ClariNet: Parallel Wave Generation in End-to-End Text-to-Speech

Wei Ping[*] **Kainan Peng**[*] **Jitong Chen**[*]
Baidu Research

Abstract

In this work, we propose an alternative solution for parallel wave generation by WaveNet. In contrast to parallel WaveNet (Oord et al., 2018), we distill a Gaussian *inverse autoregressive flow* from the autoregressive WaveNet by minimizing a novel regularized *KL divergence* between their highly-peaked output distributions. Our method computes the KL divergence in closed-form, which simplifies the training algorithm and provides very efficient distillation. In addition, we propose the first text-to-wave neural architecture for speech synthesis, which is fully convolutional and enables fast end-to-end training from scratch. It significantly outperforms the previous pipeline that connects a text-to-spectrogram model to a separately trained WaveNet (Ping et al., 2018). We also successfully distill a parallel waveform synthesizer conditioned on the hidden representation in this end-to-end model. [2]

1 Introduction

Speech synthesis, also called text-to-speech (TTS), is traditionally done with complex multi-stage hand-engineered pipelines (Taylor, 2009). Recent successes of deep learning methods for TTS lead to high-fidelity speech synthesis (Oord et al., 2016), much simpler "end-to-end" pipelines (Sotelo et al., 2017; Wang et al., 2017; Ping et al., 2018), and a single TTS model that reproduces thousands of different voices (Ping et al., 2018).

WaveNet (Oord et al., 2016) is an autoregressive generative model for waveform synthesis. It operates at a very high temporal resolution of raw audios (e.g., 24,000 samples per second). Its convolutional structure enables parallel processing at training by teacher-forcing the complete sequence of audio samples. However, the autoregressive nature of WaveNet makes it prohibitively slow at inference, because each sample must be drawn from the output distribution before it can be passed in as input at the next time-step. In order to generate high-fidelity speech in real time, one has to develop highly engineered inference kernels (e.g., Arık et al., 2017a).

Most recently, Oord et al. (2018) proposed a teacher-student framework to distill a parallel feed-forward network from an autoregressive teacher WaveNet. The non-autoregressive student model can generate high-fidelity speech at 20 times faster than real-time. To backpropagate through random samples during distillation, parallel WaveNet employs the mixture of logistics (MoL) distribution (Salimans et al., 2017) as the output distribution for teacher WaveNet, and a logistic distribution based *inverse autoregressive flow* (IAF) (Kingma et al., 2016) as the student model. It minimizes a set of losses including the *KL divergence* between the output distributions of the student and teacher networks. However, one has to apply Monte Carlo method to approximate the intractable KL divergence between the logistic and MoL distributions, which may introduce large variances in gradients, especially for highly peaked distributions, and lead to an unstable training process in practice.

[*]These authors contributed equally to this work. Correspondence to <weiping.thu@gmail.com>. Our method is named after the musical instrument clarinet, whose sound resembles human voice.

[2]Audio samples are in `https://clarinet-demo.github.io/`

In this work, we propose a novel parallel wave generation method based on the Gaussian inverse autoregressive flow. Specifically, we make the following contributions:

1. We demonstrate that a single variance-bounded Gaussian is sufficient for modeling the raw waveform in WaveNet without degradation of audio quality. Our Gaussian autoregressive WaveNet is simply trained with maximum likelihood estimation (MLE).
2. We distill a Gaussian IAF from the autoregressive WaveNet by minimizing a novel regularized KL divergence between their peaked output distributions. Our method provides closed-form estimation of KL divergence, which largely simplifies the distillation algorithm and stabilizes the training process.
3. In previous studies, “end-to-end" speech synthesis actually refers to the text-to-spectrogram models with a separate waveform synthesizer (i.e., vocoder) (Sotelo et al., 2017; Wang et al., 2017). We propose the first text-to-wave neural architecture for TTS, which is fully convolutional and enables fast end-to-end training from scratch. Our text-to-wave model significantly outperforms the separately trained pipeline (Ping et al., 2018) in naturalness.
4. We also successfully distill a parallel neural vocoder conditioned on the learned hidden representation within the end-to-end architecture. The text-to-wave model with the parallel vocoder obtains competitive results as the model with an autoregressive vocoder.

We organize the rest of this paper as follows. Section 2 discusses related work. We propose the parallel wave generation method in Section 3, and present the text-to-wave architecture in Section 4. We report experimental results in Section 5 and conclude the paper in Section 6.

2 Related Work

Neural speech synthesis has obtained the state-of-the-art results and gained a lot of attention recently. Several neural TTS systems were proposed, including Deep Voice 1 (Arık et al., 2017a), Deep Voice 2 (Arık et al., 2017b), Deep Voice 3 (Ping et al., 2018), Tacotron (Wang et al., 2017), Tacotron 2 (Shen et al., 2018), Char2Wav (Sotelo et al., 2017), and VoiceLoop (Taigman et al., 2018). Deep Voice 1 & 2 retain the traditional TTS pipeline, which has separate grapheme-to-phoneme, phoneme duration, frequency, and waveform synthesis models. In contrast, Tacotron, Deep Voice 3, and Char2Wav employ the attention based sequence-to-sequence models (Bahdanau et al., 2015), yielding more compact architectures. In the literature, these models are usually referred to as “end-to-end” speech synthesis. However, they depend on a traditional vocoder (Morise et al., 2016), the Griffin-Lim algorithm (Griffin and Lim, 1984), or a separately trained neural vocoder (Ping et al., 2018; Shen et al., 2018) to convert the predicted spectrogram to raw audio. In this work, we propose the first text-to-wave neural architecture for TTS based on Deep Voice 3 (Ping et al., 2018).

