WaveNet: A Generative Model for Raw Audio

Google DeepMind; van den Oord et al.
Presented by Adithya Ganesh

Feburary 28, 2017

Kundaje Lab Journal Club, Stanford University

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

- Architecture summary

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

- Architecture summary
  - Dilated convolutional layers

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

- Architecture summary
  - Dilated convolutional layers
  - Gated activation units

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

- Architecture summary
  - Dilated convolutional layers
  - Gated activation units
  - Residual / skip connections

WaveNet - Overview

- Generative deep neural network for modeling raw audio waveforms
- **Probabilistic** and **autoregressive**
  - The joint probability of a waveform $\mathbf{x} = \{x_1, \ldots, x_T\}$ is conditioned over all previous timesteps:

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).$$

- Architecture summary
  - Dilated convolutional layers
  - Gated activation units
  - Residual / skip connections
- Applied to text to speech, music generation, and phoneme recognition

Demos!

https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/1406.2661.pdf`

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator
  - Simultaneously train generator and discriminator net - classifies samples are coming from the true distribution $p(x)$ or the model distribution $\hat{p}(x)$

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator
  - Simultaneously train generator and discriminator net - classifies samples are coming from the true distribution $p(x)$ or the model distribution $\hat{p}(x)$
- Variational Autoencoders (Kingma et al. 2013)

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator
  - Simultaneously train generator and discriminator net - classifies samples are coming from the true distribution $p(x)$ or the model distribution $\hat{p}(x)$
- Variational Autoencoders (Kingma et al. 2013)
  - Use probabilistic graphical models and maximize lower bound on log likelihood

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator
  - Simultaneously train generator and discriminator net - classifies samples are coming from the true distribution $p(x)$ or the model distribution $\hat{p}(x)$
- Variational Autoencoders (Kingma et al. 2013)
  - Use probabilistic graphical models and maximize lower bound on log likelihood
- Autoregressive models (e.g. WaveNet, PixelRNN, van den Oord et al. 2016)

Three Approaches to Generative Models

- Generative Adversarial Networks (Goodfellow et al. 2014)
  - Goodfellow et al. 2014: `https://arxiv.org/pdf/`1406.2661`.pdf`
  - Pose training problem as a minimax two-player game between generator and discriminator
  - Simultaneously train generator and discriminator net - classifies samples are coming from the true distribution $p(x)$ or the model distribution $\hat{p}(x)$
- Variational Autoencoders (Kingma et al. 2013)
  - Use probabilistic graphical models and maximize lower bound on log likelihood
- Autoregressive models (e.g. WaveNet, PixelRNN, van den Oord et al. 2016)
  - Model conditional distribution of every datapoint given previous datapoints

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
    - However, often need to make independence assumptions

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
    - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
    - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)
    - Can be difficult to optimize; unstable training dynamics

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
    - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)
    - Can be difficult to optimize; unstable training dynamics
    - However, with recent developments such as WGAN (Arjovsky et al. 2017), some of these challenges can be alleviated

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
    - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)
    - Can be difficult to optimize; unstable training dynamics
    - However, with recent developments such as WGAN (Arjovsky et al. 2017), some of these challenges can be alleviated
- Autoregressive models (e.g. WaveNet) have simple and stable training process (softmax loss), but sampling can be inefficient

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
  - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)
  - Can be difficult to optimize; unstable training dynamics
  - However, with recent developments such as WGAN (Arjovsky et al. 2017), some of these challenges can be alleviated
- Autoregressive models (e.g. WaveNet) have simple and stable training process (softmax loss), but sampling can be inefficient
- Further reading: OpenAI article

Three Approaches to Generative Models

- VAEs allow efficient Bayesian inference over complex distributions (e.g. Gregor et al. 2015, DRAW: A Recurrent Neural Network For Image Generation)
  - However, often need to make independence assumptions
- GANs known for generating the most realistic images (June 2016, OpenAI)
  - Can be difficult to optimize; unstable training dynamics
  - However, with recent developments such as WGAN (Arjovsky et al. 2017), some of these challenges can be alleviated
- Autoregressive models (e.g. WaveNet) have simple and stable training process (softmax loss), but sampling can be inefficient
- Further reading: OpenAI article
  - `https://openai.com/blog/generative-models/`

Speech synthesis - A brief introduction

- Generating speech with computers (i.e. text-to-speech)

Speech synthesis - A brief introduction

- Generating speech with computers (i.e. text-to-speech)
- Techniques be classified into **concatenative** vs. **parametric**

Speech synthesis - A brief introduction

- Generating speech with computers (i.e. text-to-speech)
- Techniques be classified into **concatenative** vs. **parametric**
    - Concatenative: Database of short speech fragments are recombined to form complete sentences.

Speech synthesis - A brief introduction

- Generating speech with computers (i.e. text-to-speech)
- Techniques be classified into **concatenative** vs. **parametric**
  - Concatenative: Database of short speech fragments are recombined to form complete sentences.
  - Parametric: Generative approach, where information needed to create signals are stored in the model.

