CS 224S / LINGUIST 285
Spoken Language Processing

Andrew Maas
Stanford University
Spring 2017

Lecture 8: End-to-end neural network speech recognition

Outline

- ASR discussion thus far
- Connectionist temporal classification (CTC)
- Lexicon-free CTC
- Scaling up end-to-end neural approaches
- Alternative end-to-end approaches
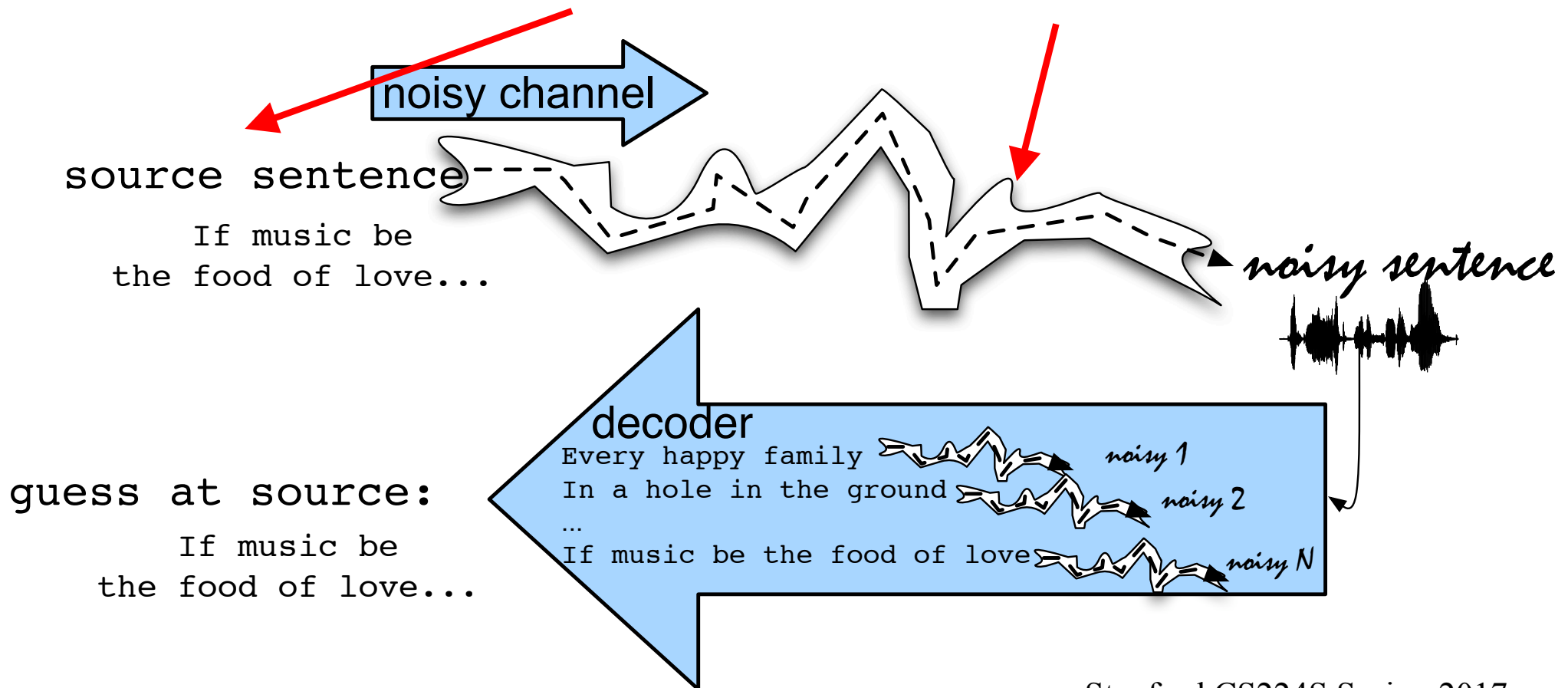- HW3 discussion

Noisy channel model

likelihood   prior

$$\hat{W} = \underset{W \in L}{\arg\max}\, P(O \mid W)\, P(W)$$

The noisy channel model

Ignoring the denominator leaves us with two factors: P(Source) and P(Signal|Source)

noisy channel

source sentence

If music be
the food of love...

noisy sentence

decoder
Every happy family
In a hole in the ground
...
If music be the food of love

noisy 1
noisy 2
noisy N

guess at source:

If music be
the food of love...

Lexicon

one	w ah n
two	t uw
three	th r iy
four	f ao r
five	f ay v
six	s ih k s
seven	s eh v ax n
eight	ey t
nine	n ay n
zero	z iy r ow
oh	ow

Phone HMM
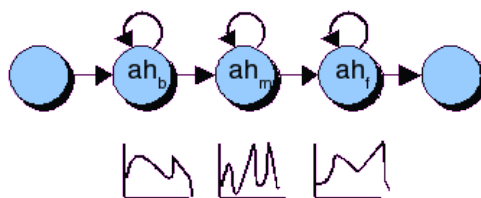
HMM for the digit recognition task
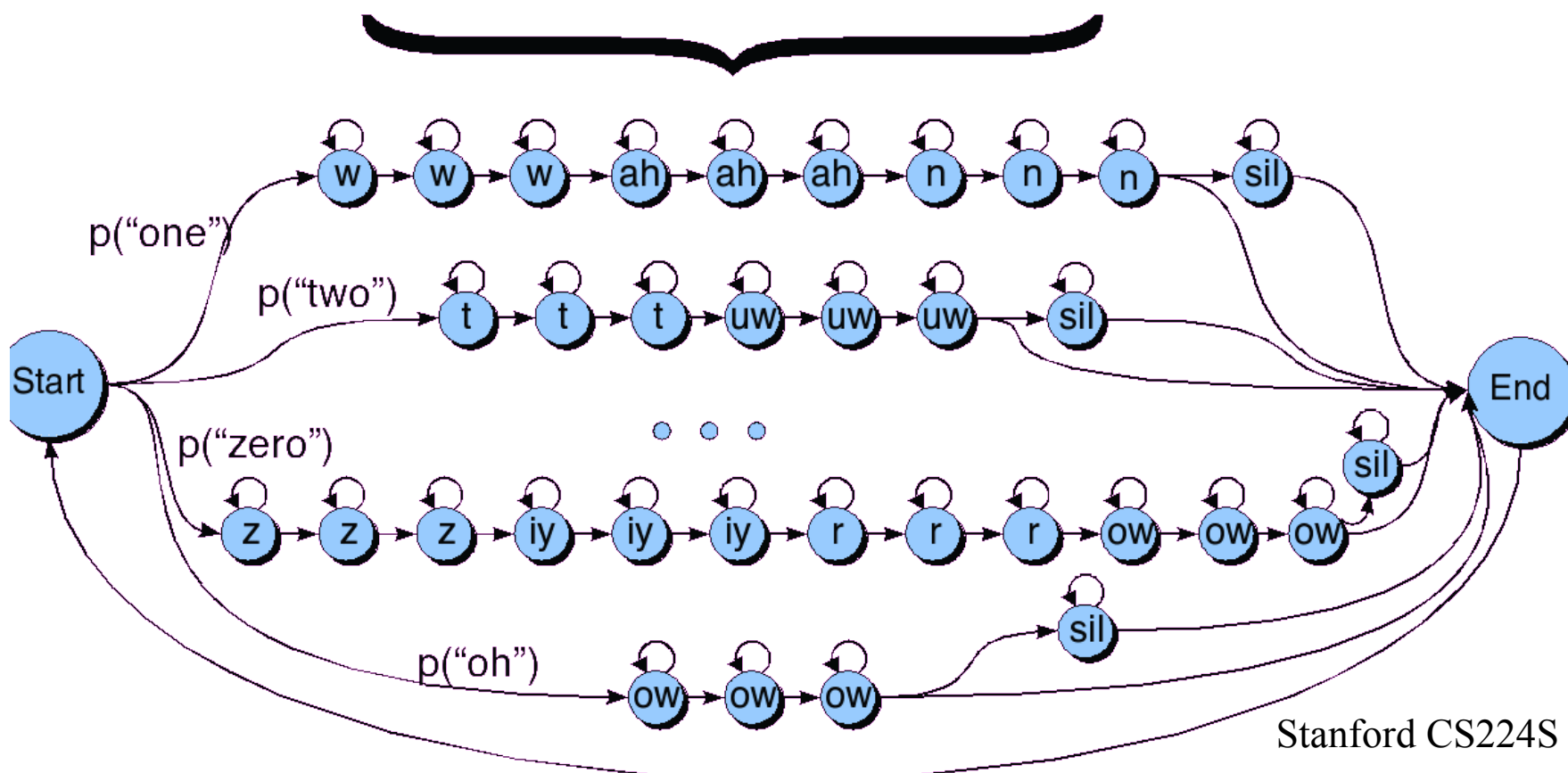
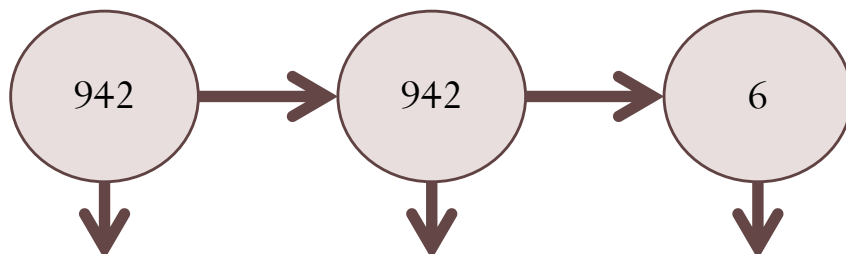# Acoustic Modeling with GMMs

