ROBUST LOSSY AUDIO COMPRESSION IDENTIFICATION

Hendrik Vincent Koops **Gianluca Micchi** **Elio Quinton**
Music and Audio Machine Learning Lab,
Universal Music Group, London, UK
{vincent.koops, gianluca.micchi, elio.quinton}@umusic.com

ABSTRACT

Previous research contributions on blind lossy compression identification report near perfect performance metrics on their test set, across a variety of codecs and bit rates. However, we show that such results can be deceptive and may not accurately represent true ability of the system to tackle the task at hand. In this article, we present an investigation into the robustness and generalisation capability of a lossy audio identification model. Our contributions are as follows. (1) We show the lack of robustness to codec parameter variations of a model equivalent to prior art. In particular, when naively training a lossy compression detection model on a dataset of music recordings processed with a range of codecs and their lossless counterparts, we obtain near perfect performance metrics on the held-out test set, but severely degraded performance on lossy tracks produced with codec parameters not seen in training. (2) We propose and show the effectiveness of an improved training strategy to significantly increase the robustness and generalisation capability of the model beyond codec configurations seen during training. Namely we apply a random mask to the input spectrogram to encourage the model not to rely solely on the training set's codec cutoff frequency.

1. INTRODUCTION

Audio codecs can be roughly categorized into two categories: *lossless* and *lossy*. *Lossless* means that an exact preservation of the signal is guaranteed by the codec. In other words, the signal resulting from encoding and decoding is exactly identical to the original. In contrast, *lossy* encoding means that some of the signal is lost in the encoding and decoding process. In other words, the signal resulting from encoding and decoding is not exactly identical to the original signal.

Popular lossy audio codecs like MP3 [1], Ogg Vorbis [2] or AAC [3] are known as "perceptual" codecs because they rely on models of human auditory cognition to prioritise the deletion of parts of the audio signal that have the least perceptual impact on human listeners. Despite the signal degradation that they result in, perceptual lossy codecs can achieve much greater compression ratios than lossless codecs, and are therefore well suited for applications where data bandwidth is limited. For example, they have been instrumental in enabling music streaming over networks with limited bandwidth.

Digital audio codecs are readily available and are integrated into many widespread professional and consumer tools such as Digital Audio Workstations, software libraries, digital music players etc., which make converting an audio file from one format to another nowadays extremely easy and accessible to anyone. As a result it is easy to mistakenly encode a source audio signal with a lossy codec, which degrades the signal, and then decode it back into a lossless file container. This process may create the illusion that a lossless file container (e.g. WAV) contains unimpaired audio when it does in fact contain lossy-compressed audio.

Guaranteeing audio integrity is essential in many applied scenarios such as large scale music distribution or archiving. Because the aforementioned case of lossy audio disguised as a lossless file would violate this guarantee, there is a need to automatically detect such occurrences. Identification of audio that has been compressed with a lossy codec is a valuable component of quality assurance processes, which form an important part of many modern musical audio content pipelines.

Contributions. In this paper, we present an investigation into the robustness and generalisation capability of a lossy audio identification model. We show that when we naively train a lossy compression detection model on a dataset of music recordings processed with a range of codecs and their lossless counterparts, we obtain near perfect performance metrics on the held-out test set. However, we obtain severely degraded performance on lossy tracks produced with codec parameters not seen in training. We also propose a new training schema in which we randomly mask the input spectrogram to improve the model's robustness. We show that our approach significantly increases the robustness and generalisation capability of the model beyond codec configurations seen during training.

2. BACKGROUND

In the following sections, we will first provide a high level overview of lossy audio codecs (Section 2.1). Next, in Sec-

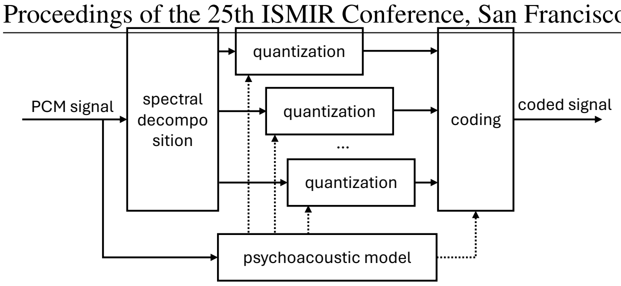

Figure 1. Basic block diagram of a perceptual audio coder. After spectral decomposition, a psychoacoustic model informs the quantization of individual spectral components.

