Exploring Audio Compression as Image Completion in Time-Frequency Domain

Giovanni Scodeller[0009−0003−2984−2949], Mara Pistellato[0000−0001−6273−290X], and Filippo Bergamasco[0000−0001−6668−1556]

DAIS, Università Ca'Foscari Venezia. 155, via Torino, Venezia Italy
giovanni.scodeller@gmail.com
{mara.pistellato, filippo.bergamasco}@unive.it

Abstract. Audio compression is usually achieved with algorithms that exploit spectral properties of the given signal such as frequency or temporal masking. In this paper we propose to tackle such a problem from a different point of view, considering the time-frequency domain of an audio signal as an intensity map to be reconstructed via a data-driven approach. The compression stage removes some selected input values from the time-frequency representation of the original signal. Then, decompression works by reconstructing the missing samples as an image completion task. Our method is divided into two main parts: first, we analyse the feasibility of a data-driven audio reconstruction with missing samples in its time-frequency representation. To do so, we exploit an existing CNN model designed for depth completion, involving a sequence of sparse convolutions to deal with absent values. Second, we propose a method to select the values to be removed at compression stage, maximizing the perceived audio quality of the decompressed signal. In the experimental section we validate the proposed technique on some standard audio datasets and provide an extensive study on the quality of the reconstructed signal under different conditions.

Keywords: Audio compression · CNN · Sparse convolutions · Spectrogram · genetic algorithm

1 Introduction

Storing or transmitting digital audio data often involves compression, that can be lossy or lossless. Classical audio compression and decompression algorithms remove redundant or irrelevant information from data exploiting different properties [6, 2]. In particular, when a lossy approach is adopted, the algorithm discards the information that are considered as inaudible or less important to the human hearing: this is performed moving from time to frequency domain (typically via the Fourier transform), so that the compression can be performed directly on frequencies, exploiting spectral properties of the signal. The literature counts several approaches to perform effective compression for speech and generic audio signals [11, 4], also targeted to specific applications such as wireless communications [7]. Recently, data-driven approaches proved to be effective in a variety of applications [20, 19], offering targeted solutions [3]. Among them, some works proposed data-driven approaches to tackle the problem of audio data compression. An early work presented in [16] proposed a speech compression technique using a three-layer perceptron, where outputs of the hidden layer are quantized to perform the compression. Despite its limitations, we can consider such work a first attempt at applying autoencoders for audio compression. The authors in [24] introduce a neural network (NN) model to compress audio signals through an encoder/decoder architecture and residual vector quantizer. Some works propose NN models specifically designed for speech compression [12, 13]. In all such examples, audio signals are treated by the model as one-dimensional input signals. Other applications of data-driven approaches involving audio signal processing include audio super-resolution (or bandwidth extension) [14] or speech dereverberation and denoising [23], where the authors provide a time-frequency approach.

In this paper we propose a first study on audio compression using data-driven techniques designed for image processing. Specifically, we investigate the usage of CNNs composed of Sparse Convolution layers applied to the 2D time-frequency domain representation of an audio signal. The idea of sparse convolutions was introduced in [21] to perform depth completion, with possible subsequent tasks such as 3D reconstruction [17, 18] or segmentation [22]. Later the same concept was extended and further improved by other authors [10, 9] to solve similar problems or to merge different kinds of data as intensity images [15], but has never been used for audio data specifically. In our work we shift from depth data to audio signal by working on the "image" obtained when applying the Short-Time Fourier Transform (STFT). Indeed, the main idea behind the proposed compression method is to first convert the 1-dimensional audio signal into a 2-dimensional spectrogram via STFT, and then to keep only part of such information by performing a sparse sampling of the image. Then, at decompression time, the original complete spectrogram image is reconstructed by a pre-trained CNN. To do so, we designed a network architecture that performs sparse convolutions to reconstruct the signal by filling in the missing values. Such model is specifically trained on audio data, and involves a specialised loss function.
To summarize, the contributions of this work are threefold: first, we present an "audio-completion" task that is novel with respect to other approaches proposed in the literature. The audio signal is transformed into an image, sparsified by an encoding algorithm, and reconstructed using sparse convolutions. Second, we introduce an improved version of the Griffin-Lim Algorithm (GLA) [5] that is able to enforce phase coherence given a sparse sampling of the phase image. Finally, we describe an effective compression method based on a Genetic Algorithm that is able to optimize the pixel selection process keeping the desired compression level. In this way the compression step is able to select only the relevant image areas so that the reconstruction preserves the best possible quality. We provide an extensive experimental evaluation where we analyse all the proposed approaches and compare the given alternatives in relation to the audio output quality and the space needed to store all the compressed values.

