Deep Learning Audio Super Resolution

Thomas Retzloff
ID: 301368976

I. Introduction

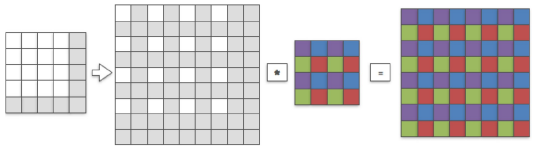
The audio super resolution task is aimed at enhancing the resolution of an audio signal. Formally stated, given an audio signal $x = \{x_{1/R_1}, \ldots, x_{R_1 T_1/R_1}\}$ with sampling rate R_1 and for time T_1, we wish to reconstruct a higher resolution signal $y = \{y_{1/R_2}, \ldots, y_{R_2 T_2/R_2}\}$ with sampling rate $R_2 > R_1$. Typical upsampling ratios $r = R_2/R_1$ are 2, 4, and 8.

Potential applications could be used for low resolution voice signals such as in telephones. In this project, deep learning methods are applied to the audio super resolution task for both speech and music data. We train existing audio super resolution models and compare the results with new ones. We specifically propose two new models: a UNet and WGAN with feature based losses calculated from pre-trained networks.

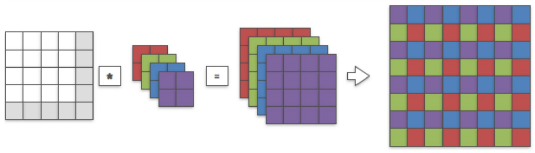
II. Background and Related Work

A. Autoencoders

Many deep learning approaches to audio super resolution modify existing image super resolution methods. One such image method is to use an autoencoder style architecture, as used in W. Shi et al. 2016. This was the first experiment to use a series of upsampling layers in the last few layers of a network. They also implemented a novel upsampling layer: the subpixel convolution, which is similar to a deconvolution layer although implemented differently. The difference between the two operations is illustrated in figures 1a and 1b.



(a) Deconvolution illustration, from Wenzhe Shi et al. 2016



(b) Subpixel convolution illustration, from Wenzhe Shi et al. 2016

Essentially, the subpixel layer reshapes a tensor of dimension $H \times W \times rC$ into a tensor of shape $rH \times rW \times C$ without having to manipulate the input. If we denote the subpixel operation by $\mathcal{PS}$ (phase shift operator), then evaluating it on a tensor T gives

$$\mathcal{PS}(T)_{x,y,c} = T_{\lfloor x/r \rfloor, \lfloor y/r \rfloor, C \cdot r \cdot mod(y,r) + C \cdot mod(x,r) + c}.$$

This method is claimed to be more efficient than the deconvolution layer (during training) as the ground truth high resolution images can be reshaped to the last layer before the shuffling one so that the pre-shuffled filters are compared rather than the post-shuffled images. The loss function is the MSE between ground truth high resolution images and generated images.

In Kuleshov, Enam, and Ermon 2017, a network with motivation from W. Shi et al. 2016 is adapted to the audio domain. The network uses B down-sampling and B up-sampling with a bottleneck layer in between and additive/stacking connections; the network is depicted in figure 2.

The network runs on raw signal and uses 1D convolutions. It implements a similar 1D subpixel operation. With this said, the paper upsamples the source signal to the desired sample rate with cubic upscaling and then pushes it through the network, which is different from the subpixel method described in W. Shi et al. 2016 and Wenzhe Shi et al. 2016.

B. Generative Adverserial Networks

Another deep learning approach to image super resolution is to use a GAN (Goodfellow et al. 2014). In a GAN architecture, a generator G is learned to map vectors from a latent and low-dimensional space $\mathcal{Z}$ to a higher dimensional space $\mathcal{X}$ (e.g. a space of natural images or high resolution audio) while a discriminator is trained to detect if its input is a generated sample or not. The low-dimensional vectors are assumed to have been sampled from a prior $P_{\mathcal{Z}}$. If G is parameterized by θ_G and D is parameterized by θ_D, then the GAN problem is given by

$$\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{x \sim P_x} \left[\log D(x; \theta_D)\right] + \mathbb{E}_{z \sim P_z} \left[\log(1 - D(G(z, \theta_G)))\right].$$

Here the generator is supposed to learn how to "fool" the discriminator, which in turn learns to catch the generator red handed. This is designed to create very realistic data samples and such methods have been shown to work as in DCGAN Radford, Metz, and Chintala 2015 and SRGAN Ledig et al. 2016. Although DCGAN inspires SRGAN and initially proposed a set of guidlines for deep convolutional GANs, we explain SRGAN instead due to its greater relevance to the project.

