Visually Indicated Sounds: Multi-modal Retrieval for Matching Sound Effects to Video Events

Giorgio Demarchi
Massachusetts Institute of Technology
giodem@mit.edu

Abstract

This report proposes a semantic search-based system for video sound design, leveraging advancements in multimodal models. This effort responds to advancements in AI video generation as well as the increasingly advanced requirements from the media creation industry, which demand AI-powered tools to create audio background tracks to couple with videos. Inspired by the work proposed by Runway [6], this end-to-end pipeline utilizes a sound database to automate the assignment of ambience sounds and sound effects to a video track based on visual clues. It first uses a scene splitting algorithm, and then proposes both an innovative multimodal retrieval system based on the ImageBind model [2], and a novel auto-sync heuristics to match a suggested audio segment to a start and end second. The qualitative results indicate a high potential for this methodology to be used as a co-pilot for video editors and sound designers to create powerful audio tracks in a fraction of the time previously required.

1. Introduction

OpenAI releasing their most recent and capable text-to-video generation model Sora [11], opened the doors to several applications in the media industry such as AI movie making and video creation. The video editing industry was already undergoing profound disruptions due to the advent of Generative AI technologies that empower operators with substantial leverage to increse their productivity and creativity. While most of the spotlight has gone to video and image generation, background audio design and generation is a critical component to deliver a complete suite of technologies.

This work leverages recent advancements in multi-modal models [2] to design and implement an end-to-end pipeline that receives a video as input, splits it into scenes, searches in a audio database the sound that most matches video events, and automatically recommends a start and end second for each audio track. The focus is restricted to ambience and sound effects, excluding music.

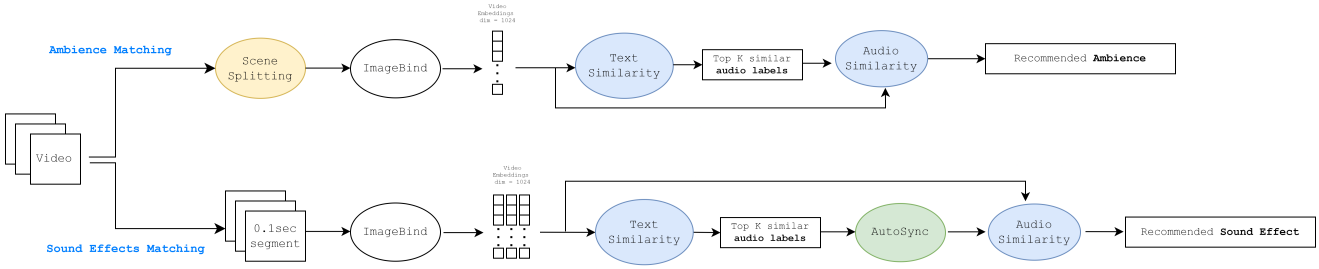
2. Related Work

Text-conditioned Audio Generation is a first possible solution for designing soundtracks and there have been many successful projects such as AudioGen [5], using an encoder-decoder model, and AudioLDM [7] using diffusion models. In order to utilize these systems for visually-indicated sound designs, a video-captioning model such as TimeSformer [1] needs to be applied. However, utilizing this two-stage system might suffer from information leakage. Meta proposed to directly condition the audio generation on video features in FoleyGen [9], with a similar approach to text-conditioning. While this work can create audio segments that are semantically very aligned with the video events, the key limitation is the output not being synchronized with the video timeframe. To overcome this limitation, the authors of [8] propose a diffusion-based system that is preceded by a pre-training with temporal and semantic contrastive losses.

In a practical application, however, deploying such large and heavy models may incur high costs and latency, with a possibly significant impact on user experience. Whereas an extensive and complete sound database is available, a semantic-search based approach can achieve similar outcomes with an order of magnitude less computational resources required.

Researchers at Runway proposed an end-to-end pipeline to match video events to sound effects using a sound library in Soundify [6], which inspired the work in this report. The proposed pipeline is based on comparing the features extracted from individual frames to an audio textual label, while sync-ing the audio with a object detection-based approach that uses bounding boxes coordinates.

This work builds on top of Soundify by proposing innovations on multiple aspects of the pipeline. Recent multimodal AI advancements open opportunities to improve the quality of the semantic search by directly comparing video (sequence of frames) to audio files or audio labels [2]. Fur-



(a) Simplified methodology overview figure

thermore, the object detection-based auto syncing is heavily limited by the sounds that can be attributed to emitter objects, and cannot generalize beyond those, while increasing latency due to the inference through the object detection model. A purely similarity-based approach to syncing audio segments, as proposed below, heavily reduces the latency.

