

# Deep Learning For Audio Processing

박수철

# Audio processing의 종류

## Speech

Speech Recognition

Speech Synthesis - Tacotron2, Wavenet

Speech Enhancement

## Sound

Acoustic Scene/Event Identification

## Music

Music Generation - Wavenet, Melnet, Music Transformer, Musenet

Source Separation - MMDenseNet

MIR(Music Information Retrieval)

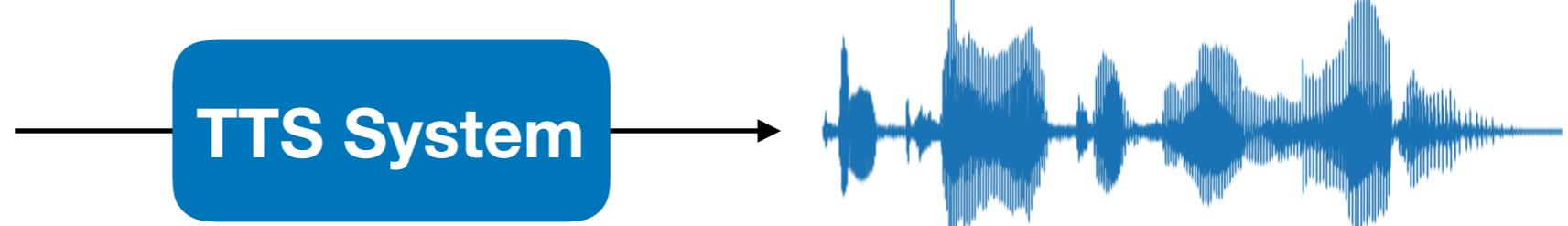
- pitch recognition
- genre classification
- tempo estimation

# Speech Synthesis (TTS, Text To Speech)

안녕하세요? 반갑습니다~

Hi! Nice to meet you.

