Danmarks Tekniske Universitet

Music-to-Image Synthesis: Controlling Image Generation through Audio Modality

Jakob Kristensen (s173174)

Master's thesis in
Mathematical Modelling and Computation
Technical University of Denmark
DTU Compute
December 30, 2024

Abstract

Multimodal image generation has seen remarkable advancements in recent years, particularly in text-to-image and image-to-image synthesis. However, audio-to-image synthesis remains relatively underexplored, with the subfield of music-to-image (M2I) synthesis being especially neglected. Generating images from audio is challenging due to the fundamental modality gap between auditory (temporal) and visual (spatial) stimuli, and M2I synthesis adds even more complexity through its musical associations. The inherently subjective process of matching music with images is difficult to quantify and model due to emotional, cultural, and individual preferences. Progress in the field is also hindered by the lack of large-scale datasets and versatile M2I frameworks. This thesis aims to address these gaps by developing a robust M2I pipeline capable of generating contextually and emotionally relevant images using only music as input. Achieving this required solutions to 3 main challenges. Firstly, *quantifying how humans pair music with images*: This was achieved through the novel VAG-framework, which quantifies the relationship between music and images based on valence (V), arousal (A), and genre (G). The framework captures emotional and contextual aspects of M2I matching in a straightforward and scalable way. Human experiments conducted with 36 participants validated the framework's ability to capture nuanced connections between audio and images. Secondly, *creating a large-scale dataset for M2I tasks*: Using the proposed VAG-framework, I constructed the BEATS (Bridging Emotions and Art through Sound) datasets. Combined, these include approximately 400,000 song-prompt pairs and 15,000 song-image pairs. The pairs are specifically aligned with human preferences, making them suitable for M2I tasks. BEATS addresses the lack of datasets and provide a foundation for a diverse set of M2I synthesis tasks. Thirdly, *designing, training, and deploying an end-to-end M2I pipeline*: I implemented a custom audio encoder and aligned it with CLIP, enabling M2I synthesis within a diffusion-based model. In addition to these 3 main contributions, I developed a labeling platform for conducting human-centered experiments and a multi-modular exploration GUI for interactive data analysis. In conclusion, the proposed M2I pipeline successfully generates visually coherent images that align with the emotional characteristics of the input music. Through a novel framework, large-scale datasets, and advanced AI models, this thesis builds a foundation for future cross-modal AI research in the field of M2I.

Contents

1 Introduction

In recent years, there has been significant progress in multimodal image generation, particularly in text-to-image (T2I) and image-to-image (I2I) synthesis. Models such as BigGAN, SDXL, MidJourney, and DALL·E excel at generating realistic and creative visuals, demonstrating AI's ability to generate multimodal content. However, audio-to-image (A2I) synthesis remains relatively underexplored, with the subfield of music-to-image (M2I) synthesis being especially neglected.

Audio and images are fundamentally different: Audio unfolds over time, while images are static and spatial. This key difference makes bridging the gap between auditory and visual stimuli tricky. Adding to this challenge are the complex and subtle ways we humans connect music to the visual domain. For instance, an upbeat tune with a fast tempo might bring to mind a joyful evening, while a slow, melancholic melody could evoke memories of a quiet and rainy afternoon. These emotions and the images we associate with them are very subjective. They are influenced by cultural backgrounds, personal experiences, and individual preferences.

These factors make it difficult to pinpoint exactly why people prefer certain images for specific songs, and without this knowledge, it is hard to create datasets suitable for M2I tasks. While it is straightforward to ask people directly about these preferences in a controlled experiment, this approach requires a lot of resources. Furthermore, relying solely on humans for labeling is impractical for creating the large datasets needed to train modern AI models. This creates a need for heuristics that use quantitative methods to capture the qualitative aspects of human preferences.

Current research in A2I synthesis often overlooks music, focusing instead on simpler sound inputs. Even methods that include music tend to address isolated sounds, while neglecting the more subjective dimensions. There is a clear need for M2I models that capture human emotions and create meaningful connections between music and visuals. Developing such models has the potential to improve creative industries, enhance multimedia applications, and deepen our understanding of cross-modal AI.

To summarize, progress in the M2I synthesis field is currently held back by 3 main factors: (1) scalable, quantitative methods for capturing human preferences when pairing music with images, (2) large-scale datasets that address the subjective aspects of auditory and visual stimuli, and (3) versatile M2I synthesis models capable of producing high-quality images that resonate with people. This thesis addresses each of these challenges directly. I introduce a novel, human-centered framework, along with large-scale datasets, and implement an end-to-end M2I synthesis pipeline.

1.1 Problem Statement

Progress in M2I synthesis is held back by several challenges. While T2I and I2I synthesis have advanced significantly, M2I generation is still largely overlooked. Current A2I methods often fail to account for the emotional and subjective nature of music, resulting in generated images that feel disconnected from the audio. The lack of large datasets mapping music to images also limits the development of better M2I models. Solving these problems is essential to move the field forward.

1.2 Research Questions

1. What human-centered heuristic can model the emotional and subjective relationship between music and imagery?

2. What approaches can be used to build a diverse and scalable dataset linking music to images with a focus on human preferences?

3. What AI architecture can generate music-related images that align with human preferences?

1.3 Thesis Overview

This thesis aims to develop a pipeline that converts raw music data into images that align emotionally and contextually with the input audio. Figure 1 provides an overview of my approach.

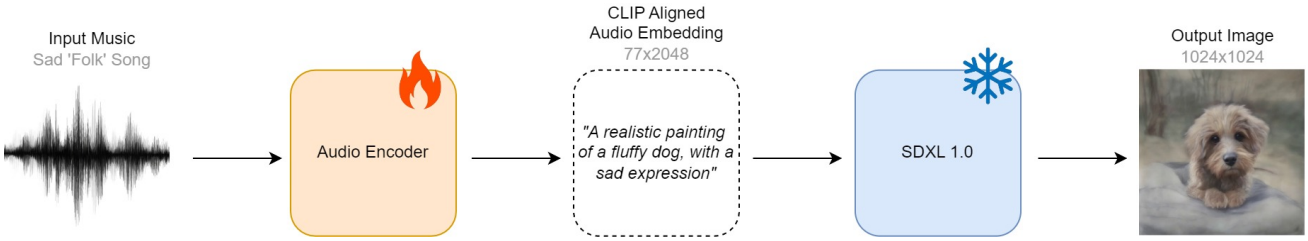

Figure 1: The framework uses a custom audio encoder aligned with CLIP embeddings to guide image generation with a frozen diffusion model. The alignment process is specifically designed to encourage a meaningful connection between the music and the image.

The thesis consists of 9 sections: *Introduction* 1 gives a general overview; *Related Work* 2 reviews M2I and related fields; *Data* 3 introduces the BEATS dataset including a novel heuristic for M2I tasks; *Music-to-Image Model* 4 explains the theoretical basis of my M2I approach; *Experimental Setup* 5 outlines implementation details; *Preliminary Studies* 6 highlights empirical insights; *Results* 7 presents model performance; and *Discussion* 8 and *Conclusion* 9 summarize findings and contributions.

1.4 Main Contributions

Below is a summary of the main contributions of this thesis.

Human-centered heuristic:
Developed a novel heuristic framework for generating soft labels to capture relationships between music and images, enabling nuanced associations between the two modalities.

BEATS
Generated datasets based on the proposed human-centered heuristic. These include 400,000 song-to-prompt pairs and 15,000 song-to-image pairs, both containing positive and negative examples suitable for contrastive learning tasks.

Aligned audio encoder:
Designed and implemented a music-to-CLIP model by aligning an audio encoder with CLIP embeddings for integration with SDXL.

Complete M2I pipeline:
Developed a fully functional A2I generation model capable of producing visually coherent and contextually relevant imagery from musical inputs.

Interactive exploration GUI:
Developed an interactive GUI for efficiently exploring audio, images, text, categories, and spatial relationships.

Labeling platform:
Developed a labeling platform (self-hosted website) that enables participants to listen to songs, select images, and indicate preferences based on their auditory experience.

2 Related Work

M2I synthesis is a relatively underexplored area with limited prior work to build upon. Research in this field touches on a very broad range of disciplines including machine learning, generative AI, natural language processing, signal processing, musical knowledge, human biology, and human psychology. While not all of these fields are necessarily required for any given M2I task, their influence highlight the interdisciplinary nature of the problem. Additionally, paring images with music is a highly subjective task, relying on human perception and individual emotions. This makes it challenging to define hard M2I labels, because such pairs inherently exist on a spectrum rather than in clear, discrete buckets. This often means that human experimentation is more reliable than purely theoretical approaches. Due to all of the above, the related work section will cover a broad range of topics, with varying levels of detail. Additionally, some of the subsections outlined below are only loosely connected, further reflecting the diverse and fragmented nature of the M2I synthesis field.

This section is organized into 7 subsections:
Music-to-Image Synthesis 2.1 provides an overview of the current state of M2I research, highlighting its challenges and gaps; *Image Synthesis Using GANs* 2.2 contains a discussion of GANs for image generation, primarily focusing on innovations important to diffusion models and M2I synthesis; *Image Synthesis Using Diffusion Models* 2.3 explores diffusion models for image generation with a focus on M2I applications; *Transformers for Cross-Modality Tasks* 2.4 examines transformer architectures and their applications in multimodal AI; *Contrastive Loss and Soft Labels* 2.5 outlines the usefulness of contrastive learning and soft labels in M2I; *Music Emotion Recognition* 2.6 explores the emotional content of music through computational and analytical methods; and *Emotion Theory in Audio-Visual Mapping* 2.7 provides the Audio-Visual content of music through human physiology and psychology.

2.1 Music-to-Image Synthesis

M2I synthesis has not seen the same research momentum as T2I [1, 2, 3, 4] and I2I [5, 6] synthesis. Related fields such as A2I generation [7, 8], A2I manipulation [9], and audio-to-video generation [10] have recently made significant progress, all while leaving M2I as a relatively underexplored area.

Recent efforts in M2I synthesis suffer from several limitations. Some systems create visuals that reflect the sentiment of music, but these visuals are often abstract and of low quality [11, 12, 13]. Ng et al. [14] produce higher-quality images that align more closely with the goals of this thesis. However, their method relies on MIDI encodings and predefined prompts, while also focusing on classical music and overlooking contemporary genres. Generative Disco [15] requires human guidance and depends on additional external inputs. One of the more promising models with M2I capabilities is AnyGPT [16]. Although not specialized in M2I, AnyGPT is capable of generating images from music using a multimodal token-based transformer framework. However, the model is not optimized for emotional alignment between the music and generated visuals. Furthermore, AnyGPT is a large and complex system designed for general multimodal tasks, making it less efficient and potentially less good than specialized models.

Other systems exist but are beyond the scope of this thesis, I will however mentioned a couple. For instance, some proprietary systems generate visuals entirely from song lyrics, making them unsuitable for instrumental pieces or songs with misleading or missing lyrics [17]. Using models like MusicBERT [18] to predict musical features for use in an image retrieval system is another approach. Additionally, some methods modify existing images based on music [19]. However, this study focuses specifically on image synthesis systems that work directly from raw music input. As a side note, while not directly applicable to M2I synthesis, combining text-to-music models [20], image-to-text models [21], and text-to-image models could help in the creating of M2I datasets. To my knowledge, there is no suitable M2I dataset that pairs music with images focusing on human preferences.

In conclusion, M2I synthesis is a underexplored and challenging field with significant room for advancement. Existing methods often produce low-quality or abstract images, rely on limited musical genres, or require additional inputs other than the raw audio. Models like AnyGPT demonstrate potential by enabling image generation directly from music. However, its general-purpose design and

lack of emotional alignment highlight the need for more specialized solutions. Developing specialized M2I systems focusing on human preferences and genre diversity could advance the M2I field.

2.2 Image Synthesis Using GANs

Generative models have revolutionized image synthesis and generative adversarial networks [22] (GANs) have played a significant role. At their core, GANs consist of 2 networks: A generator (G) and a discriminator (D). G and D are trained simultaneously in a minimax game. The adversarial nature of these two networks enable the creation of realistic images from random noise. Initially, GANs were not directly controllable through conditional inputs. Later models such as Conditional GANs (cGANs) [23] incorporated additional information (e.g. class labels) to guide the image generation process, thus allowing for more controlled image synthesis.

Variational Autoencoders (VAEs) [24] use a probabilistic approach to encode data into a latent space and decode it to reconstruct the original input—functioning much like a lossy compression and decompression system, inherently introducing a trade-off between compression and fidelity. Unlike traditional GANs, VAEs focus on learning latent representations of the data, enabling way more efficient modeling of the data's underlying distribution. However, standalone VAEs often produce blurrier images partially due to the nature of their pixel-wise reconstruction loss. Hybrid models such as VAE-GANs [25] seek to combine the advantages of VAEs and GANs, by leveraging VAEs for efficient encoding and GANs for generating high-quality images. This approach retains the computational efficiency of operating in a lower-dimensional latent space (i.e. not in pixel space directly) while addressing the image quality issues of standalone VAEs.

Building on these foundational innovations, advancements in GAN architectures have pushed the boundaries of image synthesis. For example, Brock et al. [26] demonstrated the effectiveness of training GANs at scale with BigGAN. They showed that enhanced image fidelity and diversity could be archive by using larger GAN architectures and by training on very large datasets. Meanwhile, CycleGAN [5] enabled unpaired I2I translation, opening doors for applications like style transfer. StyleGAN [27] further advanced the field by introducing style-based generators that provided control over certain features during image synthesis.

Progress in A2I synthesis focuses on translating audio features into imagery, often using GANs. Early works, such as those by Wan et al. [28] and Fanzeres et al. [29], used GANs for specific domains like musical instruments or simple sound categories. More recent methods like Sound2Scene [7], have successfully used GANs trained on aligned audio-visual spaces to generate high-quality images from audio alone.

2.3 Image Synthesis Using Diffusion Models

Diffusion models have emerged as a powerful class of generative models for image synthesis. Originally introduced by Sohl-Dickstein et al. [30], diffusion probabilistic models consist of a forward and a reverse process. The forward process gradually adds Gaussian noise to the data, transforming it into pure noise. While the reverse process learns to denoise the data step by step, reconstructing it from the noise. Ho et al. [31] refine this process with Denoising Diffusion Probabilistic Models (DDPMs). They simplified the training objective and demonstrated that diffusion models could achieve impressive image generation results (in certain cases comparable to or even surpassing the results of SOTA GANs). Further advancements by Nichol and Dhariwal [32] introduced architectural and training improvements, resulting in faster convergence and higher-quality image synthesis. The same authors showed that diffusion models were able to outperform GANs on image synthesis benchmarks, and also introduced techniques like classifier guidance to enhance sample quality [33]. To simplify conditional image generation, Ho and Salimans [34] proposed classifier-free diffusion guidance. This eliminated the need for a separate classifier by essentially training the diffusion model with and without conditioning.

Building upon these advancements, Rombach et al. [1] introduced the highly influential Latent Diffusion Models (LDMs). This would later give rise to models like Stable Diffusion. LDMs operate in a compressed latent space instead of the high-dimensional pixel space, significantly reducing computational

requirements while preserving high-quality image generation. They utilize a pretrained autoencoder to map images to and from the latent space where the diffusion process is carried out. Cross-attention mechanism handle various conditioning inputs, thus enabling applications such as T2I generation.

For cross-modal conditioning, frameworks like CLIP [21], ControlNet [6], and AudioCLIP [35] theoretically allow LDMs to generate images guided by text, image, and audio information. This is particularly relevant for M2I synthesis, where the ability to simultaneously use text, images, and auditory features as conditioning inputs to e.g. a stable diffusion model is very valuable.

Recent advancements have explored audio-driven image generation and editing using e.g. AudioToken [8], Align, Adapt, and Inject (AAI) [9], and SonicDiffusion [36]. AudioToken uses pre-trained audio and text encoders to align audio embeddings with a latent text space, enabling A2I generation through a audio-to-text pipeline. AAI extends this by creating a all-in-one framework for sound-guided generation, editing, and stylization. They successfully maps audio into tokens compatible with pre-trained T2I models while aligning them with multi-modal embeddings. The SonicDiffusion model adds audio-specific cross-attention layers to diffusion models. This enables the combination of audio and text conditioning to support semantically meaningful image synthesis. These methods underscores the growing role of audio as a meaningful conditioning input in diffusion-powered image generation.

In addition to foundational advancements, techniques like DreamBooth [37], Textual Inversion [38], LoRA [39], and Hypernetworks [40] significantly enhance the training, control, and fine-tuning of Diffusion models. Specifically regarding M2I, these methods could be used for efficiency, adaptation of music styles and more precise control over the general visual mood.

Recent years have seen an explosion in highly capable generative models for image synthesis. Open-source models such as Stability AI's Stable Diffusion (SD) and SDXL [2], as well as Black Forest's open source Flux models (schnell and dev, pro is proprietary), have demonstrated significant advancements. The open nature of such projects provide accessible tools for diverse applications, customization and research. Closed-source models, such as Imagen [4], DALL-E 2 [41] & 3 [42] and MidJourney, have also achieved remarkable success. These models are pushing the boundaries of photorealistic image generation, ease of use, and artistic creation in proprietary ecosystems.

2.4 Transformers for Cross-Modality Tasks

Transformers [43] have transformed deep learning by effectively capturing both short and long-range dependencies through the self-attention mechanisms. Originally developed for natural language processing (NLP), transformers is at the core of current day LLMs (large language models). By levering huge quantities of data, these LLMs excel in organizing and responding with coherent, contextually relevant information. OpenAI's ChatGPT is a prominent example of LLMs' impressive NLP capabilities. Beyond NLP, transformers have been adapted for domains like computer vision [44] and multimodal learning [45], excelling in cross-modal applications that process and generate data across different modalities [16].

The transformer architecture's lack of modality-specific biases (e.g. recurrence in RNNs for text or spatial grids in CNNs for images) allows for flexible adaptation to different input types. This adaptability makes transformers particularly appealing and well-suited for aligning different modalities. A seminal work in cross-modal transformers is CLIP (Contrastive Language-Image Pre-training) [21]. CLIP uses a dual-encoder architecture where one transformer model encode images and another encodes text. The image and text embeddings are then projected into a shared latent space. Trained on image-text pairs using contrastive learning, CLIP effectively links textual descriptions together with corresponding image. This enables e.g. zero-shot image classification and retrieval without task-specific fine-tuning.

Building on CLIP's architecture, CLAP (Contrastive Language-Audio Pre-training) [46] applies a similar approach to audio and text. By training on audio-text pairs, CLAP learns a shared embedding space where audio clips and their textual descriptions align closely. This enable cross-modal retrieval tasks like using natural language queries to find relevant audio clips or generating textual descriptions from audio inputs.

In M2I synthesis, the usage of mature cross-modal transformer models like CLAP and CLIP are particularly compelling. By encoding music features into embeddings within the same latent space as text and images, it becomes feasible to generate visual content that capture the semantic and emotional

characteristics of the auditory input. For instance, AudioCLIP [35] builds on CLIP to jointly process audio, text, and images. Theoretically enabling tasks like audio-guided image generation by aligning information across modalities.

2.5 Contrastive Loss and Soft Labels

Recent advancement in A2I can be partly attributed to the effective use of contrastive learning techniques, which help reduce the gap between audio and visual media. Contrastive loss functions, such as InfoNCE [47], work by aligning paired audio-visual embeddings while simultaneously separating unrelated pairs. Compared to more naive approaches like L2-loss (which do not consider unpaired samples) contrastive loss provide more stable training and improved image quality [7]. Furthermore, contrastive learning is well-suited for the integrating of soft labels (e.g. non-binary similarities like "somewhat similar" or "highly dissimilar") through probabilistic adjustments or weighting. Methods such as temperature-scaled softmax enable more nuanced handling of similarity scores [47, 48]. The ability to incorporate soft labels is particularly valuable in subjective domains, such as M2I synthesis, where labels often lie on a very hard-to-define spectrum rather than in well-defined buckets.

2.6 Music Emotion Recognition

Music Emotion Recognition (MER) is an important area within Music Information Retrieval (MIR). MER aims to automatically identify and analysis the emotional content conveyed by music. By analyzing audio signals, MER-systems attempt to predict the emotions that music may evoke in listeners. This insight can power applications such as personalized music recommendation, affective computing, and multimedia content generation.

Like many of the earlier machine learning (ML) approaches, MER systems initially relied on handcrafted features. These features were primarily derived from musical elements such as melody, harmony, rhythm, and timbre, which were then used in ML algorithms to classify emotions [49]. The adoption of dimensional emotion models, such as the valence-arousal (VA) framework [50], provided a convenient and continuous representation of emotional states. Using such models allows for nuanced emotion transitions (e.g. from calmness to excitement) rather than relying on less flexible emotion categories (e.g. happy or sad).

With the advancement of deep learning, MER has seen significant improvements [51]. Convolutional Neural Networks (CNNs) have been employed to learn hierarchical feature representations directly from audio spectrograms [52]. Yang et al. [53] provided a comprehensive review of MER methodologies, emphasizing the importance of using a combination of music features, such as rhythm and timbre, alongside emotion annotations to model the emotional characteristics of music. Their work highlights important of both categorical (e.g. music genre) and dimensional emotion models (e.g. VA) for reliable emotion mapping.

2.7 Emotion Theory in Audio-Visual Mapping

There exists many psychological models for understanding and categorizing emotions. In combination, these models provide a framework for selecting images that aligns well with auditory stimuli[1].