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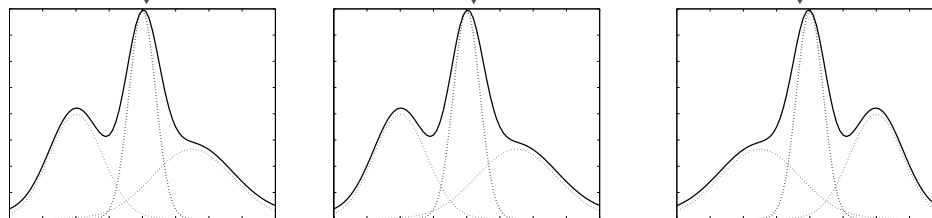
Pronunciation: S – AE – M – S –AH – N
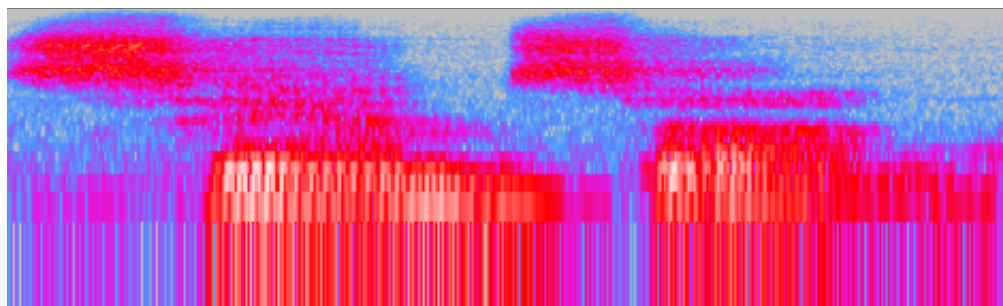
Sub-phones : 942 – 6 – 37 – 8006 – 4422 …

Hidden Markov Model (HMM):

Acoustic Model:

Audio Input:

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

Stanford CS224S Spring 2017

DNN Hybrid Acoustic Models

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

Hidden Markov Model (HMM):

$$942 \rightarrow 942 \rightarrow 6$$

Acoustic Model:

$$P(s \mid x_1) \qquad P(s \mid x_2) \qquad P(s \mid x_3)$$

Audio Input:

Features (x_1)   Features (x_2)   Features (x_3)
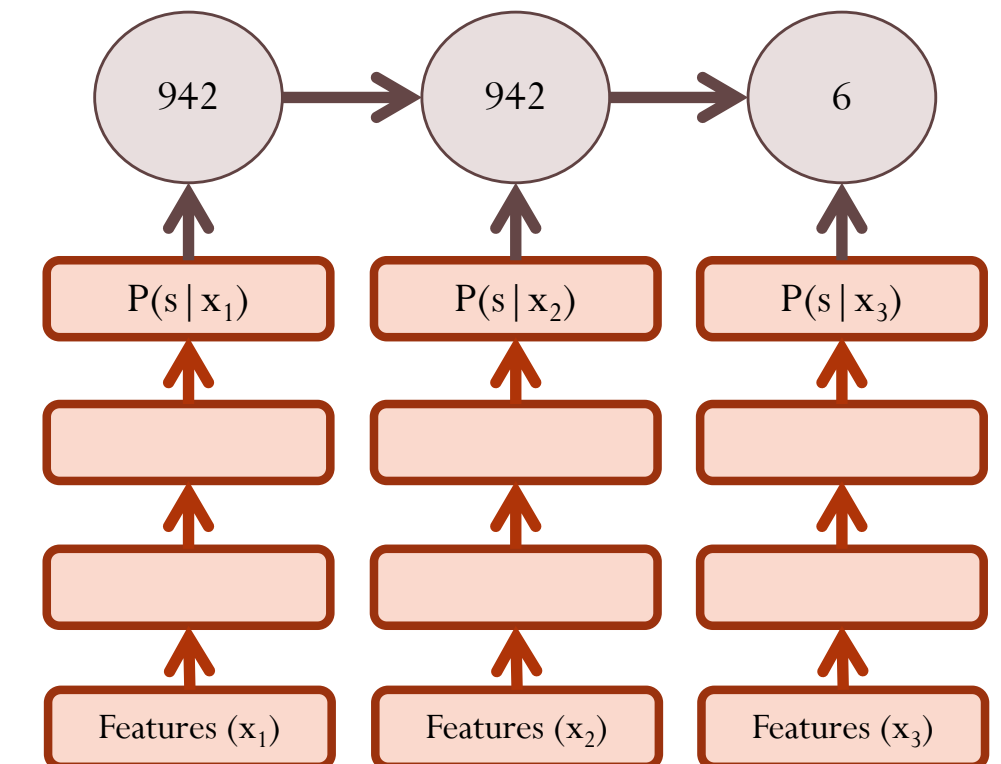
Use a DNN to approximate:
P(s|x)

