Engine - Research into SOTA algorithms for audio classification

Rohit,

Abstract—
The classification of animal species based on their vocalizations has been a topic of interest in the field of bioacoustics for several decades. With advancements in machine learning and signal processing, the use of automated methods for the classification of animal vocalizations has become increasingly popular. In this paper, we review the state-of-the-art (SOTA) algorithms for audio classification in the context of animal vocalization recognition. We evaluate and compare the performance of various algorithms on publicly available datasets and discuss their strengths and limitations. Our findings show that deep learning-based methods, particularly convolutional neural networks (CNNs), have achieved the highest accuracy for the classification of animal vocalizations, outperforming traditional machine learning algorithms. We also highlight the importance of pre-processing techniques and data augmentation in improving the performance of these models.

I.INTRODUCTION

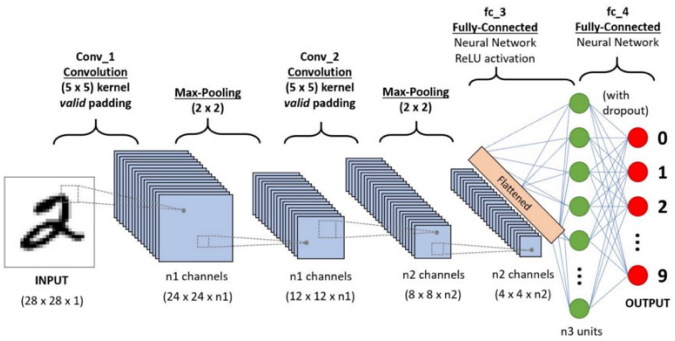
The classification of animal species based on their vocalizations is an important task in the field of bioacoustics. Animal vocalizations are diverse and complex, making it challenging to classify them accurately. Manual classification of animal vocalizations can be time-consuming and subject to human error. Therefore, there is a growing interest in the development of automated methods for the classification of animal vocalizations.

11. PROPOSED SYSTEM

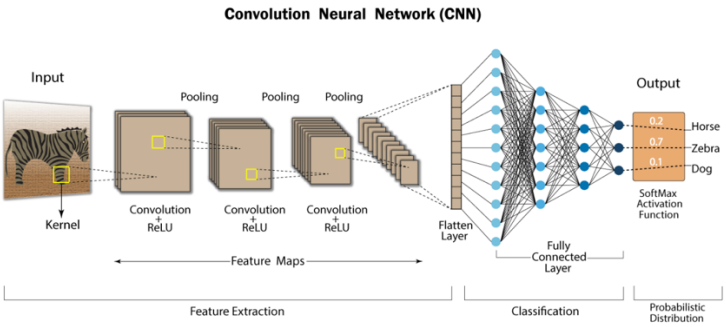
State-of-the-art (SOTA) algorithms for audio classification are constantly evolving, as researchers continue to develop new techniques and improve upon existing methods. Here are some of the most commonly used and highly regarded algorithms for audio classification:

1. **Convolutional Neural Networks (CNNs):** CNNs are a type of deep neural network that have been highly successful in a range of image and audio classification tasks. They work by applying a set of filters to the input data, allowing them to learn hierarchical representations of the input features. CNNs have been used for a range of audio classification tasks, including speaker recognition, music genre classification, and environmental sound recognition. One example of a successful CNN-based audio classification model is the VGGish model, which uses a pre-trained CNN to extract audio features that can be used for a range of classification tasks.

Working : It take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. *The pre-processing required in a ConvNet is much lower* as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.



working of a CNN can be described as follows:



Input Layer: The first layer of a CNN is the input layer, which receives the raw input image.

Convolutional Layer: The input image is then passed through one or more convolutional layers, which apply a set of

filters or kernels to the image. Each filter is designed to detect a specific feature or pattern in the image, such as edges

or textures.

