness:** Measures the likelihood that a track is acoustic (0.0 to 1.0).
- **Danceability:** Reflects how suitable a track is for dancing based on tempo and rhythm (0.0 to 1.0).
- **Energy:** Indicates intensity and activity levels in tracks (0.0 to 1.0).
- **Instrumentalness:** Suggests the probability of a track being instrumental (values closer to 1.0 indicate higher likelihood).
- **Liveness:** Measures audience presence during recording; higher values suggest live performances.
- **Loudness:** Overall loudness represented in decibels (dB), typically ranging from -60 to 0 dB.
- **Speechiness:** Detects spoken words; values near 1.0 indicate speech-like recordings.
- **Valence:** Describes musical positiveness conveyed by a track (0.0 to 1.0), where higher values indicate more positive emotions.

This comprehensive metadata enables researchers to correlate visual elements from album covers with musical characteristics. By leveraging this dataset, our project aims to explore meaningful relationships between image captions derived from album art and the music generated through our pipeline. This exploration will enhance our understanding of factors contributing to musical popularity and trends within the music industry.

3.3 Pipeline

The pipeline for our project, "AlbumGen: Image-to-Music Geffectively eneration with Textual Intermediaries," is designed to facilitate the generation of music based on album cover art through a structured, multi-step approach. This section outlines the key components of the pipeline, including the processes of image embedding, feature extraction, and music generation.

Album-to-Text: The pipeline begins with the conversion of album cover images into textual representations. This is achieved using a Vision Language Model (VLM) that extracts key musical features and attributes from the images. The model is prompted to output essential musical parameters

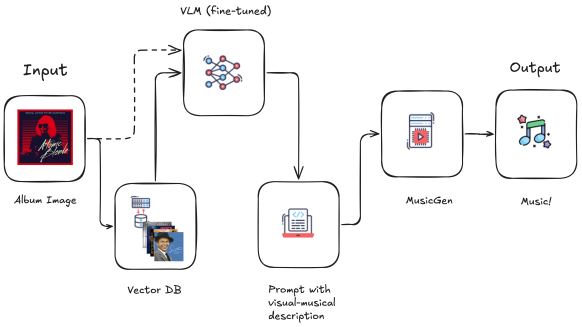

Figure 1: Flowchart of the pipeline for generating music from album cover images, illustrating the key steps from image processing to music generation.

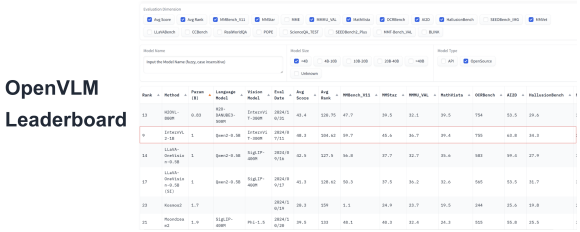

Figure 2: OpenVLM Leaderboard

alongside a descriptive caption based on the album title.

CLIP Model Selection for Image Embedding: For image embedding, we utilize the CLIP model "openai/clip-vit-base-patch16." This model employs a Vision Transformer (ViT) architecture to align image and text representations by maximizing their similarity during training. We selected this pre-trained model without fine-tuning to leverage its robust capabilities in zero-shot learning tasks, allowing it to generalize effectively across various image classification scenarios.

Image Processing: The dataset used, MusicOSet, contains URLs of album art. We scraped a total of 1,000 images, resizing them to 640x640 pixels for consistency in input dimensions for the CLIP model.

Embedding Extraction: In this step, the preprocessed album cover images are passed through the CLIP model to extract embeddings. Each image is transformed into a 512-dimensional embedding that captures its visual semantics, encoding various visual features for efficient comparison and retrieval.

3.3.1 Visual Features Representation:

The embeddings generated by CLIP encapsulate a range of visual features crucial for understanding the relationship between album art and musical genres: Color Palettes: Dominant colors may correlate with specific musical genres. Textures and Styles: Artistic styles can provide insights into the mood or theme of the music. Objects and Symbols: Elements in the artwork can suggest genre or thematic content.