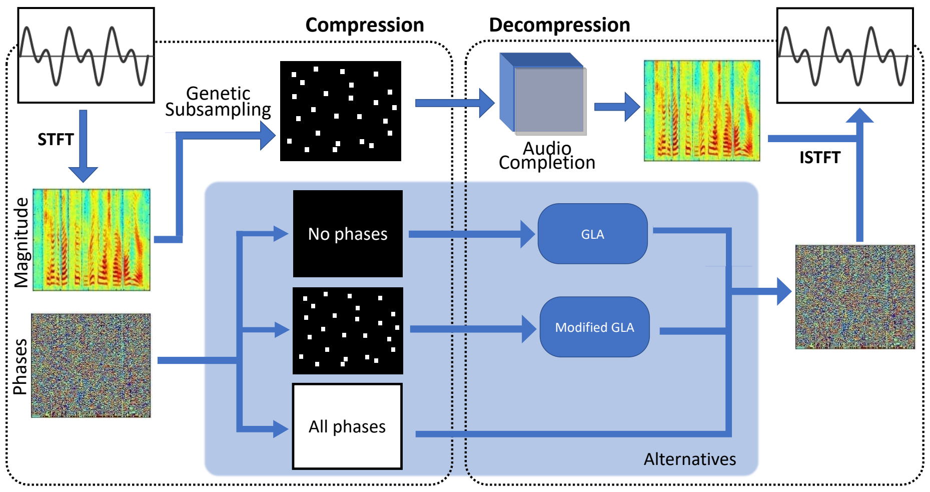

Fig. 1. Compression and decompression pipeline overview. The signal is transformed into power spectrogram and phases images. The compression happens by discarding most of the values through a custom-designed genetic sub-sampling (§2.3). We analyse three alternatives for the phases: (i) discard all of them and recover with the Griffin-Lim Algoritm, (ii) keep only the subsampled ones and apply our modified GLA, or (iii) keep all of them. At decompression stage, our Audio completion CNN (§2.1) reconstructs the spectrogram from the sparse samples, and the waveform is recovered via iSTFT.

2 Audio Compression via Sparse CNNs

Our approach works by projecting the 1-dimensional audio signal into a 2-dimensional time-frequency domain representation called spectrogram. This operation is performed with the Short-Time Fourier Transform (STFT), which is implemented by dividing the audio waveform into small overlapping segments and then applying the Fourier Transform to each of them individually. Similarly to the classical Fourier Transform, also the STFT can be inverted to get back the waveform from a spectrogram. Mathematically, it can be defined as:

$$S_{k,h} = \sum_{n=0}^{N-1} w(n)\; x(n+hH)\; e^{-i2\pi n \frac{k}{N}} \tag{1}$$

Where $h = 0, \dots, T$ is the time frame, $k = 0 \dots \frac{N}{2}$ is the frequency, N is the number of samples that we are considering for each time frame, H is the amount of overlapping defined as the hop size from one frame to another, and $w(n)$ is a windowing function to reduce the spectral leakage due to the non-integer number of harmonic periods in the time frame[1]. Since it works on different limited segments of the original data, the output spectrogram is an $F \times T$ complex

[1] the most common functions are the Hamming, Hanning, and Blackman window. We observed no relevant difference in the choice of such function for our purposes.

matrix $\mathbf{S}$ where F is the number of frequency bins, equal to half of the frame size N for the Nyquist theorem, and T is the number of frames that we analyse in the waveform depending on the length of the signal and the amount of overlap. For our purposes, it is more convenient to represent $\mathbf{S}$ in polar form, separating the log magnitude (power) of the spectrum from the phases as shown in Fig. 1.

The rationale for studying a compression method operating on the spectrogram, instead of the original waveform, is that the power spectrum is generally sparse for typical speech audio data. Indeed, most of the energy tends to cluster into just a few energy bins for each time frame. For this reason, we cast the audio compression problem as the task of reconstructing the audio signal in presence of missing samples, or gaps, in the spectrogram data. Since the power spectrum is just an image, we can apply state-of-the-art methods designed for depth completion to solve the problem in an effective way. The only difference is that the input data represents the power of each harmonic over time instead of depth information.

