CS 224S / LINGUIST 285
Spoken Language Processing

Andrew Maas
Stanford University
Spring 2022

Lecture 10: End-to-end neural network speech recognition

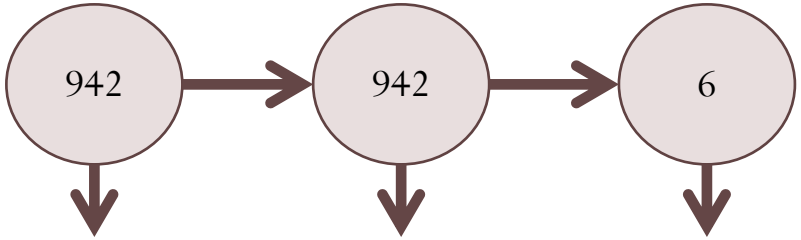
Outline

- Iterative training of HMM systems
- Connectionist temporal classification (CTC)
  - Lexicon-free CTC
- Listen, Attend, & Spell (LAS)
  - Combining CTC and LAS
- Convolutional transformer

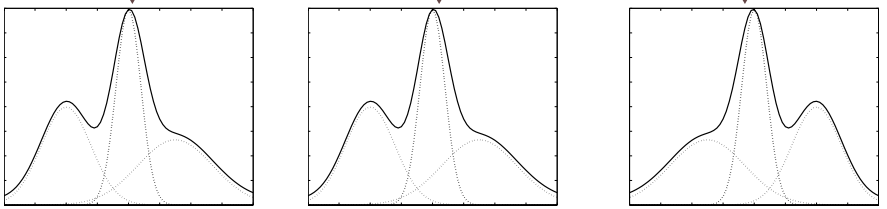
HMM-GMM ASR model

Transcription: Samson
Pronunciation: S – AE – M – S –AH – N
Sub-phones : 942 – 6 – 37 – 8006 – 4422 …

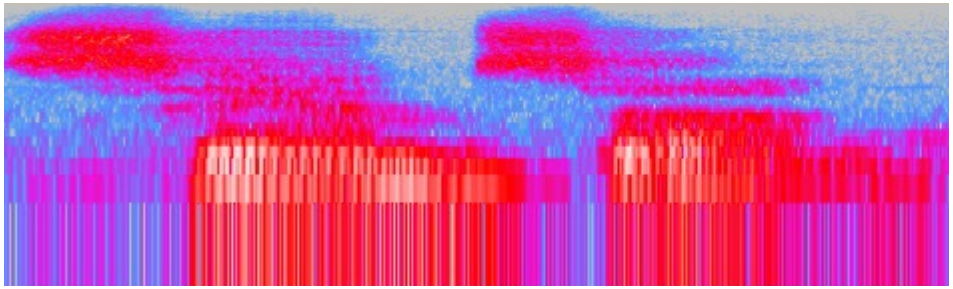
Hidden Markov Model (HMM):



Acoustic Model:



Audio Input:

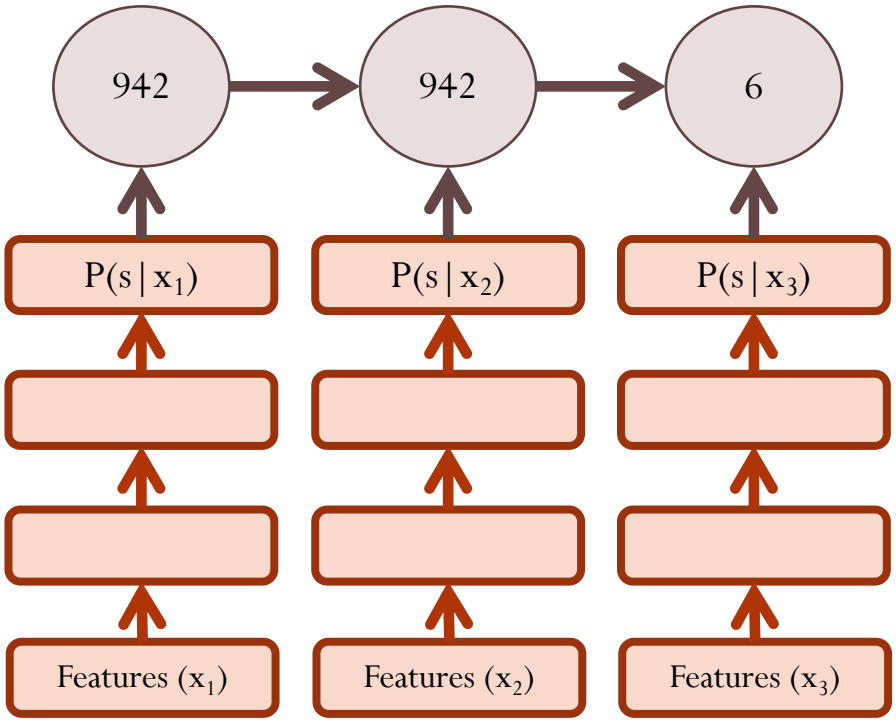


GMM models:
P(x|s)
x: input features
s: HMM state

DNN Hybrid Acoustic Models

Transcription: Samson
Pronunciation: S – AE – M – S –AH – N
Sub-phones : 942 – 6 – 37 – 8006 – 4422 …

Hidden Markov Model (HMM):



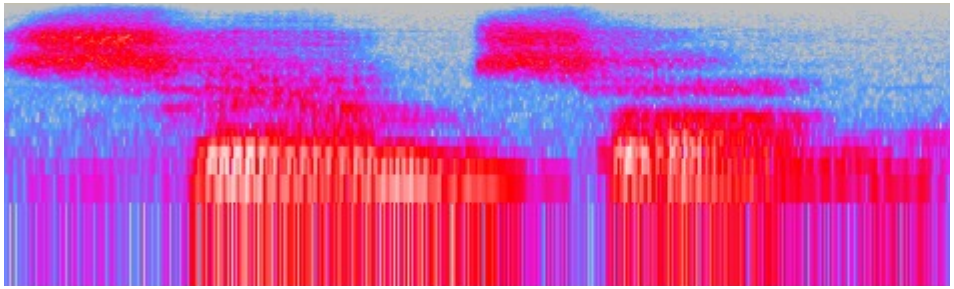
Acoustic Model:

Use a DNN to approximate: P(s|x)

Apply Bayes' Rule:
P(x|s) = P(s|x) * P(x) / P(s)

DNN * Constant / State prior

Audio Input:



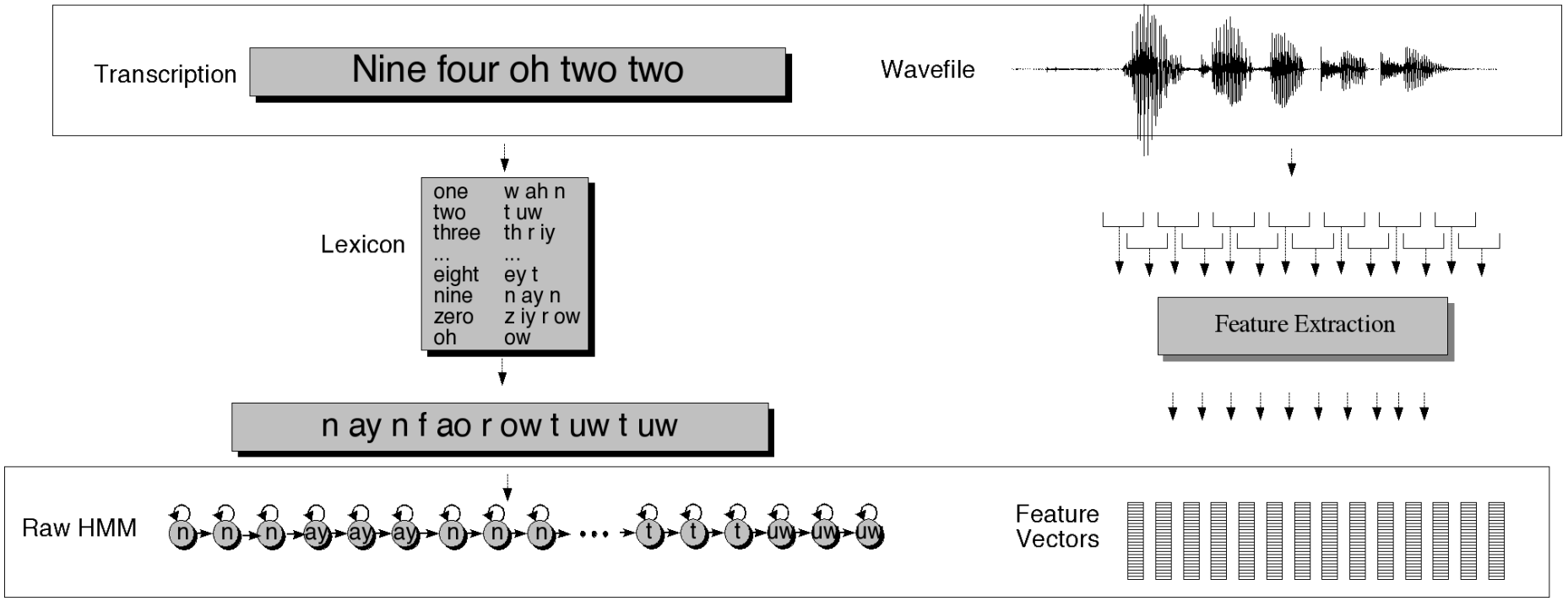
Training an HMM system (Viterbi)

- Given our lexicon + HMM structure, and some acoustic model, we can:
  - Generate the best *alignment* of HMM states to acoustic observations
- With an alignment of HMM states to observations:
  - Build a new acoustic model. Treat current state/obs mapping as training data+labels
  - This acoustic model is hopefully better than previous one
- Repeat the align -> rebuild acoustic model process until convergence
  - Add parameters / complexity to acoustic model each iteration

Forced Alignment

- Computing the "Viterbi path" over the training data is called "forced alignment"
- Because we know which word string to assign to each observation sequence.
- We just don't know the state sequence.
- So we use a_{ij} to constrain the path to go through the correct words
- And otherwise do normal Viterbi
- Result: state sequence!

HMM-GMM Embedded Training



Initialization: "Flat start"

- Transition probabilities:
  - Set to zero any that you want to be "structurally zero" (lexicon/pronunciation)
  - Set the rest to identical values
- Likelihoods:
  - Initialize GMM μ and σ of each state to global mean and variance of all training data

How can we improve?

- Lexicon introduces too many pronunciation assumptions. Requires hand engineering
- Iteratively building HMM systems requires complex “recipes” to progressively improve alignments + acoustic models
- HMM-DNN systems perform better, but still require the above
  - … can we use deep learning approaches to *replace* the HMM-based approaches so far?

