CLARI\textsc{Net}: Parallel Wave Generation in End-to-End Text-to-Speech

Anonymous authors
Paper under double-blind review

Abstract

In this work, we propose a new solution for parallel wave generation by WaveNet. In contrast to parallel WaveNet (van Oord et al., 2018), we distill a Gaussian inverse autoregressive flow from the autoregressive WaveNet by minimizing a 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 introduce 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. ¹

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 (van Oord et al., 2016a), 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 (van Oord et al., 2016a) 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, van 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 per-time-step KL divergence between the logistic and MoL distributions, which may introduce large variances in gradients for highly peaked distributions, and lead to an unstable training in practice.

In this work, we propose a novel parallel wave generation method based on the Gaussian inverse autoregressive flow (IAF). 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. In contrast to the quantized surrogate loss (Salimans et al., 2017) in parallel WaveNet, our Gaussian autoregressive WaveNet is simply trained with maximum likelihood estimation (MLE).

¹Audio samples are in https://clarinet-demo.github.io/
2. We distill a Gaussian IAF from the autoregressive WaveNet by minimizing a 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 introduce the first text-to-wave neural architecture for TTS, which is fully convolutional and enables fast end-to-end training from scratch. In our architecture, the WaveNet module is conditioned on the hidden states instead of spectrograms as in previous work (Shen et al., 2018; Arik et al., 2017b), which is crucial to the success of 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 (Arik et al., 2017a), Deep Voice 2 (Arik 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 actually depend on a traditional vocoder (Morise et al., 2016), the Griffin-Lim algorithm (Griffin & 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 (van Oord et al., 2016a) 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 (van Oord et al., 2016a; Arik 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 and learned hidden representation within the end-to-end model.

Normalizing flows (Rezende & 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 arbitrarily complex posterior distribution, making it well suited for the inference network in variational autoencoder (Kingma & 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. Thus, IAF can reuse the most successful autoregressive architecture, such as PixelCNN and WaveNet (van Oord et al., 2016b;a). 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-autoregressive student-net is distilled from an 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 knowledge distillation algorithm.

3.1 Gaussian Autoregressive WaveNet

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

\[
p(x | c : \theta) = \prod_{t=1}^{T} p(x_t | x_{<t}, c : \theta),
\]

where \( x_t \) is the \( t \)-th variable of \( x \), \( x_{<t} \) represent all variables before \( t \)-step, \( c \) is the conditioner \(^2\) (e.g., mel-spectrogram or hidden states in Section 4), and \( \theta \) are parameters of the model. The autoregressive WaveNet takes \( x_{<t} \) as input, and outputs the probability distribution over \( x_t \).

Parallel WaveNet (van 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 compared to categorical distribution (e.g., 65,536 softmax units for 16-bit audios). More importantly, the output distribution of student-net is required to be differentiable over random samples \( x \) and allow backpropagation from teacher to student in distillation. As a result, one needs to choose a continuous distribution for teacher WaveNet. Directly maximizing the log-likelihood of MoL is prone to numerical issues, and one need 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. It might raise the modeling capacity concern because we use the single Gaussian instead of mixture of Gaussians. We will demonstrate their comparable performance in experiments. Specifically, our conditional distribution of \( x_t \) given previous samples \( x_{<t} \) is,

\[
p(x_t | x_{<t}; \theta) = \mathcal{N}(\mu(x_{<t}; \theta), \sigma(x_{<t}; \theta)),
\]

where \( \mu(x_{<t}; \theta) \) and \( \sigma(x_{<t}; \theta) \) are mean and standard deviation predicted by the autoregressive WaveNet, respectively. In practice, the network predicts \( \log \sigma(x_{<t}) \) and operates at log-scale for numerical stability. Given observed data, we perform maximum likelihood estimation (MLE) for parameters \( \theta \). Note that, the model can give very accurate prediction of \( \mu(x_{<t}) \) without observation noise in \( x_t \) (i.e., \( \mu(x_{<t}) \approx x_t \)), and the log-likelihood can approach to infinity if it is free to minimize \( \sigma(x_{<t}) \). To avoid this degenerate case, we lower bound the predicted \( \log \sigma(x_{<t}) \) at \(-7\) (natural logarithm) before calculating the log-likelihood in our experiment. Smaller clipping constant (e.g., \(-8\)) may work even better. We use the similar WaveNet architecture detailed in Arık et al. (2017a) (see details in Appendix B).

