ON THE VERACITY OF LOCAL, MODEL-AGNOSTIC EXPLANATIONS IN AUDIO CLASSIFICATION: TARGETED INVESTIGATIONS WITH ADVERSARIAL EXAMPLES

Verena Praher[1*] **Katharina Prinz**[1*] **Arthur Flexer**[1] **Gerhard Widmer**[1,2]
[1] Johannes Kepler University Linz, Austria
[2] LIT AI Lab, Linz Institute of Technology, Austria
{verena.haunschmid,katharina.prinz}@jku.at

ABSTRACT

Local explanation methods such as LIME have become popular in MIR as tools for generating post-hoc, model-agnostic explanations of a model's classification decisions. The basic idea is to identify a small set of human-understandable features of the classified example that are most influential on the classifier's prediction. These are then presented as an explanation. Evaluation of such explanations in publications often resorts to accepting what matches the expectation of a human without actually being able to verify if what the explanation shows is what really caused the model's prediction. This paper reports on targeted investigations where we try to get more insight into the actual veracity of LIME's explanations in an audio classification task. We deliberately design adversarial examples for the classifier, in a way that gives us knowledge about which parts of the input are potentially responsible for the model's (wrong) prediction. Asking LIME to explain the predictions for these adversaries permits us to study whether local explanations do indeed detect these regions of interest. We also look at whether LIME is more successful in finding perturbations that are more prominent and easily noticeable for a human. Our results suggest that LIME does not necessarily manage to identify the most relevant input features and hence it remains unclear whether explanations are useful or even misleading.

1. INTRODUCTION

With the rise of deep learning methods used in Music Information Retrieval (MIR), also the desire for explaining the decisions made by such models has increased. A plethora of explanation methods ("explainers") have been originally developed for text or image data and adapted to the audio domain [1, 2], or specifically introduced for MIR systems [3]. Most notably, different versions of Local Interpretable Model-agnostic Explanations (LIME), a post-hoc explainer [4], have been used to explain models in a variety of MIR tasks [5–11].

Evaluation of explanations is known to be a hard task, due to a lack of agreement on what constitutes a "good explanation" [12] and, consequently, what may serve as ground truth for measuring the quality of an explanation. Explanations are often evaluated visually and the applied "metric" is whether the highlighted input parts appear relevant to the human observer ("confirmation bias") [13].

We propose evaluating local explanations by exploiting a known weakness of deep neural networks: adversarial examples. By adversarially perturbing examples in a way that a system's original prediction is changed, we know exactly what in the input caused the erroneous (*adversarial*) prediction and can use this as the "ground truth" that the explanation method should recover.

We are not the first to investigate the relationship between adversarial examples and interpretability. Slack et al. [14] show that a racially biased model can be attacked in a way that the (biased) prediction remains unchanged but the explanation appears as if the model did not base the prediction on sensitive attributes. This work is related in the sense that it highlights weaknesses of post-hoc explanation methods (including LIME).

Closest to our work are Göpfert et al. [15], who use "localised" adversarial attacks that target only selected segments of an image, and investigate the ability of different explainers (including LIME) to recover the affected part. In their experiments LIME outperforms the other explainers, which supports our choice of LIME as an explainer that deserves a more careful and critical investigation.

We perform four experiments with purposefully designed adversarial examples in order to obtain more insight about strengths and weaknesses of LIME-based explanations in an audio classification task. As a test bed, we have chosen a DNN-based singing voice detection model [16] that currently defines the state of the art on that task, and is suited to study the quality of explanations for several reasons: it addresses a clearly defined task (as opposed to genre classification for example), it has been used for demonstrating explanations before [5, 11], and it has the interesting previously documented property that it can be confused by directly drawing on the spectrogram [17].

[*] Equal contribution.

Attribution: V. Praher, K. Prinz, A. Flexer, and G. Widmer, "On the Veracity of Local, Model-agnostic Explanations in Audio Classification: Targeted Investigations with Adversarial Examples", in *Proc. of the 22nd Int. Society for Music Information Retrieval Conf.*, Online, 2021.

2. LOCAL INTERPRETABLE MODEL-AGNOSTIC EXPLANATIONS

LIME is an algorithm for explaining individual predictions (*local*) for any black box machine learning model (it is *model-agnostic*). The general idea of the algorithm is to train a simpler model g that approximates the neighborhood of the prediction $f(x)$ that we want to explain. The first step is to derive an interpretable representation depending on the input domain, e.g. the presence or absence of super pixels [1] in the image domain.

