university of groningen / university college groningen

Does GradCAM XAI Method Generate Faithful Explanations For Audio Recognition CNNs?

Thesis for the BSc. Liberal Arts and Sciences
University College Groningen, University of Groningen

written by

Eman Ansari

S-4317394

supervised by
Dr. Muhamed Amin
Marco Zullich

co-assessed by
Volker nannen

June 30, 2024

Abstract

Explainable Artificial Intelligence (XAI) methods have been utilized to explain the predictions of various machine learning models including Convolutional Neural Networks (CNNs). While currently available XAI methods have proven promising in generating explanations for image based CNNs, the performance of these XAI methods in explaining the predictions of audio based CNNs has not yet been evaluated. This research uses faithfulness as a metric to comparatively evaluate the performance of GradCAM, a popular XAI technique in explaining the audio classification predictions of two CNN models, Resnet18 and Resnet101 trained on the UrbanSound8k benchmark dataset. The findings of this evaluation using the monotonicity correlation metric suggests that GradCAM explanations exhibit varying degrees of faithfulness across different architectures, with an overall tendency towards negative faithfulness scores.

Contents

1 Introduction

Audio recognition tasks have improved in performance with the latest versions of CNN architectures such as Resnet50 He et al. (2016) , VGG Simonyan & Zisserman (2014) and Inception V3 Szegedy et al. (2016). In their study on CNN performances with large audio datasets such as YouTube-100M, Hershey et al. (2017) comparatively examined five different architectures, namely, a fully connected Deep Neural Networks (DNNs), AlexNet Krizhevsky et al. (2012) , VGG, Inception, and ResNet. The fully connected DNN was used as the baseline for performance assessment. The authors prove that all the new CNN architectures, which are typically trained on image datasets, are also compatible with audio datasets and significantly outperform the fully connected network as well as prior versions of image-based CNN architectures in audio classification tasks Hershey et al. (2017) . They found that Resnet and Inception offer particularly high performances compared to the other aforementioned architectures.

Artificial Intelligence (AI) models with new architectures and promising predictive abilities are deployed across a spectrum of applications. These include; applications in vehicles to enhance safety and avoid collisions, in financial institutions for managing investment strategies and loan approvals, in healthcare settings to support doctors in diagnosing diseases, in law enforcement to aid in evidence recovery and operational ease, within military contexts globally, and in insurance sectors to assess and mitigate risks among others Ali et al. (2023). These models have to be designed to handle real world data, which is often non-linear, with millions of parameters. Such data often require networks with a large number of layers with various connectivity configurations, several filters in each layer, specific activation functions, pooling operations and additional functions that come with the combination of learning techniques employed to retrieve relevant information and identify patterns Ali et al. (2023). In general, deeper networks often perform better decision-making compared to shallow networks Ali et al. (2023). However, as the performance of emerging variants of CNN architectures increases for a variety of data types, their results also become more difficult to comprehend and understand owing to the increasing complexity of the network. Thus the increased performance comes at the expense of the black box nature of deep architectures. With such a broad scope of deep learning applications in critical sectors of the economy, the need for explanations and interpretations of model outcomes becomes imperative. For instance, in biomedicine, recommendations made by the models may concern high risk tasks such as prescribing a particular medication or assessing the potential risk of developing specific diseases Miotto et al. (2018). Understanding the results of such models—specifically, being able to transparently see which phenotypes influence predictions—is essential for gaining medical professionals' confidence in the recommendations made by the predictive system Miotto et al. (2018).

1.1 Explainable Artificial Intelligence

Explainable Artificial Intelligence (XAI) techniques emerge from this need for ease of comprehension and transparency. To this end, XAI techniques are geared towards achieving this on two specific levels: a) Interpretability; which is concerned with improving a developer's understanding of the internal functioning of a model so as to offer an insight to its decision

making, b) Explainability; which is more specifically catered to the end user's understanding of whether a model's decision can be trusted or not. The full definition of explainability as given by Ali et al. (2023) is as follows:

> The process of elucidating or revealing the decision-making mechanisms of models. The user may see how inputs and outputs are mathematically interlinked. It relates to the ability to understand *why* AI models make their decisions. The capacity to make automatic interpretations and describe the inner workings of an AI system in human terms is referred to as explainability. An explainable technique summarizes the reasons for an AI model's decision. Furthermore, a model's "Post-hoc Explainability" refers to methods/algorithms that are used to explain AI model's decisions. Ali et al. (2023)

While explainability is geared towards end users, it can also be utilized in improving interpretability for developers and will thus be the focus of this study. In their review study, Ali et al. (2023) delineate four levels at which explainability should be considered for any AI pipeline: a) data explainability, b) model explainability, c) post-hoc explainability and d) assessment of explanations. This study focuses particularly on post-hoc explainability and the assessment of explanations. As opposed to model interpretability, post-hoc explanations are concerned with improving the understanding of the model after it has already been implemented. At this stage,

> The practitioner analyzes a trained model to provide insights into the learned relationships. This is particularly challenging when the model's parameters do not clearly show what relationships the model has learned. Murdoch et al. (2019)

Generally, post-hoc methods can further be classified based on whether they generate explanations locally (explaining single predictions) or globally (explaining general dataset relationships the model has learned in the training process). For instance, feature importance is an explanation method which attempts to assign scores to features in terms of their general influence in the decision making of a model at the dataset level. In feature importance, a feature is ranked highly if rearranging its values raises the model's error. While at the local level, input attribution attempts to rank features based on their contribution to an individual prediction given an input.

Additionally, existing attribution-based methodologies to generate explanations vary in the specific approaches they employ. As identified by Ali et al. (2023), these can be broadly classified into four families of approaches: Deep Taylor Decomposition (DTD), which approximates the contributions of individual input features to the final prediction using a series of Taylor expansions; Perturbation methods such as LIME Ribeiro et al. (2016), which systematically alters or removes input features to assess the impact on the model's output; Backpropagation methods such as Gradient Weighted Class Activation Map (GradCAM) Selvaraju et al. (2017), which utilizes the forward and backward passes to calculate the attribution values for all the input features; and DeepLIFT Shrikumar et al. (2017), which assigns importance scores to input features by comparing the activation of each neuron to a reference activation.