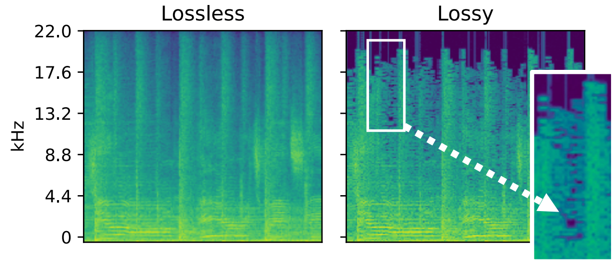

Figure 2. Spectrograms of examples of a lossless (left) and lossy version of the same audio excerpt (right). The latter is compressed with the LIBFDK_AAC codec at 128 kbps bit rate. The version on the right shows the hallmarks of lossy compression: removal of FFT coefficients, holes in the spectrum, and general loss of higher frequency content.

tion 2.2, related work on lossy audio identification is discussed. Finally, in Section 2.3, we briefly present related work in MIR on robustness evaluation.

2.1 Lossy Codecs

Figure 1 shows a basic block diagram with the common modules of a perceptual audio coder. The process of encoding an audio signal with a lossy codec is commonly as follows. First, the original uncompressed (often pulse code modulated - PCM) signal is transformed into a time-frequency representation. This is typically done using a modified discrete cosine transform (MDCT), but many other transforms have been proposed [4]. Commonly used signal block for the spectral decomposition are between 2ms and 50ms. The components of the spectral decomposition are then individually quantized. The quantization of the spectral components is controlled by a psychoacoustic model that describes the time and frequency masking properties of the human auditory system. Auditory masking is a process where one sound (maskee) becomes inaudible in the presence of another sound (mask) [5].

Auditory masking can occur in the time domain (temporal masking) or in the frequency domain (frequency masking). The quantization controlled by the psychoacoustic model effectively controls which spectral coefficients will be removed, resulting in spectral band rupture and holes in spectrograms, as observed in Figure 2. After quantization, Huffman coding (or some other form of entropy coding) is applied to remove or reduce the redundancy in the signal [6]. The bit rate of a codec effectively controls both the size and the perceptual quality of the audio. A low bit rate (like 128 kbps) will produce a small storage footprint, but generally worse perceptual quality compared to a higher bit rate (like 320 kbps). For more detail on audio codecs and standards, we refer to [4].

2.2 Lossy Compression Identification

In previous research, multiple blind lossy compression identification models have been proposed. These can broadly be categorized into two approaches. One approach is to estimate codec parameters from the audio signal, to determine factors such as the decoder framing grid, filter bank parameters and/or quantization information. This type of approach has been successfully applied for individual codecs like AAC [7], MP3 [8–10]. Although this type of approach can be very effective, it is computationally very expensive, especially when multiple codecs are considered.

The second method utilizes audio quality measures to determine whether the audio is lossy. One effect of lossy audio compression is the introduction of “holes” in the spectrogram, especially right after louder transients. This is the result of the fact that spectral coefficients can be removed when they are perceptually masked by other coefficients. Therefore, most approaches present some form of “hole-detection“, such as estimating the number of inactive spectral coefficients (e.g. [9,11]) or computing spectral fluctuations [12–15].

In [16], Hennequin et al. presented a method for detecting lossy compression based on a convolutional neural network CNN applied to audio spectrograms. Similarly, Seichter et al. in [17] also proposed a CNN approach for AAC encoding detection and bit rate estimation. All research contributions on lossy compression identification almost uniformly report near-perfect performance metrics on their test set, across a variety of codecs and bit rates.

However, most codecs can be configured with parameters other than the bitrate too, such as a cutoff frequency that controls the amount of higher frequencies that will be preserved. AAC for example has a default cutoff frequency of around 17kHz [18] for constant bit rates of 96 kbps per channel and above, which means that the bandwidth of the encoder is set to 0 - 17kHz. None of the previous research explores what happens when this parameter is changed.

