Boulder

Speech & Music

YouTube Playlist

Maziar Raissi

Assistant Professor

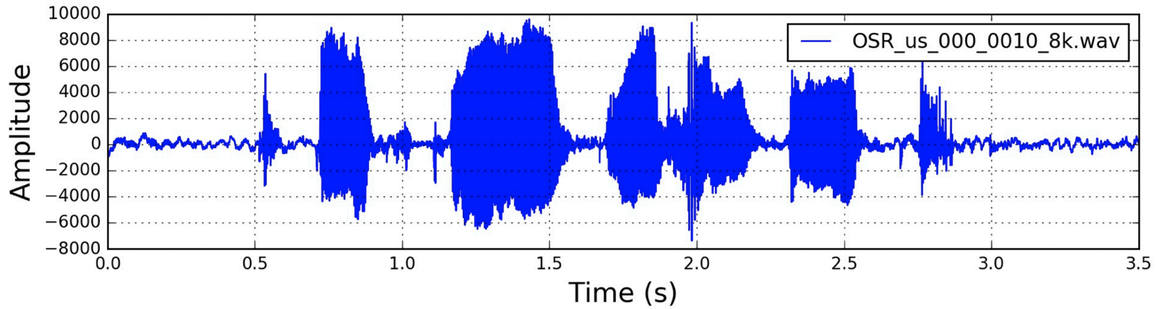
Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

Mel-Spectrogram and Mel-Frequency Cepstral Coefficients (MFCCs)

YouTube Video



https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

$X_i \in \mathbb{R}^{200} \rightarrow i$-th frame of signal x (25ms frames)

$80 \rightarrow$ frame step (10ms) $\implies X \in \mathbb{R}^{200 \times 350}$ $(350 = (28,200 - 200)/80)$

$\widetilde{X}_i \in \mathbb{C}^K \rightarrow$ discrete Fourier transform of $X_i \implies \widetilde{X} \in \mathbb{C}^{K \times 350}$

$$\widetilde{X}_i(k) = \sum_{n=1}^{N} X_i(n) g(n) e^{-j2\pi kn/N}, k = 1, \ldots, K \qquad N = 200$$

$K = 257 \rightarrow$ number of discrete Fourier transform coefficients

$P_i(k) = \frac{1}{N}|\widetilde{X}_i(k)|^2 \rightarrow$ Periodogram estimate of the power spectrum

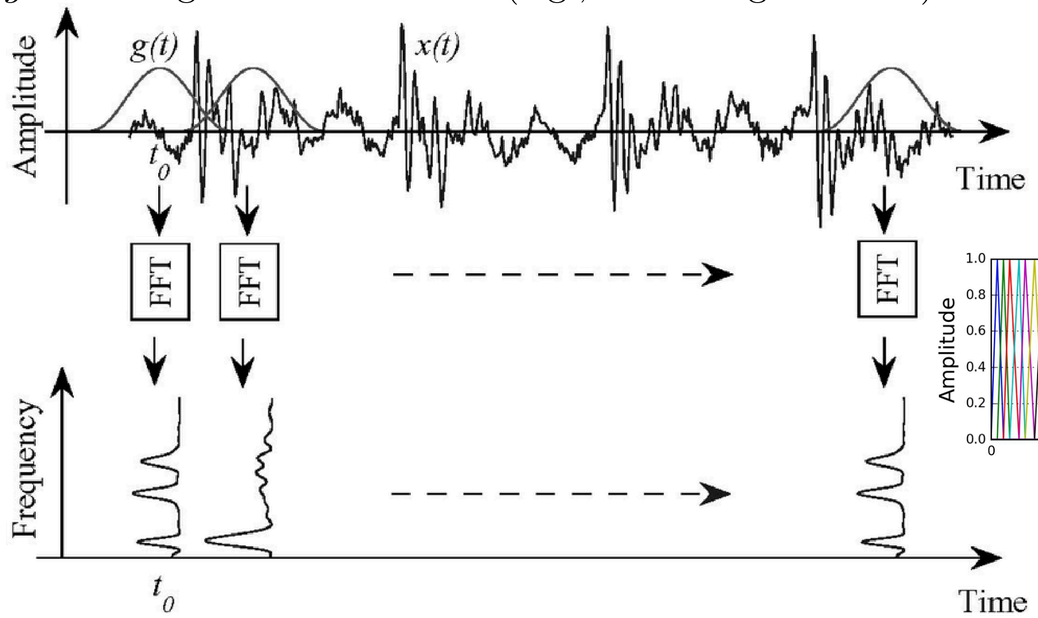
$\implies P \in \mathbb{R}^{257 \times 350}$

http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

Short-Time Fourier-Transform (STFT)

$x \rightarrow$ signal in the time domain (e.g., $x \in \mathbb{R}^{28,200=3.525 \text{ seconds} \times 8,000 \text{ Hz}}$)
sampling frequency (number of samples per seconds)

$g \rightarrow$ sliding window function (e.g., Hamming function)



Gao, Robert X., and Ruqiang Yan. "Non-stationary signal processing for bearing health monitoring." *International journal of manufacturing research* 1.1 (2006): 18-40.

Mel-spaced Filterbank

https://www.youtube.com/watch?v=WJI-17MNpdE

300 Hz $\rightarrow$ lower frequency

4,000 Hz $\rightarrow$ upper frequency

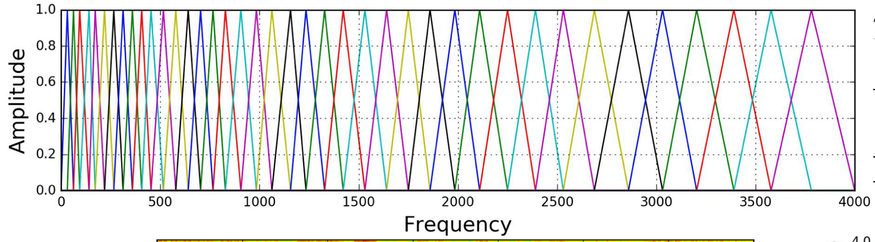
$$f(k) = 300 + \frac{k-1}{K-1}(4,000 - 300)$$

$M(f) = 1,125 \ln(1 + f/700) \rightarrow$ convert frequency to Mel scale

$M^{-1}(m) = 700(\exp(m/1,125) - 1) \rightarrow$ convert Mel scale to Hz

$M^{-1}(\text{linspace}(M(300), M(4,000), 26 + 2))$

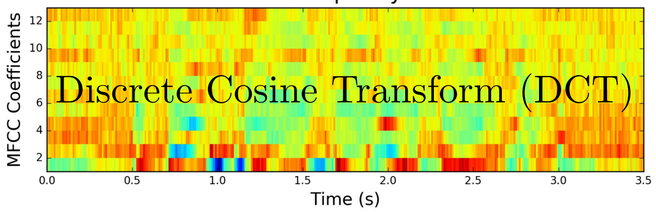
$26 \rightarrow$ number of triangular filters



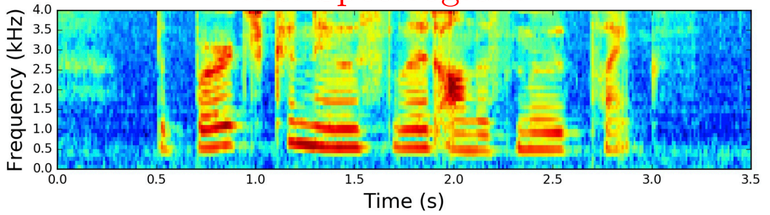
$T \in \mathbb{R}^{26 \times 257} \implies E = TP \in \mathbb{R}^{26 \times 350}$

$E_i(l) \rightarrow$ amount of energy in filter bank l at frame i

$\log(E) \in \mathbb{R}^{26 \times 350} \rightarrow$ log filter bank energy



Mel Spectrogram



Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks

YouTube Video

speech recognition: transcribe an acoustic signal into words and subwords units
noisy, real-valued input streams are annotated with strings of discrete labels (e.g., letters or words) – handwriting recongnition – speech recognition – gesture recognition
temporal classification: the task of labelling unsegmented data sequences
connectionist temporal classification: use of RNNs
framewise classification: independent labelling of each time-step or frame of the input sequence

Temporal Classification

$S \rightarrow$ set of training examples drawn from a fixed distribution $\mathcal{D}_{\mathcal{X}\times\mathcal{Z}}$
$\mathcal{X} = (\mathbb{R}^m)^* \rightarrow$ input space (set of all sequences of m dimensional real-valued vectors)
$\mathcal{Z} = L^* \rightarrow$ target space (set of all sequences over the finite alphabet L of labels
$l \in L^* \rightarrow$ label sequence or labelling
$(x, z) \in S \rightarrow$ an example pair of sequences
$x = (x_1, \ldots, x_T) \rightarrow$ input sequence
$z = (z_1, \ldots, z_U) \rightarrow$ target sequence
$U \leq T \rightarrow$ the target sequence is at most as long as the input sequance

Connectionist Temporal Classification

$\mathcal{N}_w : (\mathbb{R}^m)^T \rightarrow (\mathbb{R}^n)^T$
└ RNN with weight vector w
$y = \mathcal{N}_w(x) \rightarrow$ sequence of the network outputs
$y_k^t \rightarrow$ activation of output unit k at time t
└ probability of observing label k at time t

$$p(\pi|x) = \prod_{t=1}^{T} y_{\pi_t}^t, \forall \pi \in (\underbrace{L \cup \{\text{blank}\}}_{=:\Lambda})^T$$

└ path

defines a distribution over the set Λ^T of length T sequences over the alphabet Λ

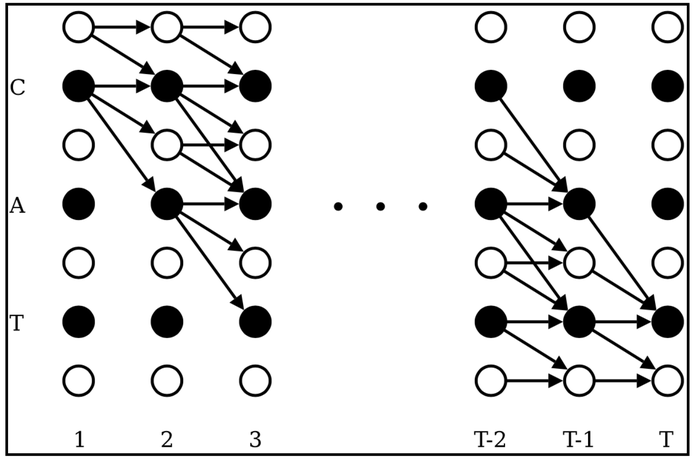
$\mathcal{B} : \Lambda^T \rightarrow L^{\leq T}$ $L^{\leq T} \rightarrow$ set of all possible labelings (set of sequences of length $\leq T$ over the original label alphabet L)
└ many-to-one map $\mathcal{B}(a - ab-) = aab = \mathcal{B}(-aa - -abb) \rightarrow$ remove of all blanks and repeated labels

$$p(l|x) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi|x)$$

$l \in L^{\leq T} \rightarrow$ labeling

Best path decoding

$h(x) \approx \mathcal{B}(\pi^*)$
$\pi^* = \arg\max_{\pi \in \Lambda^t} p(\pi|x)$



Problem: No a priori way of aligning x & z!