Speech synthesis - A brief introduction

- Generating speech with computers (i.e. text-to-speech)
- Techniques be classified into **concatenative** vs. **parametric**
  - Concatenative: Database of short speech fragments are recombined to form complete sentences.
  - Parametric: Generative approach, where information needed to create signals are stored in the model.
- Parametric models more robust and flexible, but tend to sound less natural than concatenative approaches

Historical aside: Vocoders

- Parametric models generate audio by passing their outputs through signal processing algorithms called *vocoders*

Historical aside: Vocoders

- Parametric models generate audio by passing their outputs through signal processing algorithms called *vocoders*
- Rich history of development – early vocoder hardware dates to 1928

Historical aside: Vocoders

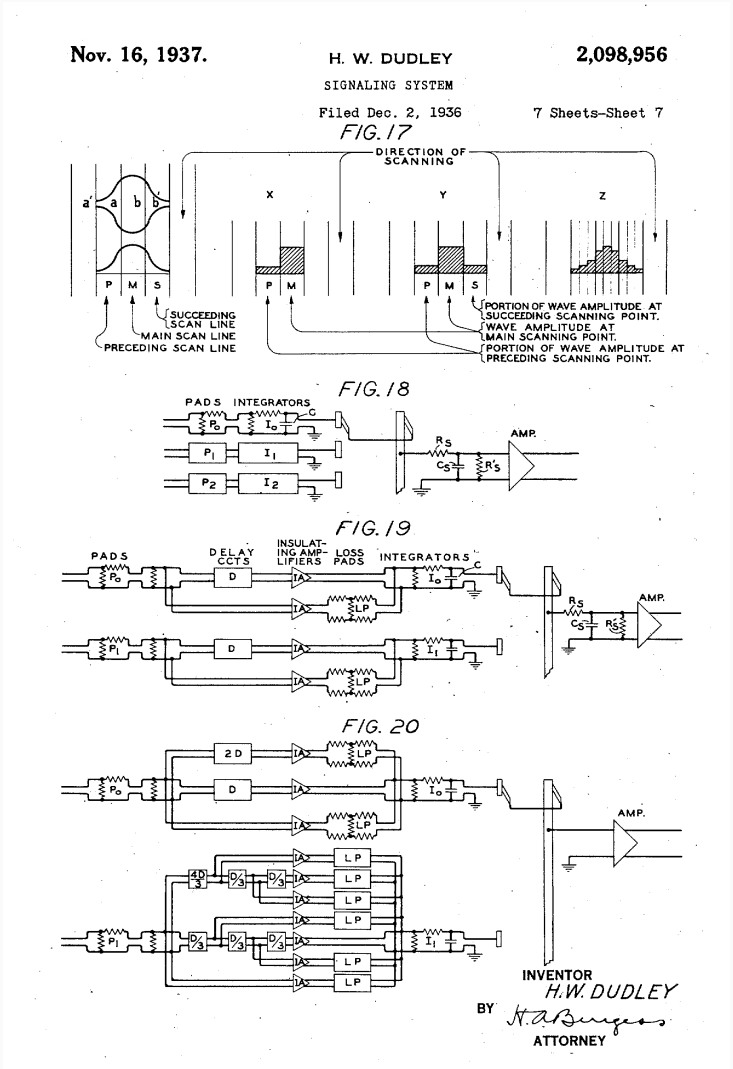

Figure 1: Bell Labs engineer Homer Dudley's 1937 patent (image: Google Patents)

