TOWARD INTERPRETABLE POLYPHONIC SOUND EVENT DETECTION WITH ATTENTION MAPS BASED ON LOCAL PROTOTYPES

Pablo Zinemanas[1], *Martín Rocamora*[2], *Eduardo Fonseca*[1], *Frederic Font*[1] *and Xavier Serra*[1]

[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain, name.surname@upf.edu
[2] Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay, rocamora@fing.edu.uy

ABSTRACT

Understanding the reasons behind the predictions of deep neural networks is a pressing concern as it can be critical in several application scenarios. In this work, we present a novel interpretable model for polyphonic sound event detection. It tackles one of the limitations of our previous work, i.e. the difficulty to deal with a multi-label setting properly. The proposed architecture incorporates a prototype layer and an attention mechanism. The network learns a set of local prototypes in the latent space representing a patch in the input representation. Besides, it learns attention maps for positioning the local prototypes and reconstructing the latent space. Then, the predictions are solely based on the attention maps. Thus, the explanations provided are the attention maps and the corresponding local prototypes. Moreover, one can reconstruct the prototypes to the audio domain for inspection. The obtained results in urban sound event detection are comparable to that of two opaque baselines but with fewer parameters while offering interpretability.

Index Terms— interpretability, sound event detection, prototypes

1. INTRODUCTION

After significant advances in computer vision, speech recognition, and natural language processing, deep learning models have also become the standard approach in environmental sound processing tasks, such as sound event detection, audio tagging, and acoustic scene classification [1, 2]. The increasing complexity of such models makes it difficult to explain the process that leads to its output in a way that humans can understand. This can be problematic in some real–world deployment scenarios. Therefore, research on interpretability and accountability of predictive models are steadily growing. In addition, interpretable models make it easier to debug, detect biases, and design defenses for adversarial attacks [3].

Instead of creating intrinsically interpretable deep neural networks, most existing works follow a *post hoc* approach, i.e., they try to explain the input-output behavior of a *black-box* model. For example, training a linear proxy model that imitates the behavior of the original model but is easier to interpret is a common approach [4]. However, since the proxy model is typically a local linear approximation of a non-linear model, it can fall short of providing a reliable explanation [5]. Other post hoc explanation methods focus on studying the representations of the input data learned by the network or highlighting the input characteristics that strongly influence the output. For instance, saliency maps are a typical example of this approach [6], where the gradient of the output with respect to the input is used to identify the most relevant portions of the input. However, this is incomplete as an explanation, as it provides no clue about how the relevant information is being used [5].

Rather than producing explanations of black-box models, some research seeks to develop inherently interpretable neural networks that provide faithful explanations to what the model actually computes [5]. By adding specific components, one can strive for rendering some form of interpretability while being as accurate as a black-box model. An example of this is the incorporation of attention mechanisms, which are network components that learn to select the part of the input that the rest of the model should focus on. Thus, besides improving predictive performance, the relative importance of the input units offers insights into the model's decision-making process. Yet, whether attention mechanisms can provide faithful explanations is a matter of current debate, as it depends on how they are implemented and the degree of interpretability pursued [7].

Learning through prototypes is another approach that can provide inherent interpretability to deep neural networks. Decision are based on a few relevant examples known as *prototypes* that serve as a distillation of the data and have a high interpretable value [8, 9]. A prototype is a vector that is close or identical to an instance of the training set. Deep neural networks can learn those prototypes in a flexible latent space. For example, the interpretable network proposed by Li et al. [10] for image classification is based on prototypes. The architecture appends a special prototype layer and uses an autoencoder. The prototypes are learned in the low-dimensional latent space produced by the encoder, and they can be reconstructed by applying the decoder. The predictions are based on the distance from the data instance to each prototype in the latent space. Thus, the explanations are the prototypes and the distances to them, which are the actual computations of the model to generate the output.