In this paper we show that a model naively trained on default parameters may not efficiently learn to discriminate lossy audio encoded with different parameters and we analyse what happens when varying the cutoff frequency as an example. Therefore, the good results previously reported must be taken with a pinch of salt.

2.3 Robustness Evaluation in Music Information Retrieval

Several studies in music information retrieval have shown that models can seemingly achieve very high evaluation performance, while further research reveals that what those models have learned is some confound with the ground truth dataset [19]. For example, in a research into the robustness of genre classification models, Sturm showed that although these systems might have high mean classification accuracies, they don't actually reflect the underlying properties of the genre [20]. Furthermore, it is shown that by filtering the audio signal in a minimal way, the models produce radically different genre predictions. For a larger overview of music adversaries in music information retrieval research, we refer to [19]. Bob Sturm in [21] introduced the term "horse" [1] to refer to system appearing capable of achieving high evaluation performance, but actually working by using irrelevant characteristics (confounds), and therefore not actually addressing the problem it appears to be solving.

3. METHOD

In the following sections, we will first describe our model setup (in Section 3.1), then our dataset (in Section 3.2) and finally our proposed evaluation methods (in Section 3.3).

3.1 Network Architecture

For the detection of lossy audio we propose a model (visualized in Figure 3) that can be divided into four parts: a spectrogram + random mask module, 4 convolutional blocks, an `lstm` block and a classification head made of a single dense layer. The architecture is partly inspired by prior work by Hennequin et al. in [16] and Seichter et al. in [17]. In the following sections, we will describe each part in detail.

The model takes as input 2 seconds of raw monophonic audio signal sampled at 44.1 kHz, which is passed to a *torchaudio* spectrogram layer that produces a magnitude spectrogram with 1024 FFT coefficients [22].

Random mask. A random mask is optionally applied to the input spectrogram. This is achieved by uniformly randomly sampling a cutoff frequency between 14 kHz and the Nyquist frequency of the sample, and nulling all *fft* coefficients above that frequency by setting them to the minimum of the input spectrogram. A similar approach called Specaugment was proposed by Park et al. in [23].

In our first experiment (as described in Section 4.1) this layer is not used, and the spectrogram is directly fed to the convolutional blocks. However, in the second experiment (as described in Section 4.2), we use this random mask layer with a different random cutoff frequency for every training example.

CNN. Each of the CNN blocks consist of four layers: a 2D convolutional layer with a kernel size of (3,3), a ReLU layer, a batch normalization layer and a 2D max-pooling layer. The max pooling size for each block is (2,2), with the exception of the last block, which is (2,4).

[1] A nod to the Clever Hans horse, see https://en.wikipedia.org/wiki/Clever_Hans

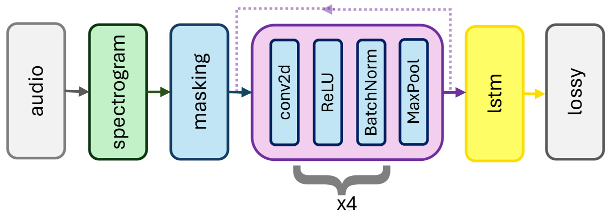

Figure 3. Proposed model for the detection of lossy audio Our model takes as input 2 seconds of audio, which is passed to a torchaudio spectrogram layer (in green). Depending on the experiment, the spectrogram is then passed to a masking layer (in blue), which simulates low-pass filtering. The spectrogram is then passed to four convolutional modules (in pink). We use a bi-directional LSTM (in yellow) for dimensionality reduction. We classify the audio into lossy or lossless in the final model head.

LSTM. We connect the CNN to a *long short-term memory* (LSTM) block for two reasons. Firstly, we want to exploit possible sequential properties of the CNN output, and secondly, for dimensionality reduction for the last (dense) part of the network. We use a bidirectional LSTM with two layers of size 128.

Classification head. Our model's lossy/lossless classification head is connected to the LSTM output with a dense layer of size 256 (2x 128 because our LSTM is bidirectional). The classification head has a softmax activation and 2 outputs that model the probability of the example being lossless or lossy.

