Visually Indicated Sounds

Giorgio Demarchi
Massachusetts Institute of Technology
giodem@mit.edu

Abstract

This report introduces a semantic search-based system for video sound design, leveraging advancements in multimodal models. The system automates the matching of ambient sounds and sound effects into videos based on visual features. The methodology involves a scene splitting algorithm, a multimodal retrieval system based on ImageBind model architecture, and an object detection algorithm to automatically sync and adjust audio segments.

1. Introduction

I leverage recent advancements in multi-modal models [1] to create an semantic search-based system for video sound design. This tool can be used by post-production teams to reduce their time investment and increase their productivity. This tool can potentially help "unmute" the videos generated from the recently developed AI models (e.g. Sora), which do not seem to include sounds. I focus on ambience and sound effects, and exclude music.

The closest work in the literature is implemented by ElevenLabs as described in [4]. However, the lack of a video processing element and the purely text-based search leaves a lot of room for improvement. I address these limitations in this work by leveraging multimodal models for any-to-any retrieval.

1.1. Data Setup

In this work I leverage the Audioset dataset, open sourced by Google, specifically focusing on the strongly labelled subset of data [2]. Strongly labelled means that it is temporally labelled. The dataset contain 100k strongly labelled video-audio pairs, in the form of TSV files with a YouTube ID. Hence, I develop a pipeline (open sourced the toolkit on my GitHub) to parse the dataset labels, download the Youtube Video-Audio pair at the specified interval, and store everything on AWS S3. In order to adapt the labels to this system, I augment the labels dataset with GPT, in a multi-agent voting fashion, to classify sounds as Ambience, Music and Sound Effects, while also associating each sound with an sound emitter object. This will be helpful downstream in my pipeline. Finally, after running the model through ImageBind, I create an indexed vector database on Pinecone to enable quick and easy semantic search based on dot product.

2. Methodology

The Video-Audio matching pipeline involves several steps. First, the video is split in different scenes using splitting algorithm based on frames differences. Each scene is matched with a ambience sound, by embedding it into a 1024-dimensional vector and performing similarity search on the audios classified as "Ambience". For Sound Effects, if the object is present in YOLOv8 classes, an object detection algorithm regulates the Sync with the video in a fully automated manner. If the object is not known in object detection, then the audio is simply recommended to the sound designer for further synching.

2.1. Scene Splitting

A scene splitting algorithm implemented with the PySceneDetect library is applied to each video input, returning the list of scenes. The algorithm to split images is based on computing the differences in score between each frame, and detecting a scene change when the difference is below a threshold. By score, here we mean a linear combination of the difference between pixel hue values of adjacent frames, pixel saturation values of adjacent frames, pixel luma (brightness) values of adjacent frames, and calculated edges of adjacent frames.

2.2. Multimodal encoder

In order to process Video, Audio and even Text data, I leverage an open source multimodal model named ImageBind [1]. Imagebind is capable of mapping inputs from several modalities (images, video, audio, text, depth, thermal and IMUs) to a single vector space, enabling endless possibilities in any-to-any retrieval. As shown in their paper, ImageBind proved to perform even better than specialized models such as AudioMAE [3]. I forked the model codebase and made minor modifications to adapt it to my con-

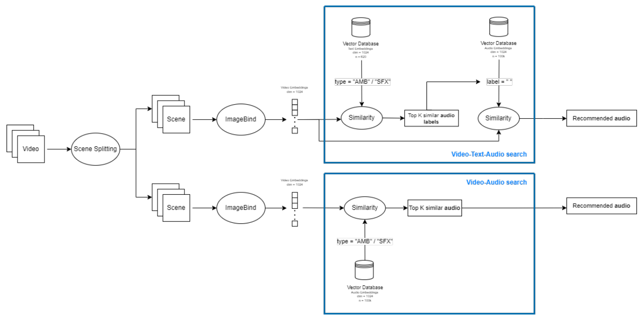

Figure 1. Visual explanation of the difference between Video-Text-Audio and Video-Audio semantic search

text, and imported the pretrained weights. For the purpose of Audio retrieval, I create a pipeline to extract labelled audio segments from my video dataset, infer with ImageBind to obtain the embeddings, and upsert them in a vector database created with Pinecone framework.

