GENERATIVE MODELLING OF SEQUENTIAL DATA

Magnus Sletfjerding
Corti.ai
Copenhagen, Denmark
ms@corti.ai

January 17th, 2022

ABSTRACT

Autoregressive convolutional neural networks such as WaveNet are powerful models that have recently achieved state-of-the-art results on both text-to-speech tasks and language modelling. In spite of this, they have so-far been unable to generate coherent speech samples when learnt from audio alone. The original configuration of WaveNet uses repeated blocks of dilated convolutions to reach a receptive field of 300 ms. In this work, we test hypotheses relating to the role of WaveNet's receptive field in learning to unconditionally generate coherent speech when not conditioned on auxiliary signals such as text. We also examine the usefulness of the learned representations for the downstream task of automatic speech recognition. By transforming the input data to stacks of multiple audio samples per timestep, we increase the receptive field to up to 5 seconds. We find that enlarging the receptive field alone is insufficient to generate coherent samples. We also provide evidence that WaveNets create representations of speech that are helpful in downstream tasks. Finally, we find that WaveNets lack capability to model natural language and argue that this is the limiting factor for direct speech generation.

1 Introduction

Learning to generate realistic time series remains a fundamental challenge for machine learning systems. Deep Neural Networks (DNNs) have shown great promise at modeling complex datasets by learning a hierarchy of abstractions within data. [**?** **?**] Specifically, Convolutional Neural Networks (CNNs) learn convolution filters at gradually increasing scales, extending to high-level structural representations [**?** **?**]. As a result, DNNs are now commonplace for non-linear modeling relationships in both sequential and non-sequential data [**?** **?** **?** **?**].

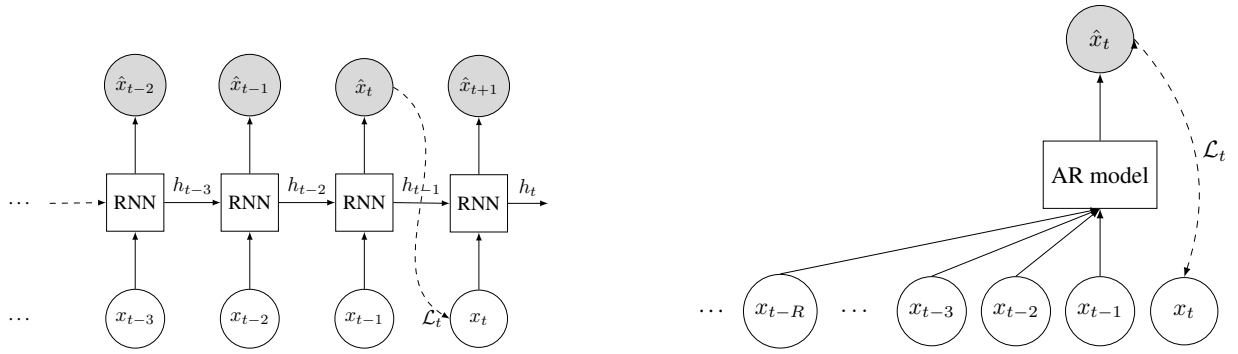
When modeling data, we optimize the model's likelihood $p(\dot{})$ over the data $\mathbf{x}$. For sequence modeling, we calculate the likelihood of each step of the data $x_t \in \mathbf{x}$ conditioned on all earlier sequence steps, i.e.

$$p(\mathbf{x}) = \prod_{t=1}^{N} p(x_t|x_{<t}), \mathbf{x} \in \mathbb{R}^N \tag{1}$$

The most commonly used models for sequential data are based on Recurrent Neural Network (RNN) units, more specifically, Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) [**?** **?** **?** **?**]. Recurrent architectures condition the likelihood in **??** on earlier timesteps through one or more "hidden states" h_t for each timestep x_t, as shown in **??**, i.e.

$$p(x_t, h_t|x_{<t}) = p(x_t|x_{t-1}, h_{t-1}) \tag{2}$$

This structure allows architectures with recurrent units to model arbitrarily long sequences with unbounded context [**?**]. However, the same recurrence causes a series of problems in training neural networks. Recurrent architectures depend on backpropagation through time (BPTT) to accurately calculate gradients. This type of backpropagation involves calculating gradients over the entire input sequence for each weight update. For a sequence of length N, the derivative of the weights with respect to the ith element in the sequence is calculated $N - i$ times, which results in an $\mathcal{O}(n^2)$



(a) Recurrent Architecture. The recurrent unit can be either a Recurrent Neural Network, a Long Short-Term Memory cell, or a Gated Recurrent Unit.

(b) Convolutional Autoregressive Architecture. The illustrated model can be any feed-forward neural network architecture.

Figure 1: Comparison between Recurrent and Autoregressive architectures for computing $\hat{x}_t$. Note that to estimate $\hat{x}_t$ accurately, the RNN needs to calculate its output multiple times, while the autoregressive model only needs to calculate its output once.

complexity for a single example [**?**]. Apart from being computationally expensive, this approach can render model training unstable, as small perturbations in the input can have a "butterfly effect" on the gradient computation - this is commonly known as vanishing and exploding gradients [**?**].

Alternative to recurrent architectures, convolutional autoregressive architectures model the likelihood in **??** by limiting the input sequence to a "receptive field" (RF) of size R. This is shown in **??**. For convolutional autoregressive models, the hidden state h_t disappears from **??**, and we can express the likelihood of a single timestep as a function of the element and the preceding R elements of the sequence:

$$p(x_t|x_{\geq(t-R-1),<t}) = f(x_t, x_{\geq(t-R-1),<t}) \tag{3}$$

This renders the likelihood of the data, **??**, as a product of these outputs, where the likelihood of each timestep can be computed independently of one another:

$$p(\mathbf{x}) = \prod_{t=R+1}^{N} p(x_t|x_{\geq(t-R-1),<t}) = \prod_{t=R+1}^{N} f(x_t, x_{\geq(t-R-1),<t}) \tag{4}$$

The "stateless" property of convolutional autoregressive models speeds up training, as the model can train on all steps of the input sequence in parallel [**?**]. However, this property limits convolutional autoregressive models compared to RNNs; autoregressive models do not natively model data dependencies longer than their receptive field. [1]

The WaveNet is a convolutional autoregressive model for generating high-fidelity audio, using dilated causal convolutions to maximize the receptive field and reduce training cost [**? ?**]. Dilated convolution layers inflate the kernel size by a dilation rate d by inserting $d-1$ "holes" between kernel elements [**?**]. WaveNets stack dilated convolution layers with exponentially increasing dilation factors, ensuring that every time step is seen once by the network, as shown in **??**. Dilated convolutional layers allow WaveNets to have exponentially growing receptive fields, so stacks of dilated convolution appear as a more lightweight alternative to regular convolutional setups with similar receptive fields [**?**]. A more detailed comparison of the number of weights is found in **??**. Reusing weights across the input also prevents gradient collapse backward in time, as the same weights are reused across the receptive field.

