WaveNet

Steve Renals
Automatic Speech Recognition – ASR Lecture 19
30 March 2017

Motivation

“Researchers usually avoid modelling raw audio because it ticks so quickly: typically 16,000 samples per second or more, with important structure at many time-scales.

“Building a completely autoregressive model, in which the prediction for every one of those samples is influenced by all previous ones (in statistics-speak, each predictive distribution is conditioned on all previous observations), is clearly a challenging task.”
WaveNet Approach

- Generative model operating directly on the raw waveform

\[ p(x) = \prod_{t=1}^{T} p(x_t | x_1, x_2, \ldots, x_{t-1}) \]

- WaveNet model is probabilistic and autoregressive

- Model using a deep stack of convolutional layers

- No pooling layers – output has same dimensionality as input
In order to deal with long-range temporal dependencies needed for raw audio generation, we develop new architectures based on dilated causal convolutions, which exhibit very large receptive fields.

We show that when conditioned on a speaker identity, a single model can be used to generate different voices.

The same architecture shows strong results when tested on a small speech recognition dataset, and is promising when used to generate other audio modalities such as music.

We believe that WaveNets provide a generic and flexible framework for tackling many applications that rely on audio generation (e.g. TTS, music, speech enhancement, voice conversion, source separation).

In this paper we introduce a new generative model operating directly on the raw audio waveform. The joint probability of a waveform $x = \{x_1, ..., x_T\}$ is factorised as a product of conditional probabilities as follows:

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

Each audio sample $x_t$ is therefore conditioned on the samples at all previous timesteps. Similarly to PixelCNNs (van den Oord et al., 2016a;b), the conditional probability distribution is modelled by a stack of convolutional layers. There are no pooling layers in the network, and the output of the model has the same time dimensionality as the input. The model outputs a categorical distribution over the next value $x_t$ with a softmax layer and it is optimized to maximize the log-likelihood of the data w.r.t. the parameters. Because log-likelihoods are tractable, we tune hyperparameters on a validation set and can easily measure if the model is overfitting or underfitting.

### 2.1 Dilated Causal Convolutions

The main ingredient of WaveNet are causal convolutions. By using causal convolutions, we make sure the model cannot violate the ordering in which we model the data: the prediction $p(x_{t+1}|x_1, ..., x_t)$ emitted by the model at timestep $t$ cannot depend on any of the future timesteps $x_{t+1}, x_{t+2}, ..., x_T$ as shown in Fig. 2. For images, the equivalent of a causal convolution is a masked convolution (van den Oord et al., 2016a) which can be implemented by constructing a mask tensor and doing an elementwise multiplication of this mask with the convolution kernel before applying it. For 1-D data such as audio one can more easily implement this by shifting the output of a normal convolution by a few timesteps.

At training time, the conditional predictions for all timesteps can be made in parallel because all timesteps of ground truth $x$ are known. When generating with the model, the predictions are sequential: after each sample is predicted, it is fed back into the network to predict the next sample.
Efficiency

• Training: predictions can be made in parallel, because all timesteps of the ground truth training data $\mathbf{x}$ are known

• Generating: predictions are sequential, each predicted sample is used as part of the context for future samples

• Sequence modelling done by stacked convolutions
  • CNN more efficient that RNN (no backprop through time)
  • Many layers needed for long temporal context
  • Dilated convolutions increase the context
Dilated causal convolutions

Because models with causal convolutions do not have recurrent connections, they are typically faster to train than RNNs, especially when applied to very long sequences. One of the problems of causal convolutions is that they require many layers, or large filters to increase the receptive field. For example, in Fig. 2 the receptive field is only 5 (= #layers + filter length - 1). In this paper we use dilated convolutions to increase the receptive field by orders of magnitude, without greatly increasing computational cost.

A dilated convolution (also called ‘a trous’, or convolution with holes) is a convolution where the filter is applied over an area larger than its length by skipping input values with a certain step. It is equivalent to a convolution with a larger filter derived from the original filter by dilating it with zeros, but is significantly more efficient. A dilated convolution effectively allows the network to operate on a coarser scale than with a normal convolution. This is similar to pooling or strided convolutions, but here the output has the same size as the input. As a special case, dilated convolution with dilation 1 yields the standard convolution. Fig. 3 depicts dilated causal convolutions for dilations 1, 2, 4, and 8.

Dilated convolutions have previously been used in various contexts, e.g. signal processing (Holschneider et al., 1989; Dutilleux, 1989), and image segmentation (Chen et al., 2015; Yu & Koltun, 2016).

Stacked dilated convolutions enable networks to have very large receptive fields with just a few layers, while preserving the input resolution throughout the network as well as computational efficiency.