The neural network based vocoders, such as WaveNet (Oord et al., 2016) and SampleRNN (Mehri et al., 2017), play a very important role in recent advances of speech synthesis. In a TTS system, WaveNet can be conditioned on linguistic features, fundamental frequency (F_0), phoneme durations (Oord et al., 2016; Arık et al., 2017a), or the predicted mel-spectrograms from a text-to-spectrogram model (Ping et al., 2018). We test our parallel waveform synthesis method by conditioning it on mel-spectrograms.

Normalizing flows (Rezende and Mohamed, 2015; Dinh et al., 2014) are a family of stochastic generative models, in which a simple initial distribution is transformed into a more complex one by applying a series of invertible transformations. Normalizing flow provides efficient sampling and arbitrarily complex posterior distribution, making it well suited for the inference network in variational autoencoder (Kingma and Welling, 2014). Inverse autoregressive flow (IAF) (Kingma et al., 2016) is a special type of normalizing flow where each invertible transformation is based on an autoregressive neural network. Learning an IAF with maximum likelihood can be very slow. In this work, we distill a Gaussian IAF from a pretrained autoregressive generative model by minimizing a numerically stable variant of KL divergence.

Knowledge distillation is originally proposed for compressing large models to smaller ones (Bucilua et al., 2006). In deep learning (Hinton et al., 2015), a smaller student network is distilled from the teacher network by minimizing the loss between their outputs (e.g., L2 or cross-entropy). In parallel WaveNet, a non-autoregressvie student-net is distilled from a autoregressive WaveNet by minimizing the *reverse KL divergence* (Murphy, 2014). Similar techniques are applied in non-autoregressive models for machine translation (Gu et al., 2018; Kaiser et al., 2018; Lee et al., 2018; Roy et al., 2018).

3 Parallel Wave Generation

In this section, we present the Gaussian autoregressive WaveNet as the teacher-net and the Gaussian inverse autoregressive flow as the student-net. Then, we develop our density distillation algorithm.

3.1 Gaussian Autoregressive WaveNet

WaveNet models the joint distribution of high dimensional waveform $\boldsymbol{x} = \{x_1, \ldots, x_T\}$ as the product of conditional distributions using the chain rules of probability,

$$p(\boldsymbol{x} \mid \boldsymbol{c}) = \prod_{t=1}^{T} p(x_t \mid x_{<t}, \boldsymbol{c}\,;\, \boldsymbol{\theta}), \tag{1}$$

where x_t is the t-th variable of $\boldsymbol{x}$, $x_{<t}$ represent all variables before t-step, $\boldsymbol{c}$ is the global conditioner [3] (e.g., mel-spectrogram, or learned hidden representation in Section 4), and $\boldsymbol{\theta}$ are parameters of the model. The autoregressive WaveNet takes $x_{<t}$ as input, and outputs the probability distribution over x_t.

The original WaveNet treats x_t as a 256-way categorical variable (Oord et al., 2016). In practice, high fidelity audio (16-bit per sample) may require as many as 65,536 softmax units to model, which could be prohibitively expensive. Parallel WaveNet (Oord et al., 2018) advocates mixture of logistics (MoL) distribution introduced in PixelCNN++ (Salimans et al., 2017) for autoregressive teacher-net, as it requires much fewer output units. More importantly, the output distribution of student-net is required to be differentiable over random samples $\boldsymbol{x}$ and allow backpropagation from teacher-net to student-net during distillation. As a result, one needs to choose a continuous output distribution for teacher-net in order to match the student-net. Directly maximizing the log-likelihood of MoL is prone to numerical issues, and one has to employ the quantized surrogate loss in PixelCNN++.

In this work, we demonstrate that a single Gaussian output distribution for WaveNet suffices to model the raw waveform. [4] Specifically, the conditional distribution of x_t given previous samples $x_{<t}$ is,

$$p(x_t \mid x_{<t}; \boldsymbol{\theta}) = \mathcal{N}\big(\mu(x_{<t}; \boldsymbol{\theta}), \sigma(x_{<t}; \boldsymbol{\theta})\big), \tag{2}$$

where $\mu(x_{<t}; \boldsymbol{\theta})$ and $\sigma(x_{<t}; \boldsymbol{\theta})$ are mean and standard deviation predicted by the autoregressive WaveNet [5], respectively. Given observed data, we perform maximum likelihood estimation (MLE) for parameters $\boldsymbol{\theta}$. Note that if the model gives very accurate prediction of $\mu(x_{<t})$ (i.e., $\mu(x_{<t}) \approx x_t$) and it is free to minimize $\sigma(x_{<t})$, then the log-likelihood can approach to infinity. To avoid this degenerate case, we lower bound the predicted $\log \sigma(x_{<t})$ at -7 (natural logarithm) before calculating the log-likelihood.

We use the similar WaveNet architecture detailed in Arık et al. (2017a). We also employ a stack of dilated convolution blocks, where each block has 10 layers and the dilation is doubled at each layer, i.e., $\{1, 2, 4, ..., 512\}$ (see details at Appendix B). We add the output hidden states from each layer through residual connection before projecting them to the number of skip channels.

We use 80-band log-mel spectrogram as the global conditioner. To upsample the conditioner from frame-level (80 per second) to sample-level (24,000 per second), we apply two layers of transposed 2-D convolution (in time and frequency) interleaved with leaky ReLU ($\alpha = 0.4$). The upsampling strides in time are 15 and 20 for the two layers, respectively. Correspondingly, we set the 2-D convolution filter sizes as $(30, 3)$ and $(40, 3)$, where the filter sizes (in time) are doubled from strides to avoid the checkerboard artifacts (Odena et al., 2016). We also find that normalizing log-mel spectrogram to the range of [0, 1] improves the audio quality (e.g., Yamamoto, 2018).