The whole pipeline is displayed in Fig. 1. First, the input waveform is converted to a spectrogram using the STFT. This results in an image where the number of rows depends on the bandwidth of the signal and the columns to the length of the signal itself. For simplicity, we consider a signal with a limited temporal extent but the same process can be used for an infinitely long audio stream by dividing the process into smaller chunks (we used 4 seconds chunks in all our tests). At this point, compression happens by discarding a desired amount of samples from the spectrogram (both in power spectrum and phases) according to certain criteria. In this work, we evaluate two approaches, namely random subsampling and genetic spectrogram subsampling (§ 2.3). The remaining samples are serialized into a one-dimensional stream, quantised to fixed point values with a user-selectable amount of bits, and compressed. Decompression works by “filling the gaps" caused by the samples that were removed during compression. To do so, the binary stream is decompressed, and the spectrogram is recreated together with a binary mask marking the location of the valid samples. We then apply our Audio Completion Network (§ 2.1) to recover the missing values of the power spectrum. Note that the same operation is not performed on the phases because the “phase image" is not locally smooth and therefore not easily addressable with the relatively limited receptive field of a CNN. Instead, we recover the phases directly when inverting the STFT with a modification of the Griffin-Lim algorithm (see § 2.2).

2.1 Audio Completion Network

The network architecture that we propose is based on the model presented by Uhrig et al. [21]. In particular, we are interested in the concept of *sparse convolution*, which is a convolutional-like operator designed to effectively account for sparse data (i.e. manage missing pixel values).

Suppose we have an $F \times T$ input image X, then we define $M \in \{0,1\}^{F \times T}$ to be a binary mask denoting the validity (with value $= 1$) or not of an input pixel in X. The sparse convolution operation takes as input both the image data

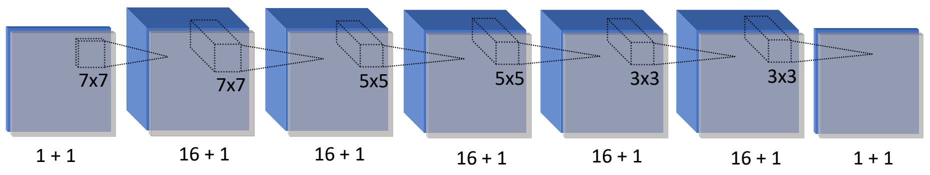

Fig. 2. CNN architecture for audio completion. We perform a sequence of 6 sparse convolutions propagating the single-channel validity mask.

$X = \{x_{i,j}\}$ and the 2-dimensional mask $M = \{m_{i,j}\}$. Considering a kernel size of $2k + 1$, the sparse convolution output at position (u, v) is computed as f as follows:

$$f_{u,v}(X, M) = \frac{\sum_{i,j=-k}^{k} m_{u+i,v+j} \;\; x_{u+i,v+j} \;\; w_{i,j}}{\sum_{i,j=-k}^{k} m_{u+i,v+j} \;\; w_{i,j} + \epsilon} + b \tag{2}$$

where $w_{i,j}$ are the kernel weights, b is bias and ϵ is a small value added to avoid division by zero. Thanks to the normalization factor computed at the denominator and the multiplication by mask values, the convolution takes into account only valid pixel values, enabling the model to be sparsity invariant. Moreover, since more valid values are computed as a result of the described operation, the input binary mask is propagated to the following layers via a function f^m, that computes the max pool operator for each (u, v) as follows:

$$f^m_{u,v}(M) = \max_{i,j=-k,\dots,k} m_{u+i,v+j} \;\; w_{i,j} \tag{3}$$

meaning that an output pixel is considered valid if at least one pixel from the input mask is equal to one.

Following a similar approach of [21], we build the audio completion network concatenating different sparse convolution blocks. In this way, the final output will be a full spectrogram image where input gaps (i.e. points for which the mask is zero) are filled. The proposed audio completion architecture is displayed in Figure 2. We start from a 2-channel input (one for the mask M and one for sparse data) and then perform six sparse convolutions with decreasing kernel sizes, namely: 7, 7, 5, 5, 3, 3. All convolutions are followed by ReLU activation and all the feature maps have 16 channels; finally, the last layer outputs the reconstructed spectrogram. We used the spectral convergence loss as proposed in [1]:

$$\mathcal{L}(Y, \hat{Y}) = \frac{||\, |Y| - |\hat{Y}| \,||_{\text{Fro}}}{||\, |Y| \,||_{\text{Fro}}} \tag{4}$$

where Y represents the ground truth spectrogram, $\hat{Y}$ is the network prediction, and $||\cdot||_{\text{Fro}}$ is the Frobenius norm.

2.2 Phase recovery

Our audio completion network reconstructs the spectrum magnitude only. Phases are recovered by taking advantage of the leakage between the sequence of time frames managed by the STFT. Since time frames overlap, phases of consecutive frames are obviously correlated depending on the frequencies. A popular technique to recover the phases in this way is the Griffin-Lim Algorithm (GLA) [5], consisting of iterative refinement of the complex spectrum obtained by applying STFT and its inverse while forcing the magnitude to the given values. We now briefly sketch the mathematical formulation to better understand our proposed modification of the original technique.