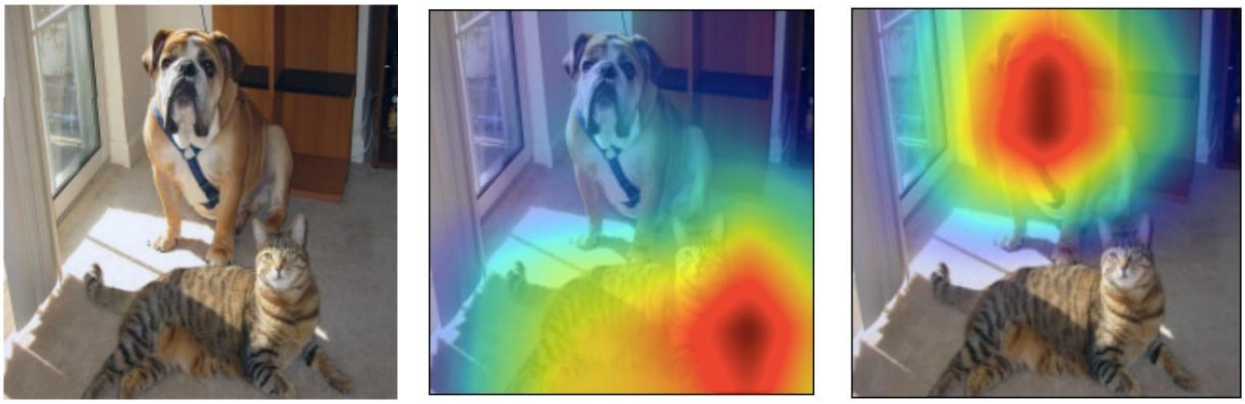

Figure 1: showing the original image as input to ResNet18 (left), the GradCAM heat map overlaid on the image for the prediction 'Cat' (middle), the GradCAM heat map overlaid on the image for the prediction 'Dog' (right). This provides a visual insight into the feature and class relationships that inform the models decision making. Figure adapted from Selvaraju et al. (2017).

This study focuses on particularly employing and evaluating GradCAM explanations. GradCAM is an input attribution method, first proposed by Selvaraju et al. (2017). It is a class discriminative method in that it attempts to identify the specific features in the spatial locations within pixel space that the model has learned to associate with a specific class Ali et al. (2023). Class discriminative methods are premised on deciphering the relationships that the model has learned between classes and features, which inform the models decision in differentiating an input's class from another. GradCAM provides insights into what these learned relationships are by generating visual explanations in the form of heatmaps, showcasing important features which cause the model to predict a specific class in accordance with what relationships it has learnt. Figure 1 shows a heatmap produced by GradCAM.

GradCAM works by weighing each feature map in the final convolutional layer according to the influence it has on the output to generate a localization heat map approximating the relative importance of that feature to a specific prediction. The weights assigned to the feature maps are computed by taking gradients of class scores present in the output layer of a trained model with respect to the activations present in each of the feature maps Selvaraju et al. (2017). GradCAM was introduced as a generalization of CAM, originally proposed by Zhou et al. (2015) to extend its applications to a larger variety of CNN architectures without the need for structural modifications to the model as in CAM. While it offers better compatibility with various models relative to CAM, its explanations are local and difficult to aggregate on a global dataset level. Ali et al. (2023). Additionally, though GradCAM has been effective in explaining image-based CNN predictions, its application to audio-based CNNs remains underexplored Akman & Schuller (2024). A formal introduction of GradCAM can be found in the Methodology section.

1.2 Evaluation of XAI

With a host of explanations provided by the aforementioned XAI methods, it becomes imperative to evaluate explanations to distinguish those that provide a truthful and sufficient insight into the decision making of a model from others that do not. From this emerges the need to define a universalized and holistic framework for evaluation of explanations. A decisive framework was proposed by Nauta et al. (2023), which delineates 12 conceptual properties (Co-12) that serve as an assessment rubric to holistically judge the quality of explanations generated by the burgeoning number of XAI methods. Figure 2 provides a list and brief description of the Co-12 properties that can be used when assessing explanations.

	Co-12 Property	Description
Content	**Correctness**	Describes how faithful the explanation is w.r.t. the black box.
		Key idea: Nothing but the truth
	Completeness	Describes how much of the black box behavior is described in the explanation.
		Key idea: The whole truth
	Consistency	Describes how deterministic and implementation-invariant the explanation method is.
		Key idea: Identical inputs should have identical explanations
	Continuity	Describes how continuous and generalizable the explanation function is.
		Key idea: Similar inputs should have similar explanations
	Contrastivity	Describes how discriminative the explanation is w.r.t. other events or targets.
		Key idea: Answers "why not?" or "what if?" questions
	Covariate complexity	Describes how complex the (interactions of) features in the explanation are.
		Key idea: Human-understandable concepts in the explanation
Presentation	**Compactness**	Describes the size of the explanation.
		Key idea: Less is more
	Composition	Describes the presentation format and organization of the explanation.
		Key idea: *How* something is explained
	Confidence	Describes the presence and accuracy of probability information in the explanation.
		Key idea: Confidence measure of the explanation or model output
User	**Context**	Describes how relevant the explanation is to the user and their needs.
		Key idea: How much does the explanation matter in practice?
	Coherence	Describes how accordant the explanation is with prior knowledge and beliefs.
		Key idea: Plausibility or reasonableness to users
	Controllability	Describes how interactive or controllable an explanation is for a user.
		Key idea: Can the user influence the explanation?

Figure 2: A list of the 12 properties (Co-12) that 'good' of explanation have as provided by Nauta et al. (2023). Given this schema, an XAI method can be judged by its 'Content' in terms of how truthful it is to the model it explains (Correctness), whether its a holistic explanation with respect to how much insight it offers (Completeness), whether it can be applicable to a wider range of models (Continuity) etc. It can also be judged by its 'Presentation' in terms of how well it presents its insights (Composition) or whether it is concise or not (Compactness) etc. Moreover, the end user of the XAI method is also crucial in the evaluation process since an explanation should cater to the user's contextual needs (Context).

1.3 Correctness

Correctness is a measure of how truthful or faithful the explanation is in revealing the reality internal to the black box of AI models. It is important to note that correctness does not relate to whether the model actually makes the correct prediction, rather it is about whether the explanation correctly describes the reasoning of the model in making the prediction Nauta et al. (2023). This property is perhaps most crucial as the very purpose of an explanation is to shed light on the internal decision making process of the model to improve our understanding of it. If the internal decision making process contradicts our understanding of it, it follows that perhaps the explanation did not help us understand it in the first place. Therefore, correctness is a necessary property which serves as a sanity check for ensuring that the explanation captures the truths internal to the model. Therefore, for the purpose of evaluating GradCAM explanations, I will focus on the property of 'Correctness'.