Label Error Rate

$h : \mathcal{X} \rightarrow \mathcal{Z}$
└ temporal classifier (to be trained using S)
$S' \subset \mathcal{D}_{\mathcal{X}\times\mathcal{Z}} \rightarrow$ test set disjoint from S

$$\text{LER}(h, S') = \frac{1}{|S'|} \sum_{(x,z)\in S'} \frac{\text{ED}(h(x), z)}{|z|}$$

$\text{ED}(h(x), z) \rightarrow$ edit distance
minimum number of insertions, substitutions, and deletions required to change $h(x)$ to z

Training the Network (Dynamic Programing)

$$\alpha_t(s) := \sum_{\substack{\pi \in \Lambda^T : \\ \mathcal{B}(\pi_{1:t}) = l_{1:s}}} \prod_{\tau=1}^{t} y_{\pi_\tau}^\tau \rightarrow \text{total probability of } l_{1:s} \text{ at time } t$$

$$\bar{\alpha}_t(s) \overset{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y_{l'_s}^t & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ \big(\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)\big) y_{l'_s}^t & \text{otherwise} \end{cases}$$

$\alpha_1(1) = y_b^1$
$\alpha_1(2) = y_{l_1}^1$
$\alpha_1(s) = 0, \ \forall s > 2$

$p(l|x) = \alpha_T(|l'|) + \alpha_T(|l'| - 1)$

$l' \rightarrow$ blanks added to the beginning and the end and inserted between every pair of labels

Illustration of the forward algorithm applied to the labelling 'CAT'. For the backward algorithm please refer to the paper.

Graves, Alex, et al. "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks." *Proceedings of the 23rd international conference on Machine learning*. 2006.

Speech Recognition with Deep Recurrent Neural Networks

YouTube Video

$x = (x_1, \ldots, x_T) \rightarrow$ input sequence
$h = (h_1, \ldots, h_T) \rightarrow$ hidden vector sequence
$y = (y_1, \ldots, y_T) \rightarrow$ output vector sequence

RNN

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y \qquad t = 1, \ldots, T$$

LSTM

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right)$$
$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right)$$
$$h_t = o_t \tanh(c_t)$$

$W_{ci}, W_{cf}, W_{co} \rightarrow$ diagonal

Bidirectional RNN

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right)$$
$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right)$$
$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y$$

Deep RNN

$$h_t^n = \mathcal{H}\left(W_{h^{n-1}h^n}h_t^{n-1} + W_{h^n h^n}h_{t-1}^n + b_h^n\right)$$
$$y_t = W_{h^N y}h_t^N + b_y$$
$h^0 = x$ and $n = 1, \ldots, N$

$\Pr(y|x) \rightarrow$ differentiable distribution parametrized by the network
$x \rightarrow$ acoustic input sequence
$y \rightarrow$ phonetic output sequence
$z \rightarrow$ target output sequence
$\log \Pr(z|x) \rightarrow$ objective function
$T \rightarrow$ length of x & y
$U \rightarrow$ length of z

Connectionist Temporal Classification

$$y_t = W_{\overrightarrow{h}^N y}\overrightarrow{h}_t^N + W_{\overleftarrow{h}^N y}\overleftarrow{h}_t^N + b_y$$
$$\Pr(k|t) = \frac{\exp(y_t[k])}{\sum_{k'=1}^{K}\exp(y_t[k'])}$$
$$k = 1, \ldots, K, \underbrace{K+1}_{\text{blank symbol } \emptyset \text{ (non-output)}}$$

RNN Transducer

Combine a CTC-like network with a separate RNN that predicts each phoneme given the previous ones, thereby yielding a jointly trained acoustic and language model.

$t \rightarrow$ input timestep
$u \rightarrow$ output timestep

$$\Pr(k|t,u) = \frac{\exp(y_{t,u}[k])}{\sum_{k'=1}^{K}\exp(y_{t,u}[k'])}$$
$$y_{t,u} = W_{hy}h_{t,u} + b_y$$
$$h_{t,u} = \tanh(W_{lh}l_t + W_{ph}p_u + b_h)$$

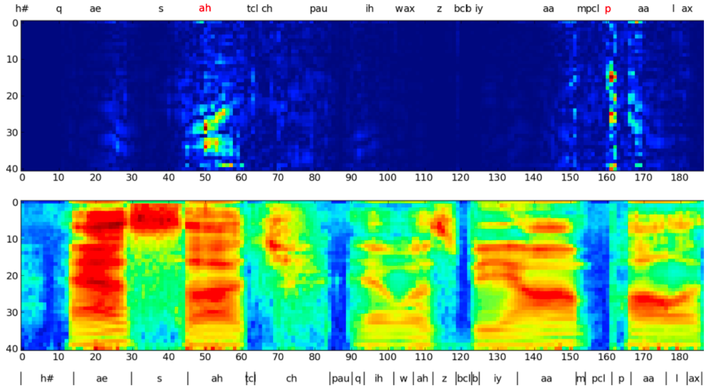
$p \rightarrow$ hidden state of the prediction network

$$l_t = W_{\overrightarrow{h}^N l}\overrightarrow{h}_t^N + W_{\overleftarrow{h}^N l}\overleftarrow{h}_t^N + b_l$$

$\overrightarrow{h}^N$ and $\overleftarrow{h}^N \rightarrow$ hidden sequences of the CTC network

Decoding: beam search

Network	Weights	Epochs	PER $\rightarrow$ phoneme error rate
CTC-3L-500H-TANH	3.7M	107	37.6% $\rightarrow$ RNN
CTC-1L-250H	0.8M	82	23.9%
CTC-1L-622H	3.8M	87	23.0%
CTC-2L-250H	2.3M	55	21.0%
CTC-3L-421H-UNI	3.8M	115	19.6% $\rightarrow$ unidirectional
CTC-3L-250H	3.8M	124	18.6%
CTC-5L-250H	6.8M	150	18.4%
TRANS-3L-250H	4.3M	112	18.3%
PRETRANS-3L-250H	**4.3M**	**144**	**17.7%**



Input Sensitivity of a deep CTC RNN. The heatmap (top) shows the derivatives of the 'ah' and 'p' outputs printed in red with respect to the filterbank inputs (bottom). The TIMIT ground truth segmentation is shown below.

Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks." *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013.

Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

YouTube Video

– Polyphonic music modeling
– speech signal modeling

RNN

$x = (x_1, x_2, \ldots, x_T)$
$h_0 = 0$
$h_t = \phi(h_{t-1}, x_t), t = 1, \ldots, T$
$y = (y_1, y_2, \ldots, y_T)$
$h_t = g(Wx_t + Uh_{t-1})$

Generative RNN

$p(x_1, \ldots, x_T) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_T \mid x_1, \ldots, x_{T-1})$
$p(x_t \mid x_1, \ldots, x_{t-1}) = g(h_t)$

LSTM

$h_t = o_t \tanh(c_t)$
$c_t \rightarrow$ memory cell
$o_t \rightarrow$ output gate
$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t)$
$V_o \rightarrow$ diagonal
$c_t = f_t c_{t-1} + i_t \tilde{c}_t$
$f_t \rightarrow$ forget gate
$i_t \rightarrow$ input gate
$\tilde{c}_t \rightarrow$ new memory
$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1})$
$f_t = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1})$
$i_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1})$
$V_f, V_i \rightarrow$ diagonal

Gated Recurrent Unit (GRU)

$h_t = (1 - z_t)h_{t-1} + z_t \tilde{h}_t$
$z_t = \sigma(W_z x_t + U_z h_{t-1}) \rightarrow$ update gate
$\tilde{h}_t = \tanh(Wx_t + U(r_t \odot h_{t-1}))$
$r_t = \sigma(W_r x_t + U_r h_{t-1}) \rightarrow$ reset gate
or $\tilde{h}_t = \tanh(Wx_t + r_t \odot (Uh_{t-1}))$
If $z_t = r_t = 1$ then $h_t = \tilde{h}_t = \tanh(Wx_t + Uh_{t-1})$.
If $z_t = 0$ then $h_t = h_{t-1}$.
If $z_t = 1$ and $r_t = 0$ then $h_t = \tilde{h}_t = \tanh(Wx_t)$.

Sequence modeling

$$\max_{\boldsymbol{\theta}} \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log p\left(x_t^n \mid x_1^n, \ldots, x_{t-1}^n; \boldsymbol{\theta}\right)$$

Polyphonic music modeling datasets

Nottingham, JSB Chorales, MuseData and Piano-midi.

Each symbol in these datasets is respectively a 93-, 96-, 105-, and 108-dimensional binary vector.

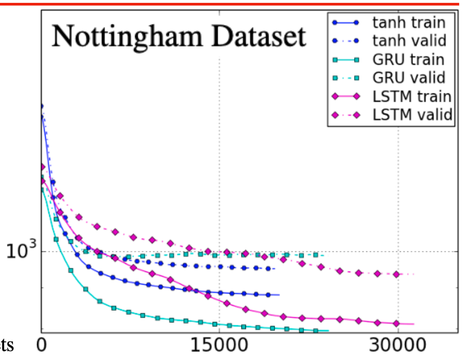
Logistic sigmoid function as output units

Speech signal modeling

Look at 20 consecutive samples to predict the following 10 consecutive samples in a one-dimensional raw audio signal.

Mixture of Gaussians with 20 components as output layer

Unit	# of Units	# of Parameters
Polyphonic music modeling		
LSTM	36	$\approx 19.8 \times 10^3$
GRU	46	$\approx 20.2 \times 10^3$
tanh	100	$\approx 20.1 \times 10^3$
Speech signal modeling		
LSTM	195	$\approx 169.1 \times 10^3$
GRU	227	$\approx 168.9 \times 10^3$
tanh	400	$\approx 168.4 \times 10^3$

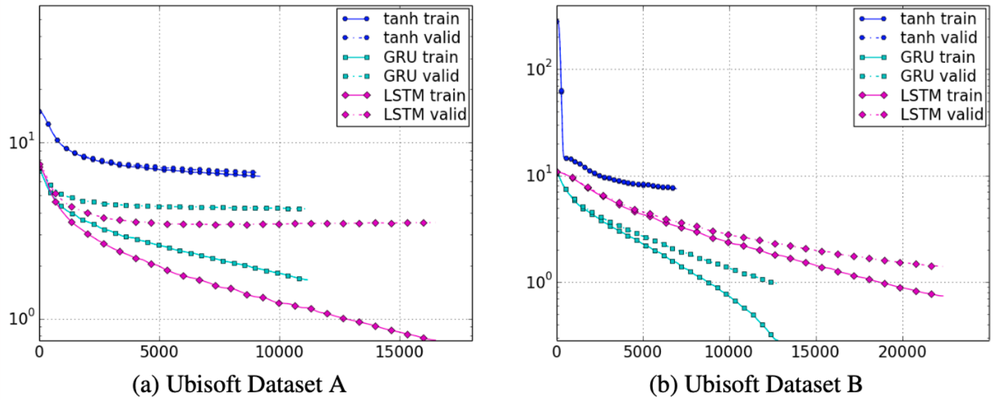


The average negative log-probabilities of the training and test sets

			tanh	GRU	LSTM
Music Datasets	Nottingham	train	3.22	2.79	3.08
		test	**3.13**	3.23	3.20
	JSB Chorales	train	8.82	6.94	8.15
		test	9.10	**8.54**	8.67
	MuseData	train	5.64	5.06	5.18
		test	6.23	**5.99**	6.23
	Piano-midi	train	5.64	4.93	6.49
		test	9.03	**8.82**	9.03
Ubisoft Datasets	Ubisoft dataset A	train	6.29	2.31	1.44
		test	6.44	3.59	**2.70**
	Ubisoft dataset B	train	7.61	0.38	0.80
		test	7.62	**0.88**	1.26

Results are not conclusive in comparing LSTM and GRU!