3.2 Gaussian Inverse Autoregressive Flow (IAF)

Normalizing flows (Rezende and Mohamed, 2015; Dinh et al., 2017) map a simple initial density $q(\boldsymbol{z})$ (e.g., isotropic Gaussian distribution) into a complex one by applying an invertible transformation

[3]We may omit $\boldsymbol{c}$ for concise notations.

[4]It might raise the modeling capacity concern because we use the single Gaussian instead of mixture of Gaussians as the output distribution. We will demonstrate their comparable performance in experiments. Also, it should be noted, parallel WaveNet eventually uses a single logistic distribution as the output of student network and still obtains very good results.

[5]Indeed, the network always predicts $\log \sigma(x_{<t})$ and operates at log-scale.

$\boldsymbol{x} = f(\boldsymbol{z})$. Given f is a bijection, the distribution of $\boldsymbol{x}$ can be obtained through the change of variables formula:

$$q(\boldsymbol{x}) = q(\boldsymbol{z}) \left| \det \left(\frac{\partial f(\boldsymbol{z})}{\partial \boldsymbol{z}} \right) \right|^{-1}, \tag{3}$$

where $\det \left(\frac{\partial f(\boldsymbol{z})}{\partial \boldsymbol{z}} \right)$ is the determinant of the Jacobian and is computationally expensive to obtain in general. Inverse autoregressive flow (IAF) (Kingma et al., 2016) is a particular normalizing flow with a simple Jacobian determinant. Suppose $\boldsymbol{z}$ has the same dimension as $\boldsymbol{x}$, the transformation in IAF is based on an autoregressive neural network taking $\boldsymbol{z}$ as the input: $x_t = f(z_{\leq t}; \boldsymbol{\vartheta})$, where $\boldsymbol{\vartheta}$ are parameters of the model. Note that the t-th variable x_t only depends on previous and current latent variables $z_{\leq t}$, thus the Jacobian is a triangular matrix and the determinant is the product of the diagonal entries,

$$\det \left(\frac{\partial f(\boldsymbol{z})}{\partial \boldsymbol{z}} \right) = \prod_t \frac{\partial f(z_{\leq t})}{\partial z_t}, \tag{4}$$

which is easy to calculate. Parallel WaveNet (Oord et al., 2018) uses a logistic distribution based IAF to match its mixture of logistics (MoL) teacher.

We use the Gaussian IAF (Kingma et al., 2016) and define the transformation $x_t = f(z_{\leq t}; \boldsymbol{\vartheta})$ as:

$$x_t = z_t \cdot \sigma(z_{<t}; \boldsymbol{\vartheta}) + \mu(z_{<t}; \boldsymbol{\vartheta}), \tag{5}$$

where the shifting function $\mu(z_{<t}; \boldsymbol{\vartheta})$ and scaling function $\sigma(z_{<t}; \boldsymbol{\vartheta})$ are modeled by an autoregressive WaveNet in Section 3.1. Importantly, if we assume $z_t \sim \mathcal{N}(z_t \mid \mu_0, \sigma_0)$, it is easy to observe that x_t also follows a Gaussian distribution,

$$q(x_t \mid z_{\leq t}; \boldsymbol{\vartheta}) = \mathcal{N}\big(\mu_q, \sigma_q\big), \tag{6}$$

where $\mu_q = \mu_0 \cdot \sigma(z_{<t}; \boldsymbol{\vartheta}) + \mu(z_{<t}; \boldsymbol{\vartheta})$ and $\sigma_q = \sigma_0 \cdot \sigma(z_{<t}; \boldsymbol{\vartheta})$. Note that $\boldsymbol{x}$ is conditionally independent given a sample of latent variables $\boldsymbol{z}$, and its distribution is fully decomposed over dimension t,

$$q(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\vartheta}) = \prod_t q(x_t \mid z_{\leq t}; \boldsymbol{\vartheta}), \tag{7}$$

which enables parallel sampling and makes efficient use of computational resource like GPU. In contrast, the marginal distribution of $\boldsymbol{x}$,

$$q(\boldsymbol{x}; \boldsymbol{\vartheta}) = \int q(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\vartheta})\, q(\boldsymbol{z})\, d\boldsymbol{z}, \tag{8}$$

lacks closed-form expression and $\boldsymbol{x} = \{x_1, \ldots, x_T\}$ are highly correlated through the marginalized latents $\boldsymbol{z} = \{z_1, \ldots, z_T\}$. Thus, the IAF indeed jointly infers its output $\boldsymbol{x}$ at all time steps.

To evaluate the likelihood of observed data $\boldsymbol{x}$, we can still use the identities Eq. (3) and (4), and plug-in the transformation defined in Eq. (5), which will give us,

$$q(\boldsymbol{x}; \boldsymbol{\vartheta}) = q(\boldsymbol{z}) \left(\prod_t \sigma(z_{<t}; \boldsymbol{\vartheta}) \right)^{-1}. \tag{9}$$

However, we need the inverse transformation of Eq. (5),

$$z_t = \frac{x_t - \mu(z_{<t}; \boldsymbol{\vartheta})}{\sigma(z_{<t}, \vartheta)}, \tag{10}$$

to compute the corresponding $\boldsymbol{z}$ from $\boldsymbol{x}$, which is autoregressive and very slow. As a result, learning an IAF directly through maximum likelihood is impractical.