Let $x \in \mathbb{R}^d$ (in the case of spectrograms, $d = T \times F$) be the representation in the original input domain and $x' \in \{0,1\}^{d'}$ be the interpretable representation where 0 and 1 denote the absence and presence of an interpretable feature, respectively. In the next step we sample N_s instances around x' by randomly generating vectors of length d' containing 0's and 1's. Each generated instance z' is mapped back to the input domain and fed through the original model f. $z \in \mathbb{R}^d$ looks like the original input example with all parts that are 0 in z' occluded (e.g. gray values in the image domain). The N_s instances z' and the corresponding predictions $f(z)$ constitute the input to the explanation model g [4]. The most commonly used explainers are linear models of the form [2]

$$g(z') = b + w_g z'. \tag{1}$$

When training the linear explanation model, the generated instances are weighted via an exponential kernel decaying with increasing distance to the input example. This distance (and based on it the weight) is computed between the binary representation of the examples being explained (all 1's) and each of the instances in the neighborhood [3] .

A requirement for an explanation is that it be *locally faithful*: we expect the explainer g to approximate f in the neighborhood of x. The faithfulness is measured by the fidelity score, which is computed by the coefficient of determination (R^2) of the linear model's predictions [4], and ranges from 0 to a perfect score of 1, with the exact interpretation of the range of values being somewhat unclear.

Different flavours of LIME have been used for explaining predictions in a variety of audio classification and tagging tasks. The general LIME algorithm was kept the same and the difference is the type of interpretable features. Mishra et al. [5] proposed the first adaption of LIME for MIR, termed SoundLIME (SLIME), which segments the spectrogram into time, frequency, or time-frequency segments. SLIME was demonstrated on the task of singing voice detection [18] and used for analysing a replay spoofing detection system [6]. Haunschmid et al. used other types of interpretable features (super pixels [7], source separation estimates [8, 9]) for explaining the predictions of a variety of models, including music taggers [8, 9] and a content-based music recommender system [10]. Mishra et al. [11] proposed different content types for replacing the "grayed out" segments (e.g. *zero*, the *min* and the *mean* value of the spectrogram) and found the mean value to work best in their setting. For our experiments we will be using time-frequency segments [5, 11]. The hyperparameters for the LIME algorithm are described in Section 4.4.

3. ADVERSARIAL ATTACKS

In this section, we briefly describe a way to compute adversarial examples for a singing voice detection system. The goal is to obtain perturbations which do not affect how a human perceives a particular example, yet simultaneously change the prediction of the system, i.e., transform "Singing Voice" to "No Singing Voice" or vice versa. To realise this, we use an adversarial attack that was originally proposed for audio [19], and was previously applied in MIR to attack an instrument classification system [20]. The attack, subsequently denoted by Carlini & Wagner (C&W), assumes a white-box scenario, i.e. full knowledge about a model and its parameters.

As C&W is an iterative targeted attack, we use it to decrease the loss of a system with respect to a new (wrong) target prediction, in multiple iterations. Following the notation in [20], let f be a system and $x \in \mathbb{R}^d$ its input, i.e., a specific prediction is denoted by $f(x)$. Also, let $\delta_{ep} \in \mathbb{R}^d$ be an adversarial perturbation at iteration ep, and an adversarial example $x + \delta_{ep}$. Furthermore, we denote the system loss by L_{sys}, and the target prediction by t. To restrict the adversarial perturbation, C&W minimises a weighted sum of the squared L2-norm of δ and a system loss, resulting in an optimisation objective as follows [19, 20]:

$$\begin{aligned} L_{\text{total}} &= \|\delta_{ep}\|_2^2 + \alpha * L_{\text{sys}}(f(x+\delta_{ep}), t), \\ \delta_{ep+1} &= \text{clip}_\epsilon(\delta_{ep} - \eta * \text{sign}(\nabla_{\delta_{ep}} L_{\text{total}})). \end{aligned} \tag{2}$$

Note that $\nabla_{\delta_{ep}}$ is the gradient w.r.t. to δ_{ep}, and updates are performed based on the sign of the gradient and the multiplicative factor η. By applying clip_ϵ, an adversarial perturbation is always in the range of $[-\epsilon, \epsilon]$, and α is used to balance the magnitude of the perturbation and the system loss w.r.t. the target prediction.