Text 입력

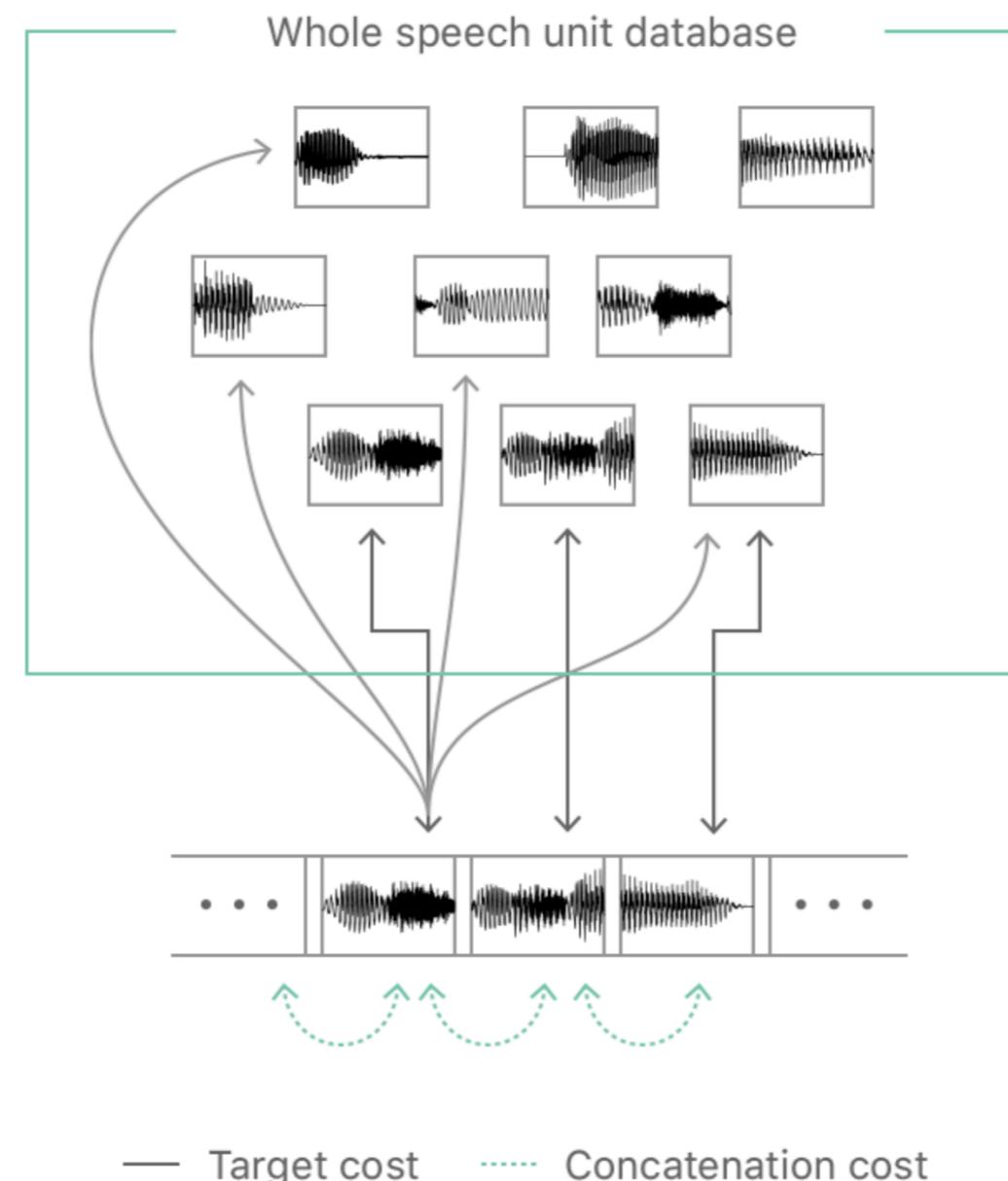


Audio 출력

# Speech Synthesis (TTS, Text To Speech)

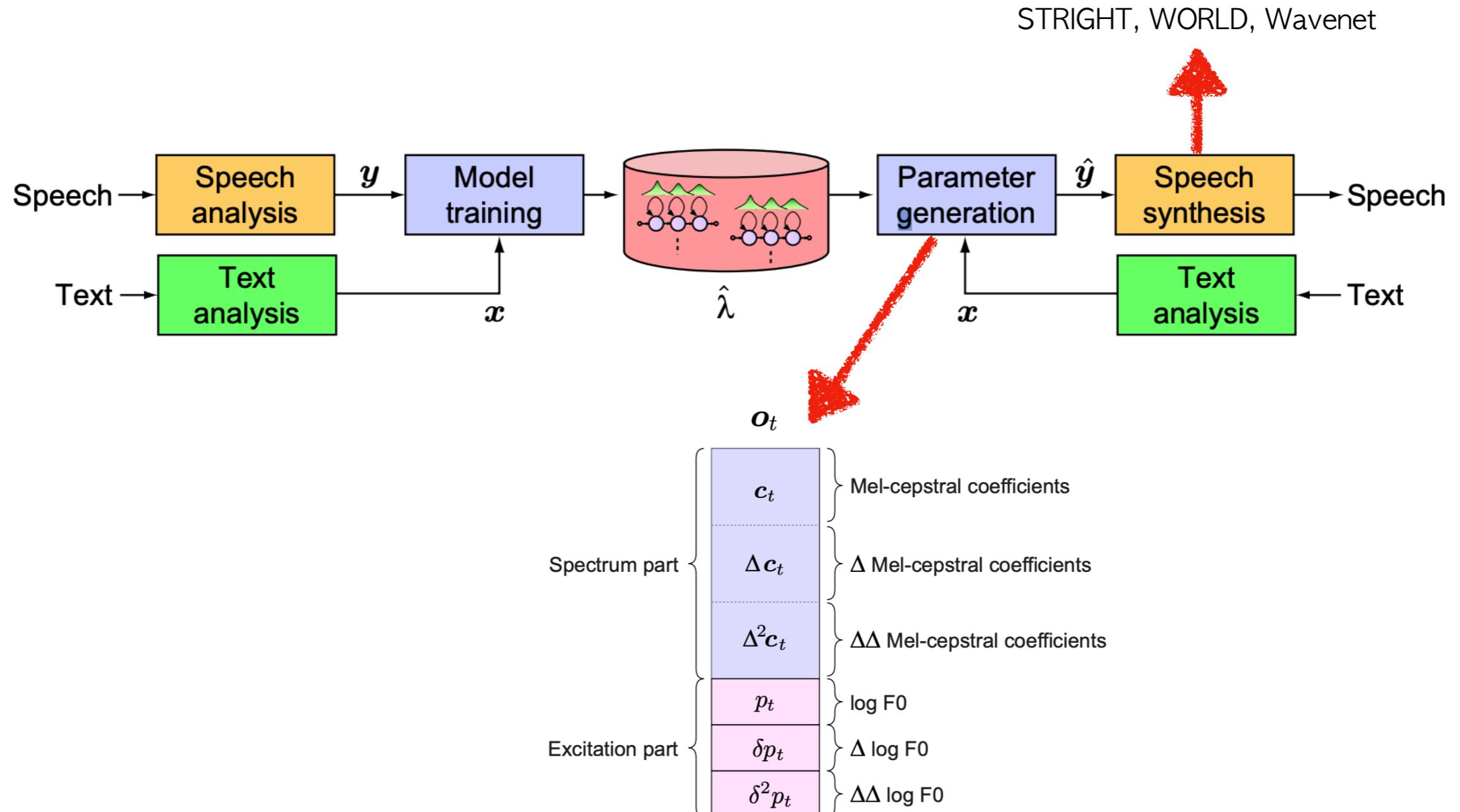
## Concatenative

Figure 4. Illustration of the unit selection method based on target and concatenation costs.