Historical aside: Vocoders

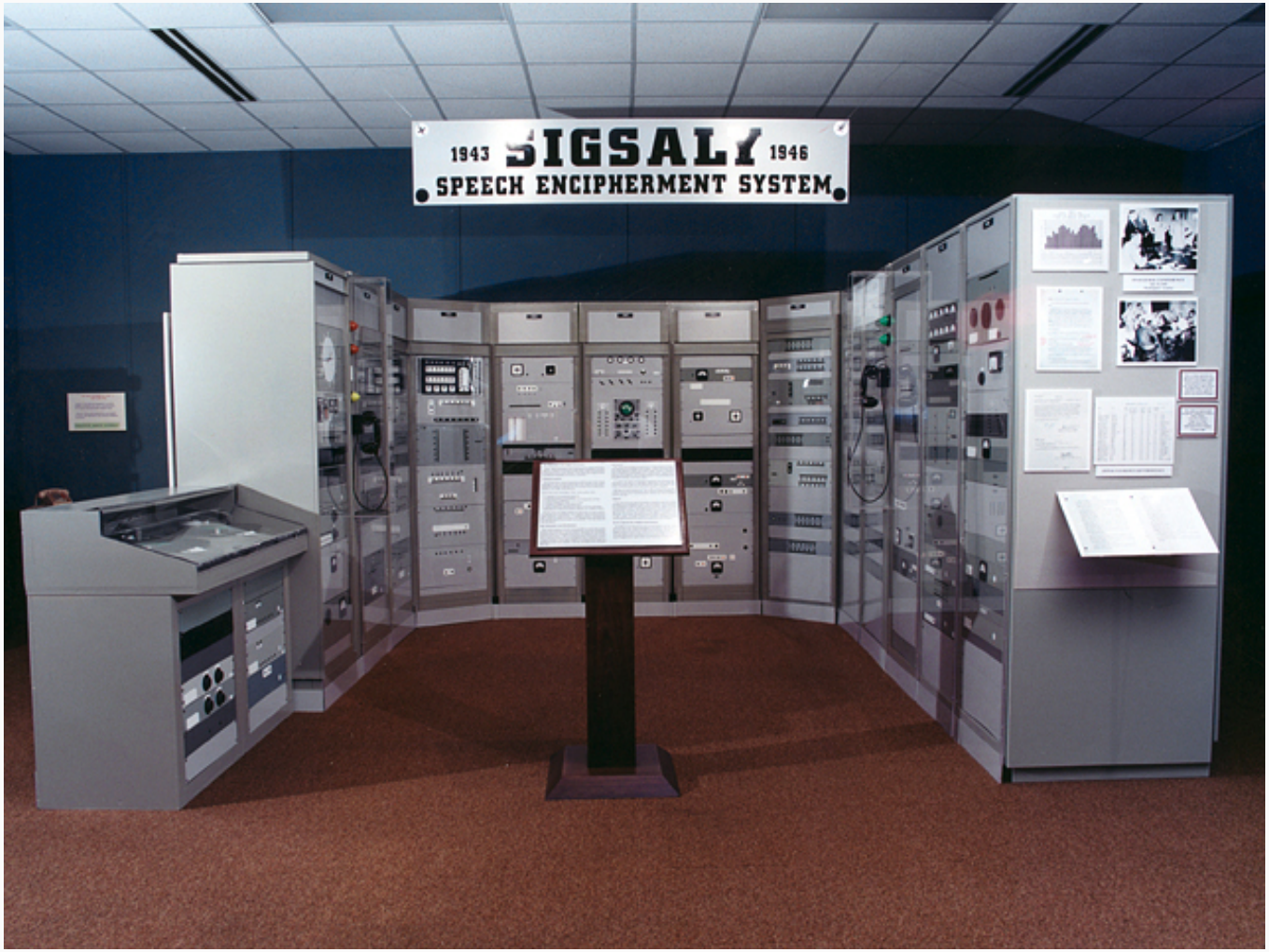

Figure 2: SIGSALY - encrypted voice communications system used in WWII, built in 1943 by Bell Labs engineers (image: Wikipedia)

Temporal Resolution

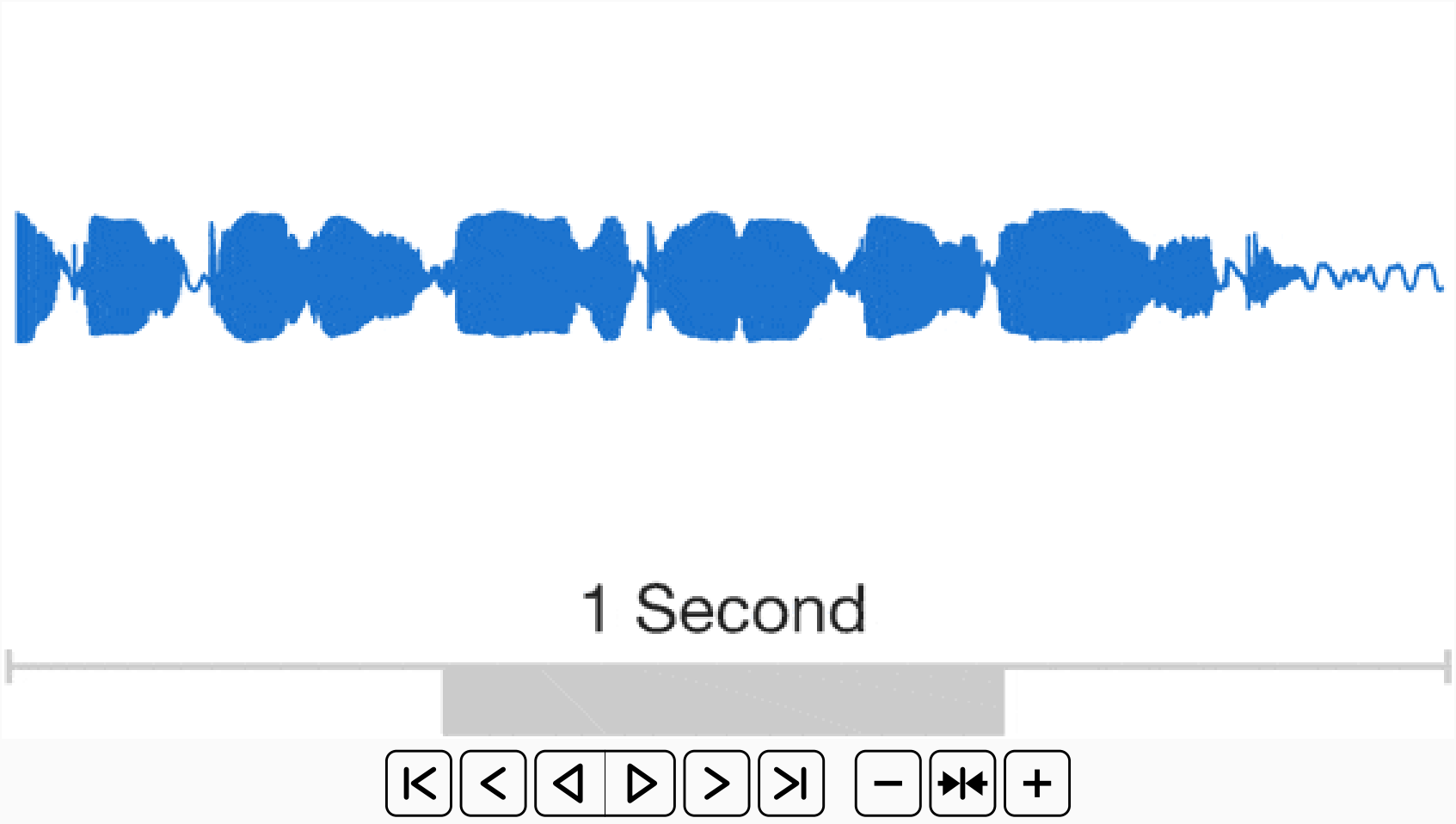

Figure 3: Using dilated convolutions, WaveNet is able to model raw waveforms at 16KHz (image: DeepMind)

