Localizing Auditory Concepts in CNNs

Pratyaksh Gautam [1] **Makarand Tapaswi** [*] [1] **Vinoo Alluri** [*] [1]

Abstract

Deep learning models are capable of complex auditory processing tasks such as keyword spotting, genre classification, and audio captioning, yet remain opaque. While several works have explored interpretability of neural networks for computer vision and natural language processing, the audio modality has been largely ignored. In this paper, we study the behavior of the *audio CNN encoder* used in the contrastively trained language-audio model, CLAP. In the domain of music and human speech sounds, we localize and identify the layers of the network that perform well on tasks of varying complexity, sometimes even outperforming the model's final outputs. Digging deeper, we also localize specific dataset classes to neuron clusters within a layer and analyze a cluster's contribution to the model's discriminability for that class. To perform these analyses, we propose an automated framework that can leverage a small dataset of a few thousand samples to evaluate and score neuron clusters for their role in classification. Our findings provide insights into the hierarchical nature of representations in audio CNNs, paving the way for improved interpretability of audio models.

1. Introduction

Interpretability in machine learning models is crucial for improving their trustworthiness (Adamson, 2023; Ribeiro et al., 2016) and enabling researchers to refine and enhance model architectures (Hanif et al., 2021; Molnar, 2020). This is particularly important as models are increasingly deployed in critical applications where their decisions have significant real-world implications (Haleem et al., 2019; Kaur et al., 2020; Thadeshwar et al., 2020; Nowotko, 2021). In natural language processing (NLP) and computer vision (CV), substantial progress has been made in interpreting how models process and understand data (Clark et al., 2019; Ribeiro et al., 2016; Zhou et al., 2015; Bau et al., 2017). However, interpretability in the audio modality lags behind from NLP and CV models, despite some recent efforts towards creating listenable interpretations (Parekh et al., 2022; Paissan et al., 2024; Choi et al., 2016).

Hierarchical encoding in deep networks refers to how representations at varying stages of the network capture different levels of abstraction of the input. Particularly, in vision models, early layers have been found to capture basic edges and textures, while deeper layers recognize more complex patterns like shapes and objects (Zhou et al., 2015). Similarly, NLP models process text in layers, with initial layers focusing on syntactic features and later layers understanding semantic content and contextual relationships (Tenney et al., 2019). While there have been notable advancements, such as listenable model-wise interpretations in the audio domain (Parekh et al., 2022; Paissan et al., 2024), which outline different approaches to identifying and presenting salient parts of the audio, these efforts are still in their early stages compared to the robust methodologies developed for NLP and CV models. Moreover, unlike our work, these studies do not identify *where* in the network are auditory concepts at varying levels of abstraction are encoded.

The expectation of hierarchical encoding of sound in neural networks, particularly in models designed for audio processing, can be thought of as being analogous to auditory processing in the brain. Starting with spectral decomposition in the ear, auditory information is then passed through successive stages of the auditory pathway, where progressively more complex analysis and integration over time results in the perception and interpretation of sounds (Leaver & Rauschecker, 2010; Giordano et al., 2013; Santoro et al., 2014; Alluri & Kadiri, 2019; Kell et al., 2018; Caucheteux et al., 2022; O'Sullivan et al., 2019). One might surmise that neural networks designed for audio processing may mimic this structure to effectively capture the various levels of acoustic information.

Localizing within a network: From Layers to Neurons Localizing specific parts of the network responsible for its performance on different tasks is crucial for improving transfer learning, debugging and interpretability, optimization,

[*]Equal contribution [1]IIIT Hyderabad, India. Correspondence to: Pratyaksh Gautam <pratyaksh.g@research.iiit.ac.in>.

Accepted at the Mechanistic Interpretability Workshop, 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

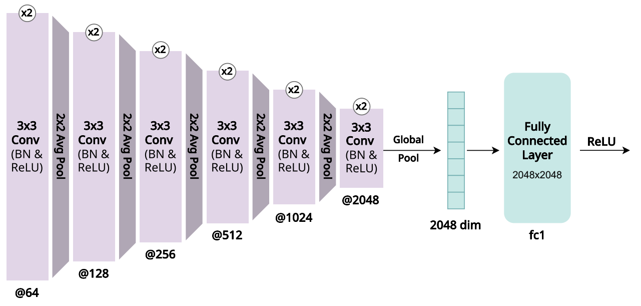

Figure 1: Brief overview of CLAP's audio encoder architecture. Our work analyzes the activations/outputs of every convolutional block (`conv1-5`), the last convolutional layer before global pool (`conv6`) and after the fully-connected layer (`fc1`). See Figure 6 in the appendix for a more detailed diagram of the audio encoder.

and even regulatory compliance. "Localizing within a network" can be performed at varying levels of granularity, *e.g.* layers, neuron clusters, channels, and to single neurons. Recent advancements in deep learning have demonstrated that it is indeed possible to pinpoint certain layers and neurons within an NLP model that are crucial for specific tasks (Tenney et al., 2019). Identifying neurons important for performing particular functions within a model, that is, neuron interpretability, has been successfully applied to NLP models to discover neurons that encode specific linguistic properties or concepts, such as syntax, semantics, or part-of-speech tags (Sajjad et al., 2022). Similar techniques have been employed in CV to identify neurons that detect particular objects or features within an image (Bau et al., 2017; Zhou et al., 2015). However, this concept has not been applied to audio models. In fact, to our best knowledge, only one study has attempted to examine neuron-level interpretability for OpenAI's Whisper (automatic speech recognition) model (Radford et al., 2022; Reid, 2023). While this is performed at the level of individual neurons, we posit that examining clusters of neurons can reveal how groups of neurons collectively contribute to certain tasks or functionalities.

In our study, we first illustrate how different layers of an audio encoder are responsible for classification tasks that rely on extracting information at various levels of abstraction. Subsequently, within layers, we isolate the behavior of neuron clusters that exhibit class-wise specialization. We show how clusters specialize towards the semantic classes across the layers and note their contribution towards classification accuracy through an experiment that selectively blocks out their activations. For our analysis, we adopt the audio encoder of CLAP (Elizalde et al., 2023) that is pretrained on Audio Set (Gemmeke et al., 2017), and then further trained with a contrastive-learning objective on audio captioning datasets. CLAP is among the state-of-the-art audio CNN models, and also enables zero-shot classification on unseen datasets through the text encoder, without the need to update any parameters.

In summary, our key contributions are: (i) we localize the information needed for different tasks in two audio domains (speech and music) to specific layers in the CNN audio encoder. (ii) we demonstrate how to further localize the information associated with specific classes to neuron clusters within a layer. (iii) finally, we propose an automated framework to examine neuronal clusters and measure their involvement in the discrimination of a specific class and apply it to the ESC-50 dataset.

2. Localizing *to* a Layer: Do Audio CNNs Encode Hierarchy?

Similar to hierarchies exhibited by visual models (edges, object parts, objects) (Zhou et al., 2015; Bau et al., 2017) or language models (syntax to semantics) (Tenney et al., 2019), we explore if such properties can be observed in audio CNN models. Specifically, are lower layers in the network better at distinguishing simpler acoustic (low-level) properties, while the higher layers encode semantic (high-level) attributes? We answer this by localizing (identifying the layer) the representation in the network that results in best performance.

2.1. Tasks and Datasets

We consider two domains: music (♬) and human speech (☺) to evaluate if the audio CNN encodes a hierarchy. Note, the CLAP's audio encoder is pretrained on Audio Set and then fine-tuned on several audio-language paired datasets: FSD50k (Fonseca et al., 2022) a sound event classification dataset, and ClothoV2 (Drossos et al., 2020), AudioCaps (Kim et al., 2019), and MACS (Martin-Morato & Mesaros, 2021), audio captioning datasets. While we expect the higher layers to encode semantic audio classes, it is interesting to see if the model learns to encode low-level properties of sounds.

Within speech and music both, we consider three tasks based on the complexity of the task, defined as how much information needs to be aggregated (over time and frequency) to identify the classes accurately. Table 1 summarizes the tasks and datasets, number of classes, instances, and average audio duration.

Tasks in music. (i) We consider *note name classification*[1] as the low-level task that requires identifying one of the 12 notes in music, while being agnostic to the octave. Specifically, we adopt a part of the NSynth dataset (Engel et al.,

[1]Also referred to as "chroma classification" by HEAR, A Holistic Evaluation of Audio Representations (Turian et al., 2022).