# Speech Synthesis (TTS, Text To Speech)

Parametric

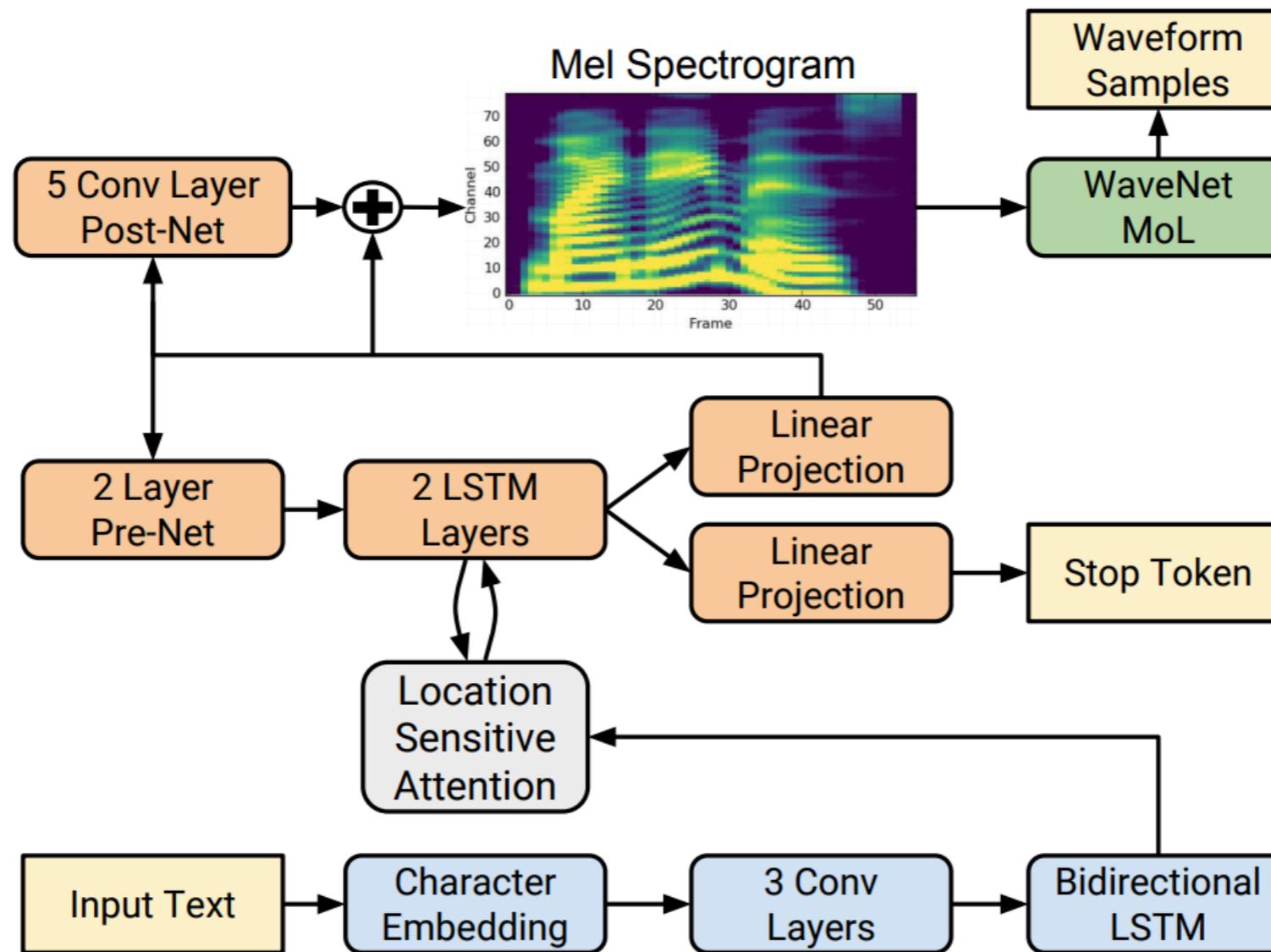


# Speech Synthesis (TTS, Text To Speech)

End-to-end

Audio Samples

<https://google.github.io/tacotron/publications/tacotron2/index.html>



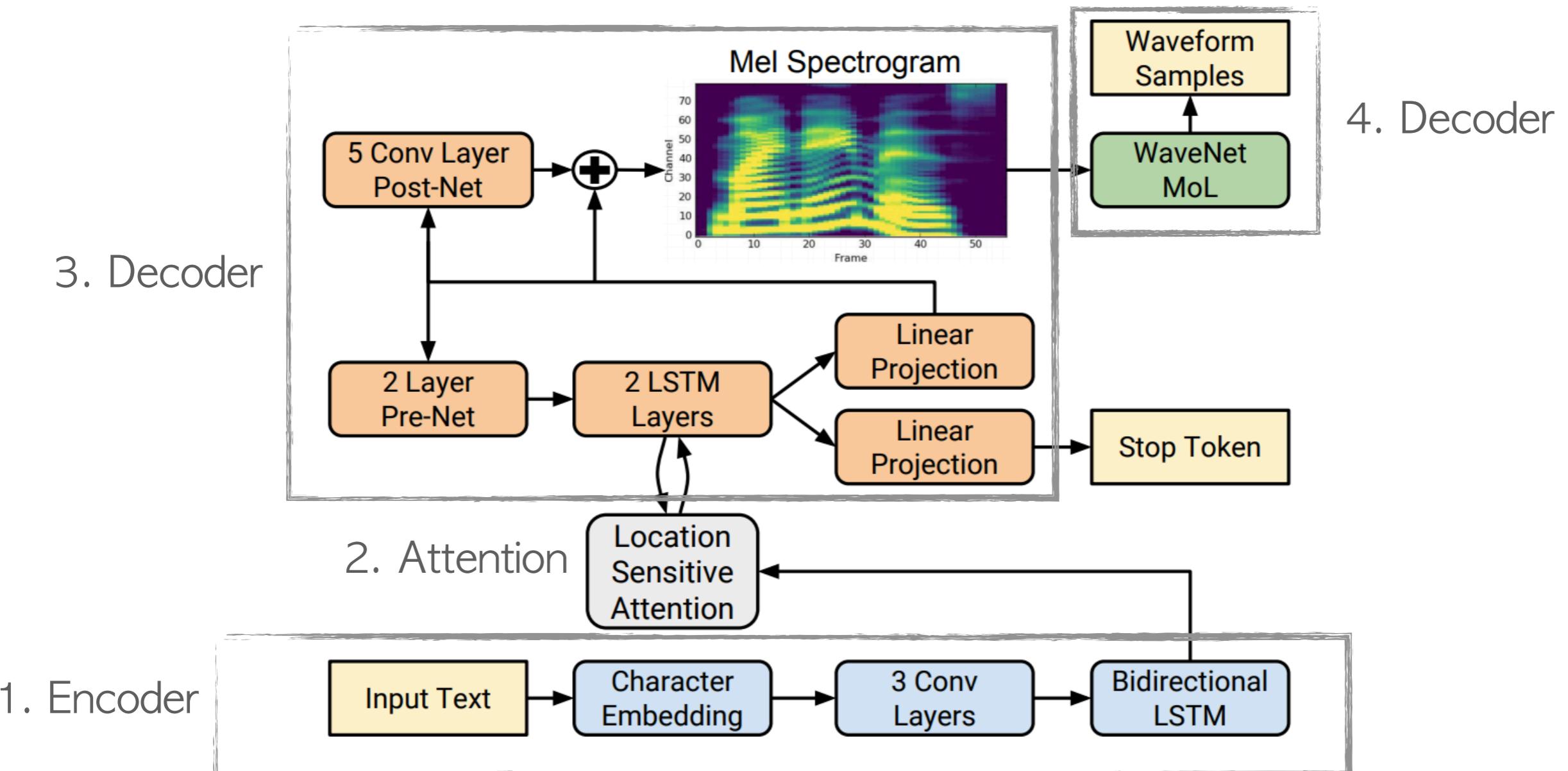
Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions  
arXiv preprint arXiv:1712.05884

# Speech Synthesis (TTS, Text To Speech)

End-to-end: Tacotron2

Audio Samples

<https://google.github.io/tacotron/publications/tacotron2/index.html>



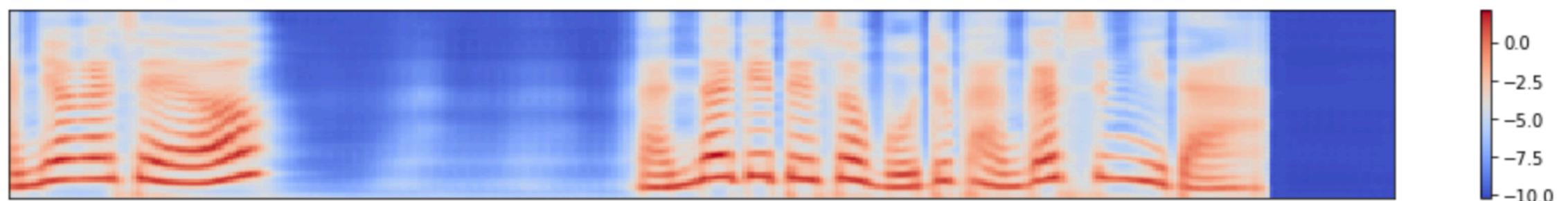
Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions  
arXiv preprint arXiv:1712.05884

# Speech Synthesis (TTS, Text To Speech)

End-to-end: Attention

안녕하세요? 만나게 돼서 모두 반갑습니다.

13, 21, 45, 4, 27, 62, 20, ...



# Speech Synthesis (TTS, Text To Speech)

## Wavenet

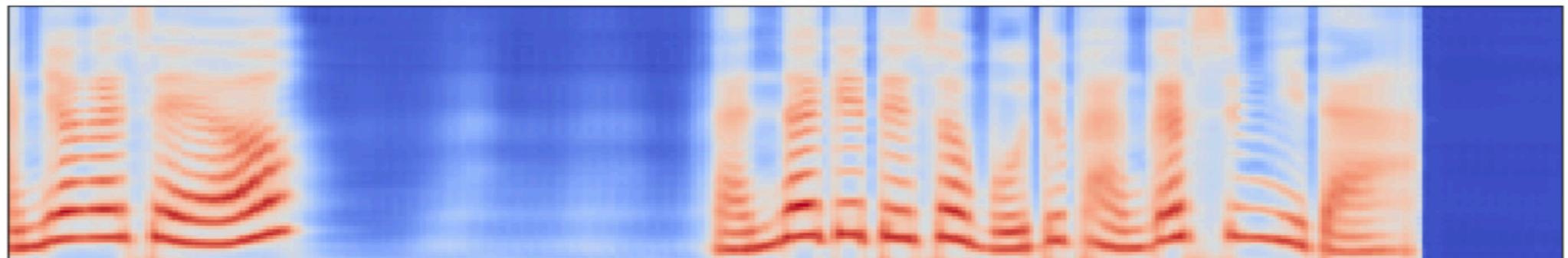
1. Unconditional Wavenet  $p(\mathbf{x}) = \prod_{t=1}^T p(x_t | x_1, \dots, x_{t-1})$