Architecture

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step
  - Allows network to operate on a coarser scale than with a normal convolution, but more efficient

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step
  - Allows network to operate on a coarser scale than with a normal convolution, but more efficient
  - A form of “dimensionality reduction” similar to pooling

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step
  - Allows network to operate on a coarser scale than with a normal convolution, but more efficient
  - A form of “dimensionality reduction” similar to pooling
- Further reading

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step
  - Allows network to operate on a coarser scale than with a normal convolution, but more efficient
  - A form of “dimensionality reduction” similar to pooling
- Further reading
  - Yu et al., “Multi-Scale Context Aggregation by Dilated Convolutions”, ICLR 2016

Dilated convolutional layers

- Causal, i.e. prediction $p(x_{t+1}|x_1, \ldots, x_t)$ does not depend on future timesteps $x_{t+1}, \ldots, x_T$.
- Filter is applied over an area larger than its length by skipping input values with a certain step
  - Allows network to operate on a coarser scale than with a normal convolution, but more efficient
  - A form of "dimensionality reduction" similar to pooling
- Further reading
  - Yu et al., "Multi-Scale Context Aggregation by Dilated Convolutions", ICLR 2016
  - Chen et al., "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs", ICLR 2015

Dilated convolutional layers

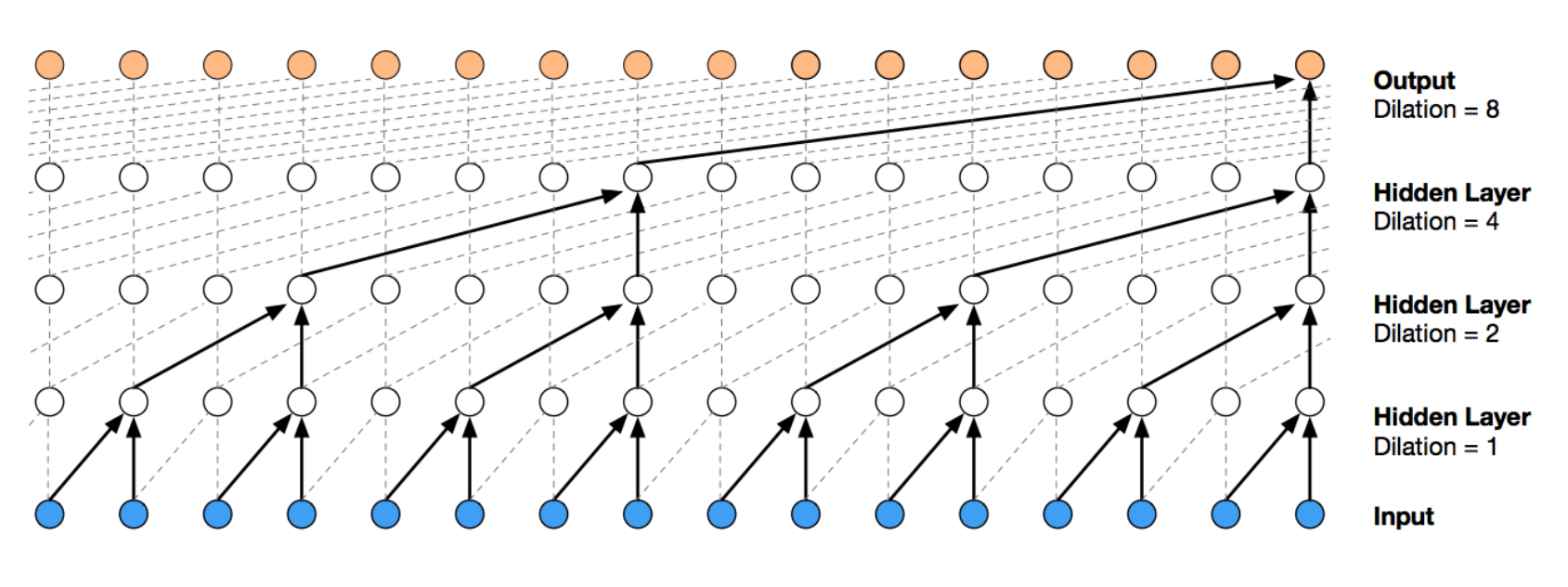

Figure 4: Visualization of a stack of dilated convolutional layers

- Stacked dilated convolutions enable very large receptive fields with few layers

Dilated convolutional layers

- Let F be a discrete function, and k be a discrete filter. The discrete convolution operator $*$ is defined as

$$(F * k)(\mathbf{p}) = \sum_{\mathbf{s}+\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

  More generally, let l be a dilation factor. The l-dilated convolution can be defined as