There are a variety of metrics available to measure faithfulness as identified by Nauta et al. (2023). These include Model Parameter Randomization Check, which is premised on tweaking the parameters of the model randomly in order to verify that the explanations capture the changes made. Other metrics are premised on changes such as tweaking the input by either deleting or perturbing it (Single Deletion), or testing the XAI method on a 'white box' model where the internal decision making process is known a-priori, and comparatively verifying if the explanation captures the decision making correctly (White box Check). Similarly to Single Deletion, Incremental Deletion is premised on the idea that for an explanation that is faithful, iteratively perturbing features of an input in the order of importance as assigned by the explanation would result in a correlated impact on the output. The Monotonicity metric is a more specific implementation of this provided by Nguyen & Martínez (2020), which utilizes the Spearman Correlation Coefficient to quantify the strength and direction of the monotonic relationship between the absolute values of feature attributions and the corresponding changes in model output. This metric is present in the Quantus python library, which offers a list of more than 25 reference metrics to focus on the evaluation of explanations Ali et al. (2023). A formal introduction of monotonicity can be found in the Methodology section. In particular, this study strives to measure how faithful GradCAM explanations are in revealing the relationships between features and classes that the model has learnt in the training process.

In their review article, Akman & Schuller (2024) point out that in the literature on XAI so far, there is a serious lack of evaluation of XAI methods when it comes to audio-based models. Due to the different nature of audio data, the application of XAI methods for explaining image or text based models is not as straightforwardly extendable to audio based-models. In general, they provide two categories of XAI methods, which are applicable for explanations of models based on audio data. These categories are a) Generic XAI methods, which are applicable to a variety of data types, and b) Audio Specific XAI methods, which are exclusively designed to handle audio based models.

They note that a crucial design choice is the type of audio representation selected, such as waveforms, spectrograms, mel-frequency cepstral coefficients (MFCCs) etc. This is because the type of representation for audio data influences methods of both data analysis and machine learning tasks. Waveforms are a time-domain representation of audio signals that describe changes in amplitude over time. They retain temporal details and phase infor-

mation, making them useful for tasks requiring precise timing information, such as speech recognition. Spectrograms, on the other hand, are a frequency-domain representation that describe how the frequency content of the audio signal changes over time. They are made by applying the Short-Time Fourier Transform to a signal in the time-domain. While useful in certain tasks that require frequency information, such as music analysis, spectrograms have a considerable information loss in terms of phase. MFCCs are a short-term power spectrum of a sound signal that is derived from spectrograms by applying a series of transformations including a Mel Filter bank, logarithmic compression and a Discrete Cosine Transform (DCT) to obtain the cepstral coefficients Gourisaria et al. (2024). Though MFCCs lack phase information too, they offer a representation that is more indicated for perceptual similarity to the human auditory perception. The review points out that due to a spectrogram's similarity with image data, spectrogram audio representations and their derivatives (MFCCs) are also compatible with Convolutional Neural Networks (CNNs) for a variety of tasks. However, the use of waveform representation is preferred as it accounts for temporal details such as phase information that would otherwise be lost in the spectrogram representation Akman & Schuller (2024). Figure 3 shows the three formats of audio data.

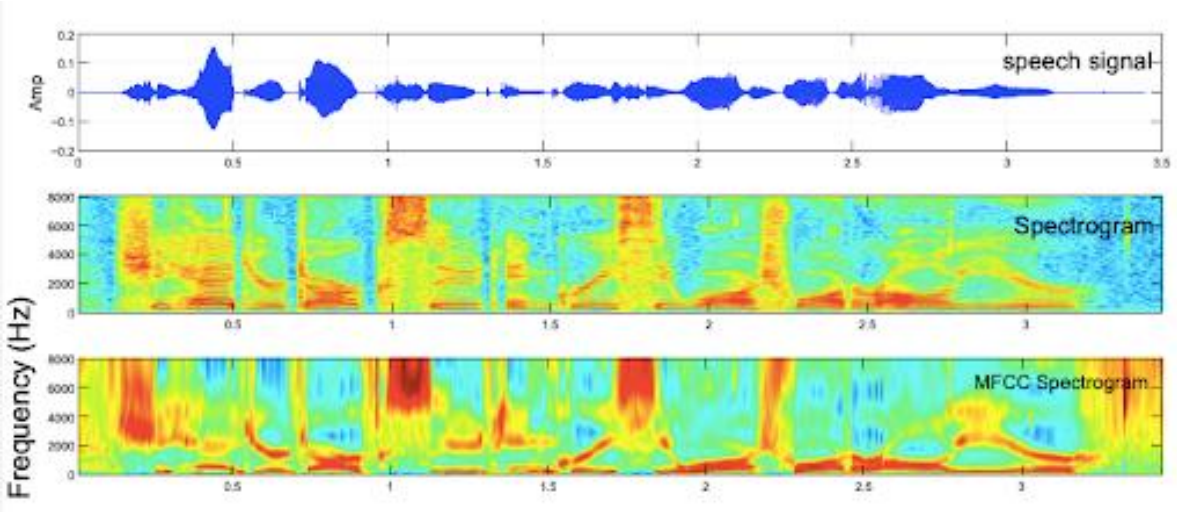

Figure 3: Presents a visual comparison of a speech signal in waveform, spectrogram and MFCC spectrogram representations. Unlike the waveform representation, the latter two do not capture the phase of the audio signal, although MFCCs are catered towards perceptual similarity to humans. Figure adapted from Fedila et al. (2018)

Hoedt et al. (2023) carried out relevant research addressing the aforementioned concerns about the lack of evaluation of XAI methods for audio-based CNN models. Their paper attempts to bridge the knowledge gap by comparatively assessing the explanations generated by several XAI explainers on both image and audio datasets across a variety of model architectures including Resnet50 and VGG. The XAI explanations are evaluated by introducing perturbations to the input data and observing how the explanations change. The specific evaluation metric employed is the label flip rate which measures the percentage of cases where the model's prediction changes (or "flips") from one class label to another due

to perturbations applied to the input data. Although this research was done before Nauta et al. (2023) presented their Co-12 assessment properties, the label flip rate metric attempts to measure the correctness property using a variation of incremental deletion. If the segments assigned the highest importance effectively explain the model's behavior, the label flip rate should be high, indicating that the model's predictions change consistently when these segments are perturbed. Their findings suggest that irrespective of the domain, task, or model architecture the label flip rates were consistently low, implying that the explanations were not able to decipher the segments of the input that actually contribute to the model's outcome Hoedt et al. (2023).