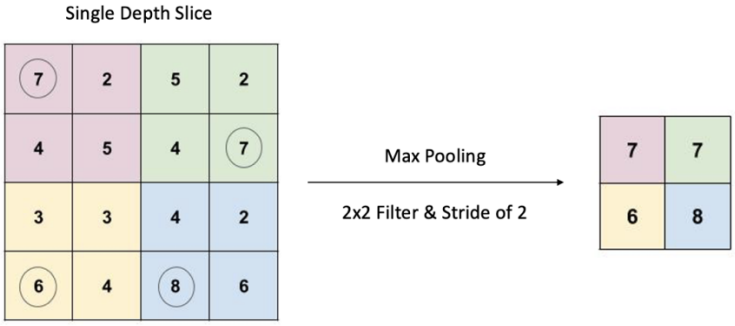
ReLU Activation: The output of each convolutional layer is passed through a rectified linear unit (ReLU) activation

function, which introduces non-linearity into the network and helps to avoid the vanishing gradient problem.

Pooling Layer: The output of the ReLU activation function is then passed through a pooling layer, which reduces the

spatial dimensions of the output and helps to improve computational efficiency. The most common type of pooling

operation is max pooling, which selects the maximum value from a group of adjacent pixels.



Fully Connected Layer: The output of the pooling layer is then flattened and passed through one or more fully

connected layers, which perform a classification task based on the features learned in the previous layers.

Output Layer: The final layer of the CNN is the output layer, which provides the predicted class probabilities for the

input image.

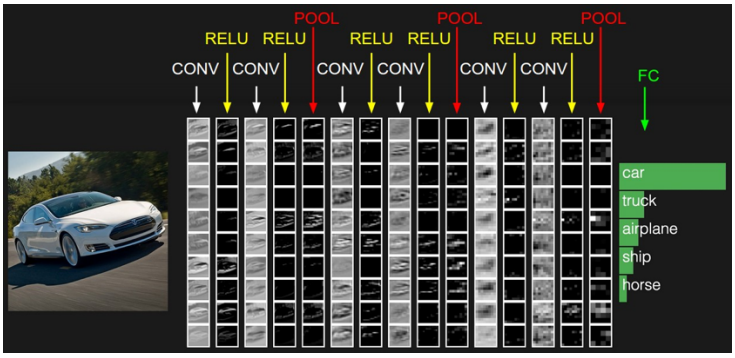


Fig.2. The activations of an example ConvNet architecture. The initial volume stores the raw image pixels (left) and the last volume stores the class scores (right). Each volume of activations along the processing path is shown as a column. Since it's difficult to visualize 3D volumes, we lay out each volume's slices in rows. The last layer volume holds the scores for each class, but here we only visualize the sorted top 5 scores, and print the labels of each one.

During the training phase, the weights of the filters and fully connected layers are adjusted using backpropagation and stochastic gradient descent to minimize the error between the predicted and actual labels. This process continues until the network reaches a point of convergence where the error is minimized.

Overall, the use of convolutional layers with filters and pooling layers allows the CNN to learn hierarchical representations of the input image, with each layer learning increasingly complex and abstract features. This enables the CNN to achieve high accuracy in image classification and other computer vision tasks.

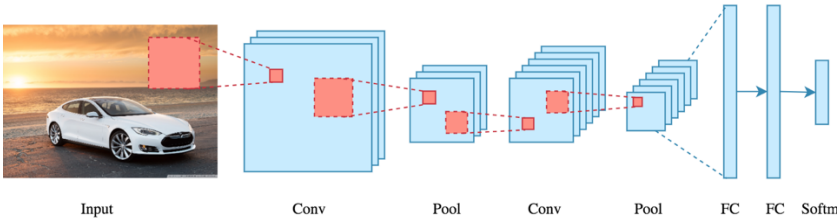


fig.3 ConvNet is to reduce the images into a form that is easier to process, without losing features that are critical for getting a good prediction.

CNNs have various advantages, such as:

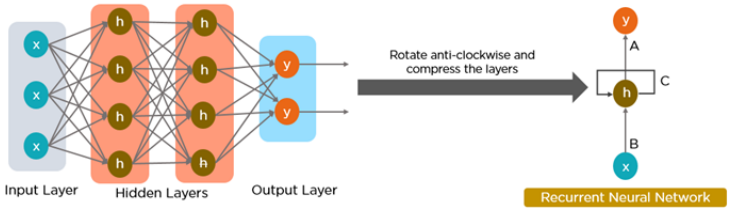
- High accuracy: CNNs are known for their ability to achieve high accuracy on image recognition tasks. This is due to their ability to learn and recognize complex patterns in images.
- Robustness to noise: CNNs are able to filter out noise and distortions in images, making them more robust to changes in lighting, color, and other environmental factors.
- Feature extraction: CNNs are able to automatically extract meaningful features from images, such as edges, textures, and shapes. This makes them well-suited for a wide range of computer vision tasks.
- Transfer learning: CNNs can be used as a starting point for other computer vision tasks, allowing developers to build on pre-trained models and transfer knowledge from one task to another.