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

Dilated convolutional layers

- Let F be a discrete function, and k be a discrete filter. The discrete convolution operator $*$ is defined as

$$(F * k)(\mathbf{p}) = \sum_{\mathbf{s}+\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

  More generally, let l be a dilation factor. The l-dilated convolution can be defined as

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

- Implemented in TensorFlow as `tf.nn.atrous_conv2d`

Dilated convolutional layers

- Let F be a discrete function, and k be a discrete filter. The discrete convolution operator $*$ is defined as

$$(F * k)(\mathbf{p}) = \sum_{\mathbf{s}+\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

  More generally, let l be a dilation factor. The l-dilated convolution can be defined as

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}).$$

- Implemented in TensorFlow as `tf.nn.atrous_conv2d`
  - Name comes from “à trous” in French, i.e. “with holes”

Dilated convolutional layers

- Paper uses exponentially doubling dilations up to a limit, and then repeated:

$$1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512.$$

Dilated convolutional layers

- Paper uses exponentially doubling dilations up to a limit, and then repeated:

$$1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512.$$

- Each 1,2,4, ..., 512 block has a receptive field of size 1024, a more efficient counterpart of 1×1024 convolution

Dilated convolutional layers

- Paper uses exponentially doubling dilations up to a limit, and then repeated:

$$1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512.$$

- Each 1,2,4, ..., 512 block has a receptive field of size 1024, a more efficient counterpart of 1×1024 convolution
- Stacking these units further increases model expressivity

Softmax distribution

- Softmax used to model output $p(x_t|x_1, \ldots, x_{t-1})$, even though audio waveforms are continuous

[1] Theis et al., 2015

Softmax distribution

- Softmax used to model output $p(x_t|x_1, \ldots, x_{t-1})$, even though audio waveforms are continuous
- Empirically found to outperform mixture models, e.g. MCGSM [1]

[1] Theis et al., 2015

Softmax distribution

- Softmax used to model output $p(x_t|x_1, \ldots, x_{t-1})$, even though audio waveforms are continuous
- Empirically found to outperform mixture models, e.g. MCGSM [1]
- Raw audio typically stored as time series of 16-bit integer values (65,356 possibilities)

[1] Theis et al., 2015

Softmax distribution

- Softmax used to model output $p(x_t|x_1, \ldots, x_{t-1})$, even though audio waveforms are continuous
- Empirically found to outperform mixture models, e.g. MCGSM [1]
- Raw audio typically stored as time series of 16-bit integer values (65,356 possibilities)
  - Authors use a μ-law companding transformation to quantize to 256 possible values

$$f(x_t) = \operatorname{sgn}(x_t) \frac{\ln(1 + \mu|x_t|)}{\ln(1 + \mu)},$$

where $-1 < x_t < 1$ and $\mu = 255$.

[1] Theis et al., 2015

Softmax distribution

- Softmax used to model output $p(x_t|x_1, \ldots, x_{t-1})$, even though audio waveforms are continuous
- Empirically found to outperform mixture models, e.g. MCGSM [1]
- Raw audio typically stored as time series of 16-bit integer values (65,356 possibilities)
  - Authors use a μ-law companding transformation to quantize to 256 possible values

$$f(x_t) = \operatorname{sgn}(x_t) \frac{\ln(1 + \mu|x_t|)}{\ln(1 + \mu)},$$

    where $-1 < x_t < 1$ and $\mu = 255$.
  - Well-known signal processing transform – better reconstruction than linear quantization

[1] Theis et al., 2015

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter
- Empirically found to outperform ReLUs

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter
- Empirically found to outperform ReLUs
- Feedforward nets with gated units found beneficial in other works:

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter
- Empirically found to outperform ReLUs
- Feedforward nets with gated units found beneficial in other works:
  - Highway networks (Srivastava et al., NIPS 2015)

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter
- Empirically found to outperform ReLUs
- Feedforward nets with gated units found beneficial in other works:
  - Highway networks (Srivastava et al., NIPS 2015)
  - Grid LSTM (Kalchbrenner et al. 2016, ICLR 2016)

Gated activation units

- Gated activation units, same as used in gated PixelCNN (van den Oord et al., 2016)

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$$

  - $*$: convolution; $\odot$: elementwise multiplication
  - σ: sigmoid function
  - k: layer index; f: filter; g: gate
  - W: learned convolution filter
- Empirically found to outperform ReLUs
- Feedforward nets with gated units found beneficial in other works:
  - Highway networks (Srivastava et al., NIPS 2015)
  - Grid LSTM (Kalchbrenner et al. 2016, ICLR 2016)
  - Neural GPUs (Kaiser et al. 2016, ICLR 2016)