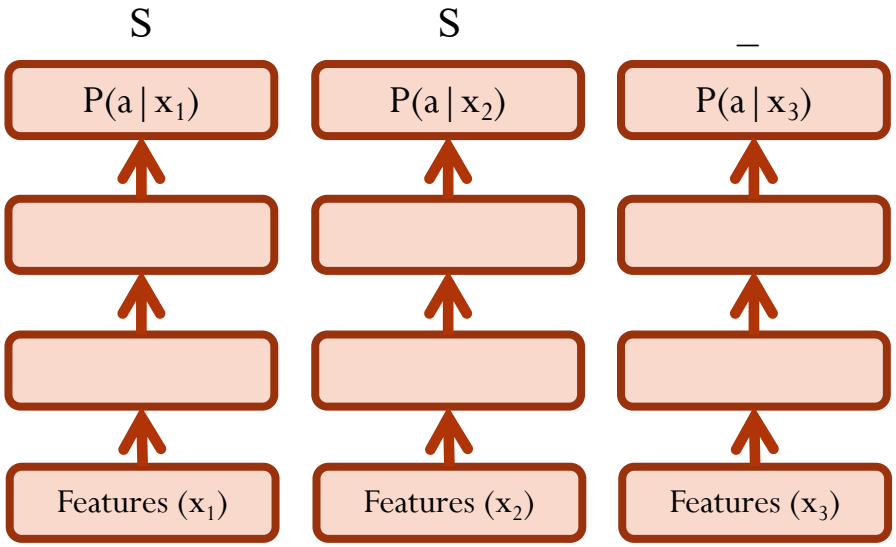
HMM-Free Recognition with CTC

Transcription: Samson

Characters: SAMSON

Collapsing function: SS___AA_M_S___O___NNNN

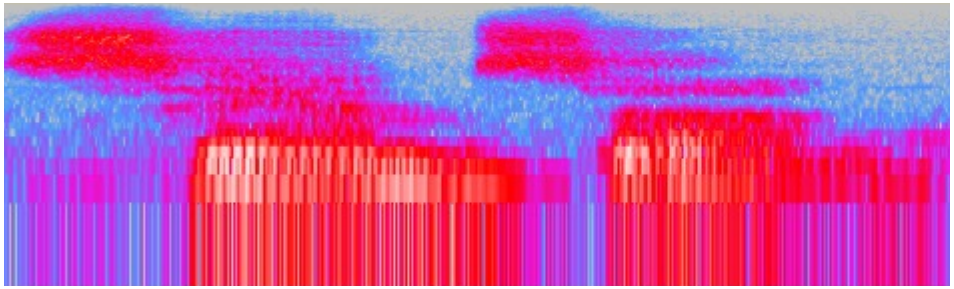
Acoustic Model:



Use a DNN to approximate: P(a|x)

The distribution over *characters*

Audio Input:



(Graves & Jaitly. 2014)

Example Results (WSJ)

YET A REHBILITATION CRU IS ONHAND IN THE BUILDING LOOGGING BRICKS PLASTER AND BLUEPRINS FOUR FORTY TWO NEW BETIN EPARTMENTS

YET A REHABILITATION CREW IS ON HAND IN THE BUILDING LUGGING BRICKS PLASTER AND BLUEPRINTS FOR FORTY TWO NEW BEDROOM APARTMENTS

THIS PARCLE GUNA COME BACK ON THIS ILAND SOM DAY SOO

THE SPARKLE GONNA COME BACK ON THIS ISLAND SOMEDAY SOON

TRADE REPRESENTIGD JUIDER WARANTS THAT THE U S WONT BACKCOFF ITS PUSH FOR TRADE BARIOR REDUCTIONS

TRADE REPRESENTATIVE YEUTTER WARNS THAT THE U S WONT BACK OFF ITS PUSH FOR TRADE BARRIER REDUCTIONS

TREASURY SECRETARY BAGER AT ROHIE WOS IN AUGGRAL PRESSED FOUR ARISE IN THE VALUE OF KOREAS CURRENCY

TREASURY SECRETARY BAKER AT ROH TAE WOOS INAUGURAL PRESSED FOR A RISE IN THE VALUE OF KOREAS CURRENCY

CTC Loss Function

Labels at each time index are conditionally independent (like HMMs). Model is *discriminative*

$$CTC(x) = -\log \Pr(y^*|x) \qquad \Pr(\mathbf{a}|x) = \prod_{t=1}^{T} \Pr(a_t, t|x)$$

Sum over all time-level labelings consistent with the output label.

$$\Pr(y|x) = \sum_{a \in \mathcal{B}^{-1}(y)} \Pr(a|x)$$

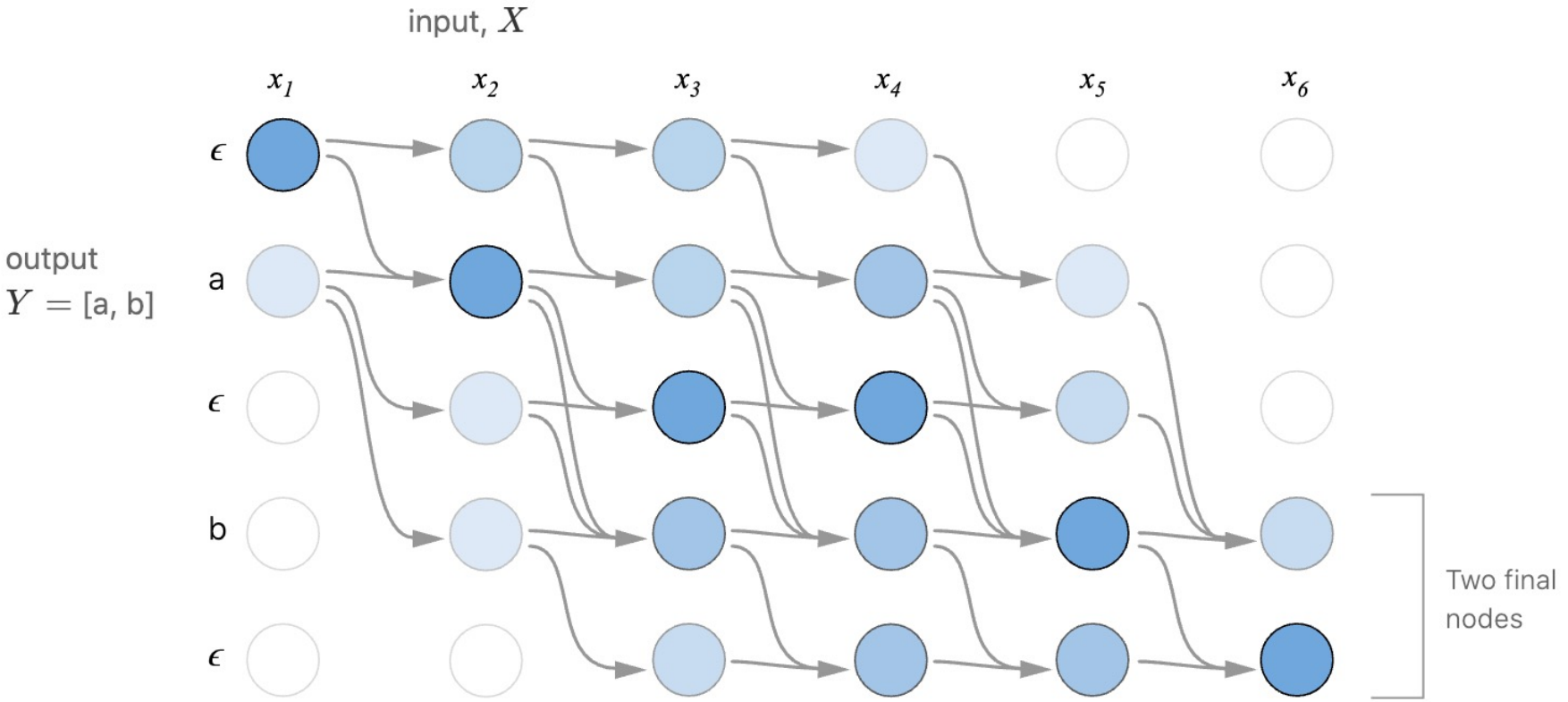
T=3, transcript: HI

Time-level labelings: HI_, H_I, _HI

Final loss maximizes probability of transcript y*

(Graves & Jaitly, ICML 2014)

CTC Loss Function: Dynamic Programming



Node (s, t) in the diagram represents $\alpha_{s,t}$ – the CTC score of the subsequence $Z_{1:s}$ after t input steps.

(Hannun. 2017)

Collapsing Example

Per-frame argmax:

_____
yy__ee_____tt_ _____a_____
_rr__e_____hh_____b___ii_____lll__i_____tt_____aa_____tt_____iio__n___
___cc_____rrr_u_____ _____ii___ss
_____o_____nn_____hhh_a_____nnddd _____i__n___
__thh_e_____ _____bb_uuii_____llldd____ii____nng_____
_____l___o___o_g__g___ii____nng_____
____b___rr_ii_____ck__s_____p___ll__a_____sstt_____eerr__
_____a___nnd_ ___b___lll_uu____ee__pp___r___i_____nnss_
_____f_____oou_____rrr_____ _____f_____oo__rrr__tt_y____
_____t____www_oo_____ ____nn___ew_____
_____b___e_____t_____i____n___
____e_____pp_____aa___rr___tt____mm_ee___nnntss
_____
_____

After collapsing:
yet a rehbilitation cru is onhand in the building loogging bricks plaster and blueprins four forty two new betin epartments

Reference:
yet a rehabilitation crew is on hand in the building lugging bricks plaster and blueprints for forty two new bedroom apartments

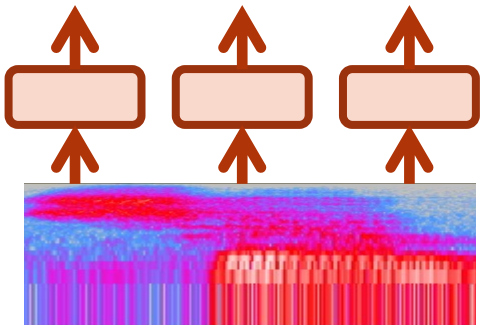
(Hannun, Maas, Jurafsky, & Ng. 2014)

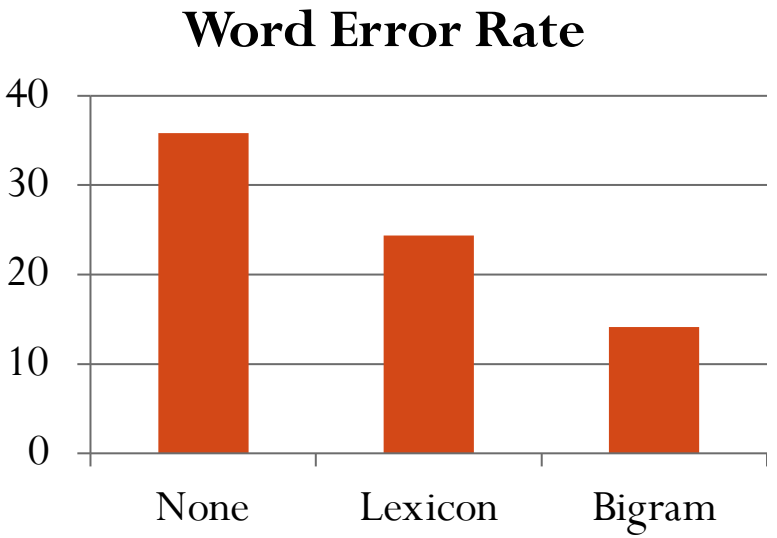
Decoding with a Language Model

Lexicon [a, …, zebra]

Language Model p(“yeah” | “oh”)