2017) for this work. (ii) Single *instrument classification* on the Medley-solos-DB dataset (Lostanlen et al., 2019) is considered as a mid-to-high level task that requires understanding *timbre*. Understanding timbre requires analyzing more than one note and is often characterized by harmonic content or attack and decay patterns. (iii) Finally, we use *genre classification* on the GTZAN dataset (Turian et al., 2022) as the high-level task that requires understanding how various aspects of the music interact to form a recognizable genre.

Note, the CNN model pretrained on Audio Set is aware of musical instruments and genre classes as they are part of the ontology; however, the model is never trained explicitly for note name classification. Based on studies that support hierarchical encoding of music (Alluri & Kadiri, 2019; Giordano et al., 2013; Leaver & Rauschecker, 2010), we expect the early layers of the CNN to better encode *note name*, middle to higher layers to better encode *instrument* while the last layers best encode the higher order percept of *genre*.

Tasks in speech. (i) We start with the simple task of *consonant classification* in speech. Specifically, we adopt the Persian Consonant Vowel Combination (PCVC) (Malekzadeh et al., 2020) dataset which includes phonemic combinations of consonants and vowels. (ii) We move on to *keyword recognition* as the mid-level task that requires models to string together multiple syllables. Here, we use the Speech Commands (Warden, 2018) dataset that consists of simple single words. (iii) Finally, similar to genre classification that requires parsing a mixture of instruments, we consider *speaker count estimation* that requires isolating and counting different voices in an audio from the LibriCount (Stöter et al., 2018) dataset as our high-level task.

Note, the CNN model is trained for distinguishing human sounds such as speech, shout, scream, *etc.*[2], but not explicitly for speech recognition. Based on studies that support hierarchical encoding of speech (Caucheteux et al., 2022; O'Sullivan et al., 2019) we expect early layers to better represent consonants, while middle layers would better encode keywords and higher layers would be responsible for integrating linguistic and paralinguistic information resulting in encoding of speaker count.

2.2. Methodology

Preprocessing. We standardize audio samples across datasets to a duration of 5 s and sampling rate of $44.1\,\text{kHz}$. Short audio files are padded with silence (on the right). From long audio files, we select the middle 5 s. Similar to CLAP (Elizalde et al., 2023), we compute the log-mel spectrogram as the model input.

[2]Audio Set human sounds ontology `https://research.google.com/audioset/ontology/human_sounds_1.html`

Layerwise representations. We extract and store intermediate representations from all samples across all datasets through CLAP's audio encoder. We obtain outputs of six convolutional layers (`conv1-5` after average-pooling, `conv6` before global pooling) and the first fully-connected layer (`fc1`, after ReLU). See Figure 1 for a brief overview of CLAP's audio encoder.

Given an audio $\mathbf{x} \in \mathbb{R}^{sr}$ with duration s and sampling rate r, we first compute the time-frequency log-mel spectrogram, $\mathbf{s} \in \mathbb{R}^{F \times T}$, where F=64 Mel-scale bands and T=690 time steps corresponding to 5 s. We denote the convolutional output at layer l from a partial encoder Φ_l as $\tilde{\mathbf{h}}_l \in \mathbb{R}^{c_l \times w_l \times h_l}$, a feature map of c_l channels and $w_l \times h_l$ width (time) and height (frequency) activations. We flatten $\tilde{\mathbf{h}}_l$ to obtain $\mathbf{h}_l \in \mathbb{R}^{d_l}, d_l = c_l w_l h_l$. Note, the output of `fc1` is already a single dimension vector. In summary, $\mathbf{h}_l = \Phi_l(\mathbf{x})$ is the feature extraction process for each layer.

k-Nearest Neighbor evaluation. Two common ways to probe the quality of any representation are nearest neighbor or linear probing (training a linear classifier) (Chen et al., 2020). We adopt the former approach as it allows us to work with relatively small datasets (around 1000-2000 instances).

We perform k-Nearest Neighbor classification using a five-fold stratified cross-validation setup, with Euclidean distance as the distance metric. Note that the model parameters are not updated in any way in this procedure. We repeat this for a range of values of k and observe that the trends are typically unchanging (see Appendix C for the complete result). Thus, as we mainly interested in the trends of how performance changes across the layers, we select the best value of k for each task. Since k-Nearest Neighbor is susceptible to class imbalance, we consider class-balanced subsets for most of the datasets (Table 1).

Zero-shot classification. Following CLAP (Elizalde et al., 2023), we also conduct zero-shot classification by creating language prompts and using the aligned representations obtained after the projection head. Prompts used for each task are given in Appendix A.

2.3. Results

Figure 2 shows how classification accuracy varies across layers. We observe strong performance variation across the layers.

Low-level tasks. Tasks hypothesized to be low-level peak earlier in the model: note name classification at `conv1`, consonant classification at `conv3`, and in fact are close to random chance in later layers. As language-audio pretraining often focuses on semantic properties, we observe that zero-shot classification performs on par with random chance

Table 1: Datasets used in our work from music (♬) and human speech (☺) domains. Note, the datasets are class-balanced when possible (* represents balancing was performed). Details in Appendix A.

Dataset	Task	Type	#Instances	#Classes	Duration (s)
NSynth (Engel et al., 2017)	Note Name Classification	♬	1800*	12	4
Medley-solos-DB (Lostanlen et al., 2019)	Instrument Classification	♬	965*	7*	3
GTZAN (Turian et al., 2022)	Genre Classification	♬	1000	10	30
PCVC (Malekzadeh et al., 2020)	Consonant Classification	☺	1794	23	2
Speech Commands (Warden, 2018)	Keyword Recognition	☺	1750*	35	1
LibriCount (Stöter et al., 2018)	Speaker Count Estimation	☺	1100*	11	5

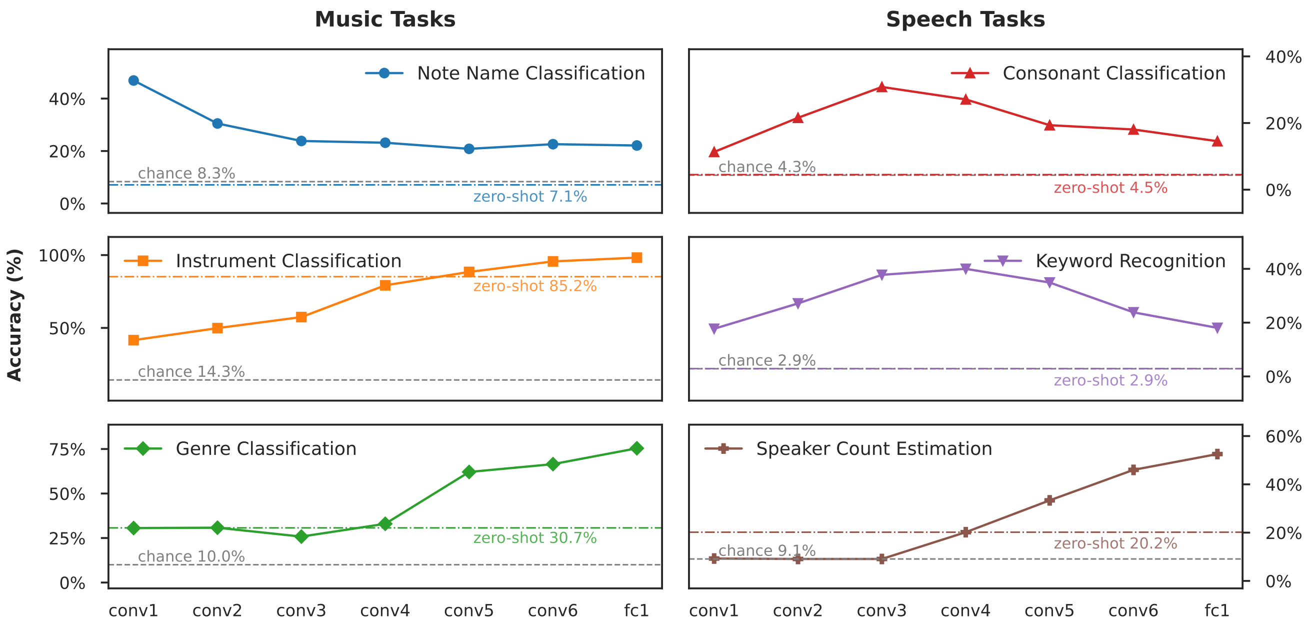

Figure 2: Classification performance using k-Nearest Neighbor with representations extracted at different layers. Results are averaged over a five-fold cross-validation. We observe that low-level tasks (note name and consonant classification) perform better in early layers while high-level tasks (genre classification, speaker count estimation) perform best in later layers.

for these low-level tasks[3].