Wall Clock Time (seconds)



(a) Ubisoft Dataset A (b) Ubisoft Dataset B

Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint arXiv:1412.3555* (2014).

Towards End-to-End Speech Recognition with Recurrent Neural Networks

Boulder

YouTube Playlist

Automatic Speech Recognition (ASR): Transcribe Audio Data with Text

$x = (x_1, \ldots, x_T) \rightarrow$ input sequence (spectrogram)

$y = (y_1, \ldots, y_T) \rightarrow$ output sequence (Deep Bidirectional LSTM)

Connectionist Temporal Classification

Sayre's Paradox for automated handwriting recognition systems: A cursively written word cannot be recognized without being segmented and cannot be segmented without being recognized!

$$p(k,t|x) = \frac{\exp(y_t^k)}{\sum_{k'} \exp(y_t^{k'})} \rightarrow \text{probability of emitting the label (or blank) with index } k \text{ at time } t$$

The output layer contains a single unit for each of the transcription labels (characters, phonemes, musical notes etc.), plus an extra "blank" unit.

$a = (a_1, \ldots, a_T) \rightarrow$ CTC alignment (sequence of blank & label indices)

$\mathcal{B} \rightarrow$ operator that first removes the repeated labels & then the blanks

$\mathcal{B}(a-bc--) = abc = \mathcal{B}(--a-bc)$

$\mathcal{B}(abbbcc) = abc = \mathcal{B}(a-b-cc)$

$\Pr(z|x) = \sum_{a \in \mathcal{B}^{-1}(z)} \Pr(a|x) \rightarrow$ can be efficiently evaluated & differentiated using dynamic programming

$z = (z_1, \ldots, z_U) \rightarrow$ target sequence

$\text{CTC}(x) = -\log \Pr(z^*|x) \rightarrow$ objective function

$$\Pr(k|\boldsymbol{y}) = \frac{\sum_{w' \in (\boldsymbol{p}+k)*} \Pr^{\gamma}(w'|\boldsymbol{w})}{\sum_{w' \in \boldsymbol{p}*} \Pr^{\gamma}(w'|\boldsymbol{w})}$$

$\Pr(w'|w)$ is the probability assigned to the transition from the word history w to the word w', $p*$ is the set of dictionary words prefixed by p and γ is the language model weighting factor.

Expected Transcription Loss

$$\mathcal{L}(x) = \sum_z \Pr(z|x) \underbrace{\mathcal{L}(x,z)}_{\text{real-valued transcription loss (e.g., word error rate)}}$$

$$\mathcal{L}(x) = \sum_z \sum_{a \in \mathcal{B}^{-1}(z)} \Pr(a|x)\mathcal{L}(x,z) = \sum_a \Pr(a|x)\mathcal{L}(x, \mathcal{B}(a))$$

$$\mathcal{L}(x) \approx \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(x, \mathcal{B}(a^i)), a^i \sim \Pr(a|x)$$

$$\frac{\partial \log Pr(\boldsymbol{a}|\boldsymbol{x})}{\partial \Pr(k,t|\boldsymbol{x})} = \frac{\delta_{\boldsymbol{a}_t k}}{\Pr(k,t|\boldsymbol{x})}$$

$$\nabla_x f(x) = f(x) \nabla_x \log f(x)$$

$$\frac{\partial \mathcal{L}(\boldsymbol{x})}{\partial \Pr(k,t|\boldsymbol{x})} = \sum_{\boldsymbol{a}} \frac{\partial \Pr(\boldsymbol{a}|\boldsymbol{x})}{\partial \Pr(k,t|\boldsymbol{x})} \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}))$$

$$= \sum_{\boldsymbol{a}} \Pr(\boldsymbol{a}|\boldsymbol{x}) \frac{\partial \log \Pr(\boldsymbol{a}|\boldsymbol{x})}{\partial \Pr(k,t|\boldsymbol{x})} \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}))$$

$$= \sum_{\boldsymbol{a}:\boldsymbol{a}_t=k} \Pr(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{a}_t = k) \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}))$$

$$\frac{\partial \mathcal{L}(\boldsymbol{x})}{\partial \Pr(k,t|\boldsymbol{x})} \approx \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}^{i,t,k}))$$

$$\boldsymbol{a}^i \sim \Pr(\boldsymbol{a}|\boldsymbol{x})$$

$$\boldsymbol{a}_{t'}^{i,t,k} = \boldsymbol{a}_{t'}^i \forall t' \neq t, \ \boldsymbol{a}_t^{i,t,k} = k$$

$$\frac{\partial \mathcal{L}(\boldsymbol{x})}{\partial y_t^k} \approx \frac{\Pr(k,t|\boldsymbol{x})}{N} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}^i)) - \mathcal{Z}(\boldsymbol{a}^i, t)$$

$$\mathcal{Z}(\boldsymbol{a}^i, t) = \sum_{k'} \Pr(k', t|\boldsymbol{x}) \mathcal{L}(\boldsymbol{x}, \mathcal{B}(\boldsymbol{a}^{i,t,k'}))$$

With CTC, label activations are the probability of making transitions into different states, and the blank activation is the probability of remaining in the current state.

Decoding

$$\arg\max_z \Pr(z|x) \approx \mathcal{B}(\arg\max_a \Pr(a|x))$$

$\Pr^-(y,t) \rightarrow$ blank probability assigned to some (partial) output transcription y, at time t by the beam search

$\Pr^+(y,t) \rightarrow$ non-blank probability ...

$\Pr(y,t) = \Pr^-(y,t) + \Pr^+(y,t) \rightarrow$ total probability ...

$$\Pr(k,\boldsymbol{y},t) = \Pr(k,t|\boldsymbol{x}) \Pr(k|\boldsymbol{y}) \begin{cases} \Pr^-(\boldsymbol{y},t-1) \text{ if } \boldsymbol{y}^e = k \\ \Pr(\boldsymbol{y},t-1) \text{ otherwise} \end{cases}$$

extension probability of y by label k at time t

$\Pr(k|y)$ is the transition probability from y to $y+k$ and y^e is the final label in y. Define $\hat{y}$ as the prefix of y with the last label removed, and $\emptyset$ as the empty sequence, noting that $\Pr^+(\emptyset, t) = 0, \forall t$.

$\Pr(k|y) \rightarrow$ prior linguistic information

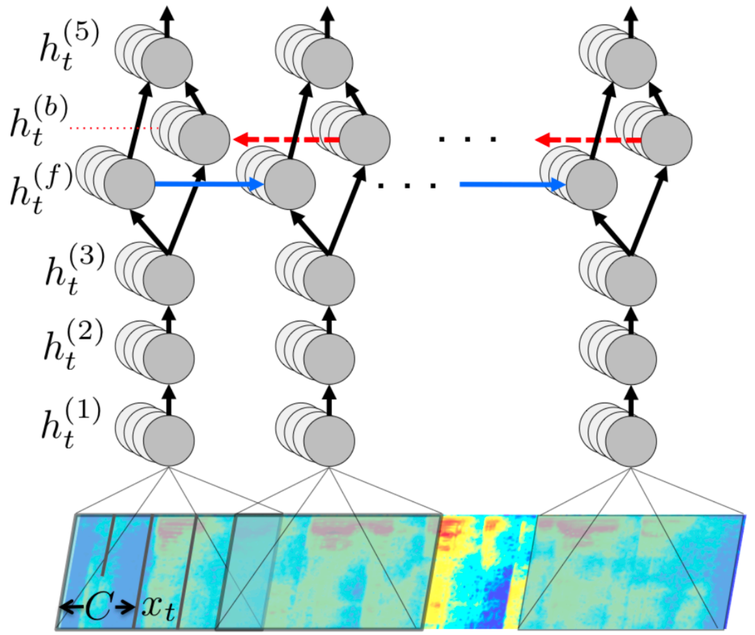
Algorithm 1 CTC Beam Search

Initalise: $B \leftarrow \{\varnothing\}$; $\Pr^-(\varnothing, 0) \leftarrow 1$
for $t = 1 \ldots T$ **do**
  $\hat{B} \leftarrow$ the W most probable sequences in B
  $B \leftarrow \{\}$
  for $\boldsymbol{y} \in \hat{B}$ **do**
    if $\boldsymbol{y} \neq \varnothing$ **then**
      $\Pr^+(\boldsymbol{y},t) \leftarrow \Pr^+(\boldsymbol{y},t-1) \Pr(\boldsymbol{y}^e, t|\boldsymbol{x})$
      if $\hat{\boldsymbol{y}} \in \hat{B}$ **then**
        $\Pr^+(\boldsymbol{y},t) \leftarrow \Pr^+(\boldsymbol{y},t) + \Pr(\boldsymbol{y}^e, \hat{\boldsymbol{y}}, t)$
    $\Pr^-(\boldsymbol{y},t) \leftarrow \Pr(\boldsymbol{y},t-1) \Pr(-, t|\boldsymbol{x})$
    Add $\boldsymbol{y}$ to B
  for $k = 1 \ldots K$ **do**
    $\Pr^-(\boldsymbol{y}+k,t) \leftarrow 0$
    $\Pr^+(\boldsymbol{y}+k,t) \leftarrow \Pr(k,\boldsymbol{y},t)$
    Add $(\boldsymbol{y}+k)$ to B
Return: $\max_{\boldsymbol{y} \in B} \Pr^{\frac{1}{|\boldsymbol{y}|}}(\boldsymbol{y}, T)$

Graves, Alex, and Navdeep Jaitly. "Towards end-to-end speech recognition with recurrent neural networks." *International conference on machine learning*. 2014.