The SRGAN uses a similar generator model that contains the subpixel operation from W. Shi et al. 2016 as well as a convolution based discriminator. The paper also proposes a feature based loss function that is added onto the MSE content loss for the generator. The feature loss has been used on numerous occasions in deep learning such as in style transfer (Gatys, Ecker, and Bethge 2016, Dosovitskiy and Brox 2016). It is computed by separately passing the high resolution ground truth and generated output through a pre-trained feature extractor and computing a loss (e.g. MSE)

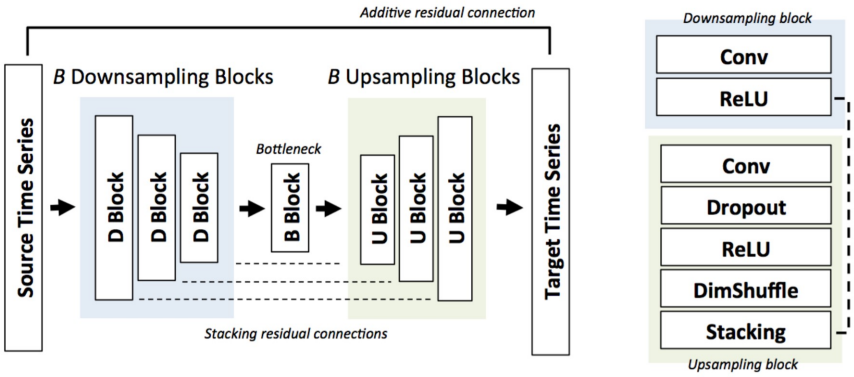


Fig. 2: UNet architecture from Kuleshov, Enam, and Ermon 2017.

between the outputs. In Ledig et al. 2016 a VGG network (Simonyan and Zisserman 2014) pre-trained on ImageNet (Deng et al. 2009) is used to compute the feature maps for the content loss.

In Kim and Sathe 2019, a similar approach is applied to raw audio input. The authors use the same generator network as from Kuleshov, Enam, and Ermon 2017, although convolution filter dimensions are different. Although they achieve comparatively better results than in Kuleshov, Enam, and Ermon 2017, the authors do not mention how they pre-trained the feature extractor nor give any code for clarification. The feature loss network is also similar to the generator used, although the feature loss network does not contain any additive or skip connections.

C. Wasserstein GANs

Although the use of GANs has been successful in generative modeling, they have been shown to be unstable (Arjovsky and Bottou 2017) and can thus be difficult to train. Since it can be shown (Goodfellow et al. 2014) that solving the GAN optimization problem is equivalent to minimizing the Jensen-Shannon divergence between P_x and P_z, Arjovsky, Chintala, and Bottou 2017 proposes instead to minimize the **earth mover** distance, or Wasserstein-1 distance, between the distributions. The Wasserstein distance between P_x and P_z is given by

$$W(P_x, P_z) = \inf_{\gamma \in \Pi(P_x, P_z)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|]$$

where $\Pi(P_x, P_z)$ is the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively P_x and P_z. One might see that the joint distribution γ gives the needed amount of mass to be transported from x to y in order to transform the distributions P_x into the distribution P_z. Arjovsky, Chintala, and Bottou 2017 shows that the Jensen-Shannon divergence can be discontinuous for continuous generators G while the Wasserstein distance is in fact continuous. The authors further show that the Wasserstein distance can be expressed as a more tractable form than its original:

$$W(P_x, P_z) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim P_x}[f(x)] - \mathbb{E}_{z\sim_z}[f(z)]$$

where $f : \mathcal{X} \to \mathbb{R}$ is the family of 1-Lipschitz functions. This finally leads to the Wasserstein-GAN problem

$$\max_w \mathbb{E}_{x\sim P_x}[D(x; w)] - \mathbb{E}_{z\sim P_z}[D(G(z; \theta); w)].$$

Instead of training to tell the difference between real and generated content, the job of D (which is known as the *critic*) is now to learn a 1-Lipschitz function that helps to compute the Wasserstein distance between the real and generated input. The idea here is that D will respond directly to the quality of the generated outputs and will assign large values to real content and small values to fake content.

Thankfully, only minor details need to be changed in a GAN architecture to implement the WGAN: one must use a linear activation function in the output layer of D and also clip the critic's weights to be constrained in range after each gradient descent mini batch update to ensure Lipschitz continuity. Additionally, in each training step, the critic is trained for n_{critic} steps and then the generator takes its training step.

The WGAN is implemented in the WaveGan model (Donahue, McAuley, and Puckette 2018). WaveGan bases its architecture on DCGAN (Radford, Metz, and Chintala 2015), which uses a transposed convolution layer instead of the subpixel convolution layer (both are identical techniques (Wenzhe Shi et al. 2016)). In DCGAN, it is seen that checkerboard type artifacts are common in generated images, making it easier for the discriminator to reject generated data. In audio data this corresponds to artifact frequencies to occur at specific phases. To fix this, the WaveGAN authors propose a *phase shuffle* operation that randomly perturbs the phase of each of the layer's activations of the critic network by $-n$ to n samples before inputting to the next layer.