Training. We back-propagate our model on the binary cross entropy of the classification head and the ground truth. For each audio track, we take a 2-second random crop at training time.

3.2 Datasets

For our experiments, we sample 10k tracks of lossless 16 bit, 44.1kHz WAV files from a large private library of commercial music. From these tracks, we create two datasets.

3.2.1 DS1.

For the first dataset we encode each track with a codec randomly chosen among LIBMP3LAME (MP3), LIBFDK_AAC (AAC) and LIBVORBIS (OGG), with bit rate also randomly chosen among 128, 256 and 320 kbps. Each encoded file is then decoded back into a 16bit, 44.1 kHz WAV file that is used as input to the model. All the encoding/decoding is done using *ffmpeg* [24]. Between lossless and lossy tracks, the dataset comprises of 20k tracks.

3.2.2 DS2.

For the second dataset, we use the same original tracks as were used to create DS1. We also use the same codec parameters, but vary the cutoff frequency of the codecs,

Codec	LIBFDK_AAC			LIBVORBIS			LIBMP3LAME			Lossless	Mean
Bit rate	128k	256k	320k	128k	256k	320k	128k	256k	320k	—	—
DS1	100.0	98.91	100.0	100.0	100.0	100.0	100.0	100.0	98.37	99.88	99.79
DS2	31.38	28.96	24.74	98.91	93.16	86.7	80.63	68.45	60.87	99.88	81.85

Table 1. Accuracy of evaluating the model without random mask on a dataset without (DS1) and with cutoff frequency variations (DS2). Varying the cutoff parameter in the codec greatly degrades model results.

choosing among 14, 16, 18 and 20 kHz. DS1 and DS2, therefore, differ only on the lossy versions obtained for each track. We use the same random 70/10/20 split for training/validation/testing for both datasets. All our experiments are run using DS1 for training and validation. Evaluation is done on DS1 (cf. Sec. 4.1) or DS2 (Sec. 4.2).

3.3 Evaluation

We evaluate the performance of our lossy/lossless detection model in three ways. Firstly, we provide quantitative evaluation and report the model accuracy. Secondly, we inspect saliency maps of the CNN blocks of our model to gain qualitative insight into what signal properties the model is sensitive to. Finally, we also inspect the errors of our model in detail to help us assess the effectiveness our proposed method to make our model more robust, and identify avenues for future work.

4. EXPERIMENTS & RESULTS

In this section we first describe our experiments and report our results on a naively trained lossy/lossless audio detection model (Section 4.1). After an analysis of our results, we report on a more robust variation of our model in Section 4.2, and an analysis of errors in Section 4.3.

4.1 Experiment 1: Naive Model Training

In our first experiment, we train our model on DS1. For each track in our test set, we extract 2-second windows of raw audio with 50% overlap. For each window, we perform a forward pass through our trained network, and collect the output of the classification head. We take the mean of all windowed local model outputs as the global output per track.

4.1.1 Results

In line with previous research (e.g. [16, 17]), we find near-perfect performance on lossy/lossless audio detection of audio with default codec settings. The top row of Table 1 shows the results broken down by codec and bit rate for DS1. We obtain near-perfect results per bit rate/code combination. On average, we obtain 99.79% accuracy across all codecs and lossless files.

However, if we slightly tweak the codec parameters at test time (i.e. we test our model on DS2) the performance drops significantly. The bottom row of Table 1 shows the results of evaluating the model on the dataset with cutoff frequency variations. The results show much poorer results for the lossy tracks across all codec/bit rate combinations. Specifically, we find a big drop in accuracy of around 70 percentage points for the LIBFDK_AAC codec and around 30 percentage points for the MP3 codec. The LIBVORBIS is less impacted, but is still significantly impacted by around 10 percentage points.

4.1.2 Analysis

To get a better sense of what our model has learned, we turn towards a feature analysis of the CNN part of the network. When inspecting the spectrogram of a potentially lossy file with the naked eye, one of the most striking aspects is the nulling of coefficients, resulting in "holes" in the spectrogram. We expected the convolutional part of the network to pick up on those, and to design features that capture this phenomenon.