WaveNet's intended use is a component of larger Text-To-Speech (TTS) systems, as a strong autoregressive decoder from predicted spectrograms to high-fidelity audio. [**? ? ? ?**] Notably, the VQ-VAE uses conditioned WaveNets for decoding audio from latent representations and has successfully generated adjacent phonemes and word fragments. [**? ? ?**] Likewise, modern production systems for TTS like Tacotron 2 use WaveNet to decode sequences of Mel spectrograms and similar audio representations in place of the Griffin-Lim audio reconstruction algorithm [**?**]. On the other hand, autoregressive models have received criticism for excelling at capturing local signal structure while missing

[1] When training RNNs with truncated backpropagation through time - the truncation effectively enforces a "receptive field" of the input sequence for the sake of stability and faster convergence [**? ?**].

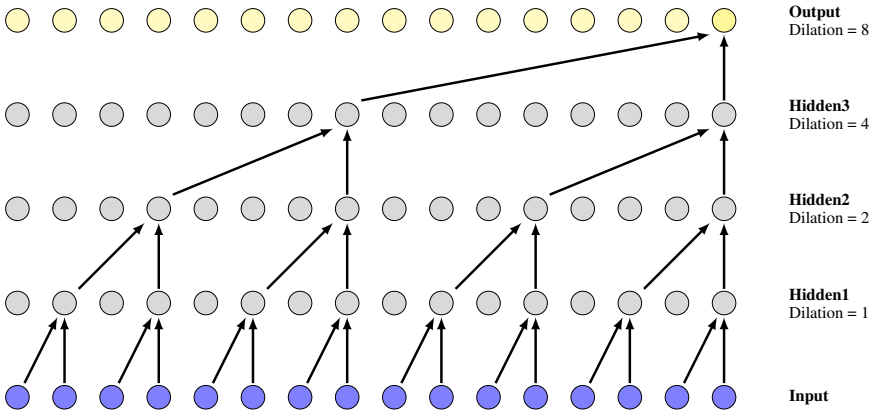


Figure 2: Illustration of a 4-layer WaveNet architecture with exponentially increasing dilation $d_i = 2^i, i \in [0, 3]$ and kernel size 2. This results in a receptive field of size $2^4 = 16$.

long-range correlations. [**?**] The original WaveNet paper discusses the same problems and attributes the low power in modeling long-range correlations to the limited receptive field of their architecture. [**? ?**] This presents a problem, as the receptive field of the WaveNet architecture is about 300 ms, significantly less than a sequence of words, much less a complete sentence.[**?**] An increased receptive field is unavoidable for a deterministic autoregressive model like WaveNet to model longer-range dependencies.

For a convolutional autoregressive model to generate realistic-sounding speech, it must contain some form of language modeling capability. Stacked convolutional layers learn arbitrarily large contexts with multiple hierarchies of abstraction, given a sufficiently large input size [**?**]. As language contains a hierarchical structure, phonemes making up words, which build phrases and construct sentences, we expect a convolutional model like WaveNet to model language well. Other convolutional autoregressive models have been reported to rival LSTM-based architectures for language modeling [**?**]. Notably, Dauphin and collaborators used stacks of gated convolutional units to calculate the language context from a "patch" of k preceding words [**?**]. As convolutional autoregressive architectures model both audio and text well, we hypothesize that a WaveNet model can produce coherent speech learned directly from audio if the receptive field is large enough.

In this work we investigate whether a WaveNet architecture learns higher-level semantic information within an audio sample. We investigate this by probing the following hypotheses:

1. WaveNet's receptive field is the main limiting factor for modeling long-range dependencies. Increasing the receptive field will increase the predictive power of the model.
2. WaveNet's stacked convolutional layers learn good representations of speech, fully or partially extracting semantic features in audio.
3. WaveNet's hierarchical structure makes it suitable to learn priors over representations of speech such as text.
4. Finally, if hypotheses **??** and **??** find support, a sufficiently large WaveNet architecture trained on speech can generate coherent words and sentence fragments.

2 Data

2.1 Datasets for self-supervised audio modeling

We use the TIMIT dataset for learning audio models using stacked WaveNets. The TIMIT dataset contains 5.4 hours of audio sampled at 16 kHz, consists of 630 speakers speaking ten sentences, and is a benchmark for Automatic Speech Recognition system [**?**]. Assuming a phoneme rate of $\approx 150/\mathrm{min}$, the TIMIT dataset contains approximately 500K tokens.

The Librispeech dataset contains 960 hours of audio from the public-domain LibriVox audiobooks [**?**]. LibriSpeech contains two "clean" splits, `clean_100` and `clean_360`, as well as a non-clean, `other_500` split. The corresponding Libri-light dataset contains over 60 000 hours of raw audio data, with a 10-hour subset of Librispeech as its labeled

`train` set. We use the Librilight dataset as a benchmark for ASR to evaluate and compare pre-trained models trained unsupervised on Librispeech.

2.2 Modality and transformation of data

The original WaveNet papers theorize that the receptive field limits the architecture from modeling full semantic content, partially due to the high sample rate used in audio datasets [**?**]. When working with high sample rates, other authors using WaveNet architectures have reported success by increasing the convolution filter size from 2 to 3 [**?**]. The authors of the Parallel WaveNet further suggest that adding more dilation stages or adding more layers to increase the receptive field of the WaveNet [**?**]. On the other hand, some probabilistic generative models for sequence modeling predict S-frame stacked vectors of audio waveforms [**? ?**]. This is equivalent to transforming $\mathbf{x}$ as below:

$$\mathbf{x}_t^* = \begin{pmatrix} x_t \\ \vdots \\ x_{t+S} \end{pmatrix}, \tag{5}$$

for $t \in \{1, S+1, \ldots, T-S\}$ and $\mathbf{x}^* \in \mathbb{R}^{N/S \times S}$ and $\mathbf{x} \in \mathbb{R}^N$.