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

DNN * Constant / State prior

Framework + Isolated Training Limitations

(Maas, Qi, Xie, Hannun, Lengerich, Jurafsky, & Ng. 2017)

Stanford CS224S Spring 2017

Recurrent DNN Hybrid Acoustic Models

Transcription: Samson

Pronunciation: S – AE – M – S –AH – N

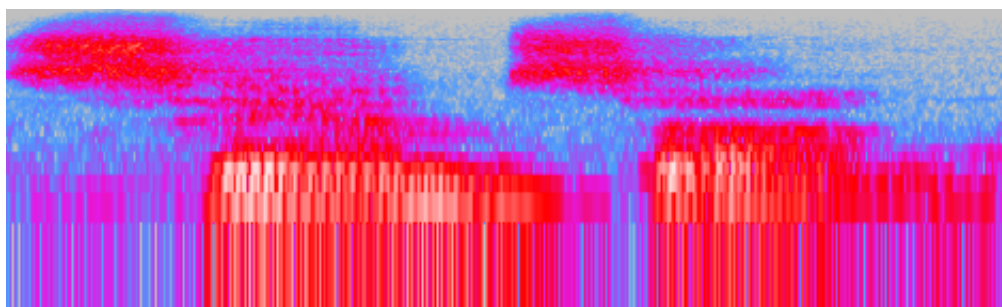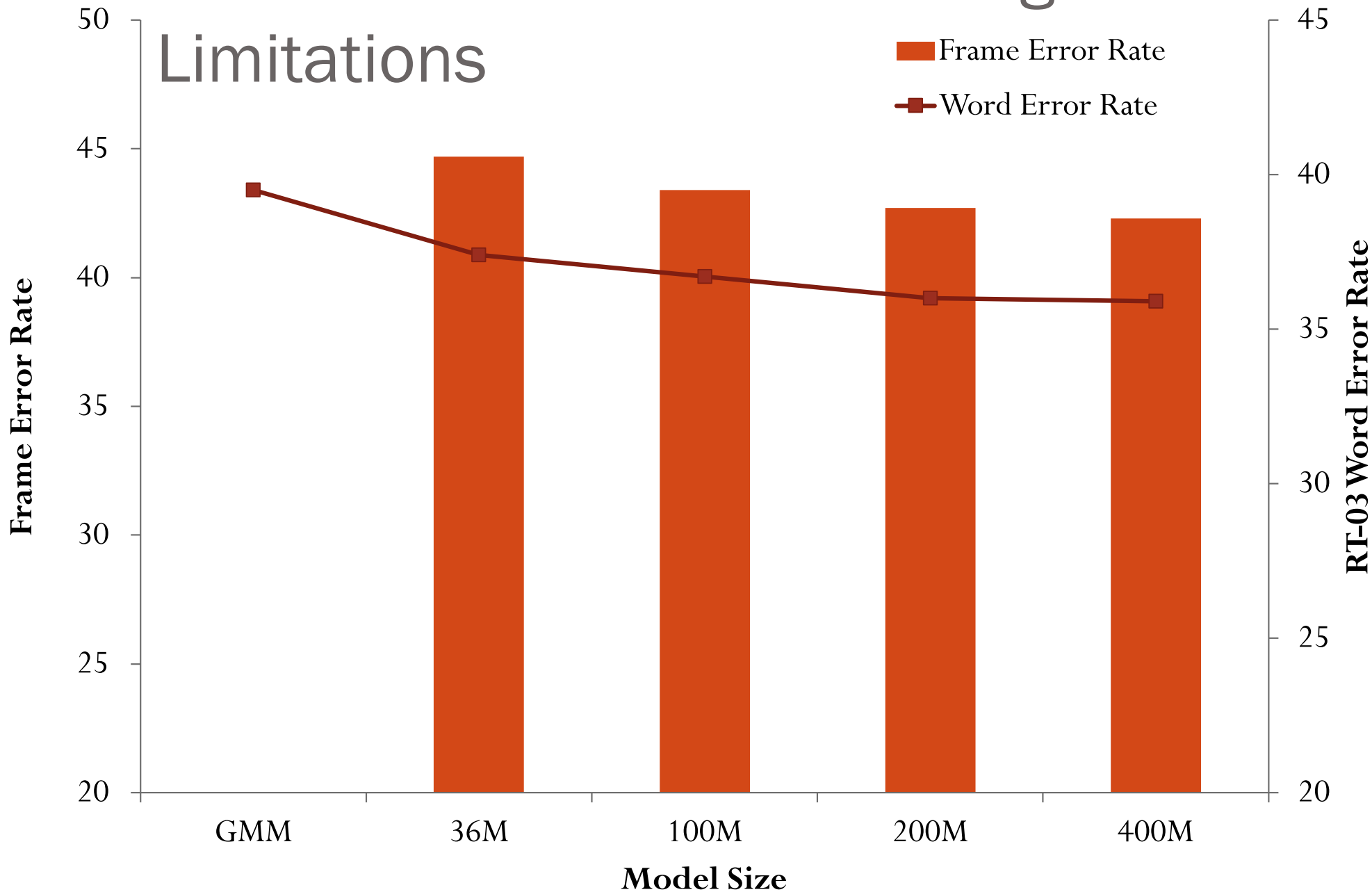**Sub-phones :** 942 – 6 – 37 – 8006 – 4422 …

Hidden Markov Model (HMM):

Acoustic Model:

Audio Input:

Stanford CS224S Spring 2017

Deep Recurrent Network

$\hat{y}^1$

$\hat{y}^2$

Output Layer
$$\hat{y}^2 = Vh^{(1)}(x_2) + c$$

(V, c)

(V, c)

$h^{(2)}(x_1)$

$h^{(2)}(x_2)$

Hidden Layer
$$h^{(2)}(x_2) = \sigma(W^{(2)}h^{(1)}(x_2) + b^{(2)})$$

$(W^{(2)}, b^{(2)})$

$(W^{(2)}, b^{(2)})$

$h^{(1)}(x_1)$

(U)

$h^{(1)}(x_2)$

Hidden Layer
$$h^{(1)}(x_2) = \sigma(W^{(1)}x_2 + b^{(1)} + Uh^{(1)}(x_1))$$

$(W^{(1)}, b^{(1)})$

$(W^{(1)}, b^{(1)})$

$x_1 = [0, x^1, x^2]$

$x_2 = [x^1, x^2, x^3]$

Input
$\mathbf{X}$

HMM-Free Recognition

Transcription: Samson

Pronunciation: S − AE − M ⊘ − N

Sub-phones : 942 − 6 ⊘ 8006 − ⊘22 …

Hidden Markov Model (HMM): 942 → 9⊘ → 6

Acoustic Model: $P(s|x_1)$ $P(s|x_2)$ $P(s|x_3)$

Audio Input: Features (x_1) Features (x_2) Features (x_3)

