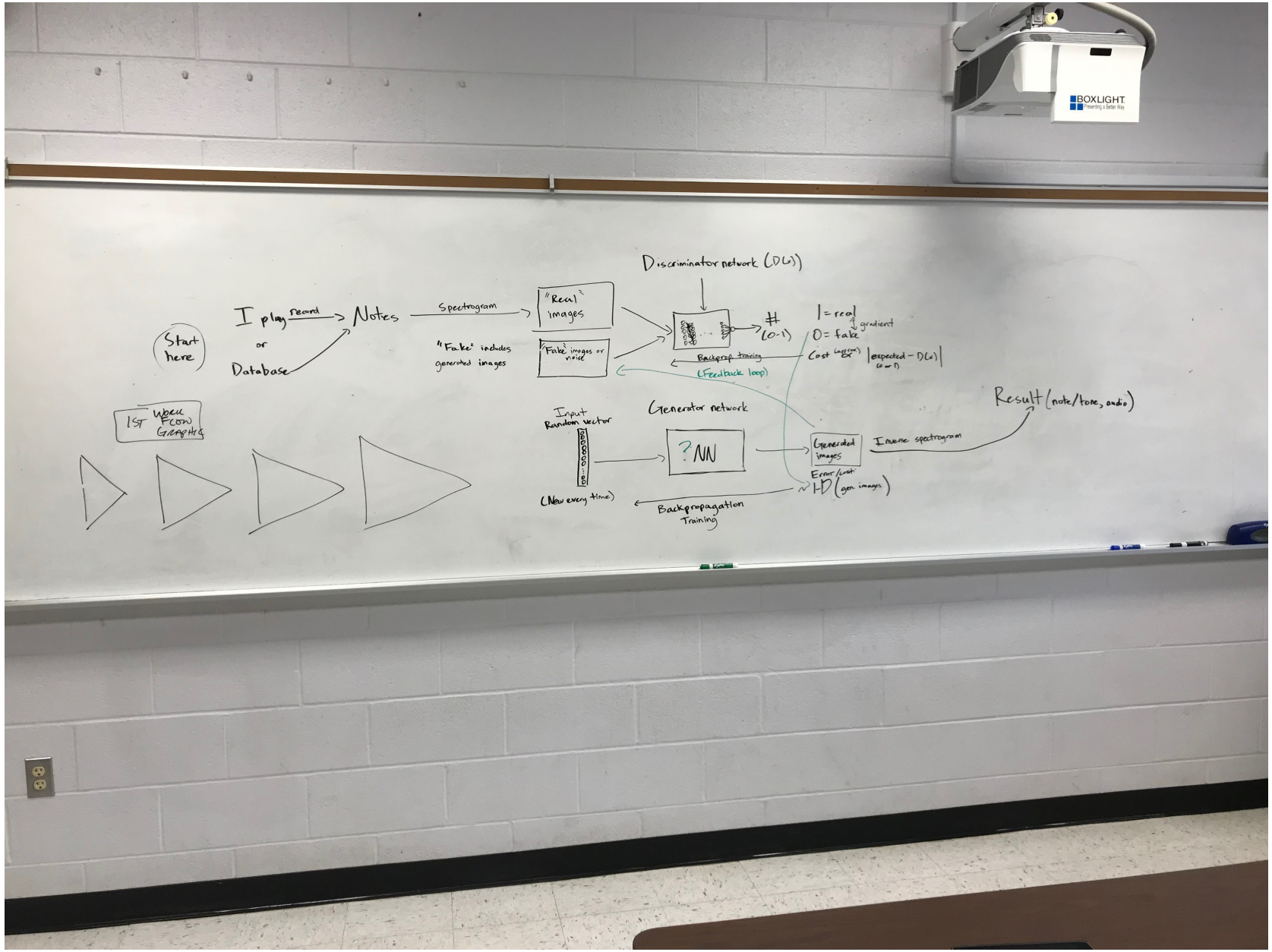
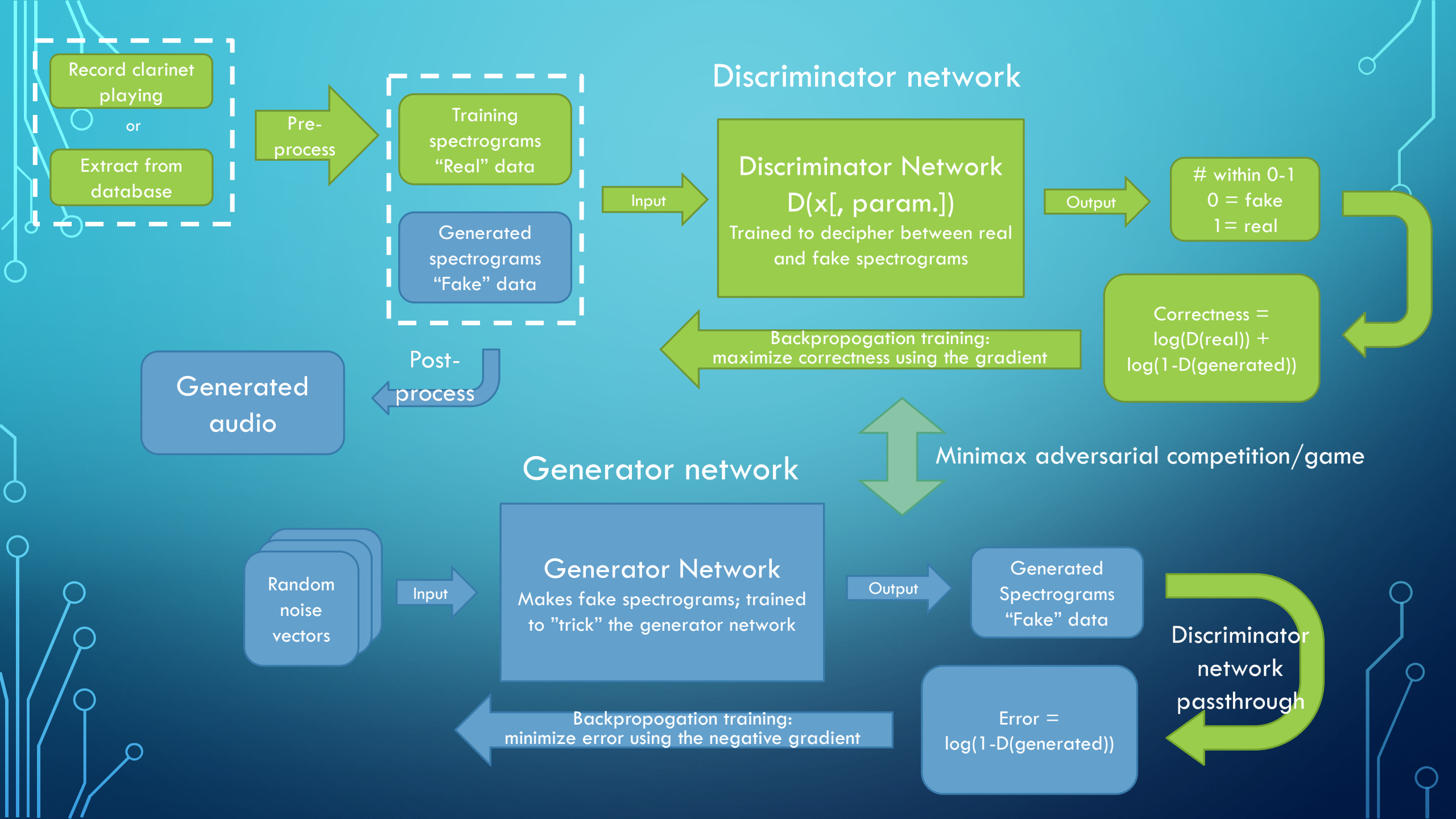
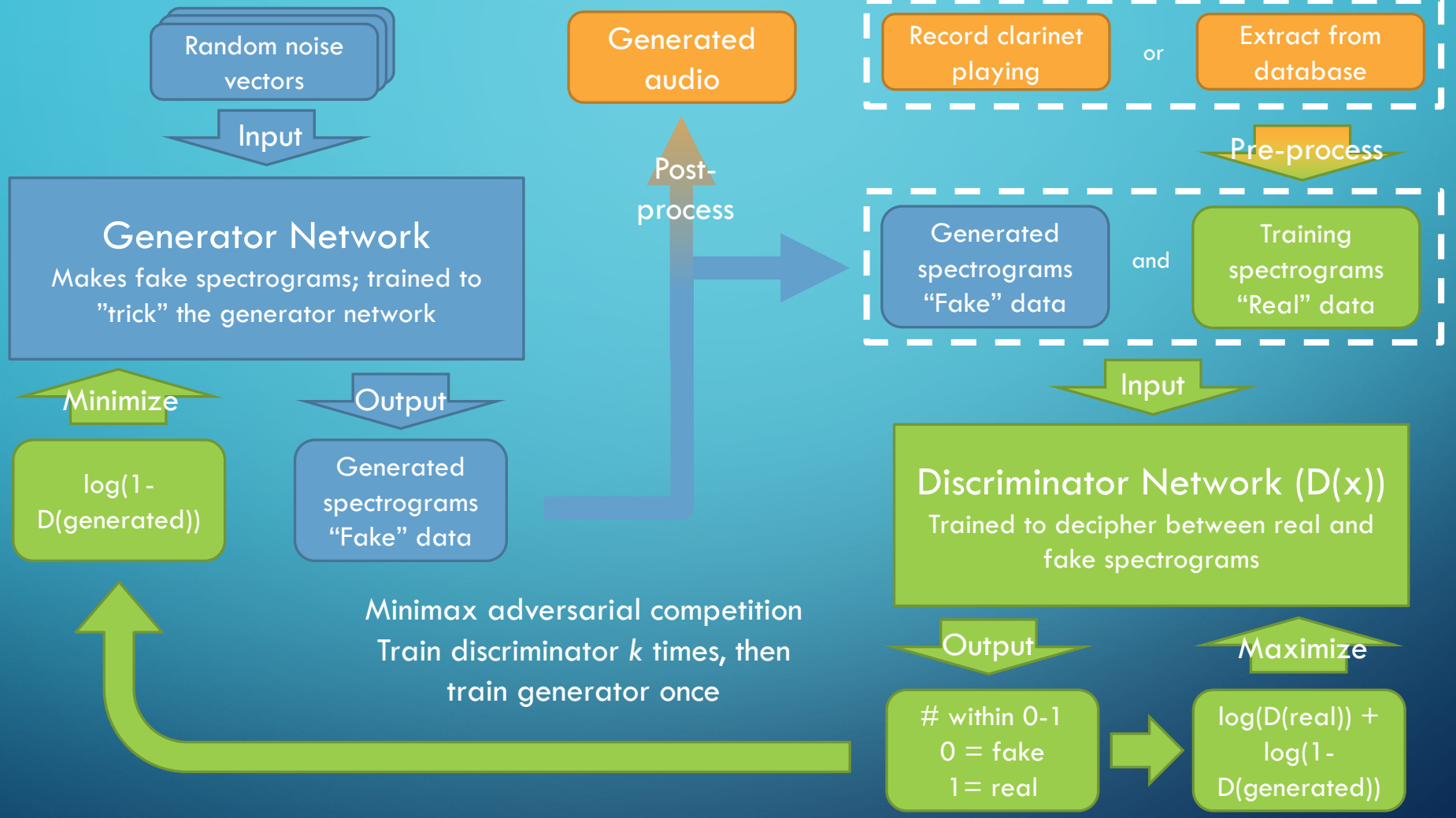

Discriminator network (D(x))
I play
record
Notes
or
Database
Start here
Spectrogram
"Real" images
"Fake" images or noise
"Fake" includes generated images
# (0-1)
1 = real
0 = fake
gradient
Backprop training
Cost (error)
|expected - D(x)|
(0 or 1)
(Feedback loop)
Result (note/tone, audio)
1ST WORK FLOW GRAPHIC
Input Random vector
Generator network
? NN
Generated images
Inverse spectrogram
Error/cost
HD (gen images)
(New every time)
Backpropagation Training


Record clarinet playing
or
Extract from database
Pre-process
Training spectrograms "Real" data
Generated spectrograms "Fake" data
Discriminator network
Input
Discriminator Network
D(x[, param.])
Trained to decipher between real and fake spectrograms
Output
# within 0-1
0 = fake
1= real
Correctness = log(D(real)) + log(1-D(generated))
Backpropogation training: maximize correctness using the gradient
Post-process
Generated audio
Minimax adversarial competition/game
Generator network
Random noise vectors
Input
Generator Network
Makes fake spectrograms; trained to "trick" the generator network
Output
Generated Spectrograms "Fake" data
Discriminator network passthrough
Error = log(1-D(generated))
Backpropogation training: minimize error using the negative gradient


Random noise vectors
Input
Generator Network
Makes fake spectrograms; trained to "trick" the generator network
Minimize
log(1-D(generated))
Output
Generated spectrograms "Fake" data
Post-process
Generated audio
Record clarinet playing
or
Extract from database
Pre-process
Generated spectrograms "Fake" data
and
Training spectrograms "Real" data
Input
Discriminator Network (D(x))
Trained to decipher between real and fake spectrograms
Output
# within 0-1
0 = fake
1= real
Maximize
log(D(real)) + log(1-D(generated))
Minimax adversarial competition
Train discriminator k times, then train generator once