In general, normalizing flows require a series of transformations until the distribution $q(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\vartheta})$ reaches a desired level of complexity. First, we draw a white noise sample $\boldsymbol{z}^{(0)}$ from the isotropic Gaussian distribution $\mathcal{N}(0, I)$. Then, we repeatedly apply the transformation $z_t^{(i)} = f(z_{\leq t}^{(i-1)}; \boldsymbol{\vartheta})$ defined in Eq. (5) from $\boldsymbol{z}^{(0)} \to \ldots \boldsymbol{z}^{(i)} \to \ldots \boldsymbol{z}^{(n)}$ and we let $\boldsymbol{x} = \boldsymbol{z}^{(n)}$. We summarize this procedure in Algorithm 1. Note the parameters are not shared across different flows.

Algorithm 1 Gaussian Inverse Autoregressive Flows as Student Network

```
**Input:** $\boldsymbol{z}^{(0)} \sim \mathcal{N}(0, I)$: white noises;
        $n$: number of flows;
        $\boldsymbol{\vartheta}^{(i)}$: parameters of autoregressive WaveNet for the $i$-th flow;
**Output:**
        samples $\boldsymbol{x}$;
        output distribution $q(\boldsymbol{x} \mid \boldsymbol{z}^{(0)})$ with mean $\boldsymbol{\mu}_q$ and standard deviation $\boldsymbol{\sigma}_q$
Initialize $\boldsymbol{\mu}_z = 0, \quad \boldsymbol{\sigma}_z = 1$
**for** $i$-th flow in $[1:n]$ **do**
    Run autoregressive WaveNet $\boldsymbol{\vartheta}^{(i)}$ by taking $\boldsymbol{z}^{(i-1)}$ as input
        $\boldsymbol{\mu}[t] \leftarrow \mu(z_{<t}^{(i-1)}; \boldsymbol{\vartheta}^{(i)})$
        $\boldsymbol{\sigma}[t] \leftarrow \sigma(z_{<t}^{(i-1)}; \boldsymbol{\vartheta}^{(i)})$
    $\boldsymbol{z}^{(i)} = \boldsymbol{z}^{(i-1)} \odot \boldsymbol{\sigma} + \boldsymbol{\mu}$
    $\boldsymbol{\sigma}_z = \boldsymbol{\sigma}_z \odot \boldsymbol{\sigma}$
    $\boldsymbol{\mu}_z = \boldsymbol{\mu}_z \odot \boldsymbol{\sigma} + \boldsymbol{\mu}$
**end for**
$\boldsymbol{x} = \boldsymbol{z}^{(n)}, \quad \boldsymbol{\mu}_q = \boldsymbol{\mu}_z, \quad \boldsymbol{\sigma}_q = \boldsymbol{\sigma}_z$
*Remark*: iterating over $\log \boldsymbol{\sigma}$ in log-scale improves numerical stability in practice.
```

3.3 Knowledge Distillation

Oord et al. (2018) proposed the probability density distillation method to circumvent the difficulty of maximum likelihood learning for IAF. In distillation, the student IAF tries to match the distribution of its own sample to the distribution of such a sample under the pretrained autoregressive WaveNet. However, the KL divergence between the logistic distribution (output in student IAF) and mixture of logistics distribution (output in teacher WaveNet) is intractable, thus one has to rely on Monte Carlo method to approximate the integral. As a result, parallel WaveNet need a double-loop sampling procedure during distillation: 1) draw a white noise sample $\boldsymbol{z}$ and pass it as an input for student-net, then 2) draw multiple different samples from the output distribution of student-net to estimate the intractable KL divergence. In contrast, thanks to the Gaussian setup, our density distillation method only need to draw one white noise sample $\boldsymbol{z}$, then it provides closed-form computation of the KL divergence. Our student IAF shares the same conditioner network (layers of transposed 2-D convolution) with teacher WaveNet during distillation. [6]

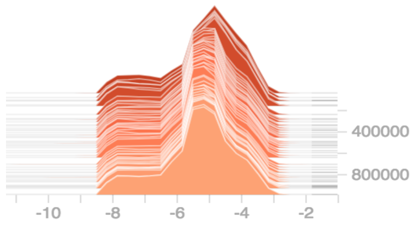
3.3.1 Regularized KL Divergence

Given a white noise sample $\boldsymbol{z}$, Algorithm 1 outputs the mapped sample $\boldsymbol{x}$, as well as the output Gaussian distribution $q(x_t \mid z_{\leq t}; \boldsymbol{\vartheta})$ with mean μ_q and standard deviation σ_q. We feed the sample $\boldsymbol{x}$ into a Gaussian autoregressive WaveNet, and obtain its output distribution $p(x_t \mid x_{<t}; \boldsymbol{\theta})$ with mean μ_p and standard deviation σ_p. One can show that the reverse KL divergence between the student output distribution $q(x_t|z_{\leq t}; \boldsymbol{\vartheta})$ and teacher $p(x_t|x_{<t}; \boldsymbol{\theta})$ has closed-form expression,

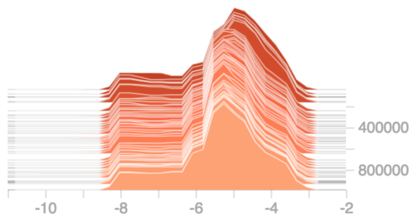
$$\mathrm{KL}\left(q \parallel p\right) = \log \frac{\sigma_p}{\sigma_q} + \frac{\sigma_q^2 - \sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_p^2}. \tag{11}$$

The detailed derivation is shown in Appendix A. We lower bound $\log \sigma_p$ and $\log \sigma_q$ at -7 before calculating the KL divergence. However, the division by σ_p^2 raises serious numerical problem, when we directly minimize the average KL divergence over all time steps. To elaborate this, we monitor the empirical histograms of σ_p from teacher WaveNet during distillation in Figure 1 (a). One can see that it is mostly distributed around (e^{-9}, e^{-2}), which incurs numerical problem if σ_p and σ_q have very different magnitudes at the beginning of training. This is because a well-trained WaveNet usually has highly peaked output distributions. The same observation holds true for other output distributions, including mixture of Gaussians and mixture of logistics.