Paul Ekman identified 6 universal emotions: *happiness*, *sadness*, *fear*, *anger*, *surprise*, and *disgust* [54]. Accodring to his work, these emotions are biologically hardwired and recognized across cultures. Theoretically, this provides a straightforward categorization of songs into discrete emotional buckets. For example, a happy song could evoke cheerful imagery, while a sad song might correspond to somber visuals [55].

[1]Please note that emotion theory is outside my main area of expertise. My motivation for researching it was to gain inspiration rather than achieve a comprehensive understanding of the field. I feel this approach is defensible as my application of emotion theory is empirical—Meaning, I will validate my understanding through human experimentation rather than rely solely on theoretical interpretations. Furthermore, my understanding is partially derived from second-hand sources (blogs, video lectures, etc.), but I have, to the best of my ability, tried to provide accurate citations.

Plutchik expanded on these basic emotions, introducing 8 primary emotions: *joy*, *trust*, *fear*, *anger*, *surprise*, *sadness*, *disgust* and *anticipation* [56]. He organized these in a wheel to represent varying intensities and combinations, such as *joy* intensifying to *ecstasy* or *joy* and *anticipation* combining into *optimism*. This model better reflects the gradual, nuanced nature of human emotions, in contrast to binary or categorical theories.

In selecting visuals for music, attributes such as color plays a crucial role in evoking and potentially aligning emotions. Warm colors (e.g. red) are linked to arousal and excitement, while cooler tones (e.g. blue) convey calmness. Research shows that hue, saturation and brightness influence emotional responses [57]. For instance, higher saturation and brightness have a tendency to increase arousal. These insights complement Russell's Circumplex Model of Affect [50] very well, theoretically enabling the alignment of color with the overall emotional tone of music.

The Cue Redundancy Model (CRM) explains emotional responses to music through universal auditory cues (e.g., tempo, loudness) and culturally specific cues (e.g., learned associations with scales or modes) [58], while the BRECVEMA model highlights diverse mechanisms, from reflexive responses to memory-based association [59]. Studies on synesthesia[2] further emphasize cross-modal associations, showing how sound attributes like pitch or tempo spontaneously evoke imagery or colors tied to emotional states [60].

[2]A condition where stimulation of one sensory pathway triggers experiences in another. People with synesthesia may, for instance, experience multimodal sensations such as "seeing sounds" or "hearing colors".

3 Data

Building useful M2I synthesis models require data that links songs and images in a meaningful way across a diverse range of musical styles. As discussed in 2, such a dataset should balance emotional depth, audio-visual resonance, and musical features. To address these needs, I introduce a novel dataset called **BEATS** (Bridging Emotions and Art Through Sound). BEATS combine music metadata, audio, AI-generated prompts, and AI-generated images in a way that ensures cohesion across modalities. The creation of this dataset was an iterative process with exploratory analysis, interactive tools, human-inspired heuristics, M2I ablation studies, custom label platforms, and human feedback. The result is a large dataset well-suited for a variety of M2I tasks.

This section is structured in 6 subsections:
Music Metadata 3.1 outlines the selection and processing of music metadata which will form the foundation of BEATS. *Music Audio* 3.2 describes the process of obtaining audio files corresponding to the music metadata. *What Makes a Good Music-to-Image Match: A Hypothesis* 3.3 provides a theoretical, but practical framework for bridging music and imagery. *Human Experiment* 3.4 details the experimental design, implementation, analysis, and validation of a novel human-centered M2I heuristic. *Music-to-Image Similarity Function* 3.5 presents a scalable, well-defined approach to quantify the relationship between music and imagery. *BEATS: Bridging Emotions and Art through Sound* 3.6 summarizes this final datasets.

3.1 Music Metadata

To build the BEATS dataset, I first needed a music dataset containing both audio (waveforms) and metadata (song names, genres, etc.). Spotify offers detailed metadata on music, including subjective features (valence, danceability, etc.) and objective features (loudness, artist name, etc.) making it an ideal source to built a dataset upon. However, directly using Spotify's API was not doable because of restricted access and a recent shift in genre tagging from track to artist level. I therefor decided to use an existing dataset, the Spotify Tracks Dataset [61]. This dataset is recent enough to remain relevant, but old enough to contain musical genres for each song. This dataset contains ~114,000 entries (~90,000 unique songs by track IDs). A comprehensive outline of the features [3] are detailed in Table 1. The cleaned dataset contains ~ 40,000 unique songs and will be referred to as the *Spotify Dataset* from here on.

Table 1: Description of Spotify dataset columns

Column Name	Description	Column Name	Description
track_id	Spotify track ID.	**artists**	Names of track artists.
album_name	Album the track appears in.	**track_name**	Name of the track.
popularity	0-100, play count and recency.	**duration_ms**	Track length in milliseconds.
explicit	Explicit lyrics: true or false.	**danceability**	Suitability for dancing (0-1).
energy	Intensity/activity level (0-1).	**key**	Musical key (0=C, 1=C#, ...).
loudness	Loudness in decibels.	**mode**	Modality: 1=major, 0=minor.
speechiness	Spoken word presence (0-1).	**acousticness**	Conf. track is acoustic (0-1).
instrumental	Likelihood of no vocals (0-1).	**liveness**	Likelihood track is live (0-1).
valence	Positivity (0-1)	**tempo**	Tempo in beats per minute.
signature	Beats per bar (3 to 7).	**track_genre**	Genre of the track.

[3]Note that Spotify's musical features will be accepted as-is. Meaning i will trust their integrity despite not knowing how they were calculated. Features such as danceability, energy, valence, etc. are not publicly documented, and their methodologies are in general very opaque. While these features appear to be widely used, they should probably be interpreted cautiously.

3.2 Music Audio

Since Spotify does not provide direct access to music files, I downloaded audio for songs in the Spotify Dataset from YouTube. This approach did, however, come with some challenges: (1) YouTube does not recognize Spotify track IDs directly, (2) YouTube's search results often include a mix of live performances, covers, and sometimes completely unrelated content, and (3) unofficial versions of songs can sometimes surpass original tracks in popularity. Trough trial-and-error, I developed an easy and reliable way to combat these challenges.

First, I constructed YouTube search queries in the following format:

```
youtube_search_query = "{artists} - {track_name}, official music video"
```

To filter the results, I defined the following conditions:

$$\begin{aligned} \text{is\_length\_outside\_range} &= (\text{video.minutes} < 1) \vee (\text{video.minutes} > 10) \\ \text{has\_low\_views} &= \text{video.views} < 100,000 \\ \text{is\_age\_restricted} &= \text{video.age\_restricted} \end{aligned}$$

Videos were discarded if they satisfied the condition:

$$\text{is\_length\_outside\_range} \vee \text{has\_low\_views} \vee \text{is\_age\_restricted}$$

The downloading process was resource-intensive and time-consuming, with unforeseen challenges that are not detailed here. In the end, 36,342 songs were successfully downloaded. Along with the audio, I also collected YouTube metadata, see Table 2.

Table 2: Description of YouTube dataset columns

Column Name	Description
search_query	Search query used on YouTube.
url	Link to the YouTube video.
title	Video title.
author	YouTube Channel name.
publish_date	Video publish date.
description	Video description.
length_ms	Video duration in milliseconds.
views	Number of views.
thumbnail_url	URL of the video thumbnail.
keywords	Video tags or keywords.
rating	Viewer rating.
video_id	Unique video identifier.
spotify_id	Corresponding Spotify ID.
channel_id	Unique channel identifier.
channel_url	URL of the channel.
is_age_restricted	True if age-restricted.
captions_english	English captions if available.

3.3 What Makes a Good Music-to-Image Match: A Hypothesis

To evaluate M2I mapping, I needed a dataset capable of linking music with images. Initially, this seemed straightforward: simply pair songs with suitable images or prompts. However, the task proved more complex than first expected. Humans intuitively associate music with imagery through a combination of auditory stimuli, visual impressions, and personal preferences. The inherent subjectiveness of these associations makes broad generalizations very tricky. For example, it is very easy to explain why Rammstein (a popular metal band) does not work well with an image of a cute pink cat. Most people would also agree that Beethoven works better with a calm nature scene, than with a vibrant techno festival. However, defining universal rules that precisely explain why such statements hold is challenging. Individual associations vary greatly based on factors such as age, culture, and personal experiences. Despite this subjectivity, the examples of Beethoven fitting a calm nature scene and Rammstein clashing with a cute pink cat suggest that there *is* a recognizable pattern—A sort of "objective signal" that can help shed light on M2I matching. This intuitive, yet elusive signal is what I am going to explore and attempt to quantify in this section. To avoid confusion going forward, I will refer to this "objective signal" simply as *The Link*.

3.3.1 Thought Experiment

To quantify *The Link*, I considered how to approach the problem with minimal assumptions. An unsupervised method would have been ideal, but I abandoned this idea early because I was unable to find any practical solutions. Instead, I concluded that some form of labeling (soft, hard, or something in between) would be necessary for each song. To explore this, I imagined an ideal scenario, ignoring real-world constraints.

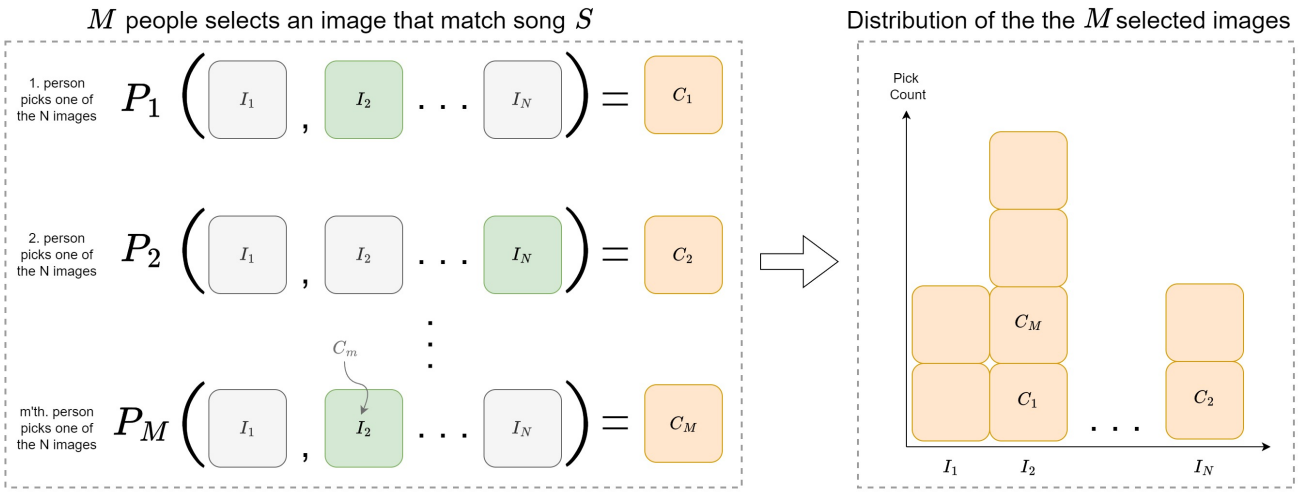

Figure 2: An idealized experimental setup for studying image-to-song preferences.

Consider Figure 2. We have a song S, a pool of N images $\mathcal{I} = \{I_1, I_2, \ldots, I_N\}$, and M human participants $\mathcal{P} = \{P_1, P_2, \ldots, P_M\}$. Each participant P_m selects an image C_m from $\mathcal{I}$ that they believe best matches the song S which gives a set of M chosen images $\{C_1, C_2, \ldots, C_M\}$. Combined, this process yields a probability function $\mathcal{D}(I \mid S\,; \mathcal{P}, \mathcal{I})$ which represents the likelihood that any given image $I \in \mathcal{I}$ is selected as the best match for the song $\mathcal{S}$ based on the decisions of all participants in $\mathcal{P}$. We get:

$$\mathcal{D}(I \mid S\,; \mathcal{P}, \mathcal{I}) = \frac{\text{Count of } I \text{ selected as a match for } S}{M}$$

Or mathematically

$$\mathcal{D}(I \mid S\,; \mathcal{P}, \mathcal{I}) = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}[C_m = I]$$

Where $\mathbf{1}[C_m = j]$ is the indicator function

$$\mathbf{1}[C_m = I] = \begin{cases} 1, & \text{if participant } P_m \text{ selects image } I \\ 0, & \text{otherwise.} \end{cases}$$

Lets explore what happens when we increase M and N. As the number of participants M increases, $\mathcal{D}(\cdot)$ converge toward the population mean. Similarly, increasing the number of images N increases the likelihood of finding images that closely match the song. In an ideal scenario where $M \to \infty$ and $N \to \infty$, we would expect $\mathcal{D}(\cdot)$ to perfectly describe the average opinions of all people.

Although the exact properties of $\mathcal{D}(\cdot)$ are unknown, we can imagine two extremes: (1) a perfectly uniform distribution where each person's preferences are basically random, (2) a distribution where everyone selects the exact same image every time. If $\mathcal{D}(\cdot)$ aligns with case (1), where individual preferences dominate and no collective pattern really exists, finding *The Link* would be nearly impossible. On the other hand, if $\mathcal{D}(\cdot)$ aligns with case (2), where a strong collective signal exists, quantifying *The Link* should be doable—At least theoretically. A plausible example of case (2) could be a rapidly decaying exponential distribution, where a few images are very popular and account for the vast majority of the distribution, while the remaining images are rarely chosen. This thought experiment will guide how I answer *What is a good music-to-image match?*

3.3.2 Approximation Attempt 1

To approximate the complex distribution of human song-to-image preferences $\mathcal{D}(.)$, I first considered creation a large database of images and presenting small random subsets (e.g. $N = 10$) to participants for labeling. However, this approach had some challenges: What images should be included in the database? How many participants would be required? Paid or voluntary labelers? How to evaluate label quality? etc.

Answering these questions while trying to balance both the pool of images and participants proved difficult. I realized that using completely random images was both impractical and wasteful. Instead, I hypothesized that certain broad categories are particularly important in M2I mapping. For example, choosing images from contrasting themes like calm vs. energetic, dark vs. bright, or nature vs. urban could accelerate the labeling process. Levering song metadata, I also considered sourcing images tailored to each song e.g. YouTube thumbnails, album covers, or AI-generated images. The motivation for this more curated approach was to help $\mathcal{D}(\cdot)$ quickly align with the average person's preferences—similar to providing a good initial guess in an optimization problem, which speeds up convergence.

Based on this idea, I created the setup shown in Table 3. Here, $N = 12$ images are divided into four broad categories: *Song Tailored*, *Setting & Scene*, *Color & Mood*, and *Random*. These images would then be randomly shuffled and presented to a human for labeling. My motivation for this was twofold: (1) to ensure that at least some images were relevant to any given song (like an initial guess in optimization), and (2) to gather insights into emotional preferences when mapping music to imagery. For instance, if a song consistently aligns with dark and gold images but not with bright and warm ones, this could reveal meaningful patterns in human preferences.

Song Tailored	**Setting & Scene**	**Color & Mood**	**Random**
Chatgpt	Old vs. Modern	Warm vs. Cold	Sample(all)
Thumbnail	Natural vs. Urban	Bright vs. Dark	Sample(all)
Genre	Quiet vs. Lively	Vibrant vs. Faded	Sample(all)

Table 3: llustration of 12 images from specific categories

I initially scraped ~20,000 images from various search engines. However, many images contained text or were of poor quality. To address this, I used text recognition software to filter out text-heavy images. Why remove text? For instance, imagine an image with the text "Bob Marley", this image could be chosen based on its text rather than its imagery, leading to *semantic* rather than *visual* M2I mapping. After cleaning, over half the images were removed, and the remaining ones still lacked consistency and quality. Due to these challenges and other considerations, I decided to shift entirely to AI-generated images, which I will explore next.

3.3.3 Approximation Attempt 2

While working on the song tailored category, see Table 3, I realized that even very naive approaches could be used to generated decent looking images. My initial idea was simple:

> *An image that embodies the spirit of {GENRE}, evoking the emotions, energy, and atmosphere that resonate deeply with its listeners.*

I used this prompt to generate images via OpenAI's DALL-E3 API, and the results were surprisingly effective given their simplicity. Although the images lacked precision, their quality suggested that Spotify metadata contained valuable information for mapping music to visuals. Encouraged by this, I set out to develop a more advanced script for generating prompts. The script relies on a large collection of predefined attributes for *style*, *general setting*, *mood*, *music genre*, and *genre-specific settings*:

```
STYLE_ADJECTIVES = [
    'photorealistic', 'abstract', 'highly-detailed', 'vintage', 'soft-focus',
        'surreal', 'oil-painting style', 'minimalist', 'cartoonish'
    ...
]

SETTING_DESCRIPTORS = {
    'high_energy': ['a bustling scene', 'an energetic environment', ...],
    'low_energy': ['a calm atmosphere', 'a serene environment', ...],
    ...
}

MOOD_DESCRIPTORS = {
    'uplifting': ['joyful', 'celebratory', 'exciting', 'positive', ...],
    'dark': ['intense', 'serious', 'moody', 'mysterious', ...],
    ...
}

GENRE_DESCRIPTORS = {
    'trip-hop': ['dark', 'experimental', 'urban', 'moody', ...],
    'afrobeat': ['rhythmic', 'energetic', 'vibrant', 'organic', ...],
    ...
}

GENRE_SETTINGS = {
    'trip-hop': ['a gritty urban landscape', 'a rainy urban night', ...],
    'afrobeat': ['a vibrant outdoor gathering', 'a lively plaza', ...],
    ...

}
```

These values are then used in together with the Spotify metadata through a combination of randomness and predefined conditional logic. Getting to this setup was an iterative and highly experimental journey. Exactly why certain values are set is entirely based on my own preferences and a few test subjects:

```
### Mood description ###
if energy > 0.8 and valence > 0.7:
    mood_type = 'uplifting'
elif energy > 0.8 and valence <= 0.7:
    mood_type = 'dark'
elif ...
mood = random.choice(MOOD_DESCRIPTORS[mood_type])

### Setting description ###
if danceability > 0.8:
    setting_type = 'high_energy' if tempo > 120 else 'low_energy'
elif acousticness > 0.6:
    setting_type = 'live' if liveness > 0.6 else 'acoustic'
elif ...
general_setting = random.choice(SETTING_DESCRIPTORS[setting_type])

### Tempo description ###
if tempo > 150 and energy > 0.7:
    tempo_description = "The scene is energetic and dynamic, full of rapid
       movement."
elif tempo > 150:
    tempo_description = "The scene has a quick tempo but maintains a calm or
       neutral energy."
elif...

### Artistic style ###
style = random.choice(STYLE_ADJECTIVES)

### Genre-specific###
genre_word = random.choice(GENRE_DESCRIPTORS[track_genre])
genre_setting = random.choice(GENRE_SETTINGS[track_genre])
```

The result of this is 6 song-tailored, dynamically assigned variables: `style`, `mood`, `genre_setting`, `general_setting`, `tempo_description`, and `genre_word`. These variables together with `track_name`, `artists`, and `track_genre` is used to construct the final prompt:

```
### Final prompt ###
prompt = (
f"Generate an image inspired by the feelings evoked by '{track_name}' "
f"by {artists}, inspired by {track_genre} music."
f"The scene is {mood}, set in {genre_setting} within {general_setting}".
f"{tempo_description}. The visual style is {style}, with elements that are {
   genre_word}. "
f"It should capture the essence of the song!"
)
```

Summarizing the entire pipeline we get:

Spotify Metadata → Prompt script → Revised prompt → DALL-E 3 → Final image

This method produced impressive results. The good results was partially due to OpenAI's ability to revise prompts before passing them to DALL-E3. While this might initially seem counterproductive, it is in fact, extremely useful. The exact mechanism by which OpenAI revises prompts is not well-documented, but an LLM is probably involved because the revised prompts often contain contextual information that is not explicitly present in the original prompt. For instance, a revised prompt about ABBA might include details about the '70s despite this not being mentioned in the original prompt, as shown in Figure 3 (Left image). It also effectively conveys a wide range of emotions, from highly energetic punk-rock, as seen in Figure 3 (Middle image), to calm and introspective moods, see Figure 3 (Right image).

Figure 3: AI-generated prompts and their corresponding images.

In conclusion, the pipeline i capable of producing visuals that match the input song. Subjective emotions, with all their nuances and inconsistencies, were often accurately captured, and the overall aesthetics were appealing in a musical sense. However, there were several challenges with this approach that ultimately led me to abandon it. Firstly, DALL-E 3 is relatively expensive to use. At the time of writing, the cost is \$0.040 per image. Considering that I aimed for 1-3 images per song, and I had a total of 36,342 songs, the total cost would range from approximately \$1,500 to \$4,400. Secondly, the fact that DALL-E 3 is not open source is a drawback from an academic perspective. Thirdly, the script I developed is very labor-intensive to write and maintain. To give a sense of the effort involved, the five hardcoded descriptors (`STYLE_ADJECTIVES`, `SETTING_DESCRIPTORS`, etc.) contains around 1,000 words each. These words must be tailored to the specific concept they describe. While generative AI can significantly speed up this process, it remains a labor-intensive and demanding task.

Despite these challenges, this approach has potential for organizations with the resources to implement and maintain it. Unfortunately, the approach is not feasible for my work. A result video featuring the 50 most popular songs across 50 different music genres is available at this URL.