Residual / Skip Connections

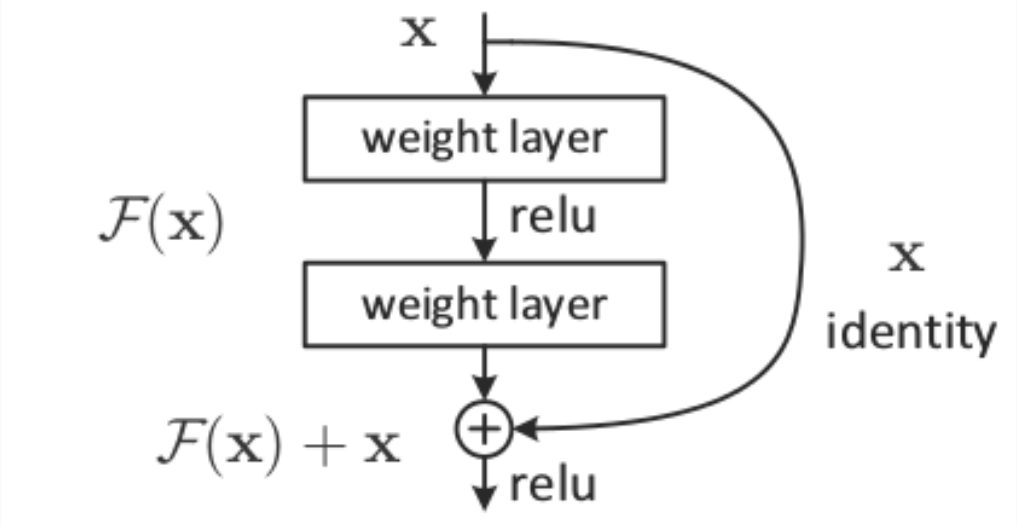

Figure 5: Review: optimizing the residual $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$ is more tractable for deeper networks (image: He et al. 2015)

Residual / Skip Connections

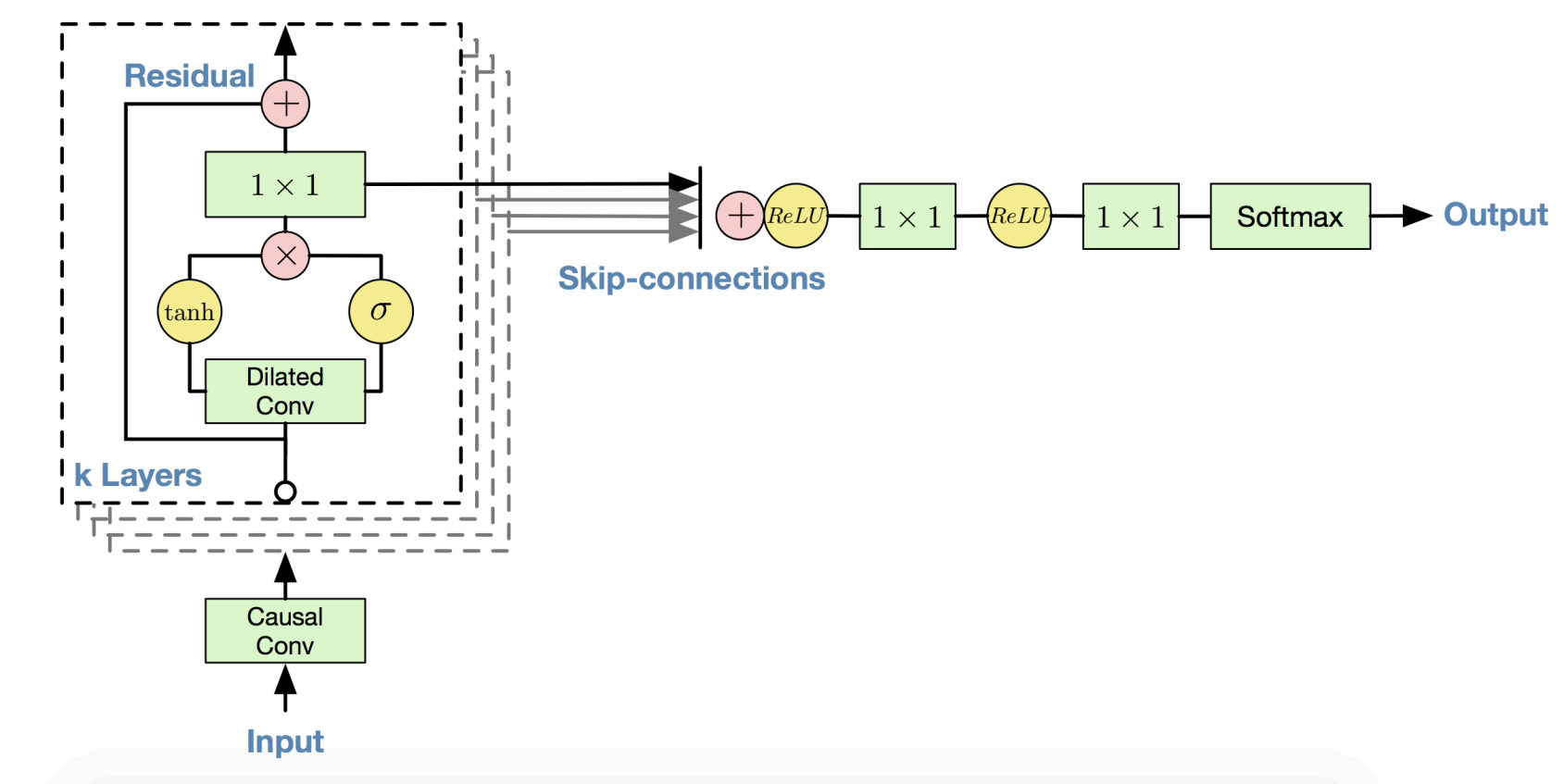

Figure 6: summary of residual block and overall architecture

Conditional Wavenets

- Given an aadditional $\mathbf{h}$, we model the conditional distribution $p(\mathbf{x}|\mathbf{h})$ of the waveform:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}, \mathbf{h}).$$

