Good, Fast, Cheap: Pick One
Transfer Learning for Multimodal Audio Retrieval

Christian Clark, Nikhil Ravi Krishnan, Walter Becerra
Georgia Institute of Technology, 225 North Ave NW, Atlanta, GA 30332, USA
christianclark@gatech.edu, nikhilrk@gatech.edu, wotoya3@gatech.edu

Abstract

Current search methods for music rely on genre tagging systems, searching for a specific title or artist, and human curation. This makes it difficult for users without domain specific knowledge to discover new music and makes it difficult for genres not categorized effectively in existing systems (non-Western music, genres of music that are defined primarily by performer, producer, or composer) to be discovered by most audiences. Neural network based methods have been applied to audio search and retrieval problems but there are a few issues with current approaches: 1. It is difficult to get access to large enough datasets to train robust models and 2. previous studies fail to take advantage of state of the architectures that allow for more robust semantic representation of audio data. For this project, we tested the impact of using three different image and audio neural network embedding architectures on audio retrieval from natural language queries, trained using contrastive loss. We compared the difference in performance for existing pretrained image and audio model architectures and these architectures after fine tuning on spectrograms (visual representations of audio data), and found that under our experimental conditions these models do not have comparable results to models explicitly trained on large audio datasets for multimodal retrieval tasks.

1. Introduction

There is a large amount of audio data that exists, both online and in personal libraries. The subscription-based music service Spotify alone has over 80 million unique tracks [1]. Unlike images, which can be viewed fully almost instantaneously and examined for desired content, music takes a greater amount of time to search due to its dependence on the time domain. It is vital to have effective methods for the search and retrieval of relevant audio in order to minimize the time needed to find relevant audio information. While many organization systems categorize music by genre, audio contains semantically rich content that can make it difficult to classify to a fixed number of categories. While many search methods try to account for this fact by allowing users to search by artist, song name, and even lyrical content [12], these do not fully account for the full semantic content of music and audio, which can lead to the retrieval of irrelevant results [7]. We aimed to address these issues by utilizing deep learning models trained on natural language captions describing audio clips using constrastive loss in order to improve their ability to retrieve relevant pieces of audio from natural language descriptions.

In general, working with audio data has some unique challenges. There are fewer audio datasets than for modalities like vision and text; for example, the Papers With Code listing of datasets has more than 4 times as many vision datasets as audio datasets [2]. Audio files are also very large. The dataset used for our project, which contains approximately 30 hours of audio, took up more than 10 GB of space even in a compressed format [5]. Many datasets, such as AudioSet, are provided as audio "features" rather than audio to deal with this issue [23], but this requires preprocessing data consistently when utilizing models trained on these features for effective inference. These opposing challenges- audio learning is a resource intense process but it can be difficult to find or have sufficient storage for quality resources- would mean a method to reduce the amount of training data needed for audio models to perform effectively would be very valuable. Transfer learning is a popular strategy in many areas such as visual and language learning, so we decided to test if performing fine tuning on models from another modality (vision instead of audio) with our modality could improve performance on our multimodal retrival task.

2. Related Works

Accurate caption retrieval is an example of a zero-shot learning task, one where there is no labelled data for a given class in the the training data; however, models can learn these unseen classes or effectively retrieve relevant re-

sults by using auxillary properties of the data [26]. Past approaches in zero-shot learning related to images have involved training models to directly project an image into an existing word or other semantic embedding space, such as Word2vec, which contain language-based information about class properties which can be transferred to unseen classes [15]. More recent approaches also include projecting semantic characteristic vectors of multiple modalities into a new jointly aligned space [27]. There have been promising results in some areas for models such as CLIP [21], which utilizes a joint image and language embedding space, and MusCALL [11], which also uses a joint embedding space, but does so for audio and text rather than images and text. While promising, research in this area has yet to examine the impact of different embedding models used to capture the semantic representation of each modality on zero-shot retrieval. For example, vision transformers [4] perform well on many other vision tasks such as classification [19], and could perform well on visual representations of audio data like spectrograms. Other vision architectures have successfully been utilized on audio related tasks, especially when they are trained on large audio datasets [9]. We build on this prior research by studying the impact of a variety of existing visual and audio signal processing model architectures on the performance of zero-shot retrieval of audio clips from associated natural language captions.