3.2 Gaussian Inverse Autoregressive Flow (IAF)

Normalizing flows (Rezende & Mohamed, 2015; Dinh et al., 2017) map a simple initial density \( q(z) \) (e.g., isotropic Gaussian) into a complex one by applying an invertible transformation \( x = f(z) \). Given \( f \) is a bijection, the distribution of \( x \) can be obtained through the change of variables formula:

\[
q(x) = q(z) \left| \det \left( \frac{\partial f(z)}{\partial z} \right) \right|^{-1},
\]

where \( \det \left( \frac{\partial f(z)}{\partial z} \right) \) 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. In IAF, \( z \) has the same dimension as \( x \), and the transformation is based on an autoregressive network taking \( z \) as the input: \( x_t = f(z_{<t}; \theta) \), where \( \theta \) are parameters of the model. Note that the \( t \)-th variable \( x_t \) only depends on previous and current latent variables \( z_{<t} \), thus the Jacobian is a triangular matrix and the determinant is the product of the diagonal entries,

\[
\det \left( \frac{\partial f(z)}{\partial z} \right) = \prod_t \frac{\partial f(z_{<t})}{\partial z_t},
\]

\(^2\)We may omit \( c \) for concise notations.
We use the Gaussian IAF (Kingma et al., 2016) and define the transformation
\[ x = z + \mu \]
where the shifting function \( \mu \) which is easy to calculate. Parallel WaveNet (van Oord et al., 2018) uses a single logistic distribution
\[ q(x_t | z_{<t}) = \mathcal{N}(\mu_q[t], \sigma_q[t]) \]
for i-th flow in \([1 : n]\) do
Run autoregressive WaveNet \( \theta^{(i)} \) by taking \( z^{(i-1)} \) as input
\[
\begin{align*}
\mu[t] &\leftarrow \mu(z_{<t}; \theta^{(i)}) \\
\sigma[t] &\leftarrow \sigma(z_{<t}; \theta^{(i)}) \\
z^{(i)} &\leftarrow \mathcal{N}(\mu[t], \sigma[t]) \\
\mu_z &\leftarrow \mu_z \odot \sigma + \mu \\
\sigma_z &\leftarrow \sigma_z \odot \sigma + \mu
\end{align*}
\]
end for
\[
x = z^{(n)}, \quad \mu_q = \mu_z, \quad \sigma_q = \sigma_z
\]
Remark: iterating over log \( \sigma \) in log-scale improves numerical stability in practice.

which is easy to calculate. Parallel WaveNet (van Oord et al., 2018) uses a single 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 \( x_t = f(z_{<t}; \theta) \) as:
\[
x_t = z_t \cdot \sigma(z_{<t}; \theta) + \mu(z_{<t}; \theta),
\]
where the shifting function \( \mu(z_{<t}; \theta) \) and scaling function \( \sigma(z_{<t}; \theta) \) are modeled by an autoregressive WaveNet in Section 3.1. The IAF transformation computes \( x \) in parallel given \( z \), which makes efficient use of resource like GPU. Importantly, if we assume \( z_t \sim \mathcal{N}(z_t | \mu_0, \sigma_0) \), it is easy to observe that \( x_t \) also follows a Gaussian distribution,
\[
q(x_t | z_{<t}; \theta) = \mathcal{N}(\mu_q, \sigma_q),
\]
where \( \mu_q = \mu_0 \cdot \sigma(z_{<t}; \theta) + \mu(z_{<t}; \theta) \) and \( \sigma_q = \sigma_0 \cdot \sigma(z_{<t}; \theta) \). Note that \( x \) are highly correlated through the marginalization of latents \( z \), and the IAF jointly models \( x \) at all timesteps.