III. Method

The models in this project are either taken from or inspired by Kuleshov, Enam, and Ermon 2017, Kim and Sathe 2019, and Donahue, McAuley, and Puckette 2018. The first model is the autoencoder style network from Kuleshov, Enam, and Ermon 2017, the second is a normal GAN that uses the UNet generator from Kuleshov, Enam, and Ermon 2017 and WaveGAN's phase shuffle discriminator, and a WGAN similar

to the GAN. The discriminator and critic models are simply five-layer convolutional networks (four convolutions) that implement the phase shuffle operation previously described between layers.

Each model is also trained separately with an additional feature based loss function, giving six models in total. We use the VGGish model (Hershey et al. 2017) for the feature loss network. VGGish is a pre-trained on AudioSet (Gemmeke et al. 2017), which contains over 1 million 10 second excerpts.

The models are trained on ADAM (Kingma and Ba 2014) with a learning rate of 10^{-4} for 50 epochs. Here upsampling ratios of $r = 2, 4$ are considered. The loss for the UNet is simply the L_2 loss

$$L_2(x_{hr}, x_{pred}) = \frac{1}{W}\sum_{i=1}^{W} \|x_{hr_i} - x_{pred_i}\|_2^2.$$

For the GAN, the generator's loss L_G is given by

$$L_G(x_{hr}, x_{pred}) = L_2(x_{hr}, x_{pred}) + \lambda_{adv} L_{adv}(x_{hr}, x_{pred})$$

where the adversarial loss term L_{adv} is given by

$$L_{adv}(x_{hr}, x_{pred}) = -\log(D(x_{pred}).$$

Here we use $\lambda_{adv} = .01$. The discriminator's loss L_D is given by

$$L_D(x_{hr}, x_{pred}) = -\log(D(x_{hr}) + \log(D(1 - x_{pred}).$$

The WGAN's generator loss is defined similarly as in the GAN, although the critic loss L_c is given by

$$L_c(x_{hr}, x_{pred}) = \frac{1}{W}\sum_{i=1}^{W} \left(C(x_{hr}) \odot C(x_{pred})\right)_i$$

where $\odot$ represents component-wise multiplication. For the content loss, we pass the generated output and high resolution ground truth through the pre-trained model and take the L_2 loss between them, finally add it to the generator loss term. The models are lastly compared to a baseline interpolation method.

IV. Data

The models in this project were trained on speech and music data. The speech data was taken from the VCTK data corpus (Veaux, Yamagishi, and Macdonald 2016), which contains 44 hours of English speech audio from 108 different speakers. The music data was taken from the Free Music Archive (FMA) (Defferrard et al. 2016). In its full form, the FMA dataset contains 1TB of songs from a myriad of genres - though the quality and musicality of this data is highly variable. A subset of 25,000 pre-selected (by the authors of Defferrard et al. 2016) tracks from 16 unbalanced genres was used.

A training set was created from each of the two datasets. To create a training set, 50000 half-second clips were randomly selected from the respective datasets. The ground truth high resolution sampling rate considered was 16KHz. No further pre-processing was performed.

V. Comparison Metrics

In this project, we compare models using the signal-to-noise ratio (SNR) and log spectral distance (LSD), which are used as in Kuleshov, Enam, and Ermon 2017. The SNR between the ground truth high resolution audio signal y_{hr} and the generated audio signal y_{pred} is defined as

$$SNR(y_{pred}, y_{hr}) = 10\log\frac{\|y_{hr}\|_2^2}{\|y_{pred} - y_{hr}\|_2^2}.$$

SNR is defined to be the ratio of signal power to the background noise power. The LSD is defined as

$$LSD(y_{pred}, y_{hr})\frac{1}{L}\sum_{\ell=1}^{L}\sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(Y(\ell, l) - \hat{Y}(\ell, k)\right)^2}$$

where Y and $\hat{Y}$ are the log-spectral power magnitudes of y_{hr} and y_{pred}, respectively, and ℓ and k are respective indexes for frames and frequencies. As the name suggests, the LSD between two signals is a measure of distance between them.

VI. Results

The models were trained for two days. Plots for the validation loss of the during training on the VCTK dataset are given in figure 4 in the appendix. Furthermore, results for the models evaluated on the SNR and LSD metrics are given in figure 3.

It should first be noted that Kuleshov, Enam, and Ermon 2017 reported a SNR of 20.3 and LSD of 4.5 for $r = 2$ and a SNR of 14.8 and LSD of 8.2 for $r = 4$. For speech data, the best model trained was the WGAN without content loss in training. Given the discrepancy between this project's results for the interpolation method and that in Kuleshov, Enam, and Ermon 2017, we see a 6.7% increase from the interpolation method to our WGAN while only a 3.9% and 5.4% increase for the UNet in Kuleshov, Enam, and Ermon 2017 and GAN in Kim and Sathe 2019, respectively. This is not the case for LSD, where the interpolation method seems to do as well or slightly better than our trained models.