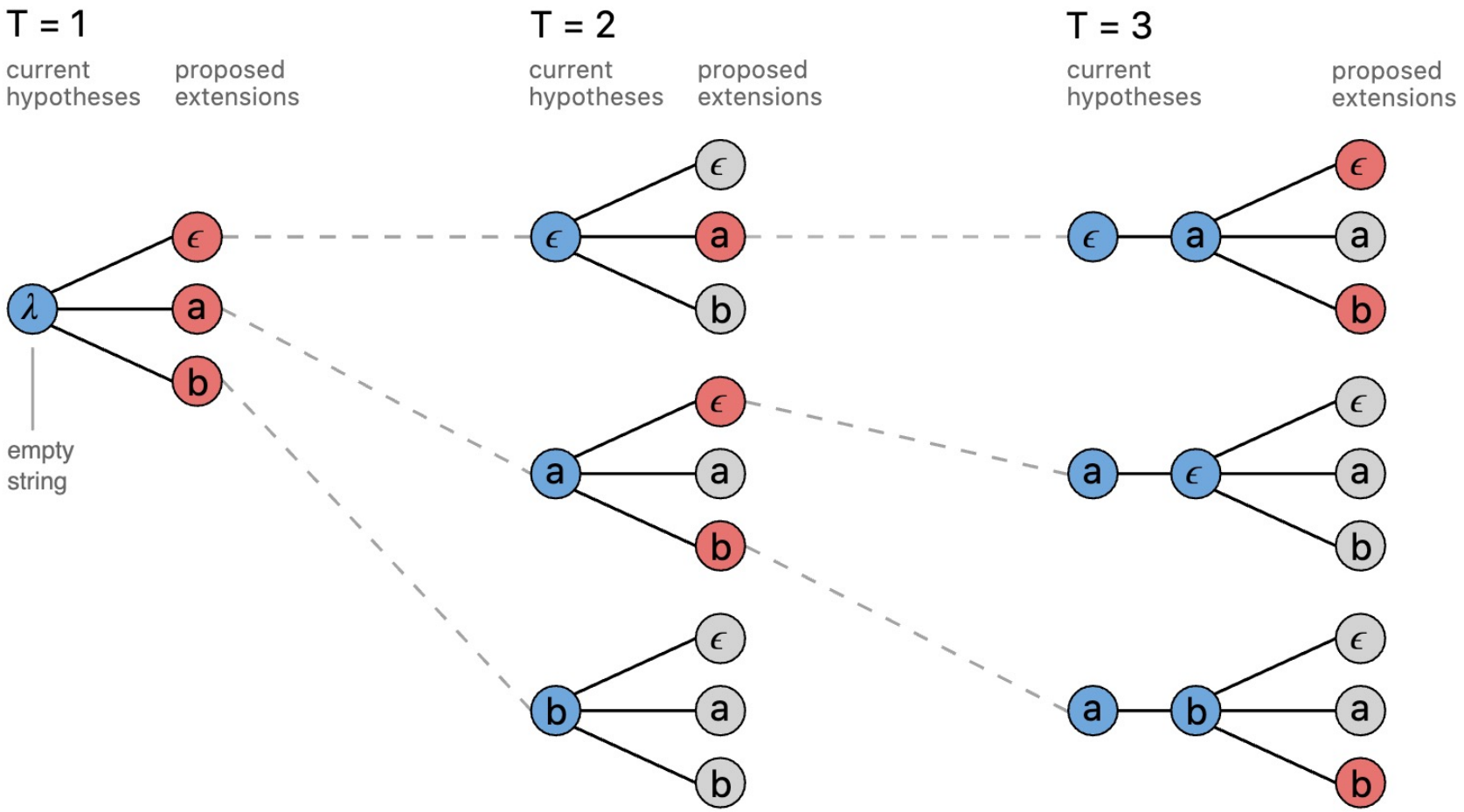
Character Probabilities __oo_h__y_e_aa_h





(Hannun, Maas, Jurafsky, & Ng. 2014)

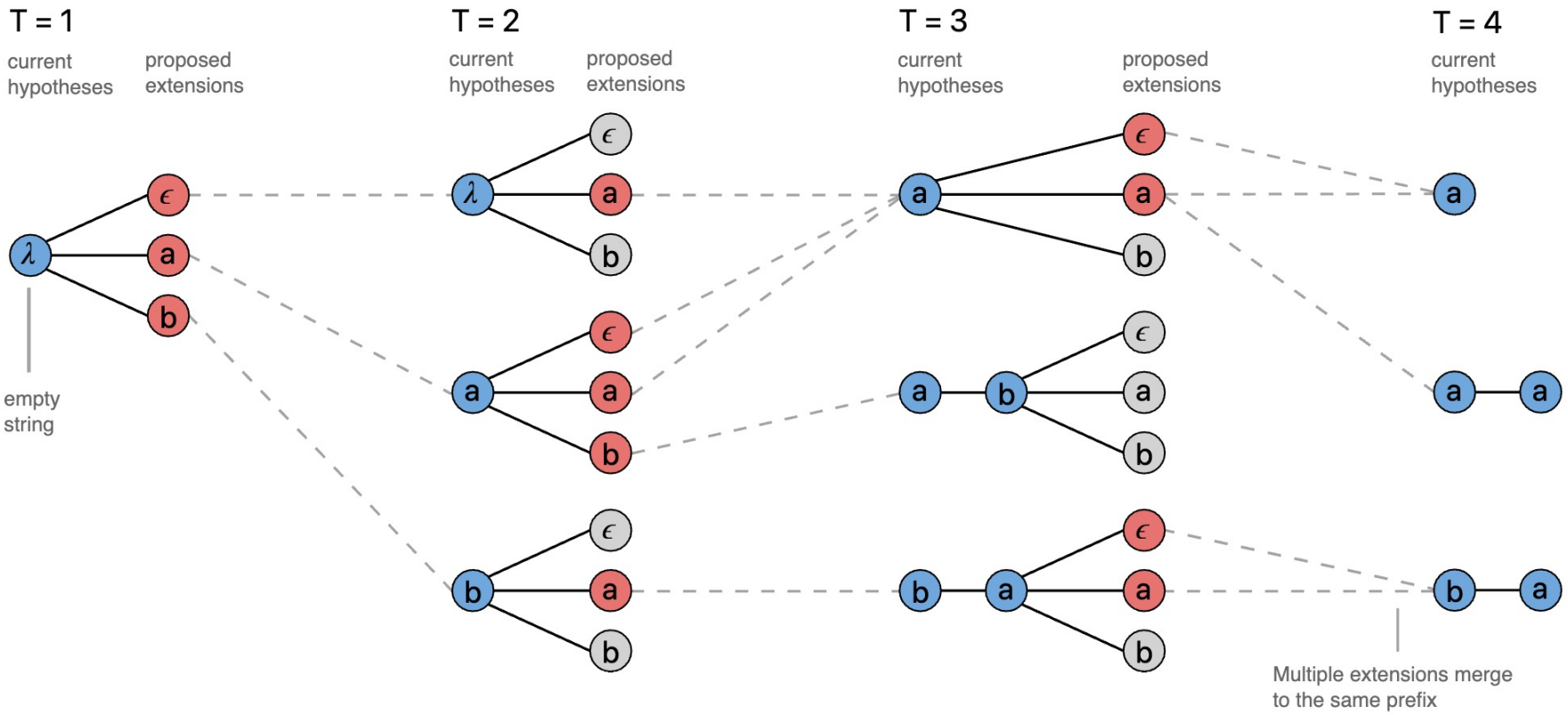
CTC Inference: Simple Beam Search



A standard beam search algorithm with an alphabet of $\{\epsilon, a, b\}$ and a beam size of three.

(Hannun. 2017)

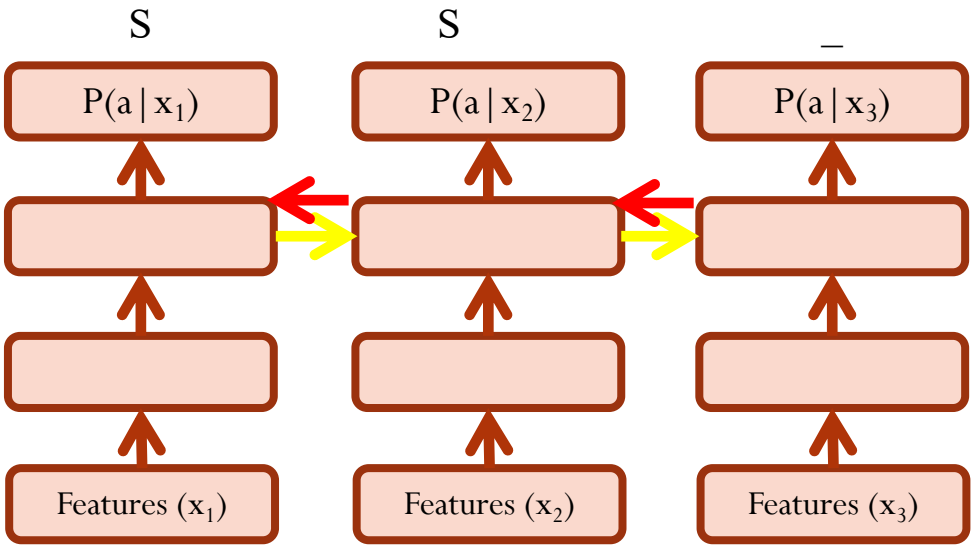
CTC Inference:
Beam Search over Collapsed Outputs

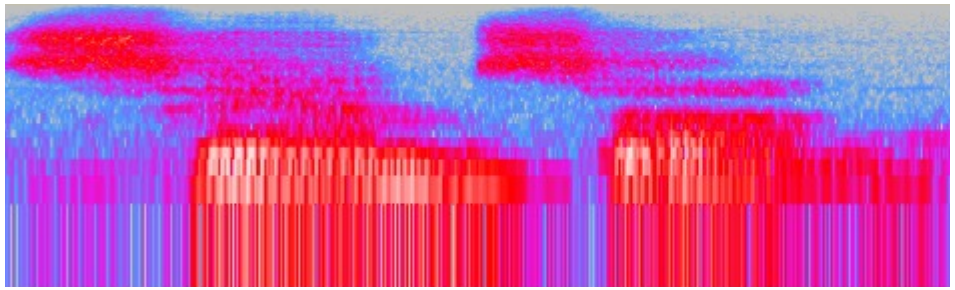


The CTC beam search algorithm with an output alphabet $\{\epsilon, a, b\}$ and a beam size of three.

(Hannun. 2017)

Recurrence Matters!





Architecture	CER
DNN	22

(Hannun, Maas, Jurafsky, & Ng. 2014)

Earlier work on CTC with phonemes

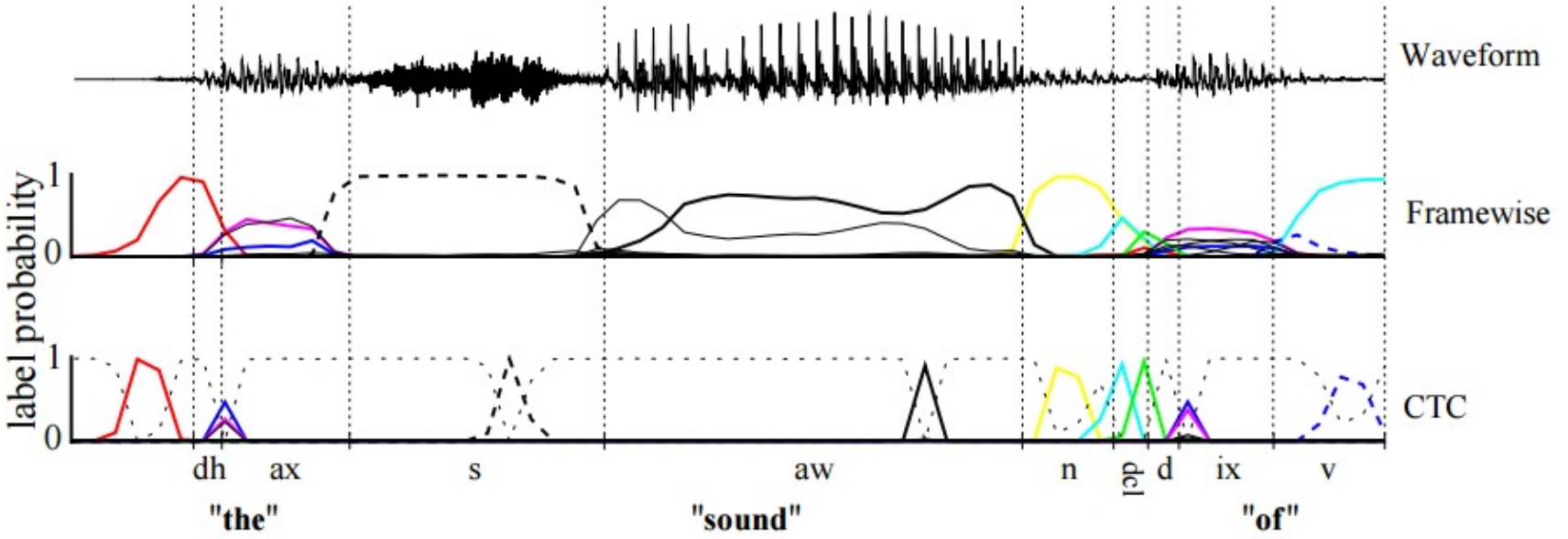


Table 1. **Label Error Rate (LER) on TIMIT**. CTC and hybrid results are means over 5 runs, ± standard error. All differences were significant ($p < 0.01$), except between weighted error BLSTM/HMM and CTC (best path).

