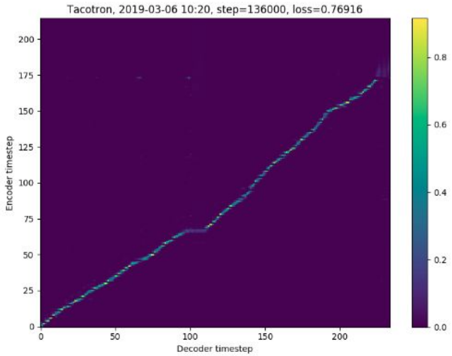
LOCATION-RELATIVE ATTENTION MECHANISMS FOR ROBUST LONG-FORM SPEECH SYNTHESIS

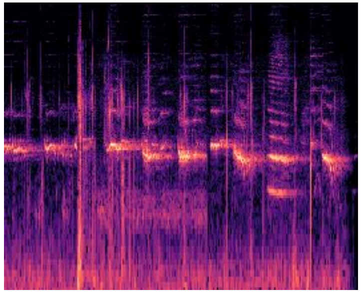
Battenberg et al. (Google Research)

Attention-based Seq2Seq

ㄱ ㅏ ㄴ ㅏ ㄷ ㅏ

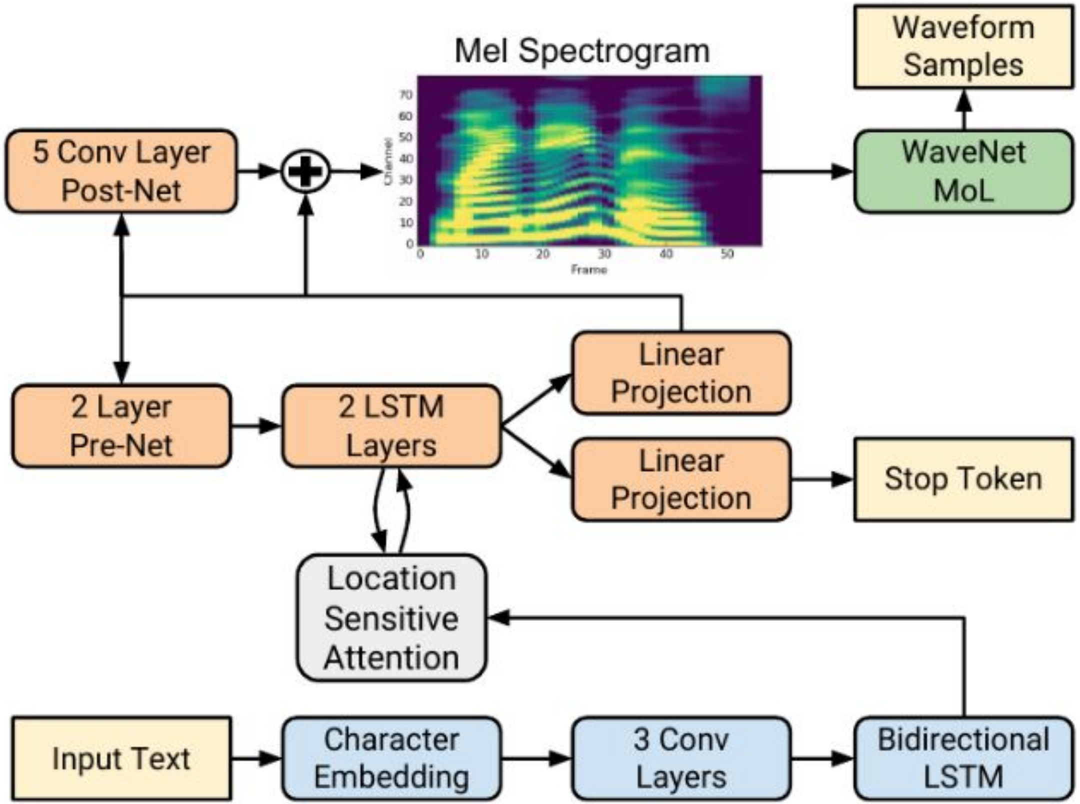
(1, seq_len_char, 512)





(1, seq_len_mel, 80)