Conditional Wavenets

- Given an aadditional $\mathbf{h}$, we model the conditional distribution $p(\mathbf{x}|\mathbf{h})$ of the waveform:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}, \mathbf{h}).$$

- Conditioning on other variables enables audio generation with desired characteristics

Conditional Wavenets

- Given an aadditional $\mathbf{h}$, we model the conditional distribution $p(\mathbf{x}|\mathbf{h})$ of the waveform:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}, \mathbf{h}).$$

- Conditioning on other variables enables audio generation with desired characteristics
- Examples:

Conditional Wavenets

- Given an aadditional $\mathbf{h}$, we model the conditional distribution $p(\mathbf{x}|\mathbf{h})$ of the waveform:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}, \mathbf{h}).$$

- Conditioning on other variables enables audio generation with desired characteristics
- Examples:
  - In multi-speaker setting, can pass in speaker identity

Conditional Wavenets

- Given an aadditional $\mathbf{h}$, we model the conditional distribution $p(\mathbf{x}|\mathbf{h})$ of the waveform:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_{t-1}, \mathbf{h}).$$

- Conditioning on other variables enables audio generation with desired characteristics
- Examples:
  - In multi-speaker setting, can pass in speaker identity
  - For text-to-speech, can pass in information about text as extra input

Global Conditioning

- Can condition the model on other inputs in two different ways - global vs. local

Global Conditioning

- Can condition the model on other inputs in two different ways - global vs. local
- Global conditioning – single latent variable $\mathbf{h}$ that influences output distribution across all timesteps (e.g. speaker embedding).

Global Conditioning

- Can condition the model on other inputs in two different ways - global vs. local
- Global conditioning – single latent variable $\mathbf{h}$ that influences output distribution across all timesteps (e.g. speaker embedding).
- New activation function:

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot (W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h}),$$

where $V_{*,k}$ is a learned linear projection

Local Conditioning

- Local conditioning - consider a second time series h_t, e.g. linguistic features in a TTS model

Local Conditioning

- Local conditioning - consider a second time series h_t, e.g. linguistic features in a TTS model
- First: upsample this time series using a transposed convolutional network. Obtain new time series $\mathbf{y} = f(\mathbf{h})$, with same resolution as audio signal.

Local Conditioning

- Local conditioning - consider a second time series h_t, e.g. linguistic features in a TTS model
- First: upsample this time series using a transposed convolutional network. Obtain new time series $\mathbf{y} = f(\mathbf{h})$, with same resolution as audio signal.
- New activation function:

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k} * \mathbf{y}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k} * \mathbf{y}),$$

where $V_{f,k} * \mathbf{y}$ is now a 1×1 convolution.

Experiments

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)
- Dataset: English multi-speaker corpus from CSTR voice cloning toolkit (VCTK)

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)
- Dataset: English multi-speaker corpus from CSTR voice cloning toolkit (VCTK)
    - 44 hours of data from 109 different speakers

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)
- Dataset: English multi-speaker corpus from CSTR voice cloning toolkit (VCTK)
    - 44 hours of data from 109 different speakers
- Conditioning on speaker with one-hot vector

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)
- Dataset: English multi-speaker corpus from CSTR voice cloning toolkit (VCTK)
  - 44 hours of data from 109 different speakers
- Conditioning on speaker with one-hot vector
- Generated non-existent but human language-like words with realistic sounding intonations

Experiment A: Multi-speaker speech generation

- Multi-speaker speech generation (free-form; not conditioned on text)
- Dataset: English multi-speaker corpus from CSTR voice cloning toolkit (VCTK)
  - 44 hours of data from 109 different speakers
- Conditioning on speaker with one-hot vector
- Generated non-existent but human language-like words with realistic sounding intonations
  - Analogous to "realistic but unnatural" generative models such as `http://karpathy.github.io/`2015/05/21/`rnn-effectiveness/`

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers
- Locally conditioned on linguistic features, derived from input texts

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers
- Locally conditioned on linguistic features, derived from input texts
    - Phone identities, syllable stress, number of syllables in a word, position of current syllable in phrase

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers
- Locally conditioned on linguistic features, derived from input texts
  - Phone identities, syllable stress, number of syllables in a word, position of current syllable in phrase
  - Computed every 5ms by phone-level alignment at train time

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers
- Locally conditioned on linguistic features, derived from input texts
  - Phone identities, syllable stress, number of syllables in a word, position of current syllable in phrase
  - Computed every 5ms by phone-level alignment at train time
- Also conditioned on logarithmic fundamental frequency $\log F_0$

Experiment B: Text-to-speech

- Dataset: Google's English (24.6 hours) / Mandain Chinese (34.8 hours) TTS corpus; professional female speakers
- Locally conditioned on linguistic features, derived from input texts
  - Phone identities, syllable stress, number of syllables in a word, position of current syllable in phrase
  - Computed every 5ms by phone-level alignment at train time
- Also conditioned on logarithmic fundamental frequency $\log F_0$
- Receptive field size: 240 ms