waveform



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

MelSpectrogram  
Condition



↓ Vocoding

waveform



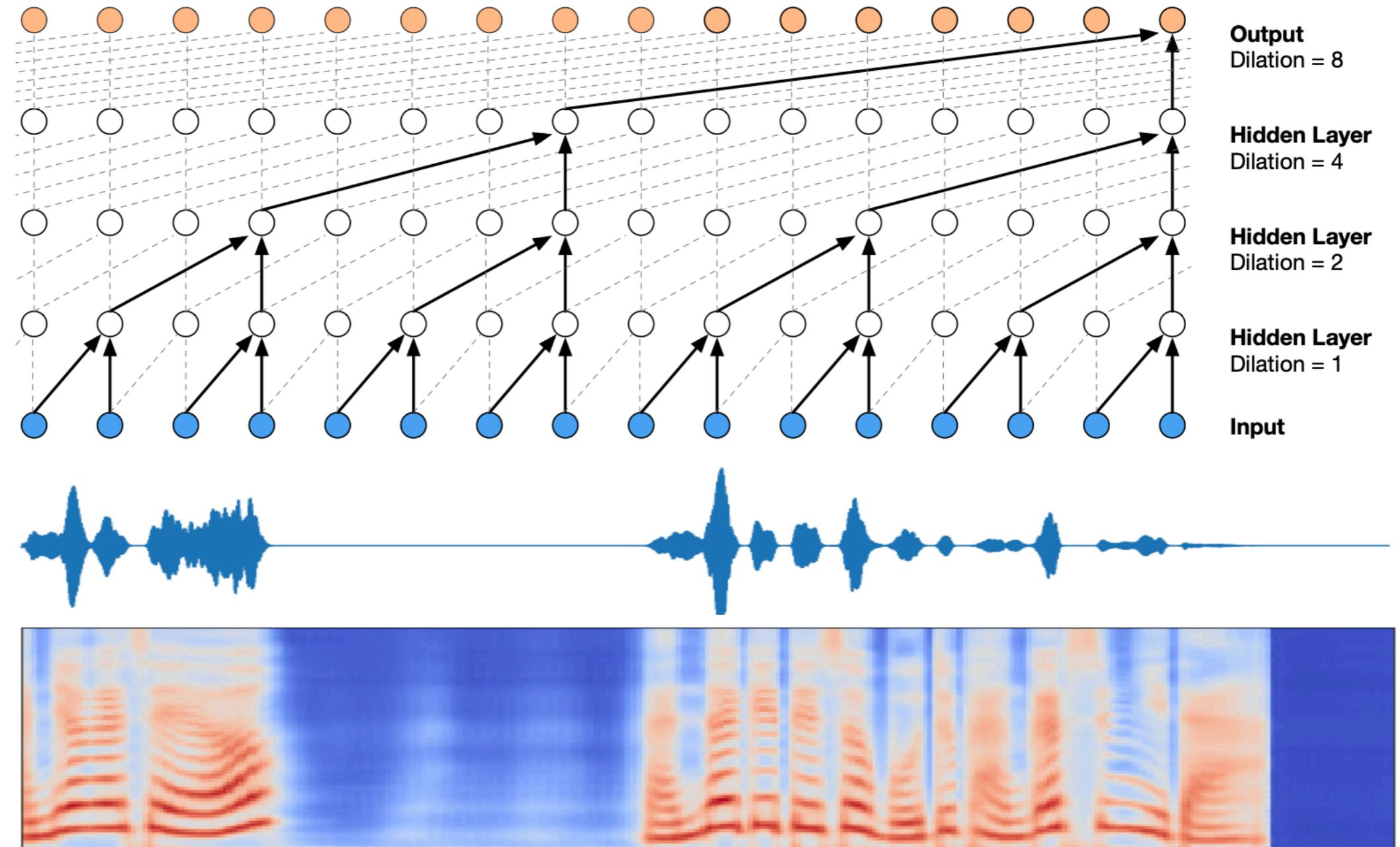
Audio Samples

<https://deepmind.com/blog/wavenet-generative-model-raw-audio>

“Wavenet: A generative model for raw audio,” arXiv:1609.03499, 2016.

# Speech Synthesis (TTS, Text To Speech)

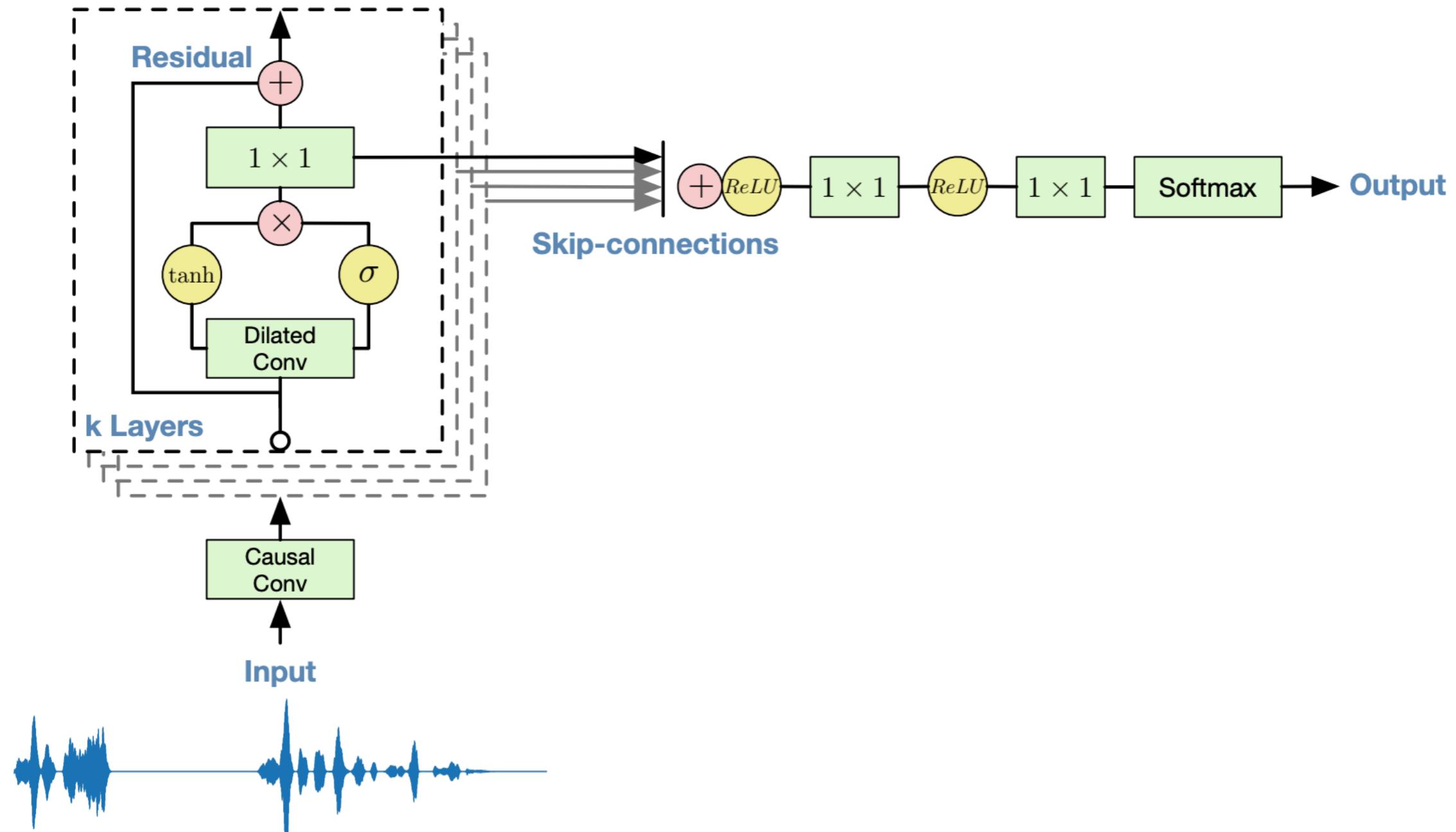
## Conditional Wavenet



“Wavenet: A generative model for raw audio,” arXiv:1609.03499, 2016.

# Speech Synthesis (TTS, Text To Speech)

Wavenet entire architecture



“Wavenet: A generative model for raw audio,” arXiv:1609.03499, 2016.