In the following experiments, we will compute adversarial examples both for raw audio waveforms and for input spectrograms. In either case, we will always analyse the resulting perturbations in the time-frequency domain. For adversaries computed for the waveform we compute the time-frequency representation and subtract the original example to obtain the adversarial perturbation in the time-frequency domain. For adversarial examples computed for the spectrogram directly, time-frequency information is already available.

[1] (perceptually) grouped pixels

[2] In the original paper [4], the explainer had the form $g(z') = w_g z'$, mistakenly omitting the intercept b. Inspecting the source code shows that the intercept was trained as well: `https://github.com/marcotcr/lime/`

[3] The original paper [4] stated that the distance for weighting the examples was computed between the images directly, but the source code shows that it is actually computed between the binary representations, which was later confirmed by the author.

4. EXPERIMENTAL SETUP

This section describes the experimental setup, including the data we use, details about the singing voice detection system and hyperparameters for both the adversarial attacks and computing the explanations.

4.1 Data

To train the singing voice detection system proposed by Schlüter and Lehner, we use a subset of the data used in their work [16], namely the openly available *Jamendo* dataset [21]. The dataset consists of 93 songs totalling around 6 hours of music, with a proposed training / validation / test split of 61 / 16 / 16 files respectively. Each audio file has a sampling rate of 44.1kHz.

The annotations for the Jamendo dataset are provided with sub-second granularity [18], and indicate the presence / absence of singing voice. The proportion of singing voice ("sing") in comparison to non-singing voice ("no sing") is close to $50 : 50$ in all three data splits.

4.2 Singing Voice Detector

For subsequent experiments, we adapt the singing voice detection system introduced by Schlüter and Lehner [16]. We use the proposed architecture of the Convolutional Neural Network (CNN) and deploy the training routine with unaltered hyperparameters on the Jamendo dataset. Prior to training, the data is resampled to 22.05kHz and then used to compute magnitude Mel spectrograms (frame length 1024, hop size 315 samples and 80 Mel bands) [16]. The Mel spectrograms are logarithmically scaled and normalised per frequency band to have zero mean and unit variance over the training data.

The CNN is trained on spectrogram excerpts with a length of 115 frames, for which the target prediction corresponds to the ground-truth annotation for the central frame [16]. During training, we use a mini-batch size of 32 and Adam to optimise the Binary Cross Entropy loss for 40.000 update steps. We start with a learning rate of 0.001 and scale it by 0.85 every epoch. To support generalisation, dropout and data augmentation [18] is used; for a more detailed description of the training procedure, hyperparameters and the CNN architecture, we refer to [16].

The final classification error of the singing voice detector (in percent) on the Jamendo test data, given as the mean ± standard deviation over 5 different runs, is 11.54 ± 0.96. Furthermore, recall and specificity over 5 runs are 89.61 ± 1.71 and 87.46 ± 1.00 respectively. The metrics are computed based on binary predictions of the network, which are obtained after applying a median filter and a threshold tuned on validation data (cf. [16]). Note that exact reproduction of previously reported errors, namely 8.0 in [18] (with a slightly different architecture) and 5.5 in [16], is difficult as Schlüter and Lehner train on a larger (in-house) dataset, whereas we only use publicly available data. Additionally, non-determinism (e.g., via data augmentation or initial weights) can influence the performance of a model noticeably.

In the following experiments, let the singing voice detection system be f, and $f(x)$ the numeric model output for class "sing". The final classification is made by checking whether $f(x)$ is above ("sing") or below ("no sing") a threshold that is optimised on the validation set, and is equal to 0.51 for our system.

4.3 Hyperparameters for the Adversarial Attack

Equation (2) shows the hyperparameters we need to determine before attacking the singing voice detector. The maximum number of iterations in which we try to find adversarial perturbations that change the prediction of an excerpt is set to 1000 in our experiments. Other hyperparameters – clipping factor ϵ, update factor η and weight factor α – are tuned on the validation set of Jamendo, and chosen as the setting that results in the highest number of successful adversarial perturbations, i.e., changed the most predictions out of all audio excerpts in the validation data. For the attack on raw audio we use $\epsilon = 0.01$, $\eta = 0.0003$ and $\alpha = 2$; for the attack on the spectrograms we take $\epsilon = 0.1$, $\eta = 0.0005$ and $\alpha = 15$. Due to the binary nature of the singing voice detection task, the target t for each excerpt is chosen to be the opposite of its original prediction. Note that we find successful perturbations for 28.4% of all excerpts for attacks on the spectrogram, and for 60.5% of excerpts for raw audio.