Intermediate tasks. As per expected behavior, keyword recognition peaks at conv4 and performs poorly in early and later layers. On the other hand, instrument classification peaks strongly at `fc1`, achieving close to perfect results. We suspect this is primarily due to the Audio Set pretraining which includes instrument classes. This is further evidenced by a high zero-shot accuracy of 85.2%. Nevertheless, given that it is a mid- to high-level task, we do see that it performs poorly in the early layers; mid-layers, starting from conv4, show performance significantly above random chance.

High-level tasks. Finally, we see that speaker count estimation and genre classification both peak at the last layer: `fc1`. Furthermore, in the early layers `conv1-3`, performance is close to random chance or the zero-shot accuracy. While CLAP's audio encoder is pretrained for genre classification (Audio Set), it is not familiar with complex tasks such as speaker count estimation.

We see that intermediate representations at different layers perform well above chance at corresponding auditory tasks – early layers at low-level tasks, and later layers at high-level tasks – as compared to other layers in the model. Notably, this happens even in the case of tasks that the model was *not* explicitly trained for, such as keyword recognition. Unless the model was incentivized to maintain this information through explicit training on the task (as in the case of genre classification), this discriminative ability degrades in later layers.

[3]Note, the zero-shot classification performance is also dependent on the text encoder's ability to discriminate well between *e.g.* the notes 'C' and 'D' in note name classification, which we do not expect it to do.

We also see that k-Nearest Neighbor classification on the later layers of the audio encoder often outperforms the zero-shot classification accuracy by a great margin. This suggests that either the text encoder or the projection matrices bottleneck CLAP's ability to perform these tasks in the zero-shot setting.

3. Localizing *Within* a Layer: Spotting Neuron Clusters

In the previous section, we attribute task accuracy to representations of specific layers in the audio encoder. We now explore whether we can further localize model performance by identifying groups of neurons that encode information relevant to specific audio classes.

Dataset. In this section, we focus our analysis to the Environmental Sounds Classification (ESC-50) dataset (Piczak, 2015). ESC-50 is chosen as it is a single dataset that consists of 50 classes that can be broadly grouped into 5: animals, natural soundscapes, human non-speech sounds, indoor domestic sounds, and outdoor urban sounds. Apart from Audio Set (which we avoid due to CLAP's pre-training), this is also a popular classification benchmark for evaluating audio representations[4]. Additionally, ESC-50 is a high-level task and accuracy peaks in the later layers as can be observed from the k-Nearest Neighbor plots (Appendix D).

3.1. Creating and Representing Neuron Clusters

From the previous section, we see that early layers tend to encode basic acoustic features, higher-levels are able to segregate them into distinct semantic categories. For instance, the sounds of a clock alarm may share similar spectrotemporal modulations of a siren, however, is clearly distinct semantically (all classes in ESC-50).

Hence, we can expect clusters of neurons from lower layers to activate in a similar fashion to these sounds, while higher layers might have separate clusters for these such categories.

Clustering neurons. Recall, $\mathbf{h}_l \in \mathbb{R}^{d_l}$ represents the feature map at layer l for a single audio $\mathbf{x}$, where $d_l = c_l w_l h_l$. For a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of N samples, we compute and stack all features:

$$\mathbf{H}_l = [\Phi_l(\mathbf{x}_1), \ldots, \Phi_l(\mathbf{x}_N)], \text{ and } \mathbf{H}_l \in \mathbb{R}^{N \times d_l}. \quad (1)$$

Each row of this matrix is a representation of a sample $\mathbf{x}_i$. We now refer to the activations of instances in the dataset as a *representation* of a neuron. Instead of rows, by indexing the columns of $\mathbf{H}_l$, we can refer to neuron j's representation: $\mathbf{h}_l^j \in \mathbb{R}^N, \forall j \in [1, \ldots, d_l]$.

Using these representations, we cluster all neurons within a layer with FINCH (Sarfraz et al., 2019), an algorithm that uses the first nearest neighbors in terms of Euclidean distance. FINCH is a hierarchical method and produces a handful of clustering partitions. We select partitions at each layer such that the number of clusters across layers is similar. [5]

Cumulative Class-wise Activation Vectors (CCAV). Let us denote the partition with P clusters as: $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_P\}$. Then, the average neuronal representation of the cluster p is

$$\mathbf{c}_{p,l} = \frac{1}{|\mathcal{C}_p|} \sum_{j \in \mathcal{C}_p} \mathbf{h}_l^j$$

. For brevity, we drop the index l and use $\mathbf{c}_p \in \mathbb{R}^N$. However, clustering is only performed within a layer.

Next, we compute the cumulative class-wise activation vector for each cluster. This is a K dimensional vector corresponding to the number of classes in the dataset (K=50 for ESC-50) that accumulates the activations across samples of the same category. More formally,

$$\mathbf{a}_p[k] = \sum_{i=1}^{N} \mathbb{1}[y_i = k]\, \mathbf{c}_p[i]\,, \quad (2)$$

where $y_i \in \{1, \ldots, K\}$ is the class label for audio sample $\mathbf{x}_i$. The CCAV, or $\mathbf{a}_p \in \mathbb{R}^K$, captures the affinity of neurons in a cluster to each class in the dataset. Note, all the activations in our work are analyzed post the ReLU layer. Thus, we do not need to worry about negative activations from different instances eroding the scores, both in $\mathbf{c}_p$ or $\mathbf{a}_p$.

3.2. Cluster Entropy

We consider three entropy measures to characterize a cluster $\mathcal{C}_p$: *semantic*, *positional*, and *channel*.

Semantic entropy (SEnt). To compute this score (per cluster), we first normalize the CCAV to sum to one: $\hat{\mathbf{a}}_p = \mathbf{a}_p / \sum_k \mathbf{a}_p$. We treat the resulting vector as a pseudo-probability (how likely are neurons of this cluster to activate for a class) and compute entropy as $e_p^s = -\hat{\mathbf{a}}_p \cdot \log(\hat{\mathbf{a}}_p)$.

A low SEnt implies that the neuron cluster activates primarily for instances from one (or few) class(es), and is a likely candidate for carrying salient information to identify those classes. A high SEnt implies that the neurons in the cluster activate for instances more uniformly across classes. Note, a higher SEnt does not necessarily mean that the cluster does not have a well-defined and interpretable behavior, it only

[4]See https://paperswithcode.com/sota/audio-classification-on-esc-50

[5]We define the count distance between two partitions as $\frac{|n_a - n_b|}{\max(n_a, n_b)}$, where n_a and n_b are cluster counts. We select one partition from each layer, such that the count distance is pairwise minimized for the selected partitions.

highlights that the behavior does not align well to one/few dataset classes.

Positional entropy (PEnt) is related to the *time-frequency* (spatial) occurrence map of a cluster. For each neuron in the cluster $\mathcal{C}_p$, we identify its time-frequency location in the original non-flattened activation of size $c_l \times w_l \times h_l$. Next, we construct a *position map*, $\mathbf{M}_p \in \mathbb{N}^{w_l \times h_l}$ that simply counts the distribution of neurons in the cluster across $w_l \times h_l$.

We normalize the position map to obtain $\hat{\mathbf{M}}_p = \mathbf{M}_p / |\mathcal{C}_p|$ and refer to it as a pseudo-probability, the likelihood of neurons of a cluster occupying a certain time and frequency region. We compute entropy as $e_p^{\text{pos}} = -\hat{\mathbf{M}}_p \cdot \log(\hat{\mathbf{M}}_p)$. Clusters with low PEnt occupy specific regions in the activation map (*e.g.* higher frequencies) and are indicative of the model dedicating neurons to selective regions of the spectrogram. High PEnt clusters typically span the entire spectrogram.