Transfer learning, or learning representations using a large scale data set before training weights at later layers of a network on a smaller, more task specific dataset, has been applied in a variety of areas from vision [16] to language [22]. This approach can allow models to make significant performance gains and reduce overfitting when utilizing a small dataset, as well as to generalize to new tasks. Transfer learning can have a variety of goals, some of the most common being utilizing a model's past general capabilities to generalize to a new, more specialized dataset ("transductive" transfer learning), utilizing these capabilities on a new task ("inductive" transfer learning), and utilizing a model on an unsupervised task (unsupervised transfer learning) [17]. While transfer learning is generally done within the same modality, there are past examples of utilizing existing architectures from one modality to improve performance on a task in another: for example, gradients from 2D image generation can be applied to improve 3 dimensional generation [20] and models originally trained for image classification can effectively generate text [24]. For our project, we aimed to determine whether architectures previously trained on another modality (images) could effectively transfer to the audio domain with additional fine tuning.

3. Technical Approach

Our model has an architecture highly similar to that of the popular CLIP model [21]. It consists of two encoders, $enc_a(\cdot)$ for the audio modality, and $enc_t(\cdot)$ for the text modality, which are then aligned by projecting each into the same dimensional embedding space using a separate fully connected network layer for each of the encoders, $f_a(\cdot)$ for the audio modality, and $f_t(\cdot)$ for the text modality. As stated in the introduction, we utilize pretrained vision and audio models as our audio encoders to attempt to reduce the amount of training data and time needed to perform effectively on our task.

Our ideal model aims to minimize the distance between the two L2-Normalized embeddings that represent the same item i in these two modalities (the audio, $s_{a,i}$ and its caption, $s_{t,i}$). We do this by minimizing the InfoNCE loss [25], the mean categorical cross-entropy for the correct prediction for each item, given N items in a batch:

$$\mathcal{L}_{InfoNCE} = -\mathbb{E}_N \left[log \frac{exp(s_{a,i} \cdot s_{t,i})}{\sum_{i \in N} exp(s_{a,i} \cdot s_{t,i})} \right] \quad (1)$$

We chose to examine the capabilities of three pretrained models: two vision-data trained architectures and one audio-data trained architecture. Residual networks have been tested previously in other research in audio-caption retrieval [11], so we felt like this would be a good baseline to test the capabilities of our technique. Transformers are an architecture-class that has been successfully been applied to a wide variety of areas, including language and vision, so we also decided to test the capabilities of a vision-transformer model [4]. Finally, convolutional architectures specifically trained on large audio datasets had shown good previous performance on audio specific tasks, so we wanted to test one of these architectures. We chose to use the "CNN-14" variant of PANN (Pre-Trained Audio Neural Networks), which has been recently popularized for many audio tasks [9].

We chose to use the same language architecture for each of the three audio encoding models tested, using the RoBERTa language transformer model [10]. In addition to being more comparable to past research in this area which also used this language architecture [11], the model allows for fine tuning on unlabeled text data such as our audio captions.