# Speech Synthesis (TTS, Text To Speech)

```
def naive_wavenet(X, dilations, name='naive_wavenet', channels=256, embedding_dim=256, hidden_dim=256):
    with tf.variable_scope(name):
        ...
        For Training
        ...
        # X : [Batch, Length]

        # [channels, embedding_dim]
        embedding_table = tf.get_variable('embedding_table', [channels, embedding_dim])
        # [Batch, Length, embedding_dim]
        X_embedded = tf.gather(embedding_table, X)

        # [Batch, Length, hidden_dim]
        outputs = tf.layers.conv1d(X_embedded, filters=hidden_dim, kernel_size=1) Embedding

        skips = []
        for dilation in dilations:
            layer_inputs = outputs
            # [Batch, dilation+Length, hidden_dim]
            layer_inputs_padded = tf.pad(layer_inputs, [[0, 0], [dilation, 0], [0, 0]]) Padding for causality

            # [Batch, Length, hidden_dim]
            filter_outputs = tf.layers.conv1d(layer_inputs_padded,
                                              filters=hidden_dim, kernel_size=2, dilation_rate=dilation)
            # [Batch, Length, hidden_dim]
            gate_outputs = tf.layers.conv1d(layer_inputs_padded,
                                              filters=hidden_dim, kernel_size=2, dilation_rate=dilation) Gated Unit

            layer_outputs = tf.nn.tanh(filter_outputs) + tf.nn.sigmoid(gate_outputs)
            # [Batch, Length, hidden_dim]
            skip_outputs = tf.layers.conv1d(layer_outputs, filters=hidden_dim, kernel_size=1)
            skips.append(skip_outputs)

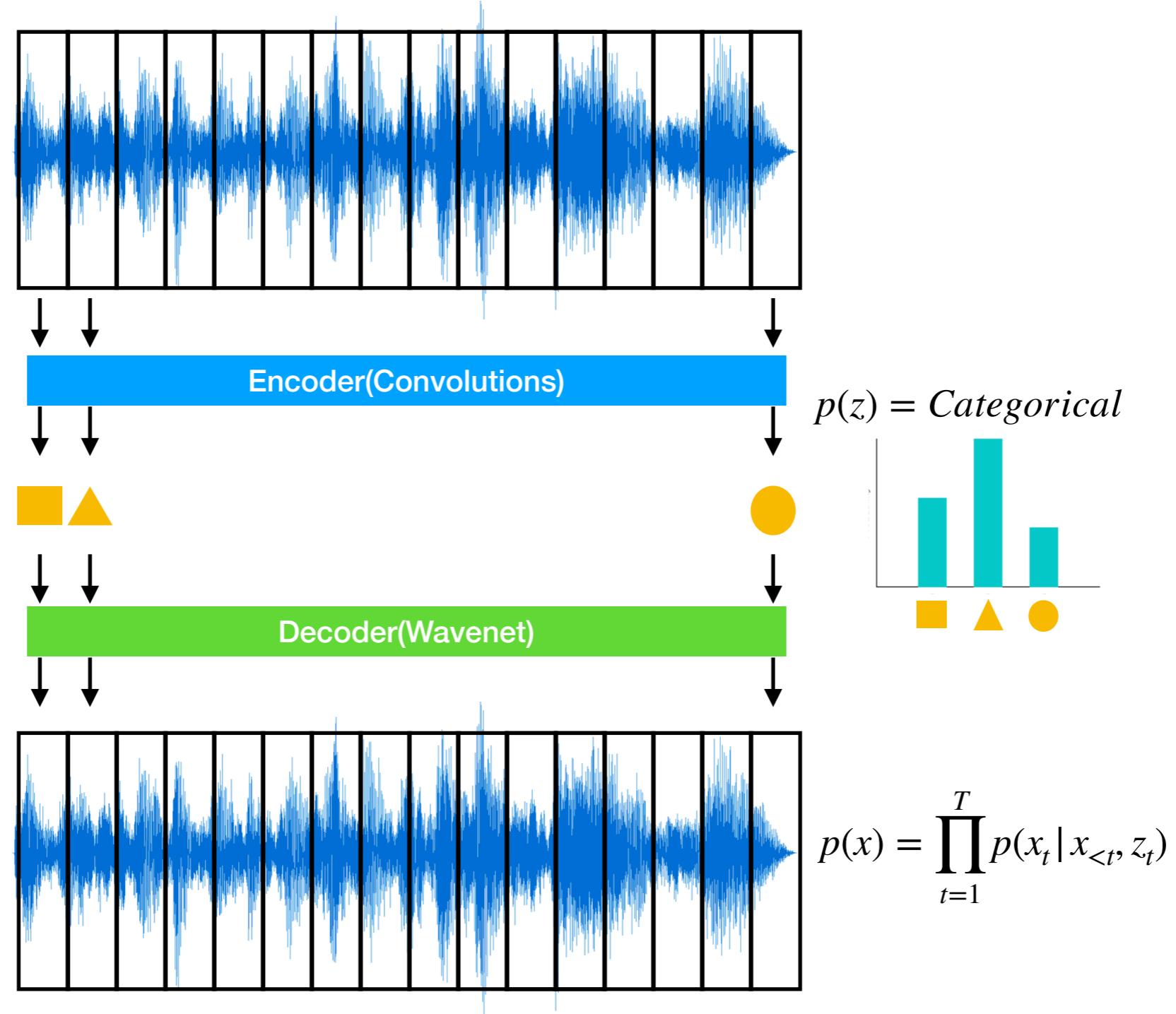
            outputs = layer_inputs + skip_outputs Residual Connection Skip Connection

        # [Batch, Length, hidden_dim]
        outputs = sum(skips)
        outputs = tf.nn.relu(outputs)
        outputs = tf.layers.conv1d(outputs, filters=channels, kernel_size=1)
        outputs = tf.nn.relu(outputs)
        logits = tf.layers.conv1d(outputs, filters=channels, kernel_size=1) Logits for softmax regression

    return logits
```

# Music Generation

## Wavenet (VQ-VAE, AMAE)



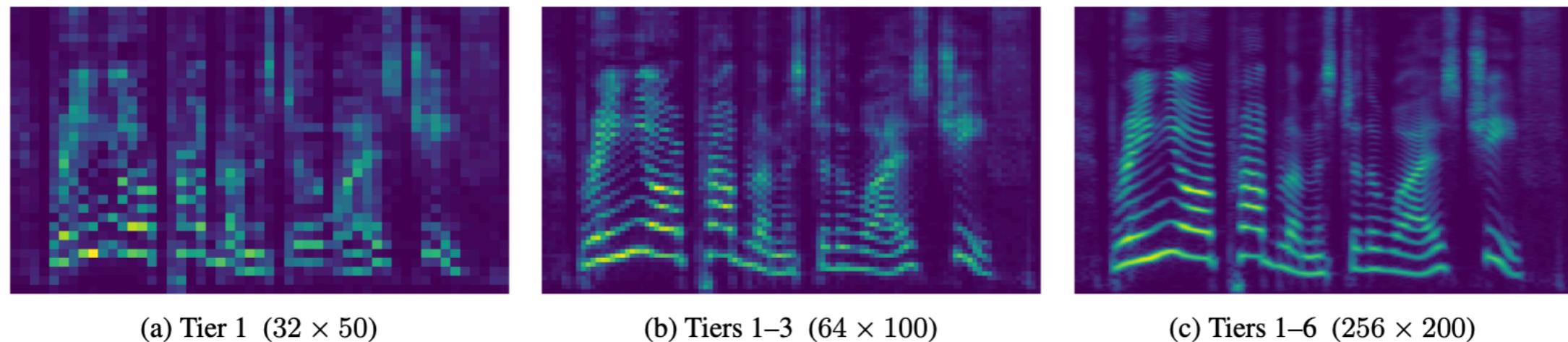
관련 논문

Neural Discrete Representation Learning. <https://arxiv.org/pdf/1711.00937.pdf>

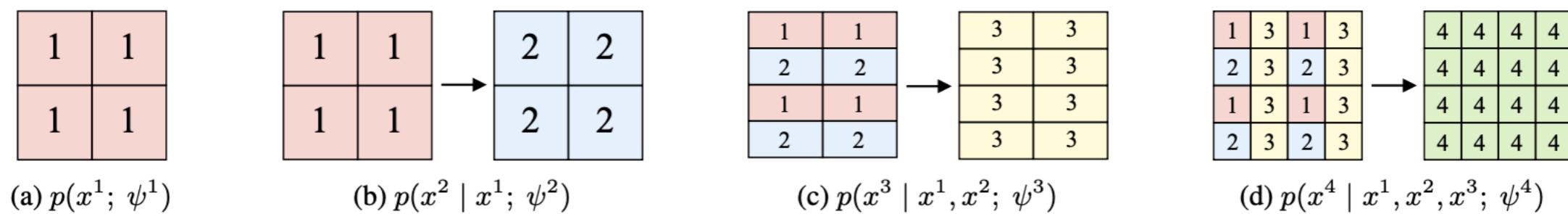
The challenge of realistic music generation: modelling raw audio at scale. <https://arxiv.org/pdf/1806.10474.pdf>

# Music Generation

## Wavenet (Melnet)



**Figure 5.** A sampled spectrogram viewed at different stages of the multiscale generation procedure. The initial tier dictates high-level structure and subsequent tiers add fine-grained details. Each upsampling tier doubles the resolution of the spectrogram, resulting in the initial tier being upsampled by a factor of 32.



**Figure 6.** Schematic showing how tiers of the multiscale model are interleaved and used to condition the distribution for the subsequent tier. a) The initial tier is generated unconditionally. b) The second tier is generated conditionally given the the initial tier. c) The outputs of tiers 1 and 2 are interleaved along the frequency axis and used to condition the generation of tier 3. d) Tier 3 is interleaved along the time axis with all preceding tiers and used to condition the generation of tier 4.