Let $\mathbf{S} = \hat{\mathbf{A}} e^{i\hat{\mathbf{\Phi}}}$ be a complex spectrum with magnitude $\hat{\mathbf{A}} \in \mathbb{R}^{T\times F}$ and phases $\hat{\mathbf{\Phi}} \in [0 \dots 2\pi]^{T\times F}$. Let $\mathcal{G}[\cdot]$, $\mathcal{G}^{-1}[\cdot]$ denote the forward and inverse STFT respectively. Given just a power spectrum $\mathbf{A}$, GLA reconstructs the full complex spectrum $\mathbf{S}$ by solving the quadratic problem:

$$\begin{aligned} \underset{\mathbf{S}}{\text{minimize}} \quad & \| \mathbf{S} - \mathcal{G}\big[\mathcal{G}^{-1}[\mathbf{S}]\big] \|_{\text{Fro}}^2 \\ \text{subject to} \quad & \hat{\mathbf{A}} = \mathbf{A} \end{aligned} \tag{5}$$

The rationale is that if $\mathbf{S}$ is not consistent (i.e. phases are not correctly correlated), applying inverse and direct STFT will produce a new (consistent) spectrum $\mathbf{S}' = \mathcal{G}\big[\mathcal{G}^{-1}[\mathbf{S}]\big]$ with different magnitudes and phases due to spectral leakage. For this reason, GLA finds the best spectrum $\mathbf{S}$ with the given magnitudes $\mathbf{A}$ minimising the Frobenius distance to a consistent spectrum $\mathbf{S}'$.

Since GLA just aims at a consistent spectrum, multiple different solutions are equally possible for the given set of amplitudes. The final result is in general intelligible, but the perceived quality may vary greatly depending on the local minima obtained during the optimisation. For this reason, we propose a modification of GLA in which some phases $\Phi_{\mathcal{S}}$ (with $\mathcal{S} = \{(k_1,h_1),\dots,(k_N,h_N)\}$) are given. The new optimization is then:

$$\begin{aligned} \underset{\mathbf{S}}{\text{minimize}} \quad & \| \mathbf{S} - \mathcal{G}\big[\mathcal{G}^{-1}[\mathbf{S}]\big] \|_{\text{Fro}}^2 \\ \text{subject to} \quad & \hat{\mathbf{A}} = \mathbf{A} \\ & \hat{\Phi}_{(k_1,h_1)} = \Phi_{(k_1,h_1)}, \cdots, \hat{\Phi}_{(k_N,h_N)} = \Phi_{(k_N,h_N)} \end{aligned} \tag{6}$$

which can be solved with an iterative projection procedure similar to what was originally proposed in [5]. Let $\mathbf{S}^{(0)}$ be an initial spectrum in which $\hat{\mathbf{A}} = \mathbf{A}$ and the values in $\hat{\mathbf{\Phi}}$ are drawn from a uniform random distribution in $0\dots 2\pi$ excepts for the values $\hat{\Phi}_{(k_1,h_1)},\dots,\hat{\Phi}_{(k_N,h_N)}$ that are set to the given $\Phi_{(k_1,h_1)},\dots,\Phi_{(k_N,h_N)}$. The iterative projection procedure is defined as:

$$\mathbf{S}^{(t+1)} = \mathcal{G}\big[\mathcal{G}^{-1}[P(\mathbf{S}^{(t)})]\big] \tag{7}$$

$$P(\mathbf{S}^{(t)})_{k,h} = \begin{cases} A_{k,h}\, e^{i\Phi_{(k,h)}}, & \text{if } (k,h) \in \mathcal{S} \\ \frac{A_{k,h}}{|S^{(t)}_{k,h}|} S^{(t)}_{k,h}, & \text{otherwise} \end{cases} \tag{8}$$