To evaluate the likelihood of observed data \( x \), we can use the identities Eq. (3) and (4), and plug-in the transformation defined in Eq. (5), which will give us,
\[
q(x; \theta) = q(z) \left( \prod_t \sigma(z_{<t}; \theta) \right)^{-1}.
\]
However, one need the inverse transformation \( f^{-1} \) of Eq. (5),
\[
z_t = \frac{x_t - \mu(z_{<t}; \theta)}{\sigma(z_{<t}; \theta)},
\]
to compute the corresponding \( z \) from the observed \( x \), which is autoregressive and slow. As a result, learning an IAF directly through maximum likelihood can be very slow.

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

### 3.3 Knowledge Distillation

#### 3.3.1 Regularized KL Divergence

van Oord et al. (2018) proposed the probability density distillation method to circumvent the difficulty of maximum likelihood learning for IAF. In distillation, the goal is to minimize the sequence-level
We lower bound \( \log \sigma_p \) in addition, it does not introduce any bias for matching their probability density functions, as we address this problem, we define the following variant of KL divergence:

\[
KL(q \parallel p) = \log \frac{\sigma_p}{\sigma_q} + \frac{\sigma_q^2 - \sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_p^2}.
\] (9)

We lower bound \( \log \sigma_p \) and \( \log \sigma_q \) at \(-7\) before calculating the KL divergence. \(^3\) However, the division by \( \sigma_q^2 \) still 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 \( \sigma_q \) 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 \( \sigma_p \) and \( \sigma_q \) 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.

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

\[
KL^{reg}(q \parallel p) = \lambda \log \sigma_q - \log \sigma_q \|^2 + KL(q \parallel p) .
\] (10)

One can interpret the first term as regularization, \(^4\) which largely stabilizes the optimization process by quickly matching the \( \sigma \)'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. (10) still satisfies the following properties: (i) \( KL^reg(q \parallel p) \geq 0 \), and (ii) \( KL^reg(q \parallel p) = 0 \) \( \text{if and only if } p = q \).

Given a sample \( z \) and its mapped \( x \), we also test the forward KL divergence between the student’s output distribution \( q(x_t|z_{<t}; \vartheta) \) and teacher’s \( p(x_t|x_{<t}; \theta) \),

\[
KL(p \parallel q) = H(p, q) - H(p),
\] (11)

\(^3\)Clipping at \(-6\) also works well and could improve numerical stability.

\(^4\)We fix \( \lambda = 4 \) in all experiments.
where $\mathbb{H}(p, q)$ is the cross entropy, 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 is similar 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. (10) 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 & Rush, 2016). Empirically, we found that training student IAF with KL divergence loss alone will lead to whisper voices. van Oord et al. (2018) advocates 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 less effective. Instead, we compute the frame-level loss between the output samples $x$ from student IAF and corresponding ground-truth audio $x_n$:

$$\frac{1}{B} \left\| \left| \text{STFT}(x) \right| - \left| \text{STFT}(x_n) \right| \right\|_2^2,$$

where $|\text{STFT}(x)|$ 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 fully 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 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 WaveNet through some intermediate processing, and train the whole model from scratch in an end-to-end manner.\(^5\)

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 information. In addition, it upsamples the hidden representation from frame-level to sample-level.
- **Vocoder**: A Gaussian autoregressive 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 autoregressive 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 \(^6\) and alleviates the common difficulties in RNN-based models (e.g., vanishing and exploding gradient problems (Pascanu et al., 2013)). Throughout the

---

\(^5\)Note that conditioning WaveNet on hidden representation is crucial to the success of training from scratch. We tried to simply connect the DV3 and mel-spectrogram conditioned WaveNet and train all parameters from scratch, but it performs worse than the separate training pipeline.

\(^6\)For example, DV3 trains an order of magnitude faster than its RNN peers.
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 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 use 80-band log-mel spectrogram as the 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
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.