(Graves & Jaitly. 2014)
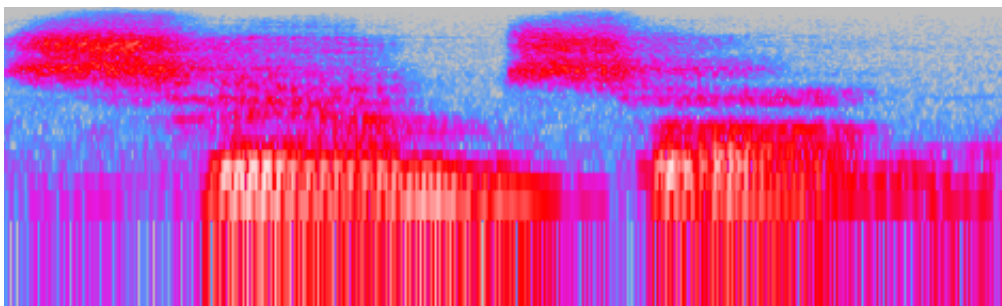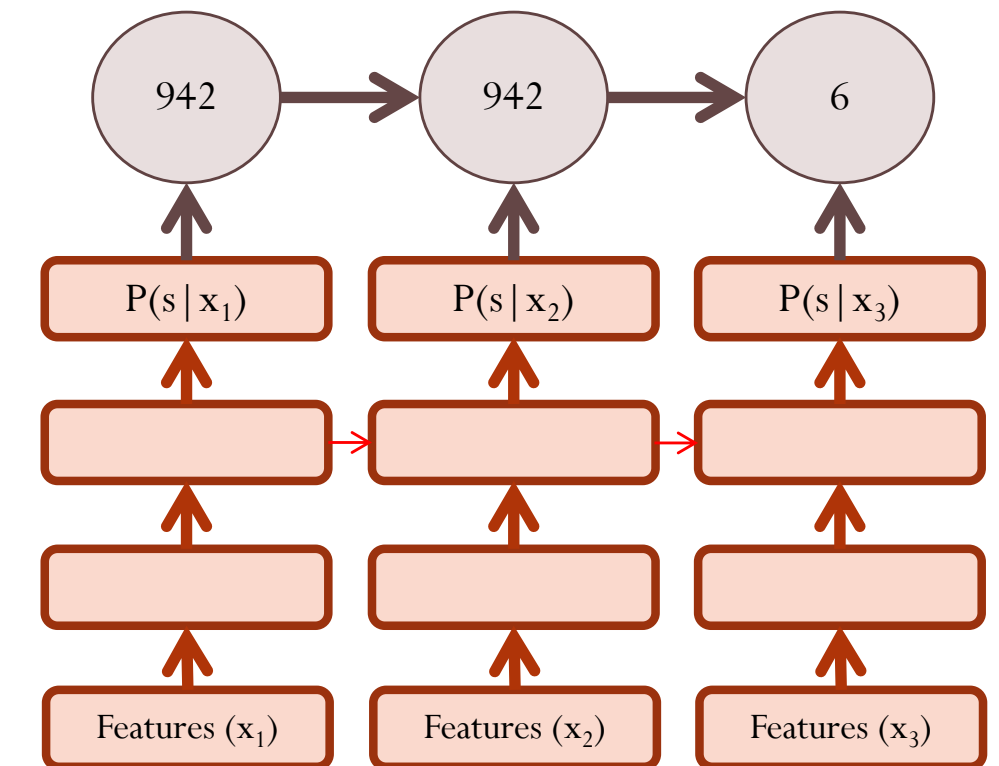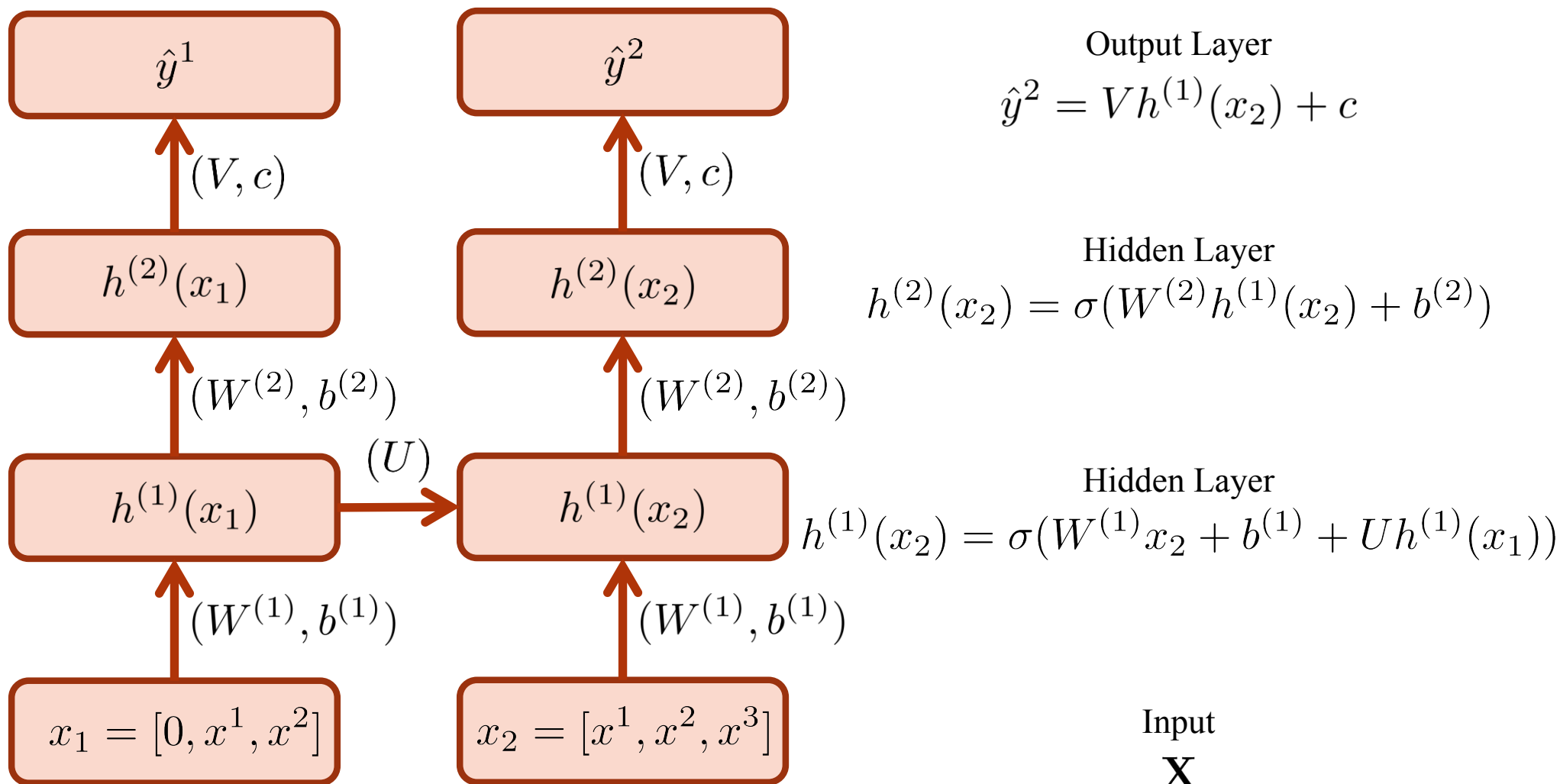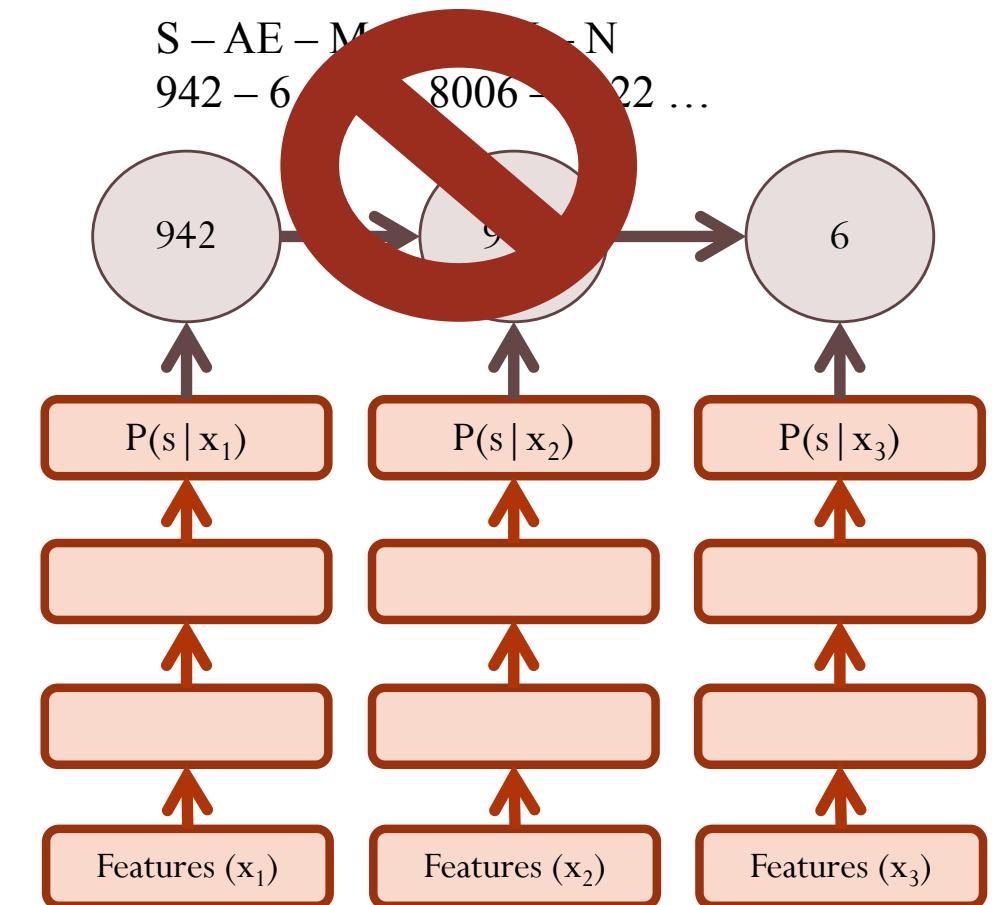
HMM-Free Recognition