However, when we visualize saliency maps from our network, we find a different pattern (see Figure 4, top row). It seems that the model is more concerned with the cutoff frequency of the lossy audio than with the holes in the spectrogram. Although the cutoff frequency is a useful feature, by itself it is neither necessary nor sufficient to determine whether an audio signal has been encoded with a lossy codec.

Table 2 shows the results of the model per cutoff frequency, in the columns marked with 'No'. Here again we see that most cutoff frequency variations are severely underperforming when compared to the previous test dataset.

The model performs best at a cutoff frequency of 16 kHz. This can be explained by the fact that this is the default cutoff frequency of LIBVORBIS, which is therefore not affected by this transformation. In the next section, we adapt the model to be robust against this cutoff effect.

4.2 Experiment 2: Creating a Robust Model

In order to increase the model's robustness against the lossy codec's cutoff frequency, we present a second experiment where we randomly mask the upper end of the spectrum. The mask, defined in 3.1, is applied to all input files.

The application of this random mask is intended to force the model not to solely rely on the codec cutoff frequency to make a prediction, and instead also rely on other signal degradations included by codecs, such as "holes" in the spectrogram. A fixed mask at a specific cutoff frequency would have meant throwing away the information given by the spectral rolloff entirely and this would have been suboptimal in the opposite direction.

We train this model on DS1 and evaluate on DS2.

	LIBFDK_AAC		LIBVORBIS		LIBMP3LAME		128k		256k		320k		MEAN	
Cutoff	No	Mask	No	Mask	No	Mask	No	Mask	No	Mask	No	Mask	No	Mask
14 kHz	24.1	**81.0**	100.0	**100.0**	76.9	**100.0**	**90.4**	82.6	53.7	**100.0**	49.3	**97.8**	65.9	**93.3**
16 kHz	83.9	**98.4**	100.0	**100.0**	98.6	**100.0**	88.2	**100.0**	97.0	**97.7**	98.6	**100.0**	94.6	**99.5**
18 kHz	0.7	**86.7**	66.9	**100.0**	25.3	**100.0**	50.0	**96.2**	28.2	**94.1**	12.4	**94.8**	29.5	**95.6**
20 kHz	11.1	**96.5**	100.0	**100.0**	82.9	**100.0**	46.2	**100.0**	76.1	**97.2**	70.2	**100.0**	65.1	**98.9**
MEAN	28.3	**90.1**	92.9	**100.0**	70.1	**100.0**	70.1	**93.6**	64.1	**97.9**	57.3	**98.8**	63.7	**96.8**

Table 2. Accuracy (in percentage points) of evaluating our models without (No) and with (Mask) random mask on DS2, per codec and bit rate, for varying cutoff frequency. Lossless accuracy is 99.9% for No and 99.8% for Mask.

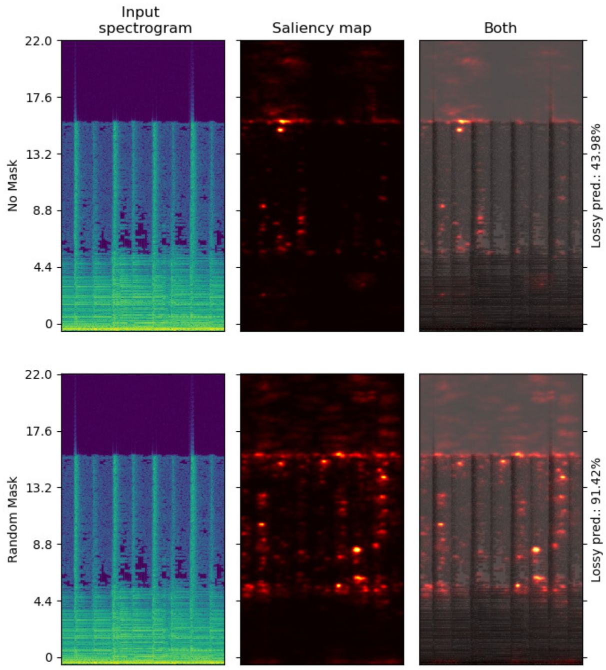

Figure 4. Saliency maps from exposing a model trained without (top) and with (bottom) random mask to lossy audio. The model with random mask shows more activation in the holes of the spectrogram without losing any of the activations at the cutoff frequency.