While these models reach state-of-the-art log-likelihoods for on-step prediction, the method has met criticism for allowing models to leak a subset of x_t through the latent variable z_t [**?**]. By optimizing an output distribution

$$p_\theta(x_t|\mathbf{x}) \approx \prod_{i=1}^{L} p_\theta(x_{t,i}|\mathbf{z}_{\leq t}, \mathbf{x}_{<t}) \ ,$$

the learnt posterior $q(z_t|x_t)$ allows for conditioning on x_t. Thus the model can interpolate the remaining steps using the intra-step correlation in the output vector and artificially increase performance. [**?**] This problem does not arise in the WaveNet architecture, as the causal architecture of the model ensures that the model predicts next-step, not on-step.

2.2.1 Mu-law encoding

μ-law encoding is a signal-processing tool used to compress 16 bit PCM to 8-bit logarithmic data. The encoding is explicitly designed to work with speech, as high-frequency components dominate speech audio signals. [**?**]

b-bit μ-law encoding transforms the input x according to the following equation:

$$F(x) = \text{sign}(x) * \frac{\ln(1+\mu|x|)}{\log(1-\mu)}, \qquad \mu = 2^b \tag{6}$$

The μ-law encoding is reversible and the decoding transforms outputs as:

$$G(x) = \text{sign}(x) * \frac{\exp(|x| * \log(1-\mu))}{\mu} \tag{7}$$

For audio modeling purposes, μ-law encoding allows the model to use a broader spectrum of its output space, as illustrated in **??**. [**?**]

3 Model

3.1 WaveNet

WaveNets are fully convolutional and autoregressive models for next-step prediction, reporting state-of-the-art performance for speech and music [**?**]. WaveNets maximize the following likelihood:

$$p(\mathbf{x}) = p(x_t|x_{t-r}, ..., x_{t-1})$$

where r is the receptive field size of the network.

WaveNets use dilated causal convolutions with the last input timestep masked to enforce next-step prediction [**?**]. This forces the network to only access the information from previous timesteps [**?**]. WaveNets stack dilated convolution layers with exponentially increasing dilation factors to ensure that the network sees every timestep once. This results in receptive fields that grow exponentially with the number of layers [**?**].

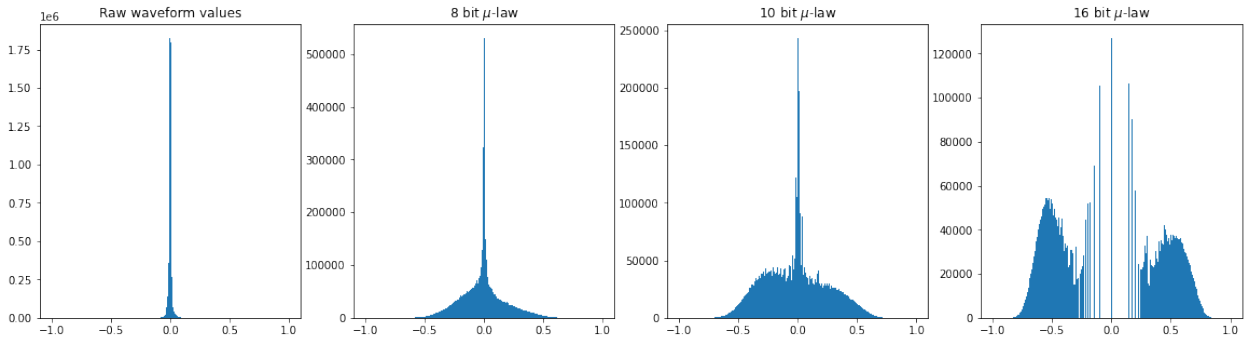


Figure 3: Illustration of the distribution of raw PCM values from the TIMIT test set which are natively 16-bit integers and the corresponding distribution after μ-law encoding the waveform values with different values of μ. At 16 bits, all areas of the spectrum are covered while not overpopulated at the extreme bins.

3.1.1 Receptive field size and stacking

The original WaveNet paper uses stacks of 10 dilated convolutions, each with a receptive field of 1024, and compares one stack to a more efficient version of a 1x1024 convolution. The original WaveNet configuration, 5 stacks of 10 layers, has a receptive field of 5117 samples, which on 16 kHz audio translates to about 300 ms [**?** **?**].

According to WaveNet's authors, the receptive field's size allows the model to learn lower-level half-syllabic sounds. Still, it stops short of producing complete word-like audio without explicitly conditioning the network on another sequential representation such as text [**?**]. This comes as no surprise, as the receptive field roughly matches the size of the duration of a phoneme. [2] By using the transformation described in **??**, we increase the receptive field of the WaveNet by a factor S while keeping the number of parameters in the model constant.

3.1.2 Output distributions in the WaveNet

In the original WaveNet paper, the authors use a 256-dimensional categorical output distribution, reasoning that they provide greater flexibility than mixture of conditional Gaussian scale mixtures (MCGSM) distributions in modeling distributions with arbitrary shapes, including multi-modal distributions [**?** **?**]. Our initial experiments showed that categorical distributions collapse to predict silence (i.e., the global mode) when predicting stacks of multiple timesteps. Instead, we use the discretized mixture of logistics (DMoL) which is widespread within image modeling and was recently used to model raw audio waveforms [**?** **?**]. The DMoL has a series of advantages over a simple categorical distribution. First, the continuous mixture model over the output space enforces an ordinality over the observed space, where numerically close values are probabilistically close as well. This allows the model to express uncertainty about the value of x_t. Second, as a mixture model, the DMoL natively models a multi-modal distributions, which aligns well with the distribution of μ-law encoded audio values seen in **??**. Empirically, we find that conditional output distributions of a trained WaveNet are often multimodal (see **??**). Finally, the DMoL is parametrized by an underlying continuous distribution over the possible values of x_t and requires fewer parameters than a categorical distribution over the same space.

3.2 Enabling larger architectures

We used PyTorch's automatic mixed-precision training to test and develop large WaveNet architectures on single GPUs. This allows for larger models to be trained while reducing memory consumption [**?**].

3.2.1 Implementing Residual and Skip connections in the WaveNet Residual Stacks

The residual block of the WaveNet includes a residual and skip output in a forward pass. One option is to feed a copy of the residual output to the skip connection. The other option is to double the size of the residual convolution layer and split the output in two. The difference between these is shown in **??**. We relied on the latter option for our implementation, as the split distinguishes information that's important for the final output layer from information that needs more processing. This is also the approach used by the original WaveNet and PixelCNN papers [**?** **?**]. The

[2] For an overview of phoneme lengths in the TIMIT dataset, see **??**.

