Paper Summaries

Some Joke About VAEs

Written @ Corti

Authors:
Magnus Berg Sletfjerding
{ms}@corti.ai
April 26, 2021

Abstract

This was written for me to understand papers in my thesis better. Don't be alarmed if you don't understand it 100%, I probably don't either.

0 Chapter 1: WaveNet

The WaveNet paper presents a CNN-based approach to generating audio samples. [?] Instead of using RNNs as a recurrent architecture, the generative model only conditions on past samples, and as such does not include any hidden "state".

The probability of a waveform $\mathbf{x} \in \mathbb{R}^T$ is expressed purely as:

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

where $p(x_t|x_1, ..., x_t)$ is parametrized only by the weights in the network.

0.1 Architecture and design

The WaveNet Architecture draws advantage from three developments: quantized output spaces (as shown in PixelRNN), dilated causal convolutions and gated activation units,

Quantized Output Space with μ law companding transformation Given an audio waveform $\mathbf{x} \in [-1, 1]^T$, transform the audio according to :

$$f(x_t) = \text{sign}(x_t) \frac{\ln(1 - \mu|x_t|)}{\ln(1 + \mu)} \tag{2}$$

with $\mu = 255$.

Dilated Causal Convolutions A Causal Convolution is a fancy way of saying that audio convolutions only work forward in time, not backward. This is to enforce the forward dependency in eq. (1).

A Dilated Convolution is a convolution where the convolution kernel skips over a dimension, increasing the receptive field and observing more of the surrounding environment. For an image the simplest dilated convolutional is illustrated in fig. 1

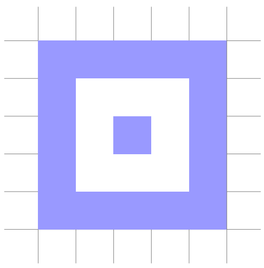

Figure 1: A Simple Pixel Dilated Convolution

Accordingly, for an audio signal, it would look like what we see in fig. 2

Gated Activation Units Each Convolution layer, Insteadof just having a filter weight, also has a **gating weight**. Hence the weights $\mathbf{W} \in \mathbb{R}^{K \times 2}$, with K as the number of layers. The operation of layer $k \in [0, K]$, is parametrized as:

$$\mathbf{z} = \tanh(\mathbf{x} * W_{k,f}) \odot \sigma(\mathbf{x} * W_{k,g}) \tag{3}$$

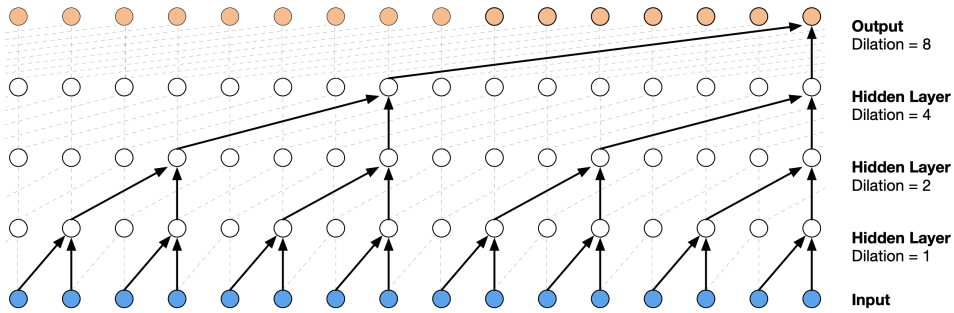

Figure 2: The Dilated Causal Convolution in WaveNet

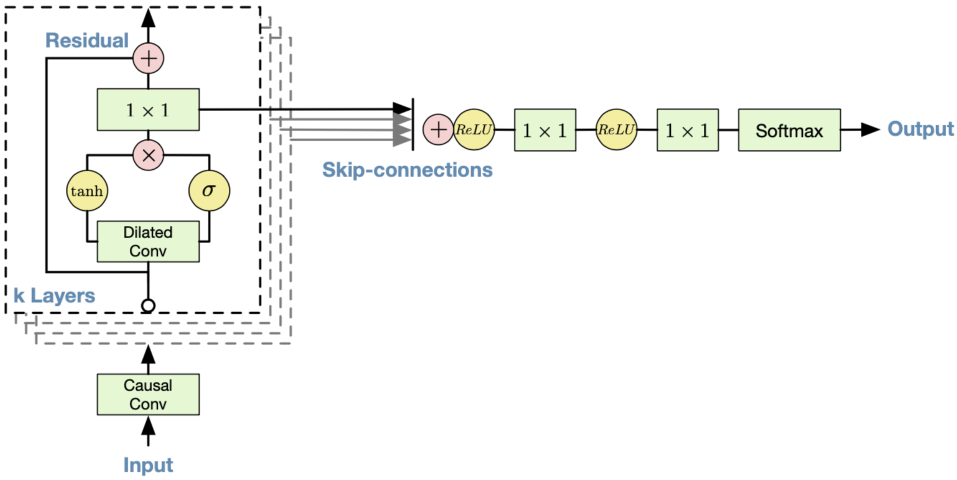

Figure 3: Overall Residual Architecture of WaveNet. Skip connections happen from every Convolutional Layer to the final softmax.