For the FMA dataset, the model trained extremely slowly. As a result of this, only one epoch was completed in the time for training. Since the loss did not decrease much for the speech data, the music data results could still be relevant. The results in figure 3b suggest that the models do worse than the baseline, more training is needed.

Spectrograms on an audio clip of data from the respective dataset are given in figures 5, 6, 7, and 8 in the appendix.

VII. Discussion

While the models trained on music data still need to heavily be improved (as in training for at least 50 epochs), the models trained on speech data did comparatively well. The models trained with an included content loss performed slightly worse than without it. This could partly have happened as the VGGish model was originally trained to detect a myriad of different types of sounds. If the VGGish model was trained on more similar data, such as speech data for the VCTK training, then the results might fair better as the features would be much

Ratio	Metric	Spline	UNet	UNet-C	GAN	GAN-C	WGAN	WGAN-C	UNet-L	GAN-L
$r = 2$	SNR	13.25	13.74	13.12	14.06	13.23	14.14	13.24	21.1	21.40
	LSD	2.26	2.46	2.40	2.28	2.46	2.35	2.46	3.2	1.63
$r = 4$	SNR	11.33	11.40	11.42	11.30	10.02	11.97	11.13	17.1	17.72
	LSD	3.18	3.35	3.16	3.22	3.23	3.15	3.21	3.6	1.92

(a) Comparison of models trained on VCTK in this project with the UNet (denoted UNet-L) from Kuleshov, Enam, and Ermon 2017, as well as the GAN (denoted GAN-L) from Kim and Sathe 2019 (data for these two models was taken from their respective papers). The baseline spline is given in comparison. Models marked with a "-C" denote that the content loss was incorporated during training.

Ratio	Metric	Spline	UNet	UNet-C	GAN	GAN-C	WGAN	WGAN-C	UNet-L	GAN-L
$r = 2$	SNR	12.37	10.45	10.56	11.69	11.46	11.73	11.67	n/a	n/a
	LSD	2.32	2.91	2.92	2.78	2.79	2.76	2.77	n/a	n/a
$r = 4$	SNR	8.48	8.42	7.84	8.22	8.19	8.55	8.46	n/a	n/a
	LSD	3.24	3.37	3.44	3.39	3.38	3.36	3.34	n/a	n/a

(b) Comparison of models trained on FMA in this project. The baseline spline is given in comparison. Models marked with a "-C" denote that the content loss was incorporated during training.

Fig. 3: Testing results for both datasets.

more relevant. In the trials here, the model that consistently performed the best was the WGAN model without content loss network - a model proposed in this project. This matches well with the theory that they perform better in training than GANs, which in turn performed better than the UNet model.

The models trained on the VCTK dataset performed better than the baseline in terms of SNR, although not on the LSD. When viewing the spectrograms, one notices that the methods are able to gain back the higher frequency bands that were lost in lowering the signal resolution. This is not really the case for the baseline interpolation method, which appears to leave most of the higher frequency bands untouched.

In listening to the audio (a pre-computed example for the VCTK trained WGAN model can be found in the readme notebook), one notices that the generated sound is more clear/crisp than the low resolution input. Although this is the case, the generated output does not sound exactly like the ground truth high resolution. In comparison, the generated output sounds more muffled and seems to have a quality in between the low resolution and ground truth high resolution audio clips.

VIII. Conclusion

In this project, multiple audio super resolution models were trained on speech and music data. The model trained on speech data performed better than the proposed baseline. In moving forward and adding improvements, larger networks (such as $B = 8$) could be trained as well as for higher upsampling ratios. A class of models could be defined for different upsampling ratios, datatypes, and target sampling rates could be defined. A further consideration might be to use models for the feature based loss network that were pre-trained on data relevant to the training dataset for audio super resolution training.

References

[AB17] Martín Arjovsky and Léon Bottou. "Towards Principled Methods for Training Generative Adversarial Networks". In: *ArXiv* abs/1701.04862 (2017).

[ACB17] Martín Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein GAN". In: *ArXiv* abs/1701.07875 (2017).

[DB16] Alexey Dosovitskiy and Thomas Brox. "Generating images with perceptual similarity metrics based on deep networks". In: *Advances in neural information processing systems*. 2016, pp. 658–666.

[Def+16] Michaël Defferrard et al. "Fma: A dataset for music analysis". In: *arXiv preprint arXiv:1612.01840* (2016).