3. Data Setup

The Audioset dataset [3], open sourced by Google, is leveraged to create a database of video-audio pairs. Specifically, this work utilizes the subset of the dataset that is temporally labelled, i.e. each audio label is associated with a start and end second. This dataset consits of 103,463 excerpts of YouTube videos of the duration of 10 seconds, annotated with 623 possible sound labels. Each excerpt contains several audio annotations at different, overlapping timeframes and the dataset is downloaded in the form of TSV files with a YouTube ID. Since the dataset comes with annotations only, a data pipeline is developed (link to GitHub) to parse the dataset labels, download the YouTube video-audio pair at the specified interval, and store them on an AWS S3 bucket.

As outlined below, the text labels quality is critical for improving the output of the search system, and hence a data augmentation with OpenAI's GPT4 [10] is carried out. Multiple LLM instances are fed the initial form of the audio labels, composed by "name" and "description", and asked to classify the sound as "Ambience", "Sound Effect" or "Music", as well as identifying a sound emitter object whereas possible. The opinion of three LLM instances is compared by majority voting to extend the dataset with the features "Type" and "Object", and ultimately enrich the semantic meaning.

Finally, a second database needs to be set up in order to operate the pipeline, this time in the form of vectors. Inference with the multimodal model ImageBind is executed on the audio database and on the labels dataset to create a Audio vector database and a text vector database stored on Pinecone. The Pinecone service enables quick and simple semantic search by hosting the database and taking care of the dot product computation and similarity matching.

4. Methodology

The Video-Audio matching pipeline involves several steps, and it changes for sound effects and ambience sounds.

For *ambience sounds*, we want to identify one highly similar sound and sync it with an entire scene. Hence, the video is split in different scenes using a scene detection algorithm based on frames differences. Each scene is matched with one ambience sound, by using ImageBind to embed the video sequence into a 1024-dimensional vector and performing similarity search on the audios classified as "Ambience".

Sound effects require a more sophisticated syncing algorithm. In order to identify the start and end second of each audio segment, the similarity score between each 0.1 seconds long video segments and the audio database is computed, allowing to extract a time series of similarity throughout the video. Then, an heuristic transforms this data into recommended audio segments by finding sequence of similarities that are both high and stable.

4.1. 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. The resulting score is bound between 0 and 255, and the default threshold is 27.

4.2. Multimodal encoder

In order to process Video, Audio and even Text data, the open source multimodal model ImageBind [2] is used. 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 Au-

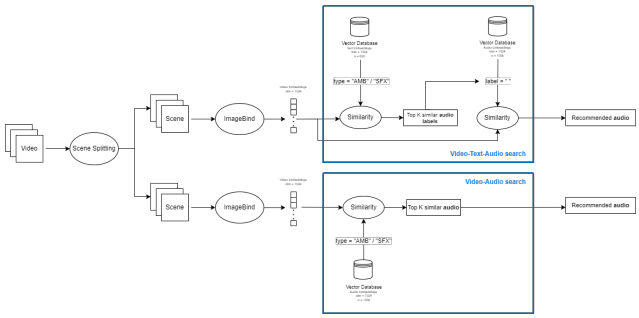


Figure 2. Visual explanation of the difference between Video-Text-Audio and Video-Audio semantic search

dioMAE [4]. The model codebase is forked to make minor modifications to adapt it to this context, and the pretrained weights are imported. For the purpose of Audio retrieval, a pipeline is developed to extract labelled audio segments from the video dataset, infer with ImageBind to obtain the embeddings, and upsert them in a vector database created with Pinecone framework.

4.3. Semantic Search

Utilizing the multimodal encoder ImageBind, coupled with the AudioSet dataset, enables two different retrieval methods to achieve Video-Audio matching. First, a similarity score between the Video embeddings and the embeddings of the audio labels description can be computed. 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. This approach can be named Video-Text-Audio search. Alternatively, one can directly compare the video and audio features in the vector database, with a direct Video-Audio search. A visualization of the difference between these two approaches is shown in Figure 2.

The first approach is selected because, while directly comparing the video features with the audio features should in theory be a less noisy and more effective method, 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. Furthermore, for automatically syncing sound effects, there are far more computations required as the similarity between each 0.1 seconds long video segment and audio needs to be computed.