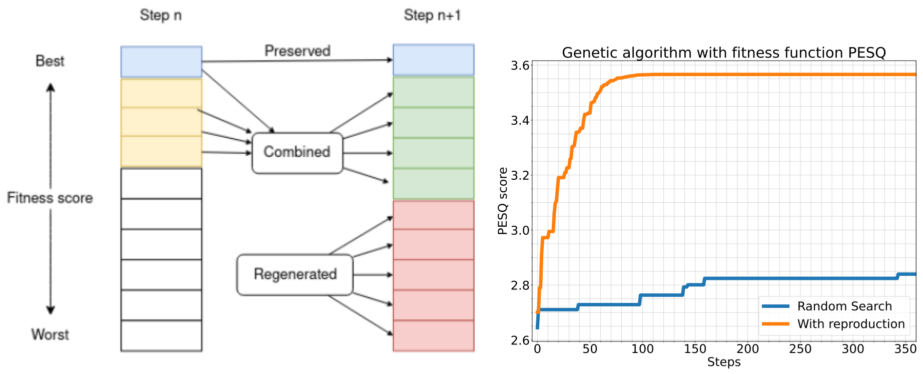

Fig. 3. Left: schema of the reproduction used for genetic subsampling (see text for details). Right: PESQ score wrt iterations for mask generation for our genetic algorithm vs. random sampling.

where P is the projection function forcing the amplitudes $|\mathbf{S}^{(t)}| = \mathbf{A}$ and the phases in $\mathcal{S}$ to $\Phi_{(k_1,h_1)}, \dots, \Phi_{(k_N,h_N)}$. The procedure let $\mathbf{S}$ converge to a consistent spectrum where amplitudes and known phases are equal to the ones that are given in input. This effectively recovers the missing phases to values that are consistent according to the overlap of subsequent time frames.

2.3 Genetic Spectrogram Subsampling

The studied compression approach works by discarding some values from the signal spectrogram. So far we gave no details on how to choose such values, but obviously this has an impact on the quality of the reconstructed signal. One simple approach can be to do a random subsampling of the input spectrogram. As we will see in the experimental section, this approach is very fast to implement but it will not consider the different energy contributions of certain frequencies in the spectrum.

In general, the subsampling operation can be solved optimally if posed as a non-linear optimization problem. Let $M \in \{0,1\}^{T \times F}$ be a binary mask defining which values to keep. Let $\mathcal{Q}(M) \in \mathbb{R}$ be a *fitness function* measuring how good is the reconstructed waveform when the mask M is applied. For example, $\mathcal{Q}$ can evaluate the PESQ (Perceptual Evaluation of Speech Quality) when the spectrogram magnitude is reconstructed with our Audio Completion Network and the phases with the modified GL as described. We can write it as:

$$\begin{aligned} \underset{M}{\text{maximize}} \quad & \mathcal{Q}(M) \\ \text{subject to} \quad & \sum M = \gamma\,, \quad M \in \{0,1\}^{T \times F} \end{aligned} \tag{9}$$

where γ is a desired *density level* (in range $0 \dots T \cdot F$), controlling the number of values to keep and therefore the amount of compression to achieve. Even if

$\mathcal{Q}$ is differentiable, the resulting problem is NP-Hard for the requirement of the mask M to be binary. To make it tractable, we can either relax the constraint of having a binary mask (i.e. using a soft mask) or use an optimization procedure based on heuristics that provides a reasonable result even with no guarantee of reaching the global optimum. In this work, we investigated the latter with an approach based on Genetic Algorithms theory.

Our Genetic Spectrogram Subsampling (GSS) starts with an initial population $\mathcal{P} = \{M_1, \ldots, M_L\}$ of valid candidate solutions. The genome of each individual in $\mathcal{P}$ is a binary mask randomly generated with the property of also being a valid solution for our optimization problem (i.e. $\sum_i M_i = \gamma \quad \forall i$). Inspired by the process of natural selection, individuals of the initial population evolve by recombining and/or mutating their genome through an iterative process spanning several generations. Individuals with a genome producing high values of fitness $\mathcal{Q}$ are more likely to survive and be selected for recombination (i.e. reproduction) across generations. After a fixed number of iterations, the best genome is returned by the algorithm. At each iteration (generation) we perform the following operations:

Selection. each element of $\mathcal{P}$ is ranked according to the fitness function $\mathcal{Q}$. This is performed by (i) applying each mask M_i to the input spectrogram, (ii) executing the Audio Completion Network to recover the missing values and, (iii) evaluating its quality against the original signal. The last step can be more or less expensive depending on the evaluation metric chosen. For example, using the MAE (Mean Absolute Error) on the spectrum amplitudes is less computationally expensive but does not take into account the perceptual quality of the final waveform. Conversely, using the PESQ metric requires computing the inverse STFT (possibly with phase recovery) but can produce better overall solutions. We use a hybrid approach here, sorting first the population in ascending order wrt. MAE. The worst 60% of the individuals are eliminated and substituted with new random individuals in the next generation. The best 40% are ranked again with PESQ and selected for either being preserved as-is or modified with a genetic operator. Specifically, the best 10% just pass through the next generation while the remaining 30% are mutated and recombined by random crossover.