Similarity Metrics: To compare embeddings retrieved from FAISS (Facebook AI Similarity Search), we employ L2 distance as our similarity metric. Given our dataset comprises fewer than 1,000 records, we prioritize quality matches over quantity to ensure that retrieved images closely align with their corresponding musical features.

Top K Selection: For each music feature extracted from MusicOSet (including 'acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'loudness', 'speechiness', 'valence'), we calculate average values from the top K similar albums identified through our similarity search. Experimentation with different values of K (ranging from 1 to 20) revealed that K = 5 yielded optimal results in balancing retrieval quality and computational efficiency.

3.3.2 Vision Language Model for Image-to-text translation

Model selection: The selection of the VLM greatly impacts the inference time and training time needed to prompt the model. To first decide on which VLM to use in our implementation, we referred to the HuggingFace vision leaderboard and selected an open-source model that performed within high-performing vision task. From this selection criterion, we arrived at using the InternVL2-1B model, a recently released VLM model adapted from the Qwen0.5B LLM, which has been fine-tuned for instructions.

VLM Prompt Design We initially designed our prompt template by assigning the LLM the role of a "music artist" tasked with describing the acoustic elements of the album for music generation. While the VLM was capable in providing a description, there was no quantitative method for us to measure how effective the model is in understanding the music album's acoustic features. Furthermore, as the role of language in this pipeline is to act as a deterministic semantic carriers, we wanted to ground the textual intermediaries to a processable, codable format.

We adopted an approach that prompts the VLM to generate a JSON-formatted output of the acous-

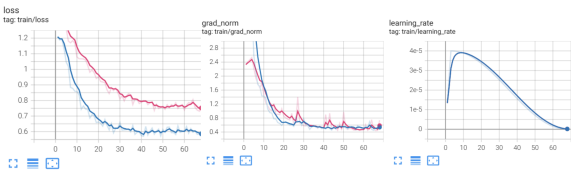

Figure 3: Training loss charts

tic feature. The prompt consists of three main components – a role assignment, an acoustic feature definition from the MusicOSet, and the expected format. In a few trial runs, we noticed that the VLM was only partially effective in generating usable JSON format, suggesting that the model needed to be finetuned further for consistency.

3.4 LoRa Finetuning

To improve the consistency in the JSON format, a LoRa adaptor was then finetuned using the training set. The training parameters ana d set-up can be found in our open-sourced repo under *finetune/internvl2_1b_finetune_lora_album.sh.* During our initial implementation, we used a LoRa model of rank=8. However, the model was >8M in parameter size by itself, causing overfitting to the training data. To reduce the effects of this, a second LoRa model was trained using only a rank=4. This still resulted in a large model with 2,199,552 trainable parameters, which suggested the model is still likely to require datasets of much larger degree of freedom to be fully configured. The training was conducted using the Nvidia Pytorch 24.05 docker container hosted on a local RTX3070Ti set-up. As this model was newly released, transformers automodels do not automatically support LoRa training for this model yet. As a result, we directly cloned the InternVL2 repository and finetuned the model using shell commands. The VLM was finetuned for the entire training set using the exact train text split in our RAG evaluation. Due to the minor training size breakdown, it consisted of only 1 epoch of finetuning. The training losses for both LoRa models can be seen in the figure below.