System	LER
Context-independent HMM	38.85 %
Context-dependent HMM	35.21 %
BLSTM/HMM	33.84 ± 0.06 %
Weighted error BLSTM/HMM	31.57 ± 0.06 %
CTC (best path)	31.47 ± 0.21 %
CTC (prefix search)	30.51 ± 0.19 %

(Graves, Fernández, Gomez, & Schmidhuber. 2006)

CTC loss during training

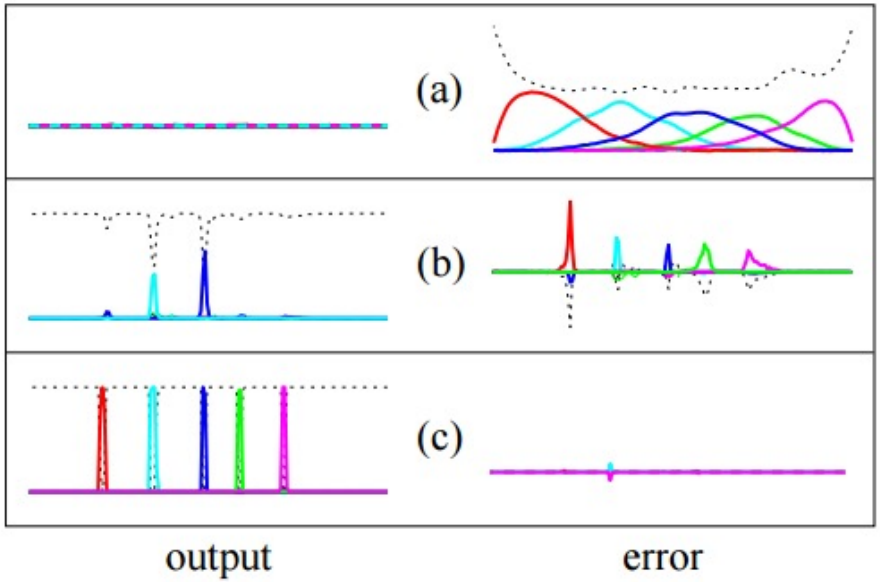


Figure 4. **Evolution of the CTC Error Signal During Training**. The left column shows the output activations for the same sequence at various stages of training (the dashed line is the 'blank' unit); the right column shows the corresponding error signals. Errors above the horizontal axis act to increase the corresponding output activation and those below act to decrease it. (a) Initially the network has small random weights, and the error is determined by the target sequence only. (b) The network begins to make predictions and the error localises around them. (c) The network strongly predicts the correct labelling and the error virtually disappears.

(Graves, Fernández, Gomez, & Schmidhuber. 2006)

Outline

- Iterative training of HMM systems
- Connectionist temporal classification (CTC)
  - **Lexicon-free CTC**
- Listen, Attend, & Spell (LAS)
  - Combining CTC and LAS
- Convolutional transformer

Rethinking Decoding

Out of Vocabulary Words

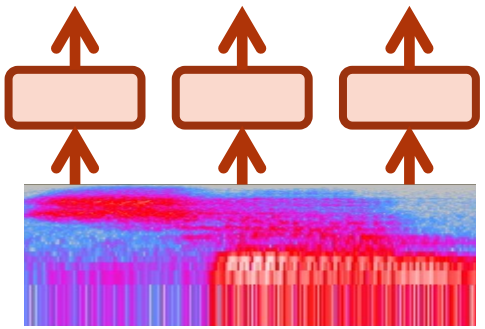
syriza rona abo-- fru-- schmidhuber

Lex [zebra]

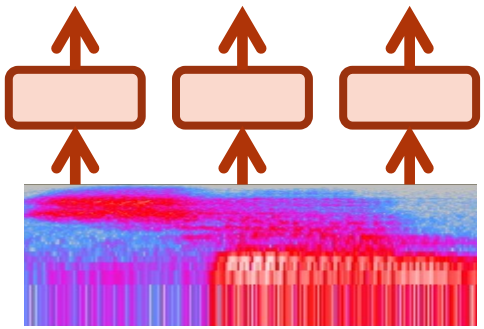
L uage (“y | “oh”)
M

***Character* Language Model** p(h | o,h, ,y,e,a,)

Character Probabilities __oo_h__y_e_aa_h



Character Probabilities __oo_h__y_e_aa_h

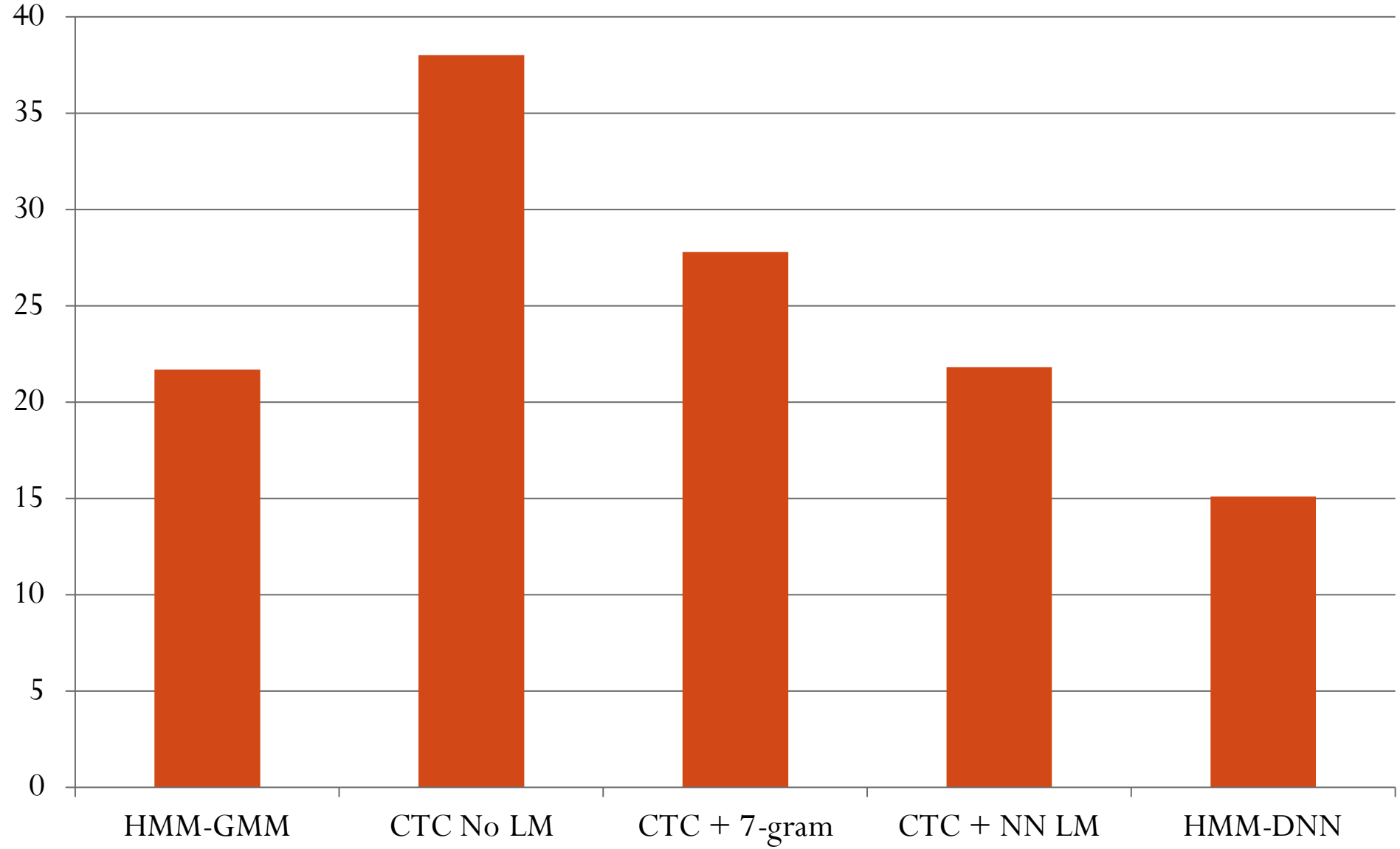


(Maas*, Xie*, Jurafsky, & Ng. 2015)

Beam Search Decoding

Inputs CTC likelihoods $p_{\text{ctc}}(c|x_t)$, character language model $p_{\text{clm}}(c|s)$
Parameters language model weight α, insertion bonus β, beam width k
Initialize $Z_0 \leftarrow \{\varnothing\}$, $p_{\text{b}}(\varnothing|x_{1:0}) \leftarrow 1$, $p_{\text{nb}}(\varnothing|x_{1:0}) \leftarrow 0$
for $t = 1, \ldots, T$ **do**
    $Z_t \leftarrow \{\}$
    for s **in** Z_{t-1} **do**
        $p_{\text{b}}(s|x_{1:t}) \leftarrow p_{\text{ctc}}(\_|x_t)p_{\text{tot}}(s|x_{1:t-1})$ ▷ Handle blanks
        $p_{\text{nb}}(s|x_{1:t}) \leftarrow p_{\text{ctc}}(c|x_t)p_{\text{nb}}(s|x_{1:t-1})$ ▷ Handle repeat character collapsing
        Add s to Z_t
        for c **in** ζ' **do**
            $s^+ \leftarrow s + c$
            if $c \neq s_{t-1}$ **then**
                $p_{\text{nb}}(s^+|x_{1:t}) \leftarrow p_{\text{ctc}}(c|x_t)p_{\text{clm}}(c|s)^\alpha p_{\text{tot}}(c|x_{1:t-1})$
            else
                $p_{\text{nb}}(s^+|x_{1:t}) \leftarrow p_{\text{ctc}}(c|x_t)p_{\text{clm}}(c|s)^\alpha p_{\text{b}}(c|x_{1:t-1})$ ▷ Repeat characters have "_" between
            end if
            Add s^+ to Z_t
        end for
    end for
    $Z_t \leftarrow k$ most probable s by $p_{\text{tot}}(s|x_{1:t})|s|^\beta$ in Z_t ▷ Apply beam
end for
Return $\arg\max_{s \in Z_t} p_{\text{tot}}(s|x_{1:T})|s|^\beta$

Lexicon-Free & HMM-Free on Switchboard



(Maas*, Xie*, Jurafsky, & Ng. 2015)

Example Results (Switchboard) ~19% CER

i i don'tknow i don't know what the rain force have to do with it but you know their chop a those down af the tr minusrat everyday

i- i don't kn- i don't know what the rain forests have to do with it but you know they're chopping those down at a tremendous rate everyday

come home and get back in to regular cloos aga

come home and get back into regular clothes again

i guess down't here u we just recently move to texas so my wor op has change quite a bit muh we ook from colorado were and i have a cloveful of sweatterso tuth

i guess down here uh we just recently moved to texas so my wardrobe has changed quite a bit um we moved from colorado where and i have a closet full of sweaters that

i don't know whether state lit state hood whold itprove there a conomy i don't i don't know that to that the actove being a state

i don't know whether state woul- statehood would improve their economy i don't i don't know that the ve- the act of being a state

(Maas*, Xie*, Jurafsky, & Ng. 2015)

Transcribing Out of Vocabulary Words

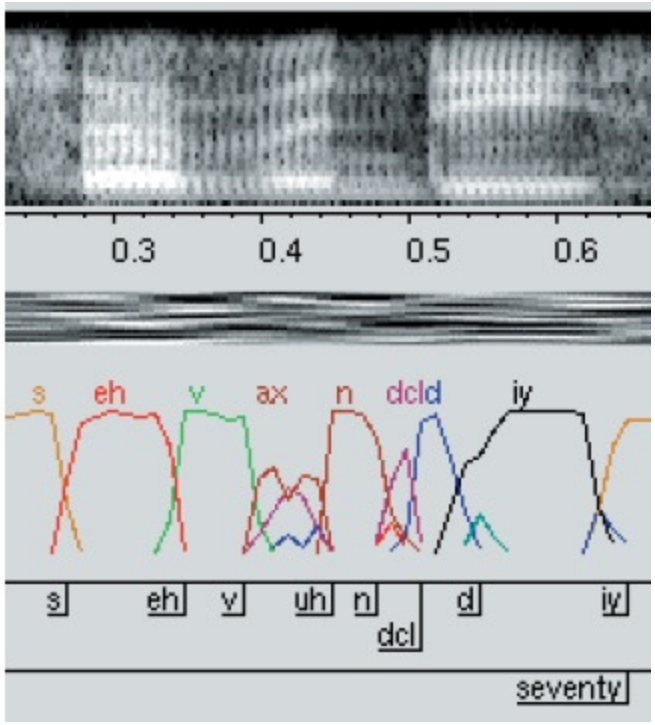
Truth: yeah i went into the i do not know what you think of *fidelity* but
HMM-GMM: yeah when the i don't know what you think of **fidel it even them**
CTC-CLM: yeah i went to i don't know what you think of **fidelity but um**