Audio Samples [sjvasquez.github.io/blog/melnet/](https://sjvasquez.github.io/blog/melnet/)

관련 논문 MelNet: A Generative Model for Audio in the Frequency Domain. arxiv.org/abs/1906.01083

# Music Generation

## Music Transformer

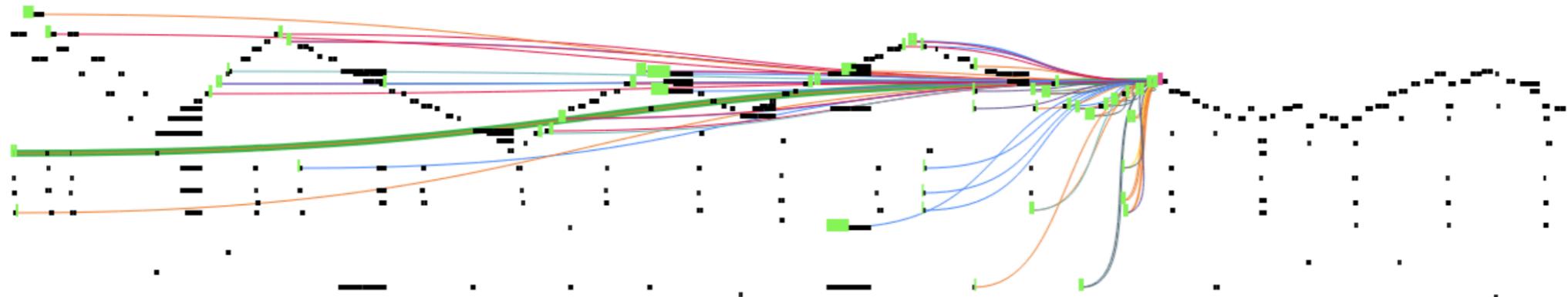


Figure 8: This piece has a recurring triangular contour. The query is at one of the latter peaks and it attends to all of the previous high notes on the peak, all the way to beginning of the piece.

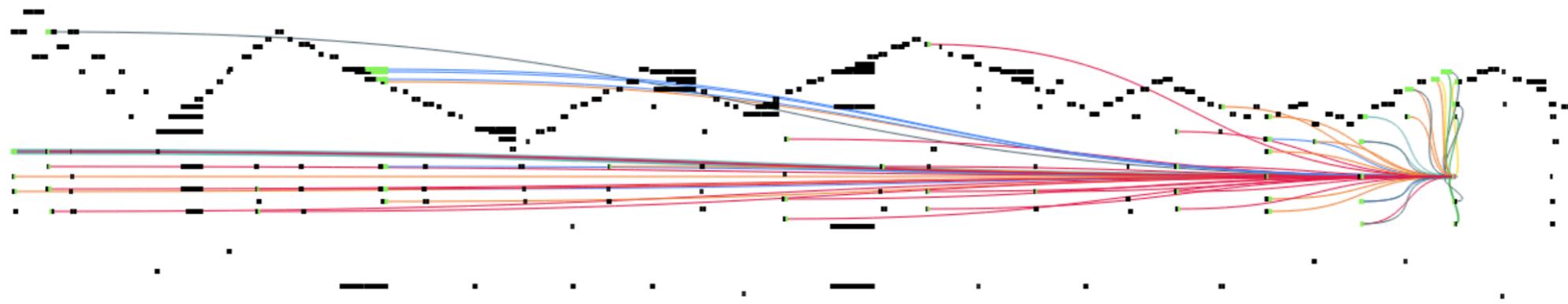


Figure 9: The query a note in the left-hand, and it attends to its immediate past neighbors and mostly to the earlier left hand chords, with most attention lines distributed in the lower half of the pianoroll.

Blog

<https://magenta.tensorflow.org/music-transformer>

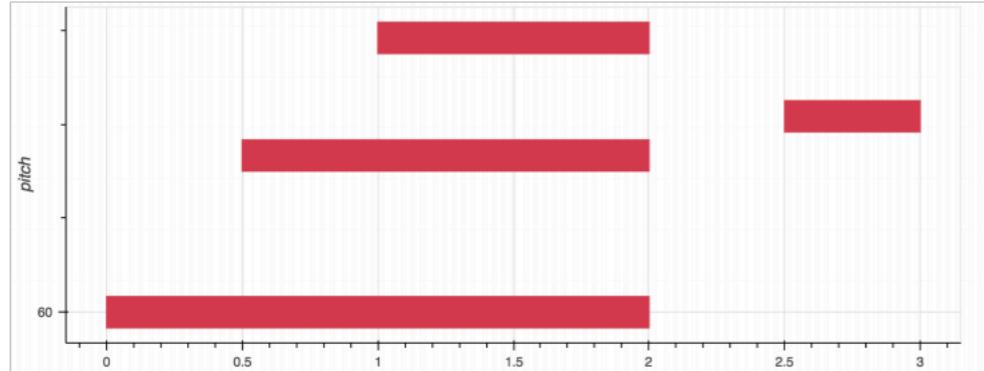
관련 논문

Attention Is All You Need. [arxiv.org/abs/1706.03762](https://arxiv.org/abs/1706.03762)

Music Transformer. [arxiv.org/abs/1809.04281](https://arxiv.org/abs/1809.04281)

# Music Generation

## Music Transformer -MIDI Representation

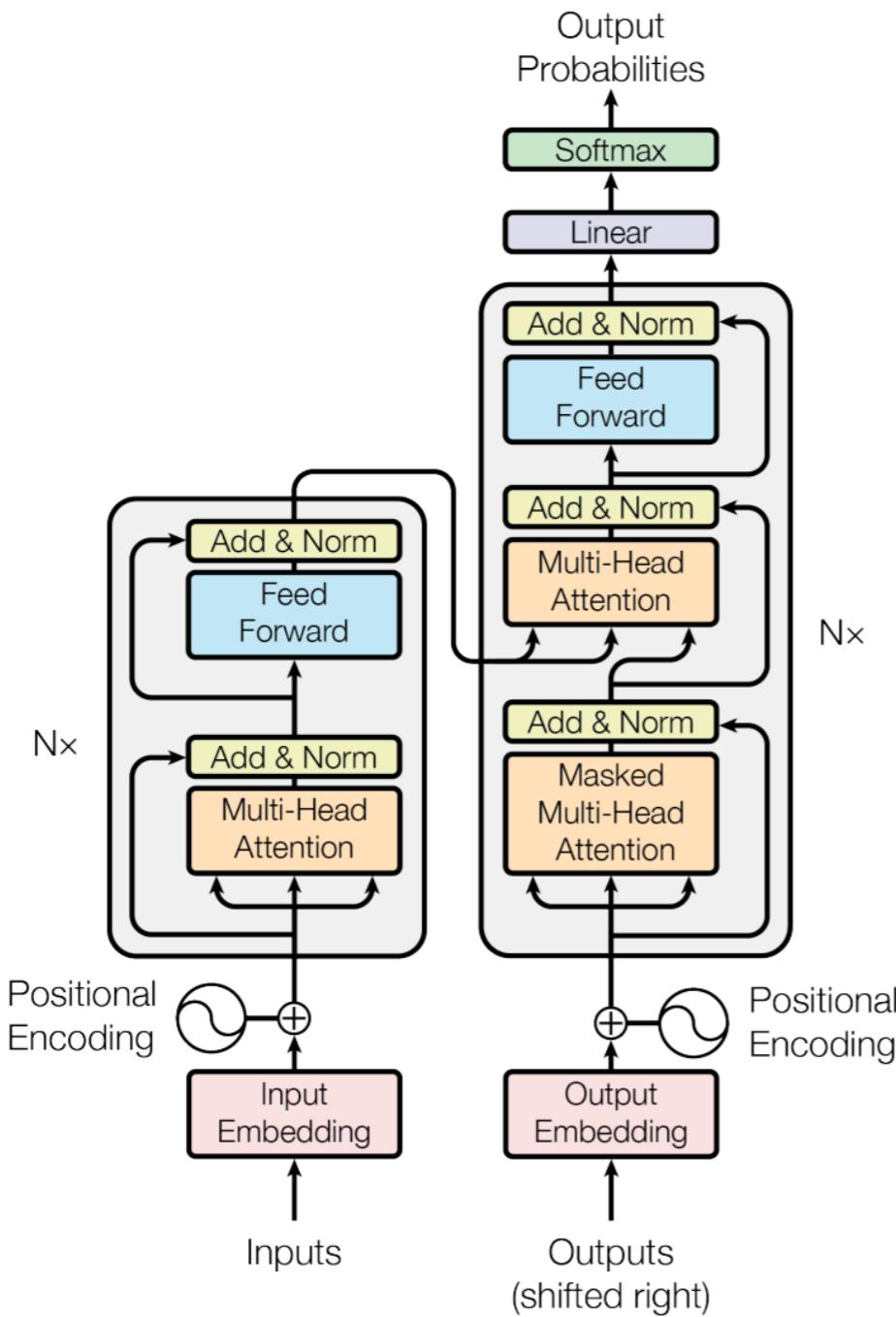


```
SET_VELOCITY<80>, NOTE_ON<60>
TIME_SHIFT<500>, NOTE_ON<64>
TIME_SHIFT<500>, NOTE_ON<67>
TIME_SHIFT<1000>, NOTE_OFF<60>, NOTE_OFF<64>,
NOTE_OFF<67>
TIME_SHIFT<500>, SET_VELOCITY<100>, NOTE_ON<65>
TIME_SHIFT<500>, NOTE_OFF<65>
```