All of the models we used, including the overall framework for both encoders and projection heads, were programmed using the Pytorch library. We were able to download pretrained weights and image transformations for the Vision-Transformer and ResNet models from the torchvision library and the RoBERTa model from the hugging-face library. The PANN model code, also written in Pytorch, came from its original GitHub repository, while its weights were downloaded from Zenodo. The two vision architectures we selected have weights resulting from training on the ImageNet dataset, while PANN was originally trained

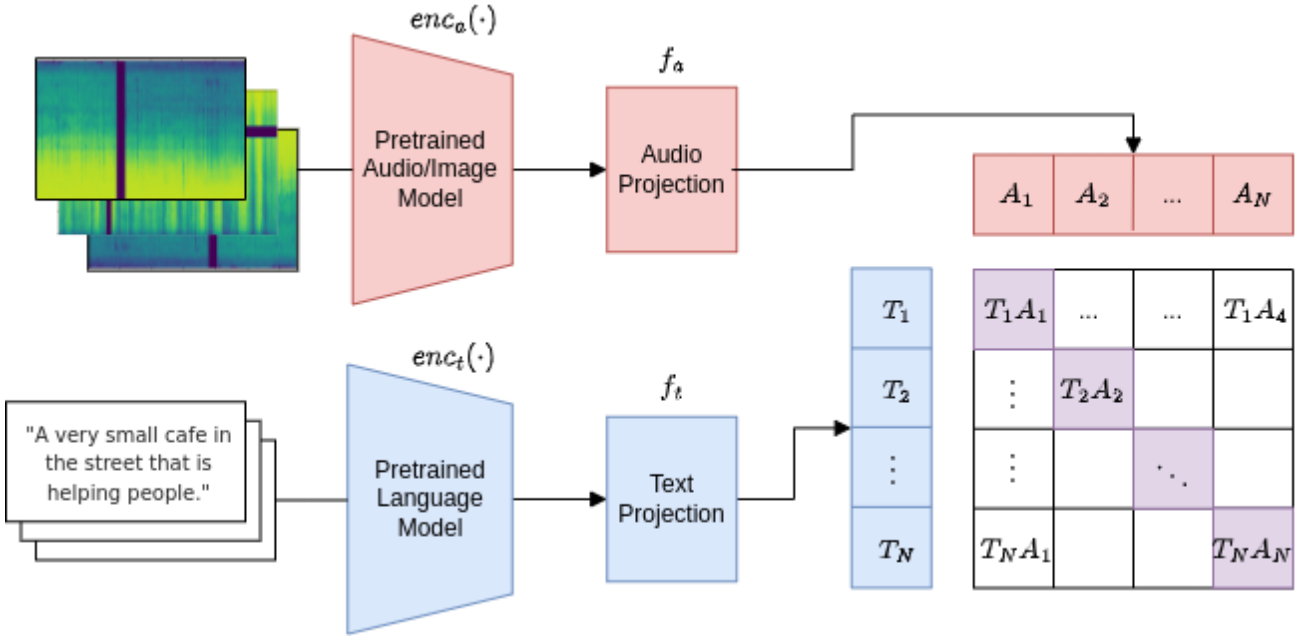

Figure 1. Our approach consists of two pretrained encoders that are each aligned in a joint space by a fully-connected projection head. The loss function aims to minimize the difference between the embeddings of corresponding items in each batch.

on the AudioSet dataset.

As an input to our models, we use spectrogram representations of each audio clip. In addition to containing both time and frequency data, these represent sound in the form of a 2D "image". We initially considered using mel-spectrograms, which rescale frequencies to be a scale in which they are perceived as equidistant by human beings; however, we ultimately decided against this because we felt it may remove informative data about certain sounds, especially at the higher range of the frequency spectrum. Additionally, for the previously trained image architectures (ResNet and Vision-Transformers) we stack multiple of these images in order to use their multiple input channels. To ensure our inputs are of fixed length, we pad each clip to be the same length (40s) with silence (0 values) on both sides of the input, centering the audio. Similarly, we pad and truncate captions to be the same length (30 tokens) with specially designated padding tokens that are ignored by the attention mechanism of the text embedding model.

To improve the robustness of the model and increase the diversity of the training set, we also utilized various audio and language augmentation strategies. For audio data, this included masking timed areas of spectrograms, masking specific frequency bands, and time stretching. For our caption data, this included swapping words for common misspellings and using synonyms and hypernyms. We applied language data augmentation to 20% of the items in each training batch and audio augmentations uniformly distributed across a batch as originally proposed in SpecAugment, a technique to support robust speech recognition models [18].