With this background in place, this study aims to address the question: How effective is GradCAM in generating faithful explanations for audio classification predictions of the ResNet18 and ResNet101 model trained on the UrbanSound8k dataset? This research addresses the knowledge gap identified by Akman & Schuller (2024) in the lack of evaluation of XAI methods for audio-based models. Though the related work by Hoedt et al. (2023) also addresses this concern, the methods tested by them did not include GradCAM specifically and the evaluation metrics employed were prior to the decisive framework for evaluation established by Nauta et al. (2023). The findings from this research provide general insights into the applicability and limitations of post-hoc explainability methods like GradCAM in domains beyond image recognition. This can help determine if the insights provided by GradCAM for image data are similarly applicable to audio data. The results of this study imply that upon evaluation using the Monotonicity Metric as a measure of the faithfulness, GradCAM explanations demonstrate and overall low faithfulness score for both ResNet18 and ResNet101.

2 Methodology

2.1 Dataset

The dataset I use for the purpose of this study is the benchmark audio dataset UrbanSound8K. This dataset was created by Salamon et al. (2014) to address two particular challenges in urban sound classification research a) the lack of a common taxonomy and b) the scarceness of large, real-world, annotated data Salamon et al. (2014). Based on their taxonomy, this dataset contains 8732 labeled audio samples in .wav format from the Freesound online repository. All samples are from real field recordings and have a maximum duration of 4 seconds, while some are less than 4 seconds. These samples are categorized into ten classes of common urban sounds: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren, and street music. The dataset comes with a csv file which contains the metadata of each sample including the sound class label, the class ID and the folder number it exists in, facilitating the use of cross-validation techniques. In creating the dataset, Salamon et al. (2014) take into consideration that it should be "large and varied in terms of sounds and recording conditions such that it will be useful for training scalable algorithms capable of analyzing real data from sensor networks or multimedia repositories" Salamon et al. (2014). Additionally they also consider that large discrepancies in samples per class are undesirable and thus set a maximum limit of 1000 slices per class,

with the lowest number of samples (374) for the 'gunshot' class. A complete breakdown of class distribution can be found in Figure 4. This dataset is a standardized benchmark for environmental sound classification which numerous studies, including those by Salamon et al. (2014), Piczak (2015), and Ahmed et al. (2020), have utilized to advance research in sound classification using various neural network architectures. The choice of this dataset to investigate the faithfulness of GradCAM explanations for CNN-based audio classifiers is based on the dataset's proven utility in sound classification research.

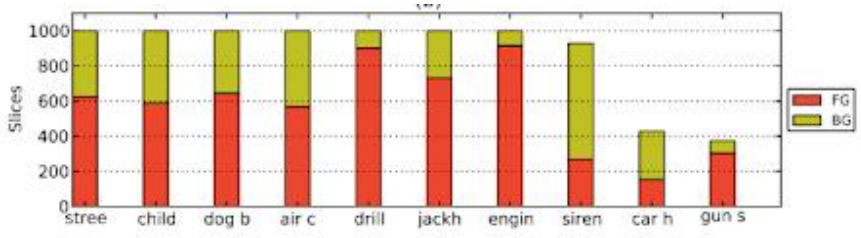

Figure 4: Slices per class in UrbanSound8K. Breakdown by foreground (FG) / background (BG). Figure adapted from Salamon et al. (2014).

2.2 Data Preprocessing

In the preprocessing pipeline of the UrbanSound8K dataset, multiple steps are implemented to standardize the audio data before training the neural network. The **Dataset** abstract class from PyTorch's **torch.utils.data** module is used as an interface for a customized dataset class called **'UrbanSound8k'** which handles the data preprocessing steps. These steps include channel normalization to ensure that audio files with multiple channels are averaged to create a single mono channel using torch.mean. Subsequently, the resampler transform (torchaudio.transforms.Resample) is applied to the audio files to standardize the sampling rate at 44,100 Hz. Following this, an MFCC (Mel-Frequency Cepstral Coefficients) transform is applied to the waveform to convert the audio waveform into a visual representation. Important parameters for the transformation include n_mfcc=13 which determines the number of Mel-Frequency Cepstral Coefficients (MFCCs) extracted from each audio frame. A larger number of cepstral coefficients retains more detailed spectral information but this comes at the expense of greater computational complexity. The length of the window over which the Fourier Transform is applied determines the resolution of the spectrogram and is fixed at n_fft=2048. Though with the limitation of losing phase information, MFCCs are chosen as the type of representation due to their perceptual similarity to the human audible perception and their increased compatibility with CNNs as compared to waveform representation. Lastly, to standardize the input length for the neural network, the MFCC outputs are padded or truncated to a maximum length of 400. The outputs from this pipeline are then used as the inputs to the neural network.

2.3 CNN model

The CNNs architectures used for the purpose of this research are the ResNet18 and Resnet101, two variants of the Residual Networks architecture proposed by He et al. (2016) at the IEEE

Conference on Computer Vision and Pattern Recognition (CVPR). As the name suggests, ResNet-18 and ResNet101 are composed of 18 and 101 layers respectively. These layers include convolutional layers, pooling layers, fully connected layers, and shortcut connections. He et al. (2016) introduce shortcut connections as the distinguishing novelty of ResNet's architecture which effectively solves the vanishing gradient problem that emerges during the training of neural networks that are very deep. Computing the gradients of each layer is a necessary step in training since it informs the direction in which the weights in a layer should be adjusted so that the value of the loss function in the final output layer is minimized. The gradient of a particular layer is computed as the vector of partial derivatives of the loss function with respect to the network's weights. Since a given layer's gradient is a product of the gradients of the previous layers, during back propagation, the presence of additional layers in deeper networks implies that the gradients become excessively small. Consequently the adjustment of weights in the latter stage of back propagation is not informed by the correct gradients resulting in the network's inability to learn effectively. He et al. (2016) add shortcut connections which serve as a method to retain the gradients of a layer. This connection does not apply any transformation to the input, rather it only provides the input a way to bypass the one or more layers and be added to the output of the final layer it bypassed. In this manner, if we anticipate running into negligible gradients in certain layers during back propagation, adding this connection would by default transfer the value of the input from previous layers to the one who's gradients have vanished. This means that even if the gradient vanishes, the network will always be able to rely on the unchanged mapping of the input from the shortcut connection to inform the adjustment needed to the layers weights. Figure 5 provides a visualization of this process.