4.2.1 Results

Table 2 shows the results obtained for the model trained with the random mask on DS1 and evaluated on DS2. We observe good classification results on average, 96.8% on lossy files and 99.8% on lossless files. Overall, we obtain 98.4% lossy/lossless classification accuracy across the entire test dataset. Comparing with the naive model, the accuracy on DS2 improves significantly across the board.

With the mean classification accuracy at 90% or above in all conditions (last column of the table), this model is broadly robust against cutoff frequency variations. It is interesting to note that performance on the AAC codec is comparatively lower than on other codecs. This result suggests that the AAC codec is more challenging to detect, and warrants further investigation, which we leave for future work. We hypothesise it may be due to the AAC codec producing less artefacts in the magnitude spectrogram.

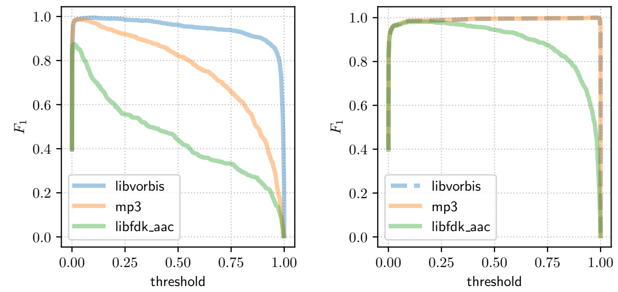

Figure 5. F1-score for varying thresholds, evaluated on DS2. Each line analyses the subset made of lossless files as negatives and the specified codec as positives; files encoded with different codecs are discarded. Left: model without random mask; Right: model with random mask.

In Fig. 5, for both the model without random mask (cf. Section 4.1) and the model with (cf. Section 4.2) we plot the F_1 score (i.e., the harmonic mean between precision and recall) as a function of the threshold of the binary classification prediction. The F_1-score for the model without the random mask peaks at very low values of the threshold and then decays for increasing threshold at a rate that highly depends on the codec analysed.

This suggests three conclusions: (1) There are a number of test set files that yield a prediction $p(x)$ in the central region $0.1 < p(x) < 0.9$, which shows a high degree of uncertainty for the model; (2) since the F1-score is monotonously decreasing, the model tends to output false negatives rather than false positives; (3) different codecs are identified with different level of proficiency.

Compare this with the output for the model with the random mask: in the case of LIBMP3LAME and LIBVORBIS codecs, the F_1-score is almost flat and close to 1 for the entire range of thresholds. The LIBFDK_AAC codec still shows some decrease in performance for increasing thresholds, but the peak value increased from 0.875 to 0.982 and the area under the F_1 curve jumped from 0.450 to 0.891. From the results above we can conclude that the introduction of the random mask brings higher peak performance and also reduces the impact of the choice of the threshold.

4.2.2 Analysis

Similarly to the analysis presented in Section 4.1.2, we visualise saliency maps of the model trained with the random mask in the bottom row of Figure 4. Compared to the saliency of the model with no mask (top row), we

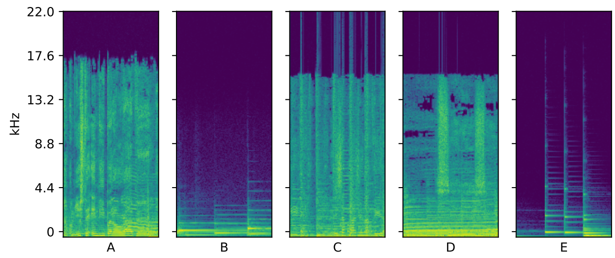

Figure 6. The five assumed lossless tracks misidentified as lossy. However, A, C, and D are in fact lossy. B and E are quiet tracks with a single instrument.

see a much brighter activation in the holes of the spectrogram without losing any of the activations at the cutoff frequency. The model has learned to rely on more markers to make its choice.

4.3 Qualitative Analysis of Errors

In this section, we present a qualitative analysis of the erroneous predictions produced by our model trained with the random mask.