Truth: no no speaking of weather do you carry a altimeter slash *barometer*
HMM-GMM: no i'm not all being the weather do you uh carry a **uh helped emitters last brahms her**
CTC-CLM: no no beating of whether do you uh carry a **uh a time or less barometer**

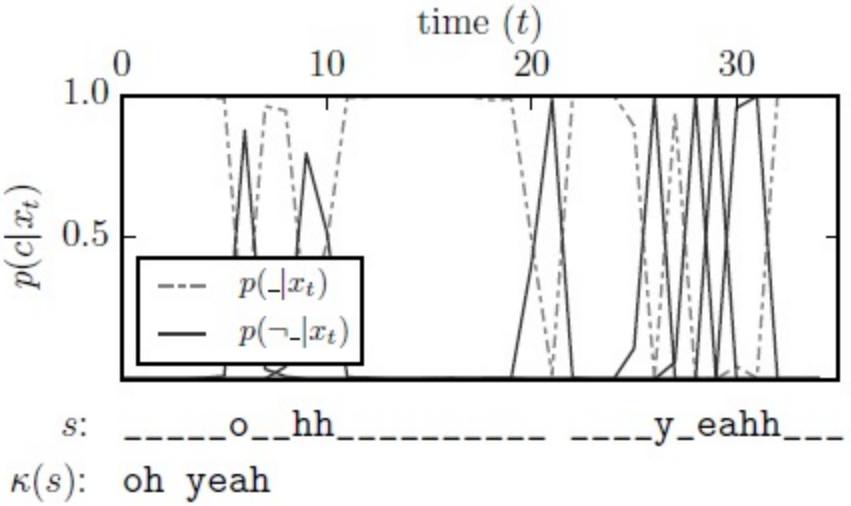
Truth: i would ima- well yeah it is i know you are able to stay home with them
HMM-GMM: i would **amount** well yeah it is i know um you're able to stay home with them
CTC-CLM: i would **ima-** well yeah it is i know uh you're able to stay home with them

(Maas*, Xie*, Jurafsky, & Ng. 2015)

Comparing Alignments



HMM-GMM phone probabilities

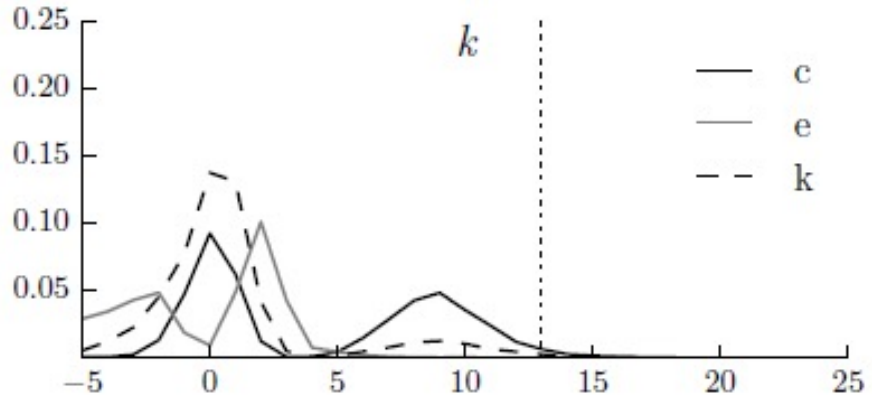


CTC character probabilities

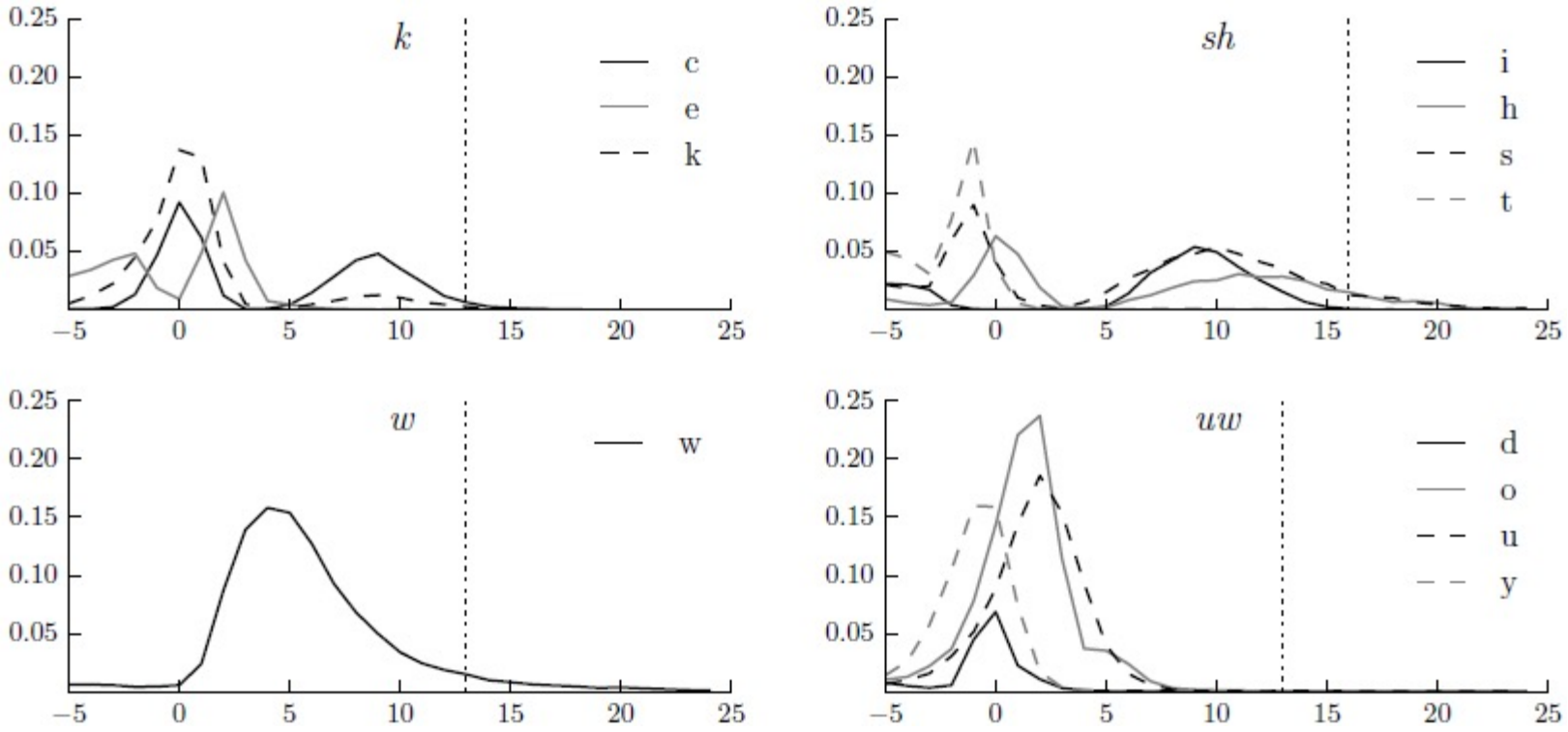
(HMM slide from Dan Ellis)

Learning Phonemes and Timing

- Take all phone segments from HMM-GMM alignments (*k*)
- Align all segments to start at the same time = 0
- Compute the average CTC *character* probabilities during the segment (c, e, k)
- Vertical line shows median end time of phone segment from HMM-GMM alignments



Learning Phonemes and Timing



(Maas*, Xie*, Jurafsky, & Ng. 2015)

Outline

- Iterative training of HMM systems
- Connectionist temporal classification (CTC)
  - Lexicon-free CTC
- **Listen, Attend, & Spell (LAS)**
  - Combining CTC and LAS
- Convolutional transformer

Listen, Attend, and Spell

- Discriminative, character-based encoder-decoder
- Unlike CTC:
  - outputs also condition on previous outputs so far
  - No blank/epsilon. LAS just outputs characters
- Attention-based decoder. Precursor to modern encoder-decoder and transformer approaches

$$P(\mathbf{y}|\mathbf{x}) = \prod_i P(y_i|\mathbf{x}, y_{<i})$$

$$\mathbf{h} = \text{Listen}(\mathbf{x})$$
$$P(\mathbf{y}|\mathbf{x}) = \text{AttendAndSpell}(\mathbf{h}, \mathbf{y})$$

(Chan, Jaitly, Le, & Vinyals. 2015)