[6]Training conditioner network of student model from scratch leads to worse result.



(a)



(b)

Figure 1: The empirical histograms of (a) $\log \sigma_p$ in teacher WaveNet and (b) $\log \sigma_q$ in student IAF during density distillation using reverse $\mathrm{KL}^{\mathrm{reg}}$ divergence.

To address this problem, we define the following variant of KL divergence:

$$\mathrm{KL}^{\mathrm{reg}}\left(q \parallel p\right) = \lambda \left| \log \sigma_p - \log \sigma_q \right|^2 + \mathrm{KL}\left(q \parallel p\right). \tag{12}$$

One can interpret the first term as regularization,[7] which largely stabilizes the optimization process by quickly matching the σ's from student and teacher models, as demonstrated in Figure 1 (a) and (b). In addition, it does not introduce any bias for matching their probability density functions, as we have the following proposition:

Proposition 3.1. *For probability distributions in the location-scale family (including Gaussian, logistic distribution etc.), the regularized KL divergence in Eq.* (12) *still satisfies the following properties: (i)* $\mathrm{KL}^{\mathit{reg}}\left(q \parallel p\right) \geq 0$, *and (ii)* $\mathrm{KL}^{\mathit{reg}}\left(q \parallel p\right) = 0$ *if and only if* $p = q$.

We also test the *forward KL divergence* for probability density distillation:

$$\mathrm{KL}\left(p \parallel q\right) = \mathbb{H}(p, q) - \mathbb{H}(p), \tag{13}$$

where $\mathbb{H}(p, q)$ is the cross entropy between teacher p and student q, and $\mathbb{H}(p)$ is the entropy of teacher model. Note that one can ignore the entropy term $\mathbb{H}(p)$ since we are optimizing student q under a pretrained teacher p, which reduces to the typical cross-entropy loss for knowledge distillation (Hinton et al., 2015). To make it numerically stable, we apply the same regularization term in Eq. (12) and observe very similar empirical distributions of $\log \sigma$ in Figure 1.

3.3.2 Spectrogram Frame Loss

In knowledge distillation, it is a common practice to incorporate an additional loss using the ground-truth dataset (e.g., Kim and Rush, 2016). Indeed, training student IAF with KL divergence loss alone will lead to whisper voices. Oord et al. (2018) advocate the *average* power loss to solve this issue, which is actually coupled with the short length of training audio clip (i.e. $0.32s$) in their experiments. As the clip length increases, the average power loss will be ineffective. Instead, we compute the frame-level loss between the output samples $\boldsymbol{x}$ from student IAF and corresponding ground-truth audio $\boldsymbol{x}_n$:

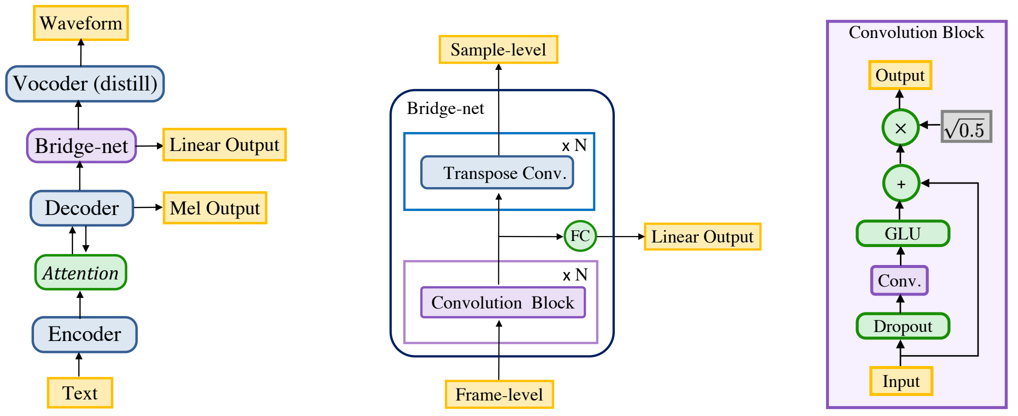
$$\frac{1}{B} \left\| \left| \mathrm{STFT}(\boldsymbol{x}) \right| - \left| \mathrm{STFT}(\boldsymbol{x}_n) \right| \right\|_2^2,$$

where $|\mathrm{STFT}(\boldsymbol{x})|$ are the magnitudes of short-term Fourier transform (STFT), and $B = 1025$ is the number of frequency bins as we set FFT size to 2048. We use a 12.5ms frame-shift, 50ms window length and Hanning window. Our final loss function is a linear combination of average KL divergence and frame-level loss, and we simply set their coefficients to one in all experiments.

4 Text-to-Wave Architecture

In this section, we present our convolutional text-to-wave architecture (see Fig. 2 (a)) for end-to-end TTS. Our architecture is based on Deep Voice 3 (DV3), a convolutional attention-based TTS system (Ping et al., 2018). DV3 is capable of converting textual features (e.g., characters, phonemes

[7]We fix $\lambda = 4$ in all experiments.