Our previous work extended this approach to audio classification [11]. There, we proposed the Audio Prototype Network (APNet) and showed compelling results when applied to speech, music, and environmental audio, for problems with a single class label per audio clip. However, in a polyphonic setting (i.e multi-label), an input instance corresponding to several classes should be simultaneously close to prototypes of those classes in the latent space. Unfortunately, learning such latent space proved challenging in practice, thus motivating the alternative approach proposed herein.

In this work, we propose a novel interpretable deep neural network for polyphonic sound event detection. To provide interpretability, we leverage the prototypes network approach and attention mechanisms. The network learns *local prototypes*, i.e. data points in the latent space representing a patch in the input representation. The approach is similar to that of [12] for single class image classification, which compares image parts to learned prototypes. However, we extend the scope to a multi-label setting with promising results in sound event detection. Besides, the proposed model learns attention maps used for positioning the local prototypes and reconstructing the latent space properly. Then, the detection is solely based on the attention maps. Thus, the explanations of the network are in the form of local prototypes and attention maps.

2. RELATED WORK

Some post hoc visualization methods have been applied in the audio domain. For instance, in [13] a convolutional–recurrent network trained for polyphonic sound event detection was evaluated using saliency maps. In [14], a gradient–based approach was proposed for visualizing heat maps in the first layer of an end–to–end convolutional neural network. Regarding proxy models, SLIME [15] is a variation of the LIME [4] algorithm for audio content analysis, which produces visual explanations in the form of temporal, frequency, and time-frequency segmentation. The model we propose herein also generates visual explanations of its predictions, but these are faithful to its computations instead of post hoc explanations.

Prototypical learning has been applied to audio problems but not necessarily looking for interpretability. For example, Pons et al. [16]—following [17]—used prototypical networks for audio classification with few data. However, their system is not intended to be interpretable, so one can not reconstruct the prototypes to the input space. In contrast, APNet and the model we propose herein allow for reconstructing the prototypes to the input space through the decoder and then mapping them to the audio domain.

The first models that used attention mechanisms in the audio domain applied them in conjunction with recurrent networks for speech recognition [18, 19]. Nowadays, attention mechanisms are widely used for speech, music, and other audio-related problems because of their ability to capture long-term temporal information. Self-attention mechanisms are used instead of recurrent layers to integrate temporal information; for instance, they were applied for music generation [20] and tagging [21]. Attention mechanisms can also be used for weighting the frequency dimension to create interpretable adaptive filter banks [22]. In contrast, our model does not use attention maps to weight input's features. Instead, we use them as the only information to classify sound sources. Furthermore, since we devise the attention maps for proper reconstruction from the local prototypes, they are interpretable by design.

Some other models combine attention mechanisms and prototypes. For instance, ProtoAttend [23] selects input-dependent prototypes based on a relational attention mechanism that connects the encoded representation and the prototype candidates. In this case, the prototypes are instances from the training data. However, other methods use mean vectors as prototypes for few-shot learning [24].

3. PROPOSED MODEL

Let $\boldsymbol{X}_i \in \mathbb{R}^{\mathcal{T}\times\mathcal{F}}$ be the i-th mel-spectrogram where $\mathcal{T}$ and $\mathcal{F}$ are the number of time frames and frequency bins, respectively. Therefore we define the training set as $\{(\boldsymbol{X}_i, \boldsymbol{Y}_i)\}_{i=1}^{N}$, where $\boldsymbol{Y}_i \in \mathbb{R}^K$ are the one-hot encoded labels, N is the number of instances and K is the number of classes. APNet is formed by two main components: an autoencoder and a classifier [11]. Our proposed model uses the same autoencoder from APNet, which is represented in the upper branch of Figure 1, and utilizes a novel classifier. The encoder is aimed at extracting meaningful features from the input: $\boldsymbol{Z}_i = f(\boldsymbol{X}_i)$, where $\boldsymbol{Z}_i$ is a tensor of shape (T, F, C) and represents the transformed input in the latent space. C is the number of channels of the encoder's last convolutional layer. The decoder part of the autoencoder is used for reconstructing the mel-spectrogram: $\widetilde{\boldsymbol{X}}_i = g(\boldsymbol{Z}_i) \in \mathbb{R}^{\mathcal{T}\times\mathcal{F}}$. Both the encoder and the decoder are formed by three convolutional layers with leaky ReLu activations. The encoder includes two max-pooling layers interspersed between the convolutions and the decoder applies the corresponding unpooling layers. Please refer to [11] for more details. The classifier of APNet is based on the distance from $\boldsymbol{Z}_i$ to a set of M prototypes with the same shape (T, F, C). Therefore, a prototype is a point in the latent space corresponding to the full mel-spectrogram representation in the input space. This makes it troublesome for APNet to represent a multi-label input instance as it should be close to prototypes from different classes.