3.3.4 Approximation Attempt 3

From Attempt 1, I gathered that AI-generated images were the best approach for this thesis. From Attempt 2, I learned that Spotify features could be successfully used in combination with ChatGPT and DALL-E 3. However, given that the "manual" approach in Attempt 2 was not viable, I considered whether ChatGPT could handle the entire task itself instead of simply revising pre-generated prompts. After extensive experimentation, I developed the following song-specific prompt to use in collaboration with ChatGPT:

```
INSTRUCTIONS:
I'll send you song metadata (from Spotify) like this:
{
    track_id: int,
    artists: [Artist Name(s)],
    track_name: Track Name,
    danceability: float, # 0.0 (least danceable) to 1.0 (most danceable)
    energy: float, # 0.0 (least energetic) to 1.0 (most energetic)
    key: int, 0=C, 1=C#/Db, etc.
    mode: int, 1=Major, 0=Minor
    speechiness: float, #0.0 (least spoken words) to 1.0 (most spoken)
    liveness: float, #0.0 (studio recording) to 1.0 (live performance)
    valence: float, #0.0 (sad/negative) to 1.0 (happy/positive)
    track_genre: Genre
}
Based on this, generate 10 different, concise, 2-sentence visual image prompts that
reflect the song's mood, emotion, and atmosphere. Be extremely creative with the
styling. Go crazy with different artistic approaches and aesthetics. Respond simply
with 10 lines, one for each prompt, and nothing else!
```

Figure 4 shows prompts alongside their generated images using this approach.

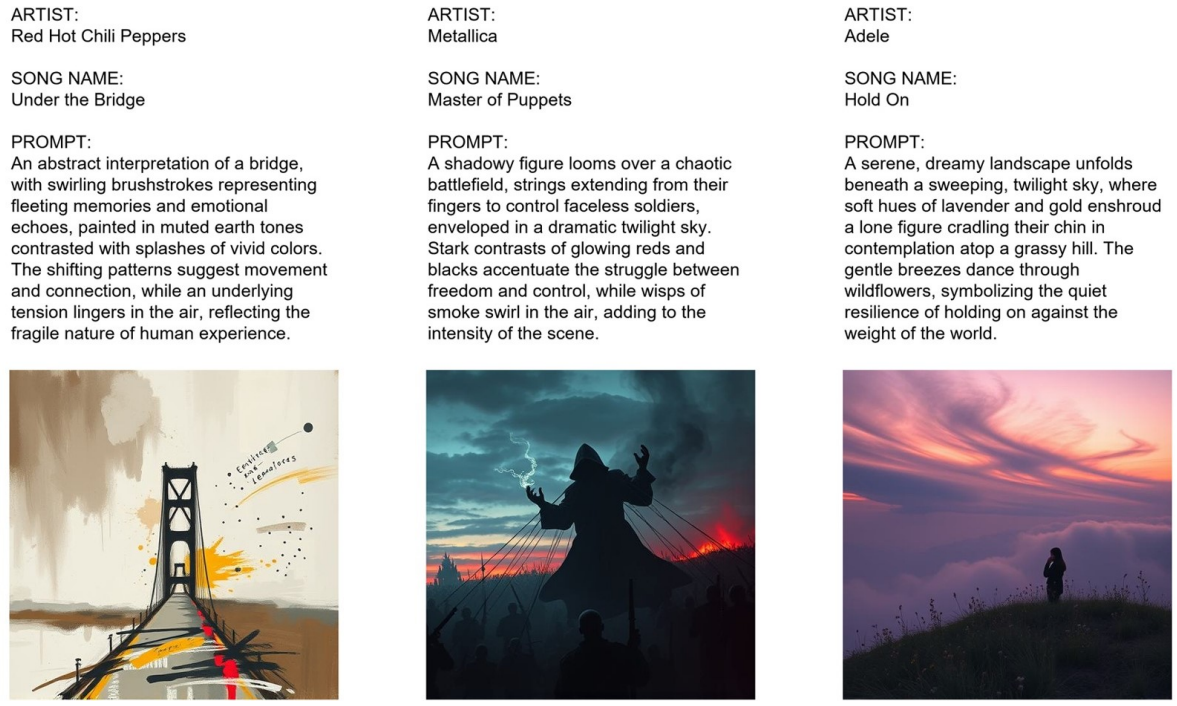

Figure 4: Song-specific AI-generated prompts and their corresponding images.

3.3.5 Interactive GUI

While working on attempts 1, 2, and 3, it became apparent that certain features were very informative for mapping music to images. For instance, I intuitively understood that music genres carry visual associations (e.g. Reggae's red, yellow, and green, or ambient music's calm nature scenes). Beyond that, the relationships between features were unclear.

To investigate these dynamics, I conducted various exploratory experiments, but this quickly became tedious. Constantly switching between modalities (songs, images, text, metadata) worked initially, but the process became a bottleneck eventually. To overcome this, I developed an interactive GUI to make exploration faster, allowing me to test hypotheses and understand relationships in an interactive manner.

The tool evolved through many iterations, with features being added, modified, and removed based on what I was testing at the time. A visualization of the final version is shown in Figure 5. In short, this GUI enabled fast, intuition-building exploration, and it was essential to the success of this thesis.

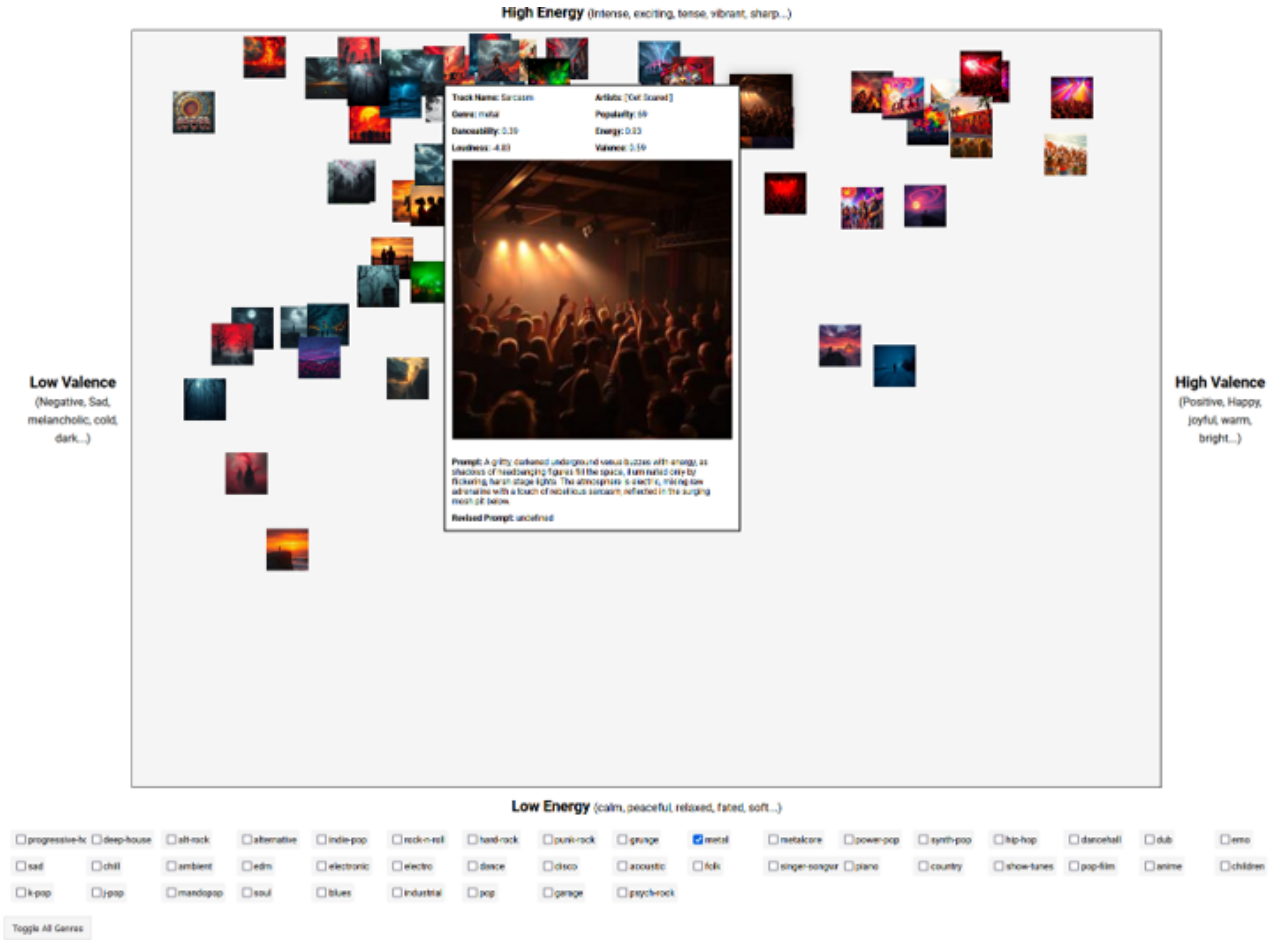

Figure 5: Interactive GUI for exploration

3.3.6 Key Findings and Final Hypothesis

To summarize, our goal is to quantify *The Link* using the idealized probability distribution $\mathcal{D}(\cdot)$. This section highlights the most important factors I found for effectively approximating $\mathcal{D}(\cdot)$. These insights are derived from my 3 approximation attempts, interactive exploration, literature review, and human pilot studies. While not exhaustive, I believe they provide a solid foundation for analytically and practically quantifying *The Link*. A comprehensive outline can be seen in Table 4.

Table 4: Key Factors Influencing Music-to-Image Mapping

Factor	**Why It is Important**	**How to Model It**	**Address**
Valence & Arousal	Explains human emotions through positivity/negativity (valence) and calm/excited (arousal).	Spotify's valence and energy scores provide practical proxies for valance and arousal respectively.	Directly
Musical Genre	Provides cultural and emotional associations, encapsulating collectively agreed-upon themes.	Spotify's genre metadata offers robust and sufficiently granular categorization.	Directly
Cultural Context	Shapes emotional connections to music and imagery through cultural understanding that resonate with people.	ChatGPT automatically injects cultural understanding into image prompts when presented with metadata.	Indirectly
Image Color	Influences emotional perception with e.g. warm tones evoking high arousal and cool tones evoking calmness.	Emotionally resonant colors emerge naturally from ChatGPT-generated image prompts.	Indirectly
Lyrics & Semantics	Adds context by conveying information not present in the music (e.g., sad-sounding songs with happy lyrics).	Not modeled to reduce the scope of the thesis, though acknowledged as important and worthwhile.	Ignored
Individual Preferences	Links personal experiences and memories to emotional connections between music and imagery.	Beyond scope due to lack of personalized information.	Ignored

From this I developed two hypotheses, collectively referred to as the *VAG hypothesis*:

- AI-generated prompts (3.3.4) can be utilized to produce images that resonate with people, for the reasons outlined in Table 4.
- The audio-visual relationship between songs can be adequately described using *valence* (V), *arousal* (A), and *genre* (G) features from Spotify Metadata (3.3.5).

I conducted pilot experiments with several participants and performed ablation studies on AI models. However, rigorous empirical evidence is still needed to validate these claims, which I will explore next.

3.4 Human experiment

To verify the VAG hypothesis outlined in 3.3.6, I needed to conduct human experiments. Designing these experiments required several things: (1) creating an experimental setup that can test the hypotheses with limited resources, (2) creating an appropriate experimentation platform, and (3) Finding a sufficiently large and representative pool of participants. This section addresses each of these 3 concerns in chronological order, followed by an analysis of the results.

3.4.1 Theoretical Design of the 4-category Experiment

I designed the following setup: For each song S, test subjects will be shown 12 images. These images are selected from 4 categories, with 3 images chosen from each category. The 4 categories are constructed around the idea of a confusion matrix with two dimensions: (1) the euclidean distance in VA-space (close or far) and (2) whether the song shares the same genre as S (same or different). The categories are defined as follows:

1. **YY**: Songs that are close to S in the VA-space and belong to the same genre as S.
2. **YN**: Songs that are close to S in the VA-space but belong to a different genre than S.
3. **NY**: Songs that are far from S in the VA-space but belong to the same genre as S.
4. **NN**: Songs that are far from S in the VA-space and belong to a different genre than S.

This confusion matrix ensures a systematic exploration of the experiment. It examines how emotional proximity (approximated as the distance in VA-space) and genre similarity affect human M2I mapping. The goal is to evaluate whether the VAG hypothesis holds under these controlled conditions. Concretely I expect:

$$\text{Pick(YY)} > \text{Pick(YN)}, \text{Pick(NY)} > \text{Pick(NN)} \quad \text{and} \quad \text{Pick(YY)} \gg \text{Pick(NN)}$$

Or said with words, images from the YY category should be selected more frequently than those from the YN and NY categories, which themselves should be chosen more often than NN images. Furthermore, I expect that YY images are chosen significantly more often than NN images.

3.4.2 Implementation of the 4-category Experiment

The practical implementation of the 4-category outlined in 3.4.1 is as follows. For a given song S, I start with its specific AI-generated image, i.e. the image generated specifically for S as described in 3.3.4. I then select 2 additional songs that are close to S on the VA-graph and from the same genre, and use their corresponding AI-generated images. This gives me 3 images in the YY category. Next, I find 3 songs from each of the remaining categories (YN, NY, NN) based on their VA and genre values and collect their corresponding AI-generated images. In total, this process results in 12 images, 3 from each category. The selection process is illustrated in Figure 6, and a concrete example can be seen in Figure 7.

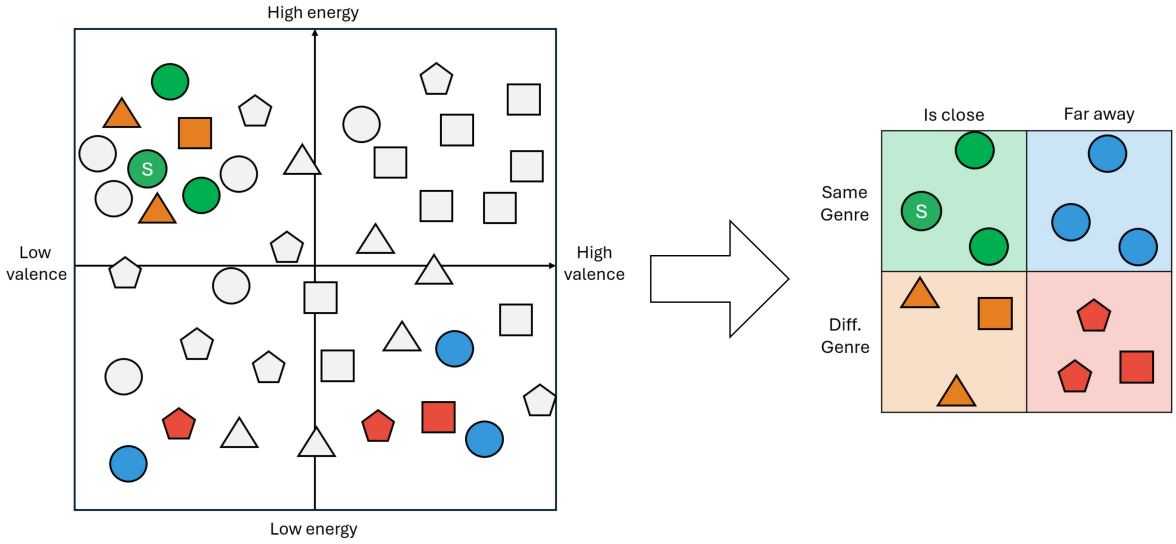

Figure 6: Abstract illustration of the 12-image, 4-category setup. The song in question, S, is depicted by a green circle labeled "S". Each shape (square, triangle, etc.) represents a song, with the shape type indicating its genre. For example, a circle could represent *Rock*, while a triangle could represent *Pop*.

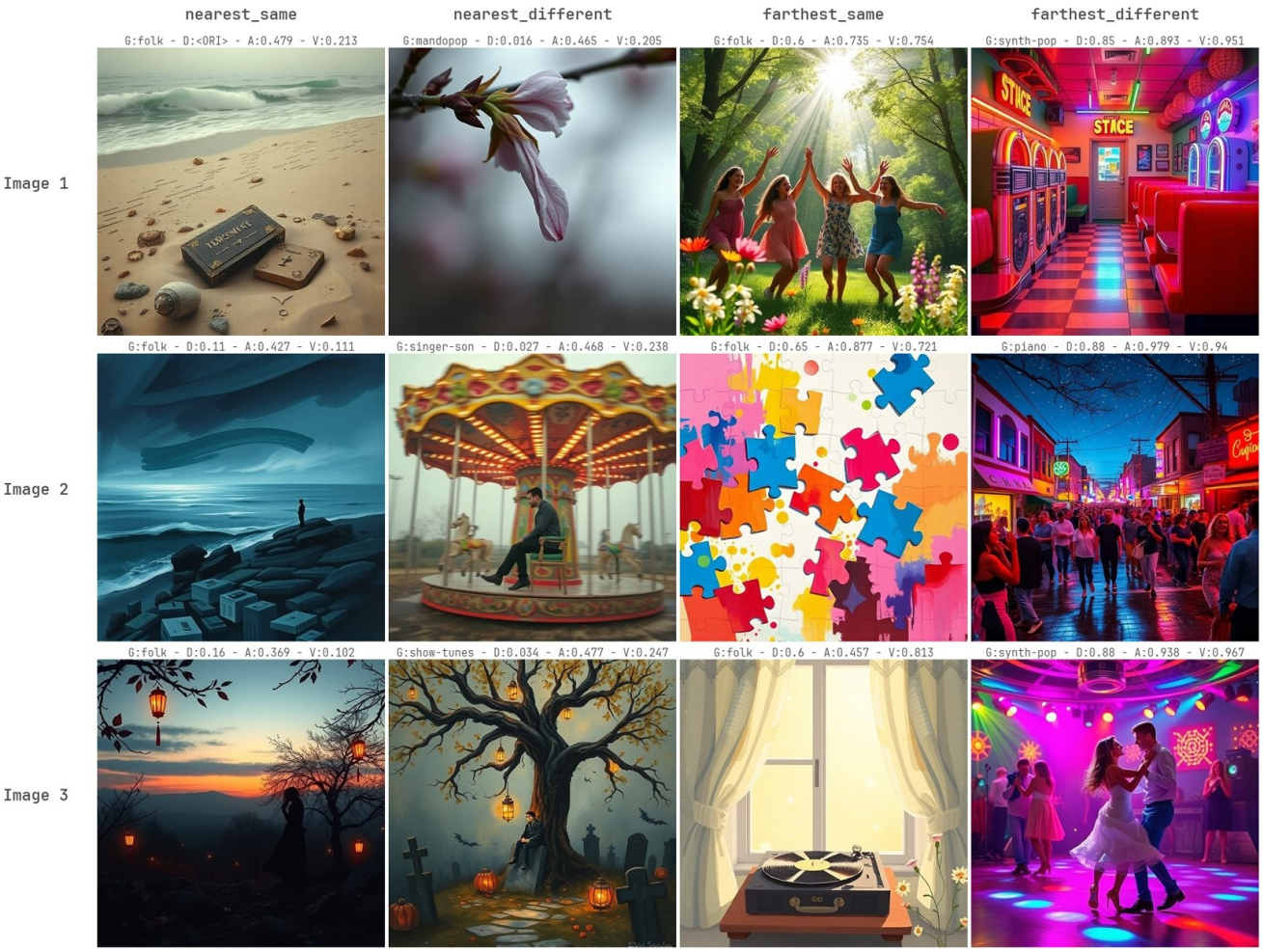

Figure 7: Illustration of the 12-image, 4-category setup.

3.4.3 Experiment Design - Labeling Platform

To conduct the experiment, I needed a platform where participants could label the images from the 4-category setup. I considered using pre-existing solutions, but found no straightforward way to adapt them to my needs. Ultimately, I decided to create a self-hosted solution. The platform is built using Python for the server-side code and vanilla JavaScript, HTML, and CSS for the front-end. The platform includes options for manual or automatic registration and a short questionnaire that users take upon their first visit, see appendix Figure 38. After completing the questionnaire, users are presented with instructions on how to use the platform. These instructions guide participants through the labeling process, see Figure 8.

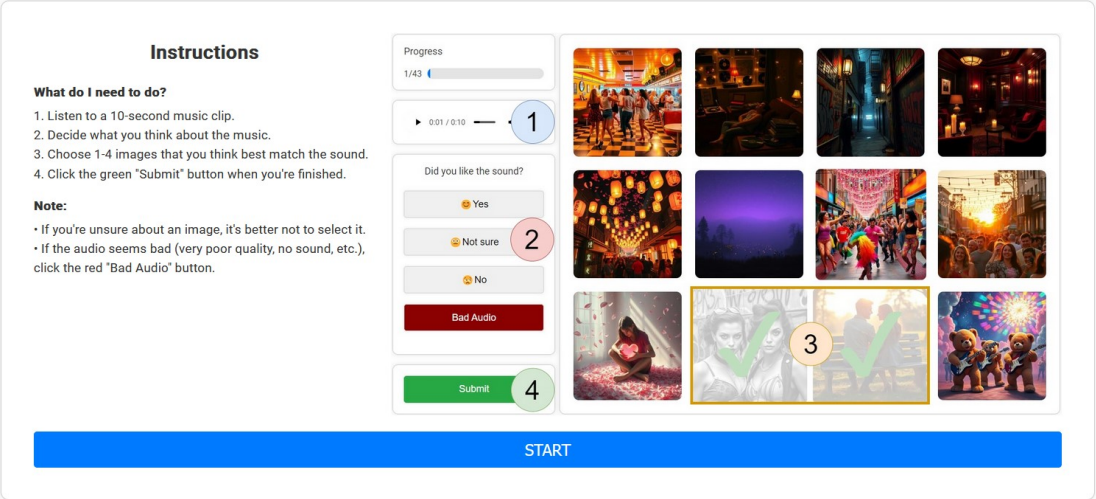

Figure 8: Instructions displayed to users.