(a) Text-to-wave architecture (b) Bridge-net (c) Convolution block

Figure 2: (a) Text-to-wave model converts textual features into waveform. All components feed their hidden representation to others directly. (b) Bridge-net maps frame-level hidden representation to sample-level through several convolution blocks and transposed convolution layers interleaved with *softsign* non-linearities. (c) Convolution block is based on gated linear unit.

and stresses) into spectral features (e.g., log-mel spectrograms and log-linear spectrograms). These spectral features can be used as inputs for a separately trained waveform synthesis model, such as WaveNet. In contrast, we directly feed the hidden representation learned from the attention mechanism to the neural vocoder through some intermediate processing, and train the whole model from scratch in an end-to-end manner.

The proposed architecture consists of four components:

- **Encoder**: A convolutional encoder as in DV3, which encodes textual features into an internal hidden representation.
- **Decoder**: A causal convolutional decoder as in DV3, which decodes the encoder representation with attention into the log-mel spectrogram in an autoregressive manner.
- **Bridge-net**: A convolutional intermediate processing block, which processes the hidden representation from the decoder and predict log-linear spectrogram. Unlike the decoder, it is non-causal and can thus utilize future context. In addition, it upsamples the hidden representation from frame-level to sample-level.
- **Vocoder**: A Gaussian autoregresive WaveNet to synthesize the waveform, which is conditioned on the upsampled hidden representation from the bridge-net. This component can be replaced by a student IAF distilled from the autoregresive vocoder.

The overall objective function is a linear combination of the losses from decoder, bridge-net and vocoder; we simply set all coefficients to one in experiments. We introduce bridge-net to utilize future temporal information as it can apply non-causal convolution. All modules in our architecture are convolutional, which enables fast training [8] and alleviates the common difficulties in RNN-based models (e.g., vanishing and exploding gradient problems (Pascanu et al., 2013)). Throughout the whole model, we use the convolution block from DV3 (see Fig. 2(c)) as the basic building block. It consists of a 1-D convolution with a gated linear unit (GLU) (Gehring et al., 2017) and a residual connection. We set the dropout probability to 0.05 in all experiments. We give further details in the following subsections.

4.1 Encoder-Decoder

We use the same encoder-decoder architecture as DV3 (Ping et al., 2018). The encoder first converts characters or phonemes into trainable embeddings, followed by a series of convolution blocks to extract long-range textual information. The decoder autoregressively predicts the log-mel spectrograms with an L1 loss (teacher-forced at training). It starts with layers of 1x1 convolution to

[8]For example, DV3 trains an order of magnitude faster than its RNN peers.

Output Distribution	Subjective 5-scale MOS
Gaussian	4.40 ± 0.20
Mixture of Gaussians	4.38 ± 0.22
Mixture of Logistics	4.03 ± 0.27
Softmax (2048-way)	4.31 ± 0.23
Ground-truth (24 kHz)	4.54 ± 0.12

Table 1: Mean Opinion Score (MOS) ratings with 95% confidence intervals using different output distributions for autoregressive WaveNet. We use the crowdMOS toolkit (Ribeiro et al., 2011), where batches of samples from these models were presented to workers on Mechanical Turk. Since batches contain samples from all models, the results naturally induce a comparison between different models.

Distillation method	Subjective 5-scale MOS
Reverse $\mathrm{KL}^{\mathrm{reg}}$ + Frame-loss	4.16 ± 0.21
Forward $\mathrm{KL}^{\mathrm{reg}}$ + Frame-loss	4.12 ± 0.20

Table 2: Mean Opinion Score (MOS) ratings with 95% confidence intervals using different distillation objective functions for student Gaussian IAF. We use the crowdMOS toolkit as in Table 1.

preprocess the input log-mel spectrogram, and then applies a series of causal convolutions and attentions. A multi-hop attention-based alignment is learned between character embeddings and log-mel spectrograms.

4.2 Bridge-net

The hidden states of decoder are fed to the bridge-net for temporal processing and upsampling. The output hidden representation is then fed to the vocoder for waveform synthesis. Bridge-net consists of a stack of convolution blocks, and two layers of transposed 2-D convolution interleaved with *softsign* to upsample the per-timestep hidden representation from 80 per second to 24,000 per second. We use the same transposed convolution strides and filter sizes described in Section 3.1.

5 Experiment

In this section, we present several experiments to evaluate the proposed parallel wave generation method and text-to-wave architecture.

Data: We use an internal English speech dataset containing about 20 hours of audio from a female speaker with a sampling rate of 48 kHz. We downsample the audios to 24 kHz.

Autoregressive WaveNet: We first show that a single Gaussian output distribution for autoregressive WaveNet suffices to model the raw waveform. We train 20-layers WaveNets conditioned on 80-band ground-truth log-mel spectrogram with various output distributions, including single Gaussian, 10-component mixture of Gaussians (MoG), 10-component mixture of Logistics (MoL), and softmax with 2048 linearly quantized channels. We set both residual channel (dimension of the hidden state of every layer) and skip channel (the dimension to which layer outputs are projected prior to the output layer) to 128. We set the filter size of dilated convolutions to 2 for teacher WaveNet. All models share the same architecture except the output distributions, and they are trained for 500K steps using the Adam optimizer (Kingma and Ba, 2015) with batch-size 8 and 0.5s audio clips. The learning rate is set to 0.001 in the beginning and annealed by half for every 200K steps. We report the mean opinion score (MOS) for naturalness evaluation in Table 1. The results indicate that the Gaussian autoregressive WaveNet provides comparable results to MoG and softmax outputs, and outperforms MoL in our experiments. [9]

Student Gaussian IAF: We distill a 60-layer parallel student-net from the 20-layer Gaussian autoregressive WaveNet. It consists of six stacked Gaussian inverse autoregressive flows and each flow is parameterized by a 10-layer WaveNet with 128 residual and skip channels. We set the filter

[9]We find that MoL is more sensitive to architecture modifications than others in our experiments.