[Den+09] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[DMP18] Chris Donahue, Julian McAuley, and Miller Puckette. "Adversarial audio synthesis". In: *arXiv preprint arXiv:1802.04208* (2018).

[GEB16] L. A. Gatys, A. S. Ecker, and M. Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2414–2423.

[Gem+17] Jort F. Gemmeke et al. "Audio Set: An ontology and human-labeled dataset for audio events". In: *Proc. IEEE ICASSP 2017*. New Orleans, LA, 2017.

[Goo+14] Ian J. Goodfellow et al. "Generative Adversarial Networks". In: *ArXiv* abs/1406.2661 (2014).

[Her+17] Shawn Hershey et al. "CNN Architectures for Large-Scale Audio Classification". In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017. URL: https://arxiv.org/abs/1609.09430.

[KB14] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2014).

[KEE17] Volodymyr Kuleshov, S Zayd Enam, and Stefano Ermon. "Audio super resolution using neural

networks". In: *arXiv preprint arXiv:1708.00853* (2017).

[KS19] Sung Kim and Visvesh Sathe. "Bandwidth Extension on Raw Audio via Generative Adversarial Networks". In: *ArXiv* abs/1903.09027 (2019).

[Led+16] Christian Ledig et al. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 105–114.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *CoRR* abs/1511.06434 (2015).

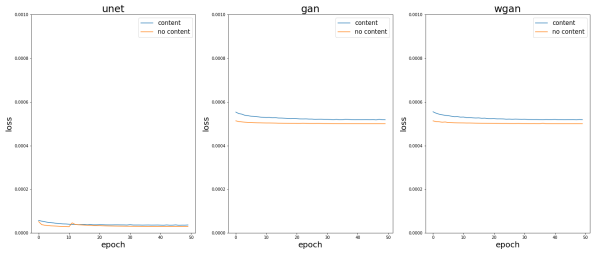
[Shi+16a] W. Shi et al. "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 1874–1883.

[Shi+16b] Wenzhe Shi et al. "Is the deconvolution layer the same as a convolutional layer?" In: *CoRR* abs/1609.07009 (2016). arXiv: 1609.07009. URL: http://arxiv.org/abs/1609.07009.

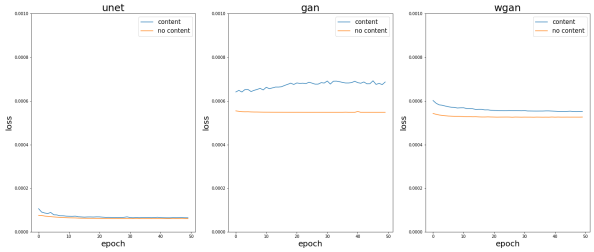
[SZ14] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014).

[VYM16] Christophe Veaux, Junichi Yamagishi, and Kirsten Macdonald. "SUPERSEDED - CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit". In: 2016.

APPENDIX



(a) Validation loss across 50 epochs for the models trained on the VCTK dataset for upsampling ratio $r = 2$



(b) Validation loss across 50 epochs for the models trained on the VCTK dataset for upsampling ratio $r = 4$

Fig. 4: Plots for the validation loss across training epochs.