Mutation and Crossover. From the 40% of the best-ranked individuals we create a number of random pairs equal to the 30% of the original population. Such random pairs of individuals are then combined with a custom genetic operator designed (heuristically) to produce a new individual with fitness higher than both parents. The mask of the new individual is obtained as follows:

$$\text{crossover}(M_i, M_j) = M_i \wedge M_j \vee \text{mutate}_{\sum M_i \wedge M_j}(M_i \oplus M_j) \tag{10}$$

where $\text{mutate}_k(M)$ is a function that randomly sets some values of M to 0 so that $\sum M = \gamma - k$. The general idea is that, after crossover, the mask of the new individual should have a 1 if both parents have a 1, or a 0 if both the parents have a 0 at each position. This is performed by the element-wise *and* operation ($\wedge$) of parent masks. Conversely, at every location where parents values disagree (i.e. the element-wise *xor* $\oplus$ is 1) the output mask can randomly have a 0 or 1

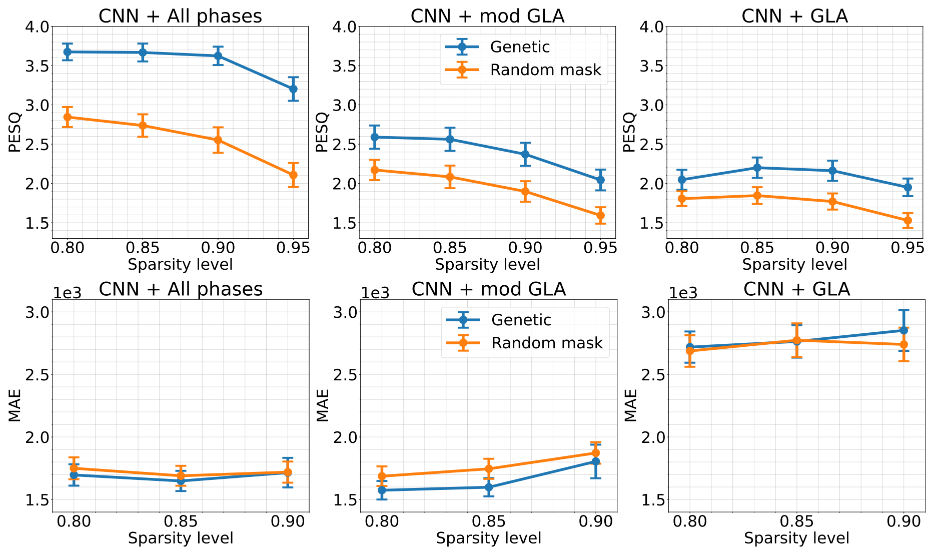

Fig. 4. Decompression quality in terms of PESQ (1st row) and MAE (2nd row) varying sparsity levels. Each column denotes a different phase reconstruction approach.

with the only requirement that the total number of ones to be equal to γ. This random flipping of values from 1 to 0 effectively represents a mutation of the new individual with respect to the parents and introduces an element of variability in the evolutionary process. In Figure3 (left) we sketched the selection process across two generations while on the right-hand side we show an example of how, albeit being a heuristic, the GSS can improve the search for optimal masks with respect to a simple random search (i.e. with no mutation and crossover).

3 Experimental Results

We tested our method on the Flickr 8k audio caption dataset [8]. To have uniform samples, we extracted audio patches with a duration of 4 seconds, avoiding parts with no signal, and we selected $10K$ audio tracks sampled at 16 KHz. Then we divided such data into training and test sets with an 80/20 ratio, obtaining $20K$ audio samples for the test set. Each audio sample was transformed with the STFT, choosing 2048 frequency bins and setting the window size to 2048 with a hop length equal to 256. In this way, we obtained complex images of size 1025×294. As for the absolute values, we converted the spectrogram to a log-power scale. The audio completion network was trained by applying uniform random masks to the training set, randomly selecting the sparsity levels between 80% to 95%. The training process was performed with Adam optimizer with an adaptive learning initialized to 10^{-4}, and run up to convergence.