Listen, Attend, and Spell

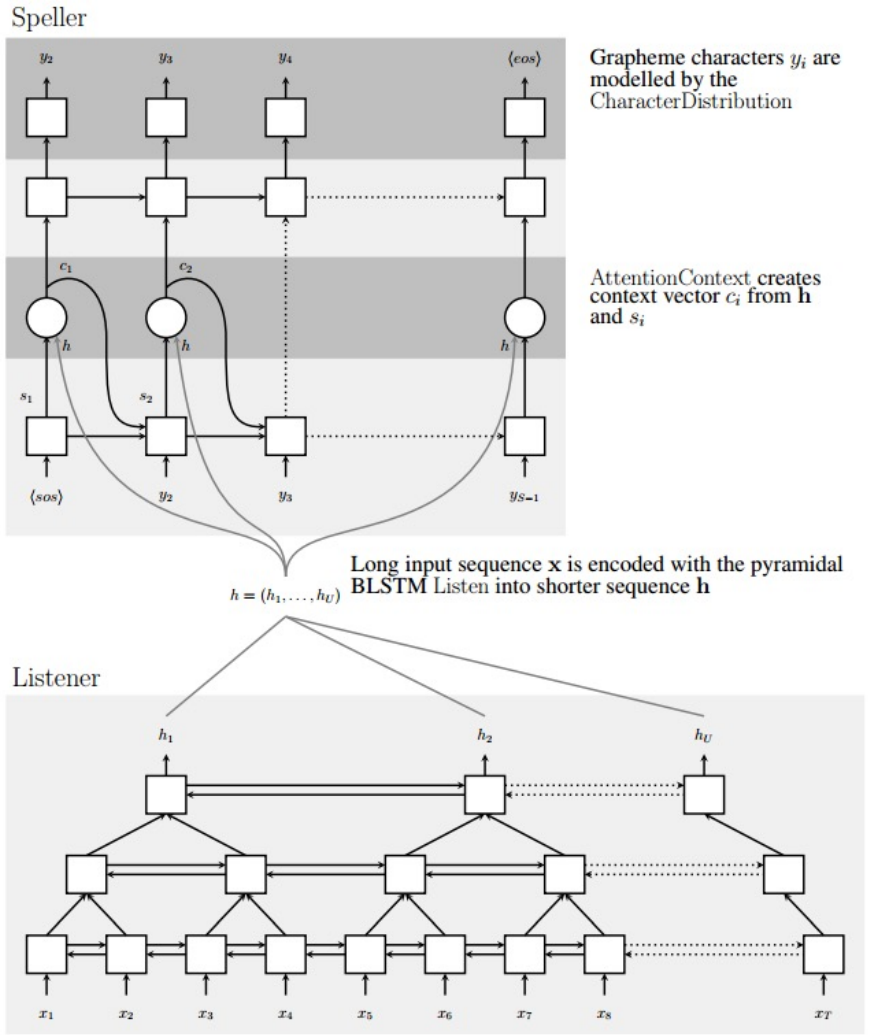
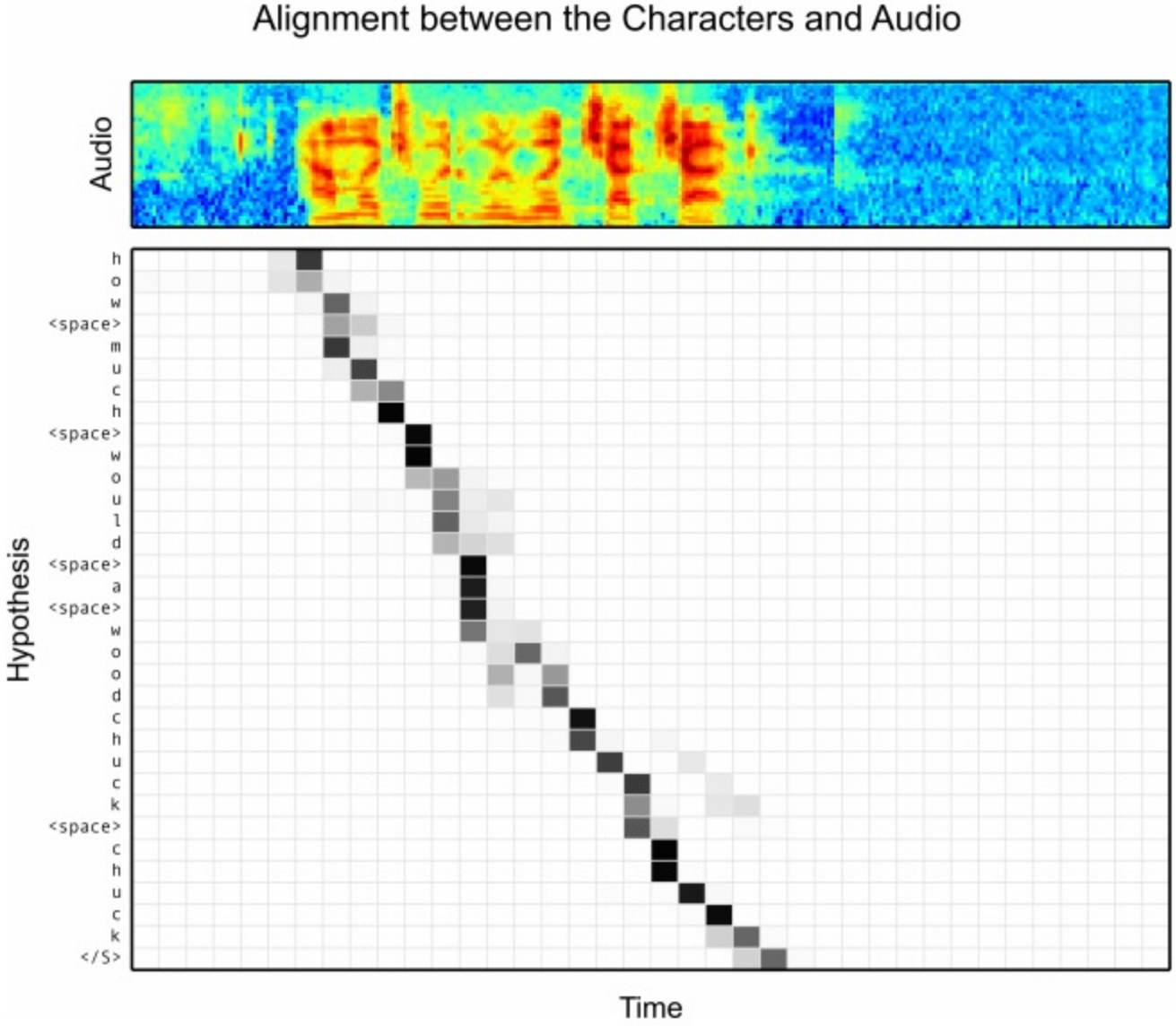


Figure 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence $\mathbf{x}$ into high level features $\mathbf{h}$, the speller is an attention-based decoder generating the $\mathbf{y}$ characters from $\mathbf{h}$.

(Chan, Jaitly, Le, & Vinyals. 2015)

Listen, Attend, and Spell



(Chan, Jaitly, Le, & Vinyals. 2015)

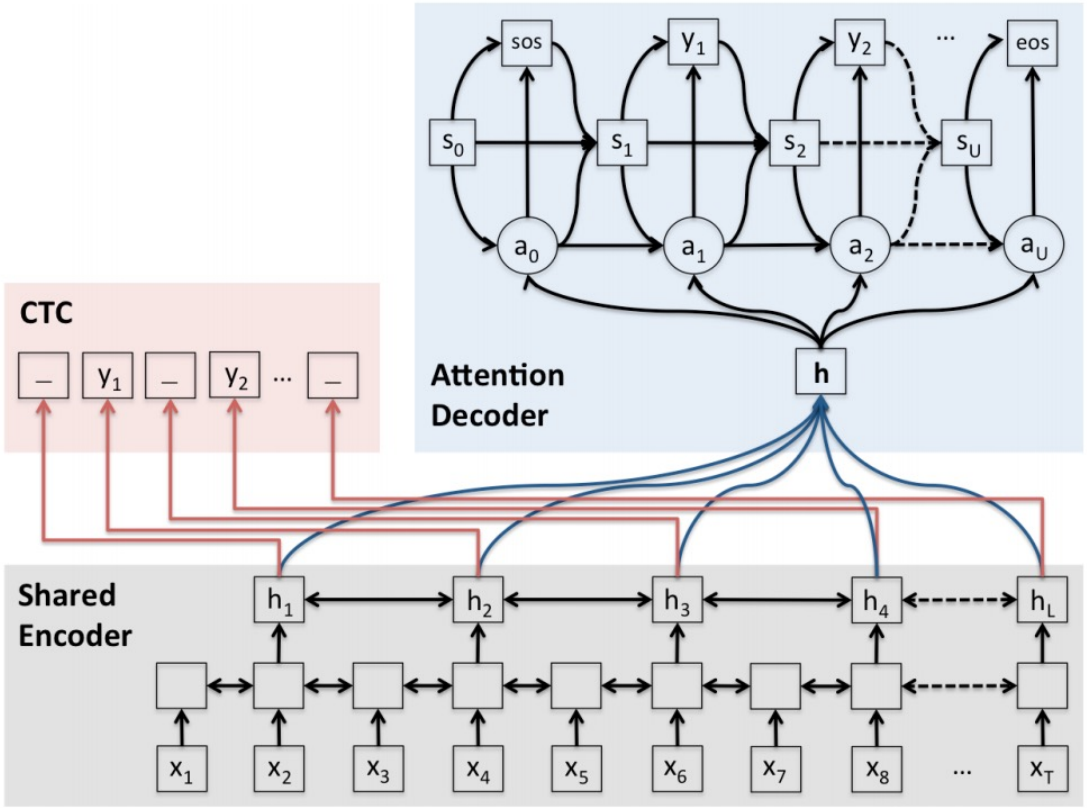
Listen, Attend, and Spell

Table 1: WER comparison on the clean and noisy Google voice search task. The CLDNN-HMM system is the state-of-the-art system, the Listen, Attend and Spell (LAS) models are decoded with a beam size of 32. Language Model (LM) rescoring was applied to our beams, and a sampling trick was applied to bridge the gap between training and inference.

Model	**Clean WER**	**Noisy WER**
CLDNN-HMM [20]	8.0	8.9
LAS	16.2	19.0
LAS + LM Rescoring	12.6	14.7
LAS + Sampling	14.1	16.5
LAS + Sampling + LM Rescoring	10.3	12.0

(Chan, Jaitly, Le, & Vinyals. 2015)

CTC + LAS Multi-Task Approach



$$\mathcal{L}_{\text{MTL}} = \lambda \mathcal{L}_{\text{CTC}} + (1-\lambda)\mathcal{L}_{\text{Attention}}$$

Fig. 1: Our proposed Joint CTC-attention based end-to-end framework: the shared encoder is trained by both CTC and attention model objectives simultaneously. The shared encoder transforms our input sequence $\boldsymbol{x}$ into high level features $\boldsymbol{h}$, the location-based attention decoder generates the character sequence $\boldsymbol{y}$.

(Kim, Hori, & Watanabe. 2017)

CTC + LAS Multi-Task Approach

Table 1: Character Error Rate (CER) on clean corpora WSJ1 (80hours) and WSJ0 (15hours), and a noisy corpus CHiME-4 (18hours). None of our experiments used any language model or lexicon information. (Word Error Rate (WER) of our model MTL($\lambda = 0.2$) was 18.2% and WER of [7] was 18.6% on WSJ1. Note that this is not an exact comparison because the hyper parameters were not completely same as [7].)

Model(train)	CER(valid)	CER(eval)
WSJ-train_si284 (80hrs)	dev93	eval92
CTC	11.48	8.97
Attention(content-based)	13.68	11.08
Attention(location-based)	11.98	8.17
MTL($\lambda = 0.2$)	**11.27**	**7.36**
MTL($\lambda = 0.5$)	12.00	8.31
MTL($\lambda = 0.8$)	11.71	8.45
WSJ-train_si84 (15hrs)	dev93	eval92
CTC	27.41	20.34
Attention(content-based)	28.02	20.06
Attention(location-based)	24.98	17.01
MTL($\lambda = 0.2$)	**23.03**	**14.53**
MTL($\lambda = 0.5$)	26.28	16.24
MTL($\lambda = 0.8$)	32.21	21.30
CHiME-4-tr05_multi (18hrs)	dt05_real	et05_real
CTC	37.56	48.79
Attention(content-based)	43.45	54.25
Attention(location-based)	35.01	47.58
MTL($\lambda = 0.2$)	**32.08**	**44.99**
MTL($\lambda = 0.5$)	34.56	46.49
MTL($\lambda = 0.8$)	35.41	48.34

$$e_{u,l} = \begin{cases} \text{content-based:} \\ \quad w^T \tanh(W s_{u-1} + V h_l + b) \\ \text{location-based:} \\ \quad f_u = F * a_{u-1} \\ \quad w^T \tanh(W s_{u-1} + V h_l + U f_{u,l} + b) \end{cases}$$

$$a_{u,l} = \frac{\exp(\gamma e_{u,l})}{\sum_l \exp(\gamma e_{u,l})}$$

$$\mathcal{L}_{\text{MTL}} = \lambda \mathcal{L}_{\text{CTC}} + (1 - \lambda) \mathcal{L}_{\text{Attention}}$$

(Kim, Hori, & Watanabe. 2017)