Attention-based Seq2Seq



기존 Attention

1. Content-based attention
2. Location-sensitive attention
3. GMM attention

기존 Attention 문제점

긴 텍스트

못 본 단어

기존 Attention 일반화

Additive Energy-Based Mechanisms

1. Content-based attention

2. Location-sensitive attention

*. Dynamic convolution attention

GMM-based Mechanisms

3. GMM attention

*. GMM V2b

GMM Attention

$$e_{i,j} = w^\top \tanh(W s_{i-1} + V h_j + b).$$

$$(\hat{\boldsymbol{w}}_i, \hat{\boldsymbol{\Delta}}_i, \hat{\boldsymbol{\sigma}}_i) = V \tanh(W \boldsymbol{s}_i + b)$$

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}_{i-1} + \boldsymbol{\Delta}_i$$

$$\alpha_{i,j} = \sum_{k=1}^{K} \frac{w_{i,k}}{Z_{i,k}} \exp\left(-\frac{(j - \mu_{i,k})^2}{2(\sigma_{i,k})^2}\right)$$

GMM Attention

$$\alpha_{i,j} = \sum_{k=1}^{K} \frac{w_{i,k}}{Z_{i,k}} \exp\left(-\frac{(j-\mu_{i,k})^2}{2(\sigma_{i,k})^2}\right) \quad \Rightarrow \quad \alpha_j = \frac{w}{Z} exp(-\frac{(j-\mu)^2}{2(\sigma)^2})$$

$J = 3$
$I = 5$

$\Delta = 0.5$
$\mu_{0,1,2,3,4} = 0.0, 0.5, 1.0, 1.5, 2.0$
$\sigma = 1$

ㄱ(α_0): $e^{-\frac{(0)^2}{2}} = 1.000$ $e^{-\frac{(-.5)^2}{2}} = 0.882$ $e^{-\frac{(-1)^2}{2}} = 0.607$ $e^{-\frac{(-1.5)^2}{2}} = 0.607$ $e^{-\frac{(-2)^2}{2}} = 0.135$

ㅏ(α_1): $e^{-\frac{(1)^2}{2}} = 0.607$ $e^{-\frac{(.5)^2}{2}} = 0.882$ $e^{-\frac{(0)^2}{2}} = 1.000$ $e^{-\frac{(-.5)^2}{2}} = 0.882$ $e^{-\frac{(-1)^2}{2}} = 0.607$

ㅇ(α_2): $e^{-\frac{(2)^2}{2}} = 0.135$ $e^{-\frac{(1.5)^2}{2}} = 0.325$ $e^{-\frac{(1)^2}{2}} = 0.607$ $e^{-\frac{(.5)^2}{2}} = 0.882$ $e^{-\frac{(0)^2}{2}} = 1.000$

GMM Attention

$$\alpha_{i,j} = \sum_{k=1}^{K} \frac{w_{i,k}}{Z_{i,k}} \exp\left(-\frac{(j-\mu_{i,k})^2}{2(\sigma_{i,k})^2}\right)$$

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}_{i-1} + \boldsymbol{\Delta}_i$$

$$(\hat{\boldsymbol{w}}_i, \hat{\boldsymbol{\Delta}}_i, \hat{\boldsymbol{\sigma}}_i) = V \tanh(W\boldsymbol{s}_i + b)$$

	$\boldsymbol{Z}_i$	$\boldsymbol{w}_i$	$\boldsymbol{\Delta}_i$	$\boldsymbol{\sigma}_i$
V0 [1]	$\mathbf{1}$	$\exp(\hat{\boldsymbol{w}}_i)$	$\exp(\hat{\boldsymbol{\Delta}}_i)$	$\sqrt{\exp(-\hat{\boldsymbol{\sigma}}_i)/2}$
V1	$\sqrt{2\pi\boldsymbol{\sigma}_i^2}$	$\mathrm{S}_{\max}(\hat{\boldsymbol{w}}_i)$	$\exp(\hat{\boldsymbol{\Delta}}_i)$	$\sqrt{\exp(\hat{\boldsymbol{\sigma}}_i)}$
V2	$\sqrt{2\pi\boldsymbol{\sigma}_i^2}$	$\mathrm{S}_{\max}(\hat{\boldsymbol{w}}_i)$	$\mathrm{S}_{+}(\hat{\boldsymbol{\Delta}}_i)$	$\mathrm{S}_{+}(\hat{\boldsymbol{\sigma}}_i)$