4.4 Hyperparameters for the Explanations

As mentioned above the LIME algorithm has a set of hyperparameters that have to be chosen. We segment the spectrogram (80 frequency bins (F) $\times 115$ time frames (T)) into 5 time and 4 frequency segments, respectively, resulting in 20 "human-interpretable" features of size 20 frequency bins × 23 time frames. Repeating the calculation of an explanation might lead to different results due to the randomness in the neighborhood generation when the number of generated instances is too small. A preliminary experiment as described in [5, 11] on a subset of the explanations, suggests that $2^{13} = 8192$ instances are sufficient to generate stable explanations, which is why we set $N_s = 8192$ for all our experiments. We also investigated different ways of replacing explanation segments [11] (*zero*, *min*, and *mean*), but as the results appeared similar, we will focus on reporting the results for the *mean* content type. For weighting the instances to train the explainer we use the cosine distance function and an exponential kernel with a kernel width of 0.25.

5. EVALUATION OF EXPLANATIONS

To investigate to what extent local explanation methods can find the underlying reason for a particular classification by a system, we perform four experiments. In the first, we demonstrate how LIME can help detecting rather obvious causes for a prediction that are due to manually altered spectrograms. The goal of the remaining experiments is to quantitatively evaluate the ability of LIME to detect adversarial perturbations: in the second experiment we try to evaluate whether the segments that LIME highlights actually caused a prediction; in the third, we analyse if the explanations can detect the correct location of the adversarial perturbation if it only affects few segments of the input spectrogram; and in the fourth, we compare the fidelity scores for correct and incorrect explanations.

Due to the computation time needed for obtaining explanations, we subsequently limit the number of analysed excerpts for each song in the test set. For the first experiment, we manually alter a single random excerpt per song with an original classification of “no sing”. In the setting of the second, third and fourth experiment, for each of the 16 test songs, we randomly select 10 adversarial excerpts which were originally classified as “sing” and another 10 excerpts originally classified as “no sing”. We conduct all experiments with adversarial attacks computed both on raw audio and on spectrograms, as we want to account for the fact that attacks on raw audio might use implicit constraints that are difficult to detect for LIME (which operates on spectrograms).

To allow reproducibility, we publish code[4] for all subsequent experiments.

5.1 Explaining Manually Altered Spectrograms

In our first experiment, we exploit the fact that the predecessor of the singing voice detector we use (cf. [18]) could easily be fooled by drawing on the input spectrogram[5] [17]. After choosing a set of excerpts originally classified as “no sing”, we adapt drawings on the respective spectrograms until the prediction is changed to “sing”. Then we ask LIME to explain these new predictions. Figure 1 shows that the explainer can correctly identify the manually altered parts of the spectrogram as the cause for the wrong prediction. Examples like these, supported by relatively high fidelity scores, might make us think that LIME knows what is going on and make us trust these explanations. We could stop our evaluation here and put trust in our model because the explanations highlight seemingly correct behaviour [5], look meaningful [7], or match the expected behavior [9], but we will continue with a more careful investigation.

5.2 Explaining Predictions on Adversarial Examples

In this set of experiments, we first compute explanations for the predictions obtained for each of the selected adversarial excerpts. Every explanation consists of a list of interpretable features (time-frequency segments) and their corresponding weights. The weight of a particular feature should indicate the importance of the feature for making a certain prediction [4, 5]. To evaluate whether the explanation actually selects the features that are responsible for the (now wrong) prediction, we could naively check if the segments selected by LIME correspond to regions of the adversarial perturbation that have the highest magnitudes (defined via the L2 norm).