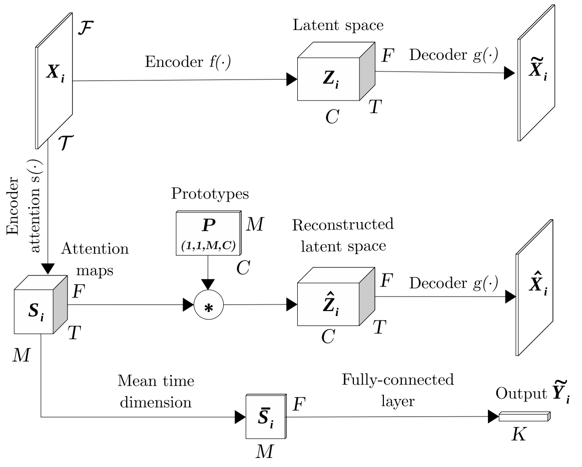

Figure 1: Diagram of the proposed model.

The model proposed in this work is devised to overcome this limitation, i.e. it is capable of detecting various simultaneous sound events. The middle and bottom branches in the diagram of Figure 1 show this novel classifier. We use another encoder, $s(\cdot)$, to extract M attention maps in the latent space: $\boldsymbol{S}_i = s(\boldsymbol{X}_i)$, where $\boldsymbol{S}_i$ is a tensor of shape (T, F, M). The encoder $s(\cdot)$ is similar to the autoencoder's one, $f(\cdot)$, but with ReLu activations to force a non-negative output. Each attention map is related to one prototype. Therefore the network learns a set of M prototypes of shape $(1, 1, C)$. We represent the M prototypes as a tensor $\boldsymbol{P}$ of shape $(1, 1, M, C)$. Note that each prototype represents one point in the time-frequency plane in the latent space. Therefore, in the input space these prototypes represent a patch of shape equal to the receptive field of the encoder network (32×32 in this work).

Using the attention maps and the learnable prototypes the model tries to reconstruct the latent representation $\boldsymbol{Z}_i$. This is done by multiplying each attention map by its corresponding prototype and then summing all maps. Note that this is equivalent to a 1×1 2D convolutional layer: $\widehat{\boldsymbol{Z}}_i = \boldsymbol{S}_i * \boldsymbol{P}$, or a dense layer $\widehat{\boldsymbol{Z}}_i = \boldsymbol{S}_i \cdot \boldsymbol{P}_s$, where $\boldsymbol{P}_s$ is the squeezed version of the tensor $\boldsymbol{P}$ with shape (M, C).

Therefore $\widehat{\boldsymbol{Z}}_i$ has the same shape of $\boldsymbol{Z}_i$ and aims to be a reconstruction of the latent space. In summary the attention maps represent the specific weight of each local prototype in each time-frequency point in order to have a good reconstruction of the latent space. Using the decoder $g(\cdot)$ from the top branch, we can project this reconstructed tensor into the input space, $\widehat{\boldsymbol{X}}_i = g(\widehat{\boldsymbol{Z}}_i) \in \mathbb{R}^{\mathcal{T}\times\mathcal{F}}$. In this way, we can visualize the reconstruction of the latent space in the input space to inspect it.