Channel entropy (CEnt) is similar to PEnt and measures the flatness of neuron distribution across the channel dimension c_l. A low CEnt means the cluster has neurons activated by the same filter.

3.3. Cluster Interventions

We now study the impact that a cluster, especially one with a high affinity to a specific class (or a few set of classes) has on the model's performance. We *corrupt* the information passed up by the neurons in a chosen cluster $\mathcal{C}_p$, and observe the effect on the model's zero-shot classification performance.

We intervene using *resampling ablations* (Chan et al., 2022), *i.e.* by replacing the activations $\mathbf{h}_l^j$, $\forall j \in \mathcal{C}_p$ for input $\mathbf{x}$ with activations picked from some other random input $\mathbf{x}'$. As a control experiment, we also intervene on a random set of neurons $\mathcal{C}_p'$ of equal count as $\mathcal{C}_p$. To mitigate the effect of randomness, we repeat and present results averaged across 10 runs of the experiment.

3.4. Results

Entropy analysis. Figure 3 presents the distributions (boxplots) for how entropy ratios of neuron clusters change across the layers of a network. Entropy ratio is defined as the actual entropy of a cluster divided by the maximum entropy based on a uniform distribution across K classes for SEnt, $w_l \times h_l$ for PEnt, and c_l channels for CEnt.

Clusters in early layers have a high SEnt ratio indicating that they do not specialize to classes in the ESC-50 dataset (which may be considered as a high-level task). However, later layers, especially `conv6`, show a notable drop in SEnt, *e.g.* indicative of their effictiveness for k-NN classification.

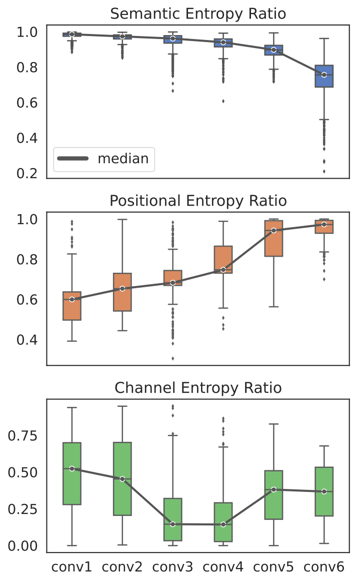

Figure 3: Trends in cluster entropy ratio across the convolutional block layers. **Top**: Semantic entropy (SEnt), **Middle**: Positional entropy (PEnt), **Bottom**: Channel entropy (CEnt).

Position entropy trends are opposite. Early layers have a low PEnt ratio, indicative of them being localized to certain time-frequency regions while later layers show high PEnt suggesting that they encode and aggregate information across larger portions of the spectrogram.

Channel entropy trends are interesting. First, the distribution often spans the entire range (0-1). Next, the middle layers show low CEnt indicative of them learning filters with mid-sized receptive fields that resulting in tight clusters. We leave an in-depth exploration for future work.

CCAV distributions can also be analyzed across the layers by finding the cluster that peaks in one of the classes. An example for the class "clock alarm" shown in Figure 4 highlights how $\mathbf{a}_p$ reduces in semantic entropy, *i.e.* becomes peakier across the layers. As the early layers (`conv2`, top) have a small receptive field, the clock alarm sound may be confused with a bunch of similar sounds such as siren, church bells, or a vacuum cleaner, rough timbral textures[6]. that are still harmonic sounds with clear fundamental frequencies (albeit with minor fluctuations).

As we go to higher layers, *e.g.* at `conv4`, the model gains higher discriminative power. Interestingly, for a similar cluster here, *crickets* are now the second most activating class in the cluster. Since the higher layers integrate information

[6]Timbral texture (e.g. rough/harsh) has been considered a mid-level sound attribute (Alluri & Kadiri, 2019; Saitis & Weinzierl, 2019)

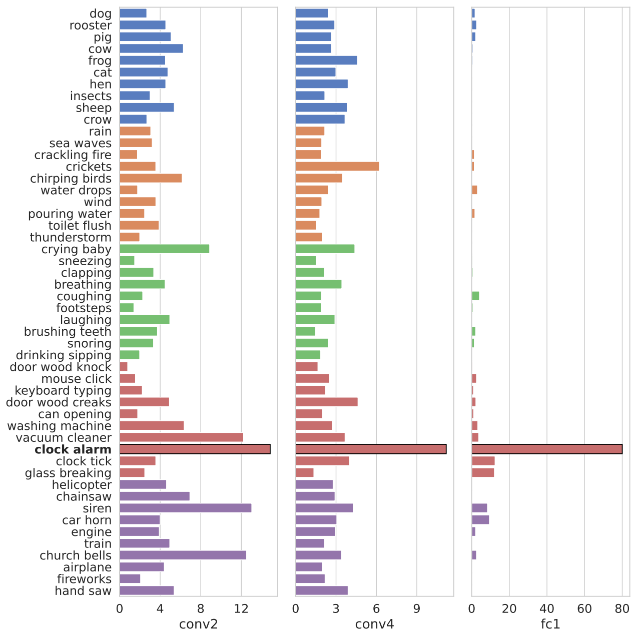

Figure 4: CCAVs for neuron clusters corresponding to "clock alarm" at different layers. Color groups are the 5 super-categories mentioned earlier in ESC-50. **Left:** `conv2`, partition 5, cluster 13. **Middle:** `conv4`, partition 4, cluster 6. **Right:** `fc1`, partition 1, cluster 19. Clusters in later layers specialize for just one or two classes. The scale of the plot also indicates the strength of the "clock alarm" activations in `fc1`. Note, we find such trends even when clustering is performed independently across layers.

over larger spectrotemporal fields, this finding can be attributed to the periodicity of the aforementioned harmonic yet rough timbral textures. Finally, at the last layer (`fc1`) of the encoder, we see that the neuron cluster specializes completely to the "clock alarm" class. A part of this may also be attributed to the CNN being pretrained on Audio Set which contains the class "alarm clock".

Cluster interventions are performed at scale on a subset of all clusters from across all layers. We compute the mean neuronal SEnt across the model, and then pick the largest 100 clusters that have SEnt below this threshold. Both a smaller SEnt (the cluster carries a clear concept with respect to the datset), and a larger neuron count (the cluster can have substantial effect on the model's performance when intervened with) are preferred.

Fig. 5 shows three examples of intervening on a cluster. Intervention on the cluster's activations often results in a performance drop for the most activating class. On Fig. 5 (top subfigure), we see that the accuracy for the *cat* drops by 44.3% while other classes are basically unaffected. This number is also reflected in Table 2 row 1. We observe that while the secondary (2.3%) and other classes (0.4%) are

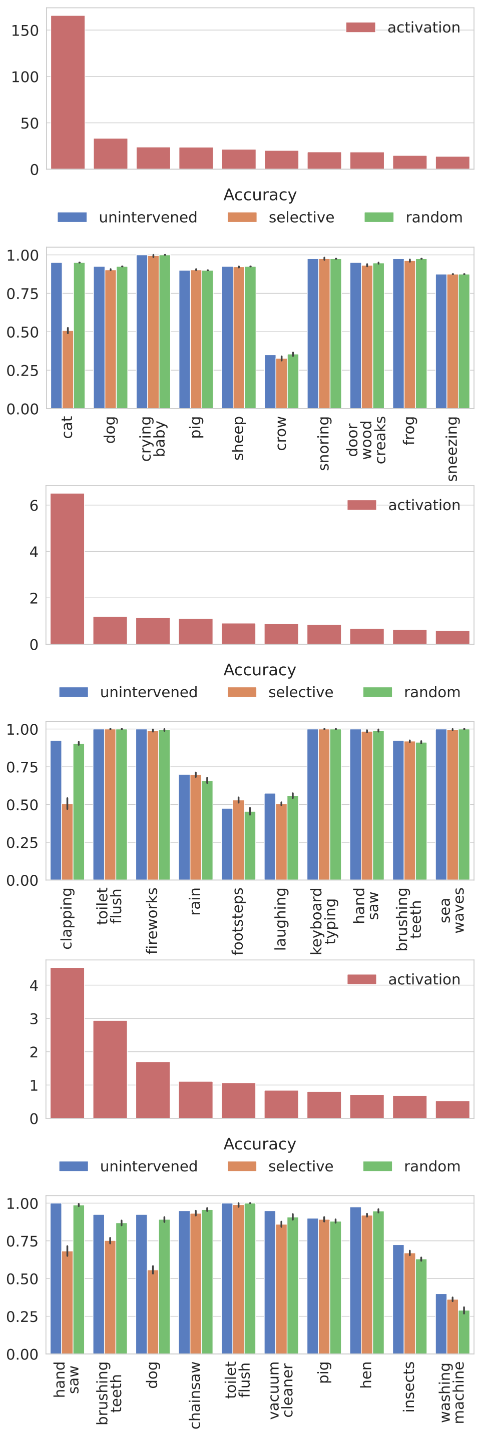

Figure 5: Each subfigure shows the first 10 maximally activating classes in the CCAV (**above**) for the labeled cluster, and how the class-wise accuracy is affected (**below**) when we intervene on (i) none of the clusters, (ii) a specifically selected cluster, and (iii) a random set of neurons of equal count as the cluster. Error bars represent 95% confidence intervals. Top: `fc1`, partition 1, cluster 7. Middle: `conv6`, partition 4, cluster 0. Bottom: `conv6`, partition 4, cluster 9.