S	N	C	RF (ms)	BPD
2	5	64	10234 (640)	**11.835**
4	5	64	20468 (1280)	12.331
8	5	64	40936 (2560)	12.621
16	5	64	81872 (5120)	13.060
2	5	92	10234 (640)	11.878
4	5	92	20468 (1280)	12.249
8	5	92	40936 (2560)	12.503
16	5	92	81872 (5120)	12.861
2	1	128	2050 (128)	12.156
4	1	128	4100 (256)	12.312
8	1	128	8200 (512)	12.740
16	1	128	16400 (1025)	13.042

Table 1: BPD for WaveNet on TIMIT Test set for 50 epochs. All Residual Stacks have 10 layers with exponentially increasing dilation, i.e. for layer i the dilation will be $d_i = 2^i, i \in 1, 10$

drawback of this approach is that each convolution layer's number of weights and operations doubles. This, in turn, limits the size of the models we can train.

3.3 Likelihood by Bits Per Dimension metric

We use bits per dimension (BPD) to compare different configurations for WaveNets, denoted as

$$b(\mathbf{x}) = -\frac{\log_2 p_\theta(\mathbf{x})}{D} \tag{8}$$

where D is the dimensionality of $\mathbf{x}$ and $\log_2 p_\theta(\mathbf{x})$ is the log-likelihood of the data $\mathbf{x}$. [**?**] Typically, D represents the sequence length of the input $\mathbf{x}$. We calculate D based on the absolute sequence length, N, i.e., scaling D by the stacking factor S.

$$D * S = \{ \begin{array}{ll} N, & N \mod S = 0 \\ N + S, & N \mod S > 0 \end{array}$$

Since $S << N$, we deem difference to be negligible in estimating comparable likelihood metrics. The use of the BPD metric allows us to easily compare models with different stacking configurations (lower is better).

4 Results

In this section we investigate the support for the hypotheses outlined in **??**. We test hypothesis **??** in **??**, then informed on those results, we examine hypothesis **??** in **??**. We then investigate hypothesis **??** in **??**, provide a discussion of hypothesis **??**, and re-investigate hypothesis **??** in **??**.

4.1 Expanding the receptive field of WaveNet by stacking

To test hypothesis **??**, that a larger receptive field will increase the predictive power of a WaveNet, we trained WaveNets with different stacking setups. The results are shown in **??**.

From these, we report the following:

Increasing the stacking does not improve likelihoods significantly. This result opposes hypothesis **??**, hinting that the information within the Receptive Field of the original WaveNet is sufficient for generating high-fidelity audio.

Increasing the number of residual channels from 64 to 92 channels **increases** evaluation likelihoods for higher stacks, but **decreases** evaluation likelihoods for 2-stacks. We infer that the limiting factor in WaveNets with larger stack sizes is the number of information channels, not the size of the receptive field.

Generated audio from the "stacked" WaveNet (can be found here) *still* sounds like babbling; we observe no significant improvement in the coherence of syllables or words.

This result hints that the WaveNet does not capture sufficient semantic information from its inputs, even with a receptive field of multiple seconds. Whether this presents a fundamental architectural limit to convolutional autoregressive models remains unknown to us at this time.

4.2 Latent space of Stacked WaveNet output space

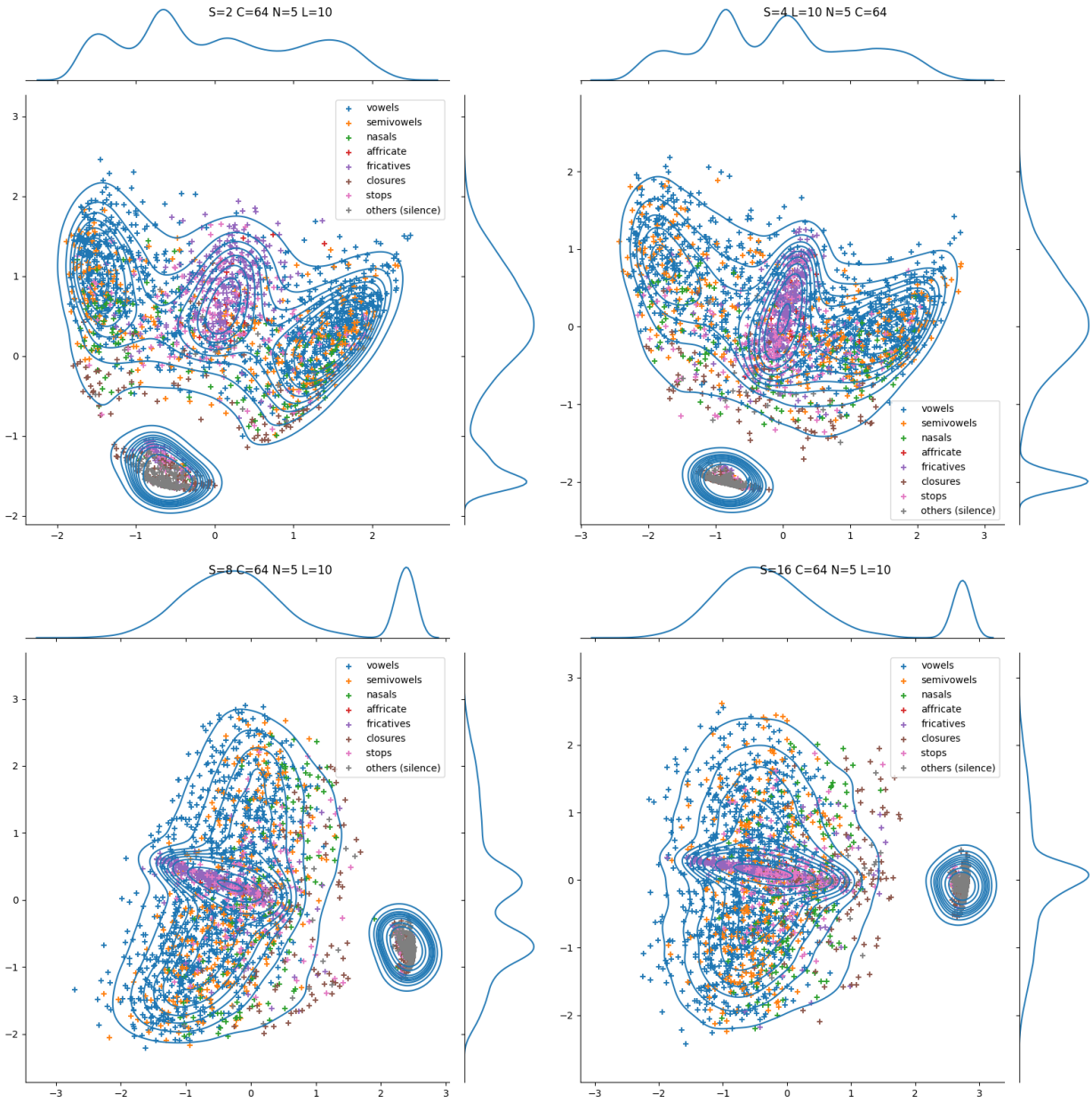


Figure 4: PCA reduction of the latent space of stacked WaveNets with different stack sizes: Top left: 2, Top right: 4, Bottom left: 8, Bottom right: 16