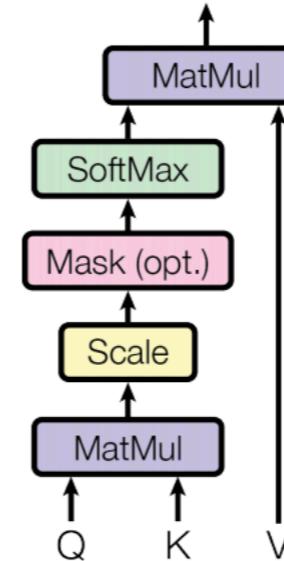
Next, the MIDI note events are converted into a sequence from the following set of vocabulary: 128 NOTE\_ON events for starting a note of with one of the 128 MIDI pitches, 128 NOTE\_OFF events for ending a note with one of the 128 MIDI pitches, 100 TIME\_SHIFT events representing forward time shifts in 10ms increments from 10ms to 1s, and 32 SET\_VELOCITY events representing the velocity for future NOTE\_ON events in the form of the 128 possible MIDI velocities quantized into 32 bins. An example performance encoding is illustrated in Figure 7.

# Music Generation

## Transformer



Scaled Dot-Product Attention



Absolute Position Attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Relative Position Attention (Music Transformer)

$$\text{RelativeAttention} = \text{Softmax}\left(\frac{QK^T + S^{rel}}{\sqrt{D_h}}\right)V.$$

Figure 1: The Transformer - model architecture.

# Music Generation

## GPT-2

```
def model(hparams, X, past=None, scope='model', reuse=False):
    with tf.variable_scope(scope, reuse=reuse):
        results = {}
        batch, sequence = shape_list(X)

        wpe = tf.get_variable('wpe', [hparams.n_ctx, hparams.n_embd],
                              initializer=tf.random_normal_initializer(stddev=0.01))
        wte = tf.get_variable('wte', [hparams.n_vocab, hparams.n_embd],
                              initializer=tf.random_normal_initializer(stddev=0.02))
        past_length = 0 if past is None else tf.shape(past)[-2]
        h = tf.gather(wte, X) + tf.gather(wpe, positions_for(X, past_length))

# Transformer
presents = []
pasts = tf.unstack(past, axis=1) if past is not None else [None] * hparams.n_layer
assert len(pasts) == hparams.n_layer
for layer, past in enumerate(pasts):
    h, present = block(h, 'h%d' % layer, past=past, hparams=hparams)
    presents.append(present)
results['present'] = tf.stack(presents, axis=1)
h = norm(h, 'ln_f')

# Language model loss. Do tokens <n predict token n?
h_flat = tf.reshape(h, [batch*sequence, hparams.n_embd])
logits = tf.matmul(h_flat, wte, transpose_b=True)
logits = tf.reshape(logits, [batch, sequence, hparams.n_vocab])
results['logits'] = logits

return results
```

Embedding

Block repeats

Logit outputs

<https://github.com/openai/gpt-2/blob/master/src/model.py>

# Music Generation

```
def block(x, scope, *, past, hparams):
    with tf.variable_scope(scope):
        nx = x.shape[-1].value
        a, present = attn(norm(x, 'ln_1'), 'attn', nx, past=past, hparams=hparams)
        x = x + a
        m = mlp(norm(x, 'ln_2'), 'mlp', nx*4, hparams=hparams)
        x = x + m
    return x, present

def attn(x, scope, n_state, *, past, hparams):
    def multihead_attn(q, k, v):
        # q, k, v have shape [batch, heads, sequence, features]
        w = tf.matmul(q, k, transpose_b=True)
        w = w * tf.rsqrt(tf.cast(v.shape[-1].value, w.dtype))

        w = mask_attn_weights(w)
        w = softmax(w)
        a = tf.matmul(w, v)

    with tf.variable_scope(scope):
        c = conv1d(x, 'c_attn', n_state*3)
        q, k, v = map(split_heads, tf.split(c, 3, axis=2))
        present = tf.stack([k, v], axis=1)
        if past is not None:
            pk, pv = tf.unstack(past, axis=1)
            k = tf.concat([pk, k], axis=-2)
            v = tf.concat([pv, v], axis=-2)
        a = multihead_attn(q, k, v)
        a = merge_heads(a)
        a = conv1d(a, 'c_proj', n_state)
    return a, present
```

Attention & Residual Connection

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Query, Key, Value

Concat past data

Merge multi-heads & output projection

# Music Generation

Musenet

Compose in the style of Chopin- starting with  
Mozart's Rondo alla Turca-

[SHOW ADVANCED SETTINGS](#)