Transcription: Samson

Characters: SAMSON

**Collapsing
function:**   SS___AA_M_S___O___NNNN

S	S	_
$P(a \mid x_1)$	$P(a \mid x_2)$	$P(a \mid x_3)$

Acoustic Model:

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

The distribution over *characters*

Audio Input:

Features (x_1)	Features (x_2)	Features (x_3)

(Graves & Jaitly. 2014)

Stanford CS224S Spring 2017
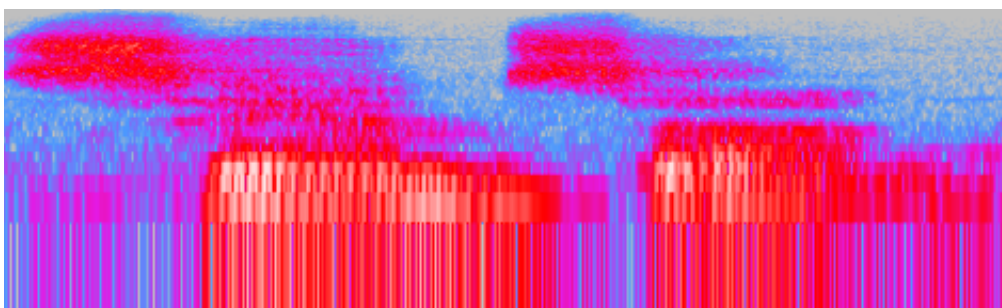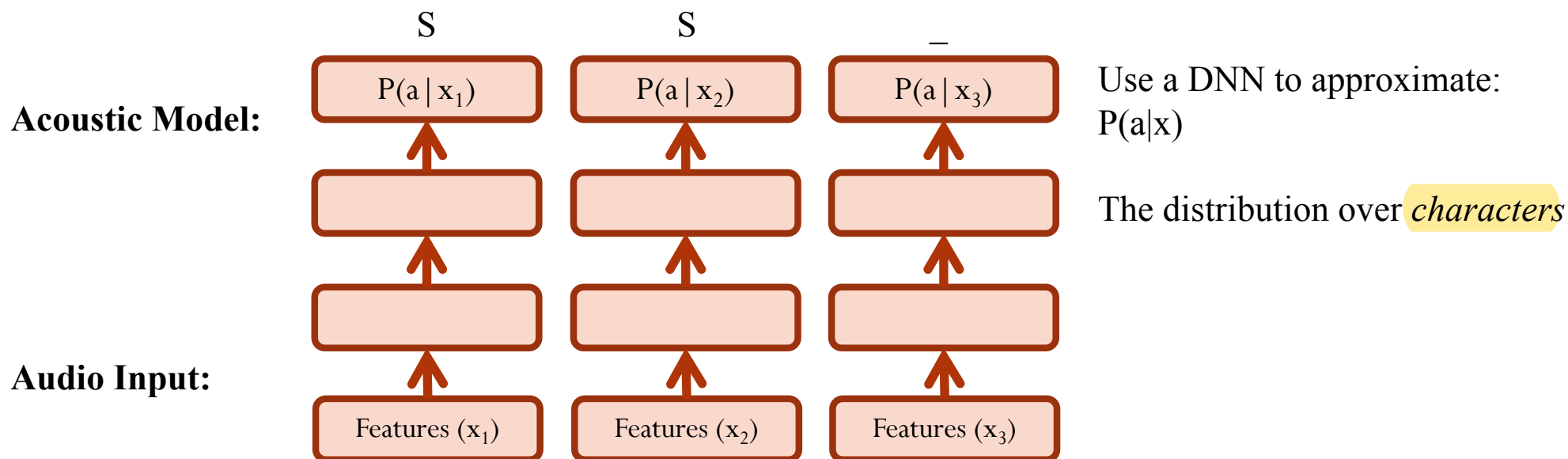
Example Results (WSJ)