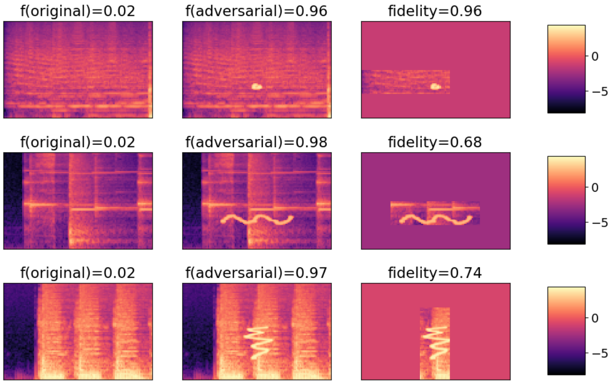


Figure 1: Different examples of spectrograms for which the model’s original prediction is “no sing” (left column) but drawing certain symbols (middle) leads to a change to “sing” with a model output f. The rightmost column shows the top 3 interpretable features (which may be directly adjacent and thus look like one segment) picked by the LIME algorithm, and their fidelity.

	Waveform	Spectrogram
“no sing” → “sing”	53 %	41 %
“sing” → “no sing”	21 %	32 %

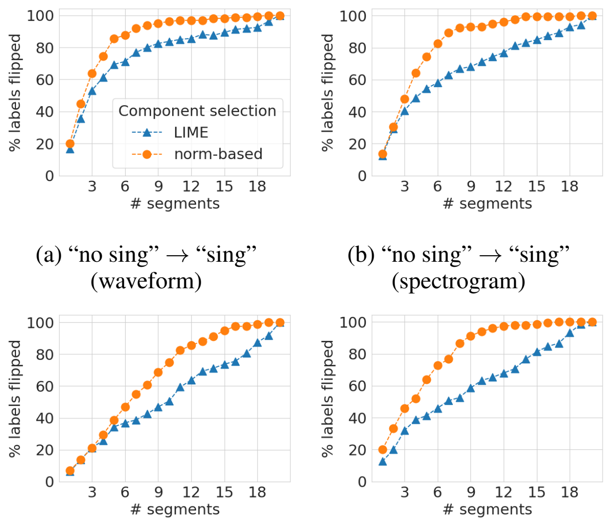
Table 1: Percentage of label flips when using only $k = 3$ segments of an adversarial perturbation, based on the most relevant features chosen by LIME. Columns denote whether perturbations were computed for the waveform / spectrogram, rows show original → target prediction.

However, as we do not know which part of a perturbation really led to a misclassification, i.e., whether parts of a perturbation with the highest magnitude have the most influence or not, this could lead to false conclusions.

To circumvent this, we follow another approach: we split the adversarial perturbation into the same time-frequency segments that LIME is using as interpretable features; we then selectively add those k segments of the perturbation to the input spectrogram that coincide with the features identified as an explanation. If the selected features are actually explaining the prediction, we expect the partial perturbation to be sufficient to achieve the same change of classification, or what we call *label flip*, as the full perturbation did. However, using $k = 3$ (which seems a reasonable number of segments to present to a user [5]) flips only $21 - 53\%$ of the predictions (see Table 1).

We therefore next look at whether larger numbers k of segments are able to flip more of the predictions. We use either LIME for selection of segments or choose the k perturbation segments with the highest magnitudes. We do this for $k \in [1, ..., 20]$ and report the results in Figure 2. For almost all k and across different settings we are more often successful in flipping the label when segment selection is based on segment magnitude rather than on LIME results. For LIME-based segments it is necessary to choose almost all 20 available segments to achieve flip rates close to 100%. This result suggests that it is indeed possible to

[4] https://github.com/CPJKU/veracity
[5] https://github.com/f0k/singing_horse



(a) "no sing" → "sing" (waveform)

(b) "no sing" → "sing" (spectrogram)

(c) "sing" → "no sing" (waveform)

(d) "sing" → "no sing" (spectrogram)

Figure 2: Percentage of classifications we can flip (y-axis) by adding only k segments (x-axis) of an adversarial perturbation to an input. Choosing which k segments are used is either based on LIME explanations (triangles), or on the norm of a perturbation (circles).

	Waveform		Spectrogram	
	LIME	norm	LIME	norm
"no sing" → "sing"	86 %	99 %	91 %	98 %
"sing" → "no sing"	49 %	74 %	44 %	55 %

Table 2: Percentage of label flips when adding all perturbation segments that contributed positively according to LIME, compared to using the same number of segments k selected based on the norm.

explain predictions with larger numbers of k interpretable features, but only when using the norm of a perturbation (which in a real application we would not know) to select the features and not when using LIME for the selection.

