Diffusion-based Vocoding for Real-Time Text-To-Speech

Lukas Gardberg

LTH

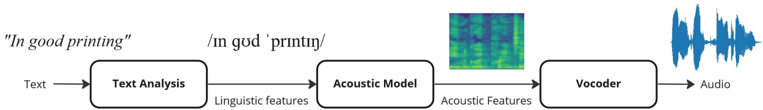
March 21, 2023

Presentation Layout

- Problem Introduction
- Past Work
- Problem Statement / Goals
- Quick background
- TODO: Add more

Typical TTS Pipeline

Text $\longrightarrow$ Speech



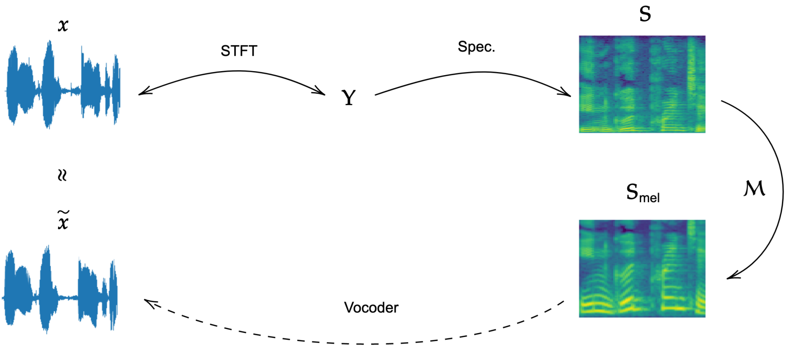
Text Analysis Acoustic Model Vocoder

The Phase Reconstruction Problem

- Goal: Reconstruct signal $\boldsymbol{x}$ from its mel spectrogram $\boldsymbol{S}_{\text{mel}}$

The Phase Reconstruction Problem

- Goal: Reconstruct signal $\boldsymbol{x}$ from its mel spectrogram $\boldsymbol{S}_{\mathrm{mel}}$



The Phase Reconstruction Problem

How has this been done before?

The Phase Reconstruction Problem

How has this been done before?

- Griffin-Lim Reconstruction

The Phase Reconstruction Problem

How has this been done before?

- Griffin-Lim Reconstruction
- Autoregressive Neural Networks (WaveNet)

The Phase Reconstruction Problem

How has this been done before?

- Griffin-Lim Reconstruction
- Autoregressive Neural Networks (WaveNet)
- GANs (HiFi-GAN)

The Phase Reconstruction Problem

How has this been done before?

- Griffin-Lim Reconstruction
- Autoregressive Neural Networks (WaveNet)
- GANs (HiFi-GAN)
- Diffusion (DiffWave)

The Phase Reconstruction Problem

How has this been done before?

- Griffin-Lim Reconstruction
- Autoregressive Neural Networks (WaveNet)
- GANs (HiFi-GAN)
- Diffusion (DiffWave)
- ...and more

Diffusion

- Main idea
- Where do Neural Networks come in?
- How can it be used for vocoding?

Diffusion

Collected training data $\mathbb{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ drawn as $\boldsymbol{x} \sim q_{\text{data}}$.

Diffusion

Collected training data $\mathbb{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ drawn as $\boldsymbol{x} \sim q_{\text{data}}$.

Want to be able to generate *new data* from the "target" distribution!

Diffusion

Collected training data $\mathbb{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ drawn as $\boldsymbol{x} \sim q_{\text{data}}$.

Want to be able to generate *new data* from the "target" distribution!

How?

Diffusion

- Transform data from a simple distribution p_{latent} (e.g. noise) into the complex target distribution q_{data}

Diffusion

- Transform data from a simple distribution p_{latent} (e.g. noise) into the complex target distribution q_{data}
- In one step is hard, in several steps easier

Diffusion

- Transform data from a simple distribution p_{latent} (e.g. noise) into the complex target distribution q_{data}
- In one step is hard, in several steps easier
- Also easy to *destroy* data into noise, hard to do the inverse

Diffusion

- Transform data from a simple distribution p_{latent} (e.g. noise) into the complex target distribution q_{data}
- In one step is hard, in several steps easier
- Also easy to *destroy* data into noise, hard to do the inverse
- Teach a model to perform each "denoising" step

Forward Process

T: Number of diffusion steps

Forward Process

T: Number of diffusion steps
$\boldsymbol{x}_0$: Ground truth sample

Forward Process

T: Number of diffusion steps
$\boldsymbol{x}_0$: Ground truth sample
$\boldsymbol{x}_t$: Sample after t diffusion steps

Forward Process

T: Number of diffusion steps
$\boldsymbol{x}_0$: Ground truth sample
$\boldsymbol{x}_t$: Sample after t diffusion steps
$\boldsymbol{x}_T$: Final sample, hopefully close to p_{latent}

Forward Process

T: Number of diffusion steps
$\boldsymbol{x}_0$: Ground truth sample
$\boldsymbol{x}_t$: Sample after t diffusion steps
$\boldsymbol{x}_T$: Final sample, hopefully close to p_{latent}

$$\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t\,;\,\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I})$$
$$\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \beta_t \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

Forward Process

T: Number of diffusion steps
$\boldsymbol{x}_0$: Ground truth sample
$\boldsymbol{x}_t$: Sample after t diffusion steps
$\boldsymbol{x}_T$: Final sample, hopefully close to p_{latent}

$$\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t\,;\ \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I})$$
$$\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \beta_t \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

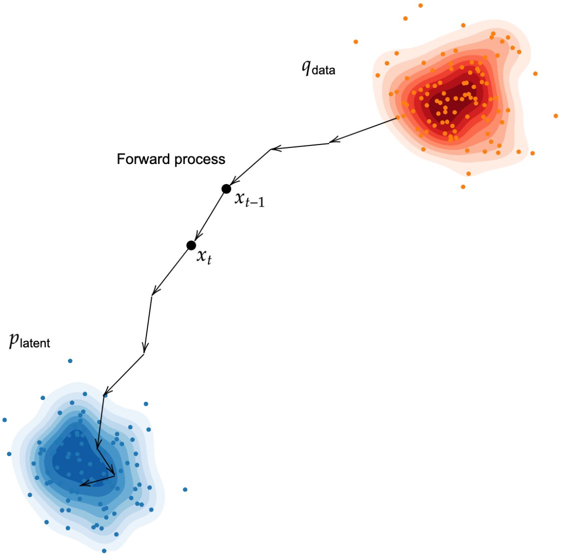
Variance schedule $\beta_t \in [0,1]$: At which "rate" we tend towards a unit Gaussian

Forward Process

Quick way to get a noisy sample $\boldsymbol{x}_t$:

$$
\begin{aligned}
\bar{\alpha}_t &= \prod_{i=1}^{T}(1-\beta_i) \\
\boldsymbol{x}_t &\sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t\,;\,\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, \sqrt{1-\bar{\alpha}_t}\boldsymbol{I}), \\
\boldsymbol{x} &= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})
\end{aligned}
$$

Forward Process



Backward Process

- We want to be able to generate samples using the backward process

Backward Process

- We want to be able to generate samples using the backward process

- Model each transition as $p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_{\boldsymbol{t}}) = \mathcal{N}(\boldsymbol{x}_{t-1}\,;\, \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t))$

Backward Process