Method	**Subjective 5-scale MOS**
Text-to-Wave Model	4.15 ± 0.25
Text-to-Wave (distilled vocoder)	4.11 ± 0.24
DV3 + WaveNet (predicted Mel)	3.81 ± 0.26
DV3 + WaveNet (true Mel)	3.73 ± 0.24

Table 3: Mean Opinion Score (MOS) ratings with 95% confidence intervals for comparing the text-to-wave model and separately trained pipeline. We use the crowdMOS toolkit as in Table 1.

size of dilated convolutions to 3 in student WaveNet. [10] We test both the forward and reverse KL divergences combined with the frame-loss, and we simply set their combination coefficients to one in all experiments. The student models are trained for 500K steps using Adam optimizer. The learning rate is set to 0.001 in the beginning and annealed by half for every 200K steps. Surprisingly, we always find good results after only 50K steps of distillation, which perhaps benefits from the closed-form computation of KL divergence. The models are trained longer for extra improvement. We report the MOS evaluation results in Table 2. Both of these distillation methods work well and obtain comparable results. We expect further improvements by incorporating perceptual and contrastive losses introduced in Oord et al. (2018) and we will leave it for future work.

Text-to-Wave Model: We train the proposed text-to-wave model from scratch and compare it with the separately trained pipeline presented in Deep Voice 3 (DV3) (Ping et al., 2018). We use the same text preprocesssing and joint character-phoneme representation in DV3. The hyper-parameters of encoder and decoder are the same as the single-speaker DV3. The bridge-net has 6 layers of convolution blocks with input/output size of 256. The hyper-parameters of the vocoder are the same as 20-layer Gaussian autogressive WaveNet. The model is trained for 1.5M steps using Adam optimizer. The learning rate is set to 0.001 in the beginning and annealed by half for every 500K steps. We also distill a Gaussian IAF from the autoregressive vocoder within this end-to-end model. Both student IAF and autoregressive vocoder are conditioned on the upsampled hidden representation from the bridge-net. For the separately trained pipeline, we train two Gaussian autoregressive WaveNets conditioned on ground-truth mel-spectrogram and predicted mel-spectrogram from DV3, respectively. We run inference on the same unseen text as DV3 and report the MOS results in Table 3. The results demonstrate that the text-to-wave model significantly outperforms the separately trained pipeline. In addition, the text-to-wave model with a distilled parallel vocoder gives comparable result to the one with autoregressive neural vocoder. In the separately trained pipeline, training a WaveNet conditioned on predicted mel-spectrograms eases the training/test mismatch, thus slightly outperforms training with ground-truth.

6 Conclusion

In this work, we first demonstrate that a single Gaussian output distribution is sufficient for modeling the raw waveform in WaveNet without degeneration of audio quality. Then, we propose a parallel wave generation method based on Gaussian inverse autoregressive flow (IAF), in which the IAF is distilled from the autoregressive WaveNet by minimizing a novel regularized KL divergence for highly peaked distributions. In contrast to parallel WaveNet, our distillation algorithm estimates the KL divergence in closed-form and largely stabilizes the training procedure. Furthermore, we propose the first text-to-wave neural architecture for TTS, which can be trained from scratch in an end-to-end manner. Our text-to-wave architecture outperforms the separately trained pipeline and opens up the research opportunities for fully end-to-end TTS. We also demonstrate appealing results by distilling a parallel neural vocoder conditioned on the hidden representation within the end-to-end model.

Acknowledgements

We thank Yongguo Kang and Yu Gu from Baidu Speech Department for very helpful discussions.

[10]The IAFs usually need larger receptive fields than the autoregressive models; a good example for modeling Fibonacci series can be found in Appendix A.2 of Oord et al. (2018).

References

S. O. Arık, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta, and M. Shoeybi. Deep Voice: Real-time neural text-to-speech. In *ICML*, 2017a.

S. O. Arık, G. Diamos, A. Gibiansky, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou. Deep Voice 2: Multi-speaker neural text-to-speech. In *NIPS*, 2017b.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

C. Bucilua, R. Caruana, and A. Niculescu-Mizil. Model compression. In *ACM SIGKDD*, 2006.

L. Dinh, D. Krueger, and Y. Bengio. NICE: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. In *ICLR*, 2017.

J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *ICML*, 2017.

D. Griffin and J. Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1984.

J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher. Non-autoregressive neural machine translation. In *ICLR*, 2018.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ł. Kaiser, A. Roy, A. Vaswani, N. Pamar, S. Bengio, J. Uszkoreit, and N. Shazeer. Fast decoding in sequence models using discrete latent variables. *arXiv preprint arXiv:1803.03382*, 2018.

Y. Kim and A. M. Rush. Sequence-level knowledge distillation. In *EMNLP*, 2016.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improving variational inference with inverse autoregressive flow. In *NIPS*, 2016.

J. Lee, E. Mansimov, and K. Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *arXiv preprint arXiv:1802.06901*, 2018.

S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio. SampleRNN: An unconditional end-to-end neural audio generation model. In *ICLR*, 2017.