For each type of audio encoder, we tested the performance of each with only training the projection head (fully connected alignment layer) as a "baseline" approach and training this fully connected layer plus additional layers in the pre-trained audio encoding model (a fine-tuning approach). Depending on the original task for a given model architecture, we used the outputs at a specified layer of the model as an output embedding from that model. The original training task, embedding layer, and embedding layer size for each of the audio encoders we used is given in table 1.

In general, training our architecture with fixed weights in the two encoders was simple; we only had to propagate gradients through the two fully connected layers to optimize their parameters. In order to fine-tune the encoding models for better performance on our specific task, we update the weights of both the projection layers and the final layers of each encoder modules. In the ResNet and Vision-Transformer models, we froze all of the model weights for fine-tuning except the final two groups of modules in the encoder. In the PANN model, we trained all layers, with earlier layers having an extremely small learning rate, and the later layers having a learning rate comparable to the other models'.

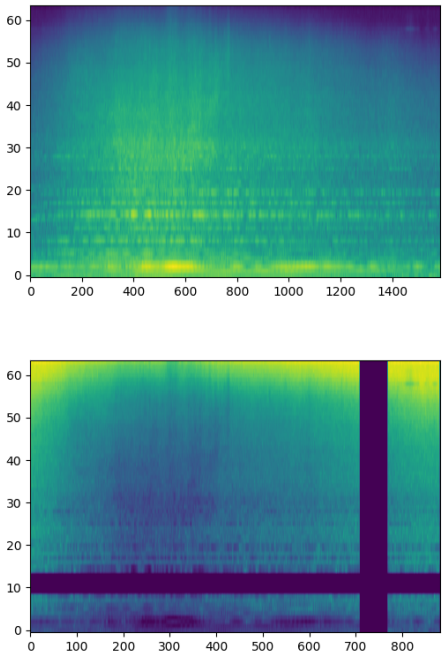

Figure 2. An example of a spectrogram before and after augmentation. Both frequency and time masking have been applied along with time stretching.

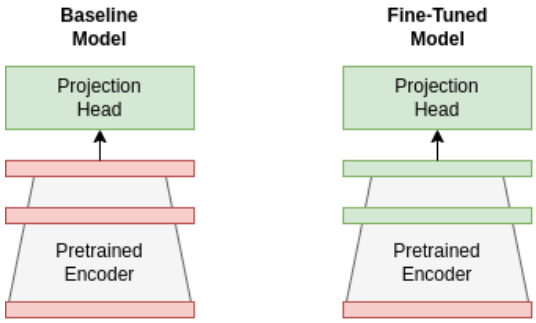

Figure 3. In both the baseline and fine-tuned models, the projection head is trained. Other layers are only trained in the fine-tuned model.

4. Dataset

To test our models, we utilized the Clotho audio captioning dataset, which contains 4,981 audio clips of varying lengths (15-30s) and 24,905 associated captions (5 per clip) [5]. This dataset is especially useful due to its varying textual representations of different sounds based on human perception rather than ground truth video data. Other captioning specific datasets we considered, such as the AudioCaps dataset [8], contained only one caption per audio clip and came directly from videos where labelers could be influenced by additional visual cues. We also considered some very large audio datasets, such as AudioSet [23] and the Million Songs Dataset [3], which contained a greater amount of training data. Neither of these were well suited for our task. AudioSet contains a large number of raw audio samples, but these do not contain associated text data beyond a categorical class, which are generally too short to be useful for fine-tuning language transformer models. The Million Songs Dataset, which also does not have associated captions, is primarily distributed as audio "features" rather than raw audio due to copyright constraints, which gave us concerns about a model trained on this dataset's ability to generalize to raw audio predictions. As a result of these concerns, we decided to only train our models on the Clotho dataset, and increase the number of training examples through data augmentation.