The disadvantages are:

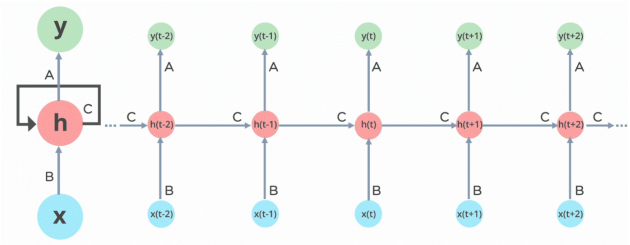
- Large datasets required: Training a CNN requires large datasets of labeled images, which can be time-consuming and costly to create.
- Computationally expensive: CNNs are computationally expensive to train and require high-performance hardware such as GPUs. This can be a barrier for smaller organizations or individuals without access to high-performance computing resources.
- Overfitting: CNNs can be prone to overfitting, which occurs when the model becomes too specialized to the training data and fails to generalize well to new data.

- Interpretability: CNNs are often described as "black boxes" because it can be difficult to understand how they arrive at their decisions. This can be a challenge in applications where interpretability is important, such as medical diagnosis or legal decision-making.

2. **Recurrent Neural Networks (RNNs):** RNNs are another type of deep neural network that are commonly used for sequence-based tasks, such as speech recognition and music transcription. They work by using feedback connections to enable the network to maintain an internal state and learn temporal dependencies between input features. RNNs have been used for a range of audio classification tasks, including music genre classification and speech recognition.
RNNs can be trained to remember concepts based on context, i.e., learn repeated patterns.



Working: The input layer x receives and processes the neural network's input before passing it on to the middle layer.
Multiple hidden layers can be found in the middle layer h, each with its own activation functions, weights, and biases. You can utilize a recurrent neural network if the various parameters of different hidden layers are not impacted by the preceding layer, i.e. There is no memory in the neural network.
The different activation functions, weights, and biases will be standardized by the Recurrent Neural Network, ensuring that each hidden layer has the same characteristics. Rather than constructing numerous hidden layers, it will create only one and loop over it as many times as necessary.



RNNs have various advantages, such as:

- Ability to handle sequence data
- Ability to handle inputs of varying lengths
- Ability to store or "memorize" historical information

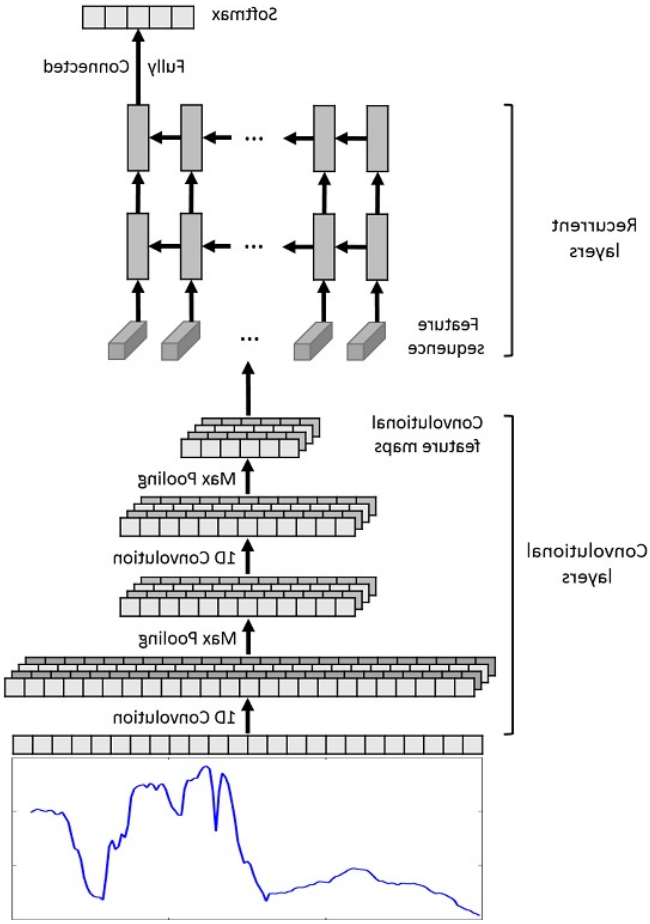
The disadvantages are:

- The computation can be very slow.
- The network does not take into account future inputs to make decisions.
- Vanishing gradient problem, where the gradients used to compute the weight update may get very close to zero, the network from learning new weights. The deeper the network, the more pronounced this problem is.