To assess hypothesis **??**, we examined the representations learned by WaveNets with stack sizes $S = 2, 4, 8, 16$ and $R = 64$, as reported in **??**. The representations were selected as the output from the residual stack. We pooled all samples from the test set before reducing the dimensionality to 2 dimensions with Principal Components Analysis (PCA). We plotted the density of all data points transformed with PCA in **??**. We use phoneme annotations supplied with the TIMIT dataset to overlay 2500 individual representations colored according to the phonemes. The phoneme

annotation is selected as the one with highest overlap with the corresponding input segment. If the WaveNet captures any semantic info, we expect to see phonemes cluster within the linear subspace obtained from PCA.

For stack size 2 (**??**, top left), we see four explicit nodes, with distinct groups of phonemes in each node. We observe a clearly separable node, predominantly populated by the `epi` and `pau` silence phonemes. In addition, we observe three large nodes, with vowels and nasals distributed to the two extremes, and the middle node containing most of the affricate, fricative, and stop phonemes. The same trend is observed in the larger stacks (4,8,16), although with a more smooth density as the stack size increases. We attribute the increased smoothness of the latent representations as stack size increases higher probability of phoneme overlap within the range of S frames, as S grows. Nevertheless, this property of the WaveNet supports hypothesis **??**; that WaveNet extracts semantically relevant features in audio. This builds a case for using WaveNet in semi-supervised learning, like Automatic Speech Recognition (ASR) tasks. We explore this hypothesis in **??**.

4.3 WaveNet as a Language Model

To address hypothesis **??**, we designed an experiment for character-level language modeling with the WaveNet. To be used for end2end speech generation, WaveNets need to learn to model both low-level features abd high-level features of speech. Language modeling experiments allow us to assess the suitability of a WaveNet for these higher-level modeling tasks, using text as a representation of speech. Previously, convolutional language models have surpassed first standard RNNs ([**?**]) and later RNNs with LSTMs for language modeling [**?**]. Notably, Bai and collaborators found Temporal Convolutional Networks, a modified form WaveNets, to outperform LSTMs, GRUs, and RNNs for character-level language modeling [**?**].

We implemented the WaveNet as a character-level language model and replaced the DMoL with a categorical distribution over the output tokens. We used a shallower architecture with a receptive field of 126 characters to better match the length of a typical sentence. **??** shows the results, where we observe an improvement up to 32 residual channels. Compared to other state-of-the-art models, the WaveNet remains below the bar, most notably far below the Temporal Convolutional Network, which contains a similar residual stack of dilated convolutions. This contradicts hypothesis **??**, and hints that the limiting factor in using a WaveNet for speech synthesis originates from its limits in modeling sequences of latent representations in audio.

Model	Dataset	BPD (test)
Mogrifier LSTM [**?**]	PTB	1.083
Temporal Convolutional Network [**?**]	PTB	1.31
WaveNet N=5 L=4 R=24 [RF 126]	PTB	1.835
WaveNet N=5 L=4 R=32 [RF 126]	PTB	**1.666**
WaveNet N=5 L=4 R=48 [RF 126]	PTB	1.678

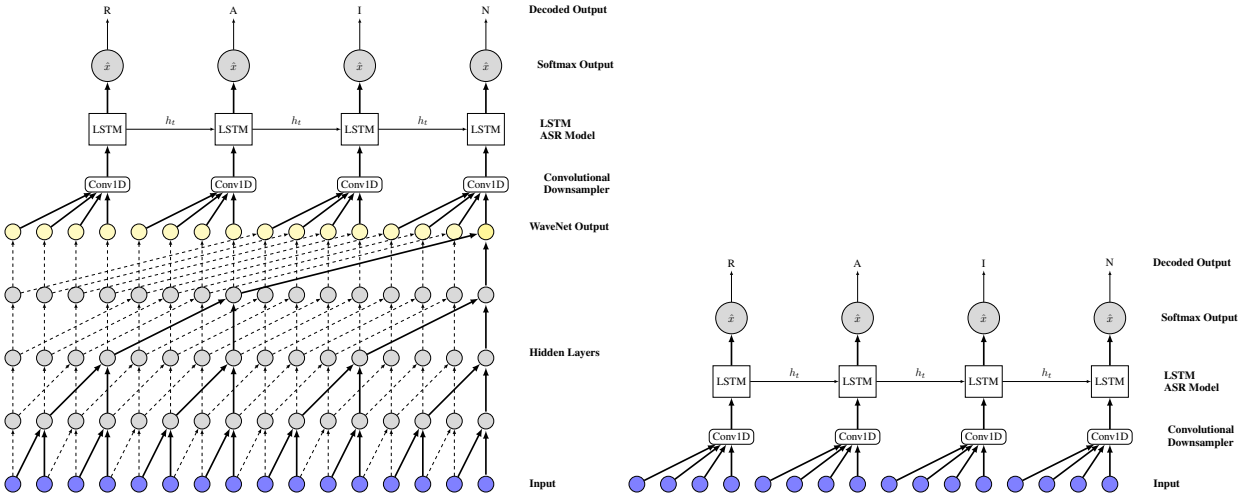
Table 2: Results of using WaveNet as a character-level language model. State-of-the-art results are shown above for comparison.

4.4 Examining the semantic content of WaveNet representations trained on raw waveforms

The bits-per-dim metric in **??** contains equal contributions from each audio waveform sample. As mentioned in **??**, WaveNets are unable to model intra-step correlation when simultaneously predicting S steps in the future. This makes errors in modeling local dependencies more likely, which result in a noisy output signal. As the BPD metric measures the likelihood of the data, it will be susceptible to give high BPD values to a noisy signal. The model's ability to model longer-term semantic information can therefore be masked by high levels of noise.