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
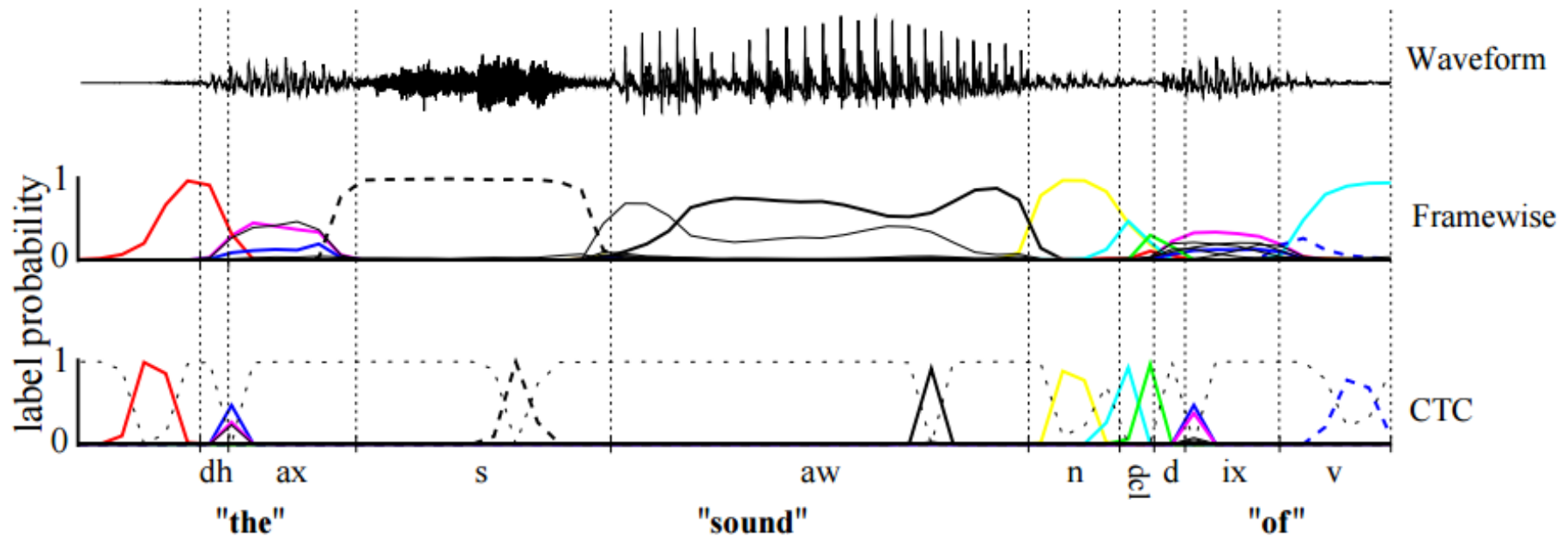
Earlier work on CTC with phonemes

Table 1. **Label Error Rate (LER) on TIMIT.** CTC and hybrid results are means over 5 runs, $\pm$ standard error. All differences were significant ($p < 0.01$), except between weighted error BLSTM/HMM and CTC (best path).
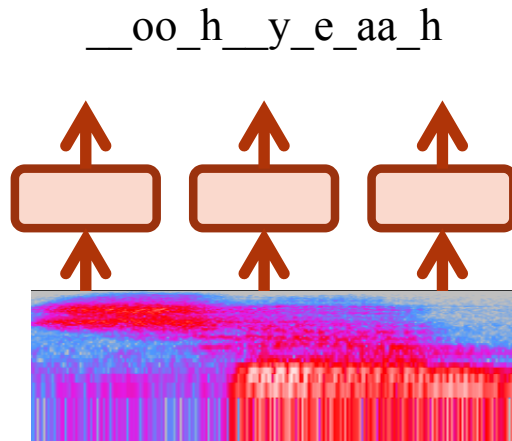
System	LER
Context-independent HMM	38.85 %
Context-dependent HMM	35.21 %
BLSTM/HMM	33.84 $\pm$ 0.06 %
Weighted error BLSTM/HMM	31.57 $\pm$ 0.06 %
CTC (best path)	31.47 $\pm$ 0.21 %
CTC (prefix search)	30.51 $\pm$ 0.19 %

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

Decoding with a Language Model

Lexicon   [a, …, zebra]

**Language
Model**   p("yeah" | "oh")

**Character
Probabilities**   __oo_h__y_e_aa_h

Character Error Rate

| | None | Lexicon | Bigram |

Word Error Rate

| | None | Lexicon | Bigram |

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

Stanford CS224S Spring 2017
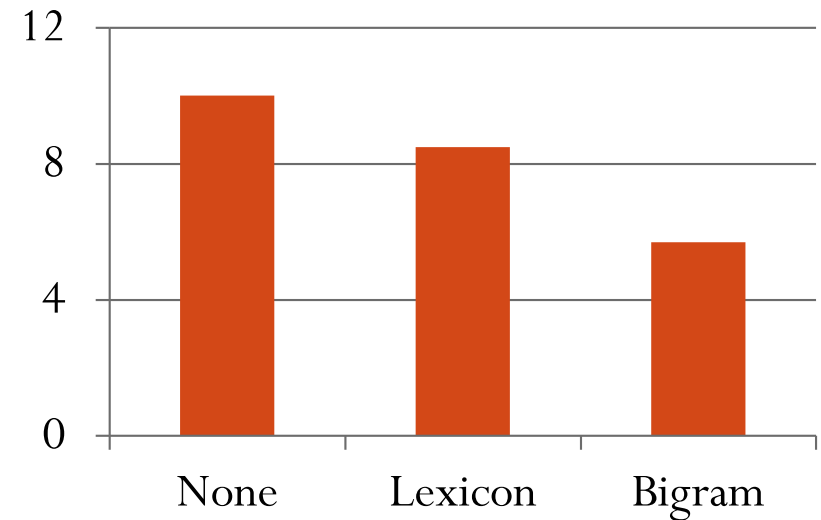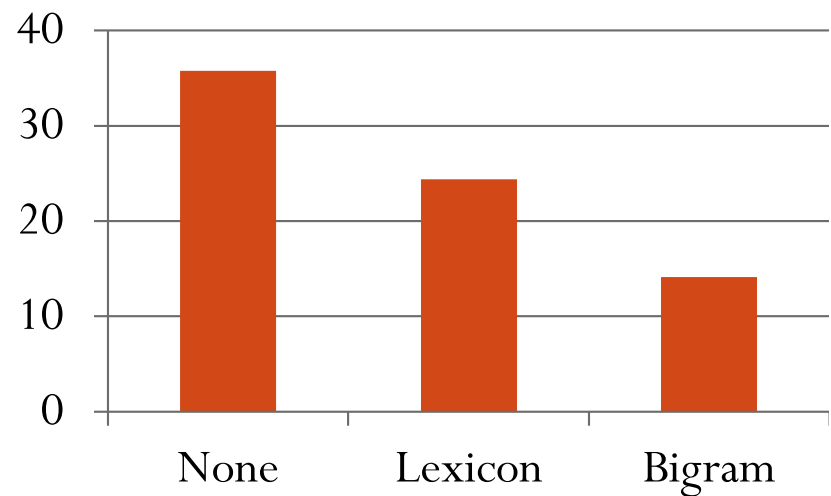
Loss functions and architecture

- *What function to fit*
- Loss function
- HMM-DNN uses independent per-frame classification with force alignment hard labels
- CTC independent per-frame but cleverly allows for multiple possible labelings