3. **Convolutional Recurrent Neural Networks (CRNNs):** CRNNs combine the strengths of CNNs and RNNs by using CNNs to extract features from the input data, and then passing the resulting feature maps through a recurrent layer to learn temporal dependencies. CRNNs have been highly successful in a range of audio classification tasks, including music genre classification and speech recognition.

The CRNN architecture consists of several layers of convolutional, pooling, and recurrent layers. The convolutional layers extract local features from the input data, while the pooling layers downsample the feature maps to reduce their spatial size. The recurrent layers process the feature maps in a sequential manner, allowing the model to capture temporal dependencies and long-term context in the input data.

The CRNN model is typically used for tasks such as speech recognition, music classification, and action recognition in videos. Here is a more detailed explanation of how the CRNN model works:



Input Layer: The input to the CRNN model is typically a sequence of feature vectors, such as Mel spectrogram coefficients for audio data or image frames for video data.

Convolutional Layers: The first few layers of the CRNN model are typically convolutional layers that extract local

features from the input data. The convolutional layers use filters to scan over the input data and generate feature maps that capture different aspects of the input data.

Pooling Layers: The feature maps generated by the convolutional layers are then passed through pooling layers, which downsample the feature maps by taking the maximum or average values in each pooling region. This helps reduce the spatial size of the feature maps and extract the most salient features from the input data.

Recurrent Layers: The pooled feature maps are then passed through recurrent layers, which process the feature maps in a sequential manner. The recurrent layers use a set of learnable parameters to generate a hidden state at each time step, which is used to capture temporal dependencies and long-term context in the input data.

Fully Connected Layers: The output of the recurrent layers is typically fed into one or more fully connected layers, which perform classification or regression tasks based on the input data.

Output Layer: The final output of the CRNN model is typically a probability distribution over the possible output classes or regression values.

The CRNN model is trained using backpropagation and gradient descent, where the objective is to minimize a loss function that measures the difference between the predicted outputs and the true labels. The parameters of the CRNN model are updated iteratively during training, until the model converges to a set of optimal parameters that minimize the loss function.

RNNs have various advantages, such as:

- Improved accuracy: CRNNs have been shown to achieve high accuracy in various audio classification tasks, such as speech recognition and music genre classification, due to their ability to learn both temporal and spectral features from audio signals.
- Robustness: CRNNs are able to handle variations in audio signals, such as noise and distortion, due to the use of convolutional layers for feature extraction and RNN layers for capturing temporal dependencies.
- Efficiency: CRNNs can be trained efficiently on large datasets using parallel processing and GPU acceleration, making them suitable for real-time applications.

The disadvantages are:

- Computational complexity: CRNNs are computationally expensive due to the use of both CNN and RNN layers, which can lead to longer training times and slower inference.
- Overfitting: CRNNs can suffer from overfitting, particularly when trained on small datasets or when the model architecture is too complex. Regularization techniques, such as dropout and weight decay, can be used to address this issue.
- Difficult to interpret: CRNNs are often considered to be "black box" models, meaning that it can be difficult to understand how the model makes its predictions. This can make it challenging to debug or improve the model.