Deep Speech: Scaling up end-to-end speech recognition

YouTube Video



$x \rightarrow$ a single utterance (speech spectrogram)
$y \rightarrow$ label (text transcription)
$\mathcal{X} = \{(x^1, y^1), (x^2, y^2), \ldots\} \rightarrow$ training set
$x^i \rightarrow$ utterance (time series of length T^i)
$x_t^i, t = 1, \ldots, T^i \rightarrow$ vector of audio features
$x_{t,p}^i \rightarrow$ power of the p-th frequency bin in the audio frame at time t
$\hat{y}_t = p(c_t|x)$
$c_t \in \{a, b, \ldots, z, space, apostrophe, blank\}$
$h^{(l)} \rightarrow$ hidden units at layers $l = 1, 2, \ldots, 5$
$h^{(0)} \rightarrow$ input

First layer: the output at each time t depends on the spectrogram frame x_t along with a context of C frames on each side.

$$h_t^{(l)} = g(W^{(l)} h_t^{(l-1)} + b^{(l)}) \qquad l = 1, 2, 3$$

$$g(z) = \min\{\max\{0, z\}, 20\}$$

The first three layers are not recurrent!

$$h_t^{(f)} = g(W^{(4)} h_t^{(3)} + W_r^{(f)} h_{t-1}^{(f)} + b^{(4)})$$

$$h_t^{(b)} = g(W^{(4)} h_t^{(3)} + W_r^{(b)} h_{t+1}^{(b)} + b^{(4)})$$

$$h_t^{(4)} = h_t^{(f)} + h_t^{(b)}$$

$$h_t^{(5)} = g(W^{(5)} h_t^{(4)} + b^{(5)})$$

$$h_{t,k}^{(6)} = \hat{y}_{t,k} \equiv \mathbb{P}(c_t = k|x) = \frac{\exp(W_k^{(6)} h_t^{(5)} + b_k^{(6)})}{\sum_j \exp(W_j^{(6)} h_t^{(5)} + b_j^{(6)})}$$

k-th column of the weight matrix

$\mathcal{L}(\hat{y}, y) \rightarrow$ CTC Loss

Decoding

$\arg\max_c Q(c) \rightarrow$ beam search

$$Q(c) = \log(\mathbb{P}(c|x)) + \alpha \log(\mathbb{P}_{\text{lm}}(c)) + \beta \, \text{word\_count}(c)$$

N-gram Language Model

Dataset	Type	Hours	Speakers
WSJ	read	80	280
Switchboard	conversational	300	4000
Fisher	conversational	2000	23000
Baidu	read	5000	9600

RNN output
what is the weather like in bostin right now prime miniter nerenr modi arther n tickets for the game

Decoded Transcription (Language Model)
what is the weather like in boston right now prime minister narendra modi are there any tickets for the game

Model	SWB	CH	Full
Vesely et al. (GMM-HMM BMMI) [44]	18.6	33.0	25.8
Vesely et al. (DNN-HMM sMBR) [44]	12.6	24.1	18.4
Maas et al. (DNN-HMM SWB) [28]	14.6	26.3	20.5
Maas et al. (DNN-HMM FSH) [28]	16.0	23.7	19.9
Seide et al. (CD-DNN) [39]	16.1	n/a	n/a
Kingsbury et al. (DNN-HMM sMBR HF) [22]	13.3	n/a	n/a
Sainath et al. (CNN-HMM) [36]	11.5	n/a	n/a
Soltau et al. (MLP/CNN+I-Vector) [40]	**10.4**	n/a	n/a
Deep Speech SWB	20.0	31.8	25.9
Deep Speech SWB + FSH	12.6	**19.3**	**16.0**

System	Clean (94)	Noisy (82)	Combined (176)
Apple Dictation	14.24	43.76	26.73
Bing Speech	11.73	36.12	22.05
Google API	6.64	30.47	16.72
wit.ai	7.94	35.06	19.41
Deep Speech	**6.56**	**19.06**	**11.85**

Hannun, Awni, et al. "Deep speech: Scaling up end-to-end speech recognition." *arXiv preprint arXiv:1412.5567* (2014).

WaveNet: A Generative Model for Raw Audio

Boulder

YouTube Playlist

$x = \{x_1, x_2, \ldots, x_T\} \rightarrow$ waveform

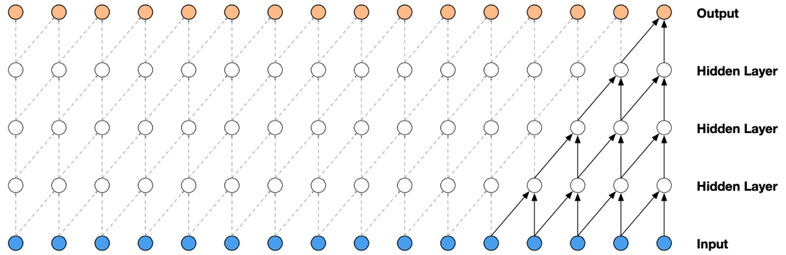
$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

$p(x_t | x_1, \ldots, x_{t-1}) \rightarrow$ modeled by a stack of convolutional layers

$x_t \rightarrow$ audio sample

categorical distribution over the next value x_t with a softmax layer

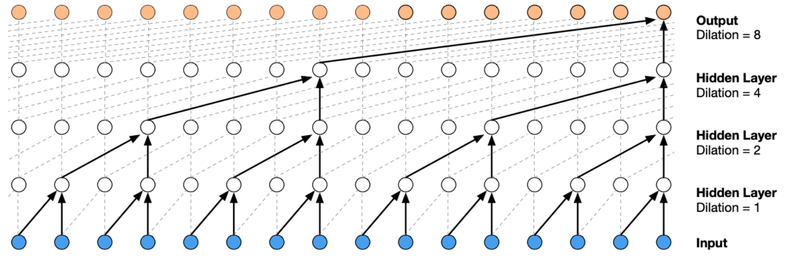
Causal Convolutions



receptive field $= 5 = (\underbrace{\#\text{ layers}}_{=4} + \underbrace{\text{filter length}}_{=2} - 1)$

Dilated Causal Convolutions

à trous convolutions (convolutions with holes)



$\underbrace{1, 2, 4, \ldots, 512}_{\text{receptive field size } 1024}, 1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512 \rightarrow$ dilations

Softmax Distributions

sequence of 16-bit integer values (one per time step)

$2^{16} = 65,536$ probabilities per time step $\rightarrow$ softmax

$$f(x_t) = \text{sign}(x_t) \frac{\ln(1 + \mu|x_t|)}{1 + \mu} \rightarrow \mu\text{-law companding transformation}$$

$-1 < x_t < 1$ and $\mu = 255$

Quantize the transformed output to 256 possible values

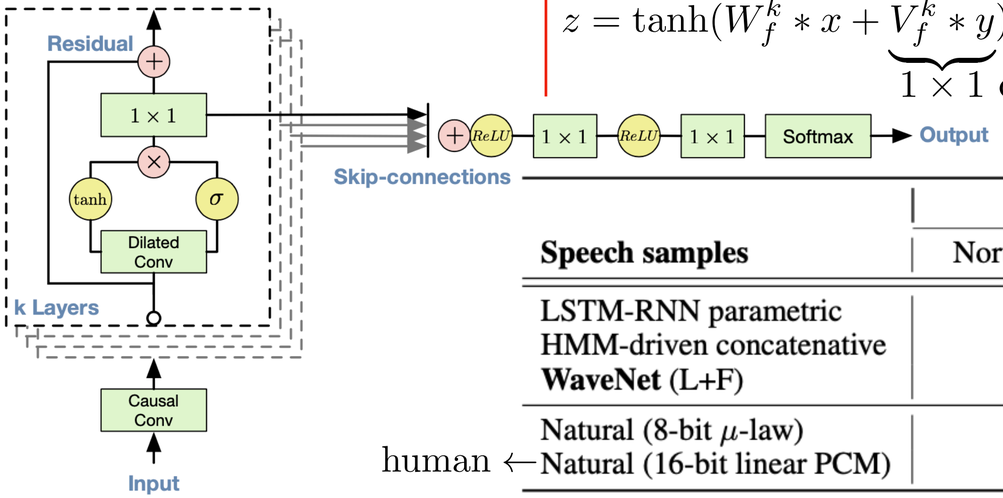
Gated Activation Units

$$z = \tanh(W_f^k * x) \odot \sigma(W_g^k * x)$$

$k \rightarrow$ layer index

f & $g \rightarrow$ filter & gate

Residual and Skip Connections



Conditional WaveNets

$h \rightarrow$ additional input

multi-speaker setting: speaker identity

text-to-speech (TTS): text

$$p(x|h) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}, h)$$

Global Conditioning

$$z = \tanh(W_f^k * x + \underbrace{(V_f^k)^T h}_{\text{broadcast over all time dimensions}}) \odot \sigma(W_g^k * x + (V_g^k)^T h)$$

Local Conditioning

$h = \{h_t\} \rightarrow$ lower sampling frequency than the audio signal

$y = f(h)$

$y \rightarrow$ same resolution as the audio signal

$f \rightarrow$ learned upsampling (transposed convolutional network)

$$z = \tanh(W_f^k * x + \underbrace{V_f^k * y}_{1 \times 1 \text{ convolution}}) \odot \sigma(W_g^k * x + V_g^k * y)$$

text-to-speech

Speech samples	Subjective 5-scale MOS in naturalness: North American English	Subjective 5-scale MOS in naturalness: Mandarin Chinese
LSTM-RNN parametric	3.67 ± 0.098	3.79 ± 0.084
HMM-driven concatenative	3.86 ± 0.137	3.47 ± 0.108
WaveNet (L+F)	$\mathbf{4.21} \pm 0.081$	$\mathbf{4.08} \pm 0.085$
Natural (8-bit μ-law)	4.46 ± 0.067	4.25 ± 0.082
human $\leftarrow$ Natural (16-bit linear PCM)	4.55 ± 0.075	4.21 ± 0.071

Oord, Aaron van den, et al. "Wavenet: A generative model for raw audio." *arXiv preprint arXiv:1609.03499* (2016).

LSTM: A Search Space Odyssey

Boulder

YouTube Video

- handwriting recognition & generation
- language modeling & translation
- acoustic modeling of speech
- speech synthesis
- protein secondary structure prediction
- analysis of audio
- video data

Large-scale analysis of eight LSTM variants on three representative tasks: speech recognition, handwriting recognition, and polyphonic music modeling.