V0: Original GMM

V1: Normalization of the mixture weights

V2: Softplus for sigma and offset

GMM Attention

```python
def get_alignment_energies(self, query, processed_memory, attention_weights_cat, attention_mu):
    (w_hat, delta_hat, sigma_hat) = torch.split(self.v(F.tanh(self.query_layer(query.unsqueeze(1)))), attention_mu.shape[-1], dim=-1)

    w = torch.nn.Softmax(dim=2)(w_hat) # ([bs, 1, gaussian_n_mixtures])
    delta = torch.nn.Softplus()(delta_hat) # ([bs, 1, gaussian_n_mixtures])
    sigma = torch.nn.Softplus()(sigma_hat) # ([bs, 1, gaussian_n_mixtures])
    z = torch.sqrt(2 * pi * (sigma ** 2)) # ([bs, 1, gaussian_n_mixtures])

    j = torch.arange(0, processed_memory.shape[1], step=1.).unsqueeze(0).unsqueeze(2).repeat(w.shape[0], 1, 1).to('cuda')
    energies = torch.sum((w / z) * torch.exp((-1. * (j - attention_mu) ** 2) / (2. * sigma ** 2)), dim=2)

    mu = attention_mu + delta

    return energies, mu
```

GMM Attention V2b

Initial bias to $\hat{\boldsymbol{\Delta}}_i$ and $\hat{\boldsymbol{\sigma}}_i$

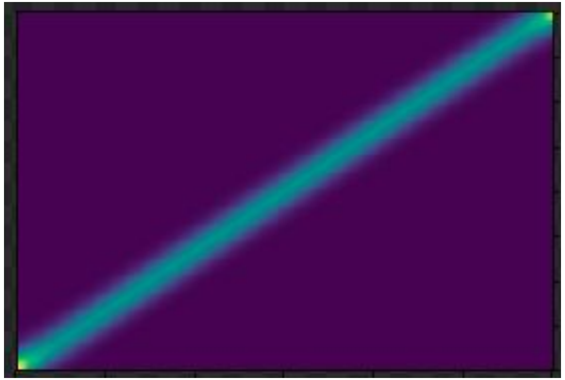
s.t. $\boldsymbol{\Delta}_i = 1$

$\boldsymbol{\sigma}_i = 10$

b of $\hat{\Delta}_0 = ln(e^1 - 1)$ $\qquad$ $\Delta_i = ln(1 + e^{ln(e^1 - 1)})$

b of $\hat{\sigma}_0 = ln(e^{10} - 1)$ $\qquad$ $\sigma_i = ln(1 + e^{ln(e^{10} - 1)})$

GMM Attention V2b



Energy-based Attention

$$e_{i,j} = w^\top \tanh(W s_{i-1} + V h_j + b).$$

$$e_{i,j} = w^\top \tanh(W s_{i-1} + V h_j + U f_{i,j} + b)$$

$$\begin{aligned}
e_{i,j} &= \boldsymbol{v}^\intercal \tanh(W\boldsymbol{s}_i + V\boldsymbol{h}_j + U\boldsymbol{f}_{i,j} + T\boldsymbol{g}_{i,j} + \boldsymbol{b}) + p_{i,j} \\
\boldsymbol{\alpha}_i &= \mathrm{S}_{\max}(\boldsymbol{e}_i) \\
\boldsymbol{f}_i &= \mathcal{F} * \boldsymbol{\alpha}_{i-1} \\
\boldsymbol{g}_i &= \mathcal{G}(\boldsymbol{s}_i) * \boldsymbol{\alpha}_{i-1}, \quad \mathcal{G}(\boldsymbol{s}_i) = V_{\mathcal{G}} \tanh(W_{\mathcal{G}}\boldsymbol{s}_i + \boldsymbol{b}_{\mathcal{G}}) \\
\boldsymbol{p}_i &= \log(\mathcal{P} * \boldsymbol{\alpha}_{i-1})
\end{aligned}$$

Table 2. The terms from (8) that are present in each of the three energy-based attention mechanisms we test.