Experiment B: Text-to-speech

Speech samples	Subjective 5-scale MOS in naturalness	
	North American English	Mandarin Chinese
LSTM-RNN parametric	3.67 ± 0.098	3.79 ± 0.084
HMM-driven concatenative	3.86 ± 0.137	3.47 ± 0.108
WaveNet (L+F)	$\mathbf{4.21} \pm 0.081$	$\mathbf{4.08} \pm 0.085$
Natural (8-bit μ-law)	4.46 ± 0.067	4.25 ± 0.082
Natural (16-bit linear PCM)	4.55 ± 0.075	4.21 ± 0.071

Figure 7: WaveNet outperforms mean opinion score in naturalness over LSTM / HMM models

Experiment C: Music generation

- Datasets

Experiment C: Music generation

- Datasets
  - MagnaTagATune dataset (Law & Von Ahn, 2009) — 200 hours of audio, annotated with tags describing genre, instrumentation, tempo, volume, mood

Experiment C: Music generation

- Datasets
  - MagnaTagATune dataset (Law & Von Ahn, 2009) — 200 hours of audio, annotated with tags describing genre, instrumentation, tempo, volume, mood
  - YouTube piano dataset — 60 hours of solo piano music (easier to model, since constrained to a single instrument)

Experiment C: Music generation

- Datasets
  - MagnaTagATune dataset (Law & Von Ahn, 2009) — 200 hours of audio, annotated with tags describing genre, instrumentation, tempo, volume, mood
  - YouTube piano dataset — 60 hours of solo piano music (easier to model, since constrained to a single instrument)
- Key finding - enlarging receptive field was crucial to obtain samples that sound musical

Experiment C: Music generation

- Datasets
  - MagnaTagATune dataset (Law & Von Ahn, 2009) — 200 hours of audio, annotated with tags describing genre, instrumentation, tempo, volume, mood
  - YouTube piano dataset — 60 hours of solo piano music (easier to model, since constrained to a single instrument)
- Key finding - enlarging receptive field was crucial to obtain samples that sound musical
- Authors conducted preliminary experiments conditioning on genre tags

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms
- Mean-pooling layer after dilated convolutions, activations over 10ms frames

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms
- Mean-pooling layer after dilated convolutions, activations over 10ms frames
- Loss consisted of two terms:

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms
- Mean-pooling layer after dilated convolutions, activations over 10ms frames
- Loss consisted of two terms:
  - Predicting the next sample

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms
- Mean-pooling layer after dilated convolutions, activations over 10ms frames
- Loss consisted of two terms:
  - Predicting the next sample
  - Classify the frame

Experiment D: Speech recognition

- Speech recognition on the TIMIT dataset (Garafolo et al. 1993)
  - 630 speakers
  - Orthographic, phonetic, and word transcriptions
  - 16-bit, 16kHz waveforms
- Mean-pooling layer after dilated convolutions, activations over 10ms frames
- Loss consisted of two terms:
  - Predicting the next sample
  - Classify the frame
- Achieved 18.8 PER on test set (state-of-the-art when trained on raw audio)

Further Reading

- Oord, Aaron van den, et al. ”Conditional Image Generation with PixelCNN Decoders.” arXiv preprint arXiv:1606.05328 (2016).

Further Reading

- Oord, Aaron van den, et al. ”Conditional Image Generation with PixelCNN Decoders.” arXiv preprint arXiv:1606.05328 (2016).
- Oord, Aaron van den, Nal Kalchbrenner, and Koray Kavukcuoglu. ”Pixel recurrent neural networks.” arXiv preprint arXiv:1601.06759 (2016).

Further Reading

- Oord, Aaron van den, et al. ”Conditional Image Generation with PixelCNN Decoders.” arXiv preprint arXiv:1606.05328 (2016).
- Oord, Aaron van den, Nal Kalchbrenner, and Koray Kavukcuoglu. ”Pixel recurrent neural networks.” arXiv preprint arXiv:1601.06759 (2016).
- Salimans, Tim, et al. ”PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications.” arXiv preprint arXiv:1701.05517 (2017).

Further Reading

- Oord, Aaron van den, et al. ”Conditional Image Generation with PixelCNN Decoders.” arXiv preprint arXiv:1606.05328 (2016).
- Oord, Aaron van den, Nal Kalchbrenner, and Koray Kavukcuoglu. ”Pixel recurrent neural networks.” arXiv preprint arXiv:1601.06759 (2016).
- Salimans, Tim, et al. ”PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications.” arXiv preprint arXiv:1701.05517 (2017).
- Gulrajani, Ishaan, et al. ”PixelVAE: A Latent Variable Model for Natural Images.” arXiv preprint arXiv:1611.05013 (2017).

Further Reading

- Oord, Aaron van den, et al. ”Conditional Image Generation with PixelCNN Decoders.” arXiv preprint arXiv:1606.05328 (2016).
- Oord, Aaron van den, Nal Kalchbrenner, and Koray Kavukcuoglu. ”Pixel recurrent neural networks.” arXiv preprint arXiv:1601.06759 (2016).
- Salimans, Tim, et al. ”PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications.” arXiv preprint arXiv:1701.05517 (2017).
- Gulrajani, Ishaan, et al. ”PixelVAE: A Latent Variable Model for Natural Images.” arXiv preprint arXiv:1611.05013 (2017).
- Mehri, Soroush, et al. ”SampleRNN: An Unconditional End-to-End Neural Audio Generation Model.” arXiv preprint arXiv:1612.07837 (2016).