Vanilla LSTM

$x^t \rightarrow$ input vector at time t

$N \rightarrow$ number of LSTM blocks

$M \rightarrow$ number of inputs

$W_z, W_i, W_f, W_o \in \mathbb{R}^{N \times M} \rightarrow$ input weights

$R_z, R_i, R_f, R_o \in \mathbb{R}^{N \times N} \rightarrow$ recurrent weights

$p_i, p_f, p_o \in \mathbb{R}^N \rightarrow$ peephole weights

not very critical

$b_z, b_i, b_f, b_o \in \mathbb{R}^N \rightarrow$ bias weights

$$g(x) = h(x) = \tanh(x)$$

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

$$\bar{z}^t = W_z x^t + R_z y^{t-1} + b_z$$

$$z^t = g(\bar{z}^t) \rightarrow \text{block input}$$

$$\bar{i}^t = W_i x^t + R_i y^{t-1} + p_i \odot c^{t-1} + b_i$$

$$i^t = \sigma(\bar{i}^t) \rightarrow \text{input gate}$$

$$\bar{f}^t = W_f x^t + R_f y^{t-1} + p_f \odot c^{t-1} + b_f$$

$$f^t = \sigma(\bar{f}^t) \rightarrow \text{forget gate}$$

$$c^t = z^t \odot i^t + c^{t-1} \odot f^t \rightarrow \text{critical cell}$$

$$\bar{o}^t = W_o x^t + R_o y^{t-1} + p_o \odot c^t + b_o$$

$$o^t = \sigma(\bar{o}^t) \rightarrow \text{output gate}$$

$$y^t = h(c^t) \odot o^t \rightarrow \text{block output}$$

NIG: No input gate: $\mathbf{i}^t = \mathbf{1}$

NFG: No forget gate: $\mathbf{f}^t = \mathbf{1}$

NOG: No output gate: $\mathbf{o}^t = \mathbf{1}$

NIAF: No input activation function: $g(\mathbf{x}) = \mathbf{x}$

NOAF: No output activation function: $h(\mathbf{x}) = \mathbf{x}$

CIFG: Coupled input and forget gate: $\mathbf{f}^t = \mathbf{1} - \mathbf{i}^t$

NP: No peepholes:

$$\bar{\mathbf{i}}^t = \mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{b}_i$$

$$\bar{\mathbf{f}}^t = \mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{b}_f$$

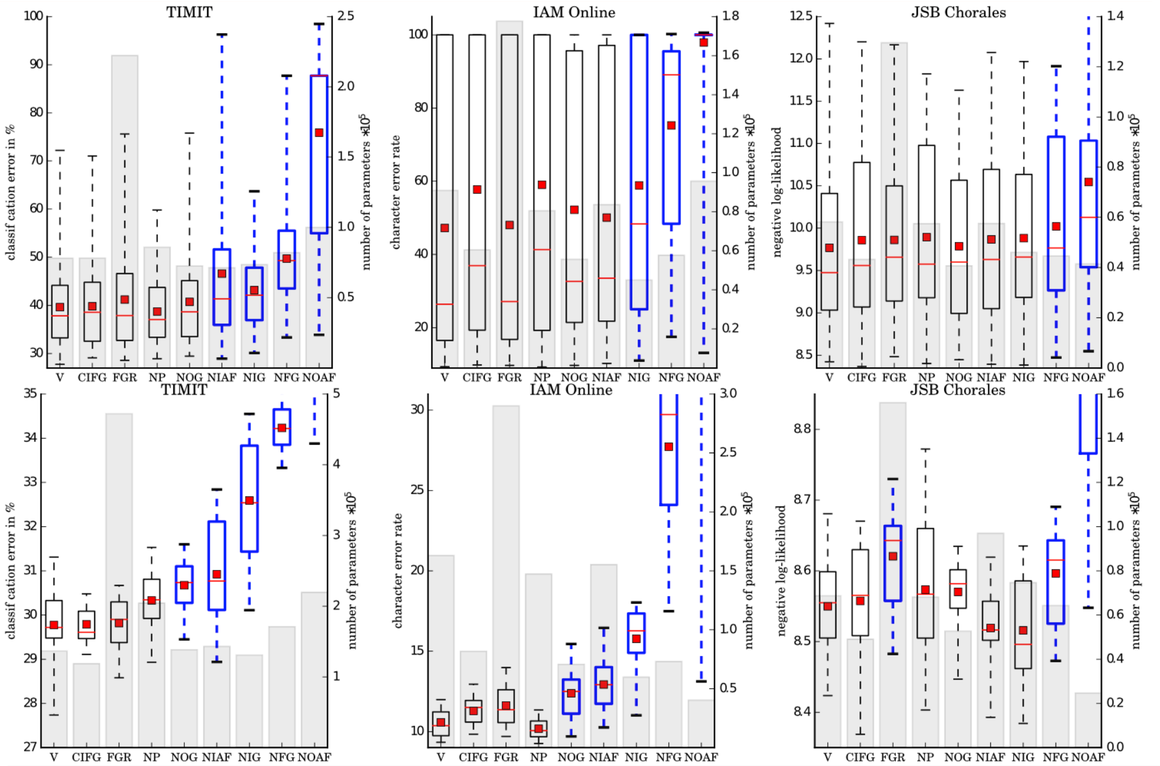
$$\bar{\mathbf{o}}^t = \mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{b}_o.$$

FGR: Full gate recurrence

$$\bar{\mathbf{i}}^t = \mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i + \mathbf{R}_{ii} \mathbf{i}^{t-1} + \mathbf{R}_{fi} \mathbf{f}^{t-1} + \mathbf{R}_{oi} \mathbf{o}^{t-1}$$

$$\bar{\mathbf{f}}^t = \mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f + \mathbf{R}_{if} \mathbf{i}^{t-1} + \mathbf{R}_{ff} \mathbf{f}^{t-1} + \mathbf{R}_{of} \mathbf{o}^{t-1}$$

$$\bar{\mathbf{o}}^t = \mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^{t-1} + \mathbf{b}_o + \mathbf{R}_{io} \mathbf{i}^{t-1} + \mathbf{R}_{fo} \mathbf{f}^{t-1} + \mathbf{R}_{oo} \mathbf{o}^{t-1}.$$



Test set performance for all 200 trials (top) and for the best 10% (bottom) trials (according to the validation set) for each data set and variant. Boxes: range between the 25th and 75th percentiles of the data. Whiskers: whole range. Red dot: mean. Red line: median of the data. Thick blue lines: boxes of variants that differ significantly from the vanilla LSTM. Gray histogram in the background: average number of parameters for the top 10% performers of every variant.

TIMIT: framewise classification task (classify each audio as one of 61 phones)

IAM Online: handwriting database

Each frame of the sequence is a 4-D vector containing ?x, ?y (the change in pen position), t (time since the beginning of the stroke), and a fourth dimension that contains value of one at the time of the pen lifting (a transition to the next stroke) and zeroes at all other time steps.

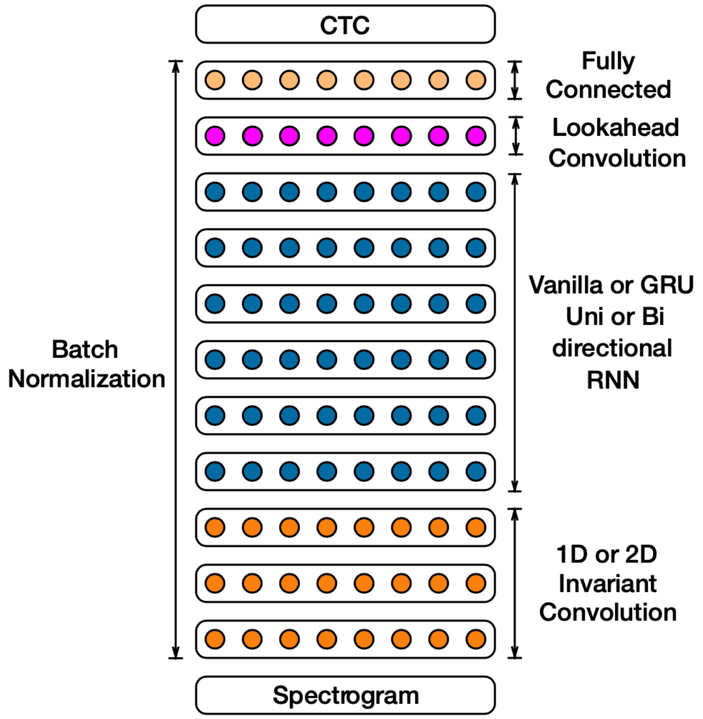
JSB Chorales: a collection of 382 four-part harmonized chorales by Bach

Greff, Klaus, et al. "LSTM: A search space odyssey." IEEE transactions on neural networks and learning systems 28.10 (2016): 2222-2232.

Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin

Boulder

YouTube Video



The inputs to the network are a sequence of log-spectrograms of power normalized audio clips, calculated on 20ms windows.

The outputs are the alphabet of each language.

$p(\ell_t|x) \rightarrow$ RNN prediction

$\ell_t \rightarrow$ either a character in the alphabet or the blank symbol

$\ell_t \in \{a, b, \ldots, z, space, apostrophe, blank\}$

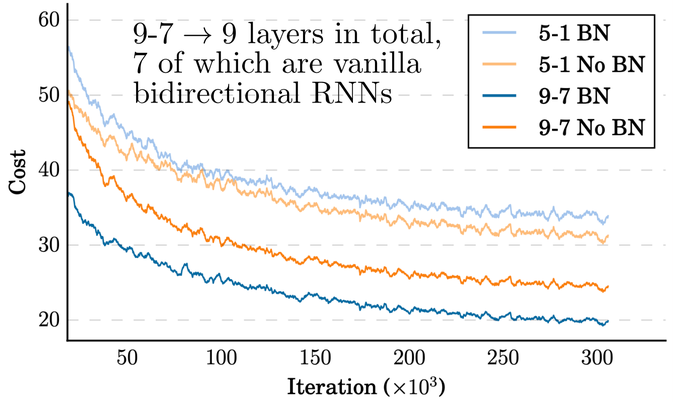
inference $\rightarrow \arg\max_y Q(y)$

$$Q(y) = \log(p_{\text{RNN}}(y|x)) + \alpha \log(p_{\text{LM}}(y)) + \beta \text{wc}(y)$$

$$h_t^l = f(W^l h_t^{l-1} + U^l h_{t-1}^l + b)$$

$$h_t^l = f(\mathcal{B}(W^l h_t^{l-1}) + U^l h_{t-1}^l)$$

$\mathcal{B} \rightarrow$ batch normalization



Architecture	Baseline		BatchNorm	GRU
5-layer, 1 RNN	13.55	WER	14.40	10.53
5-layer, 3 RNN	11.61		10.56	8.00
7-layer, 5 RNN	10.77		9.78	**7.79**
9-layer, 7 RNN	10.83		9.52	8.19
9-layer, 7 RNN no SortaGrad	11.96		9.78	

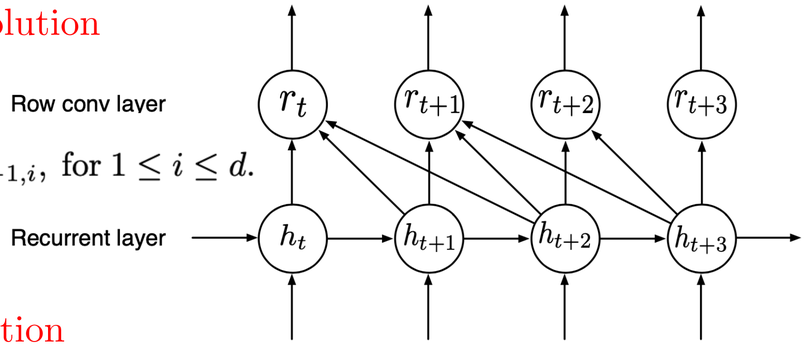
SortaGrad (curriculum learning): use the length of the utterance as a heuristic for difficulty and train on the shorter (easier) utterances first.

Convolution in the time-and-frequency domain (2D) and in the time-only domain (1D).

Lookahead Convolution

$$r_{t,i} = \sum_{j=1}^{\tau+1} W_{i,j} h_{t+j-1,i}, \text{ for } 1 \le i \le d.$$

$$W \in \mathbb{R}^{d \times \tau}$$



Dataset Construction

Given an audio-transcript pair (x, y), the most likely alignment is calculated as:

$$\ell^* = \arg\max_{\ell \in \text{Align}(x,y)} \prod_t^T p_{\text{ctc}}(\ell_t|x;\theta)$$

Fraction of Data	Hours	Regular Dev		Noisy Dev
1%	120	29.23	WER	50.97
10%	1200	13.80		22.99
20%	2400	11.65		20.41
50%	6000	9.51		15.90
100%	12000	8.46		13.59

The best English model has 2 layers of 2D convolution, followed by 3 layers of unidirectional recurrent layers with 2560 GRU cells each, followed by a lookahead convolution layer with $\tau = 80$, trained with BatchNorm and SortaGrad.