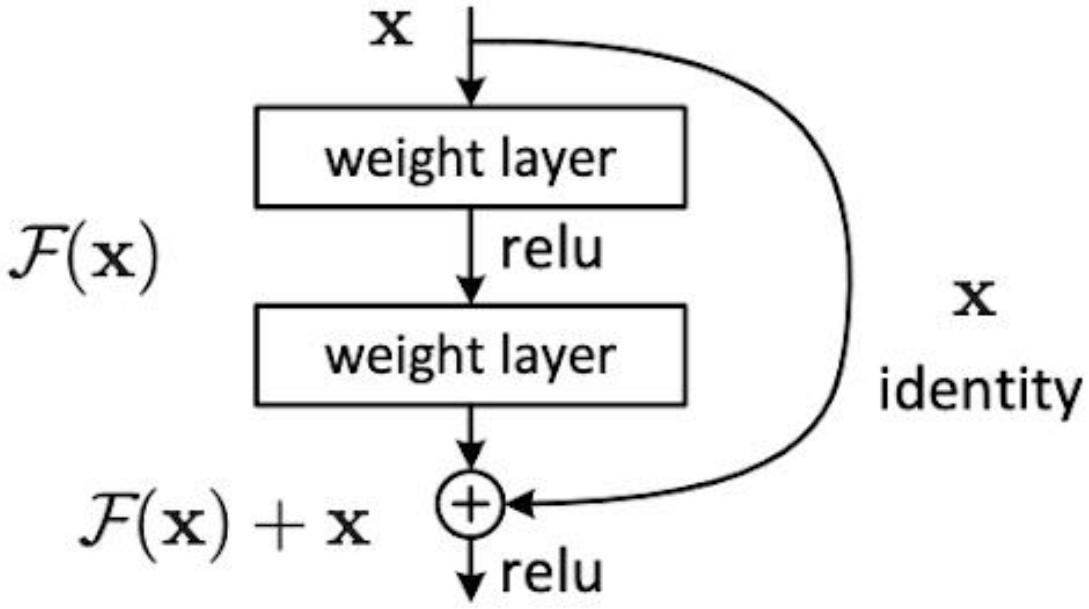

Figure 5: x is the input passing into a layer, $f(x)$ is the output of that layer, and $f(x)+x$ is the output of the subsequent layer because there is a shortcut connection before the first layer to the output of the next layer. If $f(x)$ happens to be 0, output of the final layer will still be equal to x. Figure adapted from He et al. (2016).

The ResNet architectures proposed by the authors maintain state-of-the-art performance on various image classification benchmarks such as ImageNet while still maintaining feasible optimization and lower complexity. While the deeper ResNet101 network is selected for its ability to capture more complex features owing to the larger number of layers it has, the combination of the two networks with varying depths are selected for a comparative analysis of the explanations from both variants.

Two separate classes are made for both networks and implemented using PyTorch's nn.Module. For the specific use case on the UrbanSound8k dataset, the architectures of both ResNet18 and ResNet50 remain largely unchanged except a simple modification to the first and last convolutional layers. The first convolutional layer initially expects input tensors with 3 channels but has to be changed to 1 channel to enable the models to process the aforementioned 2D MFCC inputs. Subsequently the final fully connected layer is also adjusted and made to match the number of output classes (10) specific to the UrbanSound8K dataset. During the training process, the loss function used is CrossEntropyLoss, and the optimizer is AdamW with a learning rate of 0.001. The models were trained on the first 9 folders in the dataset while the last folder (fold10) was kept hidden for testing. With a batch size of 16, the models are trained over 30 epochs, considering that sufficiently high accuracies are achieved in this duration.

2.4 Explanations from GradCAM

For generating the explanations of the aforementioned CNN's predictions, I used Gradient Weighted Class Activation Map (GradCAM), initially proposed by Selvaraju et al. (2017). GradCAM is a coarse localization method which is a generalization of the CAM method originally proposed by Zhou et al. (2015). CAM is a model specific XAI method only applicable to particular CNN architectures as it involves structural modifications to the CNN architecture such as removing the head of the network, hooking up a new global average pooling (GAP) layer and adding a fully connected layer at the end of the CNN. These modifications are done to compute the weights that are later assigned to the feature maps in the final convolutional layer. GradCAM addresses these challenges and effectively extends this applicability to a broader range of architectures. The following sections introduce GradCAM conceptually and provide the necessary mathematical formulae.

GradCAM (Selvaraju et al. (2017)) proposes a novel method to compute the weights required in generating the localization heat maps. Instead of depending on an additional GAP layer and a fully connected layer to then extract the weights from between them, GradCAM removes this dependency by utilizing gradient information, which is readily available during backpropagation, to infer the importance of each neuron in the final convolutional layer. The only condition that GradCAM imposes is that the layers after the final convolutional layer must be mathematically differentiable. With the raw probability score of each class that flows into the output layer (before the final activation function i.e softmax in this case) we can weigh each of the feature maps according to the influence they have on the output. The difference between CAM and GradCAM is in how the feature maps are weighted to make the final heatmap.

In the context of GradCAM, gradient refers to the partial derivative of y_c (the score of class C) with respect to the feature maps. Then we back propagate and end up with a tensor that has the same size as the target feature maps. A global average pooling is applied to these back propagated gradient layers resulting in a number of scalars. These scalars are used to weight all those feature maps by simply multiplying them to the corresponding feature maps as in CAM, which results in a projection to a single 2D map. An important assumption to note in doing this is that the stronger the partial derivative of y_c with respect to the feature maps, the more significant influence the k-th channel of the final feature map has on the y_c. We expect the feature maps with the higher score to have more positive gradients. A derivation of GradCAM is given by

Computing the gradients of class scores Y_c with respect to the activation A_{ij}^k:

$$\frac{\partial y_c}{\partial A_{ij}^k} \tag{1}$$

Global Average Pooling to get importance weights α_{kc}:

$$\alpha_{kc} = \frac{1}{Z} \sum_i \sum_j \frac{\partial y_c}{\partial A_{ij}^k} \tag{2}$$

Generating the heatmap by multiplying the importance weights with activations and a

final ReLU:

$$\text{Grad-CAM}_c = \text{ReLU}\left(\sum_k \alpha_{kc} A^k\right) \tag{3}$$

The reason for applying a ReLU function is to focus on the features that have a positive influence on the class of interest. The notable assumption here is that negative values more likely contribute to other classes which are not supposed to be present in the final saliency map. The final output of GradCAM is a heat map with the same size as that of the convolutional feature maps used to compute this gradient. Hence it has to be resized to match the dimensions of the original image to prepare for overlaying. In this way, Grad-CAM incorporates the gradient information to dynamically compute the importance of each spatial location, making it more compatible with a variety of architectures.