Summary of architechture The architecture is summed up in fig. 3. It's important to note that the Causal Convolution setup as described in fig. 2 only is applied once, as the first layer. This makes the entire rest of the network a simple convolutional network with dilation, as the **first (causal) convolutional stack ensures that the rest of the network will only see samples from the past.** In all other respects we can consider this a standard CNN architecture.

0.1.1 Extending architecture to include latent representations of speaker

It's possible to add a latent representation $\mathbf{h}$, extending eq. (1) to:

$$p(\mathbf{x}|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, \ldots, x_t, \mathbf{h}) \tag{4}$$

There are two ways to represent this:

Global Conditioning (Speaker, Accent, Noise level) Here we set $\mathbf{h}$ to a single global latent, representing a constant over the entire sequence. The activation from eq. (3) then becomes

$$\mathbf{z} = \tanh(\mathbf{x} * W_{k,f} + V_{k,f}^T \mathbf{h}) \odot \sigma(\mathbf{x} * W_{k,g} + V_{k,g}^T \mathbf{h}) \tag{5}$$

With $V_{k,*}$ is a linear projection , and the resulting vector $V_{k,f}^T \mathbf{h}$ is broadcast over time T.

Local Conditioning (Tone of voice, changing noise levels over the call Here we define h_t, and use a ConvNet to upsample h_t to $\mathbf{y} = f(\mathbf{h})$, so eq. (3) becomes:

$$\mathbf{z} = \tanh(\mathbf{x} * W_{k,f} + V_{k,f} * \mathbf{y}) \odot \sigma(\mathbf{x} * W_{k,g} + V_{k,g} * \mathbf{y}) \tag{6}$$

0.2 Results

The main results here are evaluated on "subjective naturalness" by human evaluators. As such, WaveNets have outperformed previos TTS methods. That's not really that interesting but it makes for a cool listen: here

1 Paper 2: Vector Quantized VAE (VQ-VAE)

Title: Neural Discrete Representation Learning [**?**].

Main points:

- Avoiding Posterior Collapse
- Discrete Latent
- With the right prior, generates speech/audio well
- Language Learning through raw speech
- Speaker Conversion

The main point of the VQ-VAE lies in that it uses a discrete (i.e. categorical) embedding space as its latent space.

1.1 Model Components and Architecture

The model is described in fig. 4. What's very important to realize is that **the VQ-VAE is deterministic!**

1.2 Discrete Latent Embedding Space

The VQ VAE uses a D-dimensional latent space with K embedding vectors. This means that the latent space **does not sample from a latent space** (so it's not a real VAE) but instead **does a nearest-neighbor embedding lookup**. We define the embedding vectors as

$$e_i \in \mathbb{R}^D, i \in \{1..K\}, \therefore e \in \mathbb{R}^{K \times D}$$

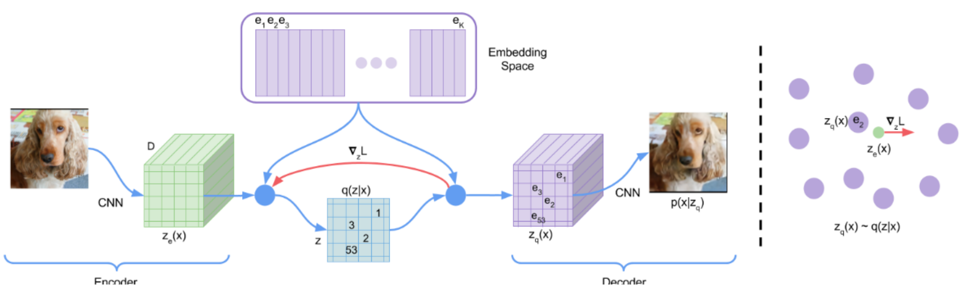

Figure 4: VQ-VAE Architecture. Right: Embedding space. Left: Overall Architecture. Note that the CNN works as a down/upsampling CNN, so as to avoid identity operations all the way through.

1.3 Loss function

The loss function eq. (7) is composed of 3 parts, here covered in detail:

$$L = \log p(x|z_q(x)) + ||\text{sg}[z_e(x)] - e||_2^2 + ||z_e(x) - \text{sg}[e]||_2^2 \tag{7}$$

We denote $z_e(x)$ as the **encoder output** and $z_q(x)$ as the **quantized encoding.** We also use sg to denote the stopgradient operator (identity function forward but 0 partial derivatives).

Reconstruction Loss

$$\log p(x|z_q(x)) +$$

This refers to the probability of the data x given the embedding $z_q(x)$.

The reconstruction loss is optimized by the **encoder** and **decoder**.

Embedding Loss

$$||\text{sg}[z_e(x)] - e||_2^2$$

The embedding loss quantifies how far away from the samples the embeddings are. This loss comes from Vector Quantization (VQ), a dictionary learning algorithm. The VQ objective seen here moves the discretized embedding vectors $e_i \in \mathbb{R}^D$ towards the continuous encoder outputs $z_e(x)$. This means that the model is able to learn embeddings and update them at need.

The embedding loss is optimized by the **embedding**.

Commitment Loss

$$||z_e(x) - \text{sg}[e]||_2^2$$

The commitment loss quantifies how far away from the embeddings a sample is. The embedding space is dimensionless in volume, and therefore the output of the encoder can grow arbitrarily large, while embeddings can't keep up. This equates to the encoder seeing an out-of-distribution sample and encoding it as very far away from the latent space embeddings.

The commitment loss is optimized by the **encoder**.