Table 2: Impact of interventions on neuron clusters. The table shows neuron cluster metadata: layer, partition (P), cluster index (C), number of neurons (#N), and semantic entropy ratio (SEntR). Next, we present the first (①) and second (②) most activating class of the cluster based on the CCAV. Finally, we present the accuracy drop (in %) for the ①, ②, and other classes (Oth) when selectively replacing neuron activations of the cluster (selective: Sel) or for random neurons, control experiment (Ctrl). We represent the mean scores separately for early layers: `conv1-3`, later layers: `conv4-6` and `fc1`, and all layers. We observe significant drops in performance for the primary class, while the secondary is typically unchanged. When a large cluster is intervened, it may affect other classes.

Neuron Cluster Metadata					Most Activating Class		Acc Drop: Sel			Acc Drop: Ctrl		
Layer	P	C	#N	SEntR	①	②	①	②	Oth	①	②	Oth
Clusters in Figure 5												
`fc1`	1	7	20	0.81	cat	dog	44.3	2.3	0.4	0.0	0.0	-0.1
`conv6`	4	0	2943	0.87	clapping	toilet flush	42.0	0.0	0.6	2.0	0.0	1.4
`conv6`	4	9	9166	0.90	hand saw	brushing teeth	31.8	17.3	4.1	1.3	5.5	6.1
Layerwise mean (later layers)												
`fc1`	100 clusters		25.2	0.77	-	-	3.0	0.6	0.1	0.2	0.1	0.2
`conv6`	100 clusters		1824.9	0.85	-	-	7.3	2.6	0.3	0.5	1.1	0.9
`conv5`	100 clusters		379.8	0.89	-	-	0.9	0.6	0.1	0.0	0.1	0.1
`conv4`	75 clusters		235.1	0.90	-	-	0.5	0.7	0.0	0.0	0.1	0.0
Mean across multiple layers												
Early	128 clusters		1033.6	0.89	-	-	0.2	0.2	0.1	0.0	0.0	0.0
Later	375 clusters		641.7	0.85	-	-	3.1	1.1	0.1	0.2	0.4	0.3
All	503 clusters		741.4	0.86	-	-	2.3	0.9	0.1	0.1	0.3	0.2

mostly unchanged, the intervening on the control group of neurons leaves model performance unaffected.

However, the effects are not always disentangled. A large cluster of over 9000 neurons (about 10.7% of `conv6`) is shown on Fig. 5 (bottom subfigure). First, we note that it activates for multiple classes. Next, intervening on the scores of this neuron affects performance across multiple classes: *hand saw* and *brushing teeth*, roughly proportional to how much they activate; but also *dog*, despite the milder activations. Table 2 row 3 shows that it also results in a mean accuracy drop of 4.1% on other labels, perhaps due to its large size.

There are also several clusters with relatively milder activations where interventions result in no change. Thus, beyond the individual cluster examples, Table 2 also reports the mean accuracy drop due to interventions on many clusters of each layer. As expected, selective intervention on semantic clusters of layers `conv6` and `fc1` affects performance the most (3.0% and 7.3% respectively). Layers `conv5` and below show lesser influence, perhaps owing to other neuron clusters that step in and restore model performance. This can also be observed in the difference between the means of the early (0.2%) and later (3.1%) layers.

4. Conclusion

While CNNs mimic vision processing in the brain by integrating information based on spatial proximity, this is not the case for auditory processing. Auditory analysis relies on a distributed spectro-temporal integrative process wherein integration occurs at multiple timescales and across the frequency spectrum that aids in giving rise to auditory concepts at varying levels of abstraction.

Towards better interpretability of audio models, we demonstrated layer-wise trends indicative of hierarchical encoding in audio CNNs, specifically for music and speech. We showed, for the first time, that clusters of neurons within layers encode auditory concepts. Especially in the higher layers, we observed clusters adhering to specific dataset classes and observed that corrupting their activations can affect the model's performance on those classes. Finally, we introduced an automated entropy-based framework to examine the degree of involvement of neuron clusters in specific classes of sounds.

5. Limitations

We see two limitations of the current work: (i) Our current analysis is limited to the behavior of the CNN audio encoders, specifically the one used in CLAP. Transformer encoders such as AST (Gong et al., 2021) or BEATs (Chen et al., 2022) may require different interpretability studies. However, our methodology of neuron representations, entropy calculations, CCAV, and interventions are generic and may be applied to any neural network. (ii) Pretraining strategies with different datasets may modify the behavior of the model resulting in expected mid-level tasks also performing well in later layers (*e.g.* as is the case with instrument classification).

6. Acknowledgements

The authors thank Abhinav Menon, Shashwat Singh, Jatin Agarwala, Vedansh Agrawal, and other colleagues for their insightful comments and support.

References

Adamson, G. Explaining Technology We do not Understand. *IEEE Transactions on Technology and Society*, 4(1):34–45, 2023.

Alluri, V. and Kadiri, S. R. Neural Correlates of Timbre Processing. *Timbre: Acoustics, Perception, and Cognition*, pp. 151–172, 2019.

Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. Network Dissection: Quantifying Interpretability of Deep Visual Representations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Caucheteux, C., Gramfort, A., and King, J.-R. Deep language algorithms predict semantic comprehension from brain activity. *Scientific Reports*, 12(1):16327, 2022.

Chan, L., Garriga-Alonso, A., Goldwosky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris, B., and Thomas, N. Causal scrubbing, a method for rigorously testing interpretability hypotheses. *AI Alignment Forum*, 2022. URL https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing

Chen, S., Wu, Y., Wang, C., Liu, S., Tompkins, D., Chen, Z., and Wei, F. BEATs: Audio Pre-Training with Acoustic Tokenizers. *arXiv preprint arXiv:2212.09058*, 2022.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In *International Conference on Machine Learning (ICML)*, 2020.

Choi, K., Fazekas, G., and Sandler, M. Explaining deep convolutional neural networks on music classification. *arXiv preprint arXiv:1607.02444*, 2016.

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. What does BERT Look At? An Analysis of BERT's Attention. In *ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Drossos, K., Lipping, S., and Virtanen, T. Clotho: An Audio Captioning Dataset. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

Elizalde, B., Deshmukh, S., Al Ismail, M., and Wang, H. CLAP: Learning Audio Concepts From Natural Language Supervision. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Eck, D., Simonyan, K., and Norouzi, M. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. *arXiv preprint arXiv:1704.01279*, 2017.

Fonseca, E., Favory, X., Pons, J., Font, F., and Serra, X. FSD50k: An Open Dataset of Human-labeled Sound Events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852, 2022.

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. Audio Set: An Ontology and Human-labeled Dataset for Audio Events. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

Giordano, B. L., McAdams, S., Zatorre, R. J., Kriegeskorte, N., and Belin, P. Abstract encoding of auditory objects in cortical activity patterns. *Cerebral cortex*, 23(9):2025–2037, 2013.

Gong, Y., Chung, Y.-A., and Glass, J. AST: Audio Spectrogram Transformer. In *Interspeech*, 2021.