Finally, users are directed to the labeling interface, where the actual experiment takes place. Here, (1) users listen to a song, (2) indicate whether they like it, (3) select images they believe match the song, (4) and submit their selections to move to the next song. A screenshot of the labeling interface is shown in Figure 9.

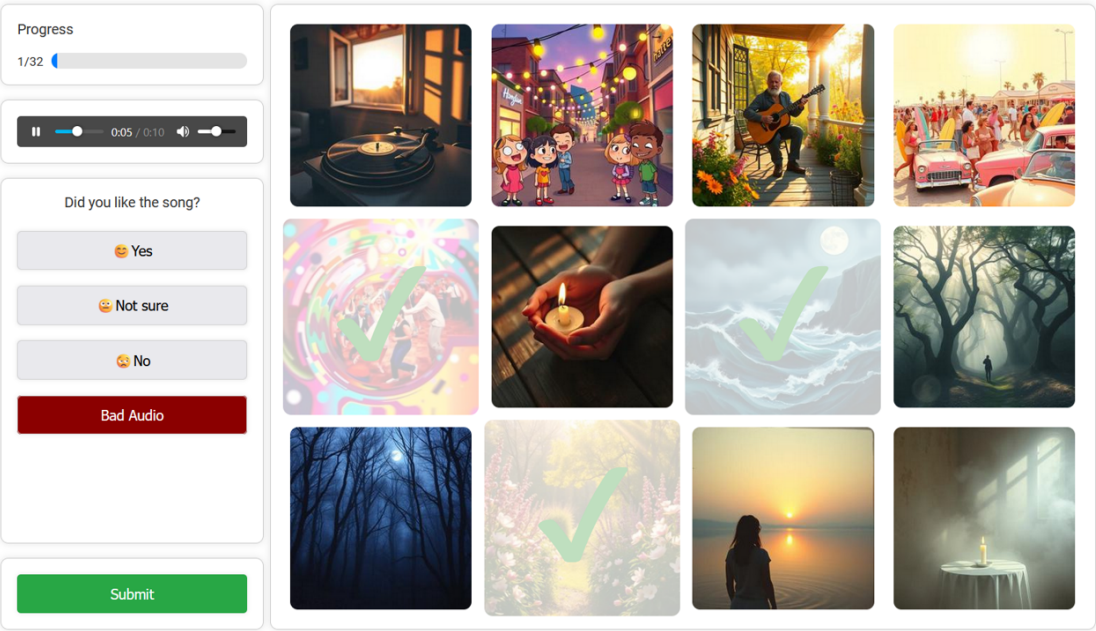

Figure 9: Labeling interface used by participants.

3.4.4 Experiment Results and Analysis

The main experiment[4] involved 32 10-second song clips. Each song was paired with 12 images selected based on the 4-category setup. These 32 songs represented the most popular tracks from 32 different musical genres. A total of 36 participants completed the study, collectively selecting 2,509 images, an average of 2.2 selected images per song.

Age and gender distribution: Participant age and gender distribution are shown in Figure 10. The participants had a male-to-female ratio of roughly 3:2 and an age range from 22 to 76, with a concentration in the 20–50 range. The study lacks representation among teenagers and children.

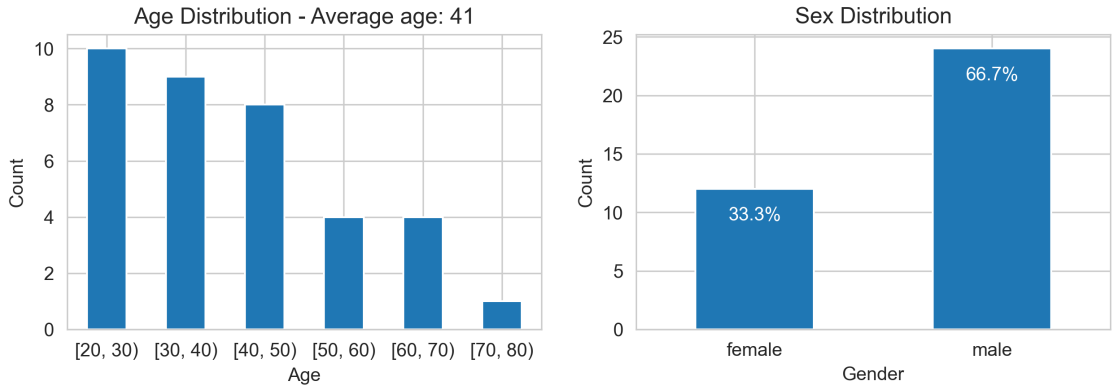

Figure 10: Age and gender distribution of participants.

Nationalities: Participants came from 11 nationalities, with a strong European bias and an even strong Danish bias, as seen in Figure 11.

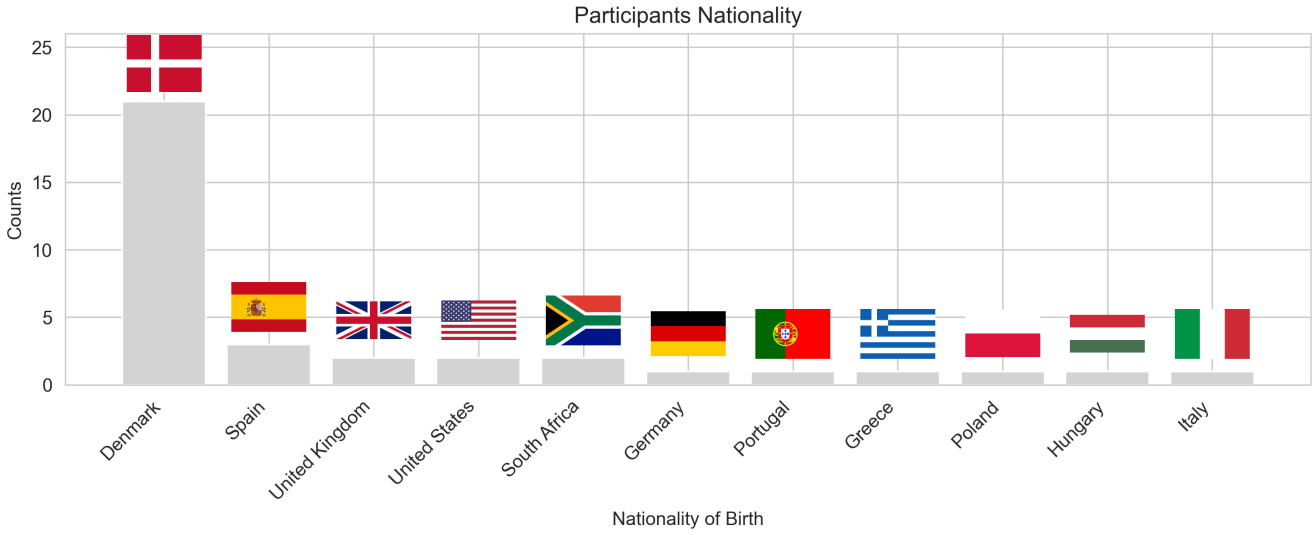

Figure 11: Participant nationalities.

[4]I conducted two experiments: A pilot study with 10 participants and a larger study (main experiment) described in this section. The pilot study is not included because the results were nearly identical to the main experiment

Song preferences: Participants generally liked the selected songs, as shown in Figure 12. But approximately 20% explicitly disliked the songs and 30% were unsure. Some variation was observed across genres, but no extreme outliers were observed.

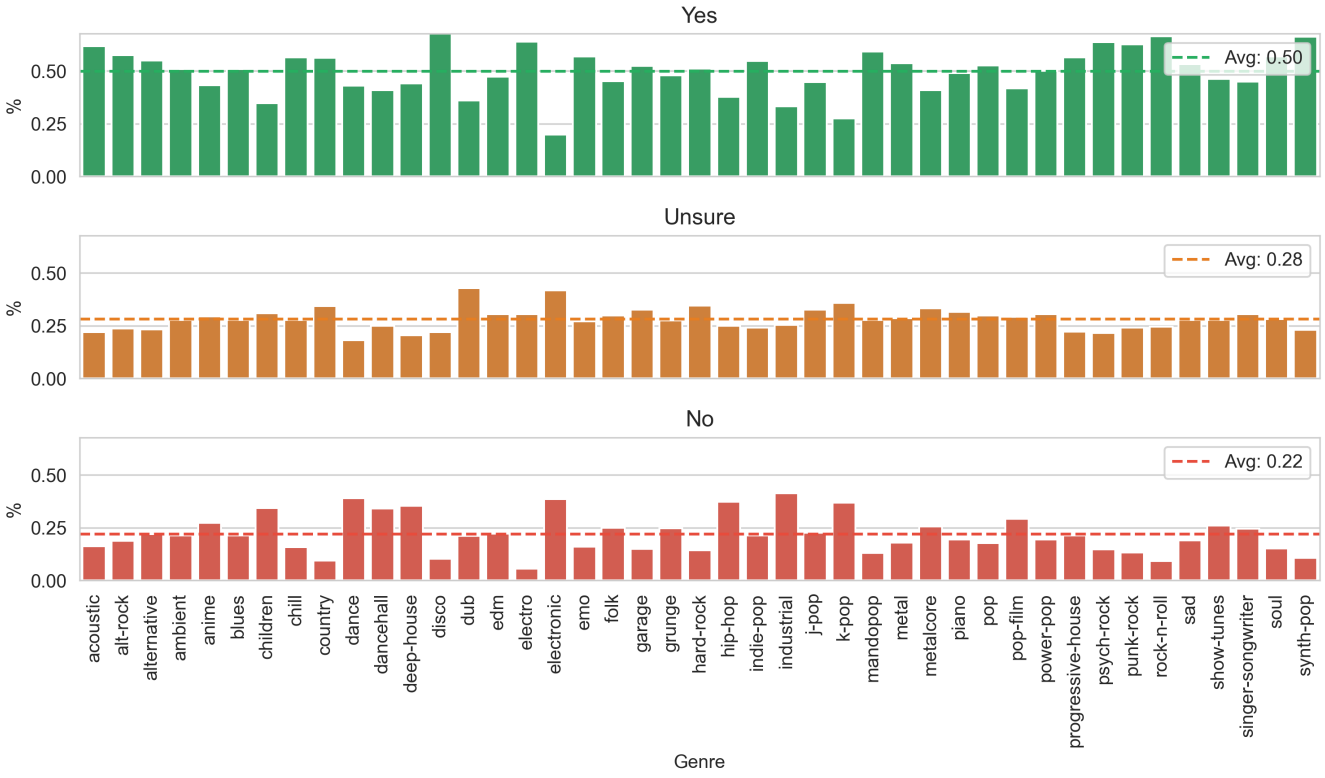

Figure 12: Participant song preferences by genre.

Completion Time: The average time to complete the experiment was 13.8 minutes, approximately 25 seconds per song. However, there was significant variation: Some participants finished in under 5 minutes, while others took over 30 minutes. A comprehensive breakdown of completion times is shown in Figure 13.

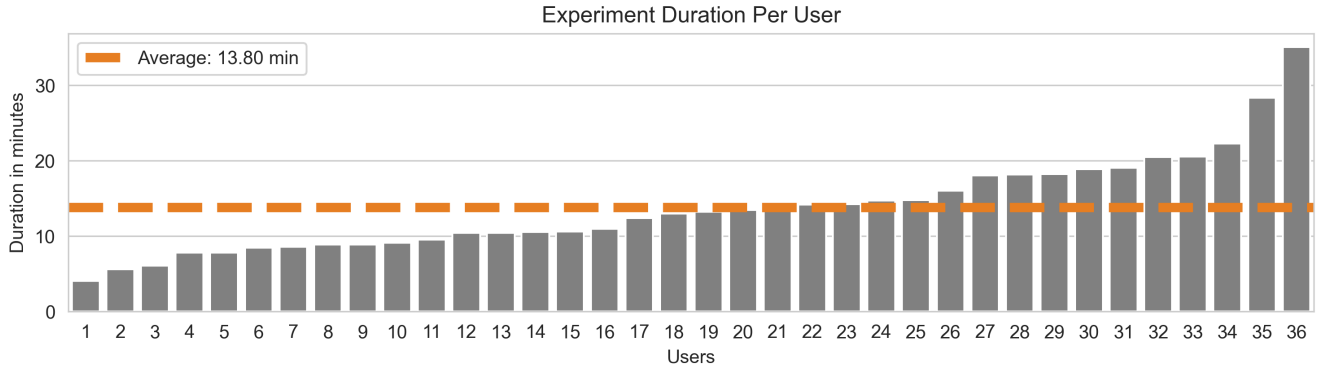

Figure 13: Completion time across participants.

Confusion matrix: The raw number of selected images for each of the 4 categories is presented in Figure 14. YY was selected 43% of the time, compared to NN's 7.9%. YN and NY were near random chance (25%). These results support the VAG hypothesis.

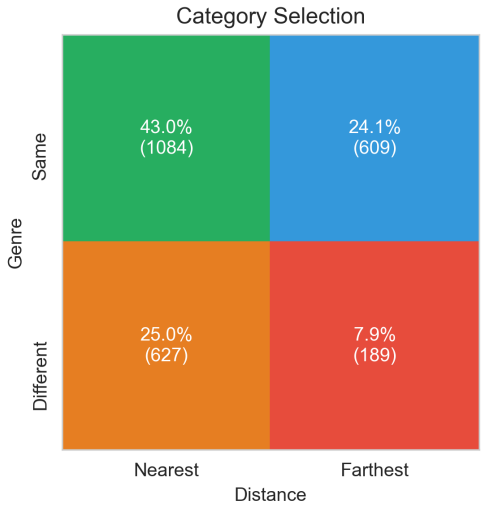

Figure 14: 4 categories confusion matrix.

Category selection - box plot: Further insights can be gained from the box plots in Figure 15, which shows the distribution of selected images across the 4 categories. A relatively large spread is observed, but NN and YY are significantly different from random chance, with NN lower and YY higher. Also worth highlighting, the original image was selected approximately 40% of the time, reinforcing the validity of my approach to generating images tailored specifically for each song.

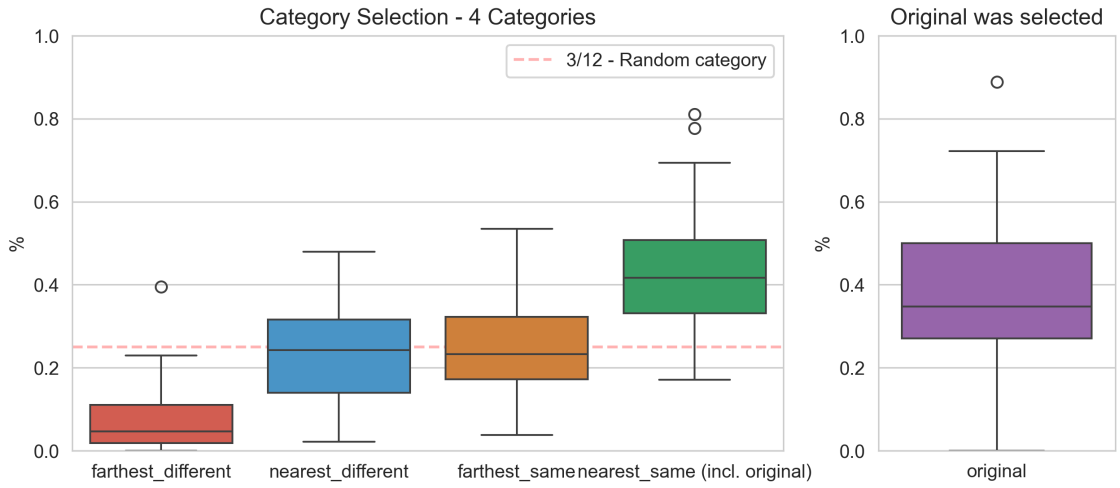

Figure 15: Selection rates for each of the 4 categories and for the original image.

Category selection - per genre: Figure 16 presents the 4-category results on a per-genre basis. The plot shows clear variations in participant preferences across genres. Notably, some discrepancies could benefit from further investigation. For example, NN in the *pop* genre is chosen nearly 50% of the time, while in *dancehall*, YY is not selected by any of the 36 participants.

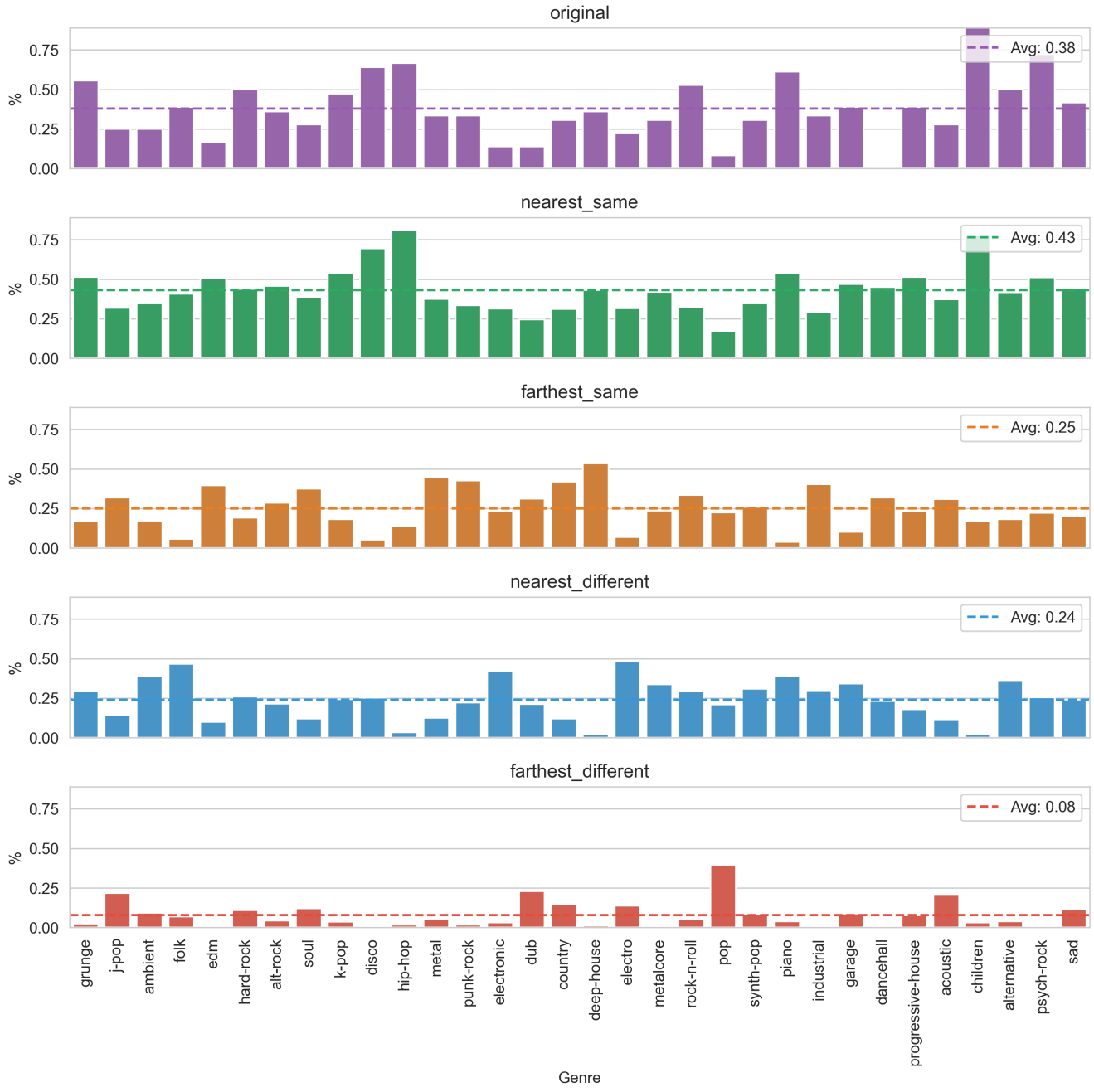

Figure 16: Per genre analysis of the 4-category selections.

Conclusion: The experiment provides evidence supporting the VAG hypothesis, with YY images being selected significantly more often than NN images. The original image selection rate of approximately 43% also validates the approach of generating song-specific images. These results suggest that soft labels based on VAG are plausible. Additionally, the results show that it is easier to identify what people *dislike* (7.9% NN selection rate) than what they *do like* (43% YY selection rate). In other words, the NN-label is more certain than the YY-label.

3.5 Music-to-image Similarity Function

With evidence supporting the VAG hypothesis, I propose the following heuristic. Let I be an AI-generated image produced as described in 3.3.4, and let S be a random song. Both I and S have a genre (I_g, S_g), valence score ($I_v, S_v \in [0,1]$), and arousal score ($I_a, S_a \in [0,1]$). Assuming the VAG results from 3.4.4 are sufficiently strong, we can define a probability function inspired by the idealized distribution $\mathcal{D}(I \mid S; \mathcal{P}, \mathcal{I})$ introduced in 3.3.1 as follows:

$$f(I \mid S) = \underbrace{\alpha \cdot d(I,S)}_{\text{VA-distance}} + \underbrace{\beta \cdot g(I,S)}_{\text{Genre-distance}}, \quad \text{where} \quad \underbrace{\alpha + \beta = 1,\ \alpha, \beta \geq 0}_{Weights}$$

here $d(I,S)$ measures the VA-distance between (I_v, I_a) and (S_v, S_a), while $g(I,S)$ measures the difference between genres I_g and S_g. Many definitions of f and g are possible. A straightforward choice, could be:

$$d(I,S) = 1 - \frac{\sqrt{(I_v - S_v)^2 + (I_a - S_a)^2}}{\sqrt{2}}$$

$$g(I,S) = \begin{cases} 1, & \text{if the genre } I_g \text{ matches the genre } S_g, \\ 0, & \text{otherwise.} \end{cases}$$

where $g(I,S)$ is an indicator function that returns 1 if the genres match and 0 otherwise. The function $d(I,S)$ gives a value in $[0,1]$ based on the VA-proximity between I and S. $d(I,S) = 1$ when I and S have identical VA-coordinates and equals $d(I,S) = 0$ when they are maximally different. A visualization of $d(I,S)$ can be found in Figure 17.