To establish whether WaveNets capture useful semantic information from raw audio (hypothesis **??**), we designed an experiment for Automatic Speech Recognition (ASR) on the outputs of trained WaveNet models. First, we trained a WaveNet model with stacking on the `clean_100` subset of the Librispeech dataset [**?**]. The model uses stacking as described in **??**, with $S = 8$, 32 residual channels, and 5 blocks of 10 layers with dilation cycles as described in **??**. This makes a receptive field of 40936 frames or 2560 ms of 16 kHz audio. The model trained until convergence and reached a BPD value of 12.368.

We then trained an LSTM-based ASR model with Connectionist Temporal Classification (CTC) loss on the outputs of the residual stack of the WaveNet, as shown in **??** [**?** **?**]. The model uses a convolution layer to downsample the input to the LSTM cell to 50 Hz. For an 8-stacked model this corresponds to a downsampling factor of 20.



(a) Setup of the WaveNet-LSTM ASR experiment.

(b) Setup of the LSTM ASR experiment.

Figure 5: Comparison of the WaveNet-LSTM and LSTM ASR experiment setups.

Model	CER (Loss) - 10min	CER (Loss) - 1h	CER (Loss) - 10h
LSTM	1.00 (1625.7)	0.691 (1463.7)	0.6050 (1131.9)
LSTM + Spectrogram	0.75 (1331.7)	0.56 (1042.2)	0.465 (825.7)
LSTM+WaveNet-10	1.0 (1611.8)	0.7066 (1457.1)	0.5122 (927.9)
LSTM+WaveNet-20	0.9965 (1615.0)	0.7969 (1527.3)	0.4618 (862.3)
LSTM+WaveNet-30	0.9983 (1634.1)	0.8681 (1502.5)	**0.4184 (800.5)**
LSTM+WaveNet-40	0.9983 (1632.0)	0.7326 (1463.5)	0.4392 (815.5)
LSTM+WaveNet-50	0.9983 (1645.8)	**0.6701 (1326.2)**	0.4757 (870.0)

Table 3: Comparisons of ASR models with and without pretrained WaveNet transformations on the input signal. Models are trained, respectively, with 10 minutes, 1 hour, and 10 hours of training data. CTC Loss and CER are reported on the Librispeech `clean-test` subset.

We train a total of seven different model configurations. The first five models train on the outputs of the first 10, 20, 30, 40, and 50 layers in WaveNet's residual stack, respectively. We also train two baselines. One baseline model trains directly on the stacked raw audio waveform (passed through a convolutional downsampling), as shown in **??**. The second is a single layered LSTM trained on 80-dimensional log-mel spectrograms with window length and hop length set to 320 (20 ms), so as to correspond to 50 Hz. This representation is commonly used for speech modeling and ASR. We train the ASR models on different amounts of subsets of the Librispeech `clean` subset - 10 minutes, 1 hour, and 10 hours, to assess the effect of using WaveNet as a preprocessor in resource settings.

This approach allows us to compare whether the information captured by the lower layers of the WaveNet are *more* valuable than the information that flows through the entire network. We evaluated all of these models against the entire `clean-test` subset for comparability. From these results, summarized in **??**, we see the following:

1. Using WaveNet as a preprocessor decreases the loss when trained on the 1 hour and 10-hour training subsets.
2. The best performance occurs when using 30 layers of the WaveNet trained on 10 hours of training data.
3. Notably, WaveNet's use as a preprocessor grows more competitive when increasing the training data size.

5 Conclusion

In this paper we studied convolutional autoregressive models and their limits for unconditional speech generation. We adapted the WaveNet to simultaneously predict multiple future timesteps by stacking audio samples in order to increase the receptive field. We found WaveNet's inability to produce semantically coherent speech samples to not be due to limited receptive field size alone. Furthermore, we provided evidence that WaveNet representations are useful as input for a downstream automatic speech recognition task. We showed that WaveNet architectures have limited language

modelling capabilities compared to similarly structured convolutional autoregressive language models, and attribute the lacking semantic coherence in speech samples to this. We expect that further refinement of the model structure (including intermediate downsampling, or variable dimension residual blocks) could allow for a WaveNet-type model to generate speech that is more semantically coherent.

Software and Data

All code used for this work is available at https://github.com/JakobHavtorn/vseq/tree/wavenet-exps

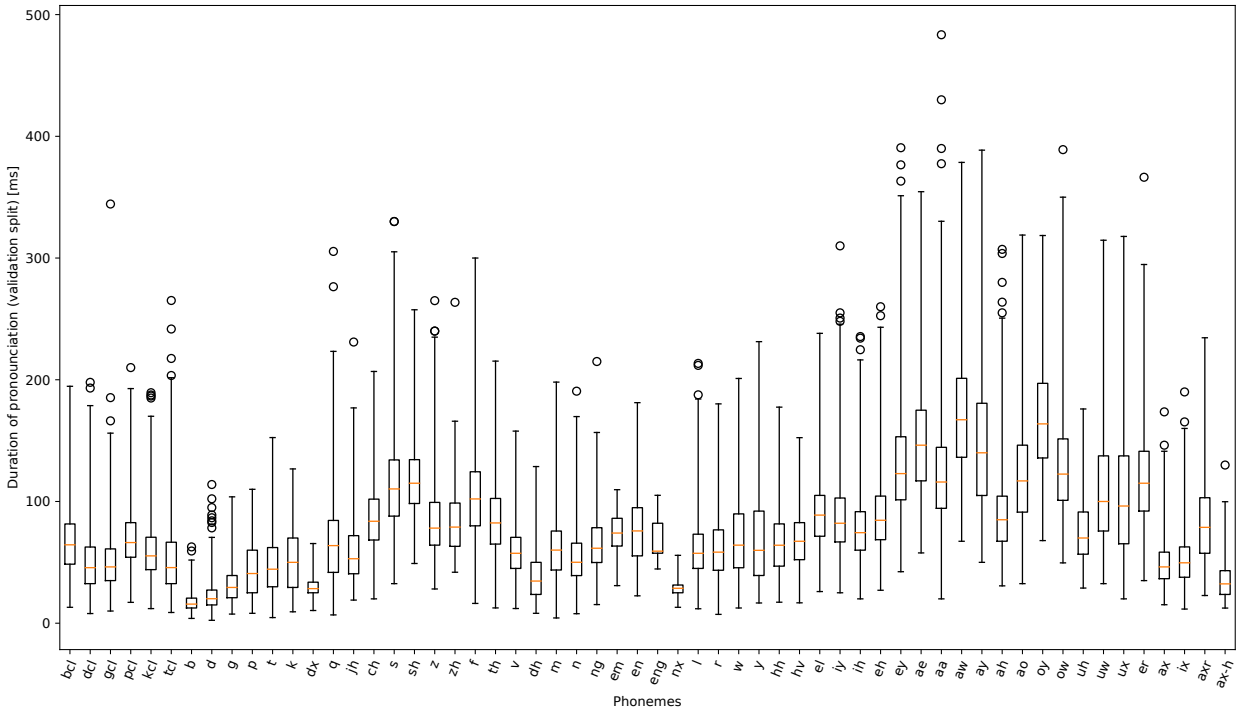


Figure 6: Boxplot of the duration of the pronunciation of phonemes in the TIMIT validation set.