M. Morise, F. Yokomori, and K. Ozawa. WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, 2016.

K. Murphy. Machine learning, a probabilistic perspective, 2014.

A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, et al. Parallel WaveNet: Fast high-fidelity speech synthesis. In *ICML*, 2018.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 2013.

W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller. Deep Voice 3: Scaling text-to-speech with convolutional sequence learning. In *ICLR*, 2018.

D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *ICML*, 2015.

F. Ribeiro, D. Florêncio, C. Zhang, and M. Seltzer. CrowdMOS: An approach for crowdsourcing mean opinion score studies. In *ICASSP*, 2011.

A. Roy, A. Vaswani, A. Neelakantan, and N. Parmar. Theory and experiments on vector quantized autoencoders. *arXiv preprint arXiv:1805.11063*, 2018.

T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. PixelCNN++: Improving the pixelCNN with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017.

J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, et al. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. In *ICASSP*, 2018.

J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio. Char2wav: End-to-end speech synthesis. *ICLR workshop*, 2017.

Y. Taigman, L. Wolf, A. Polyak, and E. Nachmani. Voiceloop: Voice fitting and synthesis via a phonological loop. In *ICLR*, 2018.

P. Taylor. *Text-to-Speech Synthesis*. Cambridge University Press, 2009.

A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelCNN decoders. In *NIPS*, 2016.

Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous. Tacotron: Towards end-to-end speech synthesis. In *Interspeech*, 2017.

R. Yamamoto. WaveNet vocoder, 2018. URL `https://github.com/r9y9/wavenet_vocoder`.

Appendices

A KL Divergence between Gaussian Distributions

Given two Gaussian distributions $p(x) = \mathcal{N}(\mu_p, \sigma_p)$ and $q(x) = \mathcal{N}(\mu_q, \sigma_q)$, their KL divergence is:

$$\text{KL}\left(q \parallel p\right) = \int q(x) \log \frac{q(x)}{p(x)} dx = \mathbb{H}(q,p) - \mathbb{H}(q)$$

where $\log \equiv \log_e$, the entropy,

$$\begin{aligned}
\mathbb{H}(q) &= -\int q(x) \log q(x) dx \\
&= -\int q(x) \log \left[(2\pi\sigma_q^2)^{-\frac{1}{2}} \exp\big(-\frac{(x-\mu_q)^2}{2\sigma_q^2} \big) \right] dx \\
&= \frac{1}{2} \log\left(2\pi\sigma_q^2\right) \int q(x) dx \; + \; \frac{1}{2\sigma_q^2} \int q(x)(x-\mu_q)^2 dx \\
&= \frac{1}{2} \log\left(2\pi\sigma_q^2\right) \cdot 1 \; + \; \frac{1}{2\sigma_q^2} \cdot \sigma_q^2 \\
&= \frac{1}{2} \log\left(2\pi\sigma_q^2\right) + \frac{1}{2}
\end{aligned}$$

and the cross entropy,

$$\begin{aligned}
\mathbb{H}(q,p) &= -\int q(x) \log p(x) dx \\
&= -\int q(x) \log \left[(2\pi\sigma_p^2)^{-\frac{1}{2}} \exp\big(-\frac{(x-\mu_p)^2}{2\sigma_p^2} \big) \right] dx \\
&= \frac{1}{2} \log\left(2\pi\sigma_p^2\right) \int q(x) dx \; + \; \frac{1}{2\sigma_p^2} \int q(x)(x-\mu_p)^2 dx \\
&= \frac{1}{2} \log\left(2\pi\sigma_p^2\right) \; + \; \frac{1}{2\sigma_p^2} \int q(x)(x^2 - 2\mu_p x + \mu_p^2) dx \\
&= \frac{1}{2} \log\left(2\pi\sigma_p^2\right) \; + \; \frac{\mu_q^2 + \sigma_q^2 - 2\mu_p\mu_q + \mu_p^2}{2\sigma_p^2} \\
&= \frac{1}{2} \log\left(2\pi\sigma_p^2\right) \; + \; \frac{\sigma_q^2 + (\mu_p - \mu_q)^2}{2\sigma_p^2}.
\end{aligned}$$

Combining $\mathbb{H}(q)$ and $\mathbb{H}(q,p)$ together, we obtain

$$\text{KL}\left(q \parallel p\right) = \log \frac{\sigma_p}{\sigma_q} + \frac{\sigma_q^2 - \sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_p^2}.$$

B Details of Dilated Convolution Block

In dilated convolution block, we compute the i-th hidden layer $\boldsymbol{h}^{(i)}$ with dialation 2^{i-1} by gated convolutions (van den Oord et al., 2016):

$$\boldsymbol{h}^{(i)} = \text{sigmoid}(W_g^{(i)} * \boldsymbol{h}^{(i-1)} + A_g^{(i)} \cdot \boldsymbol{c} + b_g^{(i)}) \odot \tanh(W_f^{(i)} * \boldsymbol{h}^{(i-1)} + A_f^{(i)} \cdot \boldsymbol{c} + b_f^{(i)}),$$

therein $\boldsymbol{h}^0 = \boldsymbol{x}$ is the input of the block, $*$ denotes the causal dilated convolution, $\cdot$ represents 1×1 convolution over the *upsampled* conditioner $\boldsymbol{c}$, $\odot$ denotes the element-wise multiplication, $W_g^{(i)}, A_g^{(i)}, b_g^{(i)}$ are convolutions and bias parameters at i-th layer for *sigmoid* gating function, and $W_f^{(i)}, A_f^{(i)}, b_f^{(i)}$ are analogous parameters for *tanh* function.