CTC + LAS Multi-Task Approach

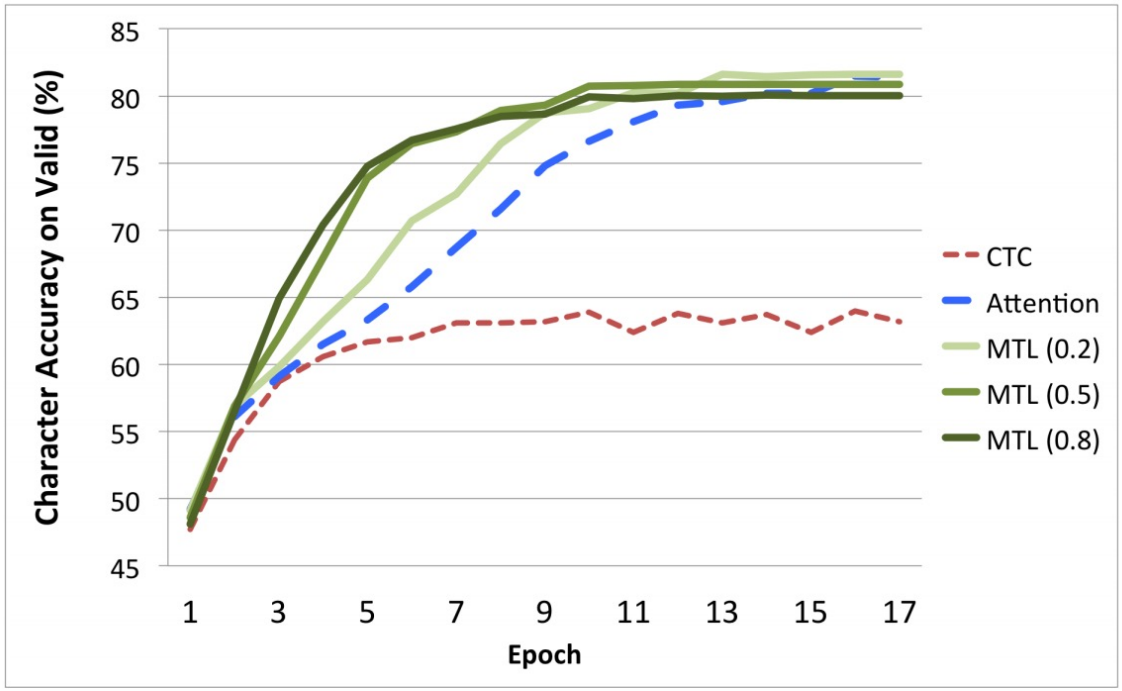
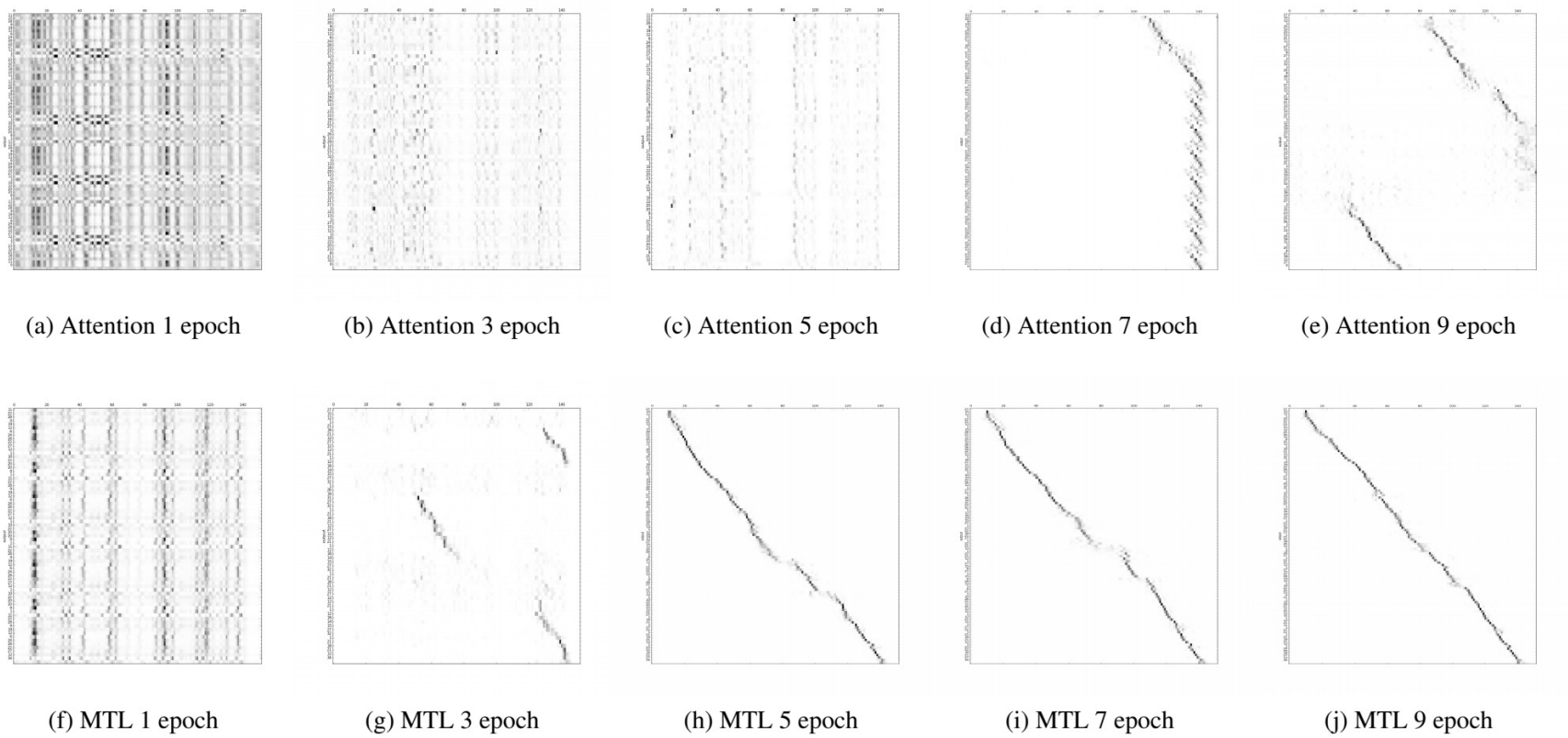


Fig. 2: Comparison of learning curves: CTC, location-based attention model, and MTL with ($\lambda = 0.2, 0.5, 0.8$). The character accuracy on the validation set of CHiME-4 is calculated by edit distance between hypothesis and reference. Note that the reference history were used in the attention and our MTL models.

(Kim, Hori, & Watanabe. 2017)

CTC + LAS Multi-Task Approach



(a) Attention 1 epoch (b) Attention 3 epoch (c) Attention 5 epoch (d) Attention 7 epoch (e) Attention 9 epoch

(f) MTL 1 epoch (g) MTL 3 epoch (h) MTL 5 epoch (i) MTL 7 epoch (j) MTL 9 epoch

Fig. 3: Comparison of speed in learning alignments between characters (y-axis) and acoustic frames (x-axis) between the location-based attention model (1st row) and our model `MTL` (2nd row) over training epoch (1,3,5,7, and 9). All alignments are for one manually chosen utterance (F05_442C020U_CAF_REAL - "THE ONE HUNDRED SHARE INDEX CLOSED SIX POINT EIGHT POINTS LOWER AT ONE THOUSAND SEVEN HUNDRED FIFTY NINE POINT NINE") in the noisy CHiME-4 evaluation set.

(Kim, Hori, & Watanabe. 2017)

CTC + LAS Multi-Task Approach

Table 1: Character error rate (CER) for conventional attention and hybrid CTC/attention end-to-end ASR. Corpus of Spontaneous Japanese speech recognition (CSJ) task.

Model	Hour	Task1	Task2	Task3
Attention	581	11.4	7.9	9.0
MTL	581	10.5	7.6	8.3
MTL + joint decoding (rescoring)	581	10.1	7.1	7.8
MTL + joint decoding (one pass)	581	10.0	7.1	7.6
MTL-large + joint decoding (rescoring)	581	**8.4**	6.2	**6.9**
MTL-large + joint decoding (one pass)	581	**8.4**	**6.1**	**6.9**
GMM-discr. (Moriya et al., 2015)	236 for AM, 581 for LM	11.2	9.2	12.1
DNN/HMM (Moriya et al., 2015)	236 for AM, 581 for LM	9.0	7.2	9.6
CTC-syllable (Kanda et al., 2016)	581	9.4	7.3	7.5

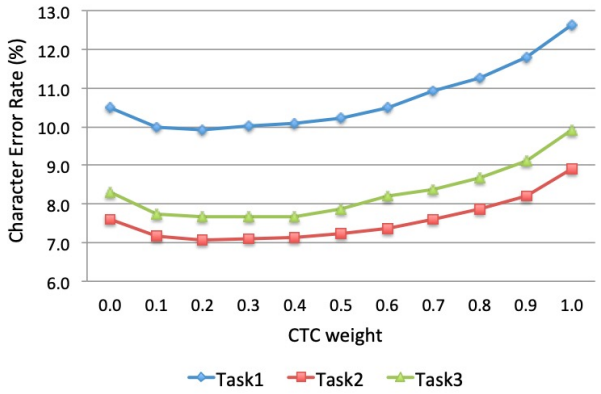


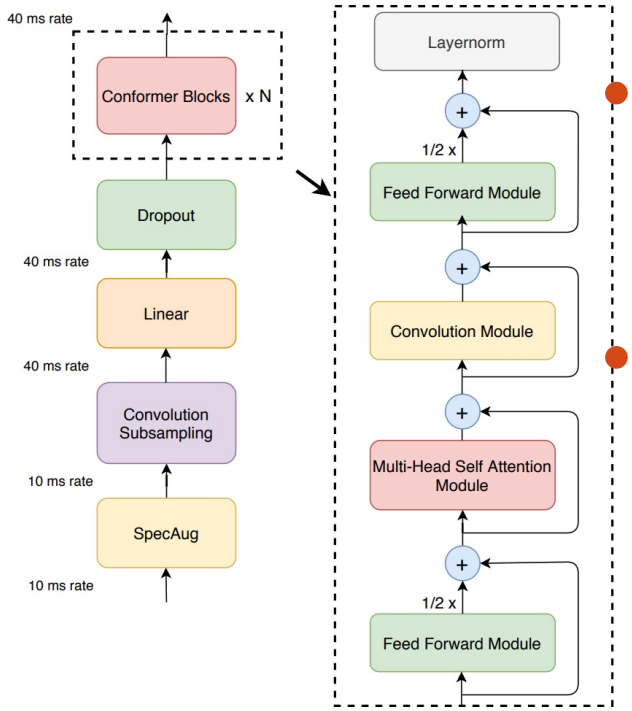
Figure 2: The effect of weight parameter λ in Eq. (14) on the CSJ evaluation tasks (The CERs were obtained by one-pass decoding).

(Hori, Watanabe, Zhang, & Chan. 2017)

Outline

- Iterative training of HMM systems
- Connectionist temporal classification (CTC)
  - Lexicon-free CTC
- Listen, Attend, & Spell (LAS)
  - Combining CTC and LAS
- **Convolutional transformer**

Conformer: Convolution-augmented Transformer for Speech Recognition



- Sequence-to-sequence transformer with multi-headed self attention.
- Transformer encoder combines attention (global context) with convolution (local invariance)
- Transducer loss directly optimizes for output sequence

Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

(Gulati *et al.* 2020)

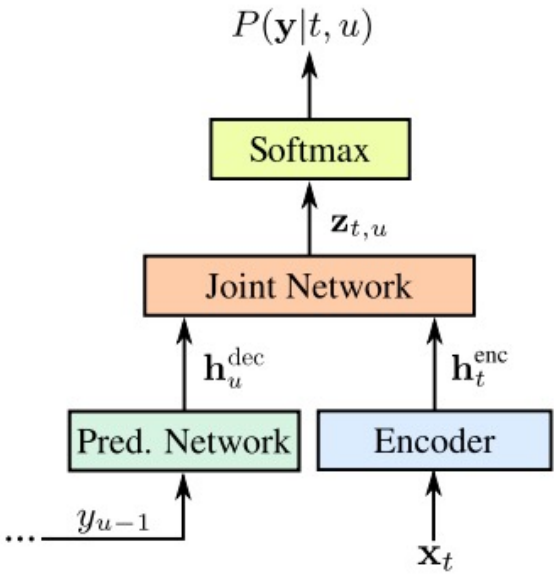
Conformer: RNN-Transducer loss

- Directly optimizes target word sequence as correct label
  - Graphemes (letters) or word parts (10k-50k) used in practice
- Learned combination of acoustic + language model pieces
- Conditions on sequence output so far (y_{t-1})
- Single alignment:

$$P(\mathbf{z}|\mathbf{x}) = \prod_i P(z_i|\mathbf{x}, t_i, \text{Labels}(z_{1:(i-1)}))$$

- Maximize P(y|x) by summing over all consistent alignments (like CTC encoder can output *blank*):

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{z} \in \mathcal{Z}(\mathbf{y}, T)} P(\mathbf{z}|\mathbf{x})$$



RNN-T loss: (Graves, 2012) Figure: Rao, Sak, & Prabhavalkar. 2017)

Conformer: Convolutional transformer encoder

- Sequence-to-sequence transformer with multi-headed self attention. Directly optimizes target word sequence
- Combines attention (global context) with convolution (local invariance)