Finally, the bottom branch deals with the detection task, which is solely based on the attention maps. First, we average the time dimension of $\boldsymbol{S}_i$:

$$\bar{S}_i[f, m] = \frac{1}{T}\sum_{t=1}^{T} S_i[t, f, m] \tag{1}$$

where $\bar{\boldsymbol{S}}_i$ has shape (F, M) and integrates the attention map for each prototype and frequency bin in the latent space. Then, a dense layer connects a flattened version of $\bar{\boldsymbol{S}}_i$ with the classification output:

$$\widetilde{\boldsymbol{Y}}_i = \text{sigmoid}(\bar{\boldsymbol{S}}_i \cdot \boldsymbol{W}), \tag{2}$$

where $\boldsymbol{W} \in \mathbb{R}^{MF\times K}$ is the kernel of the layer and $\widetilde{\boldsymbol{Y}}_i = \{\widetilde{Y}_{ik}\} \in \mathbb{R}^{1\times K}$. We do not use bias in order to keep this layer more interpretable. We seek to audit how the model connects each prototype and each frequency bin to the corresponding output.

3.1. Objective function

We want the model to be able to detect sound events while maintaining the interpretability of the parameters and the explainability of the predictions. For this purpose, we define three losses to train the model. First we have a loss for learning the detection task. Since this is a multi-label problem, we use binary cross-entropy, $\mathcal{L}_c$. Then we define a mean squared error loss to have good reconstruction quality in the autoencoder of the top branch: $\mathcal{L}_r = \frac{1}{N}\sum_{i=1}^{N}\left\|\boldsymbol{X}_i - \widetilde{\boldsymbol{X}}_i\right\|_2^2$. This loss ensures that we can transform the data from the latent space back to the input space, in particular the learned prototypes.

Finally, we define a loss for enforcing a correct process of reconstruction using the attention maps and the prototypes. In other words, we let the network learn how to position the prototypes using the attention maps. In this sense, we define a mean squared error loss in both latent and input spaces:

$$\mathcal{L}_p = \frac{1}{N}\sum_{i=1}^{N}\left\|\boldsymbol{Z}_i - \widehat{\boldsymbol{Z}}_i\right\|_2^2 + \frac{1}{N}\sum_{i=1}^{N}\left\|\boldsymbol{X}_i - \widehat{\boldsymbol{X}}_i\right\|_2^2. \tag{3}$$

This loss ensures two assets of the model related to its interpretability. First, this loss establishes that the attention maps are learned to be an explicit explanation of how the model makes its predictions. Note that the attention maps are the only information used for the final prediction. And these maps are interpretable since they show how to position each prototype in the latent space in order to have a good reconstruction. Moreover this loss ensures that the prototypes are similar to the data and therefore we can transform them to the input space and audit them.

Besides, we use $l1$ regularization to force some sparsity in the attention map: $\mathcal{R}_s = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{Z}_i\|_1$. This is to prevent the network from reconstructing the latent space by mixing many prototypes. We also apply the same type of regularization to the kernel of the dense layer that connects the attention maps and the output: $\mathcal{R}_w = \|\boldsymbol{W}\|_1$. The idea is that the output for a given class is activated with only a few points on the attention map, both in the frequency and prototype dimension. Therefore we keep the explanations as simple as possible.

While training the proposed system, we optimize the weighted sum of all losses and regularization terms defined previously:

$$\mathcal{L} = \alpha\mathcal{L}_c + \beta\mathcal{L}_r + \gamma\mathcal{L}_p + \delta\mathcal{R}_s + \epsilon\mathcal{R}_w \tag{4}$$

where the weights $(\alpha, \beta, \gamma, \delta, \epsilon)$ are real-valued hyperparameters.