In this paper, the dilation is doubled for every layer up to a limit and then repeated: e.g. 1, 2, 4, …, 512, 1, 2, 4, …, 512. The intuition behind this configuration is two-fold. First, exponentially increasing the dilation factor results in exponential receptive field growth with depth (Yu & Koltun, 2016). For example each 1, 2, 4, …, 512 block has receptive field of size 1024, and can be seen as a more efficient and discriminative (non-linear) counterpart of a 1⇥1024 convolution. Second, stacking these blocks further increases the model capacity and the receptive field size.

2.2 SOFTMAX DISTRIBUTIONS

One approach to modeling the conditional distributions \( p(x_t | x_1, \ldots, x_{t-1}) \) over the individual audio samples would be to use a mixture model such as a mixture density network (Bishop, 1994) or mixture of conditional Gaussian scale mixtures (MCGSM) (Theis & Bethge, 2015). However, van den Oord et al. (2016a) showed that a softmax distribution tends to work better, even when the data is implicitly continuous (as is the case for image pixel intensities or audio sample values). One of the reasons is that a categorical distribution is more flexible and can more easily model arbitrary distributions because it makes no assumptions about their shape.

Because raw audio is typically stored as a sequence of 16-bit integer values (one per timestep), a softmax layer would need to output 65,536 probabilities per timestep to model all possible values. To make this more tractable, we first apply a \( \mu \)-law companding transformation (ITU-T, 1988) to the data, and then quantize it to 256 possible values:

\[
\text{\texttt{f}(x_t)} = \text{sign}(x_t) \times \ln (1 + \mu |x_t|/\mu) \times \ln (1 + \mu),
\]

In WaveNet dilations increase to a limit, then repeated: 1, 2, 4, …, 512, 1, 2, 4, …, 512, 1, 2, 4, …512

Each 1, 2, 4, …, 512 block has a context of 1024 – more efficient and discriminative than a 1024-convolution.
WaveNet Output

- Use a softmax distribution to model the outputs – but if sample is x is 16 bits, then we would have 65,536 outputs
  - 8-bit sample coding using μ-law compression
  - 256 outputs
- This is like a “language model” for audio samples
Residual/skip connections

Residual connections and parameterised skip connections are used throughout the network, to speed up convergence and enable training of much deeper models. In Fig. 4 we show a residual block of our model, which is stacked many times in the network.

2.3 GATED ACTIVATION UNITS

We use the same gated activation unit as used in the gated PixelCNN (van den Oord et al., 2016b):

$$z = \text{tanh}(W_f,k \ast x) \odot (W_g,k \ast x),$$

where \(\ast\) denotes a convolution operator, \(\odot\) denotes an element-wise multiplication operator, \(\sigma\) is a sigmoid function, \(k\) is the layer index, \(f\) and \(g\) denote filter and gate, respectively, and \(W\) is a learnable convolution filter. In our initial experiments, we observed that this non-linearity worked significantly better than the rectified linear activation function (Nair & Hinton, 2010) for modeling audio signals.

2.4 RESIDUAL AND SKIP CONNECTIONS

Both residual (He et al., 2015) and parameterised skip connections are used throughout the network, to speed up convergence and enable training of much deeper models. In Fig. 4 we show a residual block of our model, which is stacked many times in the network.

2.5 CONDITIONAL WAVETRETS

Given an additional input \(h\), WaveNets can model the conditional distribution \(p(x|h)\) of the audio given this input. Eq. (1) now becomes

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

By conditioning the model on other input variables, we can guide WaveNet’s generation to produce audio with the required characteristics. For example, in a multi-speaker setting we can choose the speaker by feeding the speaker identity to the model as an extra input. Similarly, for TTS we need to feed information about the text as an extra input.

We condition the model on other inputs in two different ways: global conditioning and local conditioning. Global conditioning is characterised by a single latent representation \(h\) that influences the output distribution across all timesteps, e.g. a speaker embedding in a TTS model. The activation function from Eq. (2) now becomes:

$$z = \text{tanh}(W_f,k \ast x) + W_T f,k h \odot (W_g,k \ast x) + W_T g,k h.$$
Control: Conditional WaveNets

\[ p(\mathbf{x} \mid \mathbf{h}) = \prod_{t=1}^{T} p(x_t \mid x_1, x_2, \ldots, x_{t-1}, \mathbf{h}) \]

- By conditioning the model on other variables can control the characteristics of generated audio
  - crucial for speech synthesis
  - for multi-speaker modelling, \( \mathbf{h} \) could encode speaker identity
WaveNet Generation

• Free-form speech generation
  • WaveNet conditioned on speaker identity
  • Trained on 44h speech from 109 speakers

• Text-to-speech synthesis
  • locally conditioned on linguistic features and log F0
  • trained on multispeaker data, conditioned on speaker identity
WaveNet for Speech Recognition

• Use WaveNet as learned front end to ASR neural network

• Mean pooling layer after the dilated convolutions
  • aggregate to 10ms frames (mean-pooling)
  • followed by a “few non-causal convolutions”
  • multi-task training to simultaneously predict the next sample and classify the frame

• 18.6% PER on TIMIT
The End.