Finally, we analyse whether taking all segments that receive a positive weight from LIME can flip a prediction, as positive weights should correlate positively with the explained class [5]. For this analysis, we no longer add a fixed number of segments of a perturbation to each excerpt, but instead select all k segments of a perturbation for which the corresponding interpretable feature received a positive weight. This number of segments k can be different for each excerpt we look at. Additionally, we choose the same number of k segments based on the largest magnitude, and compare how often this results in label flips. The results of this experiment are summarised in Table 2. Again, we observe that taking partial perturbations is more successful when selecting the segments with the highest magnitude, as opposed to the segments most relevant for LIME.

These results suggest that LIME does not pick the input features that are most relevant for making a particular prediction.

5.3 Explaining "Localised" Perturbations

Considering the results in the previous section, one might argue that an adversarial perturbation can be present in the whole spectrogram, which could make it hard for LIME to decide which segments are most relevant. We make use of the finding that it is often sufficient to add only a small number of selected segments of the perturbation to flip a label, and conduct an additional experiment where we analyse how often LIME detects those segments.

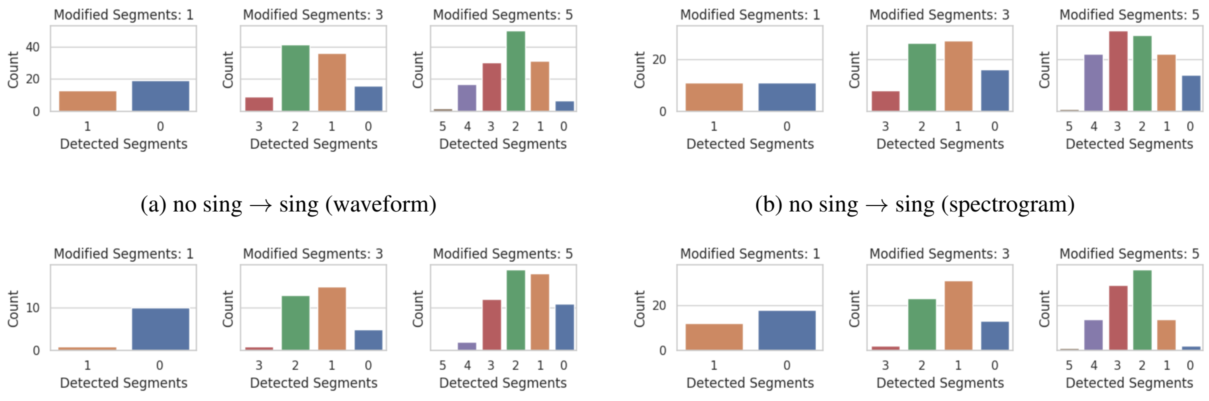
This is similar to [15], where adversarial perturbations were constrained beforehand such that only selected regions of the input could be modified, but in contrast we do not restrict the perturbations directly. Instead we refine our adversarial perturbations by first splitting them into segments that correspond exactly to the time-frequency segments that LIME uses. We then use only k such segments of a perturbation, where the segments themselves are chosen based on the highest magnitude.

We then identify the subset of adversarial excerpts whose label can be flipped by only adding $k \in 1, 3, 5$ segments of the perturbation. These subsets are created for the four different settings, namely the two types of adversarial attacks (waveform / spectrogram), and for the two possible label flips ("sing" to "no sing" and vice versa). This leads to $3 * 2 * 2 = 12$ subsets of adversarial excerpts.

After determining this subset of adversarial excerpts with partial ("localised") perturbations, we once again compute explanations with LIME. Here we set the number of interpretable features that LIME should display to k, i.e. the number of segments we previously added as localised perturbations. Since the time-frequency segments of LIME and the partial perturbations align, we can then examine how often LIME correctly identifies the regions causing a particular prediction. Figure 3 shows the result of this experiment, for all 12 adversarial subsets. Each plot depicts how many of the k segments that were added as adversarial perturbation, and that hence were crucial for a particular classification, are correctly detected by LIME. It is easiest to interpret results with only one modified segment (left column), since we know exactly what the correct explanation should look like. Overall, for less than half of the excerpts is the correct segment presented as an explanation. For 3 and 5 modified segments we can also check how many segments are correctly identified, and we can observe that it is rarely all of them and for some settings none of them. This result suggests that even when the cause is quite localised and aligns with the segments that we are using as interpretable features, LIME is rarely able to recover all affected features.