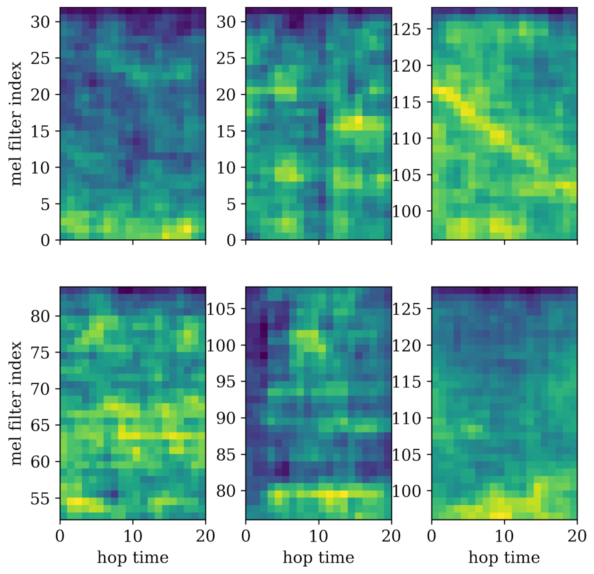

Figure 2: Reconstructed learned local prototypes. The y axis represents the mel bands where the prototypes were reconstructed.

4. EXPERIMENTS AND RESULTS

We train the proposed model by optimizing the objective function defined in Eq. (4). We use Adam optimizer with a learning rate of 0.001 for 50 epochs and we select the model with the top performance in the validation set. We use the following set of hyperparameters $(10, 5, 5, 10^{-5}, 10^{-6})$ and a batch size of 256. The experiments are conducted using the DCASE-models library [25] and the code is available under an open-source license[1].

We compare the performance of the proposed model to that of two different opaque baselines: (1) a convolutional neural network (CNN) formed by three convolution layers and two dense layers [26]; and (2) a multi-layer perceptron (MLP) whose input is the embedding vector extracted from the pre-trained Openl3 model [27]. We optimize a binary cross-entropy loss with the same optimizer and strategy for these baselines as for the proposed model.

We train and evaluate the proposed model and the baselines on the URBAN-SED dataset v2.0 [26]. This is formed by 10-second length audio files corresponding to synthetic mixtures of sound sources obtained from the UrbanSound8k dataset. Each sound event is tagged with one of the following classes: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music. The three models use log-scaled mel-spectrogram as input representation, but with different parameters. Both the CNN and the proposed model uses 128 mel bands and a sampling rate of 22050 Hz. The proposed model uses a window size of 4096 and hop size of 1024 for calculating the spectrograms. On the other hand, CNN uses a window size of 512 and hop size of the same length. Openl3 has predefined parameters [27].

To evaluate the models, we use F-measure ($F1$) and error rate (ER) in a 1-second grid as commonly used for sound event detection [28]. We run the training 10 times and calculate the mean and standard deviation of both metrics. Table 1 shows the performance comparison of the three models along with their number of

[1]`https://github.com/pzinemanas/attprotos`

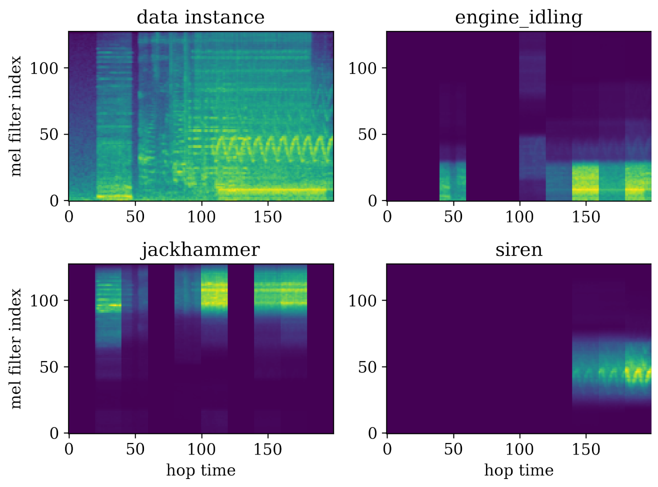

Figure 3: Example of an input instance from the test set masked by the attention maps. At the top left plot we show the mel-spectrogram, which includes sound events of six different classes. The other plots are the same mel-spectrogram but masked by each of the reconstructed attention maps for the corresponding classes.

parameters. Note that the performance of the proposed model is comparable to that of the baselines, but with fewer parameters.

Table 1: Performance comparison of the proposed model and the two baselines. The performance metrics are the F-measure ($F1$) and the error rate (ER). The number (#) of parameters in millions (M) are also included in the comparison.