2.5 Implementation of GradCAM

Using the pytorch-grad-cam library by Jacob Gildenblat to generate GradCAM explanations for 50 random samples from the test dataset. GradCAM is initialized with the last convolutional layer of the ResNet model as the target layer. These explanations are then saved into a NumPy file as a stack of 50 arrays for further analysis. Also included in the NumPy file are the corresponding original MFCC arrays for all 50 samples and the list of corresponding ground truths. Figure 3 shows a visualization of the resulting explanation generated from a random sample in the test set, one explaining the predictions of ResNet18 (figure 3a) and the other explaining predictions for ResNet101 (figure 3b the other ResNet101.

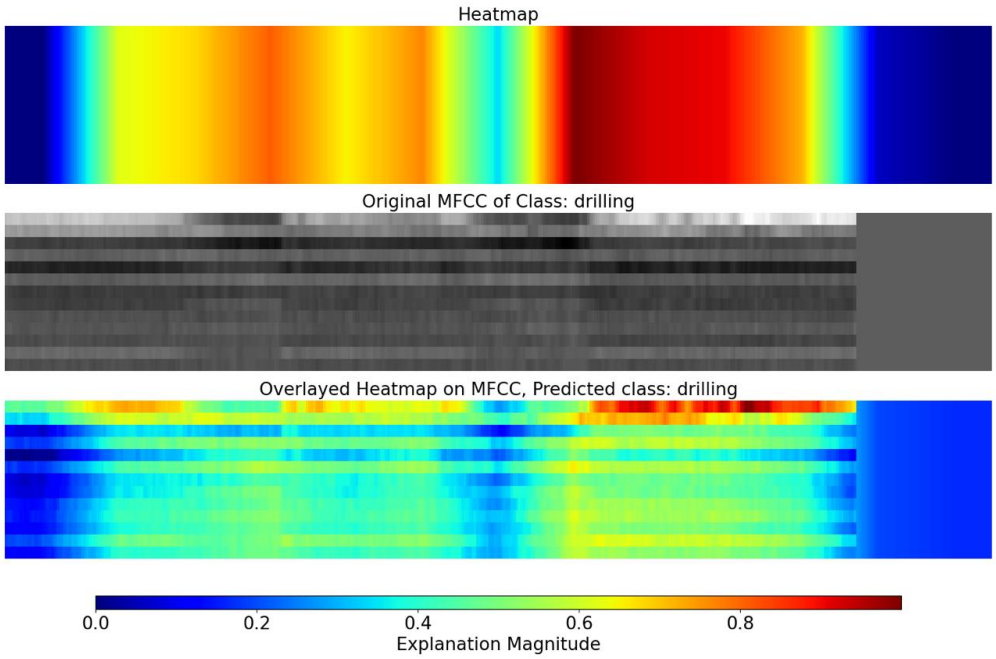

Figure 6: Visual comparison of the course localization heat map generated by GradCAM (first bar) for a single audio sample highlighting important regions, the original MFCC (second bar) indicating the ground truth of the sample is the 'drilling' class, and the heatmap overlaid on the image (third bar) indicating the pixels of importance that contribute to the predicted 'drilling' class. The color bar (bottom most) indicates the explanation magnitude scale, blue being the lowest importance and red being the highest. These explanations are for ResNet18.

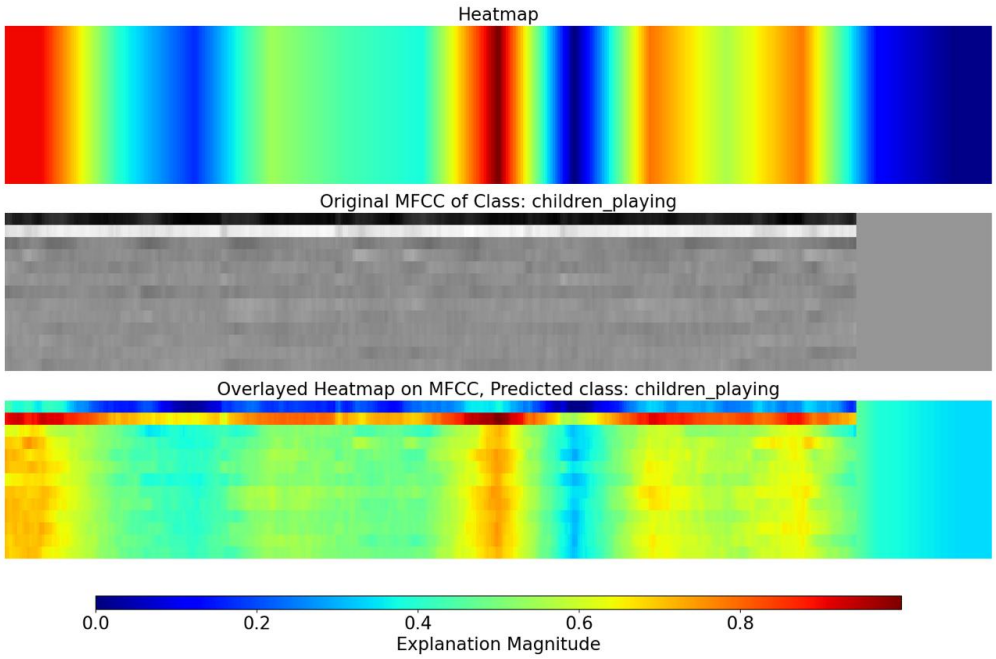

Figure 7: Visual comparison of the course localization heat map generated by GradCAM (first bar) for a single audio sample highlighting important regions, the original MFCC (second bar) indicating the ground truth of the sample is the 'children_playing' class, and the heatmap overlaid on the image (third bar) indicating the pixels of importance that contribute to the predicted 'children_playing' class. The color bar (bottom most) indicates the explanation magnitude scale, blue being the lowest importance and red being the highest. These explanations are for ResNet18.

2.6 Evaluation of Explanation

In order to evaluate explanations, the monotonicity metric proposed by Nguyen & Martínez (2020) is used. Monotonicity essentially tests whether features that are considered important by GradCAM are truly substantial in the model's prediction when perturbed. Monotonicity is therefore a measure of how faithful the explanations are with respect to the model they promise to explain. According to Nguyen & Martínez (2020);

the importance of a feature should be proportional to how imprecise would the prediction be if we did not know its value. They propose the use of Spearman's rank correlation coefficient S to quantify the relationship between the importance of the feature and the error in the model's prediction. This metric serves as a reassurance that the attributions produced by GradCAM align with the actual influence of features on the model's output. The higher the value of S, the stronger the correlation between the explanations and the error in the model's performance. Spearman's correlation coefficient is given by:

$$\rho S(\mathbf{a}, \mathbf{e}) \tag{4}$$

Where: $\mathbf{a} = (\ldots, |a_i|, \ldots)$ is a vector of attribution values given by GradCAM for i features and $\mathbf{e} = (\ldots, \mathbb{E}(u(y*, \mathbf{f}_i); \mathbf{X}_i|\mathbf{x}_i^*), \ldots)$ is a vector of the corresponding expected performance impact upon perturbing each feature in the heatmap.