	$W\boldsymbol{s}_i$	$V\boldsymbol{h}_j$	$U\boldsymbol{f}_{i,j}$	$T\boldsymbol{g}_{i,j}$	$p_{i,j}$
Content-Based [2]	✓	✓	-	-	-
Location-Sensitive [8]	✓	✓	✓	-	-
Dynamic Convolution	-	-	✓	✓	✓

Energy-based Attention

$$e_{i,j} = \boldsymbol{v}^{\intercal} \tanh(W\boldsymbol{s}_i + V\boldsymbol{h}_j + U\boldsymbol{f}_{i,j} + T\boldsymbol{g}_{i,j} + \boldsymbol{b}) + p_{i,j}$$

$$\boldsymbol{\alpha}_i = \mathrm{S}_{\max}(\boldsymbol{e}_i)$$

$$\boldsymbol{f}_i = \mathcal{F} * \boxed{\boldsymbol{\alpha}_{i-1}}$$

$$\boldsymbol{g}_i = \mathcal{G}(\boldsymbol{s}_i) * \boldsymbol{\alpha}_{i-1}, \quad \mathcal{G}(\boldsymbol{s}_i) = V_{\mathcal{G}} \tanh(W_{\mathcal{G}}\boldsymbol{s}_i + \boldsymbol{b}_{\mathcal{G}})$$

$$\boldsymbol{p}_i = \log(\mathcal{P} * \boldsymbol{\alpha}_{i-1})$$

query Ws_i: [B, 1, attention_dim]

key Vh_j: [B, char_seq_len, attention_dim]

filters Uf_j, Tg_j : [B, char_seq_len, attention_dim]

prior filter p_j: [B, char_seq_len, 1]

```
class LocationLayer(nn.Module):
    def __init__(self, attention_n_filters, attention_kernel_size,
                 attention_dim):
        super(LocationLayer, self).__init__()
        padding = int((attention_kernel_size - 1) / 2)
        self.location_conv = ConvNorm(2, attention_n_filters,
                                      kernel_size=attention_kernel_size,
                                      padding=padding, bias=False, stride=1,
                                      dilation=1)
        self.location_dense = LinearNorm(attention_n_filters, attention_dim,
                                         bias=False, w_init_gain='tanh')

    def forward(self, attention_weights_cat):
        processed_attention = self.location_conv(attention_weights_cat)
        processed_attention = processed_attention.transpose(1, 2)
        processed_attention = self.location_dense(processed_attention)
        return processed_attention
```

```
def get_alignment_energies(self, query, processed_memory,
                           attention_weights_cat):
    processed_query = self.query_layer(query.unsqueeze(1))
    processed_attention_weights = self.location_layer(attention_weights_cat)
    energies = self.v(torch.tanh(
        processed_query + processed_attention_weights + processed_memory))

    energies = energies.squeeze(2)
    return energies
```

Energy-based Attention: Dynamic Convolution

Motivation

Location-relative

Fully normalized attention weights

*GMM attention weights unnormalized, sampled from continuous pdf

Distribution with finite support

*GMM may violate monotonicity due to its infinite support

Energy-based Attention: Dynamic Convolution

$$e_{i,j} = \boldsymbol{v}^{\intercal} \tanh(\text{[redacted]} U\boldsymbol{f}_{i,j} + T\boldsymbol{g}_{i,j} + \boldsymbol{b}) + p_{i,j}$$
$$\boldsymbol{\alpha}_i = \mathrm{S}_{\max}(\boldsymbol{e}_i)$$
$$\boldsymbol{f}_i = \mathcal{F} * \boldsymbol{\alpha}_{i-1}$$
$$\boldsymbol{g}_i = \mathcal{G}(\boldsymbol{s}_i) * \boldsymbol{\alpha}_{i-1}, \quad \mathcal{G}(\boldsymbol{s}_i) = V_{\mathcal{G}} \tanh(W_{\mathcal{G}}\boldsymbol{s}_i + \boldsymbol{b}_{\mathcal{G}})$$
$$\boldsymbol{p}_i = \log(\mathcal{P} * \boldsymbol{\alpha}_{i-1})$$

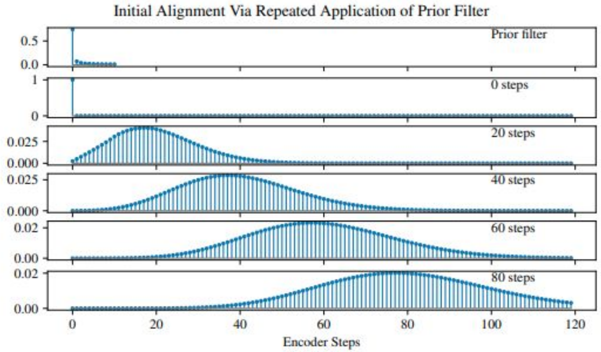
Ws_i와 Vh_j 제거: 뒤쪽의 비슷한 contents에 대한 노출을 차단