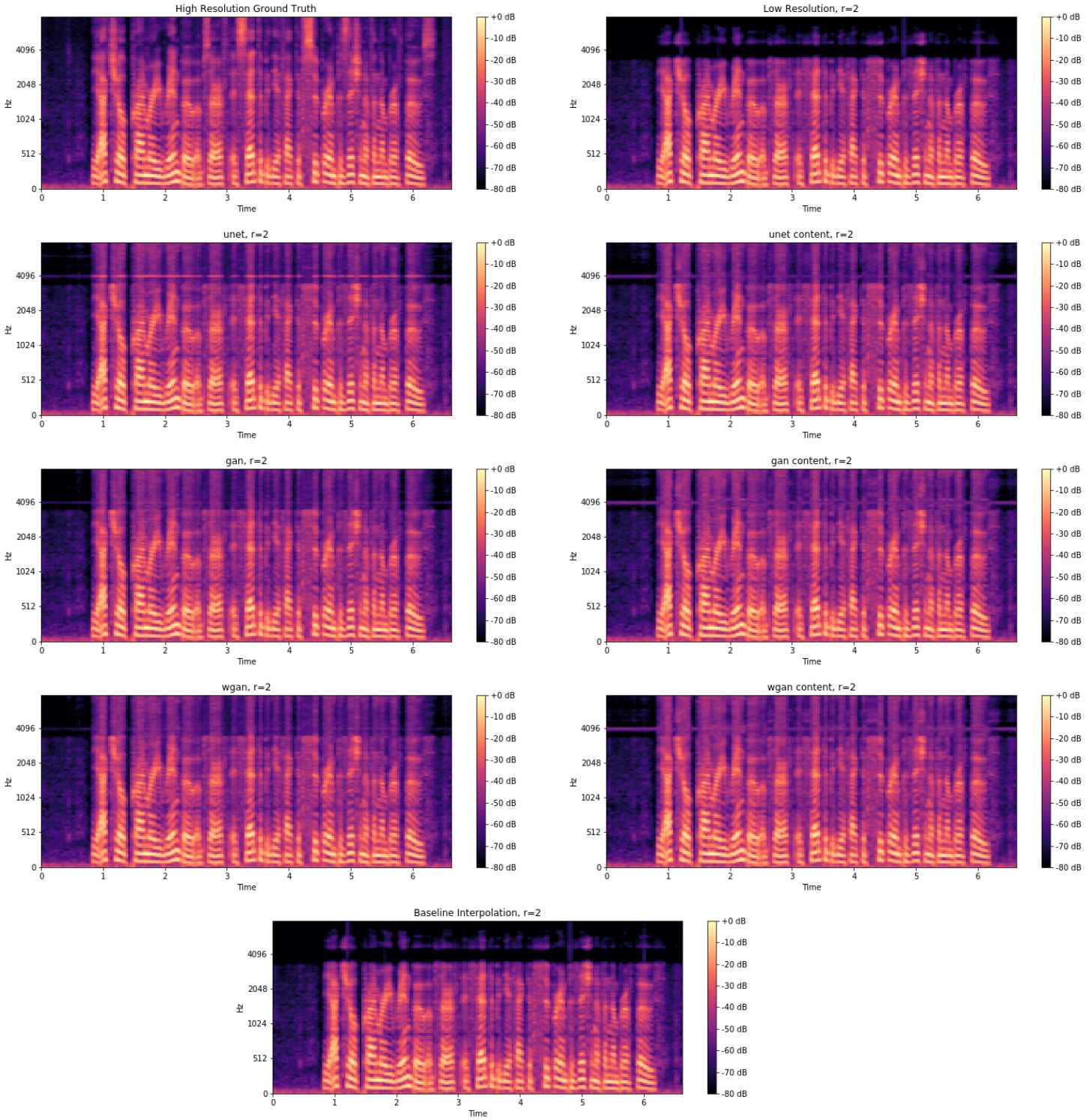


Fig. 5: Spectrograms for models trained on the VCTK dataset with upsampling ratio $r = 2$