Network	$\boldsymbol{F1}$ **(%)**	$\boldsymbol{ER}$	**# Params. (M)**
CNN	57.3 ± 0.6	0.568 ± 0.006	0.5
Openl3+MLP	58.2 ± 0.3	0.558 ± 0.004	9.5
Ours	58.8 ± 0.9	0.572 ± 0.007	0.15

4.1. Prototypes

The reconstruction of the latent space helps the network to learn prototypes similar to patches from the training data. We use the decoder part of the autoencoder, $g(\cdot)$, to reconstruct the learned prototypes in the input space. We follow a process similar to that performed in APNet for this purpose [11]. But in this case, we have to extend the prototypes tensor $\boldsymbol{P}$ to have the same shape of the latent space, i.e. (M, T, F, C). To this end, we create a zero tensor of this shape and select a point in the time-frequency plane where to position each prototype. The time is selected arbitrarily at the center, and the frequency is selected by minimizing the distance of each prototype to the data instances. By doing this, we reconstruct the patches in the frequency bands where the closest data instances have these prototypes present. Figure 2 shows a set of selected prototypes. Note that the network learns different types of shapes and textures related to environmental sounds present in the data set.

4.2. Attention maps

For each data instance it is possible to extract the corresponding attention maps to provide and explanation on how the model makes its predictions. For a given class k, we follow the following process:

1. Mask the prediction $\widetilde{\boldsymbol{Y}}_i \in \mathbb{R}^{1\times K}$ by a unit vector of the same shape whose k-th component is the only one equal to 1: $\widetilde{\boldsymbol{Y}}_i^{(k)} = \widetilde{\boldsymbol{Y}}_i \odot \boldsymbol{1}_k$
2. Get the points of the previous layer that are more connected to the output k by calculating the gradient: $\nabla\bar{\boldsymbol{S}}_i^{(k)} = \widetilde{\boldsymbol{Y}}_i^{(k)} \cdot \boldsymbol{W}^T \in \mathbb{R}^{1\times FM}$
3. Reshape the gradient to (F, M), apply a half-wave rectifier to keep only positive connections and multiply it by the time-averaged attention maps: $\bar{\boldsymbol{S}}_i^{(k)} = \text{ReLu}\left(\nabla\bar{\boldsymbol{S}}_i^{(k)}\right) \odot \bar{\boldsymbol{S}}_i$. This represents the attention maps masked by the most important connections to the output k.
4. Find the most connected prototype by maximizing the energy of the masked attention map: $\widehat{m} = \arg\max_{m\in[1,\ldots,M]} \sum_{f=1}^{F} \left(\bar{S}_i^{(k)}[f, m]\right)^2$
5. Extract the frequency-dependent attention function: $\mathcal{S}_i^{(k)}[f] = \bar{S}_i^{(k)}[f, \widehat{m}]$
6. Convert the attention function to the input space. To do this, we first upsample the sequence by a rate of 4 to emulate the two max-pooling operations. Then we apply a moving-average filter to emulate the receptive field. Thus, the length of the filter is equal to the receptive field (32).

Figure 3 shows an example of the attention maps for three different classes. We multiply the attention maps in the input space by the mel-spectrograms, similarly to how the model does in the latent space. Note that the model can detect simultaneous sound events whose energy is concentrated in different frequency bands. Since the attention maps are designed to reconstruct the latent space and are the only information used for classification, these represent the inherent explanation of how the network makes its predictions.

5. CONCLUSION

In this work, we present a novel interpretable model for polyphonic sound event detection. Its predictions are based on attention maps learned for reconstructing the latent space by positioning a set of local prototypes. The network also learns the local prototypes as data points in the latent space representing a patch in the input representation. The attention maps provide a form of explanation that is faithful to the model computations and can give valuable insights into its decision process. Moreover, the prototypes can be reconstructed and thus can be listened to and audited.