$Uf_{i,j}$는 alignment가 fixed amount만큼만 앞으로 움직이도록 강제

$Tg_{i,j}$는 alignment가 동적으로 움직이도록 조절

$P_{i,j}$는 causal prior filter로 alignment의 forward progression만 허용

$$p(k) = \binom{n}{k} \frac{\mathrm{B}(k+\alpha, n-k+\beta)}{\mathrm{B}(\alpha, \beta)}, \quad k \in \{0, \ldots, n\}$$



결과

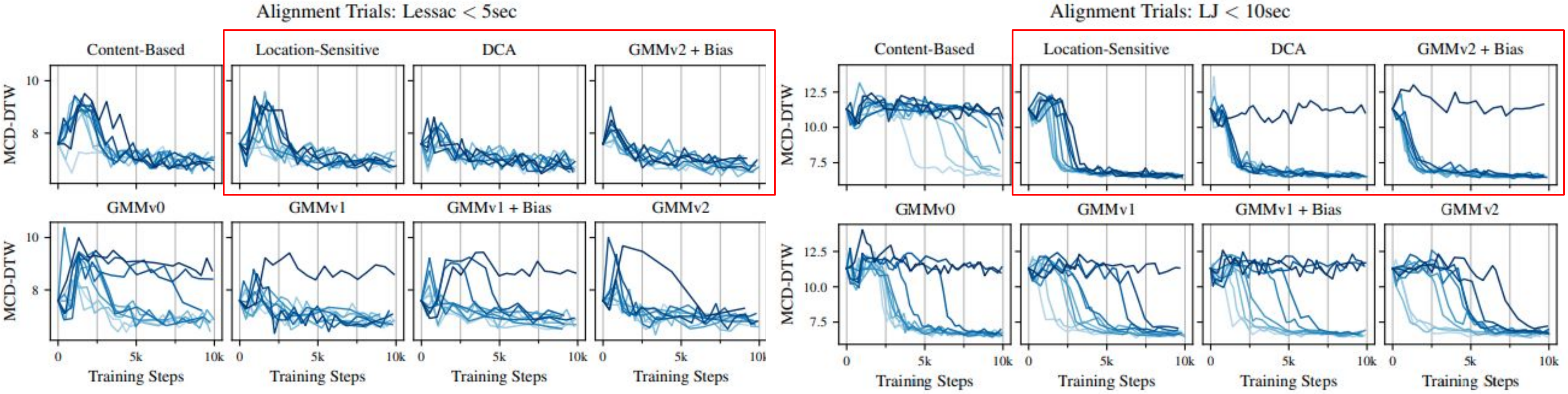


Table 3. MOS naturalness results along with 95% confidence intervals for the Lessac and LJ datasets.

	Lessac	LJ
Content-Based	4.07 ± 0.08	4.19 ± 0.06
Location-Sensitive	4.31 ± 0.06	4.34 ± 0.06
GMMv2b	4.32 ± 0.06	4.29 ± 0.06
DCA	4.31 ± 0.06	4.33 ± 0.06
Ground Truth	4.64 ± 0.04	4.55 ± 0.04

결 과

https://google.github.io/tacotron/publications/location_relative_attention/

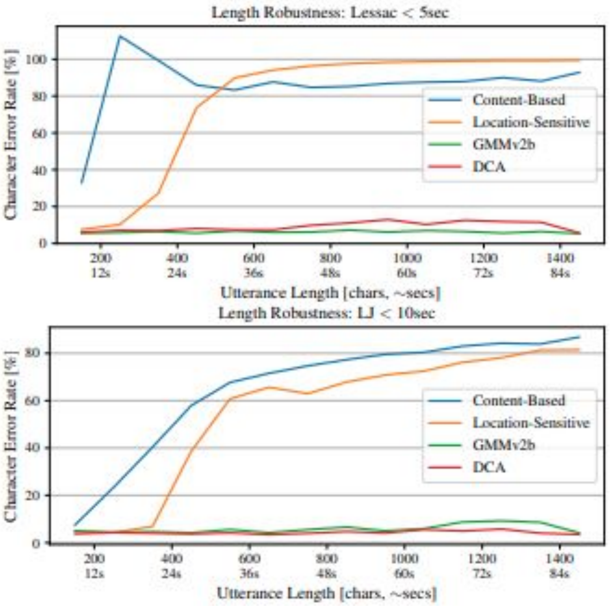


Fig. 3. Utterance length robustness for models trained on the Lessac (top) and LJ (bottom) datasets.