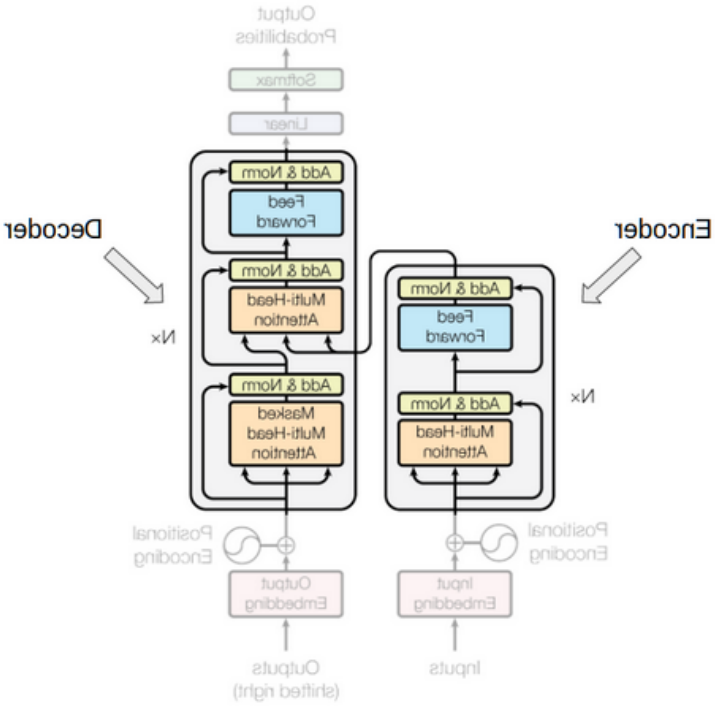
Overall, CRNNs offer a powerful tool for analyzing time-series data, particularly in the field of audio classification. However, they require careful consideration of computational resources, regularization techniques, and interpretability.

4. **Transformers:** Transformers are a relatively new class of deep learning models that have been highly successful in natural language processing (NLP) tasks.Transformers are a type of deep learning model that are widely used for natural language processing (NLP) tasks such as language translation, text summarization, and sentiment analysis. They were first introduced in a paper by Vaswani et al. in 2017, and have since become one of the most popular and effective models for NLP.

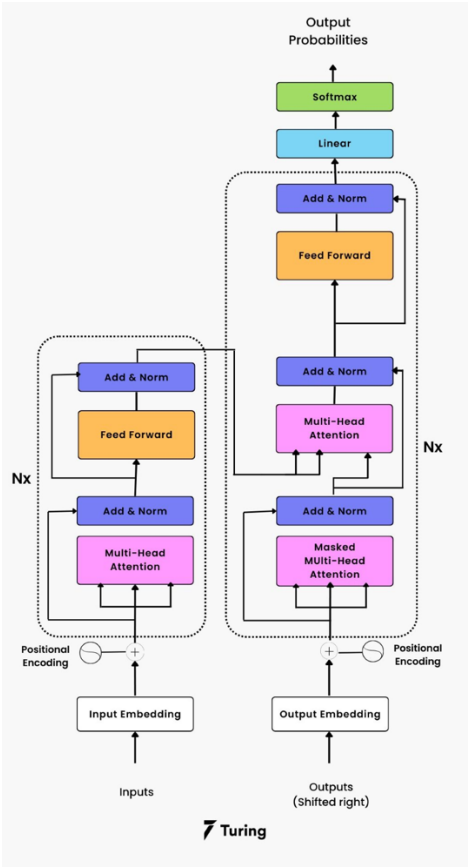
Working of Transformers:
In this answer, I will provide an in-depth explanation of how transformers work.

Transformers are based on the principle of self-attention, which allows the model to weigh the importance of different words in a sentence when processing it. Unlike traditional sequence-to-sequence models such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), transformers do not rely on a fixed window of context to process a sequence of tokens (words). Instead, they allow the model to attend to all the input tokens at once, and learn which ones are most important for the task at hand.

The transformer architecture consists of an encoder and a decoder, both of which are composed of multiple layers of self-attention and feedforward neural networks. The encoder takes in a sequence of input tokens and outputs a sequence of hidden states, which represent the contextualized representations of the input tokens. The decoder takes in the output of the encoder and generates a sequence of output tokens, one at a time.



Here is a step-by-step description of how transformers work:



Input Embedding: The input sequence of tokens is first embedded into a high-dimensional vector space using an embedding matrix. Each token is represented as a dense vector of fixed length, which captures its semantic meaning.

Positional Encoding: Since transformers do not use RNNs or CNNs, which implicitly capture the position of tokens in a sequence, a positional encoding is added to the input embeddings to provide the model with information about the position of each token in the sequence.