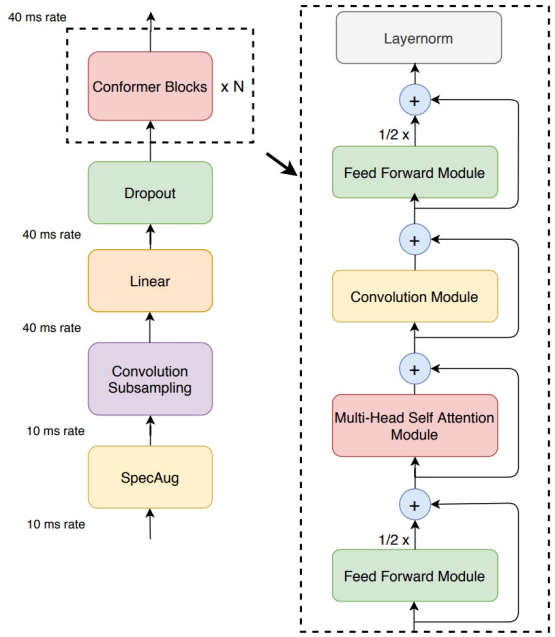


Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

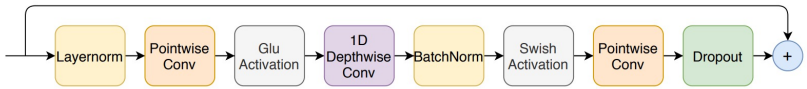


Figure 2: ***Convolution module.*** *The convolution module contains a pointwise convolution with an expansion factor of 2 projecting the number of channels with a GLU activation layer, followed by a 1-D Depthwise convolution. The 1-D depthwise conv is followed by a Batchnorm and then a swish activation layer.*

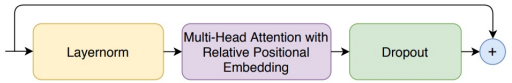


Figure 3: ***Multi-Headed self-attention module.*** *We use multi-headed self-attention with relative positional embedding in a pre-norm residual unit.*

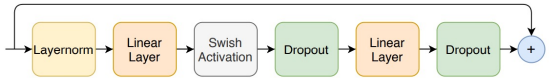


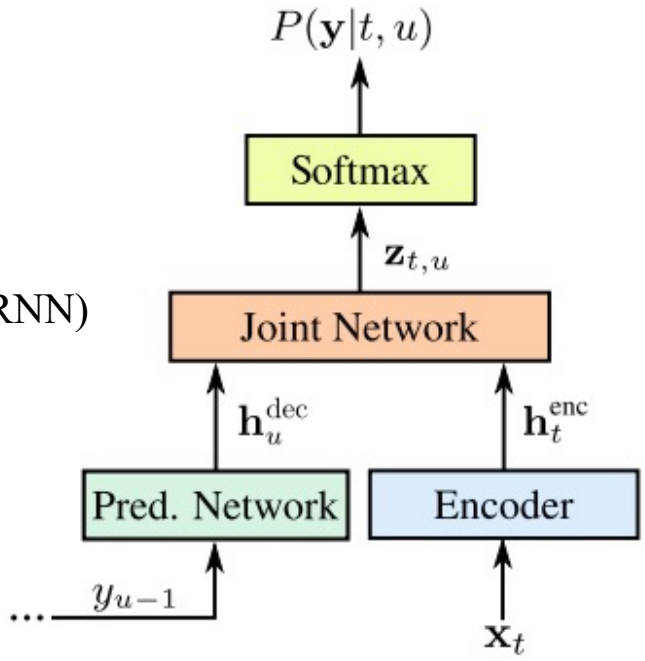
Figure 4: ***Feed forward module.*** *The first linear layer uses an expansion factor of 4 and the second linear layer projects it back to the model dimension. We use swish activation and a pre-norm residual units in feed forward module.*

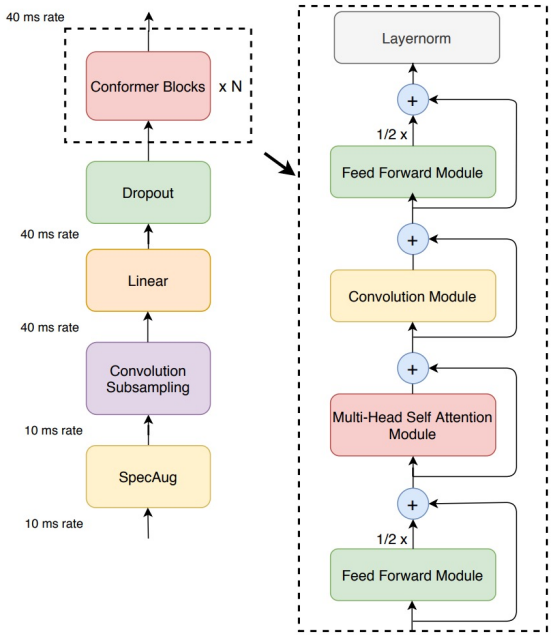
(Gulati *et al.* 2020)

Conformer: Putting it all together

1 hidden layer LSTM (RNN)

3 hidden layer LSTM (RNN). Pre-trained as language model





(Gulati *et al.* 2020)

Conformer: Convolution-augmented Transformer for Speech Recognition

Table 2: *Comparison of Conformer with recent published models. Our model shows improvements consistently over various model parameter size constraints. At 10.3M parameters, our model is 0.7% better on testother when compared to contemporary work, ContextNet(S) [10]. At 30.7M model parameters our model already significantly outperforms the previous published state of the art results of Transformer Transducer [7] with 139M parameters.*

Method	#Params (M)	WER Without LM		WER With LM	
		testclean	testother	testclean	testother
Hybrid					
Transformer [33]	-	-	-	2.26	4.85
CTC					
QuartzNet [9]	19	3.90	11.28	2.69	7.25
LAS					
Transformer [34]	270	2.89	6.98	2.33	5.17
Transformer [19]	-	2.2	5.6	2.6	5.7
LSTM	360	2.6	6.0	2.2	5.2
Transducer					
Transformer [7]	139	2.4	5.6	2.0	4.6
ContextNet(S) [10]	10.8	2.9	7.0	2.3	5.5
ContextNet(M) [10]	31.4	2.4	5.4	**2.0**	4.5
ContextNet(L) [10]	112.7	**2.1**	4.6	**1.9**	4.1
Conformer (Ours)					
Conformer(S)	10.3	**2.7**	**6.3**	**2.1**	**5.0**
Conformer(M)	30.7	**2.3**	**5.0**	**2.0**	**4.3**
Conformer(L)	118.8	**2.1**	**4.3**	**1.9**	**3.9**

(Gulati *et al.* 2020)

Dual Mode ASR: Joint encoder + training for streaming & full context models

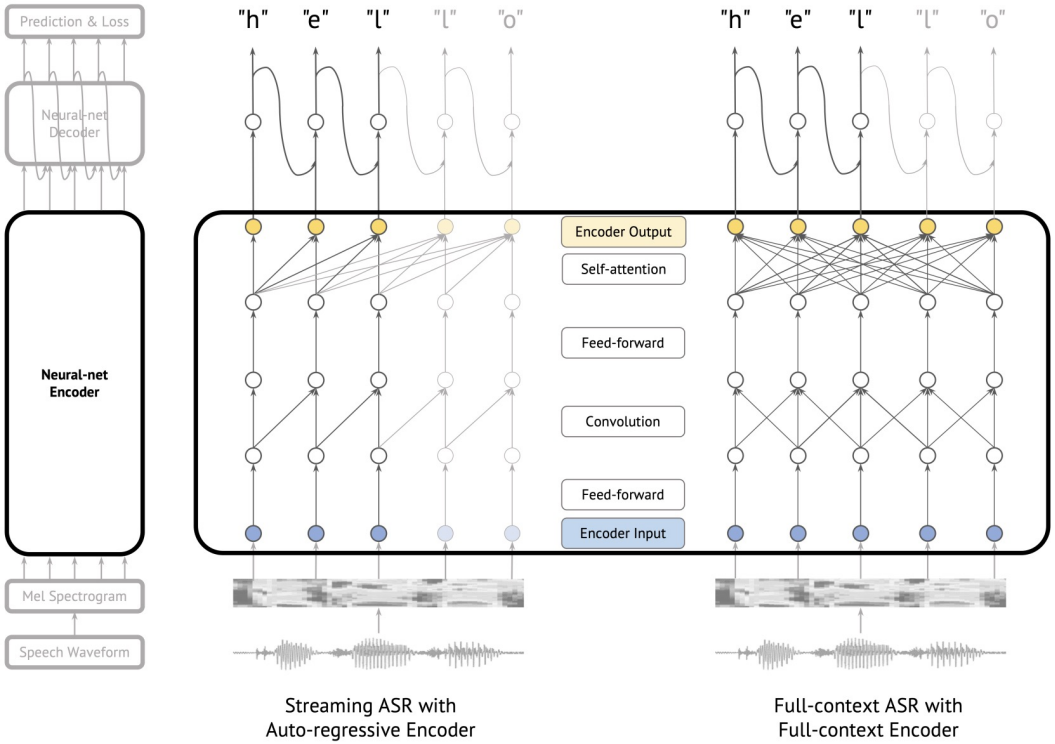


Figure 1: A simplified illustration of the similarity and difference between Streaming ASR and Full-context ASR networks. Modern end-to-end streaming and full-context ASR models share most of the neural architectures and training recipes in common, with the most significant difference in the **ASR encoder (highlighted)**. Streaming ASR encoders are auto-regressive models, with each prediction of the current timestep conditioned on previous ones (no future context). We show examples of feed-forward layer, convolution layer and self-attention layer in the encoder of streaming and full-context ASR respectively. With Dual-mode ASR, we unify them without parameters overhead.

(Yu *et al.* 2021)

Dual Mode ASR: Joint encoder + training for streaming & full context models

Table 3: Summary of our results on Librispeech dataset (Panayotov et al., 2015). We report WER on TestClean and TestOther (noisy) set. Compared with standalone ContextNet and Conformer models, Dual-mode ASR models have both higher accuracy in average and better streaming latency.

Method	**Mode**	**# Params (M)**	**Test Clean/Other** WER(%)	**Latency@50** (ms)	**Latency@90** (ms)
LSTM-LAS	Full-context	360	2.6 / 6.0	——	——
QuartzNet-CTC	Full-context	19	3.9 / 11.3	——	——
Transformer	Full-context	29	3.1 / 7.3	——	——
Transformer	Full-context	139	2.4 / 5.6	——	——
ContextNet	Full-context	31.4	2.4 / 5.4	——	——
Conformer	Full-context	30.7	2.3 / 5.0	——	——
Transformer	Streaming	18.9	5.0 / 11.6	80	190
ContextNet	Streaming	31.4	4.5 / 10.0	70	270
Conformer	Streaming	30.7	4.6 / 9.9	140	280
ContextNet Look-ahead	Streaming	31.4	4.1 / 9.0	150	420
Dual-mode Transformer	Full-context	29	3.1 / 7.9	——	——
	Streaming		4.4 (-0.6) / 11.5 (-0.1)	-50 (-130)	30 (-160)
Dual-mode ContextNet	Full-context	31.8	2.3 / 5.3	——	——
	Streaming		3.9 (-0.6) / 8.5 (-1.5)	40 (-30)	160 (-110)
Dual-mode Conformer	Full-context	30.7	2.5 / 5.9	——	——
	Streaming		3.7 (-0.9) / 9.2 (-0.7)	10 (-130)	90 (-190)

Table 4: Ablation studies of weight sharing, joint training and inplace distillation. We report WER on TestOther (noisy) set (Panayotov et al., 2015) using ContextNet with same training settings.

Weight Sharing	**Joint Training**	**Inplace Distillation**	**TestOther** WER(%)	**Latency@50** (ms)	**Latency@90** (ms)
✔	✔	✔	8.5	40	160
✔	✔	✘	10.2 (+1.7)	120 (+80)	310 (+150)
✔	✘	✘	10.6 (+2.1)	90 (+50)	290 (+130)
✘	✔	✔	9.9 (+1.4)	50 (+10)	210 (+50)

(Yu *et al.* 2021)

Appendix