4.3.1 Lossless Errors.

From our entire test subset of DS2, we observe only 5 cases (0.2%) where the model made a "lossy" prediction while the recording is in the lossless part of the dataset. The spectrogram of three out of the five tracks (A, C and D in Figure 6) show the hallmarks of lossy compression. It appears that our model was indeed correct in predicting a lossy encoding, and therefore revealed "in-the-wild" cases of accidental lossy compression that were present in our dataset.

The other two tracks (B and E) are quiet and sparsely orchestrated tracks. It is notable that the spectrogram also appears sparse, with very little energy in the upper frequency range. Given that lossy codecs often feature energy depletion in the top part of the frequency range, we hypothesise that the misclassification may be due to the model relying on the absence of energy in the upper register in this case.

4.3.2 Lossy Errors.

Table 2 shows that the entirety of cases where the model erroneously classified recordings as lossless when it should be lossy comes from tracks encoded with the LIBFDK_AAC codec. In Figure 7, spectrograms of 2 second excerpts from a random selection of error tracks are visualized.

From inspecting the spectrograms of LIBFDK_AAC encoded tracks, we find that common characteristics are (1) the spectral roll-off is relatively stable over time, (2) the preservation of transients above the cutoff frequency, which can often span upwards to the Nyquist frequency, and (3) less nulling of spectral coefficients, resulting in fewer holes in the spectrogram. The LIBFDK_AAC codec is a superior codec in terms of compression efficiency, meaning it can provide better audio quality at lower bitrates than other codecs [25].

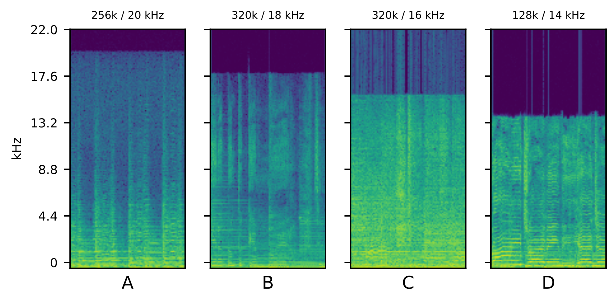

Figure 7. A random selection of lossy tracks misidentified as lossless. All tracks are encodec with LIBFDK_AAC. The spectrograms show less holes and band rupture compared to other codecs, especially under 14 kHz.

Table 2 shows that AAC with cutoff 14kHz is only 81% accuracy. We hypothesize that the LIBFDK_AAC codec does not produce as much “holes” in the spectrogram below this threshold. Our model applies the random mask to every example in our training dataset, which can be confusing on LIBFDK_AAC samples. That is, as the random mask is applied at a relatively low cutoff frequency, the resulting spectrogram is almost identical to a lossless example. One avenue for future work could be to apply the random mask with a lower probability, to allow the model to also learn other spectral characteristics of LIBFDK_AAC samples.

5. CONCLUSION

In this paper, we presented a lossy audio compression detection method that can robustly estimate whether a given audio file has been lossy encoded before. We show that naively training a model results in near-perfect lossy audio compression detection on the held-out test set generated using the same encoding parameters.

However, we find that, for several widely used lossy codecs, the performance of this model catastrophically degrades when exposed to variations of the cutoff frequency parameter that were not seen during training. This result suggests that a naively trained model is overly reliant on the cutoff value. In response to this shortcoming, we propose to amend the training strategy by applying a random mask to the upper range of the spectrogram, in order to reduce the model’s reliance on the codec cutoff frequency value.

We show that this method results in a model that is significantly more robust against frequency cutoff variations. Our experiments reveal compelling performance on all codec and bit rate combinations we considered, but reveal that there remains room for improvement on the detection of the LIBFDK_AAC codec. We hypothesise that the AAC codec is comparatively more difficult to detect than MP3 and Ogg Vorbis because it generates less artefacts in the magnitude spectrogram. An avenue for future work may consist in exploring further development of the training strategy in order to improve performance on the AAC codec.

6. REFERENCES

[1] H. G. Musmann, "Genesis of the mp3 audio coding standard," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 3, pp. 1043–1049, 2006.

[2] Xiph. (2024) Vorbis audio compression. [Online]. Available: https://www.xiph.org/vorbis/

[3] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs, and M. Dietz, "Iso/iec mpeg-2 advanced audio coding," *Journal of the Audio engineering society*, vol. 45, no. 10, pp. 789–814, 1997.