- *How do we approximate that function*
- Neural network architecture
- HMM-DNN typically fine with just DNN
- CTC needs recurrent NN
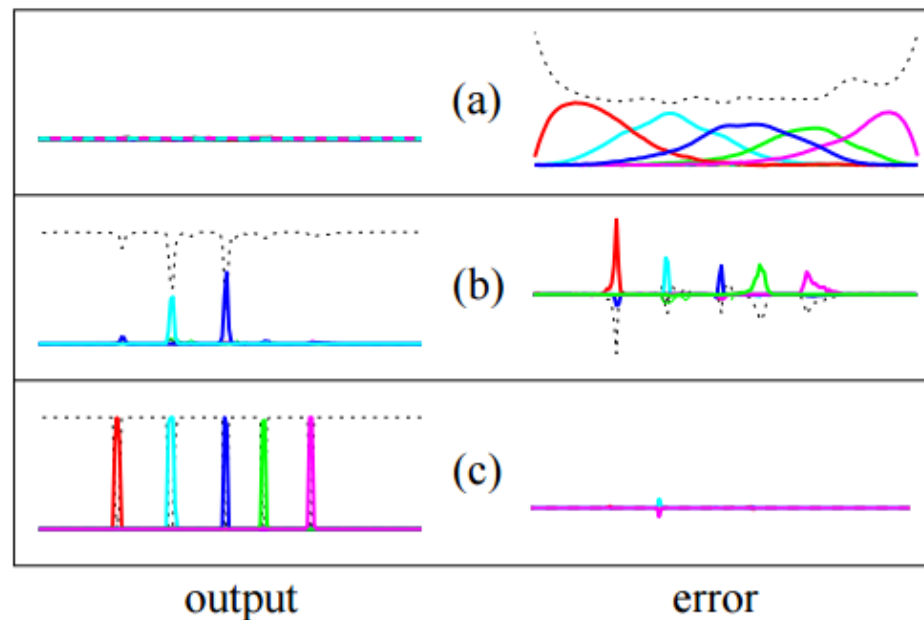
CTC loss during training

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)

Recurrence Matters!
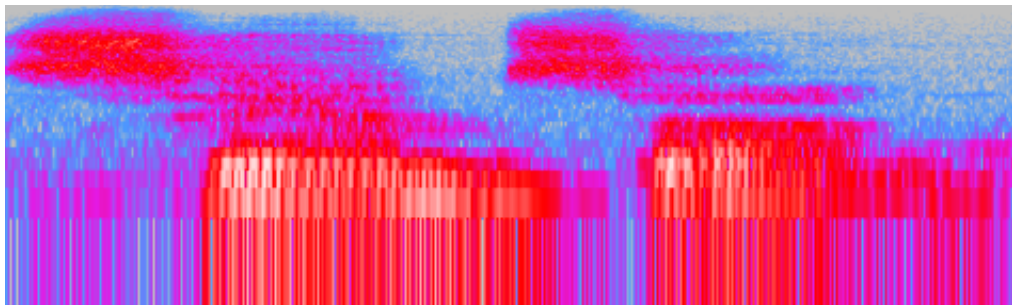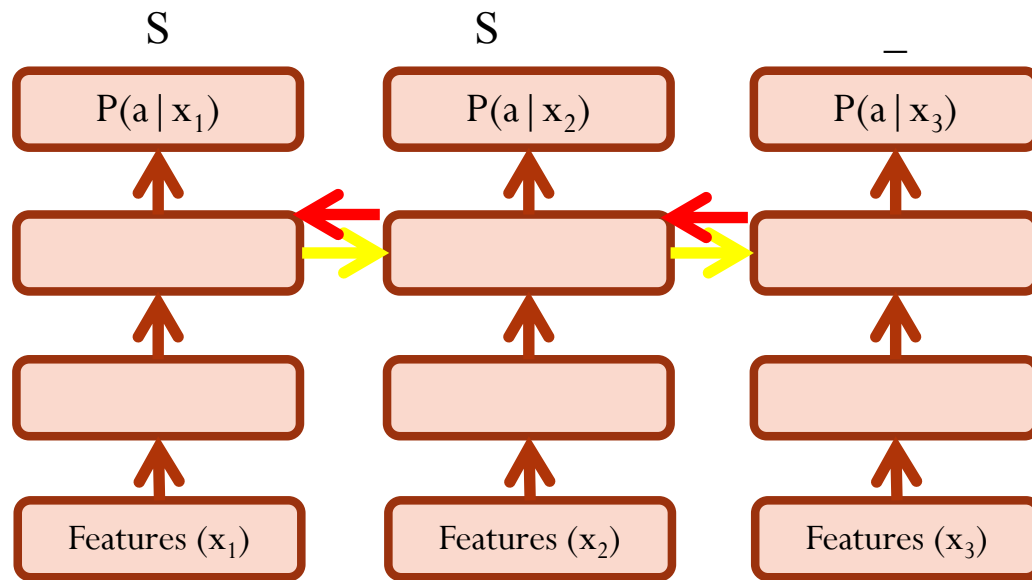
Architecture	CER
DNN	22

CTC Loss Function

- Maximum log likelihood training of transcript

- Intuition: Alignments are unknown so integrate over all possible time-character alignments

$$\mathcal{L}_{\text{CTC}}(X, W) = \sum_{C: \ \kappa(C)=W} p(C|X)$$

$$= \sum_{C: \ \kappa(C)=W} \prod_{t=1}^{T} p(c_t|X).$$

- Example: W = "hi", T = 3
  possible C such that K(C) = W:
  `hhi, hii, _hi, h_i, hi_`

(Graves & Jaitly. 2014)

CTC Objective Function

Labels at each time index are conditionally independent (like HMMs)

$$\Pr(\mathbf{a}|\boldsymbol{x}) = \prod_{t=1}^{T} \Pr(a_t, t|\boldsymbol{x})$$

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

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

Output label: AB

Time-level labelings: AB, _AB, A_B, … _A_B_

Final objective maximizes probability of true labels:

$$CTC(\boldsymbol{x}) = -\log \Pr(y^*|\boldsymbol{x})$$

(Graves & Jaitly, ICML 2014)

Stanford CS224S Spring 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_____lllldd_____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

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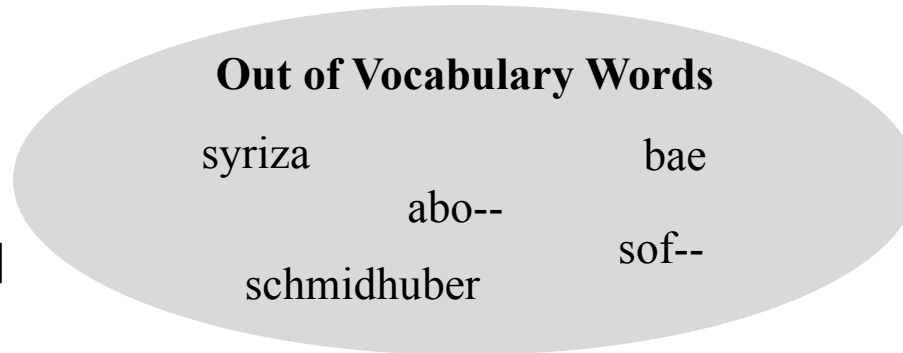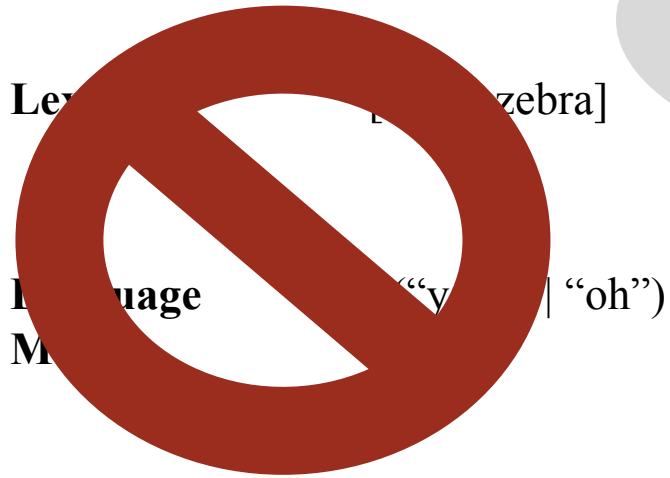
After collapsing:
yet a rehbilitation cru is onhand in the building loogging bricks plaster and blueprins four forty two new betin epartments