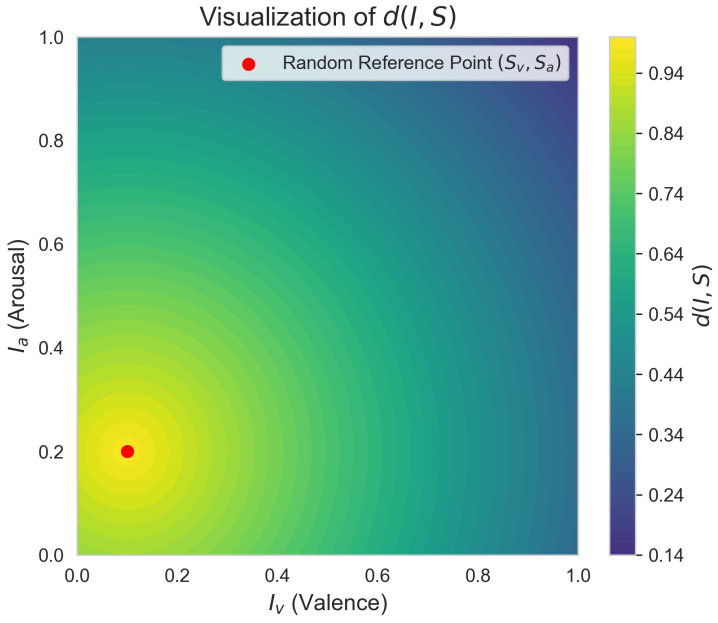

Figure 17: VA-distance function

You can imagine that more advanced d and g functions could improve f. For instance, a genre function that accounts for closely related genres would be beneficial (e.g. treating "rock" and "alternative rock" as more similar than "rock" and "classical"). Furthermore, there is no inherent reason why f needs to be a probability function. It could just as well be a similarity function not restricted to the domain of $[0,1]$. I simply find it more intuitive to work with probabilities.

3.6 BEATS: Bridging Emotions and Art through Sound

Based on the experiment results, I was confident that the VAG method could be used to train an M2I model. The dataset used during experimentation became BEATS-mini and BEATS-15K. With confidence in the VAG approach, I went on to construct a larger dataset named BEATS-400K. Due to its size, AI-generated images were not created for this dataset. A summary of the BEATS datasets can be found in Table 5.

Table 5: Comparison of BEATS Datasets

Feature	**BEATS-mini**	**BEATS-15K**	**BEATS-400K**
Unique songs	250	5000	36,342
Unique genres	50	50	101
Song-to-prompt pairs	750	15,000	363,420
Song-to-image pairs	750	15,000	0
YouTube meta data	X	✓	✓

Alongside the dataset itself, the similarity function introduced in 3.5 can be used to generate soft labels. For instance, by setting $\alpha = \beta = 0.5$ and filtering potential matches with $f(I, S) > 0.75$, you can ensure different genres and sufficiently large VA-distances. while filtering on $f(I, S) < 0.1$ would accomplish the opposite. You can than e.g. select randomly from these pools of images, and hereby dynamically assign NN and YY-labels to each song during training.

4 Music-to-Image Model

This section will explain the M2I pipeline summarized in Figure 18 in three parts: First, *Audio Encoder and Preprocessing* 4.1 discusses the audio encoder's architecture and the necessary preprocessing steps. Second, *SDXL Integration and Image Synthesis* 4.2 describes how the components come together to produce synthesized images. Third, *Audio-CLIP Alignment* 4.3 explains the alignment process which enable the audio encoder's use in SDXL [2].

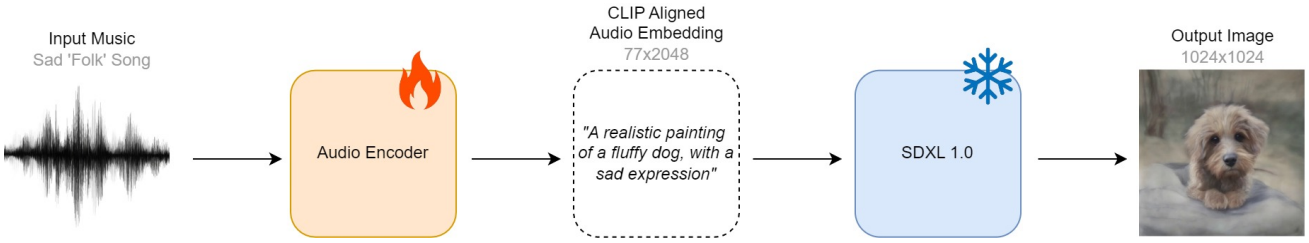

Figure 18: M2I pipeline overview. Blue boxes indicate AI components with frozen weights, and orange boxes represent AI components with trainable weights that are updated during training.

4.1 Audio Encoder and Preprocessing

Consider Figure 19. The pipeline begins with preprocessing the input audio, which is then passed to CLAP [46]. CLAP requires a 10-second mono-channel audio clip sampled at 48 kHz and represented as a 32-bit float. Once the audio clip is formatted correctly, it is sent through a pretrained CLAP model. The idea is to leverage its existing knowledge of music and audio, potentially making the training process easier. CLAP outputs a single 512-dimensional embedding, which is duplicated and sent through two separate transformer pipelines. These pipelines are identical in functionality but differ in dimensionality. One uses a 768-dimensional latent space, and the other a 1280-dimensional latent space. These dimensions are chosen for compatibility with the dual CLIP [21] text encoders used in SDXL. I will elaborate on this in 4.3.

The transformer pipeline begins with a linear projection layer that maps the 512-dimensional CLAP embeddings to 768 and 1280 dimensions, respectively. Both embeddings are then copied 77 times to match the dual CLIP encoders' output. A learned positional embedding is added to each, and the embeddings are passed through two separate 8-headed, 4-layer transformers. This process produces tensors of size 77×768 and 77×1280, which are concatenated into a 77×2048 tensor—the exact dimensions required by SDXL for image synthesis.

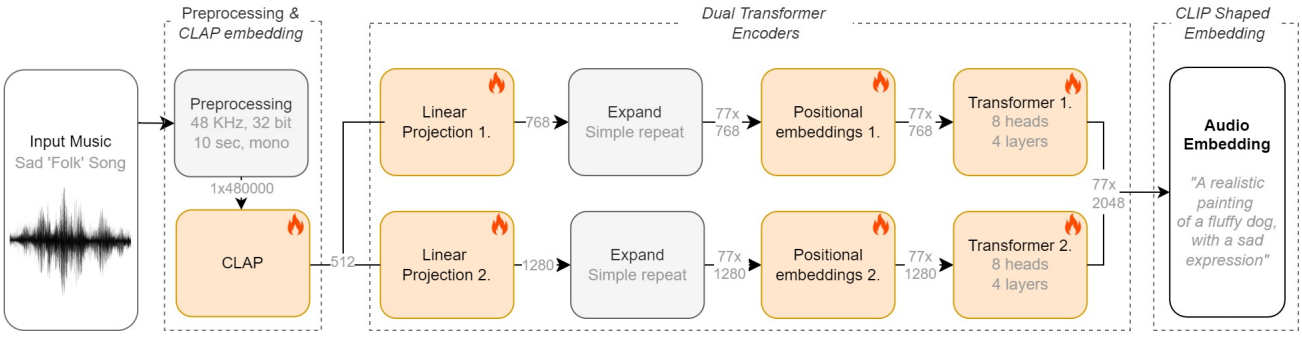

Figure 19: Audio Encoder architecture. White boxes represent raw data, grey boxes indicate programming steps without AI, and orange boxes represent AI components with trainable weights that are updated during training.

4.2 SDXL Integration and Image Synthesis

My work builds on Stability AI's implementation of SDXL 1.0 [62]. I used a simplified version that relies solely on positive text prompts, as shown in Figure 20. A more detailed overview of the model architecture can be found in the appendix Figure 39.

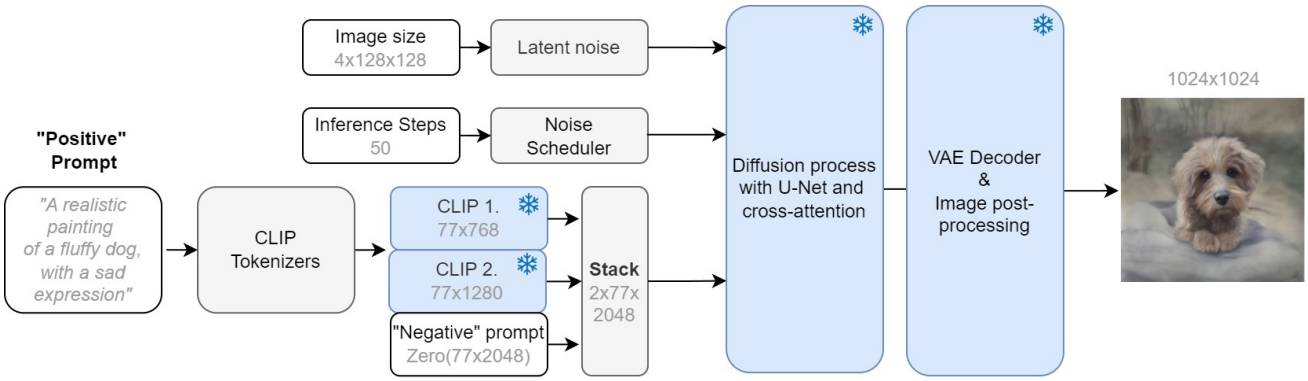

Figure 20: Simplified SDXL implementation. White boxes represent raw data, grey boxes indicate programming steps without AI, and blue boxes indicate AI components with frozen weights

This pipeline transforms a text prompt of up to 77 tokens into high-quality images. The output image size is determined by the shape of the initial latent noise. For example, latent noise of 128×128 produces a 1024×1024 image, while noise of 64×64 generates a 512×512 image.

The *Noise scheduler* controls the iterative removal of noise from the latent image representation during inference, ensuring a smooth and controlled denoising process. The *Inference steps* parameter determines how many steps the noise scheduler uses for denoising. More steps generally improve image quality and fidelity but also increase inference time. A value of 50 inference steps is commonly used, presumably because it strikes a good balance between quality and speed.

Text conditioning is achieved using two CLIP models: One operating in a 768-dimensional latent space and the other in a 1280-dimensional latent space. Theoretically, this dual-encoder setup enables the smaller encoder to capture high-level details, while the larger encoder focuses on fine-grained details. Together, these embeddings create a combined 2048-dimensional encoding for each token. SDXL 1.0 supports a maximum of 77 tokens. Although SDXL supports negative prompt conditioning (using the same process as positive prompts), I did not use this in my implementation. Instead, a simple 77×2048 zero tensor concatenation is used for negative prompts.

The degree to which the image synthesis process adheres to the content of the prompt is controlled by the *guidance scale* (not shown in the figure). High guidance scale values make the image match the text more closely, while lower values give the model more artistic freedom. Extremely high values may lead to overfitting or artifacts. On the other hand, very low values risk losing the intent of the prompt, resulting in images that do not reflect the user's expectations.

Once text conditioning and latent noise are produced, image generation proceeds through the *Diffusion process with U-Net and cross-attention*. During this process, U-Net refines the latent noise step by step into a meaningful representation of an image. Cross-attention layers guide this process to ensure the image matches the content and style of the text prompt. The guidance scale is also applied during this stage by combining unconditional and text-conditioned predictions.

This is followed by the *VAE Decoder & Image post-processing*. Before decoding, the latents are scaled and denormalized. The VAE Decoder then converts the refined latent representation into pixel space, producing a full-resolution image. Post-processing applies some final adjustments, resulting in the final output.

The integration of my audio encoder, introduced in 4.1, is shown in Figure 21. The modifications to the original SDXL pipeline are minimal. The CLIP tokenizer is replaced with an audio preprocessing step to format the audio correctly, and the dual CLIP text encoders are replaced with the audio encoder. The goal is for the audio encoder to align its embeddings with CLIP's text embeddings. If successful,

the audio encoder can mimic the output of CLIP, enabling the generation of image prompts that match the music input. This alignment process is detailed in the next section.

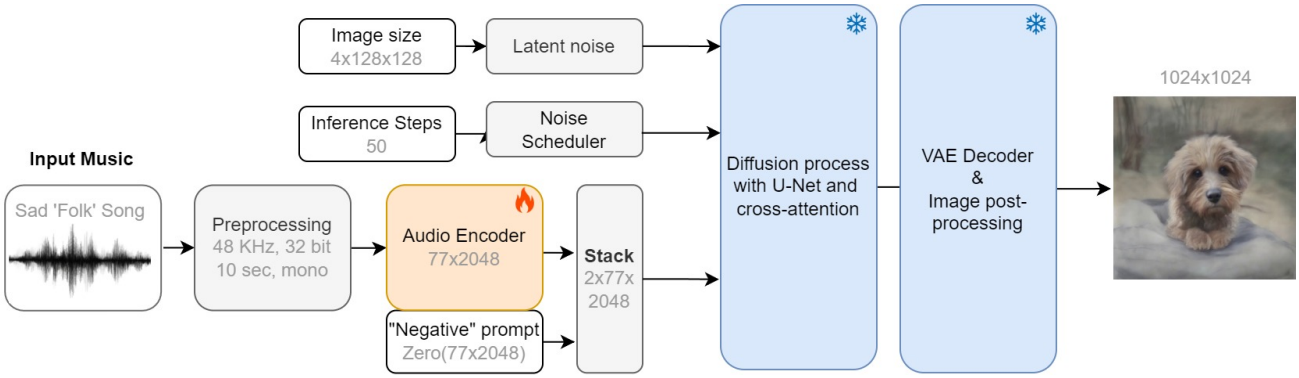

Figure 21: Audio encoder's integration with SDXL. White boxes represent raw data, boxes grey boxes indicate programming steps without AI, blue boxes indicate AI components with frozen weights, and orange boxes represent AI components with trainable weights that are updated during training.

4.3 Audio-CLIP Alignment

Figure 22 illustrates how the audio encoder is aligned with CLIP. The goal is to bring the audio embedding closer to a positive text embedding and push it away from the negative text embeddings. For example, consider an audio clip with "Sad Folk Music". A suitable positive text prompt might be "A realistic painting of a fluffy dog with a sad expression", while a negative prompt could be "An image of a high-energy colorful festival, joyful chaos". In this case, the objective is for the audio encoder to align the "Sad Folk Music" embedding with the "sad fluffy dog" prompt while distancing it from the "festival" prompt. This alignment is achieved using a custom loss function inspired by infoNCE [47].

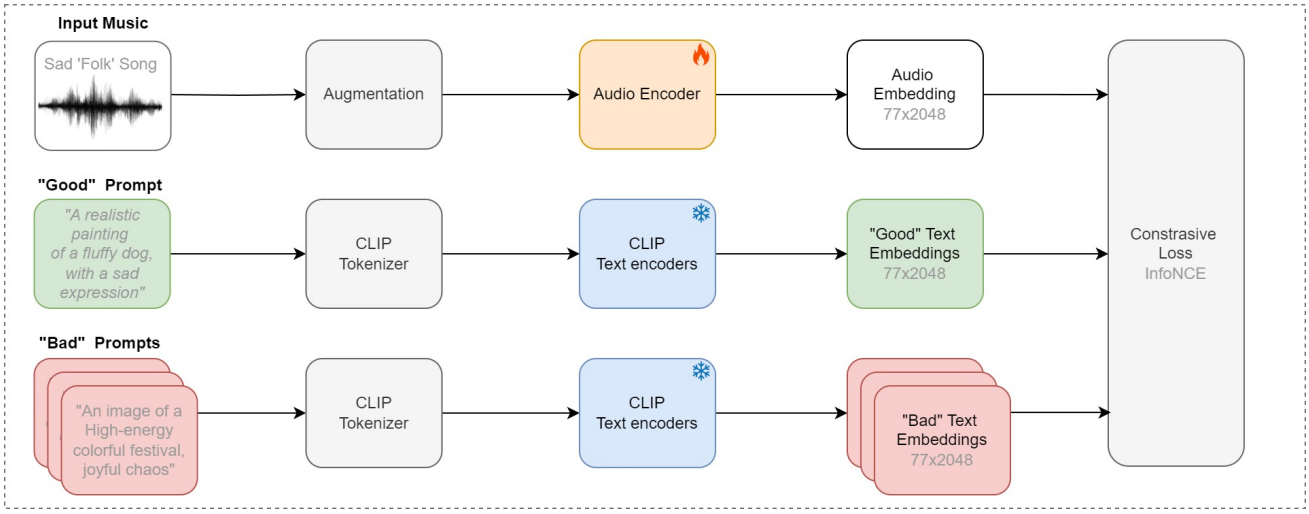

Figure 22: Illustration of the audio encoder's alignment with CLIP. White boxes represent audio-related data, green boxes indicate text prompts that match the audio well, and red boxes indicate prompts that poorly match the audio. Grey boxes are non-AI steps, blue boxes are frozen AI components, and orange boxes are trainable AI components. The idea is to use contrastive loss to encourage the audio embedding to align closely with the "good" text embedding while discouraging alignment with the "bad" text embeddings.

Batch Structure:
A training batch is defined as a set of B triplets:

$$\mathcal{B} = \{(\mathbf{a}_i, \mathbf{p_i}, \{\mathbf{n}_{i,0}, \ldots, \mathbf{n}_{i,N}\})\}_{i=1}^{B}.$$

Here $\mathbf{a}_i$ represents the audio embedding, $\mathbf{p}_i$ is a positive text embedding explicitly linked to $\mathbf{a}_i$, and $\mathbf{n}_{i,j}$ is a sequence of negative text embeddings also explicitly linked to $\mathbf{a}_i$. How audio, positive and negatives prompts are selected is detailed in 5.

The Loss Function:
First, the embeddings are averaged across the 77 tokens, and normalized to unit length:

$$\hat{\mathbf{a}}_i = \frac{\overline{\mathbf{a}}_i}{\|\overline{\mathbf{a}}_i\|}, \quad \hat{\mathbf{p}}_i = \frac{\overline{\mathbf{p}}_i}{\|\overline{\mathbf{p}}_i\|}, \quad \hat{\mathbf{n}}_{i,j} = \frac{\overline{\mathbf{n}}_{i,j}}{\|\overline{\mathbf{n}}_{i,j}\|}$$

Here $\overline{\mathbf{a}}_i$, $\overline{\mathbf{p}}_i$, and $\overline{\mathbf{n}}_{i,j}$ are the averaged embeddings across tokens, and $\hat{\mathbf{a}}_i$, $\hat{\mathbf{p}}_i$, and $\hat{\mathbf{n}}_{i,j}$ are their unit-normalized versions. Cosine similarities are then computed as follows:

$$s_i^+ = \hat{\mathbf{a}}_i \cdot \hat{\mathbf{p}}_i, \quad s_{i,j}^- = \hat{\mathbf{a}}_i \cdot \hat{\mathbf{n}}_{i,j}$$

With a temperature parameter τ, the InfoNCE loss is defined as:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{B}\sum_{i=1}^{B} \log\left(\frac{\exp\left(\frac{s_i^+}{\tau}\right)}{\exp\left(\frac{s_i^+}{\tau}\right) + \sum_{j=1}^{N}\exp\left(\frac{s_{i,j}^-}{\tau}\right)}\right)$$

This loss function encourages higher similarity between audio embeddings and positive text embeddings (s_i^+) while reducing similarity to negative text embeddings ($s_{i,j}^-$), thus aligning the audio encoder with CLIP. The temperature parameter τ controls the "sharpness" of the similarity scores. Lower τ values increase the emphasis on the gap between positive pairs s_i^+ and negative pairs $s_{i,j}^-$, whereas higher τ values smooths out differences, making the loss less sensitive to small score gaps.

5 Experimental Setup

This section covers dataset preprocessing, training configurations, hardware limitations, augmentation, and other implementation details.

5.1 Splits

The BEATS-400K dataset was split into training, validation, and test sets. 80% of the data was assigned to training, 20% to validation, and 32 songs (those used in the experiment 3.4.4) were used as a test set. Splits were made across unique songs to prevent data leakage i.e. each song appears in only one of the three sets. The train-validation split was performed using a stratification process. Valence and energy scores were divided into four equal bins: $[0, 0.25]$, $(0.25, 0.5]$, $(0.5, 0.75]$, and $(0.75, 1.0]$. These bins were combined with genre labels to produce balanced splits:

$$\text{stratification\_key} = \text{"\{valance-bin\}\_\{energy-bin\}\_\{genre\}"}$$

Combinations with fewer than 2 samples were labeled as "rare". After additional cleaning (details omitted, not important), the final dataset consisted of 290,530 song-to-prompt pairs in the training split and 72,570 song-to-prompt pairs in the validation split.

5.2 Song Preprocessing

To ensure the sound clips better reflect the underlying music, I removed the start and end of each song. This avoids low-energy intros, silent endings, and similar issues. The process is illustrated in Figure 23.