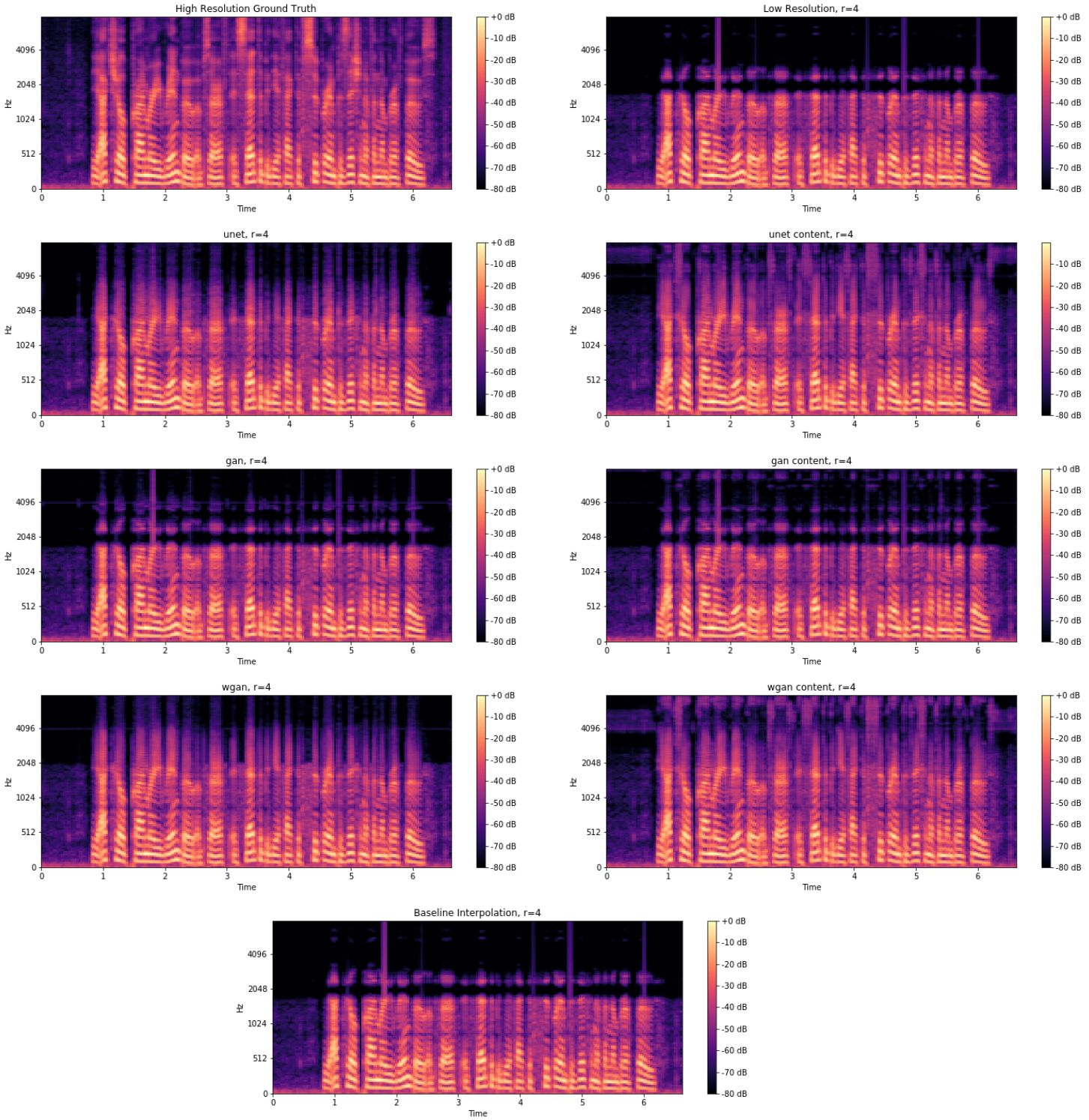


Fig. 6: Spectrograms for models trained on the VCTK dataset with upsampling ratio $r = 4$