Audio Embedding Model	**Embedding Layer**	**Layer Size**	**Original Training Task**
ResNet-50	FC Layer	1000	Image Classification
Vision-Transformer (Base 16x16)	Encoder Layer 11-MLP	768	Image Classification
PANN CNN-14	AudioSet FC Layer	2048	Audio Classification

Table 1. Details about each of the embedding models tested for our experiments.

Dataset	**# of Items**	**# of Captions per Clip**
AudioSet	2M	0
Million Songs Dataset	1M	0
AudioCaps	46K	1
Clotho	24K	5

Table 2. Basic characteristics of datasets we considered for this project.

We are used the default split of this dataset for our experiments (60% training, 20% evaluation, 20% testing). Because this dataset contains a relatively small number of samples, we increased the diversity of the training set through data augmentation as previously described in the technical approach section.

5. Experimental Setup and Results

To determine whether pretrained models trained on other modalities (vision) or tasks (audio classification) could

Statistic	Training	Evaluation	Testing
# of Audio Clips	2989	996	996
# of Captions	14945	4980	4980

Table 3. The number of samples in each partition of the Clotho Audio Captioning dataset.

effectively perform well on audio-caption retrieval, we trained multiple models of each type with different hyperparameters, and measured their performance using three metrics commonly used for search and retrieval tasks: MAP (Mean Average Precision) @ K, Recall @ K, and Mean Reciprocal Rank. Each of these require taking the cosine similarity between the model's output audio embeddings and language embeddings, then ranking them from highest to lowest. For mean reciprocal rank, we take the reciprocal of the rank of the cosine similarity between the two embeddings representing an audio and caption pair. For mean average precision and recall @ K, we also rank these cosine similarity values, and find the number of relevant items in the top K ranked items (precision) and the proportion of relevant items that are recommended in the top K ranked items (recall). We consider any of the 5 spectrograms representing the same audio clip for a particular caption (or vise versa) as "relevant". All three of these metrics are averaged across all items in our dataset. Both MAP and Recall are general metrics that can be performed on a set of the first K items. For our specific case, we used K=10. In an ideal setting, all of these metrics are as close to 1 (100%) as possible.

For our three models, we explored different values for two primary hyperparameters during training: batch size and starting learning rate. Both of these had an impact on values ultimately backpropagated through our models- the learning rate by directly multiplying with the change in gradient and the batch size by influencing the loss values (the larger diversity of samples associated with a larger overall number of samples appeared to increase the overall loss in each batch). A sampled list of trials, with their training and validation loss values and hyperparameter values are listed in table 6. All of these models used the Adam optimizer for training. Our parameters were input to our model through a configuration file and tracked using the service Weights and Biases, which also allowed us to track loss curves and the status of models as they trained. The batch size appeared to have made a greater difference to the model's validation and training loss than the initial learning rate. This may have been because we used a learning rate scheduler that decreased the learning rate on a plateau, which happened very quickly with some of our models.

A hyperparameter we were unable to explore during this project was the different sets of layers trained in each model during fine tuning; this is a variable that would likely be interesting to explore in future research.

Figure 4. The training loss of a model that quickly hit a plateau during training.

Model	Batch Size	Starting LR	Training Loss	Validation Loss
ResNet	32	1e-4	5.593	3.524
	256	7e-2	7.161	5.541
ViT	16	1e-3	6.575	5.493
	32	1e-3	8.172	6.909
PANN	8	1e-4	6.498	4.056
	32	1e-4	8.064	5.435

Table 4. A sample of hyperparameters tested and their impact on model training and validation loss.