To get this correlation, we first calculate $\mathbf{e}$ for all attribution values a_i:

$$\mathbb{E}(u(y^*, \mathbf{f}_i); \mathbf{X}_i|\mathbf{x}_{-i}^*) \tag{5}$$

Where: $\mathbb{E}$ denotes the expected value operator which calculates the average outcome of a random variable based on its probability distribution. $\mathbf{u}(y^*, \mathbf{f}_i)$ denotes the uncertainty which in this case is the inverse of the confidence in the predicted class. y^* denotes the predicted class for a specific input x^*. $\mathbf{f}_i$ is the restriction of the function f to the feature i, which fixes all feature values except x_i^*. $\mathbf{X}_i|\mathbf{x}_{-i}^*$ refers to the distribution of the i-th feature given that the other features are fixed at their specific values x_i^*.

Each feature value $\mathbf{X}_i$ is perturbed in the input by altering or masking the pixels while keeping all other pixel values the same. In this case uniform perturbation is applied. The perturbed image is then passed into the model and the change in the model's performance is computed by subtracting the model's cross-entropy loss given the perturbed input from the cross-entropy loss given the original input. The expected performance impact is calculated for each of the features and then aggregated to produce the vector e.

The Spearman's Rank Correlation Coefficient can then be calculated using the two vectors a and e by the following equation:

$$\rho S = 1 - \frac{6 \sum d_i^2}{(N^2 - 1)N} \tag{6}$$

Where: d_i is the difference between the ranks of each pair of values $\mathbf{a}_i$ and $\mathbf{e}_i$. $\mathbf{N}$ is the number of features.

2.7 Implementation of Monotonicity Metric:

The aforementioned steps to calculate faithfulness scores are implemented using the quantus.MonotonicityCorrelation module of the Quantus Python library by Anna Hedström. The library contains the monotonicity metric along with 25 other referenced metrics for evaluating explanations. Within the module, number_of_samples parameter is set to 10, implying that 10 perturbed versions of the input for each feature will be generated to assess the impact on the output. The features_in_step parameter is set to 13, implying that only 13 features will be perturbed iteratively. The perturb_baseline parameter is set to "uniform" indicating that the values for the perturbed features will be taken from a uniform distribution. This is beneficial because as in the context of audio data which is temporal in nature, perturbing the MFCCs by values from a uniform distribution provides for a relatively natural variation in the audio signal.

3 Results and Discussion

In this section, the results obtained after applying the monotonicity correlation metric to GradCAM explanations generated for both ResNet18 and ResNet50 architectures are presented. The set of explanations generated for each architecture contains 50 explanations

corresponding to 50 random samples from the test set. The faithfulness scores for Grad-CAM explanations are calculated for each set, producing 50 corresponding scores. The density plots are produced as shown in figure 4, visualizing the distribution of the scores obtained.

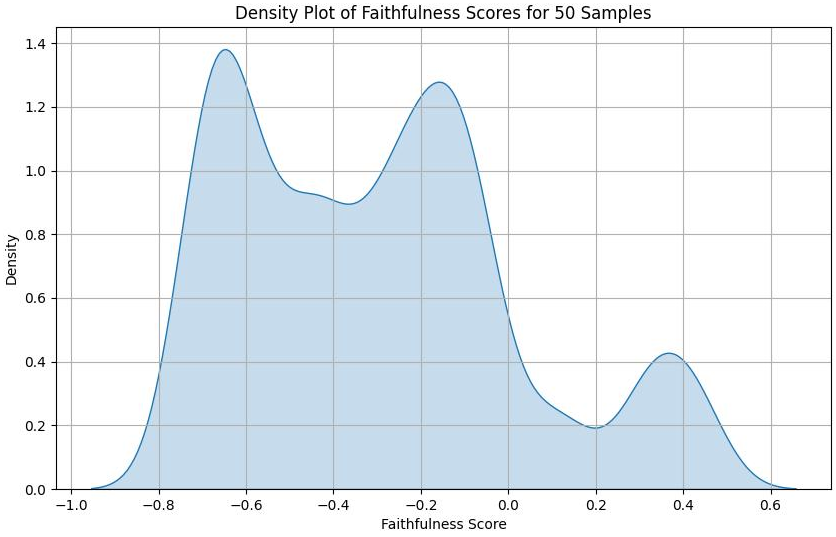

Figure 8: The faithfulness score distribution of 50 random explanations generated for ResNet18 predictions from the test set. The resulting distribution indicates that the majority of the faithfulness scores obtained from 50 samples are less than 0, with the highest density of scores (approximately 1.38) concentrated between a score range of -0.5 to -0.7. The second highest density of scores (approximately 1.28) is found between the score range of -0.1 to -0.3. Conversely, on the positive side of the distribution, with a density of approximately 0.4, the score ranges between 0.3 and 0.5. The density plot is visibly skewed with two large clusters around negative scores, indicating that most of the explanations are unfaithful to the internal relationships between features and classes learnt by ResNet18.