A Data

A.1 Phoneme grouping in TIMIT dataset

Phonemes are fundamental units of speech, detailing the pronunciation of a word. The TIMIT dataset provides phoneme annotations in addition to text annotations, with phoneme durations in the 10-400 millisecond range. [**?**]

The Phonemes in the TIMIT dataset were grouped as described by the authors. [**?**] **??** shows the groupings used in this work. We keep the closure intervals of stops, denoted by a `*cl`, separated from the corresponding stop release, instead of keeping the groupings. The start and stop token, `h#` was omitted in phoneme mapping to latent spaces.

A.2 Phoneme lengths in the TIMIT dataset

?? shows the length of phonemes pronouced in the TIMIT validation set.

Group	Phonemes
Vowels	`iy, ih, eh, ey, ae, aa, aw, ay, ah, ao, oy, ow, uh, uw, ux, er, ax, ix, axr, ax-h`
Stops	`b, d, g, p, t, k, dx, q`
Closures	`bcl, dcl, gcl, pcl, tck, kcl, tcl`
Affricates	`jh, ch`
Fricatives	`s, sh, z, zh, f, th, v, dh`
Nasals	`m, n, ng, em, en, eng, nx`
Semivowels and Glides	`l, r, w, y, hh, hv, el`
Others	`pau, epi, h#, 1, 2`

Table 4: TIMIT phoneme groupings

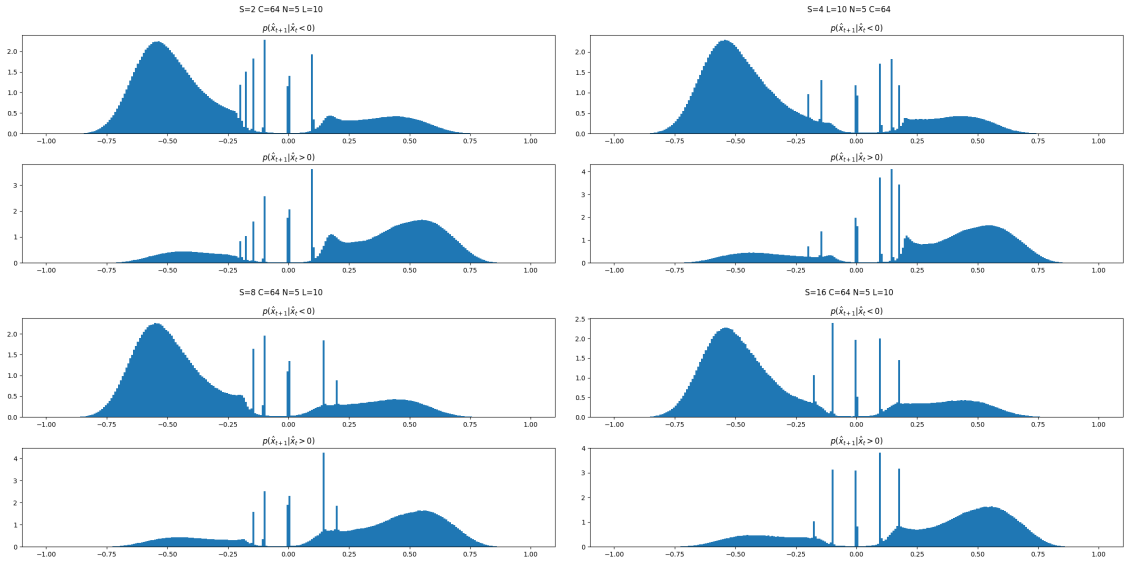


Figure 7: Sampled Output Distributions for WaveNet models trained on the Librispeech `clean-100h` subset. Distributions are in 16 bit μ-law space and binned into 256 bins from -1 to 1.

B Output Distribution of raw audio WaveNet models

In **??** we plot following output distributions for each model:

$$p(\hat{x}_{t+1}|\hat{x}_t < 0)p(\hat{x}_{t+1}|\hat{x}_t > 0)$$

Likewise, in **??** we plot a more constrained output distribution:

$$p(\hat{x}_{t+1}|\hat{x}_t \in [-0.25, 0])p(\hat{x}_{t+1}|\hat{x}_t \in [0, 0.25])$$

From these we see:

- A clear tendency to predict samples with the same sign as x_t, as expected.
- A multimodality, assigning non-zero probability to sample the opposite sign. This gives clear support that WaveNet's output distribution learns characteristic features of audio sequences, even when predicting multiple timesteps simultaneously.
- Some overespresented peaks close to 0. We attribute these to the stretching of values close to 0 from a μ-law transform. A similar phenomenon is seen when transforming the TIMIT dataset in **??**, far right.

C Calculation of weights in a dilated stack similar convolutions

C.1 Single Convolution

For a normal 1D Convolution, we calculate the number of weights, N, as:

$$N = \text{in\_channels} \cdot \text{out\_channels} \cdot \text{kernel\_size}$$

With a kernel size 1024, and in_channels $=$ out_channels $= 16$:

$$N = 16 \cdot 16 \cdot 1024 = 2^{18} = 262.144$$

C.2 Stack of non-dilated convolutions

The receptive field of a stack of 1D convolutions with the same kernel size is:

$$RF = \text{num\_layers} + \text{kernel\_size} - 1$$

For a receptive field of 1024 and a kernel size of 2, this requires num_layers $= 1023$.