3.5 Musical prompt generation:

Acoustic description generation Given a JSON output with acoustic feature variables, our following task is to translate this to a usable music prompt for the MusicGen model. To explore how we can translate an acoustic variable to a music description, we prompted the GPT4o model, a description of the MusicOSet feature and the acoustic variable

```
torchrun \
  --nnodes=1 \
  --node_rank=0 \
  --master_addr=127.0.0.1 \
  --nproc_per_node=${GPUS} \
  --master_port=${MASTER_PORT} \
  internvl/train/internvl_chat_finetune.py \
  --model_name_or_path "./pretrained/InternVL2-1B" \
  --conv_style "Hermes-2" \
  --output_dir ${OUTPUT_DIR} \
  --meta_path "./shell/data/album_caption.json" \
  --overwrite_output_dir True \
  --force_image_size 448 \
  --max_dynamic_patch 6 \
  --down_sample_ratio 0.5 \
  --drop_path_rate 0.0 \
  --freeze_llm True \
  --freeze_mlp True \
  --freeze_backbone True \
  --use_llm_lora 4 \
  --vision_select_layer -1 \
  --dataloader_num_workers 4 \
  --bf16 True \
  --num_train_epochs 1 \
  --per_device_train_batch_size ${PER_DEVICE_BATCH_SIZE} \
  --gradient_accumulation_steps ${GRADIENT_ACC} \
  --evaluation_strategy "no" \
  --save_strategy "steps" \
  --save_steps 200 \
  --save_total_limit 1 \
  --learning_rate 4e-5 \
  --weight_decay 0.01 \
  --warmup_ratio 0.03 \
  --lr_scheduler_type "cosine" \
  --logging_steps 1 \
  --max_seq_length 4096 \
  --do_train True \
  --grad_checkpoint True \
  --group_by_length True \
  --dynamic_image_size True \
  --use_thumbnail True \
  --ps_version 'v2' \
  --deepspeed "zero_stage1_config.json" \
  --report_to "tensorboard" \
  2>&1 | tee -a "${OUTPUT_DIR}/training_log.txt"
```

Figure 4: LoRa Finetune parameters

to generate a mapping of acoustic variables and its corresponding words interpretation. For instance, the *valence* variable is "A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track", a floating variable from 0 to 1.

Given this set of definition to the GPT4o model, the following valence mapping was devised. As we wanted to keep textual intermediaries consistent across generations, we decided to create the music description through a hard-coded approach. Whilst this approach is deterministic, this decision does introduce musical bias, particularly in how the music descriptions are defined.

Text-to-Music: A custom-generated prompt will then used for music generation with MusicGen.

The last step in our pipeline was to integrate our pipeline with the MusicGen model. The HuggingFace musicgen-small model implementation was used in this section, and the original pretrained weights was used.

Connecting all of the components, we were able to successfully develop a novel pipeline that is able to generate music from textual intermediaries at zero-shot and few-shot (through RAG). To improve the interaction between the model and user, a simple chat bot version was created under inference/musicgenchat.py, in which upon initialization, the user is prompted to enter a .jpg image in the test images folder, and the resulting music prompt

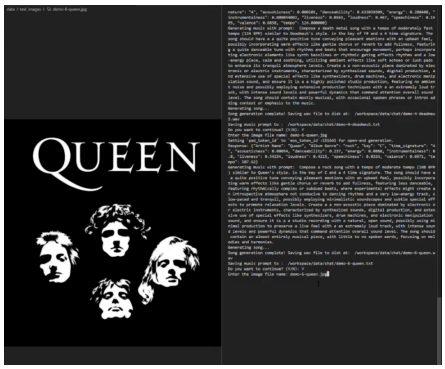

Figure 5: Example of generation chat

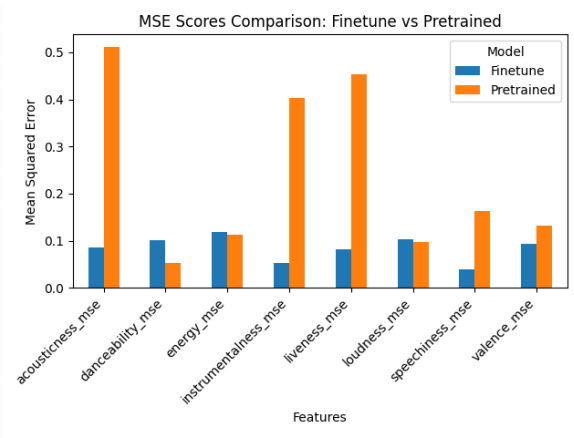

Figure 6: MSE Comparison

and music is then generated in one forward pass. An demo video of this instance was recorded, and the Figure below shows this demo in operation.

Based on the pipeline, a subset of the test images were generates, with corresponding prompts saved for evaluation. The corresponding images and audio files are in our GitHub repository (https://github.com/JosephLaiCY/6000N-Multi-modality-LLM).