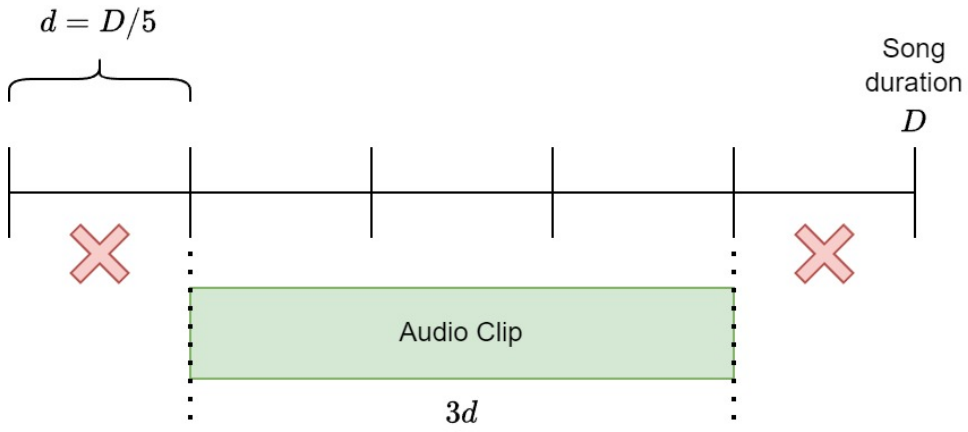

Figure 23: Audio preprocessing. A song with total duration D was divided into 5 equal chunks, each of duration $d = D/5$. The first and last chunks were removed, leaving the 3 middle chunks for training.

5.3 Text Embeddings

Initially, I considered text augmentations such as back-translation, word deletion, word swapping, and synonym swapping. However, I decided against them due to the computational overhead. Avoiding augmentation also allowed me to precompute and cache the CLIP embeddings. This significantly reduced training time. The only preprocessing required was handling rare cases where prompts exceeded CLIP's 77-token limit. Since there were fewer than 100 instances of this, I simply truncated them without further analysis.

5.4 Batch Construction

Batches are constructed using the VAG-inspired similarity function introduced in 3.5. Each batch consists of triplets: A song S, a positive prompt T_p, and 20 negative prompts $T_n = \{T_{n_1}, T_{n_2}, \ldots, T_{n_{20}}\}$. The positive prompt T_p is the original text prompt assigned to the song S, while the negative prompts T_n are randomly selected from a pool where $f(T, S) > 0.70$.

Ideally, the entire dataset would be used to construct each batch, allowing for a maximum number of samples when creating the 20 negative pairs. However, due to computational limitations, I had to use sub-batches, which will be explained in the next section on caching.

5.5 Batch Caching

Audio I/O is computationally expensive, so I was forced to design a training pipeline that utilized cached sub-batches. This allowed fast data loading while still making use of dynamic batch construction (e.g. dynamic sound augmentations and dynamically assigned negative prompts for each batch).

As a result, batch construction was limited to a subset of the dataset. The full dataset consists of approximately 400,000 song-to-prompt pairs across 40,000 unique songs. For sub-batching, training and validation were handled separately, with each sub-batch containing around 5,000 unique song-to-prompt pairs to ensure a diverse mix of valence, energy, and genre combinations.

The creation and caching of all sub-batches took roughly one day of compute[5] and used about 2 TB of disk space. Each sub-batch requires about 50 GB of RAM when loaded. Even though the initial caching process was computationally expensive, it made training feasible with my available resources.

5.6 Audio Augmentation

To prevent overfitting and encourage generalization, 5 augmentations were applied to the raw audio waveforms:

- **Random Cropping:** Crop out a random 10 second audio clip.
- **Time Stretching:** Simulate variations in tempo by adjusting playback speed.
- **Pitch Shifting:** Modify the pitch.
- **Additive Noise:** Add random Gaussian noise.
- **Random Volume Scaling:** Multiply the waveform's amplitude by a random scaling factor.

5.7 Validation Experiment

I conducted a small-scale human experiment to verify whether the final results aligned with human expectations. Due to time constraints, I could not conduct a large-scale experiment or deploy a website for labeling. Instead, I synthesized 32 images using the final model, corresponding to the 32 songs in the test set (the same songs used in 3.4.4). I created a video where each image was displayed for 10 seconds alongside its corresponding song and asked participants to indicate whether they liked the match or not.

5.8 Equipment

I trained on a personal PC, using approximately 1,000 hours of compute time during testing and training. The hardware specifications are as follows:

- **CPU:** Intel(R) Core(TM) i9-14900KF, 3.20 GHz
- **GPU:** NVIDIA GeForce RTX 4090, 24 GB VRAM
- **RAM:** 64 GB DDR5

[5] I used a purely sequential implementation. This could have been significantly sped up with even mild parallelization.

6 Preliminary Studies

Once the BEATS dataset was ready, I conducted a series of smaller training runs to explore and refine details. This included the choice of loss function, suitable learning rates, schedulers, regularization, and model architecture. Drawing exact conclusions from these non-rigorous studies is difficult. Nonetheless, I include them here because they were instrumental in forming the final model and may provide insights into M2I models in general.

6.1 SDXL is Robust to Noisy CLIP Embeddings

A key motivation for using the CLIP alignment method was SDXL's remarkable robustness to noise. Even with significant noise in its CLIP embeddings, SDXL can still produce high-quality imagery—even at high guidance scales. Figure 24 demonstrates this by progressively adding noise to the prompt embeddings across different guidance scales. The noise was constructed as follows:

$$\mathrm{E}_{\text{noisy}} = (1 - \alpha) \cdot \mathrm{E}_{\text{original}} + \alpha \cdot \mathrm{E}_{\text{noise}}$$

where $\mathrm{E}_{\text{noisy}}$ is the noisy embedding, $\mathrm{E}_{\text{original}}$ is the pure signal CLIP embedding, $\mathrm{E}_{\text{noise}}$ is Gaussian noise, and $\alpha \in [0, 1]$ is the noise scaling factor. At $\alpha = 0$ the embedding is entirely original, and at $\alpha = 1$, it becomes pure noise.

These results suggest that the audio encoder does not need to perfectly align with CLIP embeddings. It only needs to achieve a reasonably good alignment to produce high-quality results.

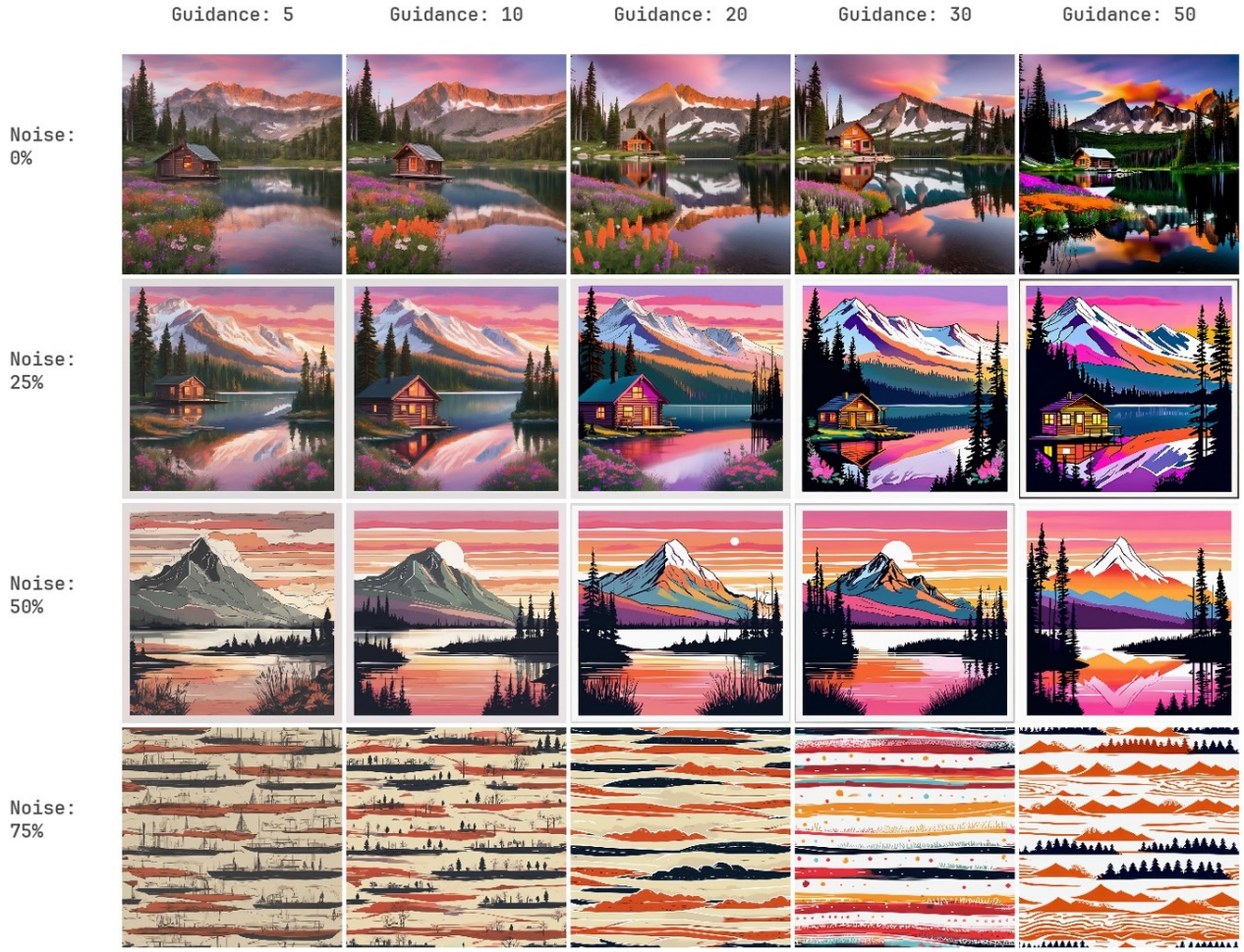

Figure 24: Noisy prompt embeddings across different guidance scales. Prompt used: *A serene mountain landscape at sunrise, with snow-capped peaks reflecting in a clear lake, vibrant wildflowers in the foreground, a cozy cabin by tall pines, and a colorful sky blending orange, pink, and purple.*

6.2 A Pretrained Audio Encoder is Necessary

CLAP proved to be Necessary. I experimented with alternative methods, such as LSTMs and CNNs, but none of these produced usable results. The model generated abstract images with a "patchy" appearance, as seen in Figure 25. These images appeared to be composed of arbitrary primitive shapes, such as broad brush strokes, square-like patterns, or cut-and-paste style collages. Despite this, the model managed to capture the overall "feel" of the music.

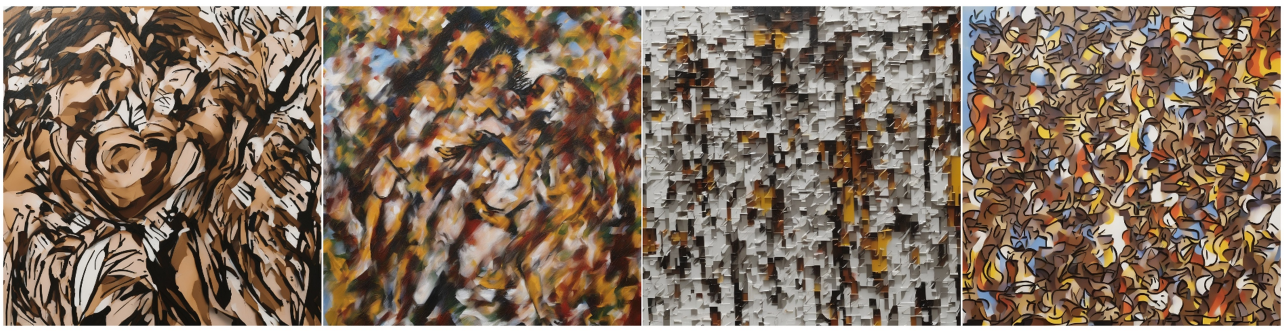

Figure 25: Images without CLAP appear abstract and patchy. However, the colors and shapes subjectively corresponded well to the underlying music style.

Given these results, I pivoted to use an already established audio encoder architecture for initial audio processing. However, I discovered that this approach was only successful when using a pretrained model. Figure 26 shows significantly improved results compared to Figure 25. That said, the model still exhibited "mode collapse" behavior. It consistently generate variations of a specific scene (e.g. two people kissing), regardless of the music type. For instance images for *heavy metal* and *classical* music would be similar. Scene collapse was a recurring issue during the development of the M2I pipeline. Using pretrained CLAP models combined with a sufficiently large transformer network alleviated some of this behavior, though it did not eliminate it entirely.

Figure 26: Images with pretrained CLAP have improved image quality. Furthermore, he model responded well to musical color variations e.g. sad music tended to produce grayish tones, while happy, energetic music resulted in brighter, warmer colors.

6.3 Network on Top of CLAP is Important

The first attempt to use CLAP involved adding a simple projection layer, which completely failed. I then experimented with larger but still relatively simple networks. These models did not produce usable results, but they provided some interesting insights. As shown in Figure 27, the outputs are all combinations of blue and red. Although the model failed to produce detailed images, it captured a signal between color and valence. Blue tones generally corresponded to low valence, while red tones were linked with high valence. Songs with ambiguous valence (e.g. 0.5) often displayed a mix of blue and red.

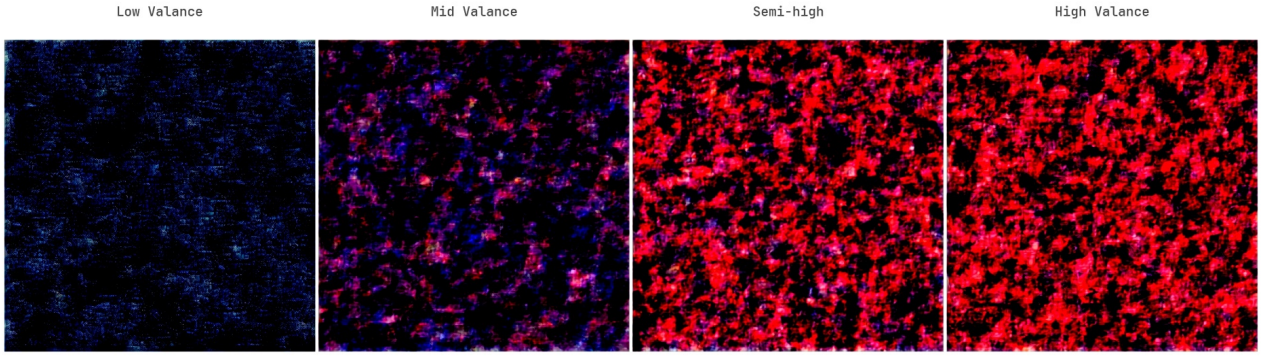

Figure 27: Outputs from a simple model. Blue tones for low valence and red tones for high valence.

My results drastically improved, when I introduced a transformer network on top of the CLAP encoder. This architecture was the first to produce outputs with real potential, as shown in Figure 28. Encouraged by these results, I decided to focus exclusively on refining this type of model. This ultimately led to the final architecture described in 4.1. The model not only captured the overall "feel" of the music but also specific emotions, colors, and real-world objects. Additionally, it demonstrated a solid understanding of the VA-model and a rudimentary grasp of subtle genre-specific nuances.

Figure 28: Preliminary results using a transformer network on top of CLAP. Images are significantly less abstract than previous attempts and have good emotional resonance.

6.4 Optimizer

I found that learning rate had a profound impact on the model's performance. Learning rates below 5e-4 often led to unstable training, and in some cases complete model collapse, where the output images stopped making sense. I found that learning rates in the range of 1e-4 to 5e-6 performed well. While very low learning rates occasionally caused overfitting, the training pipeline generally handled them effectively and seemed to benefit.

I experimented with different learning rate schedulers and found that a simple exponential decay inspired one worked well. Starting at 1e-4 and decaying to 5e-6 provided a good balance. The idea was, faster learning initially and refined results later. The learning rate scheduler is defined as:

$$l_i = \max(l_{min} \cdot r^i, l_{max}), \quad \text{where} \quad r = \sqrt[E]{\frac{l_{max}}{l_{min}}} \tag{1}$$

Here, i is the current epoch, l_i is the learning rate at epoch i, $l_0 = 1 \cdot 10^{-4}$ is the initial learning rate, l_{max} is the maximum learning rate, and r is the decay rate derived from the total number of epochs E, ensuring that l_E is very close to l_{max}.

6.5 Freedom vs. Regularization

During my preliminary studies I realized that there was an obvious tug-of-war between freedom and regularization. When the model was given more freedom (minimal augmentation, no weight decay, etc.) the model tended to produce *hit-and-miss* results as shown in Figure 29. By *hit-and-miss*, I mean the outputs ranged from excellent to very poor. Additionally, lower-represented genres would sometimes be completely overlooked. For instance, the *kid* category eventually stopped producing kid-friendly results, while higher-frequency genres became even more well-defined.

On the other hand, implementing stronger regularization prevented the model from going completely off track and ensured better representation of low-frequency classes. However, this came at the cost of generating less visually pleasing results.

This trade-off persisted throughout my experiments, suggesting it may be an inherent property of the system. You can either (1) achieve excellent results with occasional misses or (2) maintain consistency with less appealing results overall, but not both.

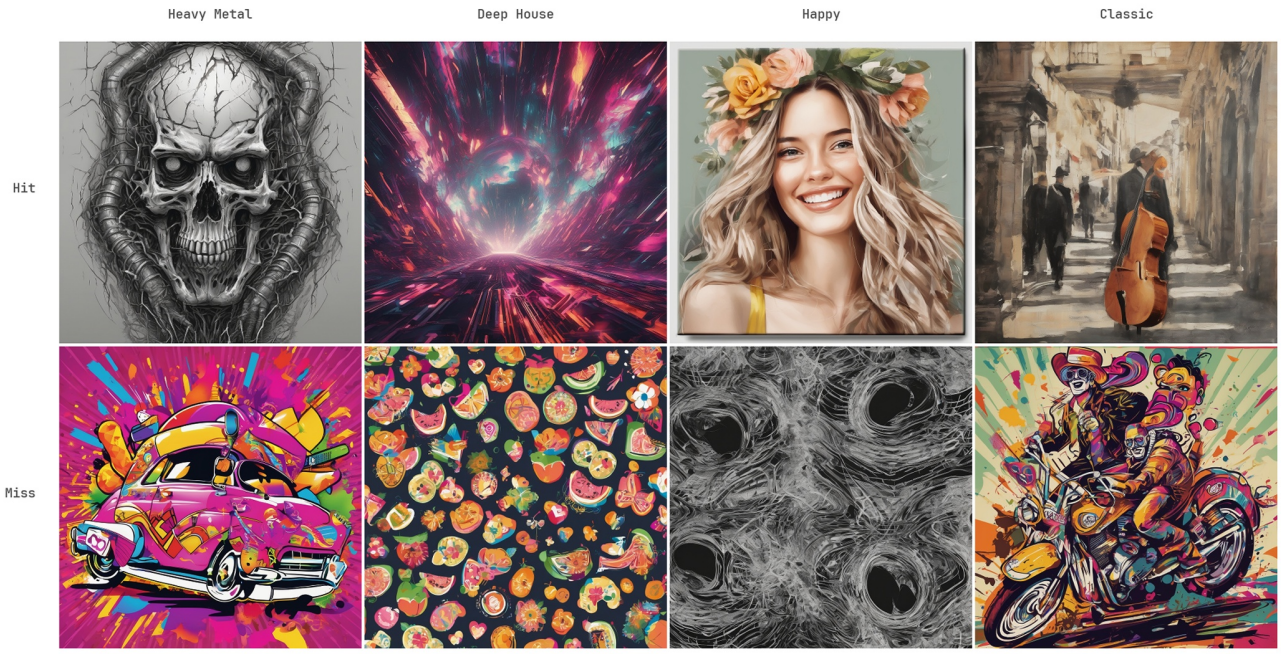

Figure 29: Trade-off between freedom and regularization. Low regularization produces *hit-and-miss* results.

7 Results

This section presents results of the final M2I model, highlighting its overall performance, strengths, and weaknesses. It is divided into six parts: First, *Training*, *Results Overview*, *Human Experiment*, *Results Across a Single Song*, *Repeated Themes*, and *High Guidance Scale*.

7.1 Training

I trained the audio encoder (illustrated in Figure 19) for 80 epochs on 50% of the BEATS-400K dataset. Using only half the dataset was a necessary because of computational constraints. The training process was demanding and took approximately 10 days on an RTX 4090 GPU. I maxed out the 24 GB VRAM and approximately 50 GB of system RAM during training.

I used PyTorch's ADAM optimizer [63] with its default settings, as it is known for being reliable and efficient. The exponentially decaying learning rate scheduler from Equation 1 was applied, with an initial learning rate of l_{min} = 1e-4 and a maximum learning rate of l_{max} = 1e-6.

While the model did not show clear convergence as seen in Figure 30, the qualitative improvements over time were clearly visible, as can be seen in Figure 31.