<table>
<thead>
<tr>
<th>Output Distribution</th>
<th>Subjective 5-scale MOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>4.40 ± 0.20</td>
</tr>
<tr>
<td>Mixture of Gaussians</td>
<td>4.38 ± 0.22</td>
</tr>
<tr>
<td>Mixture of Logistics</td>
<td>4.03 ± 0.27</td>
</tr>
<tr>
<td>Softmax (2048-way)</td>
<td>4.31 ± 0.23</td>
</tr>
<tr>
<td>Ground-truth (24 kHz)</td>
<td>4.54 ± 0.12</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Distillation method</th>
<th>Subjective 5-scale MOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reverse KL$^{\text{reg}}$ + Frame-loss</td>
<td>4.16 ± 0.21</td>
</tr>
<tr>
<td>Forward KL$^{\text{reg}}$ + Frame-loss</td>
<td>4.12 ± 0.20</td>
</tr>
</tbody>
</table>

Student Gaussian IAF: We distill a 60-layer parallel student-net from a pre-trained 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 channels, 128 skip channels, and filter size 3 in dilated convolutions. 7 Note that the student IAF shares the same conditioner network (layers of transposed 2-D convolution) with teacher WaveNet during distillation. Training conditioner network of student model from scratch leads to worse result. 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 the Adam optimizer (Kingma & 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.

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 vocoders are the same

7We find the same result with 64 residual and skip channels afterwards.
8Optimized inference kernels (e.g. Arık et al., 2017a) can provide additional speed-up for both methods, but they are not the focus of this work.
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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Subjective 5-scale MOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text-to-Wave Model</td>
<td>4.15 ± 0.25</td>
</tr>
<tr>
<td>Text-to-Wave (distilled vocoder)</td>
<td>4.11 ± 0.24</td>
</tr>
<tr>
<td>DV3 + WaveNet (predicted Mel)</td>
<td>3.81 ± 0.26</td>
</tr>
<tr>
<td>DV3 + WaveNet (true Mel)</td>
<td>3.73 ± 0.24</td>
</tr>
</tbody>
</table>

as previous subsections. The vocoder part is trained by conditioning on sliced hidden representations corresponding to 0.5s audio clips. Other parts of model are trained on whole-length utterances. The model is trained for 1.5M steps using Adam optimizer with batch size 16. 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 trained 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. The text-to-wave model with a distilled parallel vocoder gives slightly worse result to the one with autoregressive vocoder. In the separately trained pipeline, training a WaveNet conditioned on predicted mel-spectrograms eases the training/test mismatch, thus 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 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.
REFERENCES


## 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:

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

where \( \mathbb{H}(q) = -\int q(x) \log q(x) dx \)

\[
= -\int q(x) \log \left( (2\pi\sigma_q^2)^{-\frac{1}{2}} \exp \left( -\frac{(x - \mu_q)^2}{2\sigma_q^2} \right) \right) dx
\]

\[
= \frac{1}{2} \log (2\pi\sigma_q^2) + \frac{1}{2} \int q(x) dx + \frac{1}{2\sigma_q^2} \int q(x)(x - \mu_q)^2 dx
\]

\[
= \frac{1}{2} \log (2\pi\sigma_q^2) + 1 + \frac{1}{2\sigma_q^2} \cdot \sigma_q^2
\]

and the cross entropy,

\[
\mathbb{H}(q, p) = -\int q(x) \log p(x) dx
\]

\[
= -\int q(x) \log \left( (2\pi\sigma_p^2)^{-\frac{1}{2}} \exp \left( -\frac{(x - \mu_p)^2}{2\sigma_p^2} \right) \right) dx
\]

\[
= \frac{1}{2} \log (2\pi\sigma_p^2) + \frac{1}{2\sigma_p^2} \int q(x) dx + \frac{1}{2\sigma_p^2} \int q(x)(x - \mu_p)^2 dx
\]

\[
= \frac{1}{2} \log (2\pi\sigma_p^2) + \frac{\mu_q^2 + \sigma_q^2 - 2\mu_p\mu_q + \mu_p^2}{2\sigma_p^2}
\]

\[
= \frac{1}{2} \log (2\pi\sigma_p^2) + \frac{\sigma_q^2 + (\mu_p - \mu_q)^2}{2\sigma_p^2}.
\]

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

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

### B Details of Dilated Convolution Block

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\}. We add the output hidden states from each layer through residual connection before projecting them to the number of skip channels.

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

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

where \( h^0 = x \) is the input of the block, \( \ast \) denotes the causal dilated convolution, \( \cdot \) represents \( 1 \times 1 \) convolution over the upsamped condition \( c \), \( \odot \) 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.