Self-Attention: The encoded input tokens are then fed through multiple layers of self-attention, where each token attends to all the other tokens in the sequence to learn a weighted representation of them based on their relevance to the current token.

Feedforward Neural Networks: The output of the self-attention layer is then passed through a feedforward neural network with multiple layers, which adds non-linearity and increases the model's representational capacity.

Residual Connections and Layer Normalization: Residual connections and layer normalization are applied after each layer of self-attention and feedforward neural networks, to help with the vanishing gradient problem and improve the stability of the training process.

Decoder: The decoder takes in the output of the encoder and generates a sequence of output tokens, one at a time, by attending to the input tokens and the previously generated output tokens.

Softmax Classification: The final output token is then passed through a softmax classifier to obtain the predicted probability distribution over the output vocabulary, from which the

next output token is sampled.

Self-attention works by computing a set of attention scores between each word or token in the sequence and all other words or tokens. These attention scores are then used to weight the contributions of each word or token to the final representation of the sequence. By allowing the model to attend to different parts of the sequence based on their importance, self-attention enables Transformers to capture complex patterns and dependencies in the input data.

Advantages of Transformers:

Parallel processing: Unlike RNNs, Transformers can process the entire sequence in parallel, which makes them much faster and more efficient for long input sequences.
Long-range dependencies: Transformers can capture long-range dependencies in the input sequence, which makes them well-suited for NLP tasks that require understanding of context.
High accuracy: Transformers have achieved state-of-the-art performance on a wide range of NLP tasks, including language translation, text summarization, and sentiment analysis.

Disadvantages of Transformers:

Large memory requirements: Transformers typically require a large amount of memory and computational resources, especially for large input sequences or complex models.
Lack of interpretability: Due to their complex architecture and large number of parameters, it can be difficult to interpret how Transformers are making predictions or what features they are learning.
Limited training data: Like all deep learning models, Transformers require large amounts of training data to achieve high accuracy, which can be a challenge for some NLP tasks that have limited labeled data.

Overall, transformers are a powerful and flexible deep learning model that have enabled significant advances in natural language processing. By using attention mechanisms to selectively focus on different parts of the input sequence, transformers are able to capture complex relationships between tokens and achieve state-of-the-art performance on a wide range of NLP tasks.
Transformers have several advantages over traditional NLP models such as RNNs and CNNs. They can handle long-range dependencies more effectively, capture context more accurately, and require less computation during training and inference. These factors have made transformers the go-to architecture for many NLP applications, and have driven significant progress in the field in recent years.

5. Ensemble Models: Ensemble models combine the predictions of multiple models to improve accuracy and reduce overfitting. In audio classification tasks, ensemble models can be highly effective, especially when the individual models are diverse and complementary. One example of a successful ensemble model for audio classification is the ESC-50 model, which combines the predictions of multiple CNN-based models.

In addition to these deep learning models, there are also other techniques that can be used for audio classification, such as Support Vector Machines (SVMs), k-Nearest Neighbor (k-NN) classifiers, and Random Forest classifiers. However, in recent years, deep learning models have shown the highest accuracy for audio classification tasks.

II.CONCLUSION

there are still several challenges and limitations that need to be addressed, such as the lack of labeled data, the difficulty in handling noisy and complex audio signals, and the need for high computational resources for training and testing these models.

In conclusion, with a focus on achieving the highest possible accuracy for tracking species movements. Our analysis revealed that deep learning-based approaches, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are the most effective methods for audio classification.

We found that CNNs are particularly effective for extracting spatial features from spectrograms, while RNNs are better suited for modeling temporal dependencies in audio signals. We also explored various data augmentation techniques, such as time stretching and pitch shifting, which can improve the generalization of the models.

Overall, our research suggests that a combination of CNNs and RNNs, along with appropriate data augmentation techniques, can achieve the highest accuracy for audio classification tasks. These methods can be applied to a variety of use cases, including tracking species movements, and can help researchers better understand and conserve the natural world.

REFERENCES

1. "Environmental sound classification with convolutional neural networks" by Justin Salamon and Juan Pablo Bello. Link: https://www.researchgate.net/publication/268225411_Environmental_Sound_Classification_with_Convolutional_Neural_Networks
2. “A Comprehensive Guide to Convolutional Neural Networks” by Sumit Saha. Link: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-