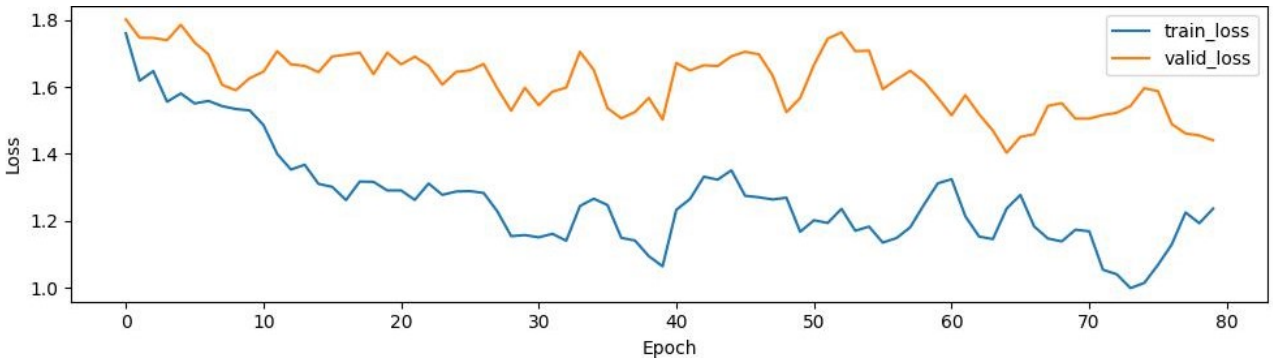

Figure 30: Learning rate behavior during training. Due to several interruptions during training, I had to stitch this graph together and took some liberties smoothing out the data and adding missing epochs. However, the overall graph accurately reflects the true loss trend.

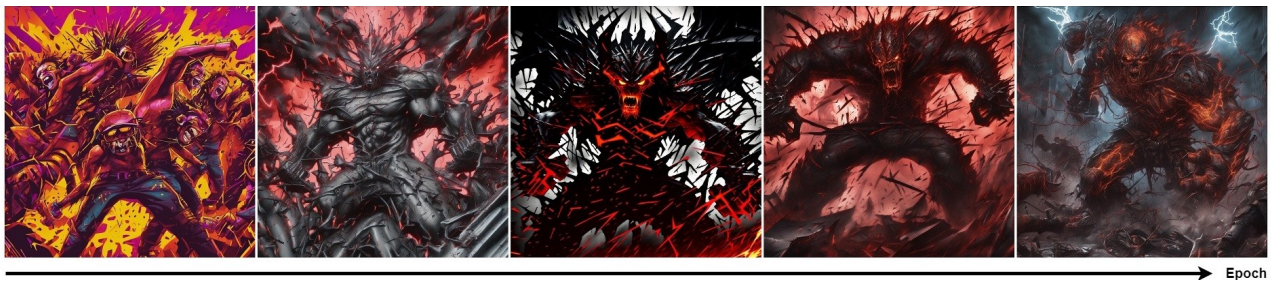

Figure 31: Evolution of images during training. The 5 images shown corresponds to the same song at different epoches.

7.2 Results Overview

Given the audio-visual nature of my results, the best way to experience them is by listening to the music while viewing the generated images. I therefore highly encourage you to watch the video this URL for a fuller understanding. The video showcases a broad range of music genres alongside notable results I felt were worthwhile highlighting.

It is challenging to fully convey the musical emotions here, even with song titles, as the specific clips used for generation may differ from what listener has in mind. However, Figure 32 provides a visual summary of key findings suitable for paper format.

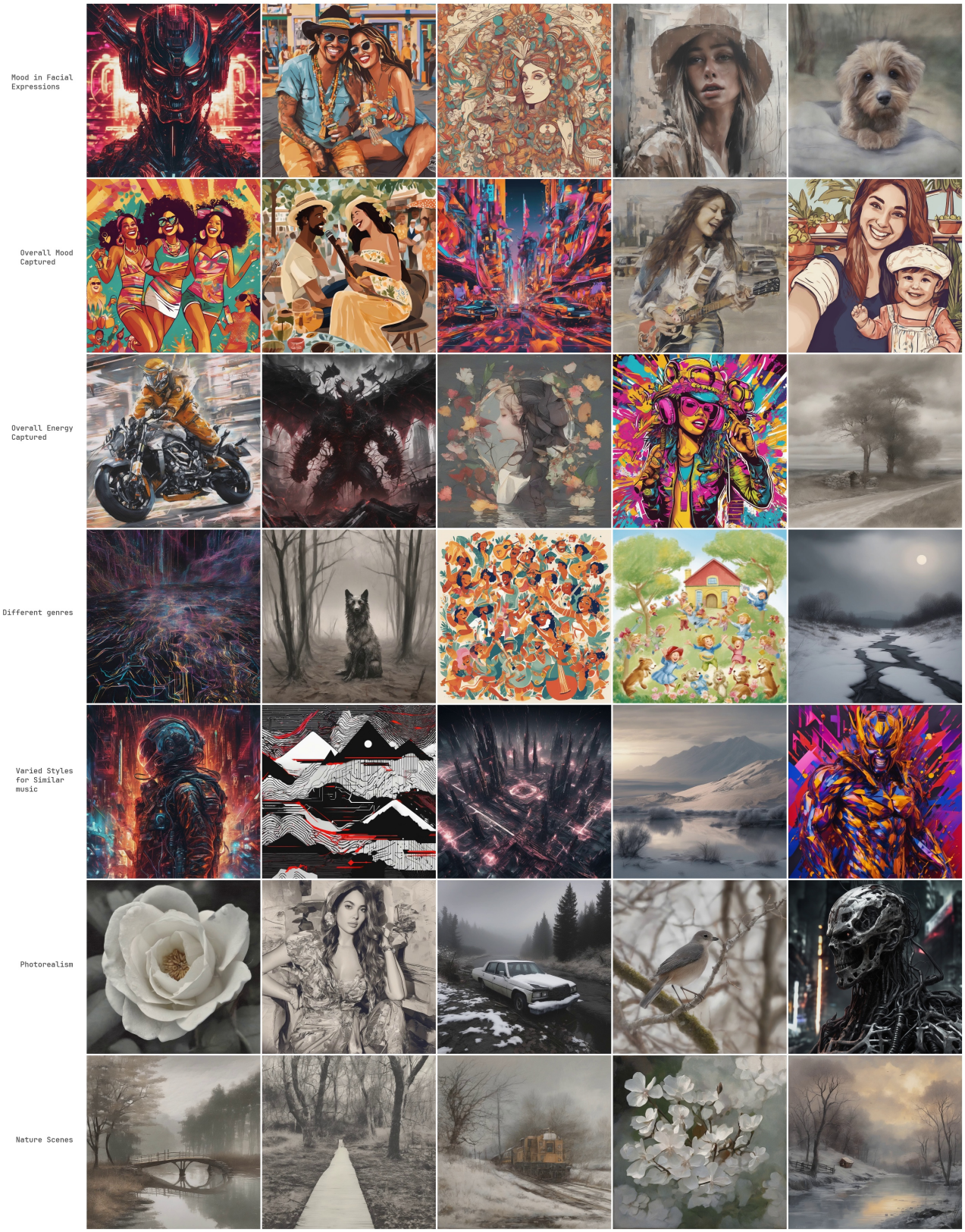

Figure 32: Visual breakdown of findings highlighted in the results video. The figure contains 7 rows with descriptions of what is being highlighted.

7.3 Human Experiment

Consider Figure 33. The experiment involved 11 participants, all with a personal connection to me, which may bias their opinions. This was not designed as a rigorous study but rather as a proof of concept. On average 78% of participants agreed that the images matched the songs played.

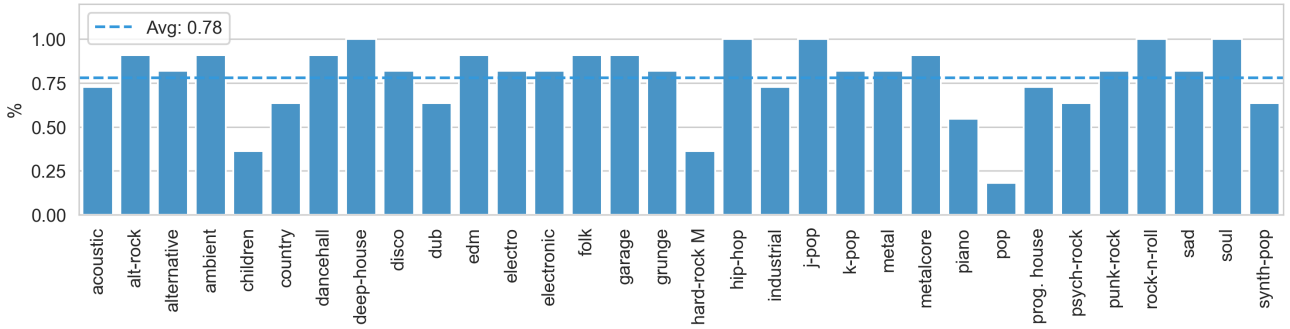

Figure 33: Participants' preferences across all 32 songs.

There is a significant spread. Most images aligned well with people's expectations, but a few fell short. Examples are shown in Figure 34. Images in the *Worst* category all seem to be poorly refined. A video containing all 32 songs and their corresponding images can be found at this URL. The images alone are available in appendix Figure 40. .

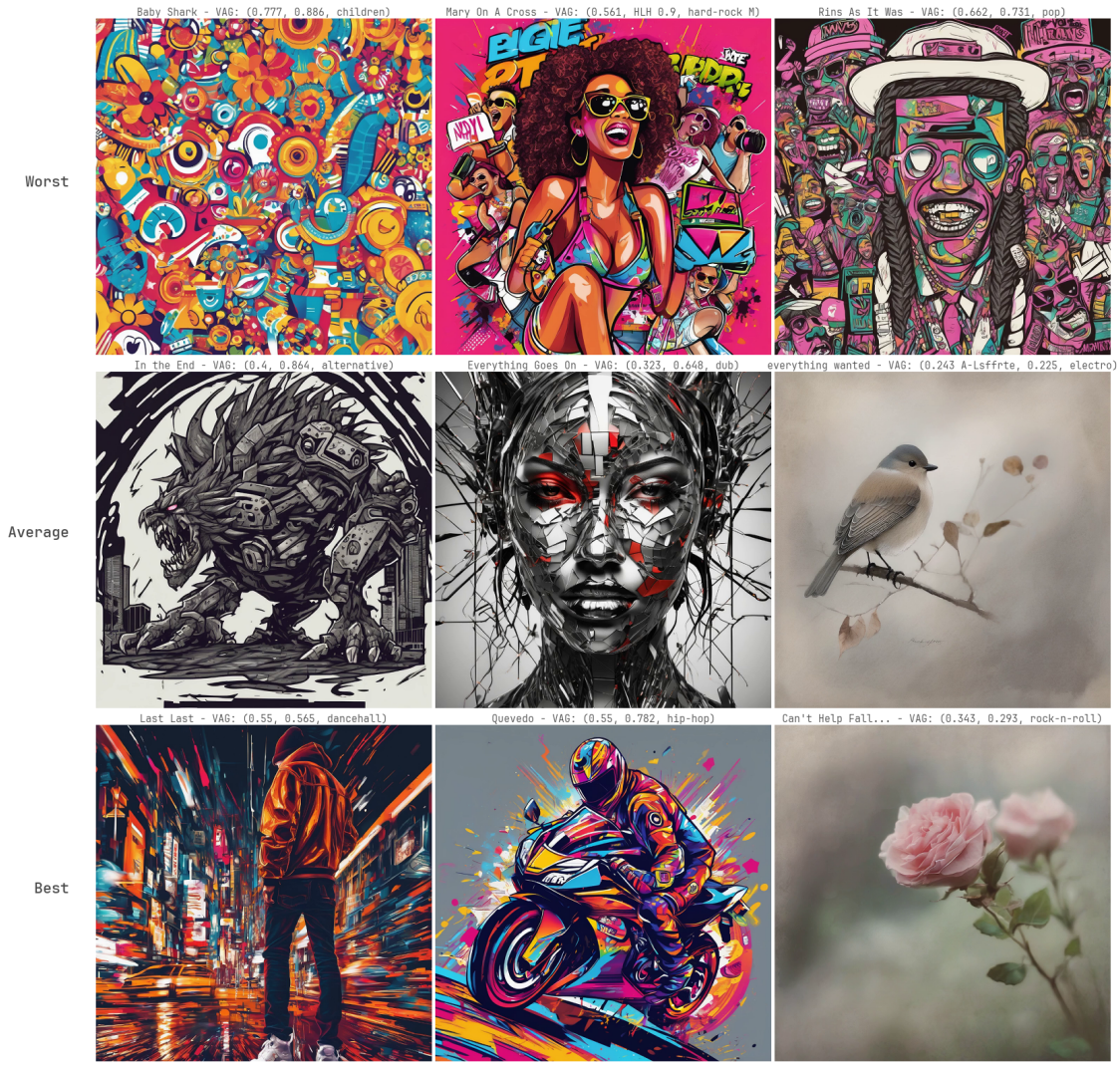

Figure 34: Highlights a selection of images from the experiment.

7.4 Results Across A single Song

The model demonstrates relatively stable behavior when processing the same song, as shown in the video at this URL. Figure 35 illustrates this with images generated from 10-second audio clips spanning a full song. The last two images are very different from the rest because the song was silent in these clips. In other words, the model generated images based on what is essentially background noise.

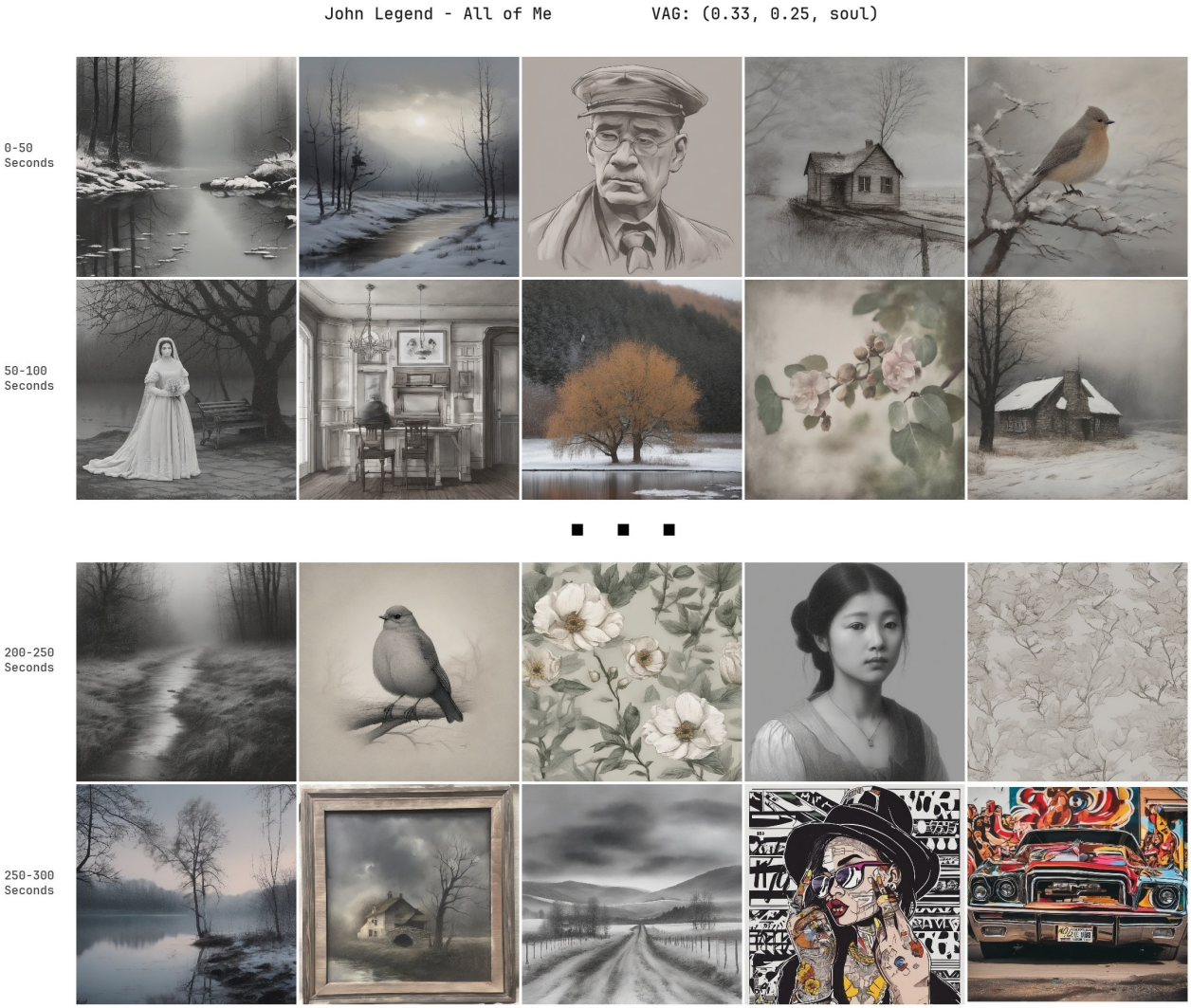

Figure 35: Generated images for a full song. Each image represents a 10-second audio clip and should be viewed sequentially, starting from the top-left corner and then going down line by line.

7.5 Repeated Themes

Throughout this thesis, I have examined thousands of generated images. Over time, certain recurring themes are noticeable. The model occasionally produces images that are not just closely related, but appear as simple permutations of each other rather than unique images. This behavior, while rare, occurs frequently enough to be noticeable. Examples of these repeated patterns are shown in Figure 36.

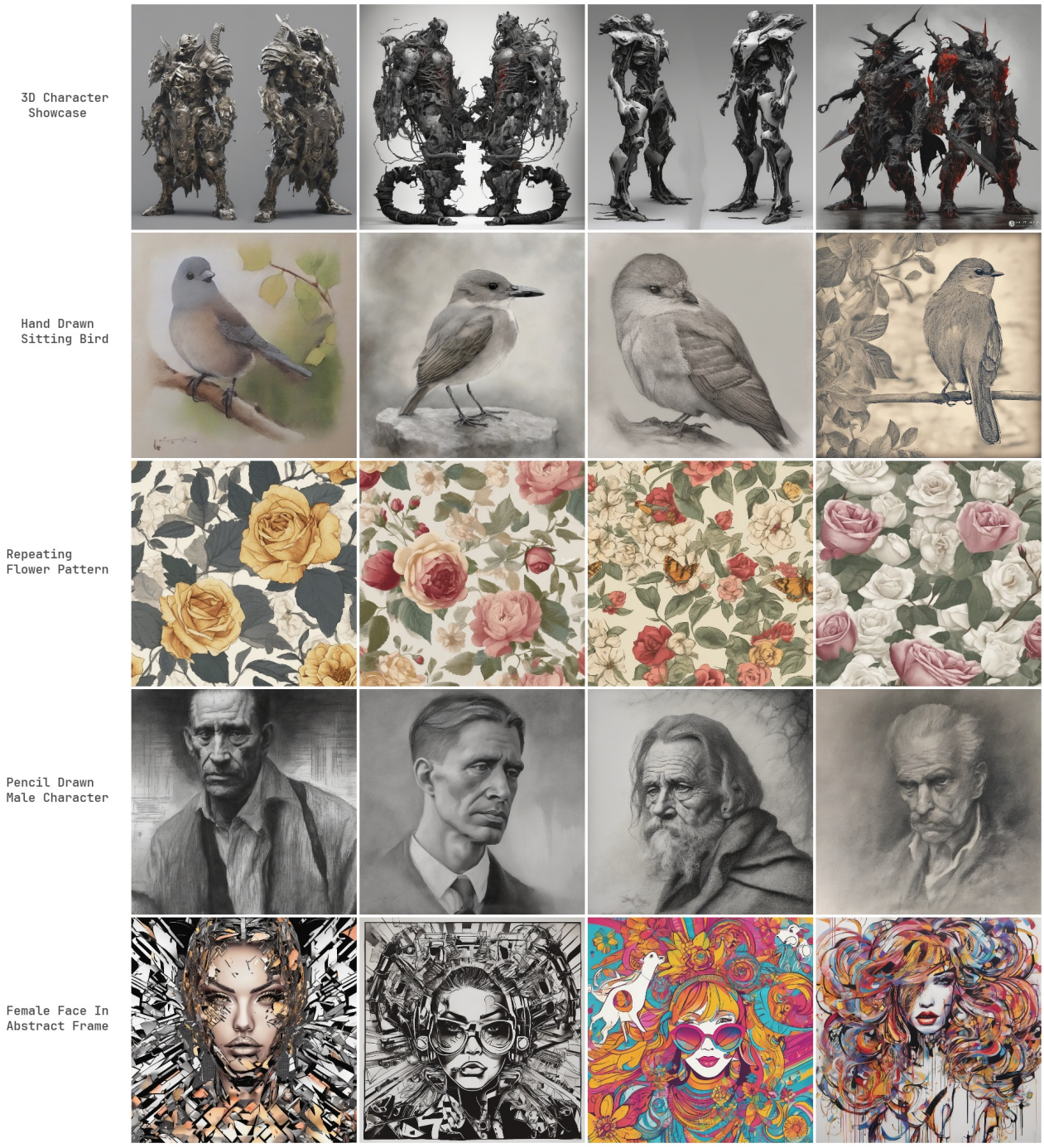

Figure 36: Examples of repeated patterns in generated images.

7.6 High Guidance Scale

At high guidance scales, the model generates images that are crisp, with more precise lines and well-defined shapes. However, these images also become less detailed and more cartoonish with exaggerated features. Figure 37 show examples of images generated with a guidance scale of 200. For comparison, most other images were generated with guidance scales ranging from 5 to 50.