We aimed to keep these hyperparameters as consistent across our models as possible; however, one of the issues we faced as students was getting easy access to powerful hardware for model training and evaluation. As a result of time and financial constraints, our training was done on a mix of machine settings, all available as preset Deep Learning VMs on the Google Cloud and Google Collab platforms with a single or single pair of GPUs. This is one reason for different batch sizes being used when testing hyperparameters and varying training speeds across models. A high number of model parameters also made it challenging to use a large batch size for training; to compute embeddings for each set of items, both the RoBERTa and audio encoder model weights must be loaded into memory. For these reasons, to ensure consistency in evaluation, we loaded all of our pretrained models onto the same machine for evaluation which allowed us to use a uniform batch size, with one exception (the PANN audio encoder based model) due to an out of memory error.

After training and adjusting the hyperparameters stated previously, we evaluated the models of each type (baseline and fine-tuned) on our test set. The results for our selected metrics for each model from are listed in table 5. These are compared to two other existing models for audio-caption retrieval that were trained solely on audio examples and tested on their own training sets.

Model	MRR	MAP@10	R@10
ResNet (Baseline)	0.0411	0.84%	7.71%
ResNet (Fine-Tuned)	0.0391	0.85%	7.86%
ViT (Baseline)	0.0294	10.92%	8.21%
ViT (Fine-Tuned)	0.0297	13.15%	7.84%
PANN (Baseline)	0.0262	3.09%	19.42%
PANN (Fine-Tuned)	0.1294	3.22%	31.42%
MusCALL	N/A	36.0%	63.3%
AudioCLIP	N/A	30.79%	N/A

Table 5. The performance of our models on the Clotho audio captioning dataset's test set compared with existing audio-caption retrieval models. The results for the comparison models come from their own datasets, rather than Clotho.

6. Conclusion

Overall, none of the models we trained for in our experiments appeared to perform well. In comparison, models fully-trained on audio have much better recall and mean average precision @ K scores; for example, the MusCALL model, which uses the ResNet and RoBERTa architectures for two encoders aligned with fully connected projection layers has superior performance over our model with a ResNet encoder pretrained on image data. It appears that neither weights from image nor audio classification pretraining effectively transferred well to audio related retrieval task in our specific case.

We have a few ideas about why our models were unable to effectively generalize previously learned audio or image information to a new task or domain. One is a lack of training examples. The models that performed well on this task previously (MusCALL and AudioCLIP) benefited from a much larger set of training data and greater computing resources. For example, AudioCLIP was able to align in an additional modality (images) with a large amount of training examples provided in the form of videos in the previously mentioned AudioSet dataset. The authors of MusCALL had access to a proprietary dataset with 250,000 examples of captioned musical audio clips. Our dataset was about 1/10th the size of MusCALL's, and we may have seen more similar performance with a similar amount of training data. As a continuation of the research done for our project, we could do additional training using the AudioCaps dataset; while less useful to get recall and precision metrics using our current methods (because there is only 1 relevant caption per audio clip), we could easily integrate it into our training process.

Model	# of Samples in Dataset
MusCALL	250K
AudioCLIP	1.8M
Ours	25.4K

Table 6. Comparable models for this task and their dataset sizes.

Another reason our models may have been unable retrieve audio captions effectively, despite pre-training, is the representational differences between the audio and visual domains. The digital representation of images is limited to 256 discrete values in each pixel color channel, and more generally, the visible spectrum spans only about 400 nm [13]. In humans, the audible spectrum spans a range of over 10,000 Hz [14]. When translating this into spectrogram images, this means that "images" of sound are likely to have a different distribution of values than standard images. However, testing this conjecture would require a greater analysis of visual and sound perception and representation more generally.

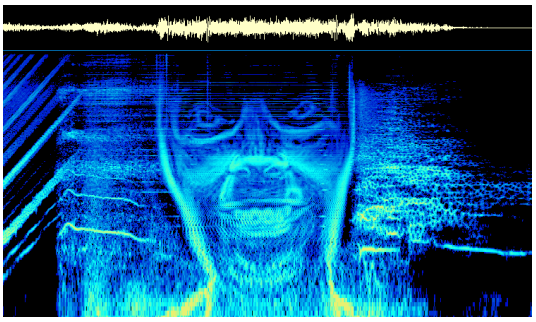

Figure 5. Spectrograms that resemble images are highly uncommon. This rare example from Apex Twin's "Windowlicker" used a specialized synthesizer (MetaSynth) to create an audio sample with this unique spectrum.