The proposed model achieves encouraging results in urban sound event detection for a data set of synthetic mixtures, which are comparable to that from two opaque baselines but with fewer parameters, while at the same time offering interpretability. This is consistent with some previous work that claims that it is often possible to incorporate interpretability into deep learning models to tackle complex tasks without sacrificing performance [10, 12, 5].

Future work includes ablation studies to understand better the impact of the proposed losses and regularization terms in the final model. In addition, more experiments are needed to evaluate the effect of some hyperparameter values, such as the loss weights and the number of prototypes. Besides, we should evaluate the model with datasets recorded in natural conditions. Finally, we seek further development of interpretable models to analyze environmental sounds, including those that learn disentangled representations.

6. REFERENCES

[1] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic Scene Classification in DCASE 2020 Challenge: Generalization Across Devices and Low Complexity Solutions," in *Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, Nov. 2020, pp. 56–60.

[2] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and evaluation of sound event localization and detection in DCASE 2019," *Trans. on Audio, Speech, and Language Processing*, vol. 29, pp. 684–698, 2021.

[3] C. Molnar, *Interpretable Machine Learning*, 2019, https://christophm.github.io/interpretable-ml-book/.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the predictions of any classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, New York, NY, USA, 2016, p. 1135–1144.

[5] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, p. 206–215, May 2019.

[6] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *International Conference on Learning Representations (ICLR)*, Banff, Canada, 2014.

[7] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," in *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 11–20.

[8] J. Bien and R. Tibshirani, "Prototype selection for interpretable classification," *The Annals of Applied Statistics*, vol. 5, no. 4, Dec 2011.

[9] B. Kim, C. Rudin, and J. A. Shah, "The bayesian case model: A generative approach for case-based reasoning and prototype classification," in *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[10] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," in *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, vol. 32, 2018.

[11] P. Zinemanas, M. Rocamora, M. Miron, F. Font, and X. Serra, "An interpretable deep learning model for automatic sound classification," *Electronics*, vol. 10, no. 7, 2021.

[12] C. Chen, O. Li, D. Tao, A. J. Barnett, J. Su, and C. Rudin, "This looks like that: Deep learning for interpretable image recognition," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[13] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *Trans. on Audio, Speech and Language Processing*, vol. 25, no. 6, pp. 1291–1303, Jun. 2017.

[14] H. Muckenhirn, V. Abrol, M. Magimai-Doss, and S. Marcel, "Understanding and Visualizing Raw Waveform-Based CNNs," in *Interspeech 2019*, 2019, pp. 2345–2349.

[15] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis." in *18th International Society for Music Information Retrieval Conference*, Suzhou, China, 2020, pp. 537–543.

[16] J. Pons, J. Serrà, and X. Serra, "Training neural audio classifiers with few data," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 16–20.

[17] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results," *arXiv e-prints*, p. arXiv:1412.1602, Dec. 2014.

[19] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4960–4964.

[20] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. M. Shazeer, A. M. Dai, M. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *ICLR*, 2019.

[21] M. Won, S. Chun, and X. Serra, "Toward Interpretable Music Tagging with Self-Attention," *arXiv e-prints*, p. arXiv:1906.04972, Jun. 2019.

[22] P. Agrawal and S. Ganapathy, "Interpretable Filter Learning Using Soft Self-attention For Raw Waveform Speech Recognition," *arXiv e-prints*, p. arXiv:2001.07067, Jan. 2020.

[23] S. O. Arik and T. Pfister, "Protoattend: Attention-based prototypical learning," *Journal of Machine Learning Research*, vol. 21, no. 210, pp. 1–35, 2020.

[24] S. Sun, Q. Sun, K. Zhou, and T. Lv, "Hierarchical attention prototypical networks for few-shot text classification," in *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 476–485.

[25] P. Zinemanas, I. Hounie, P. Cancela, F. Font, M. Rocamora, and X. Serra, "DCASE-models: a Python library for computational environmental sound analysis using deep-learning models," in *Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, Nov. 2020, pp. 240–244.

[26] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello., "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, Oct. 2017.

[27] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen and learn more: Design choices for deep audio embeddings," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, May 2019, p. 3852–3856.

[28] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.