

VoiceFixer

TFGAN Vocoder - Training - Discriminator Losses

- Loss Function: $\mathbb{L}_{syn} = L^T + L^F + \lambda_1 L^D$
- Discriminator Losses:

- $D(\hat{s}) = D^{T-sub}(\hat{s}) + D^F(\hat{s}) + \sum_{r=1}^4 D_r^T(\hat{s})$
- $L^D = \min_G \max_D (\mathbb{E}_s(\log(D(s))) + \mathbb{E}_{\hat{s}}(\log(1 - D(\hat{s}))))$.

Table.5 The architecture of time domain discriminator

T-discriminator
Conv1d(1, 128, ks=16), LeakyRelu(0.2)
Conv1d(128, 128, ks=41, stride=4, padding=20, groups=8), LeakyRelu(0.2)
Conv1d(128, 128, ks=41, stride=4, padding=20, groups=16), LeakyRelu(0.2)
Conv1d(128, 128, ks=41, stride=4, padding=20, groups=32), LeakyRelu(0.2)
Conv1d(128, 1, ks=3, stride=1, padding=1), LeakyRelu(0.2)

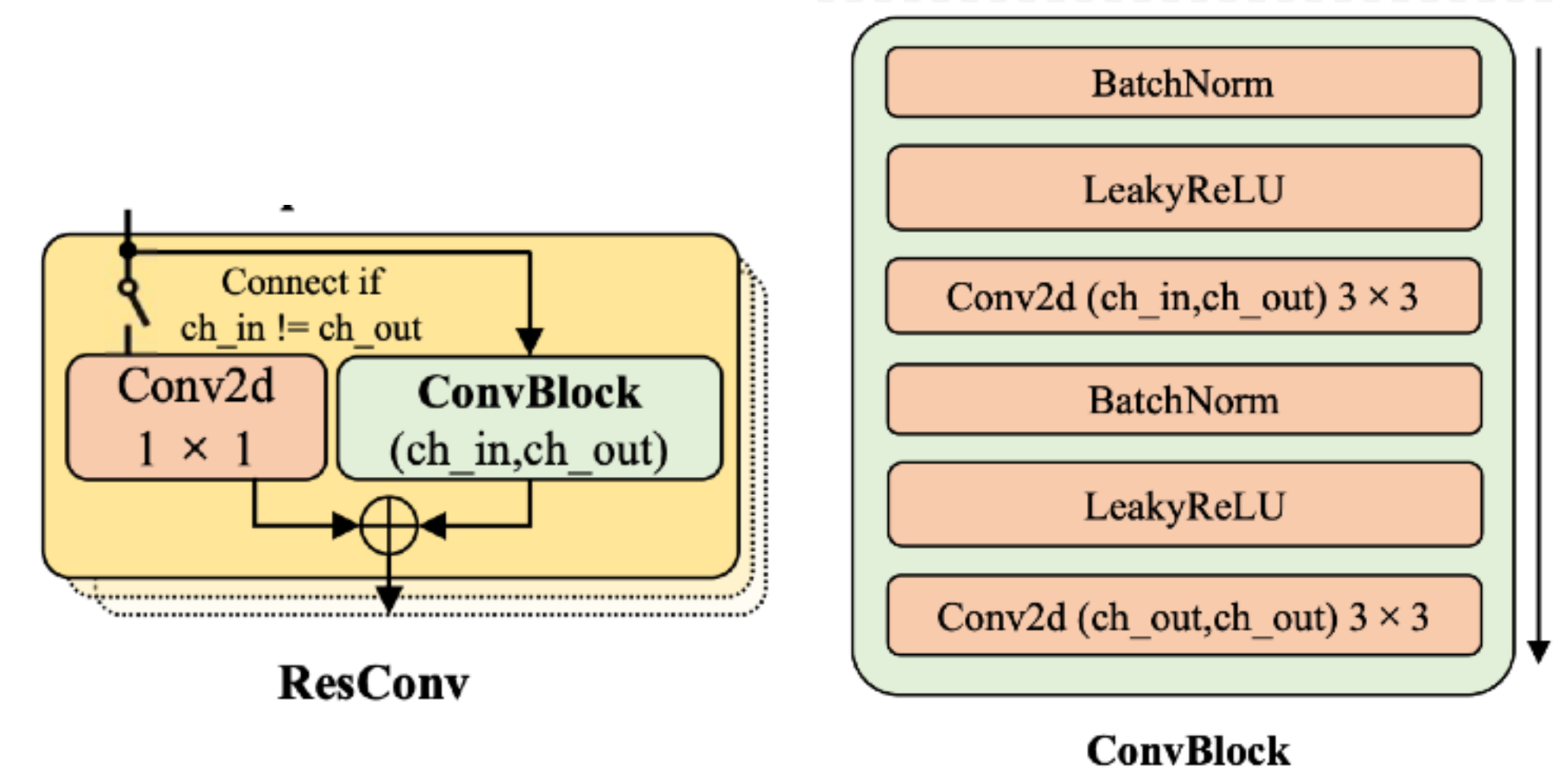


Table.6 The architecture of frequency domain discriminator

F-discriminator
Conv2d(1,32,kernal_size=(3,3))
ResConv(32, 32, stride=1,kernal_size=(3,3))
ResConv(32, 32, stride=1,kernal_size=(3,3))
ResConv(32, 64, stride=2,kernal_size=(3,3))
ResConv(64, 64, stride=1,kernal_size=(3,3))
ResConv(64, 32, stride=2,kernal_size=(3,3))
ResConv(32, 32, stride=1,kernal_size=(3,3))
ResConv(32, 32, stride=2,kernal_size=(3,3))
ResConv(32, 32, stride=1,kernal_size=(3,3))

Experiments

Training data distortion simulation

Algorithm 1: Add high quality speech x with random distortions

In: Speech $x \leftarrow S(\mathcal{X})$; Noise $n \leftarrow S(\mathcal{N})$; Room impulse response $r \leftarrow S(\mathcal{R})$

```
1   $x' = x$ 
2  with  $p_1$  probability:
3       $x' = x * r$ ;                                /* Convolute with RIR filter */
4  with  $p_2$  probability:
5       $\theta = S(\mathcal{U}(\Theta_{low}, \Theta_{high}))$ ;        /* Choose clipping ratio */
6       $x' = \max(\min(x', \theta), -\theta)$ ;            /* Hard clipping */
7  with  $p_3$  probability:
8       $t = randomFilterType()$ ;
9       $c = S(\mathcal{U}(C_{low}, C_{high}))$ ;                /* Random cutoff frequency */
10      $o = S(\mathcal{U}(O_{low}, O_{high}))$ ;                /* Random filter order */
11      $x' = x' * getFilter(t, c, o)$ ;                /* Low pass filtering */
12      $x' = Resample(Resample(x', 44100, c * 2), c * 2, 44100)$ ; /* Resample */
13 with  $p_4$  probability:
14      $n = n * getFilter(t, c, o)$ ;                /* Low pass filtering on noise */
15      $n = Resample(Resample(n, c * 2), 44100)$ ;        /* Resample */
16 with  $p_5$  probability:
17      $s_1 = S(\mathcal{U}(S_{1low}, S_{1high}))$ ;                /* Random SNR */
18      $s_2 = S(\mathcal{U}(S_{2low}, S_{2high}))$ ;                /* Random Scale */
19      $n = \frac{n}{\frac{mean(abs(n))}{mean(abs(x'))}}$ ; /* Normalize the energy of noise */
20      $x' = (x' + \frac{n}{10^{snr/20}})$ ;                /* Add noise */
21  $x' = x' * s_2$ ;                                /* Scaling */
22  $x = x * s_2$ ;                                    /* Scaling */
```

Out: The randomly distorted speech x' and its target x

Distortion simulation algorithm we used for general speech restoration.