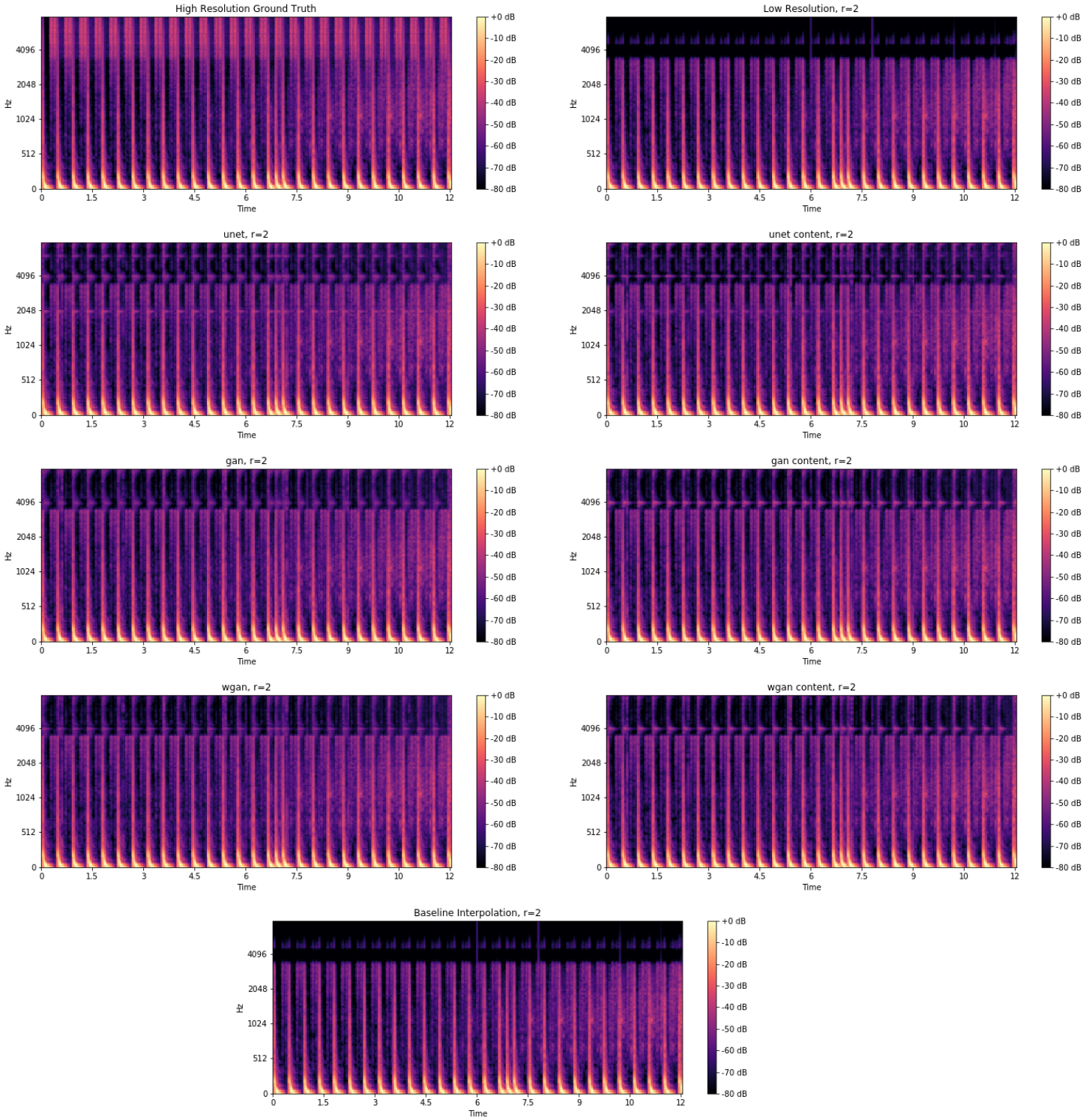


Fig. 7: Spectrograms for models trained on the FMA dataset with upsampling ratio $r = 2$

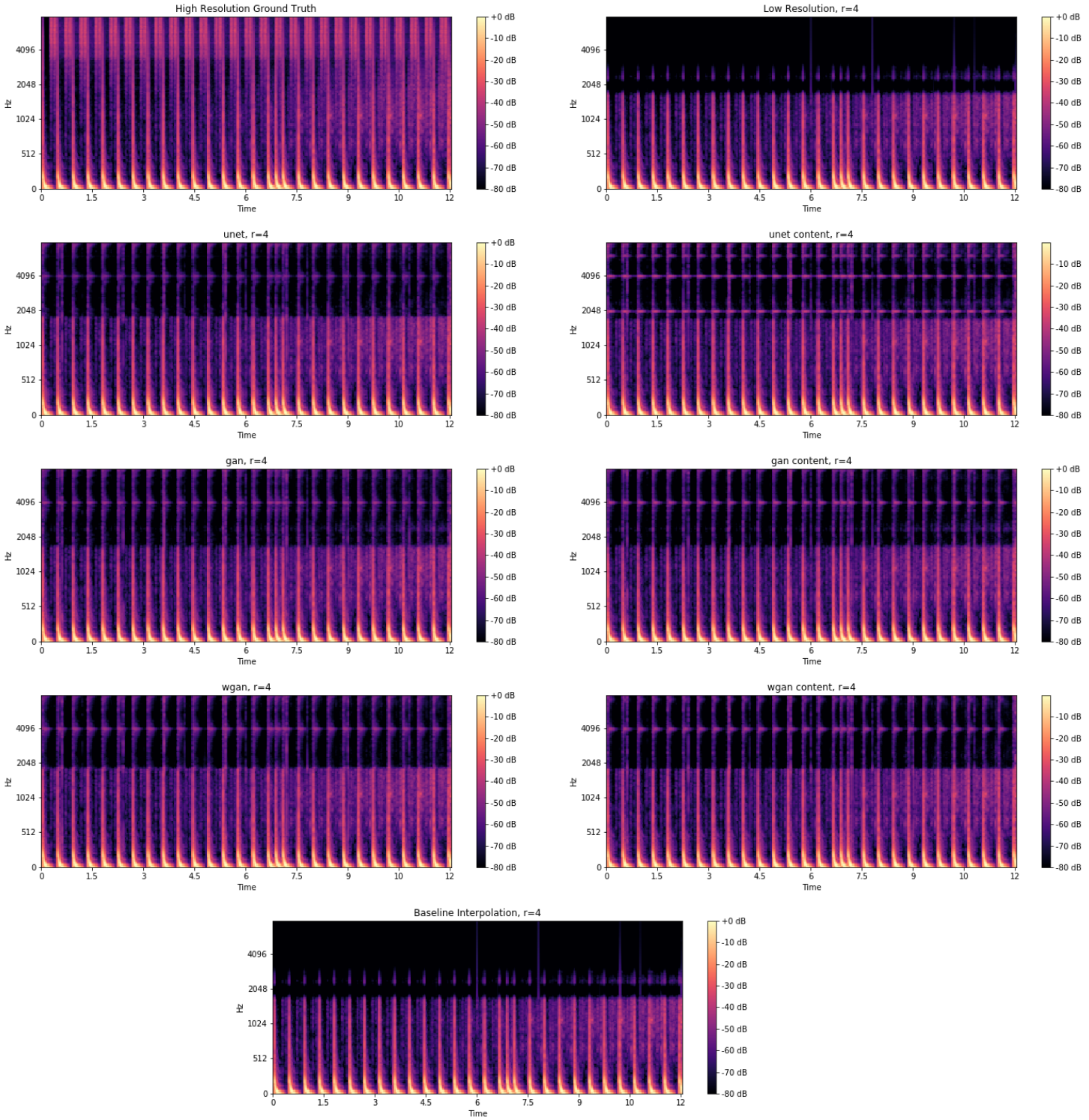


Fig. 8: Spectrograms for models trained on the FMA dataset with upsampling ratio $r = 4$