In our study we explore the phases reconstruction step with different methods: (i) keep no phases and compute them with GLA, (ii) keep only the phases

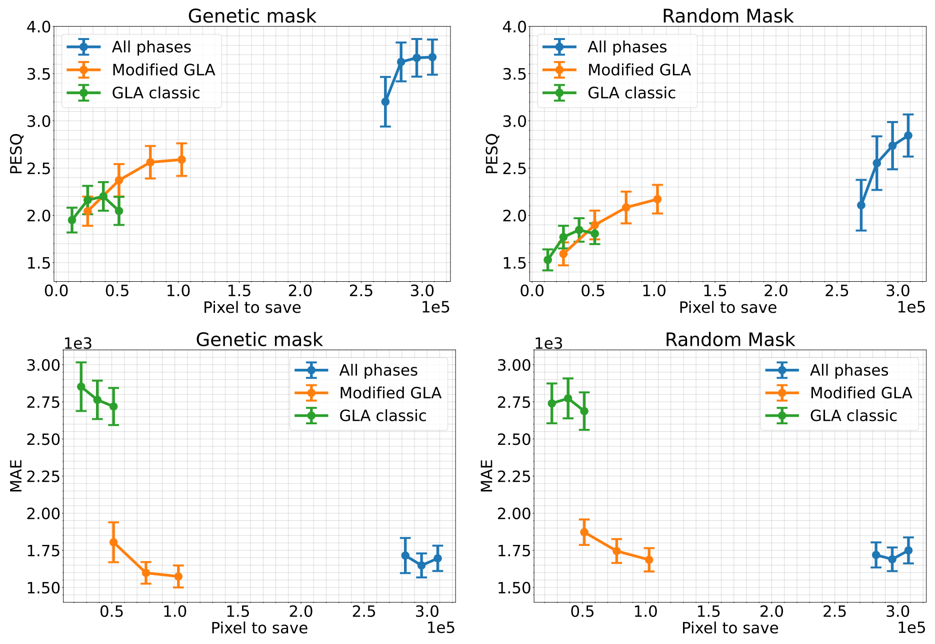

Fig. 5. Quality of reconstruction wrt values to be stored for different approaches and subsampling methods.

corresponding to the selected power values and use the improved version of GLA to ensure coherency, and (iii) keep all the input phases. Therefore, we first analyse the behaviour of these three approaches when reconstructing audio at different sparsity levels. Figure 4 shows the decompression quality for different phase reconstructions: all phases (1st column), masked phases (2nd column), and no phases (3rd column). We show PESQ (1st row) and MAE (Mean Absolute Error, 2nd row) metrics computed on all test set data with different sparsity levels up to 0.95. As expected, the quality decreases as the sparsity level raises, and keeping all the phases gives the best quality both in terms of perceptual quality and MAE. Observing the PESQ values, we can see that the modified GLA improves the quality up to PESQ $= 2.5$, offering a better outcome with respect to the classical GLA (no phases), meaning that our approach is effective while requiring less space. The modified GLA works well also when looking at the MAE, being comparable with the "all phases" method. This is also an indication of the effectiveness of our proposed modification to GLA when also the compressed phases are sparse. The second test we performed is meant to assess the effectiveness of the proposed genetic algorithm to perform the subsampling during compression. In particular, we show the reconstruction results for a random uniform sampling versus the genetic subsampling. In order to evaluate the compression step together with the three phase reconstructions alternatives, we include such results in Figure 4 as different curves on the already discussed plots.

We can see that the genetic algorithm always computes a better mask for the given sparsity levels, in particular for the perceptual quality, reaching values around 3.5 for all the phases and 2.5 for the sparse phases (with modified GLA). As a last experiment, we compared the memory space needed to store the compressed data for each different approach. Note that we do not plot the bitrate, but rather we prefer to show the number of values to be saved. This is because the final amount of bytes depends on a number of external factors such as the STFT parameters, the quantization level, and the data structure that is actually employed to store the sparse matrix. In this way we only show the number of values, discarding other possible choices. Figure 5 compares the number of stored values for all three methods (all phases, modified GLA and classic GLA) and for the two subsampling approaches: genetic (left) and uniform random (right). Again, the genetic method offers better quality for the same number of values, and the "all phases" approach (blue curve) offers the best results, but at the cost of having almost 3 times more memory consumption with respect to the others.

4 Conclusions

In this paper we proposed a first study involving the adoption of CNNs to perform audio compression as an image-reconstruction task. We presented a decompression method that takes as input a sparse spectrogram and reconstructs the dense image via a data-driven model. Moreover, we proposed an improved version of the well-known Griffin-Lim algorithm to effectively recover the original waveform with just a few available phases. Then, we proposed a novel sub-sampling approach based on a genetic algorithm. Experimental results show the validity of both our compression and decompression methods in terms of MAE and quality perception. The genetic approach effectively selects the values to be compressed, while the decompression based on a CNN model and on a constrained implementation of GLA offers quantitative significant results.