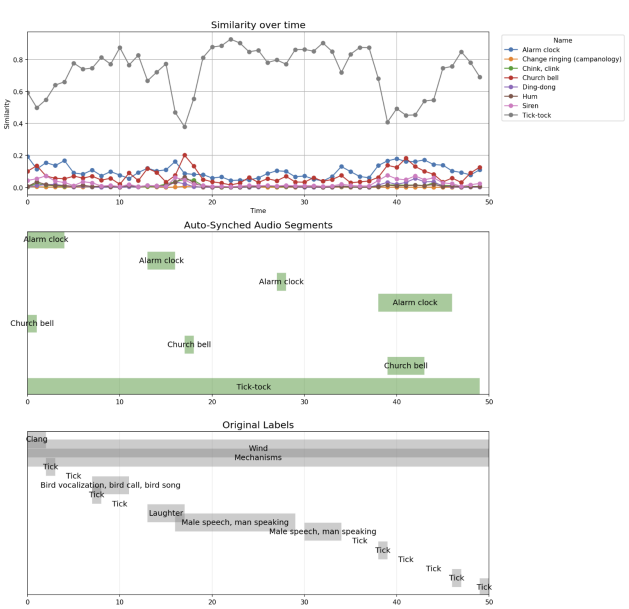
4.4. AutoSync

For the subset of sound effects labels that correspond to a sound emitter object, as identified by the GPT labels augmentation process, one possibility is to use an object detection algorithm and the corresponding bounding boxes for automatically sync the audio segments as done in [6]. The start and end seconds of the sound effect can be indicated by the series of frames where the object emitter appears. However, this approach is limited by the number of classes that can be detected, and hence it does not generalize well. This work investigates a fully generalizable heuristic that syncs audio segments purely based on semantic similarity over time.

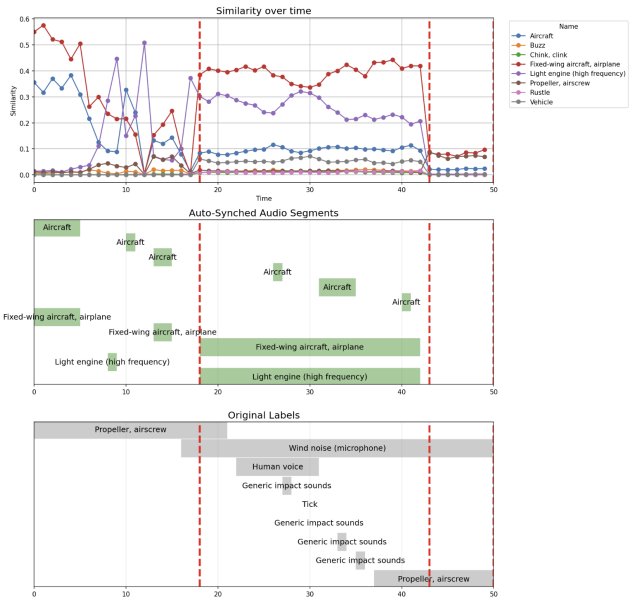
The input video is divided in 0.1 seconds long segments that are embedded into a 1024-dimensional vector with ImageBind and compared with the audio labels through dot product similarity. The dot product undergoes the softmax operation to scale the values to sum up to 1. This process results in a table of similarity scores for every segment (see Fig 3). In order to separate noise from significant information, a threshold-based method is needed. Every consecutive set of video segments where there is a sound that is consistently significant and stable, is considered a sound effect track. Hence, a sound effect track is defined as a set of segments where the similarity is higher than a threshold t and the similarity value stays within $d\%$ of the mean value. These hyper parameters, t and d, need to be optimized with a grid search according to a loss function. A proxy loss function that compares the assigned audio segments to the original label is constructed to penalize the difference between start/end seconds, as well as the number of correctly identified audio labels. The resulting optimal hyper parameters are $t = 0.1$ and $d = 0.5$.

5. Results and discussion

The best way to qualitatively evaluate the output of the end-to-end pipeline is by comparing the composed layout of sound effects with the original labels. Two examples of the visualizations required for this analysis is displayed in Figure 3. A major advantage of this approach compared to existing alternatives is the drastically reduced latency. Specifically, the model ImageBind is highly optimized for batch processing and can embed 300 video segments in less than a second. Modern vector database such as Pinecone are also highly optimized for speed, enabling efficient vector search in milliseconds even when the vector database is composed by hundreds of thousands of samples. These characteristics make this pipeline a valuable tool for video editors and sound designers, by enabling them to increase their productivity with fast suggestions.



(a) An example output



(b) Another example output. Red lines representing different scenes.

Figure 3. Pipeline complete output

References

[1] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding?, 2021. 1

[2] 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

[3] 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. 2

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

[5] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation, 2023. 1

[6] David Chuan-En Lin, Anastasis Germanidis, Cristóbal Valenzuela, Yining Shi, and Nikolas Martelaro. Soundify: Matching sound effects to video, 2023. 1, 3

[7] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D. Plumbley. Audioldm: Text-to-audio generation with latent diffusion models, 2023. 1

[8] Simian Luo, Chuanhao Yan, Chenxu Hu, and Hang Zhao. Diff-foley: Synchronized video-to-audio synthesis with latent diffusion models, 2023. 1

[9] Xinhao Mei, Varun Nagaraja, Gael Le Lan, Zhaoheng Ni, Ernie Chang, Yangyang Shi, and Vikas Chandra. Foleygen: Visually-guided audio generation, 2023. 1

[10] OpenAI. Gpt-4 technical report, 2022. 2

[11] OpenAI. Sora: Creating video from text. https://openai.com/sora, 2024. Accessed: 2024-05-13. 1