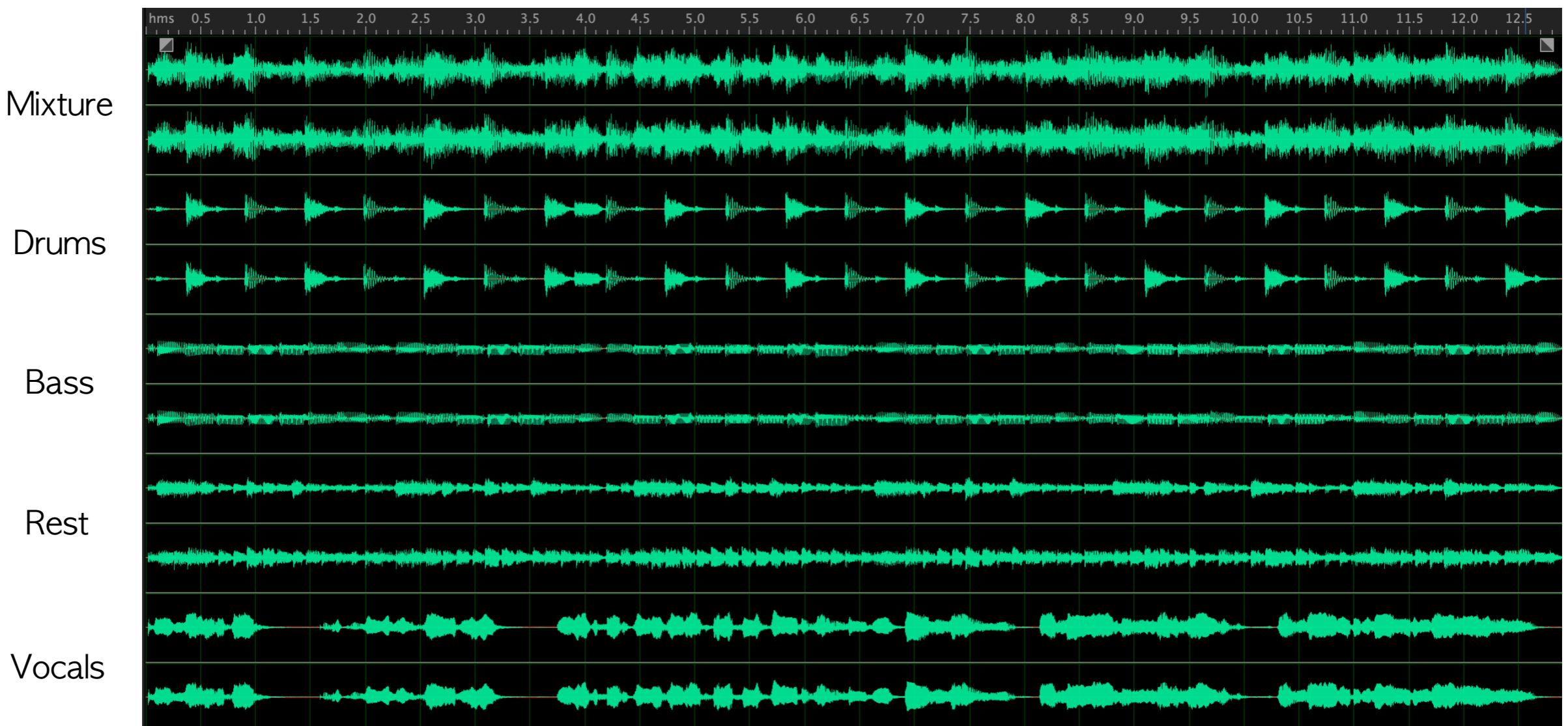
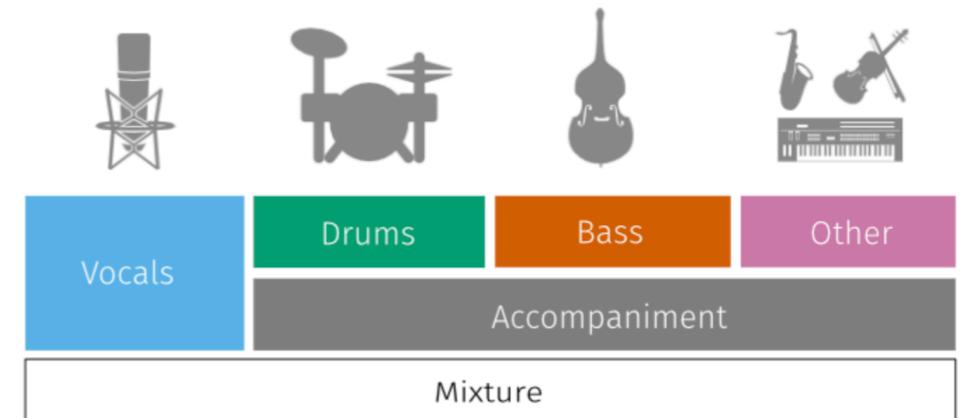


Blog

<https://openai.com/blog/musenet>

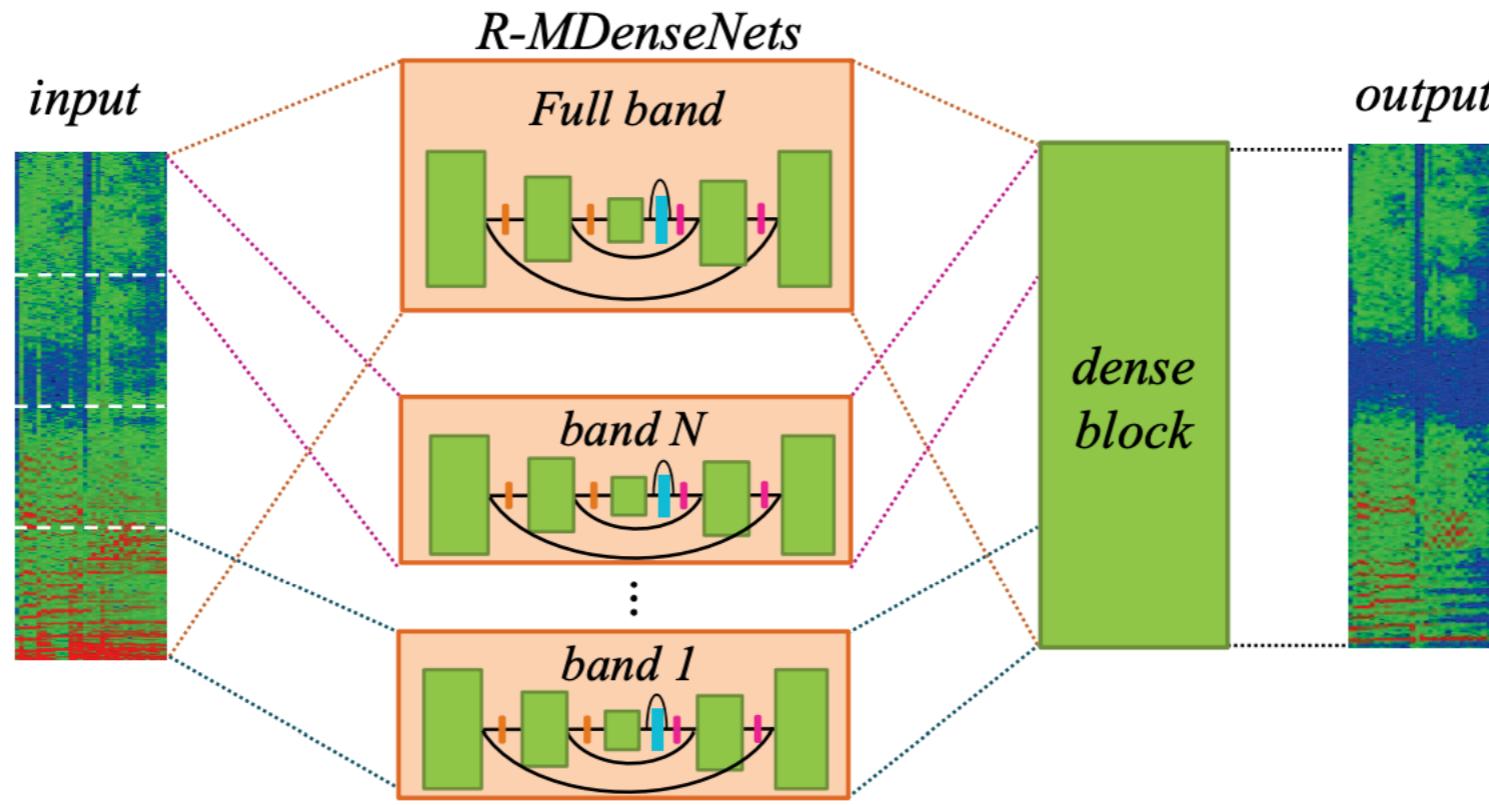
# Source Separation

## Dataset



# Source Separation

## MMDenseLSTM



**Fig. 3.** MMDenseLSTM architecture. Outputs of MDenseLSTM dedicated to different frequency band including the full band are concatenated and the final dense block integrates features from these bands to create the final output.

SiSEC 2018 website

[sisec18.unmix.app](http://sisec18.unmix.app)

관련 논문

MMDenseLSTM: An efficient combination of convolutional and recurrent neural networks for audio source separation  
<https://arxiv.org/abs/1805.02410>