Haleem, A., Javaid, M., and Khan, I. H. Current Status and Applications of Artificial Intelligence (AI) in Medical Field: An Overview. *Current Medicine Research and Practice*, 9(6):231–237, 2019.

Hanif, A. M., Beqiri, S., Keane, P. A., and Campbell, J. P. Applications of Interpretability in Deep Learning Models for Ophthalmology. *Current Opinion in Ophthalmology*, 32(5):452–458, 2021.

Kaur, S., Singla, J., Nkenyereye, L., Jha, S., Prashar, D., Joshi, G. P., El-Sappagh, S., Islam, M. S., and Islam, S. R. Medical Diagnostic Systems using Artificial Intelligence (AI) Algorithms: Principles and Perspectives. *IEEE Access*, 8:228049–228069, 2020.

Kell, A. J., Yamins, D. L., Shook, E. N., Norman-Haignere, S. V., and McDermott, J. H. A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. *Neuron*, 98(3):630–644, 2018.

Kim, C. D., Kim, B., Lee, H., and Kim, G. AudioCaps: Generating Captions for Audios in The Wild. In *North*

American Chapter of Association of Computational Linguistics: Human Language Technologies (NAACL-HLT), 2019.

Leaver, A. M. and Rauschecker, J. P. Cortical representation of natural complex sounds: Effects of acoustic features and auditory object category. *Journal of Neuroscience*, 30(22):7604–7612, 2010.

Lostanlen, V., Cella, C.-E., Bittner, R., and Essid, S. Medley-solos-DB: a cross-collection dataset for musical instrument recognition, 2019. URL `https://doi.org/10.5281/zenodo.3464194`.

Malekzadeh, S., Gholizadeh, M. H., Razavi, S. N., and Zadeh, H. G. The Recognition of Persian Phonemes using PPNet. *J. Med. Signals Sens.*, 10(2):86–93, 2020.

Martin-Morato, I. and Mesaros, A. What is the ground truth? Reliability of multi-annotator data for audio tagging. In *European Signal Processing Conference (EUSIPCO)*, 2021.

Molnar, C. *Interpretable Machine Learning*. Lulu.com, 2020.

Nowotko, P. M. AI in Judicial Application of Law and the Right to a Court. *Procedia Computer Science*, 192: 2220–2228, 2021.

O'Sullivan, J., Herrero, J., Smith, E., Schevon, C., McKhann, G. M., Sheth, S. A., Mehta, A. D., and Mesgarani, N. Hierarchical encoding of attended auditory objects in multi-talker speech perception. *Neuron*, 104(6):1195–1209, 2019.

Paissan, F., Ravanelli, M., and Subakan, C. Listenable Maps for Audio Classifiers. *arXiv preprint arXiv:2403.13086*, 2024.

Parekh, J., Parekh, S., Mozharovskyi, P., d'Alché Buc, F., and Richard, G. Listen to Interpret: Post-hoc Interpretability for Audio Networks with NMF. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Piczak, K. J. ESC: Dataset for Environmental Sound Classification. In *ACM Multimedia (MM)*, 2015.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust Speech Recognition via Large-Scale Weak Supervision. *arXiv preprint arXiv:2212.04356*, 2022.

Reid, E. Interpreting OpenAI's Whisper. `https://er537.github.io/blog/2023/09/05/whisper_interpretability.html`, 2023.

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Saitis, C. and Weinzierl, S. The Semantics of Timbre. *Timbre: Acoustics, perception, and cognition*, pp. 119–149, 2019.

Sajjad, H., Durrani, N., and Dalvi, F. Neuron-level Interpretation of Deep NLP Models: A Survey. *Transactions of the Association for Computational Linguistics*, 10:1285–1303, 11 2022.

Santoro, R., Moerel, M., De Martino, F., Goebel, R., Ugurbil, K., Yacoub, E., and Formisano, E. Encoding of natural sounds at multiple spectral and temporal resolutions in the human auditory cortex. *PLoS Computational Biology*, 10(1):e1003412, 2014.

Sarfraz, M. S., Sharma, V., and Stiefelhagen, R. Efficient Parameter-free Clustering Using First Neighbor Relations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8934–8943, 2019.

Stöter, F.-R., Chakrabarty, S., Habets, E., and Edler, B. Libricount, A Dataset For Speaker Count Estimation, 2018.

Tenney, I., Das, D., and Pavlick, E. BERT Rediscovers the Classical NLP Pipeline. In *Association of Computational Linguistics (ACL)*, 2019.

Thadeshwar, H., Shah, V., Jain, M., Chaudhari, R., and Badgujar, V. Artificial Intelligence based Self-driving Car. In *International Conference on Computer, Communication and Signal Processing (ICCCSP)*, 2020.

Turian, J., Shier, J., Khan, H. R., Raj, B., Schuller, B. W., Steinmetz, C. J., Malloy, C., Tzanetakis, G., Velarde, G., McNally, K., Henry, M., Pinto, N., Noufi, C., Clough, C., Herremans, D., Fonseca, E., Engel, J., Salamon, J., Esling, P., Manocha, P., Watanabe, S., Jin, Z., and Bisk, Y. HEAR: Holistic Evaluation of Audio Representations. In *NeurIPS: Competitions and Demonstrations Track*, 2022.

Warden, P. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Object Detectors Emerge in Deep Scene CNNs. In *International Conference on Learning Representations (ICLR)*, 2015.

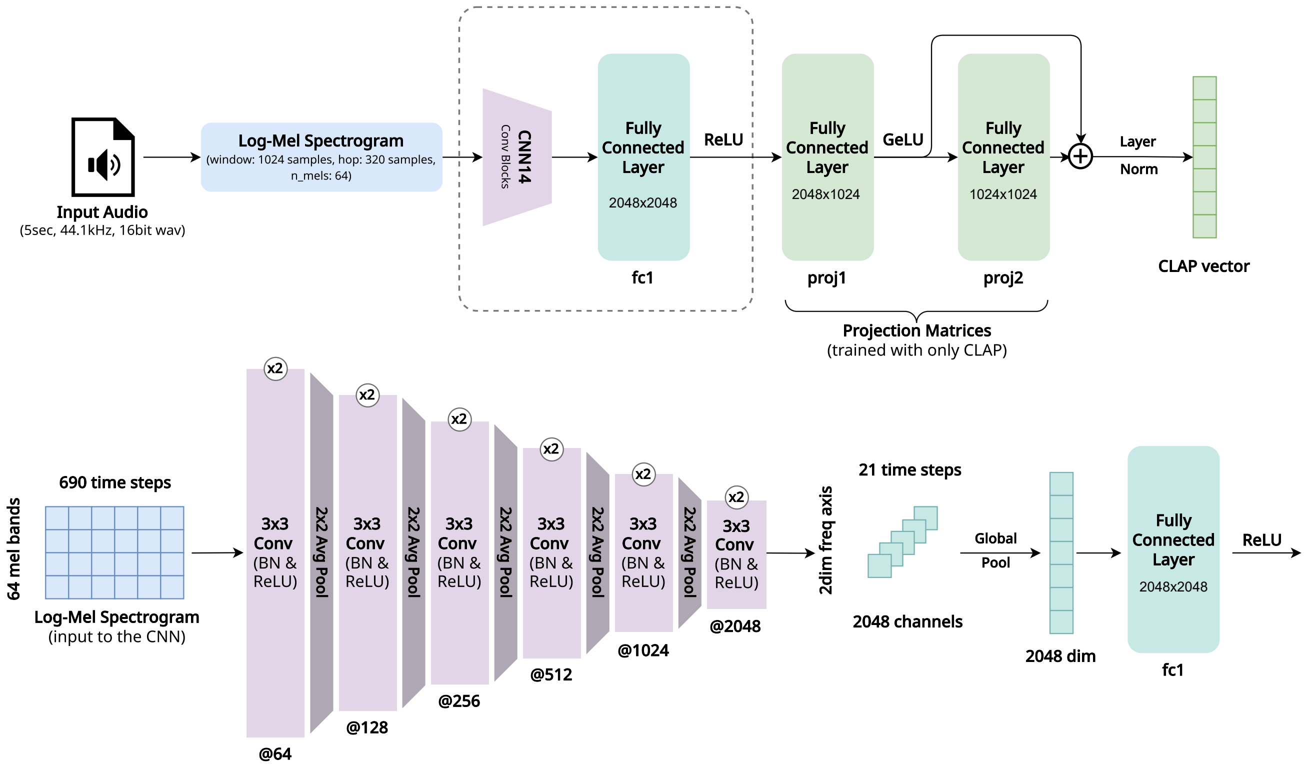

Figure 6: Architecture of the Contrastive Language-Audio Pretraining (CLAP) encoder (Elizalde et al., 2023). We focus on the CNN audio encoder in this work. **Top:** An audio is converted into a log-mel spectrogram and fed to the CNN audio encoder (shown in dashed border). Two linear projection layers are used to obtain the audio representation (CLAP vector) that is aligned with the language representation (encoded using the BERT model (Devlin et al., 2018), not shown for brevity) through the constrastive loss. **Bottom:** Shows the CNN14 audio encoder in greater detail. The CNN is pretrained on Audio Set (Gemmeke et al., 2017) and the classifier layer (`fc2` not shown here) is discarded.