Evaluation

3.5.1 Evaluation textual intermediary consistency

Using the same prompt above, the pretrained and VLM outputs were evaluated on the test data and using the same MSE metric used in the RAG evaluation section above. We adopted to use a strict cut-off approach, in which any output that does not produce a valid JSON format will be assigned a Null score. We used this as a metric evaluating intermediary consistency. Amongst the 107 test-images, a LoRA adaptor successfully generated 75 valid outputs, compared to the 21 valid outputs the pretrained model generated. This suggested that the LoRa adaptor increases the consistency of generating a codified JSON object for further processing in the pipeline. Since the LoRa adaptor was only trained on a small dataset, this large increase does suggest that further data could potentially further improve the model's generational consistency.

3.5.2 Music Compatibility to prompt evaluation

Both visual and audio inspection was then conducted to explore the generated results, along with it intermediary outputs. From our initial demonstration, we generated music using the album cover of "Forcefield" by Canadian rock band Tokyo Police Club. Based on the image, two music variations

Figure 7: Tokyo Poluce Club album image

were generated - one from zero-shot inference by the LoRa trained VLM, and another one supported by RAG.

Based on the test album image, the models generated the two corresponding musical prompts that were then fed into MusicGen seen below. The resulting music can be found in our open sourced repository. Inspecting the resulting outputs, it can be seen that at both zero-shot and few-shot, both methods were able to identify the album to be a rock genre, potentially due to the pretrained VLM model being exposed to this album before. Both RAG and zero-shot models generated acceptable music quality, with prompt descriptions seen in the Figures below.

However, there were some scenarios in which the VLM pipeline tended to hallucinate and generate erroneous music prompts that led to noticeably un-matched music for the album. For example, in another music generation evaluation, music was generated using the "Bo Bice 3" music album, which features country-rock artist Bo Bice. In this

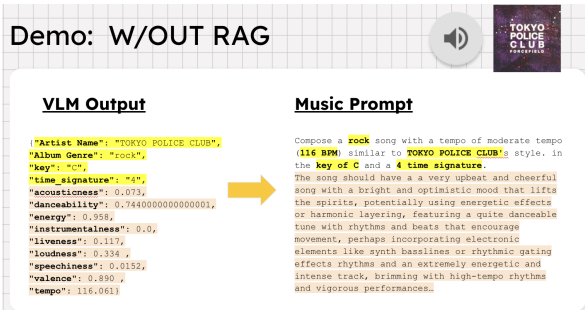

Figure 8: Zero-shot prompt generation of Tokyo Police Club

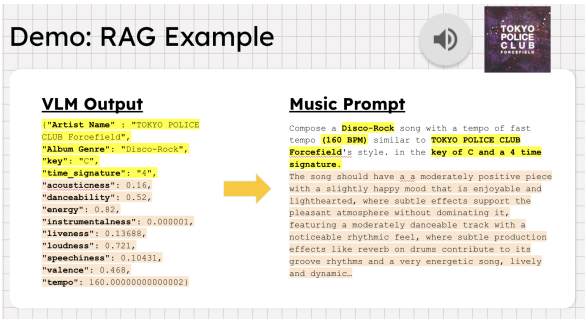

Figure 9: Few-shot prompt generation of Tokyo Police Club

generation, both the zero-shot VLM and the RAG-enabled VLM were able to identify the artist as "Bo Bice", but the genre predictions had variations. Specifically, the RAG model hallucinated and resulted in a prediction of a genre "male," which is an invalid music genre.

In the above cases, while a image-to-text generation provides an explainable JSON format for us to debug and understand the nature of the model, how the resulting text-to-music generation produce a valid ouput is relatively subject. Hence, to holistically evaluate and quantify the effectiveness of