	Test set	Ours	Human
Read	WSJ eval'92	3.10	5.03
	WSJ eval'93	4.42	8.08
	LibriSpeech test-clean	5.15	5.83
	LibriSpeech test-other	12.73	12.69
Accented	VoxForge American-Canadian	7.94	4.85
	VoxForge Commonwealth	14.85	8.15
	VoxForge European	18.44	12.76
	VoxForge Indian	22.89	22.15
Noisy	CHiME eval real	21.59	11.84
	CHiME eval sim	42.55	31.33

Test	Mandarin	Human	RNN
100 utterances / committee		4.0	3.7
250 utterances / individual		9.7	5.7

Architecture	Dev	Test
5-layer, 1 RNN	7.13	15.41
5-layer, 3 RNN	6.49	11.85
5-layer, 3 RNN + BatchNorm	6.22	9.39
9-layer, 7 RNN + BatchNorm + frequency Convolution	5.81	7.93

Amodei, Dario, et al. "Deep speech 2: End-to-end speech recognition in english and mandarin." *International conference on machine learning*. 2016.

X-Vectors: Robust DNN Embeddings For Speaker Recognition

Boulder

The x-vector system

variable-length utterances $\mapsto$ fixed-dimensional embeddings (x-vectors)

$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T \rightarrow$ input segment of T frames (24 dimensional filterbanks)

Layer	Layer context	Total context	Input x output	
frame1	$[t-2, t+2]$	5	120x512	$\rightarrow 120 = 24 \times 5$
frame2	$\{t-2, t, t+2\}$	9	1536x512	$\rightarrow 1536 = 512 \times 3$
frame3	$\{t-3, t, t+3\}$	15	1536x512	
frame4	$\{t\}$	15	512x512	
frame5	$\{t\}$	15	512x1500	
stats pooling	$[0, T)$	T	1500Tx3000	$\rightarrow$ mean & std
segment6	$\{0\}$	T	3000x512	$\rightarrow$ embeddings
segment7	$\{0\}$	T	512x512	
softmax	$\{0\}$	T	512xN	

Probabilistic linear discriminant analysis (PLDA) classifier

The representations are centered, and projected using LDA.

Training Data

The extractor (embedding DNN) is trained on SWBD and SRE and the PLDA classifier is trained on just SRE.

Telephone and microphone speech

Evaluation

Equal error-rate (EER) and the minimum of the normalized detection cost function (DCF) at $P_{\text{Target}} = 10^{-2}$ and $P_{\text{Target}} = 10^{-3}$

Data Augmentation

– additive noise

– reverberation: convolving room impulse response (RIR) with audio

- **babble**: Three to seven speakers are randomly picked from MUSAN speech, summed together, then added to the original signal (13-20dB SNR).
- **music**: A single music file is randomly selected from MUSAN, trimmed or repeated as necessary to match duration, and added to the original signal (5-15dB SNR).
- **noise**: MUSAN noises are added at one second intervals throughout the recording (0-15dB SNR).
- **reverb**: The training recording is artificially reverberated via convolution with simulated RIRs.

```
Both MUSAN and the RIR datasets are freely available from
http://www.openslr.org.
```

Speakers in the Wild (SITW) & NIST SRE 2016 Cantonese (SRE16)

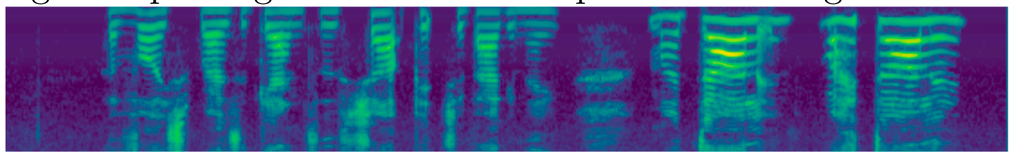
			SITW Core			SRE16 Cantonese		
			EER(%)	DCF10^{-2}	DCF10^{-3}	EER(%)	DCF10^{-2}	DCF10^{-3}
4.1	Original systems	i-vector (acoustic)	9.29	0.621	0.785	9.23	0.568	0.741
		i-vector (BNF)	**9.10**	**0.558**	**0.719**	9.68	0.574	0.765
		x-vector	9.40	0.632	0.790	**8.00**	**0.491**	**0.697**
4.2	PLDA aug.	i-vector (acoustic)	8.64	0.588	0.755	8.92	0.544	0.717
		i-vector (BNF)	8.00	**0.514**	**0.689**	8.82	0.532	0.726
		x-vector	**7.56**	0.586	0.746	**7.45**	**0.463**	**0.669**
4.3	Extractor aug.	i-vector (acoustic)	8.89	0.626	0.790	9.20	0.575	0.748
		i-vector (BNF)	7.27	**0.533**	0.730	8.89	0.569	0.777
		x-vector	**7.19**	0.535	**0.719**	**6.29**	**0.428**	**0.626**
4.4	PLDA and extractor aug.	i-vector (acoustic)	8.04	0.578	0.752	8.95	0.555	0.720
		i-vector (BNF)	6.49	0.492	0.690	8.29	0.534	0.749
		x-vector	**6.00**	**0.488**	**0.677**	**5.86**	**0.410**	**0.593**
4.5	Incl. VoxCeleb	i-vector (acoustic)	7.45	0.552	0.723	9.23	0.557	0.742
		i-vector (BNF)	6.09	0.472	0.660	8.12	0.523	0.751
		x-vector	**4.16**	**0.393**	**0.606**	**5.71**	**0.399**	**0.569**

Snyder, David, et al. "X-vectors: Robust dnn embeddings for speaker recognition." 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018.

SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition

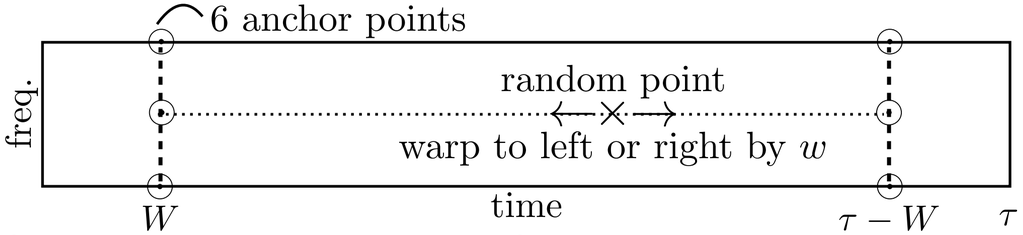
YouTube Playlist

log mel spectrogram of the base input with no augmentation

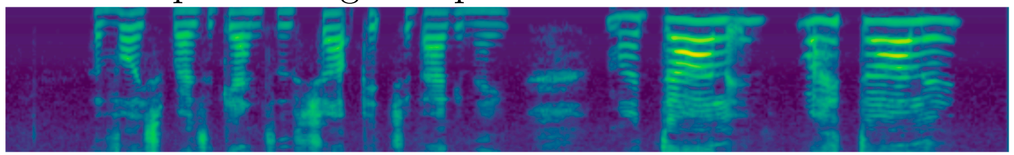


Time Warping

$\tau \rightarrow$ number of time steps in the log mel spectrogram
$W \rightarrow$ time warp parameter
$w \sim U(0, W)$

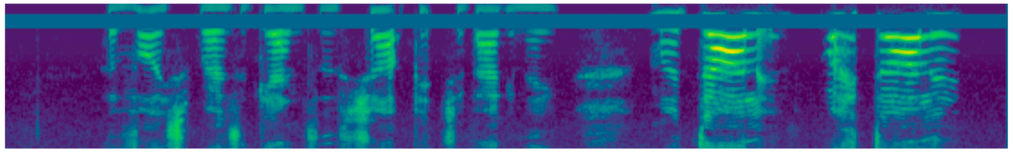


function sparse_image_warp of TensorFlow



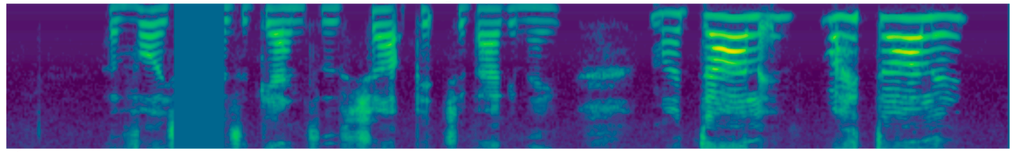
Frequency Masking

$[f_0, f_0 + f] \rightarrow f$ consecutive mel frequency channels to be masked
$f \sim U(0, F)$ $\quad F \rightarrow$ frequency mask parameter
$f_0 \rightarrow$ chosen from $[0, \nu - f]$
$\nu \rightarrow$ number of mel frequency channels



Time Masking

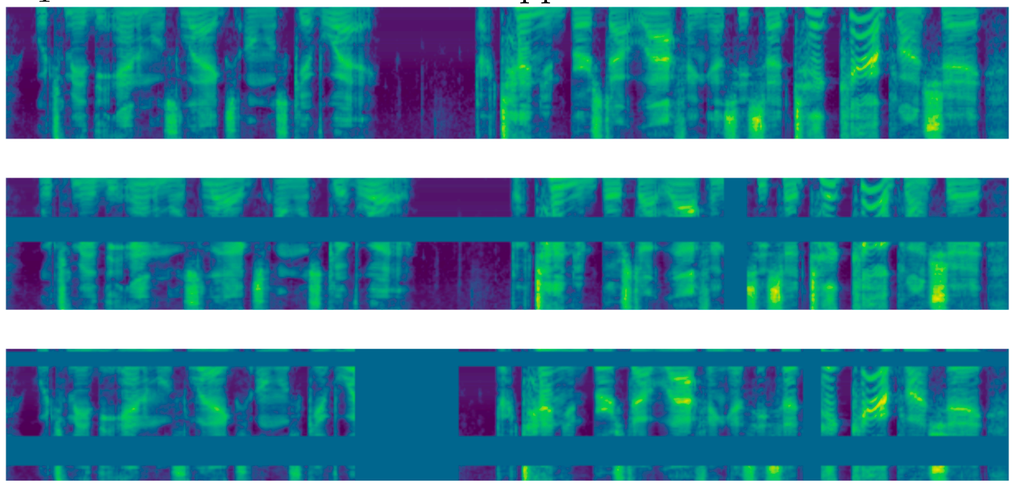
$[t_0, t_0 + t] \rightarrow t$ consecutive time steps to be masked
$t \sim U(0, T)$ $\quad T \rightarrow$ time mask parameter
$t_0 \rightarrow$ chosen from $[0, \tau - t]$