A. Evaluation Tasks

In Table 3, we show the best k-Nearest Neighbor and zero-shot performance obtained on each task. We also show the prompts in the table.

For our k-Nearest Neighbor evaluation setup, we use class-balanced datasets. GTZAN and PCVC are already class-balanced, but for the other datasets, we take appropriate subsets of data such that the resulting set is class-balanced. Additionally, we stratify our dataset folds in five-fold validation such that they are class-balanced as well.

The Medley-solos-DB dataset takes short excerpts of solo instrumental from a set of songs, with multiple dataset instances belonging to the same song, and different excerpts can have very similar melodies. In order to prevent this from contributing to instrument classification performance, we also account for the song from which the dataset instance is taken from when constructing the dataset folds, such that no two folds contain excerpts from the same song. Due to a further lack of samples from the class ‘tenor saxophone’ compared to the other classes, we remove the class, leaving a total of 7 classes.

B. Compute Requirements

All experiments are conducted on RTX 2080 GPUs with 12 GB memory. The computational bottleneck in our work is clustering millions of neurons for neuronal cluster analysis; however, this is resolved by using FINCH which is efficient and uses approximate nearest neighbors.

Table 3: Best performance of the k-NN and zero-shot (ZS) classification on each dataset. Random chance (R) accuracy is shown in the last column.

Dataset	Best Performing			Accuracy (%)		
	k	Layer	Prompt	k-NN	ZS	R
NSynth (Engel et al., 2017)	1	`conv1`	"This is a ___ note"	46.9	7.1	8.3
Medley-solos-DB (Lostanlen et al., 2019)	1	`fc1`	"This is a sound of ___"	98.2	80.7	12.5
GTZAN (Turian et al., 2022)	1	`fc1`	"This is a ___ song"	75.4	30.7	10.0
PCVC (Malekzadeh et al., 2020)	7	`conv3`	"___"	30.8	4.5	4.3
Speech Commands (Warden, 2018)	1	`conv4`	"___"	40.0	2.9	2.9
LibriCount (Stöter et al., 2018)	20	`fc1`	"___ people speaking"	52.5	20.2	9.1

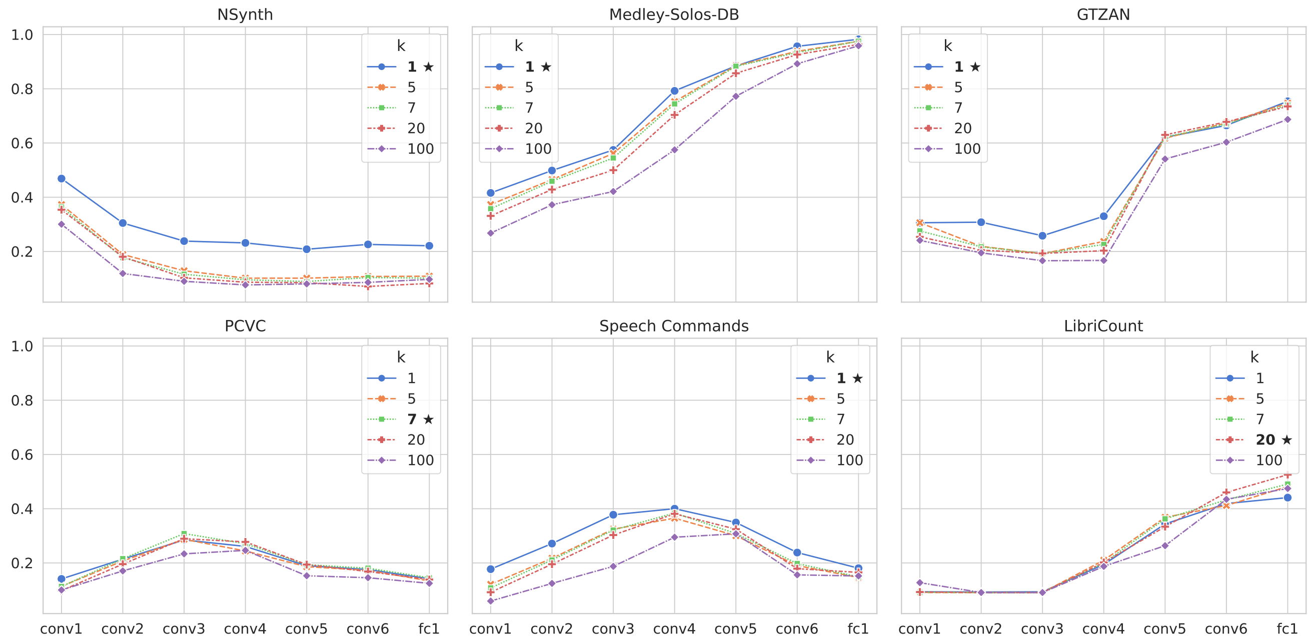

Figure 7: Layer-wise classification accuracy on different datasets with k-Nearest Neighbor classifier using different values of k. Best value of k indicated with star (★) in the legend. Only some values of k plotted for sake of clarity.

C. Effect of k in k-nearest neighbors classification

For the layer-wise classification we use values of k varying from 1 through 9, and also 20, 50, and 100. We then use the value of k which produces the best peak accuracy as the k-value for analysis. In Figure 7, we plot the layer-wise classification accuracy on the three music (NSynth, Medley-Solos-DB, GTZAN) and three speech (PCVC, Speech Commands, LibriCount) datasets used in our layer-wise analysis. In most cases, the effect of k is not very noticeable. For some tasks, such as instrument classification as speaker count estimation, larger values of k perform better. For note name classification, the best value of k is 1; the difference in performance from the next k value is noteworthy.

D. ESC-50 Layer-wise results

k-Nearest Neighbor classification accuracy in Figure 8 shows that ESC-50 peaks at the later layers, indicative of it being a high-level task focused primarily on the semantic classes. For this specific task, the value of k does not have a large effect.

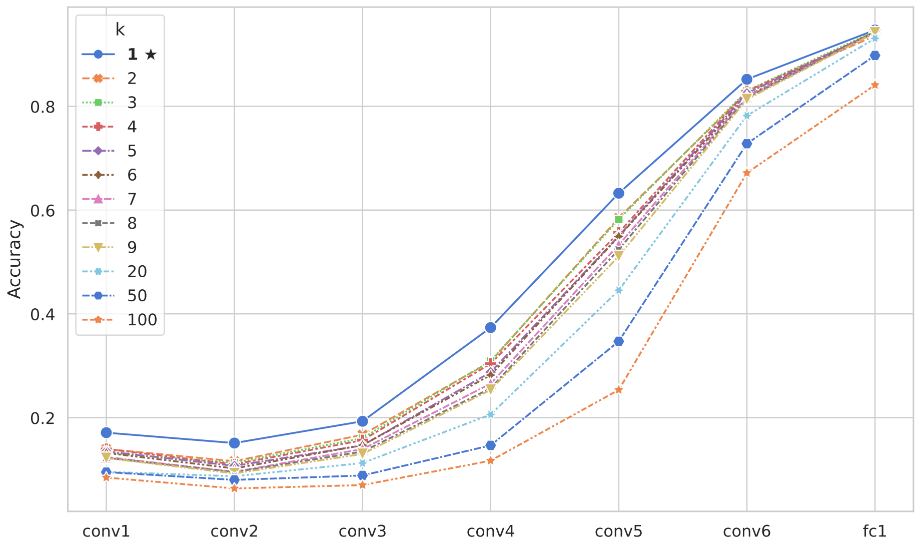

Figure 8: Layer-wise classification accuracy on the ESC-50 dataset with k-Nearest Neighbor classifier using different values of k. Best value of k indicated with star (★) in the legend.