[4] M. Bosi and R. E. Goldberg, *Introduction to digital audio coding and standards*. Springer Science & Business Media, 2002, vol. 721.

[5] S. A. Gelfand, *Hearing: An introduction to psychological and physiological acoustics*. CRC Press, 2017.

[6] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[7] J. Herre and M. Schug, "Analysis of decompressed audio-the-inverse decoder," in *Audio Engineering Society Convention 109*. Audio Engineering Society, 2000.

[8] P. Bießmann, D. Gärtner, C. Dittmar, P. Aichroth, M. Schnabel, G. Schuller, and R. Geiger, "Estimating MP3PRO encoder parameters from decoded audio," 2013. [Online]. Available: https://dl.gi.de/handle/20.500.12116/20701

[9] S. Moehrs, J. Herre, and R. Geiger, "Analysing Decompressed Audio with the "Inverse Decoder" - Towards an Operative Algorithm," in *Audio Engineering Society Convention 112*. Audio Engineering Society, 2002.

[10] S. Hiçsönmez, H. T. Sencar, and I. Avcibas, "Audio codec identification through payload sampling," in *2011 IEEE International Workshop on Information Forensics and Security*. IEEE, 2011, pp. 1–6.

[11] R. Yang, Y.-Q. Shi, and J. Huang, "Defeating fake-quality MP3," in *Proceedings of the 11th ACM Workshop on Multimedia and Security*, 2009, pp. 117–124.

[12] R. Yang, Z. Qu, and J. Huang, "Detecting digital audio forgeries by checking frame offsets," in *Proceedings of the 10th ACM Workshop on Multimedia and Security*, 2008, pp. 21–26.

[13] B. Kim and Z. Rafii, "Lossy audio compression identification," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2459–2463.

[14] O. Derrien, "Detection of genuine lossless audio files: Application to the MPEG-AAC codec," *Journal of the Audio Engineering Society*, vol. 67, no. 3, pp. 116–123, 2019.

[15] B. D'Alessandro and Y. Q. Shi, "MP3 bit rate quality detection through frequency spectrum analysis," in *Proceedings of the 11th ACM Workshop on Multimedia and Security*, 2009, pp. 57–62.

[16] R. Hennequin, J. Royo-Letelier, and M. Moussallam, "Codec independent lossy audio compression detection," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 726–730.

[17] D. Seichter, L. Cuccovillo, and P. Aichroth, "Aac encoding detection and bitrate estimation using a convolutional neural network," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2069–2073.

[18] Fraunhofer. (2024) Fraunhofer FDK AAC. [Online]. Available: https://wiki.hydrogenaud.io/index.php?title=Fraunhofer_FDK_AAC#Bandwidth

[19] C. Kereliuk, B. L. Sturm, and J. Larsen, "Deep learning and music adversaries," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 2059–2071, 2015.

[20] B. L. Sturm, "Two systems for automatic music genre recognition: what are they really recognizing?" in *Proceedings of the Second International ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies*, ser. MIRUM '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 69–74. [Online]. Available: https://doi.org/10.1145/2390848.2390866

[21] ——, "A Simple Method to Determine if a Music Information Retrieval System is a "Horse"," *IEEE Transactions on Multimedia*, vol. 16, no. 6, pp. 1636–1644, 2014.

[22] J. Hwang, M. Hira, C. Chen, X. Zhang, Z. Ni, G. Sun, P. Ma, R. Huang, V. Pratap, Y. Zhang, A. Kumar, C.-Y. Yu, C. Zhu, C. Liu, J. Kahn, M. Ravanelli, P. Sun, S. Watanabe, Y. Shi, Y. Tao, R. Scheibler, S. Cornell, S. Kim, and S. Petridis, "Torchaudio 2.1: Advancing speech recognition, self-supervised learning, and audio processing components for pytorch," 2023.

[23] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Interspeech 2019*, Sep 2019. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680

[24] FFMpeg. (2024) FFMpeg. [Online]. Available: https://ffmpeg.org/

[25] K. Brandenburg, "MP3 and AAC explained," in *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding*. Audio Engineering Society, 1999.