2.3. Semantic Search

Utilizing the multimodal encoder ImageBind, coupled with the AudioSet dataset, enables two different retrieval methods to achieve Video-Audio matching (see Fig. 1). First, I can compute a similarity score between the Video embeddings and the embeddings of the audio labels description. Given the top matching audio labels, we can filter the audio vector search for those labels, and compute the similarity score between video embeddings and audio embeddings. I call this approach Video-Text-Audio search. Alternatively, I can directly compare the video and audio features in the vector database, with a direct Video-Audio search.

Directly comparing the video features with the audio features should in theory be a less noisy, more effective method in terms of quality of output. However, the impact on latency is substantial: Given p labels and n audio in dataset, the Video-Text-Audio approach requires the computation of $\frac{n}{p} + p$ vector multiplications, while the Video-Audio approach requires n vector multiplications. As n and p grow, the difference in latency between the two approaches becomes more significant. In my setup, Video-Text-Audio requires 781 multiplications per search, while Video-Audio requires 100000 multiplications per search.

2.4. AutoSync with Object Detection

For the subset of sound effects labels that correspond to a sound emitter object, as identified by the GPT labels augmentation process, I have the opportunity to leverage light object detection algorithms to automatically sync a retrieve audio segment to the video. Specifically, three components can be controlled:

1. The start and end second of the sound effect component. This can be achieved by looking at the interval of the video where the object is detected.

2. Audio Gain. Using the area of the bounding box, we can compute audio volume differences in time. For example, if the video contains an aircraft coming towards the camera's direction, the bounding box of the object will be increasingly bigger, and the volume will be automatically increased proportionally.

3. Audio Panning. Using the X-axis position of the bounding box centroid, the audio is automatically adjusted to origin from the right or from the left of the video.

In order to implement this object detection-based auto sync method, I leverage a YOLOv8 architecture pretrained on the Object365 dataset.

3. Evaluation

Structuring a robust evaluation framework is essential for the success of the project, as I seek a quantitative comparison between the output quality of Video-Text-Audio search versus Video-Audio search. For this purpose, I will evaluate the output of each approach on a test set, by assembling the different Ambience sounds and Sound Effects that were matched by the algorithm, and compute a pass through the ImageBind model to obtain $\hat{y}^{(i)}$. I can then measure the MSE between the newly design audio vector and the original background audio $y^{(i)}$ of the video.

$$E = \frac{1}{n}\sum_{i=0}^{n}(y^{(i)} - \hat{y}^{(i)})^2 \tag{1}$$

4. Preliminary Results and next experiments

As of today, I have successfully handled dataset downloading, processing and migration to the Pinecone vector database. I have implemented the scene splitting algorithm and forked the ImageBind model. I have set up the code to perform Video-Text-Audio and Video-Audio similarity search, and qualitatively assessed the performance on a small set of data.

The next steps will be:

1. Expand the vector database from 20k vectors to 100k by running the migration pipeline to the entire dataset.

2. Develop and test the Auto Sync component with Object Detection

3. Develop the evaluation framework and code, and perform the comparison between the two semantic searches approaches

4. Develop a Streamlit Interface and test the end-to-end pipeline

References

[1] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all, 2023. 1

[2] Shawn Hershey, Daniel P W Ellis, Eduardo Fonseca, Aren Jansen, Caroline Liu, R Channing Moore, and Manoj Plakal. The benefit of temporally-strong labels in audio event classification, 2021. 1

[3] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen, 2023. 1

[4] David Chuan-En Lin, Anastasis Germanidis, Cristóbal Valenzuela, Yining Shi, and Nikolas Martelaro. Soundify: Matching sound effects to video, 2023. 1