5.4 Are Fidelity Scores Reliable?

As Figure 1 in Section 5.1 suggested previously, "correct" explanations are often accompanied by relatively high fidelity values. We do not know, however, whether we can use fidelity to determine if an explanation highlights the segments of the input that are really the most relevant for a prediction, and it has been demonstrated that a high-fidelity explanation of a black box model might not reflect



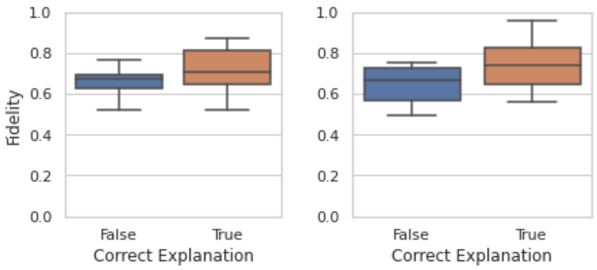
(a) no sing → sing (waveform)

(b) no sing → sing (spectrogram)

(c) sing → no sing (waveform)

(d) sing → no sing (spectrogram)

Figure 3: The number of segments that are correctly identified when adding k perturbed segments to the original input and asking LIME for an explanation containing the same k number of interpretable features.



(a) no sing → sing (waveform)

(b) no sing → sing (spectrogram)

Figure 4: Fidelity scores for explanations that correctly and incorrectly identified the most relevant segment.

the actual cause for a prediction [22].

In the last experiment described above we have created a setting with $k = 1$ where we exactly know which segment is relevant for the prediction, and we can use this information to compare whether the fidelity is different for "correct" and "wrong" explanations, i.e. explanations which match the added segments of the perturbation as opposed to explanations which do not. As can be seen from Figure 4, the fidelity for "correct" explanations is on average only slightly higher than for "wrong" predictions, with ranges of fidelity values strongly overlapping. Based on the fidelity score alone, a user hence cannot decide whether an explanation is "correct" and whether it shows the most relevant segment(s) for a prediction.

6. DISCUSSION AND CONCLUSION

In a nutshell, the central results of our investigations can be summarised as follows:

- Figure 1: "Obvious" causes for predictions can be detected with high fidelity.
- Table 1: When computing the 3 most relevant interpretable features with LIME, we detect segments that are able to flip the label for only 21-53% of the excerpts.
- Figure 2: When using a varying number of interpretable features, we detect fewer "correct" segments than when simply taking the same number of segments with the highest magnitude.
- Table 2: When using all interpretable features that received a positive weight, we detect fewer "correct" segments than when taking the same number of segments with the highest magnitude.
- Figure 3: In a setting where we only add partial perturbations, only few segments are correctly detected.
- Figure 4: Based on the fidelity score it is impossible to judge the quality of an explanation.

Taken together, we believe that these results support the following conclusions: (1) local model-agnostic explanations cannot reliably detect the input regions most relevant for a prediction unless they are rather obvious; (2) evaluation based on "what looks reasonable" leads to accepting explanations that do not reveal the real cause of a prediction; (3) the fidelity score may give a false sense of security about the quality of explanations.

In [15], Göpfert et al. write, "It might be tempting to judge explanatory methods on whether they succeed in identifying features that a human observer thinks should be relevant to the classification, but the existence of adversarial examples shows that the reasoning of humans and neural networks can differ dramatically." This observation points to a fundamental dilemma between "veridical" explanations that faithfully reflect the workings of the classification model (and may not be accessible to model-agnostic explanation methods such as LIME), and "human-interpretable" explanations that would connect a machine decision to concepts familiar to us. Our results confirm that one should be careful in interpreting model-agnostic explanations as explanations of the underlying model, and that experiments with carefully crafted examples are important to get more detailed insights into the properties of such methods.

7. ACKNOWLEDGEMENTS

This work is supported by the Austrian National Science Fund (FWF, project P31988).

8. REFERENCES

[1] S. Mishra, B. L. Sturm, and S. Dixon, “Understanding a Deep Machine Listening Model Through Feature Inversion,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, 2018, pp. 755–762.