Figure 10: Bo Bice 3 Music Album

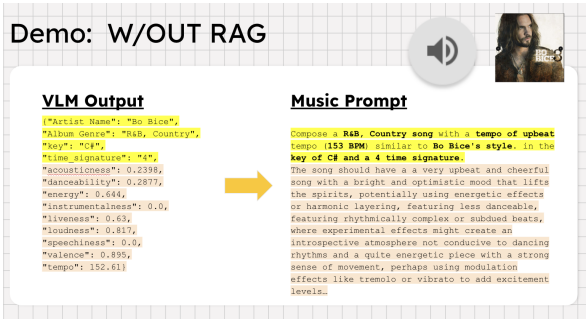

Figure 11: Zero-shot prompt generation of Bo Bice

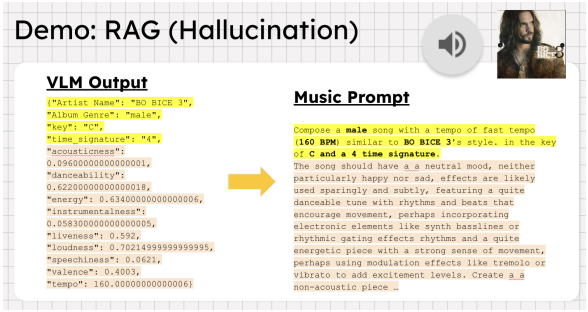

Figure 12: Few-shot prompt generation of Bo Bice 3 with RAG

a model, a further study with human labelers will need to be conducted. To evaluate the musicality and the validity of the generated musical results, we propose a study inspired by Reinforcement Learning with Human Feedback (RLHF), a technique used during ChatGPT training. RLHF enables models to better align their outputs with human preferences by incorporating feedback from human evaluators into the training process.

In this context, we envision a process where human labelers evaluate image-audio pairs generated by the model. For each pair, labelers would assess criteria like musicality (the quality and structure of the music) and compatibility (how well the music aligns with the mood, theme, and aesthetics of the album cover). These evaluations would form the basis for training a reward model, which predicts the quality of the image-audio pairs.

The reward model would then guide the base generative model—responsible for creating audio descriptions or music tokens—through reinforcement learning. The model would iteratively refine its outputs to maximize alignment with human preferences. Variants of music for each album could be generated by altering the seed inputs to the model, providing diverse outputs for evaluation. The RLHF process would involve multiple iterations, with labelers scoring and ranking outputs at

each stage to fine-tune the model further.

In preparation for such studies, we have generated image-audio pairs in our GitHub repository for evaluation. This dataset will serve as a foundation for human labeling and subsequent model improvements. The insights gained from this RLHF-inspired process will enhance the model's ability to generate high-quality, contextually appropriate music tailored to specific album cover art.

Limitations & Future Work

We proposed the following areas for future exploration. Firstly, given the large trainable parameter size of the LoRa adaptor, more training data is needed to be conducted as it is likely overfitting to the dataset at the moment. As future explorations, a larger set of training data should be used to enhance the VLM's accuracy in predicting acoustic features. Secondly , how a music description can be mapped remains largely unexplored. In our implementation, we adopted a naïve approach by hardcoding the music descriptions generated by a LLM. A future area for exploration would be to bypass this phase and directly train the LoRa adaptor to the music descriptor embedding. Lastly, how to evaluate the musicality of a music is subjective and requires further studies to better assess the effectiveness of music generated through this approach.

Conclusion

Overall, our work demonstrated the ability to generate music from image through controllable, explainable textual intermediaries. We conducted a series of explorations ranging from conducting RAG evaluations, to use of LoRA finetuning, and finally the creation of a novel pipeline with capabilities of generating explainable music in zero-shot.

References

Ioannou, Markos. (2020). Looking at Sound: Contextualising Album Cover Multimodality. 10.13140. RG.2.2.32809.62565.

Oord, A. V. D. (2016). WaveNet: A Generative Model for Raw Audio. *arXiv preprint arXiv:1609.03499*.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., ... & McGrew, B. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Défossez, A., Copet, J., Synnaeve, G., & Adi, Y. (2022). High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*.

Copet, J., Kreuk, F., Gat, I., Remez, T., Kant, D., Synnaeve, G., ... & Défossez, A. (2024). Simple and controllable music generation. *Advances in Neural Information Processing Systems*, *36*.

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., & Sutskever, I. (2020). Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*.

Silva, M.O., Rocha, L.M., & Moro, M. MusicOSet : An Enhanced Open Dataset for Music Data Mining.