E. Clusters Become Specific Across the Layers

Three additional figures similar to Figure 4 of the main paper are included in the supplementary. They all show similar trends for: (i) *siren* in Figure 9; (ii) *church bells* in Figure 10, and (iii) *frog* in Figure 11.

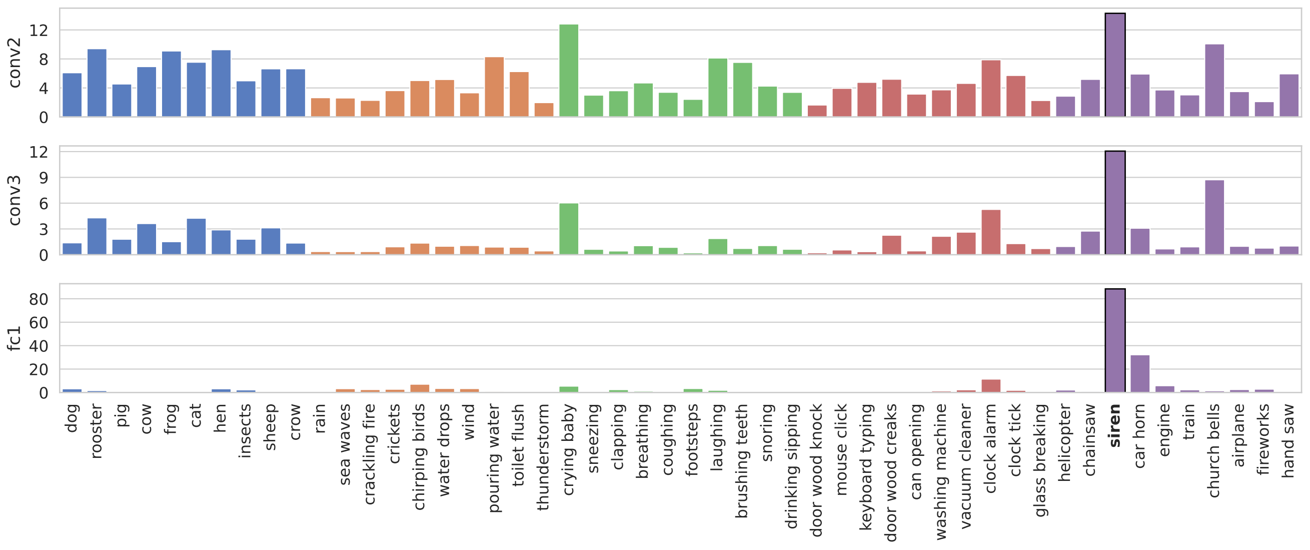

Figure 9: Clusters across layers which activate maximally for ‘siren’. Top: conv2, partition 5, cluster 13, Middle: conv3, partition 4, cluster 16, Bottom: fc1, partition 1, cluster 25.

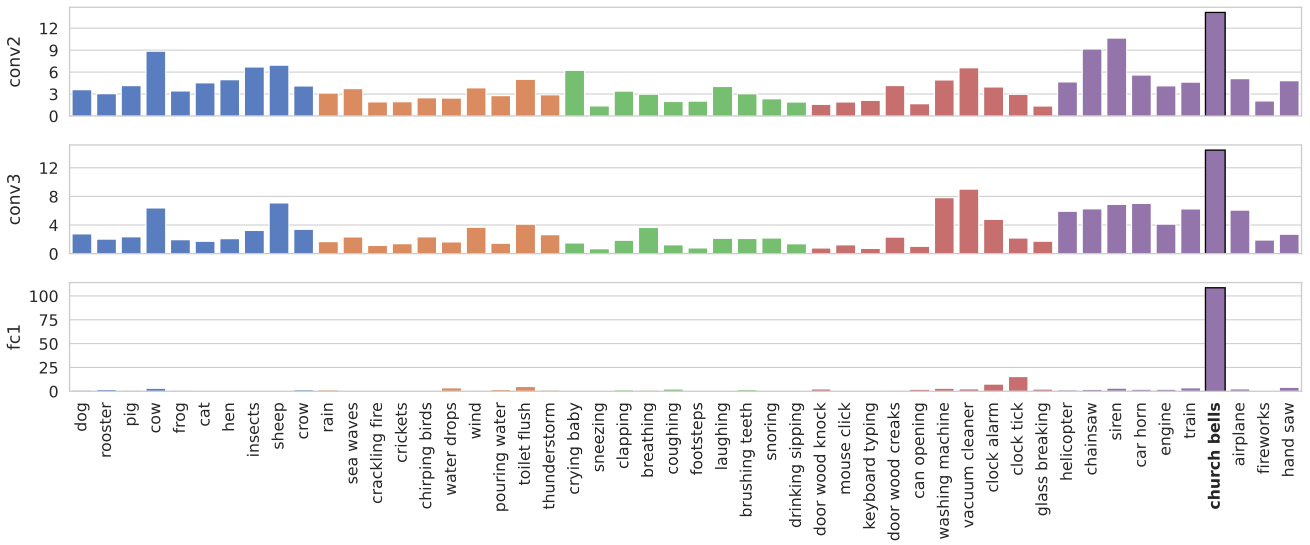

Figure 10: Clusters across layers which activate maximally for ‘church bells’. Top: conv2, partition 5, cluster 3, Middle: conv3, partition 4, cluster 29, Bottom: fc1, partition 0, cluster 81.

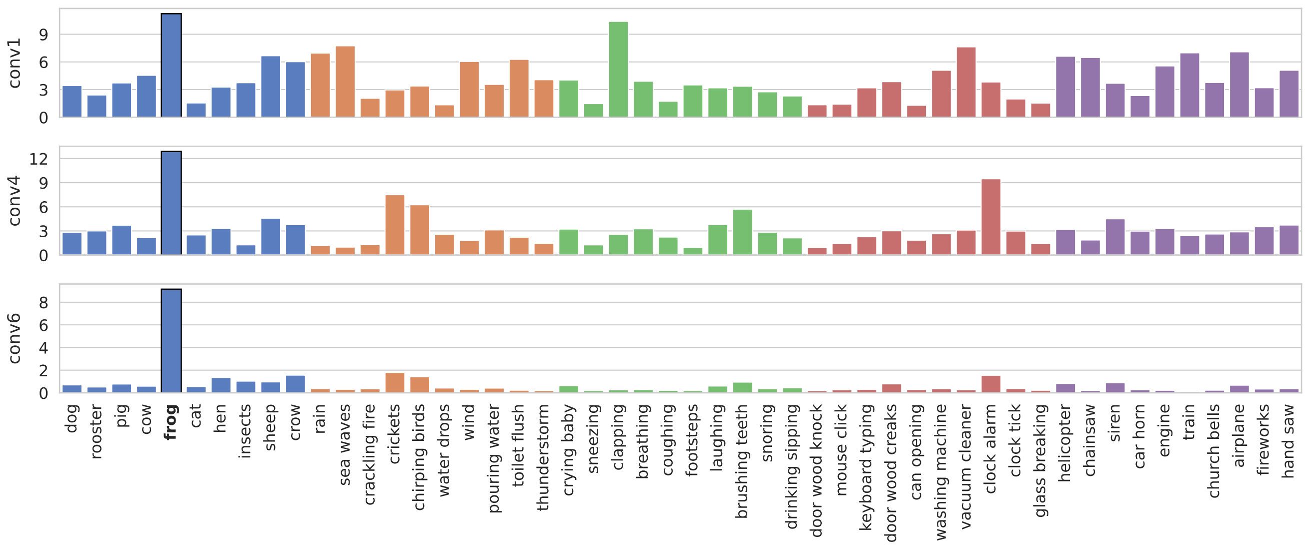

Figure 11: Clusters across layers which activate maximally for 'frog'. Top: conv1, partition 5, cluster 18, Middle: conv4, partition 4, cluster 5, Bottom: conv6, partition 4, cluster 15.