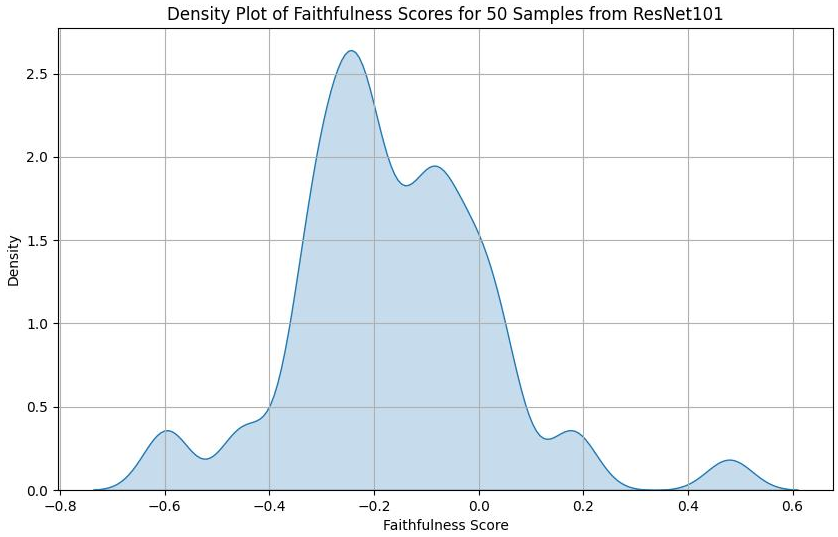

Figure 9: The faithfulness score distribution of 50 random explanations generated for ResNet101 predictions from the test set. The resulting distribution indicates that the majority of the faithfulness scores obtained from 50 samples are less than 0 with the highest density of scores (approximately 2.7) concentrated between a score range of -0.1 to -0.3. The second highest density of scores (approximately 1.9) is found between the score range of -0.1 to 0. Conversely, on the positive side of the distribution, with a density of approximately 0.2 to 0.3, 2 clusters are found around the faithfulness score of 0.5 and 0.17 respectively. The density plot is more skewed towards negative faithfulness scores compared to the density plot for ResNet18 explanations. This implies that Grad-CAM explanations are even less faithful to the internal relationships between features and classes learnt by ResNet101 as compared to that of ResNet18.

3.1 Interpretation of Results

The peaks in the negative score ranges imply that most heat maps generated by GradCAM do not assign the correct importance scores to features of the input MFCCs that the model bases its predictions on. The clusters of faithfulness scores found around the negative values can be interpreted as a common failure mode underlying the GradCAM methodology.

While GradCAM offers a novel method to generalize CAM and extend its applications to a wider range of CNN architectures, the aforementioned failures also raise significant concerns. The current methodology employed to compute feature importance scores by back propagation of gradients seems to entail a significant information loss with respect to the models internal decision making.

In addition to this, though only the two architectures are compared in this study, a comparative analysis between shallow and deep architectures suggests that as the complexity and depth of a given architecture increases, the explanations become progressively unfaithful, as reflected in the higher density of negative scores in ResNet101. This observation exacerbates the challenge of explainability further by reaffirming the issue of the trade-offs

between model complexity and explainability in deep neural networks.

3.2 Limitations and Future Directions

The results indicate the need for further fine tuning the methods employed for generating explanations. Perhaps a more granular diagnostic approach can be taken to examine the dataset, input features and various perturbation baselines to lend further insights into identifying potential commonalities between unfaithful explanations or conversely between faithful ones. Alternatively, an examination of the trends GradCAM faithfulness can be explored for other representations of audio data to offer a more holistic comparative analysis. Moreover, the performance of GradCAM can also be tested in a controlled experiment where only the depth of the explained networks is varied in an incremental manner to validate whether there is a correlation between unfaithful GradCAM explanations and the depth of the network.

4 Conclusions

To conclude, this evaluation using the monotonicity correlation metric suggests that GradCAM explanations exhibit varying degrees of faithfulness across different CNN architectures trained on audio data. The findings illustrate that explanations for both architectures (ResNet18 and ResNet101) have an overall tendency towards negative faithfulness scores, with explanations for ResNet101 having a significantly greater density of negative scores comparatively. This implies serious shortcomings in GradCAM's explanations in their inability to decipher the internal relationships between features and classes learned by the models during the training process. Moreover, the distribution for ResNet101 being more skewed towards negative scores suggest that GradCAM explanations often fail to faithfully represent the decision-making process of the models, particularly for deeper and more complex architectures. A more controlled experiment with incremental increases in architecture depth can provide further insight to this claim. Future work could explore alternative explanation or diagnostic approaches that offer more granular insight to potential commonalities between unfaithful explanations.

References

Ahmed, M. R., Robin, T. I., & Shafin, A. A. (2020). Automatic environmental sound recognition (aesr) using convolutional neural network. *International Journal of Modern Education & Computer Science*, *12*(5).

Akman, A., & Schuller, B. W. (2024). Audio explainable artificial intelligence: A review. *Intelligent Computing*, *2*, 0074.

Ali, S., Abuhmed, T., El-Sappagh, S., Muhammad, K., Alonso-Moral, J. M., Confalonieri, R., ... Herrera, F. (2023). Explainable artificial intelligence (xai): What we know and what is left to attain trustworthy artificial intelligence. *Information fusion*, *99*, 101805.

Fedila, M., Bengherabi, M., & Amrouche, A. (2018). Gammatone filterbank and symbiotic combination of amplitude and phase-based spectra for robust speaker verification under noisy conditions and compression artifacts. *Multimedia tools and applications*, *77*, 16721–16739.

Gourisaria, M. K., Agrawal, R., Sahni, M., & Singh, P. K. (2024). Comparative analysis of audio classification with mfcc and stft features using machine learning techniques. *Discover Internet of Things*, *4*(1), 1.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., ... others (2017). Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 131–135).

Hoedt, K., Praher, V., Flexer, A., & Widmer, G. (2023). Constructing adversarial examples to investigate the plausibility of explanations in deep audio and image classifiers. *Neural Computing and Applications*, *35*(14), 10011–10029.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012, 01). Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, *25*. doi: 10.1145/3065386

Miotto, R., Wang, F., Wang, S., Jiang, X., & Dudley, J. T. (2018). Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, *19*(6), 1236–1246.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, *116*(44), 22071–22080.

Nauta, M., Trienes, J., Pathak, S., Nguyen, E., Peters, M., Schmitt, Y., ... Seifert, C. (2023). From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *ACM Computing Surveys*, *55*(13s), 1–42.

Nguyen, A.-p., & Martínez, M. R. (2020). On quantitative aspects of model interpretability. *arXiv preprint arXiv:2007.07584*.

Piczak, K. J. (2015). Environmental sound classification with convolutional neural networks. In *2015 ieee 25th international workshop on machine learning for signal processing (mlsp)* (pp. 1–6).

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1135–1144).

Salamon, J., Jacoby, C., & Bello, J. P. (2014). A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd acm international conference on multimedia* (pp. 1041–1044).

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: visual explanations from deep networks via gradient-based localization. *International journal of computer vision*, *128*, 336–359.

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning important features through propagating activation differences. In *International conference on machine learning* (pp. 3145–3153).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2818–2826).

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2015). Learning deep features for discriminative localization. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2921–2929).

By submitting this thesis/ uploading this document to the Student Portal / Brightspace, I certify:

- that I am the author of this document,
- that nothing was taken from other sources without proper references,
- that this document is the result of my own discussions and preparations,
- that I have not used AI tools except when this was part of the research question.