References

1. Arık, S.Ö., Jun, H., Diamos, G.: Fast spectrogram inversion using multi-head convolutional neural networks. IEEE Signal Processing Letters **26**(1), 94–98 (2018)
2. Brandenburg, K., Stoll, G.: Iso/mpeg-1 audio: A generic standard for coding of high-quality digital audio. Journal of the Audio Engineering Society (1994)
3. Gasparetto, A., Ressi, D., Bergamasco, F., Pistellato, M., Cosmo, L., Boschetti, M., Ursella, E., Albarelli, A.: Cross-dataset data augmentation for convolutional neural networks training. vol. 2018-August, p. 910 – 915 (2018). https://doi.org/10.1109/ICPR.2018.8545812
4. Ghido, F., Tabus, I.: Sparse modeling for lossless audio compression. IEEE Transactions on Audio, Speech, and Language Processing **21**(1), 14–28 (2012)
5. Griffin, D., Lim, J.: Signal estimation from modified short-time fourier transform. IEEE Transactions on acoustics, speech, and signal processing (1984)
6. Hans, M., Schafer, R.W.: Lossless compression of digital audio. IEEE Signal processing magazine **18**(4), 21–32 (2001)

7. Hanzo, L., Somerville, F.C.A., Woodard, J.: Voice and audio compression for wireless communications. John Wiley & Sons (2008)
8. Harwath, D., Glass, J.: Deep multimodal semantic embeddings for speech and images. In: 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU). pp. 237–244. IEEE (2015)
9. Huang, Z., Fan, J., Cheng, S., Yi, S., Wang, X., Li, H.: Hms-net: Hierarchical multiscale sparsity-invariant network for sparse depth completion. IEEE Transactions on Image Processing **29**, 3429–3441 (2019)
10. Jaritz, M., De Charette, R., Wirbel, E., Perrotton, X., Nashashibi, F.: Sparse and dense data with cnns: Depth completion and semantic segmentation. In: 2018 International Conference on 3D Vision (3DV). pp. 52–60. IEEE (2018)
11. Kanade, J., Sivakumar, B.: A literature survey on psychoacoustic models and wavelets in audio compression. International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE) (2014)
12. Kankanahalli, S.: End-to-end optimized speech coding with deep neural networks. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing
13. Kleijn, W.B., Storus, A., Chinen, M., Denton, T., Lim, F.S., Luebs, A., Skoglund, J., Yeh, H.: Generative speech coding with predictive variance regularization. In: IEEE International Conference on Acoustics, Speech and Signal Processing
14. Lim, T.Y., Yeh, R.A., Xu, Y., Do, M.N., Hasegawa-Johnson, M.: Time-frequency networks for audio super-resolution. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 646–650. IEEE (2018)
15. Ma, F., Karaman, S.: Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In: 2018 IEEE international conference on robotics and automation (ICRA). pp. 4796–4803. IEEE (2018)
16. Morishima, S., Harashima, H., Katayama, Y.: Speech coding based on a multi-layer neural network. In: IEEE International Conference on Communications
17. Pistellato, M., Albarelli, A., Bergamasco, F., Torsello, A.: Robust joint selection of camera orientations and feature projections over multiple views. vol. 0, p. 3703 – 3708 (2016). https://doi.org/10.1109/ICPR.2016.7900210
18. Pistellato, M., Bergamasco, F., Albarelli, A., Torsello, A.: Dynamic optimal path selection for 3d triangulation with multiple cameras. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **9279**, 468 – 479 (2015). https://doi.org/10.1007/978-3-319-23231-7_42
19. Pistellato, M., Bergamasco, F., Albarelli, A., Torsello, A.: Robust cylinder estimation in point clouds from pairwise axes similarities. p. 640 – 647 (2019). https://doi.org/10.5220/0007401706400647
20. Pistellato, M., Bergamasco, F., Fatima, T., Torsello, A.: Deep demosaicing for polarimetric filter array cameras. IEEE Transactions on Image Processing **31**, 2017 – 2026 (2022). https://doi.org/10.1109/TIP.2022.3150296
21. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: 2017 international conference on 3D Vision (3DV) (2017)
22. Wang, W., Neumann, U.: Depth-aware cnn for rgb-d segmentation. In: Proceedings of the European conference on computer vision (ECCV). pp. 135–150 (2018)
23. Williamson, D.S., Wang, D.: Time-frequency masking in the complex domain for speech dereverberation and denoising. IEEE/ACM transactions on audio, speech, and language processing **25**(7), 1492–1501 (2017)
24. Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., Tagliasacchi, M.: Soundstream: An end-to-end neural audio codec. IEEE/ACM Transactions on Audio, Speech, and Language Processing **30**, 495–507 (2021)