Augmentation Policies

Policy	W	F	m_F	T	p	m_T
None	0	0	-	0	-	-
LibriSpeech Basic $\leftarrow$ LB	80	27	1	100	1.0	1
LibriSpeech Double $\leftarrow$ LD	80	27	2	100	1.0	2
Switchboard Mild $\leftarrow$ SM	40	15	2	70	0.2	2
Switchboard Strong $\leftarrow$ SS	40	27	2	70	0.2	2

$m_F \rightarrow$ number of frequency masks applied
$m_T \rightarrow$ number of time masks applied



Listen, Attend and Spell (LAS)

log mel spectrogram $\triangleright$
2-layer CNN with max-pooling & stride 2 $\triangleright$
encoder (d stacked bidirectional LSTMS with cell size w)
$\rightarrow$ attention vectors
attention vectors $\triangleright$
2-layer RNN decoder of cell dimension w
$\rightarrow$ tokens for transcription

Text is tokenized using a Word Piece Model (WPM)

Language Model

$$\mathbf{y}^* = \underset{\mathbf{y}}{\text{argmax}} \left(\log P(\mathbf{y}|\mathbf{x}) + \lambda \log P_{LM}(\mathbf{y}) \right)$$

Results

Three training results for which time warping, time masking and frequency masking have been turned off, respectively.

test-other	test	
10.0	3.7	$\rightarrow$ Test set WER (%)
10.1	3.8	$\rightarrow$ Time Warping
11.0	4.0	$\rightarrow$ Time Masking
10.9	4.1	$\rightarrow$ Freq. Masking

Datasets

– LibriSpeech960h
– Switchboard300h

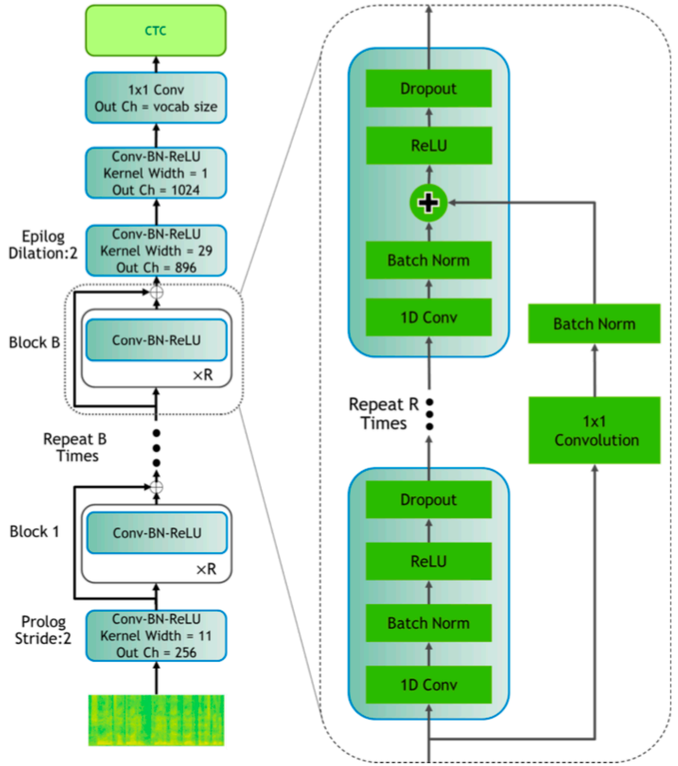
Park, Daniel S., et al. "Specaugment: A simple data augmentation method for automatic speech recognition." *arXiv preprint arXiv:1904.08779* (2019).

Jasper: An End-to-End Convolutional Neural Acoustic Model

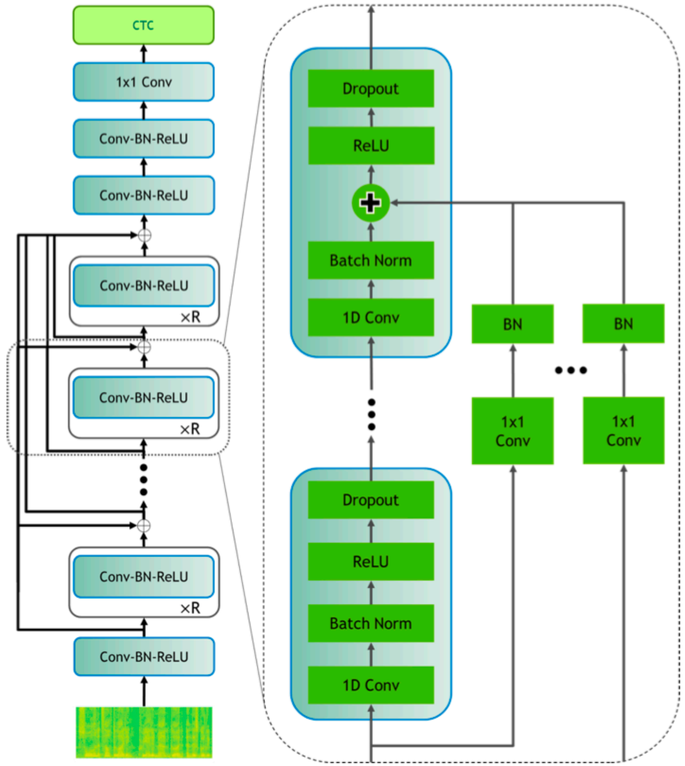
Boulder

YouTube Video

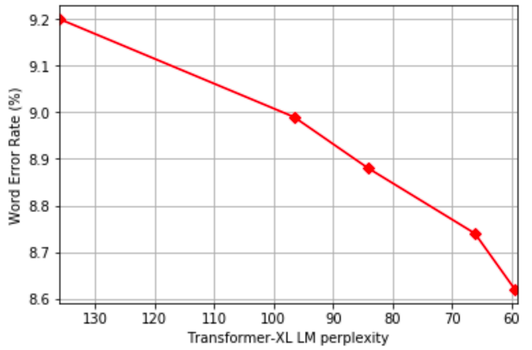
Jasper $B \times R$ model: B - number of blocks, R - number of sub-blocks



Jasper Dense Residual



# Blocks	Block	Kernel	# Output Channels	Dropout	# Sub Blocks
1	Conv1	11 *stride=2*	256	0.2	1
2	B1	11	256	0.2	5
2	B2	13	384	0.2	5
2	B3	17	512	0.2	5
2	B4	21	640	0.3	5
2	B5	25	768	0.4	5
1	Conv2	29 *dilation=2*	896	0.4	1
1	Conv3	1	1024	0.4	1
1	Conv4	1	# graphemes	0	1



NovoGrad

NovoGrad is similar to Adam, except that its second moments are computed per layer instead of per weight.

$$v_t^l = \beta_2 \cdot v_{t-1}^l + (1 - \beta_2) \cdot ||g_t^l||^2$$

$$m_t^l = \beta_1 \cdot m_{t-1}^l + \frac{g_t^l}{\sqrt{v_t^l + \epsilon}}$$

$$m_t^l = \beta_1 \cdot m_{t-1}^l + \frac{g_t^l}{\sqrt{v_t^l + \epsilon}} + d \cdot w_t \quad \text{(weight decay)}$$

$$w_{t+1} = w_t - \alpha_t \cdot m_t$$

Model	#params, M	Dev Clean	Dev Other
Residual	201	4.65	14.36
Dense Residual	211	4.51	**14.15**
DenseNet	205	4.77	14.55
DenseRNet	211	**4.32**	14.21

LibriSpeech, WER (%)

Model	E2E	LM	dev-clean	dev-other	test-clean	test-other
CAPIO (single) [23]	N	RNN	3.02	8.28	3.56	8.58
pFSMN-Chain [25]	N	RNN	2.56	7.47	2.97	**7.5**
DeepSpeech2 [26]	Y	5-gram	-	-	5.33	13.25
Deep bLSTM w/ attention [21]	Y	LSTM	3.54	11.52	3.82	12.76
wav2letter++ [27]	Y	ConvLM	3.16	10.05	3.44	11.24
LAS + SpecAugment [4] [28]	Y	RNN	-	-	*2.5*	*5.8*
Jasper DR 10x5	Y	-	3.64	11.89	3.86	11.95
Jasper DR 10x5	Y	6-gram	2.89	9.53	3.34	9.62
Jasper DR 10x5	Y	Transformer-XL	2.68	8.62	**2.95**	8.79
Jasper DR 10x5 + Time/Freq Masks [4]	Y	Transformer-XL	2.62	7.61	2.84	7.84

WSJ End-to-End Models, WER (%)

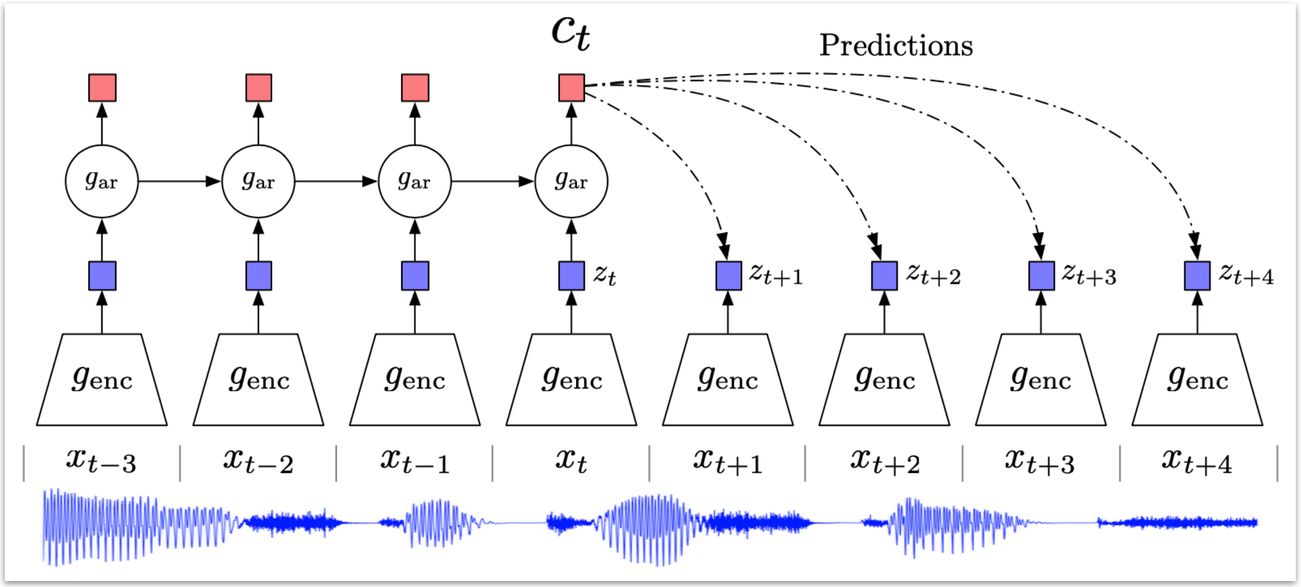
Model	LM	nov93	nov92
seq2seq + deep conv [35]	-	-	10.5
wav2letter++ [27]	4-gram	9.5	5.6
wav2letter++ [27]	ConvLM	7.5	4.1
E2E LF-MMI [14]	3-gram	-	4.1
Jasper 10x3	-	16.1	13.3
Jasper 10x3	4-gram	9.9	7.1
Jasper 10x3	Transformer-XL	9.3	6.9