A way to experimentally see whether a model's original training modality had a large effect on audio caption retrieval modality, would be to try training additional audio-based architectures on this task, such as ESResNet [6] or

Group Member	Responsibilities (Code)	Responsibilities (Other)
Christian	ViT model, model configuration and evaluation	Poster design, related works research
Walter	Data augmentation, model configuration for Google Collab	Collecting code for submission, running model evaluation for experiments
Nikhil	PANN model, training loop, dataloader	Running model evaluation for experiments
All	ResNet model, metrics, bug fixes	Report

Table 7. Group members and their responsibilities

other PANN architectures, with a larger training set, and seeing if they have improved performance over those tested in our project.

In conclusion, while the models tested for our project did not appear to perform well when retrieving audio from captions, there are changes that could be made to these models nad our training process to possibly improve their performance (using additional training data, training additional layers in the audio encoders, training layers in the text encoders). More detailed analysis is required to effectively understand the differences in distribution between values in images and audio spectrograms and make a more informed judgement about the effectiveness of trying to transfer learned parameters from one modality to the other. Despite our models' somewhat disappointing performance, the members of the group working on this project learned a great deal about audio processing, machine learning project infrastructure, hardware constraints, and effective communication. You can find additional information about the division of labor for this project in table 7.

References

[1] About spotify, Oct 2022. 1

[2] Papers with code: Machine learning datasets, December 2022. 1

[3] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. pages 591–596, 2011. 4

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 2

[5] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: An Audio Captioning Dataset, Oct. 2019. arXiv:1910.09387 [cs, eess]. 1, 4

[6] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. Esresnet: Environmental sound classification based on visual domain models. In *International Conference on Pattern Recognition*, 2020. 6

[7] Christine Hosey, Lara Vujović, Brian St. Thomas, Jean Garcia-Gathright, and Jennifer Thom. Just give me what i want: How people use and evaluate music search. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery. 1

[8] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *NAACL-HLT*, 2019. 4

[9] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D. Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020. 2

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019. 2

[11] Ilaria Manco, Emmanouil Benetos, Elio Quinton, and György Fazekas. Contrastive Audio-Language Learning for Music, Aug. 2022. arXiv:2208.12208 [cs, eess]. 2

[12] Meinard Müller, Frank Kurth, David Damm, Christian Fremerey, and Michael Clausen. Lyrics-based audio retrieval and multimodal navigation in music collections. In László Kovács, Norbert Fuhr, and Carlo Meghini, editors, *Research and Advanced Technology for Digital Libraries*, pages 112–123, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. 1

[13] Kurt Nassau. Color: The visible spectrum, 2022. 6

[14] Rod Nave. Sensitivity of the human ear, 2005. 6

[15] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning Representations*. Association for Computing Machinery, 2014. 2

[16] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1717–1724, 2014. 2

[17] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 2

[18] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech 2019*, pages 2613–2617, 2019. 3

[19] Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(2):2071–2081, Jun. 2022. 2

[20] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022. 2

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, Feb. 2021. arXiv:2103.00020 [cs]. 2

[22] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jun 2022. 2

[23] Google Research. Audioset, 2022. 1, 4

[24] Baohua Sun, Lin Yang, Michael Lin, Charles Young, Jason Dong, Wenhan Zhang, and Patrick Dong. Superchat: Dialogue generation by transfer learning from vision to language using two-dimensional word embedding. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*, DLP-KDD '19, New York, NY, USA, 2019. Association for Computing Machinery. 2

[25] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748v2, 2019. 2

[26] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2251–2265, 2019. 2

[27] Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via semantic similarity embedding. In *International Conference on Computer Vision*. Institute for Electrical and Electronics Engineers, 2015. 2