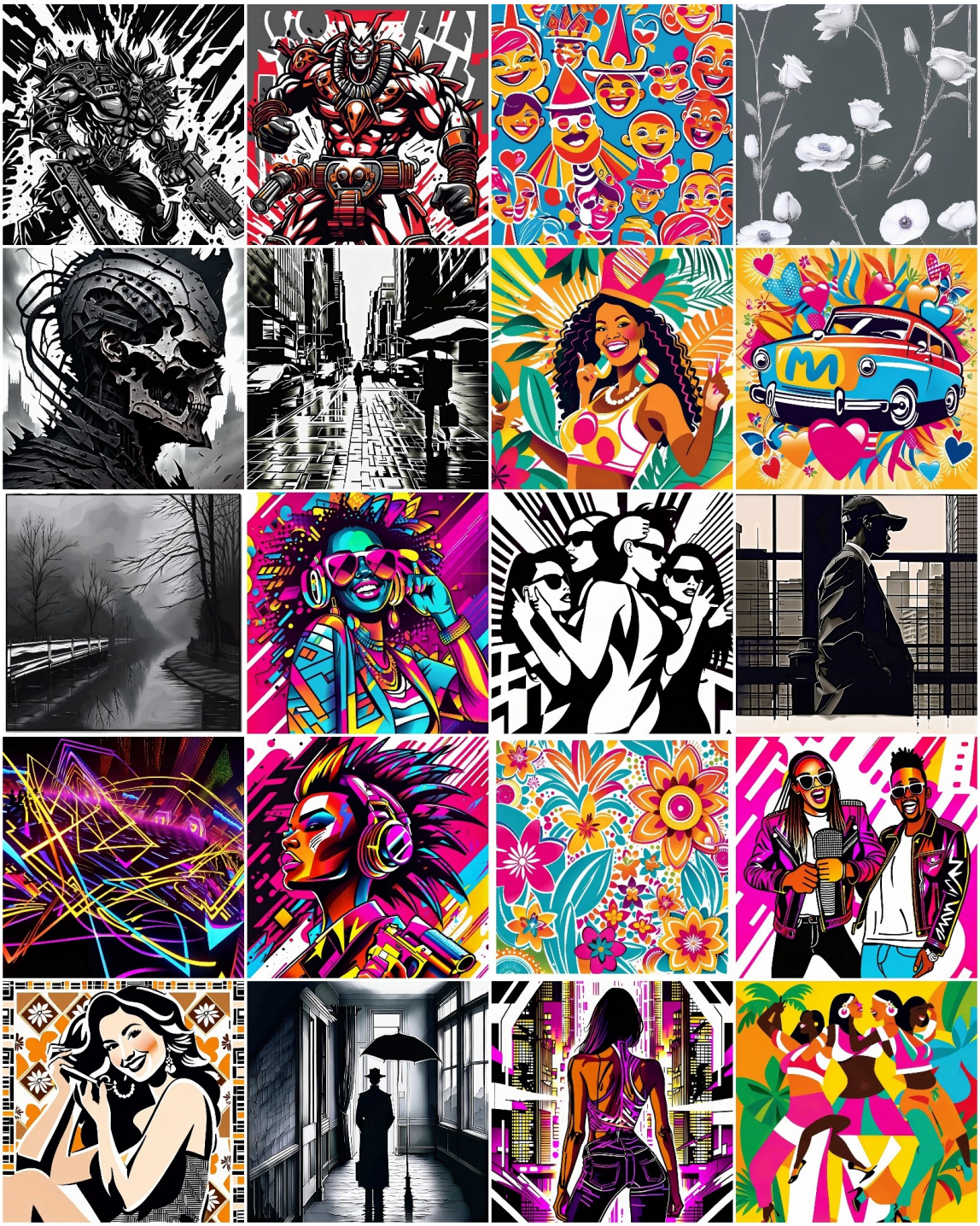

Figure 37: Examples of images generated with a guidance scale of 200.

8 Discussion

This section summarizes the main findings and challenges of my thesis. It starts by revisiting the 3 research questions. After that, it examines the limitations and potential areas for improvement. Finally, it offers suggestions for future research.

8.1 Research Questions

The thesis set out to investigate 3 core questions:

1. What human-centered heuristic can model the emotional and subjective relationship between music and imagery?
I developed the VAG-heuristic, which postulate that valence (V), arousal (A), and genre (G) features can partially explain the connection humans perceive between music and imagery. Through human experimentation, I validated the usefulness of the VAG-framework and my approach of generating song-matching prompts and images.

2. What approaches can be used to build a diverse and scalable dataset linking music to images with a focus on human preferences?
Using the VAG-framework, I created 3 datasets *BEATS-mini*, *BEATS-15K*, and *BEATS-400K*. These link music to images, enabling scalable multimodal AI-tasks. In addition to these 3 datasets, I designed and implemented tools for exploratory data analysis and human labeling. These can aid future research in the M2I domain.

3. What AI architecture can generate music-related images that align with human preferences?
I introduced a training approach for M2I synthesis, relying on VAG-labels from the BEATS-400K dataset. My approach works by aligning a custom audio encoder with CLIP embeddings for use in SDXL. The result is a fully functional, end-to-end M2I framework capable of generating images directly from short song clips. The pipeline only requires the raw audio waveform as input. I showed that the framework is capable of generating high-quality images in a consistent manner. Overall the results demonstrates the framework's potential for creative applications such as personalized art, dynamic music visualizations, and multimedia content.

8.2 Challenges

Given the very subjective nature of M2I mapping, the lack of a large-scale human experiment to validate the final model's performance is problematic. Without this, it is difficult to make definitive claims about the framework's ability to map between musical inputs and emotionally resonant images.

The creation of the BEATS datasets relied heavily on two external sources: Spotify for music metadata and ChatGPT for prompt generation. Both of these lack transparency, making it hard to analyze their exact workings. In addition to this, future attempts to expand on the BEATS datasets may be problematic if Spotify or OpenAI change their functionality.

The VAG-heuristic proved effective, but certain details about its implementation could be improved. For instance, its genre function treats genres too discretely. This means that similarities between related genres are ignored.

Computational limitations made training of the audio encoder challenging and ultimately meant only half of the BEATS-400K dataset was used. It also limited the number of training epochs, resulting in the model being analyzed before having shown clear signs of convergence.

Ablation studies were limited to small experiments. This increase the risk of conclusions being drawn from spurious correlations or half truths.

Preliminary findings suggested the system is highly sensitive to the exact design of the loss-function. Despite this, my limited resources prevented deeper analysis. Refinements to the loss function (e.g.

incorporating soft YY-labels and doing temperature tuning) could significantly improve the model's performance.

Another concern is the lack of content filters in my I2M framework. Because of this, there is a risk of generating inappropriate images, such as adult content or material violating intellectual property rights. Finally, the audio encoder only works with isolated 10-second clips and cannot handle longer songs. Adding support for arbitrary song lengths would make the framework more versatile.

8.3 Future Research

Future research could focus on many different areas to advance the M2I framework, I will provide some suggestions in this section.

Comparative analysis with other M2I or A2I models would provide valuable info about relative performance. Incorporating negative prompts could help filter undesirable outputs, such as adult content or mismatched themes. Integrating lyrics or other semantic data could enrich the emotional and contextual mapping between music and images. This could be used to expand the BEATS datasets to include culturally diverse inputs and a broader range of general features important for M2I mapping.

Improvements could also include developing more sophisticated similarity functions to account for e.g. overlapping genres. Extending the model to handle song with arbitrary lengths, rather than fixed 10-second clips, would increase its usefulness. Along the same line, incorporating conditional images or stylistic controls would offer a better user-experience. Additionally, researching methods to ensure sequential consistency could enable the generation of temporally coherent image sequences for use in e.g. music videos.

9 Conclusion

This thesis explored how to generate images from music, specifically focusing on aligning visuals with the emotions humans feel when listening to songs. The proposed framework is an M2I framework capable of generating images directly from short audio clips. The framework is end-to-end, requiring only the raw audio waveform as input. It is capable of generating high-quality images that manage to capture the energy, emotions and overall feel of songs from a wide range of musical genres. The system also maintains consistency across similar songs and allow for some controllability through a guidance scale parameter.

The proposed audio encoder model was trained using the novel VAG framework, which uses valence (v), arousal (A), and genre (G) to link music together with images. Experiments showed that human preferences align with the VAG model, supporting its use in M2I synthesis. In addition to the framework and its validation, I developed an interactive exploration platform and a self-hosted human labeling tool, which could be valuable for future research.

A key part of my work was developing the BEATS datasets: *BEATS-mini*, *BEATS-15K*, and *BEATS-400K*. Together, these datasets contain 400,000 music-prompt pairs, 15,000 music-image pairs, and a variety of metadata spread across a diverse set of genres and musical eras. To the best of my knowledge, these datasets are the first to explicitly attempt to link music with images while focusing on human emotions, making them a valuable resource for the M2I field.

The subjective nature of M2I mapping means that a model's ability to generate emotionally aligned images must ultimately be verified by humans for conclusive validation. As such, large-scale human studies are necessary to thoroughly evaluate the model's performance, particularly concerning its ability to create images that resonate with a broader audience.

In conclusion, this work demonstrates how AI can be used to connect music and images, creating richer, more engaging experiences for people. The findings highlight the creative potential of bridging the auditory and visual domains. This provides a foundation for interesting cross-modal applications, such as music-to-video synthesis and other multimedia systems.

10 Appendix

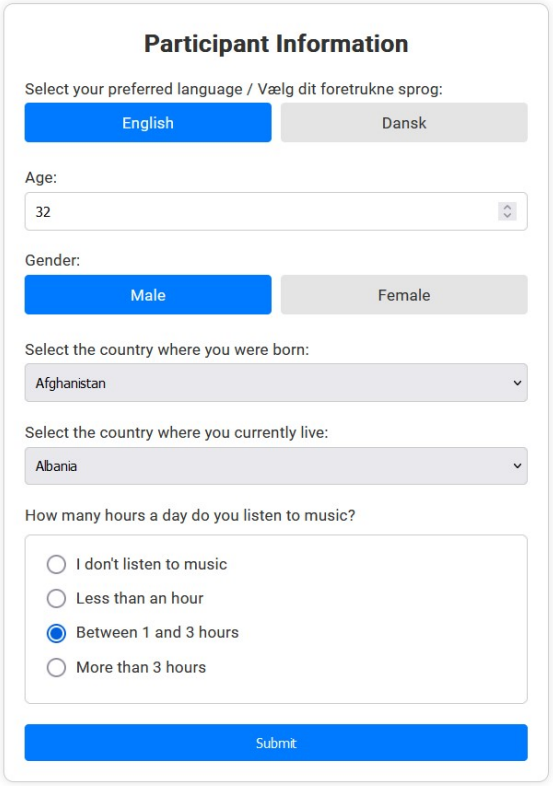

Figure 38: Initial questionnaire presented to users.

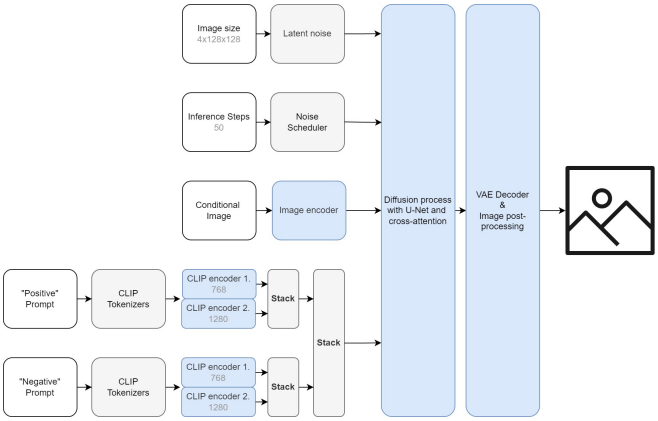

Figure 39: SDXL 1.0 full model overview. White boxes represent raw data, grey boxes indicate programming steps without AI, and blue boxes indicate AI components with frozen weights.

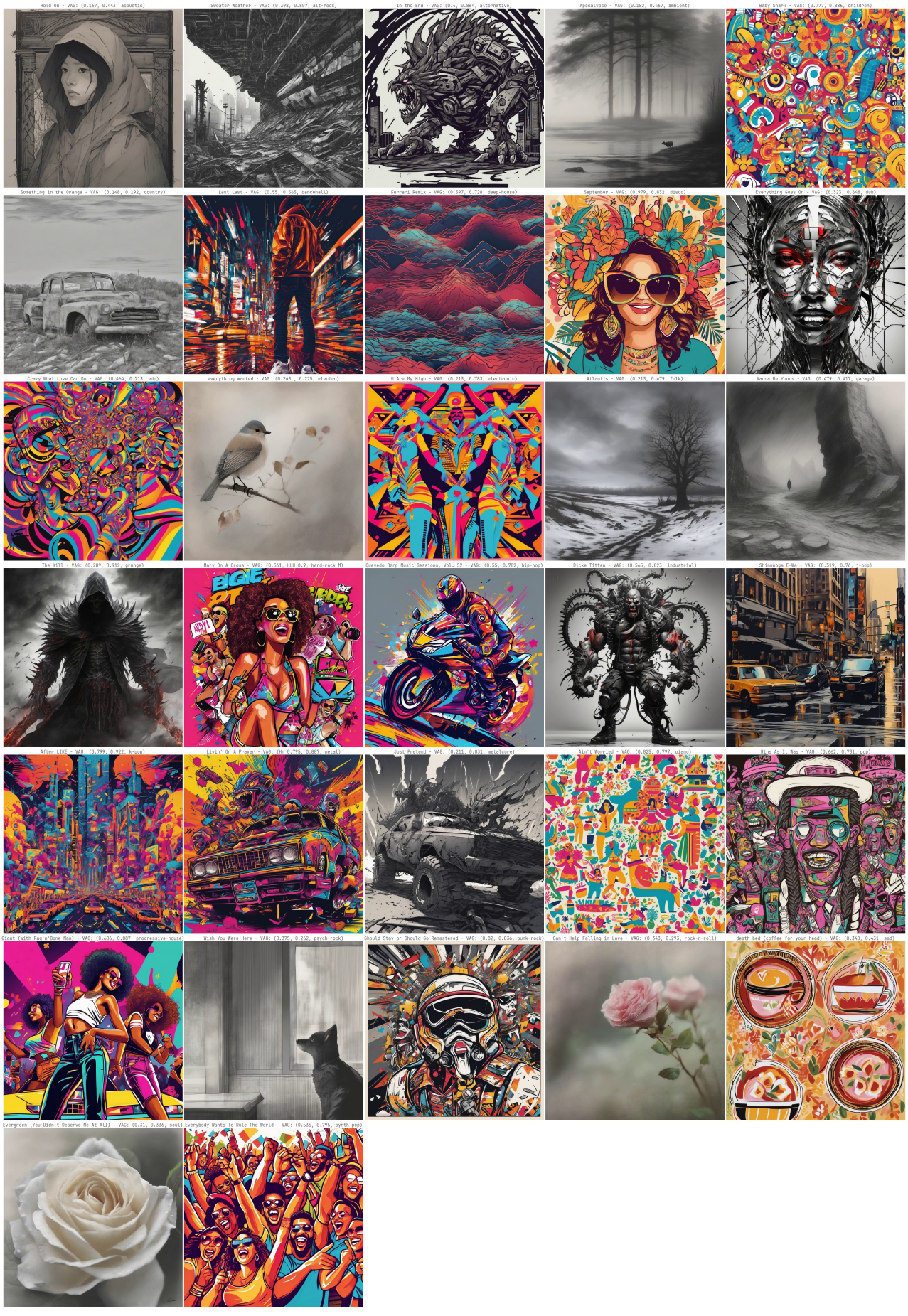

Figure 40: All 32 images used for the validation study.

References

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[2] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[3] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

[4] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[5] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[6] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[7] Kim Sung-Bin, Arda Senocak, Hyunwoo Ha, Andrew Owens, and Tae-Hyun Oh. Sound to visual scene generation by audio-to-visual latent alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6430–6440, 2023.

[8] Guy Yariv, Itai Gat, Lior Wolf, Yossi Adi, and Idan Schwartz. Audiotoken: Adaptation of text-conditioned diffusion models for audio-to-image generation. *arXiv preprint arXiv:2305.13050*, 2023.

[9] Yue Yang, Kaipeng Zhang, Yuying Ge, Wenqi Shao, Zeyue Xue, Yu Qiao, and Ping Luo. Align, adapt and inject: Sound-guided unified image generation. *arXiv preprint arXiv:2306.11504*, 2023.

[10] Guy Yariv, Itai Gat, Sagie Benaim, Lior Wolf, Idan Schwartz, and Yossi Adi. Diverse and aligned audio-to-video generation via text-to-video model adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6639–6647, 2024.

[11] Nikolaos Passalis and Stavros Doropoulos. deepsing: Generating sentiment-aware visual stories using cross-modal music translation. *Expert Systems with Applications*, 164:114059, 2021.

[12] Deepsing. `https://www.deepsing.com/`. Accessed: 2024-22-09.

[13] Modem. Op-z stable diffusion. `https://modemworks.com/projects/op-z-stable-diffusion/`, 2021. Accessed: 2024-22-09.

[14] Brian Man-Kit Ng, Samantha Rose Sudhoff, Haichang Li, Joshua Kamphuis, Tim Nadolsky, Yingjie Chen, Kristen Yeon-Ji Yun, and Yung-Hsiang Lu. Visualize music using generative arts. In *2024 IEEE Conference on Artificial Intelligence (CAI)*, pages 1516–1521. IEEE, 2024.

[15] Vivian Liu, Tao Long, Nathan Raw, and Lydia Chilton. Generative disco: Text-to-video generation for music visualization. *arXiv preprint arXiv:2304.08551*, 2023.

[16] Jun Zhan, Junqi Dai, Jiasheng Ye, Yunhua Zhou, Dong Zhang, Zhigeng Liu, Xin Zhang, Ruibin Yuan, Ge Zhang, Linyang Li, et al. Anygpt: Unified multimodal llm with discrete sequence modeling. *arXiv preprint arXiv:2402.12226*, 2024.

[17] YesChat.ai. Lyric-visualizer, 2024. Accessed: 2024-23-09.

[18] Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. Musicbert: Symbolic music understanding with large-scale pre-training. *arXiv preprint arXiv:2106.05630*, 2021.

[19] Cheng-Che Lee, Wan-Yi Lin, Yen-Ting Shih, Pei-Yi Kuo, and Li Su. Crossing you in style: Cross-modal style transfer from music to visual arts. In *Proceedings of the 28th ACM international conference on multimedia*, pages 3219–3227, 2020.

[20] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[23] Mehdi Mirza. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[24] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint d*, 2013.

[25] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *International conference on machine learning*, pages 1558–1566. PMLR, 2016.

[26] Andrew Brock. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[27] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[28] Chia-Hung Wan, Shun-Po Chuang, and Hung-Yi Lee. Towards audio to scene image synthesis using generative adversarial network. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 496–500. IEEE, 2019.

[29] Leonardo A Fanzeres and Climent Nadeu. Sound-to-imagination: Unsupervised crossmodal translation using deep dense network architecture. *arXiv preprint arXiv:2106.01266*, 1(2):3, 2021.

[30] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[32] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[33] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[34] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[35] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. Audioclip: Extending clip to image, text and audio. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 976–980. IEEE, 2022.

[36] Burak Can Biner, Farrin Marouf Sofian, Umur Berkay Karakaş, Duygu Ceylan, Erkut Erdem, and Aykut Erdem. Sonicdiffusion: Audio-driven image generation and editing with pretrained diffusion models. *arXiv preprint arXiv:2405.00878*, 2024.

[37] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023.

[38] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[39] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[40] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[41] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[42] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[43] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[44] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 32–42, 2021.

[45] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[46] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[47] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[48] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

[49] Yi-Hsuan Yang and Homer H Chen. Music emotion recognition, 2011.

[50] James A Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.

[51] Youngmoo E Kim, Erik M Schmidt, Raymond Migneco, Brandon G Morton, Patrick Richardson, Jeffrey Scott, Jacquelin A Speck, and Douglas Turnbull. Music emotion recognition: A state of the art review. In *Proc. ismir*, volume 86, pages 937–952, 2010.

[52] Xin Liu, Qingcai Chen, Xiangping Wu, Yan Liu, and Yang Liu. Cnn based music emotion classification. *arXiv preprint arXiv:1704.05665*, 2017.

[53] Xinyu Yang, Yizhuo Dong, and Juan Li. Review of data features-based music emotion recognition methods. *Multimedia systems*, 24:365–389, 2018.

[54] Paul Ekman and Wallace V Friesen. Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124, 1971.

[55] Pinaki Gayen, Junmoni Borgohain, and Priyadarshi Patnaik. The influence of music on image making: An exploration of intermediality between music interpretation and figurative representation. In *Advances in Speech and Music Technology: Proceedings of FRSM 2020*, pages 285–293. Springer, 2021.

[56] R Pluchik. A general psychoevolutionary theory of emotion. *Emotion: Theory, research and experience*, 1:3–33, 1980.

[57] Lisa Wilms and Daniel Oberfeld. Color and emotion: effects of hue, saturation, and brightness. *Psychological research*, 82(5):896–914, 2018.

[58] Laura-Lee Balkwill and William Forde Thompson. A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues. *Music perception*, 17(1):43–64, 1999.

[59] Patrik N Juslin and Daniel Västfjäll. Emotional responses to music: The need to consider underlying mechanisms. *Behavioral and brain sciences*, 31(5):559–575, 2008.

[60] Guilherme Francisco F Bragança, João Gabriel Marques Fonseca, and Paulo Caramelli. Synesthesia and music perception. *Dementia & neuropsychologia*, 9(1):16–23, 2015.

[61] Maharshi Pandya. Spotify tracks dataset, 2022.

[62] Stability AI. Stable diffusion xl 1.0 (sdxl 1.0), 2023.

[63] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.