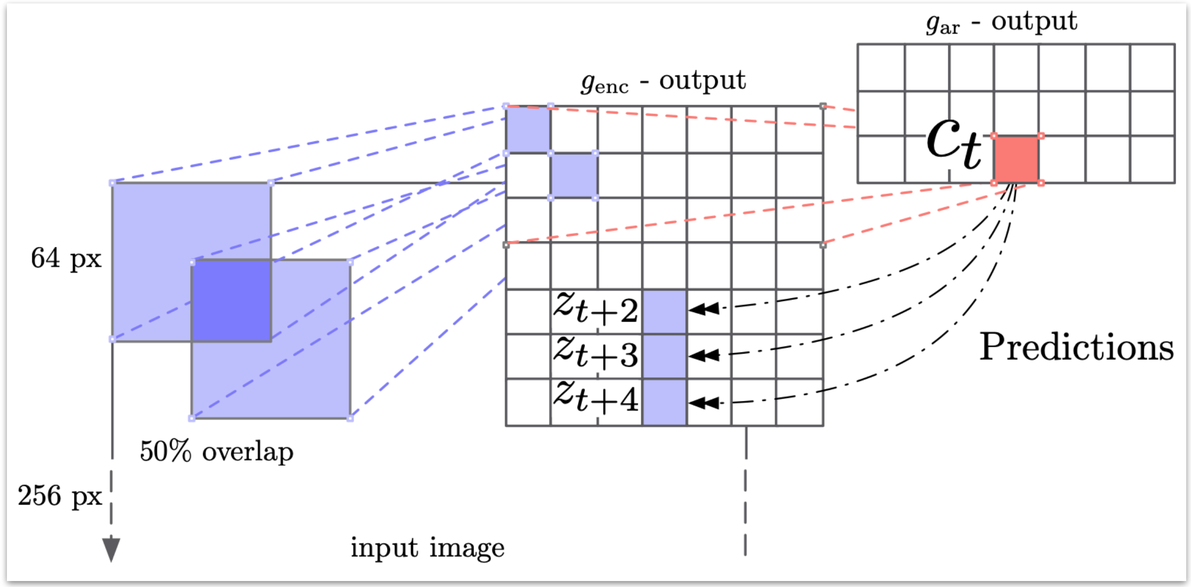
Hub5'00, WER (%)

Model	E2E	LM	SWB	CHM
LF-MMI [14]	N	RNN	7.3	14.2
Attention Seq2Seq [36]	Y	-	8.3	15.5
RNN-T [37]	Y	4-gram	8.1	17.5
Char E2E LF-MMI [14]	Y	RNN	8.0	17.6
Phone E2E LF-MMI [14]	Y	RNN	7.5	14.6
CTC + Gram-CTC	Y	N-gram	7.3	14.7
Jasper DR 10x5	Y	4-gram	8.3	19.3
Jasper DR 10x5	Y	Transformer-XL	7.8	16.2

Li, Jason, et al. "Jasper: An end-to-end convolutional neural acoustic model." *arXiv preprint arXiv:1904.03288* (2019).

Representation Learning with Contrastive Predictive Coding



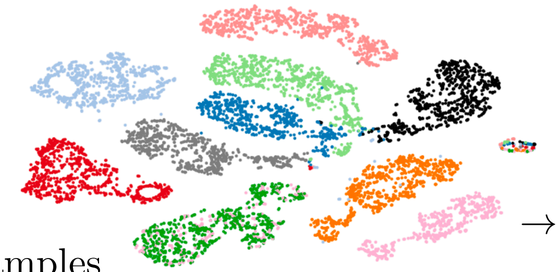


$g_{enc} \rightarrow$ non-linear encoder (e.g., strided convolutional layers with resnet blocks)
$x_t \rightarrow$ input observation
$z_t = g_{enc}(x_t) \rightarrow$ latent representation
$g_{ar} \rightarrow$ autoregressive model (e.g., GRUs)
$c_t = g_{ar}(z_{\leq t}) \rightarrow$ context latent representation (summarizing all $z_{\leq t}$ in the latent space)
$f_k(x_{t+k}, c_t) := \exp(z_{t+k}^T \underbrace{W_k c_t}_{\hat{z}_{t+k}})$

InfoNCE Loss

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$X = \{x_1, x_2, \ldots, x_N\} \rightarrow$ set of N random samples



$\rightarrow$ t-SNE visualization of speech (each color represents a different speaker)
Speech, images, text and reinforcement learning!

Containing one positive sample from $p(x_{t+k}|c_t)$ and $N-1$ negative samples from the proposal distribution $p(x_{t+k})$

Mutual Information Estimation

The optimal value for $f(x_{t+k}, c_t)$ is proportional to $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$!

No need to predict future observations x_{t+k} directly with a generative model $p(x_{t+k}|c_t)$. InfoNCE (Noise Contrastive Estimation) relieves the model from modeling the high dimensional distributions x_{t+k}.

$I(x_{t+k}; c_t) \geq \log N - \mathcal{L}_N$

$$I(x; c) = \sum_{x,c} p(x, c) \log \frac{p(x|c)}{p(x)} \rightarrow \text{mutual information}$$

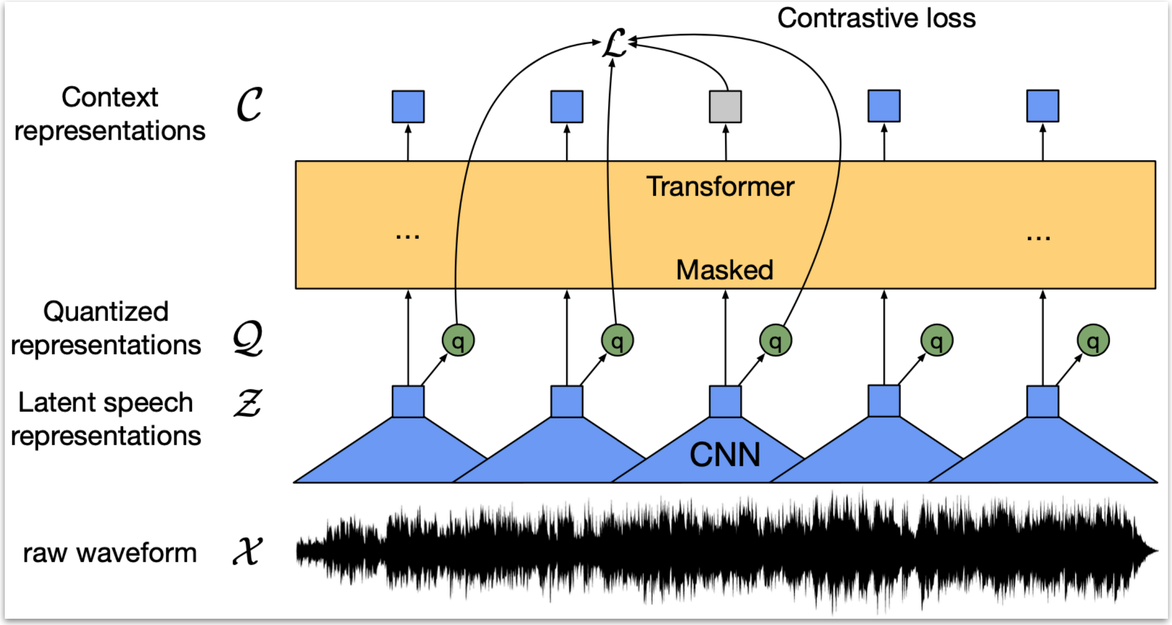
Minimizing $\mathcal{L}_N$ implies maximizing $I(x_{t+k}; c_t)$.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding." *arXiv preprint arXiv:1807.03748* (2018).

wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations

Boulder

YouTube Video



$f : \mathcal{X} \mapsto \mathcal{Z}$
↳ multi-layer convolutional feature encoder
$\mathcal{X} \rightarrow$ input raw audio
$\mathcal{Z} = (z_1, z_2, \ldots, z_T) \rightarrow$ latent speech representations
$g : \mathcal{Z} \mapsto \mathcal{C}$
↳ transformer
$\mathcal{C} = (c_1, c_2, \ldots, c_T) \rightarrow$ representations capturing information from the entire sequence

Instead of fixed positional embeddings which encode absolute positional information, use a convolutional layer which acts as relative positional embedding.

$\mathcal{Z} \mapsto \mathcal{Q}$
↳ quantization module
$\mathcal{Q} = (q_1, q_2, \ldots, q_T)$

diversity loss: encourage the model to use the codebook entries equally often $\leftarrow$

Quantization Module

$G \rightarrow$ number of codebooks/groups
$V \rightarrow$ number of entries
$e \in \mathbb{R}^{V \times d/G}$

Choose one entry/row from each codebook e and concatenate the resulting vectors $e_1, \ldots, e_G$ and apply a linear transformation $\mathbb{R}^d \rightarrow \mathbb{R}^f$ to obtain $q \in \mathbb{R}^f$.
The Gumbel softmax enables choosing discrete codebook entries in a fully differentiable way!
$z \mapsto l$
$z \rightarrow$ feature encoder output
$l \in \mathbb{R}^{G \times V} \rightarrow$ logits

$$p_{g,v} = \frac{\exp(l_{g,v} + n_v)/\tau}{\sum_{k=1}^{V} \exp(l_{g,k} + n_k)/\tau}$$

↳ probability of choosing the v-th codebook entry for group g
$\tau \rightarrow$ non-negative temperature
$n = -\log(-\log(u)) \rightarrow$ Gumbel noise $\quad u \sim U(0,1)$
Forward pass: $i = \arg\max_j p_{g,j} \rightarrow$ codeword i
Backward pass: true gradient of the Gumbel softmax outputs

	avg. WER	std.
Continuous inputs, quantized targets (Baseline)	7.97	0.02
Quantized inputs, quantized targets	12.18	0.41
Quantized inputs, continuous targets	11.18	0.16
Continuous inputs, continuous targets	8.58	0.08

TIMIT phoneme recognition accuracy in terms of phoneme error rate (PER).

	dev PER	test PER
CNN + TD-filterbanks [59]	15.6	18.0
PASE+ [47]	-	17.2
Li-GRU + fMLLR [46]	–	14.9
wav2vec [49]	12.9	14.7
vq-wav2vec [5]	9.6	11.6
This work (no LM) Connectionist Temporal Classification (CTC) loss		
LARGE (LS-960)	7.4	8.3

Pre-training

$$\mathcal{L}_m = -\log \frac{\exp(sim(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp(sim(\mathbf{c}_t, \tilde{\mathbf{q}})/\kappa)} \qquad sim(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|\|\mathbf{b}\|$$

↳ contrastive loss: identify the true quantized latent speech representation for a masked time step within a set of distractors.

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^{G} -H(\bar{p}_g) = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \bar{p}_{g,v} \log \bar{p}_{g,v} \qquad \mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_d$$

Baevski, Alexei, et al. "wav2vec 2.0: A framework for self-supervised learning of speech representations." *arXiv preprint arXiv:2006.11477* (2020).

Boulder

Questions?

YouTube Playlist