[2] ——, “"What are You Listening to?" Explaining Predictions of Deep Machine Listening Systems,” in *Proceedings of the 26th European Signal Processing Conference, EUSIPCO 2018, Roma, Italy, September 3-7, 2018*. IEEE, 2018, pp. 2260–2264.

[3] S. Mishra, D. Stoller, E. Benetos, B. L. Sturm, and S. Dixon, “GAN-based Generation and Automatic Selection of Explanations for Neural Networks,” *CoRR*, vol. abs/1904.09533, 2019.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, “"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, San Francisco, CA, USA, August 13-17, 2016*. ACM, 2016, pp. 1135–1144.

[5] S. Mishra, B. L. Sturm, and S. Dixon, “Local Interpretable Model-agnostic Explanations for Music Content Analysis,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, 2017, pp. 537–543.

[6] B. Chettri, B. L. Sturm, and E. Benetos, “Analysing Replay Spoofing Countermeasure Performance under Varied Conditions,” in *28th IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2018, Aalborg, Denmark, September 17-20, 2018*. IEEE, 2018, pp. 1–6.

[7] V. Haunschmid, S. Chowdhury, and G. Widmer, “Two-level Explanations in Music Emotion Recognition,” *Machine Learning for Music Discovery Workshop, ML4MD at ICML2019*, 2019.

[8] V. Haunschmid, E. Manilow, and G. Widmer, “Towards Musically Meaningful Explanations Using Source Separation,” *CoRR*, vol. abs/2009.02051, 2020.

[9] ——, “audioLIME: Listenable Explanations Using Source Separation,” in *13th International Workshop on Machine Learning and Music*, 2020, pp. 20–24.

[10] A. B. Melchiorre, V. Haunschmid, M. Schedl, and G. Widmer, “LEMONS: Listenable Explanations for Music recOmmeNder Systems,” in *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part II*, ser. Lecture Notes in Computer Science, vol. 12657. Springer, 2021, pp. 531–536.

[11] S. Mishra, E. Benetos, B. L. Sturm, and S. Dixon, “Reliable Local Explanations for Machine Listening,” in *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 2020, pp. 1–8.

[12] Z. C. Lipton, “The Mythos of Model Interpretability,” *Communications of the ACM*, vol. 61, no. 10, pp. 36–43, 2018.

[13] J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim, “Sanity Checks for Saliency Maps,” in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 2018, pp. 9525–9536.

[14] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, “Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods,” in *AIES ’20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*. ACM, 2020, pp. 180–186.

[15] J. P. Göpfert, H. Wersing, and B. Hammer, “Recovering Localized Adversarial Attacks,” in *Artificial Neural Networks and Machine Learning - ICANN 2019: Theoretical Neural Computation - 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17-19, 2019, Proceedings, Part I*, ser. Lecture Notes in Computer Science, vol. 11727. Springer, 2019, pp. 302–311.

[16] J. Schlüter and B. Lehner, “Zero-Mean Convolutions for Level-Invariant Singing Voice Detection,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, 2018, pp. 321–326.

[17] J. Schlüter, “Deep Learning for Event Detection, Sequence Labelling and Similarity Estimation in Music Signals,” Ph.D. dissertation, Johannes Kepler University Linz, Austria, Jul. 2017.

[18] J. Schlüter and T. Grill, “Exploring Data Augmentation for Improved Singing Voice Detection with Neural Networks,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, 2015, pp. 121–126.

[19] N. Carlini and D. A. Wagner, “Audio Adversarial Examples: Targeted Attacks on Speech-to-Text,” in *Proceedings of the IEEE Security and Privacy Workshops, SP Workshops*. IEEE, 2018, pp. 1–7.

[20] K. Prinz, A. Flexer, and G. Widmer, “On End-to-End White-Box Adversarial Attacks in Music Information Retrieval,” *Transactions of the International Society for Music Information Retrieval*, vol. 4, no. 1, pp. 93—-104, 2021.

[21] M. Ramona, G. Richard, and B. David, “Vocal Detection in Music with Support Vector Machines,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2008, Las Vegas, Nevada, USA*. IEEE, 2008, pp. 1885–1888.

[22] H. Lakkaraju and O. Bastani, “"How do I fool you?": Manipulating User Trust via Misleading Black Box Explanations,” in *AIES ’20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*. ACM, 2020, pp. 79–85.