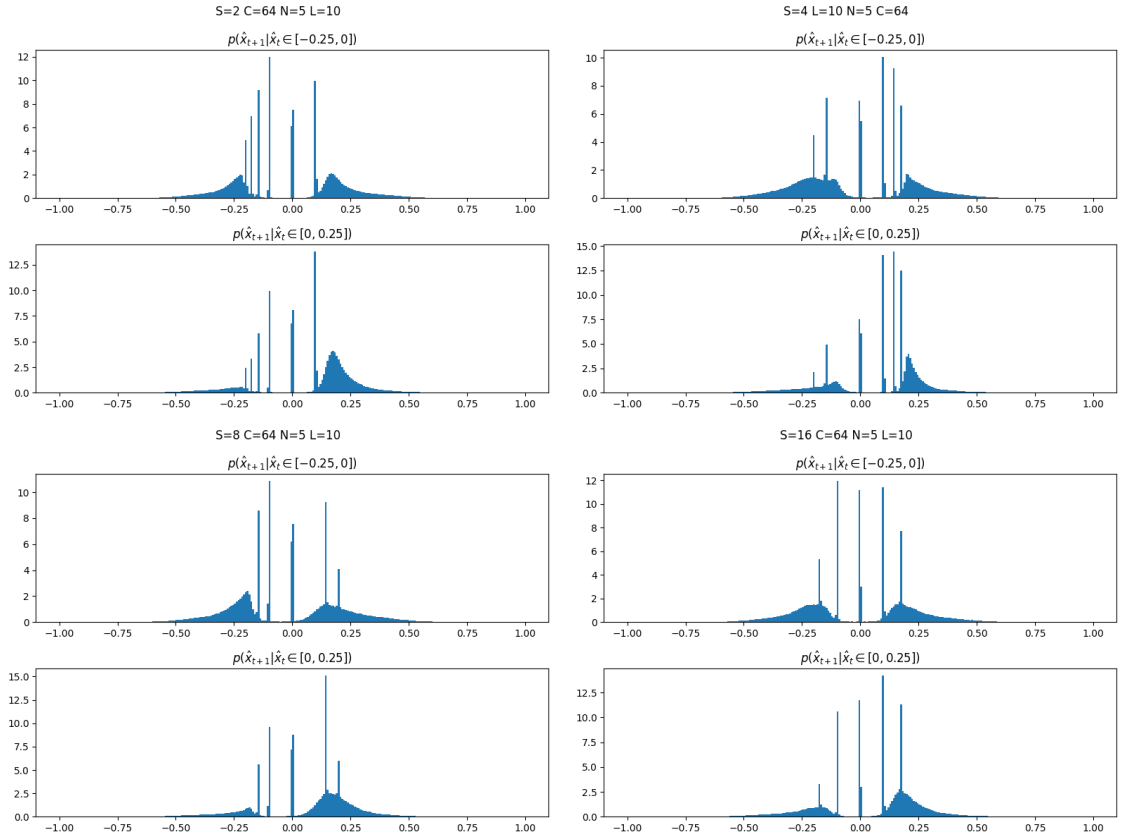


Figure 8: Sampled Output Distributions for WaveNet models trained on the Librispeech `clean-100h` subset. Distributions are in 16 bit μ-law space and binned into 256 bins from -1 to 1.

The number of parameters, N in a stack of 1D convolutions (dilated or not) is:

$$N = \text{num\_layers} * \text{in\_channels} * \text{out\_channels} * \text{kernel\_size}$$

Here kernel_size $= 2$, so:

$$N = 1023 \cdot 16 \cdot 16 \cdot 2 = 523.776$$

C.3 Dilated Stack

Calculating the receptive field of a layer l_i in a dilation stack:

$$r_{l_0} = 2$$
$$r_{l_i} = r_{l_i - 1} + (\text{kernel\_size} - 1) \cdot d_i$$

A stack of 10 dilated layers, with exponentially increasing dilation ($d_i \in \{2^0, 2^2, ..., 2^9\}$) has a receptive field of 1024:

$$r_{l_9} = 1 + \sum_{i=0}^{9} 2^i = 2^{10} = 1024$$

The number of parameters, N is calculated similarly as above:

$$N = \text{num\_layers} \cdot \text{in\_channels} \cdot \text{out\_channels} \cdot \text{kernel\_size}$$

Here kernel_size $= 2$, so:

$$N = 10 \cdot 16 \cdot 16 \cdot 2 = 10 \cdot 2^9 = \mathbf{5120}$$

With this in mind a stack of dilated convolutions is significantly more lightweight for acheiving large receptive fields, with two orders of magnitude fewer weights than a standard stack with similar receptive field.

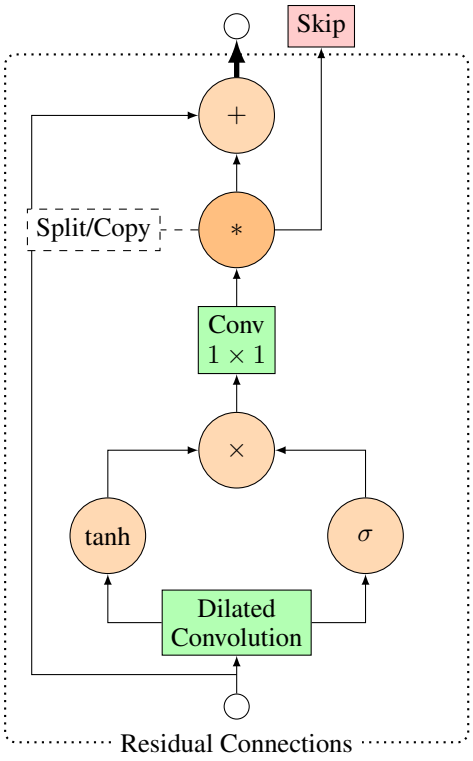


Figure 9: Overview of WaveNet's residual block. The "Split/Copy" operation, representing the two configurations of the residual block is highlighted.

D Results of WaveNet+LSTM Models for Automatic Speech Recognition on TIMIT

?? shows results from running similar ASR experiments as in **??** on the TIMIT dataset. Similarly to the results in **??**, we see that intermediate outputs of the WaveNet give the largest improvement. This supports the case that WaveNet captures semantically relevant information in its intermediate layers.

Model	CTC Loss	CER
LSTM	141.9	0.575
LSTM+WaveNet-10	144.1	0.561
LSTM+WaveNet-20	**134.3**	**0.520**
LSTM+WaveNet-30	141.8	0.616
LSTM+WaveNet-40	142.5	0.573

Table 5: Comparisons of ASR models with and without pretrained WaveNet transformations on the input signal. CTC Loss and CER are reported on the TIMIT test set.

E WaveNet's Residual block

When implementing the Residual Block (see **??**) in WaveNet, we double the size of the residual convolution layer relative to the input and split the output in two. This corresponds to implementing the "Split" mode of the residual block. The alternative, to keep the output the same size as the input, correspond to implementing the residual block in "Copy" mode, as the output from the 1x1 convolution is copied and fed to both the next residual block as well as the skip connection.