Reference:
yet a rehabilitation crew is on hand in the building lugging bricks plaster and blueprints for forty two new bedroom apartments
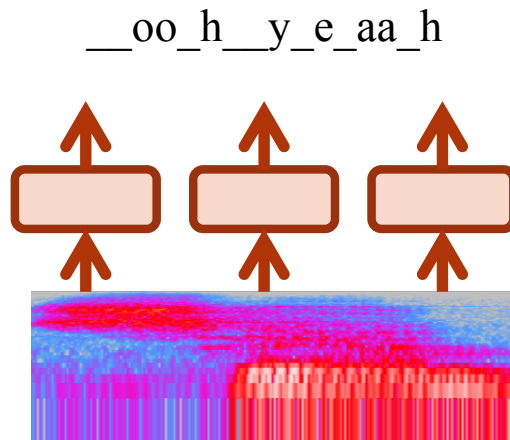
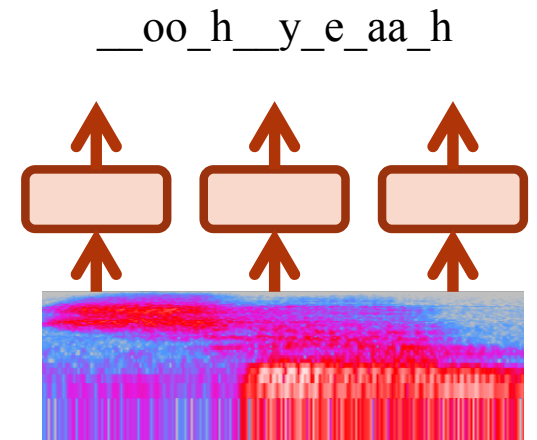(Hannun, Maas, Jurafsky, & Ng. 2014)   Stanford CS224S Spring 2017

Rethinking Decoding

Out of Vocabulary Words

syriza   bae

abo--

sof--

schmidhuber

Lex [... zebra]

Language Model ("y... | "oh")

Character Language Model $p(h \mid o,h, ,y,e,a,)$

Character Probabilities   __oo_h__y_e_aa_h

Character Probabilities   __oo_h__y_e_aa_h

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

Stanford CS224S Spring 2017

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
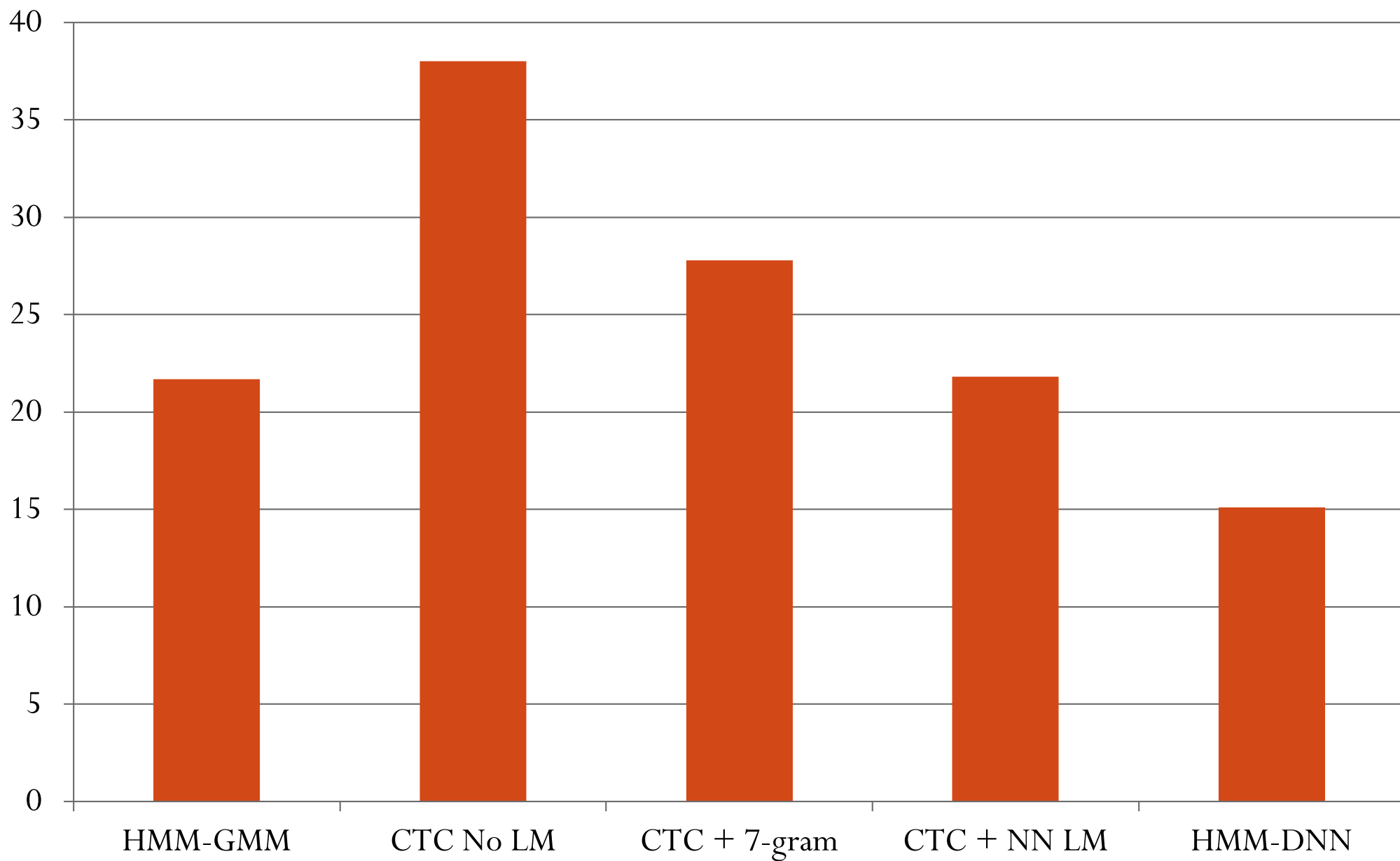   **end for**
    end for
    $Z_t \leftarrow k$ most probable s by $p_{\text{tot}}(s|x_{1:t})|s|^{\beta}$ in Z_t ▷ Apply beam
end for
Return $\arg\max_{s \in Z_t} p_{\text{tot}}(s|x_{1:T})|s|^{\beta}$

Lexicon-Free & HMM-Free on Switchboard

(Maas*, Xie*, Jurafsky, & Ng. 2015)   Stanford CS224S Spring 2017

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)

Comparing CLMs

Switchboard Word Error Rate
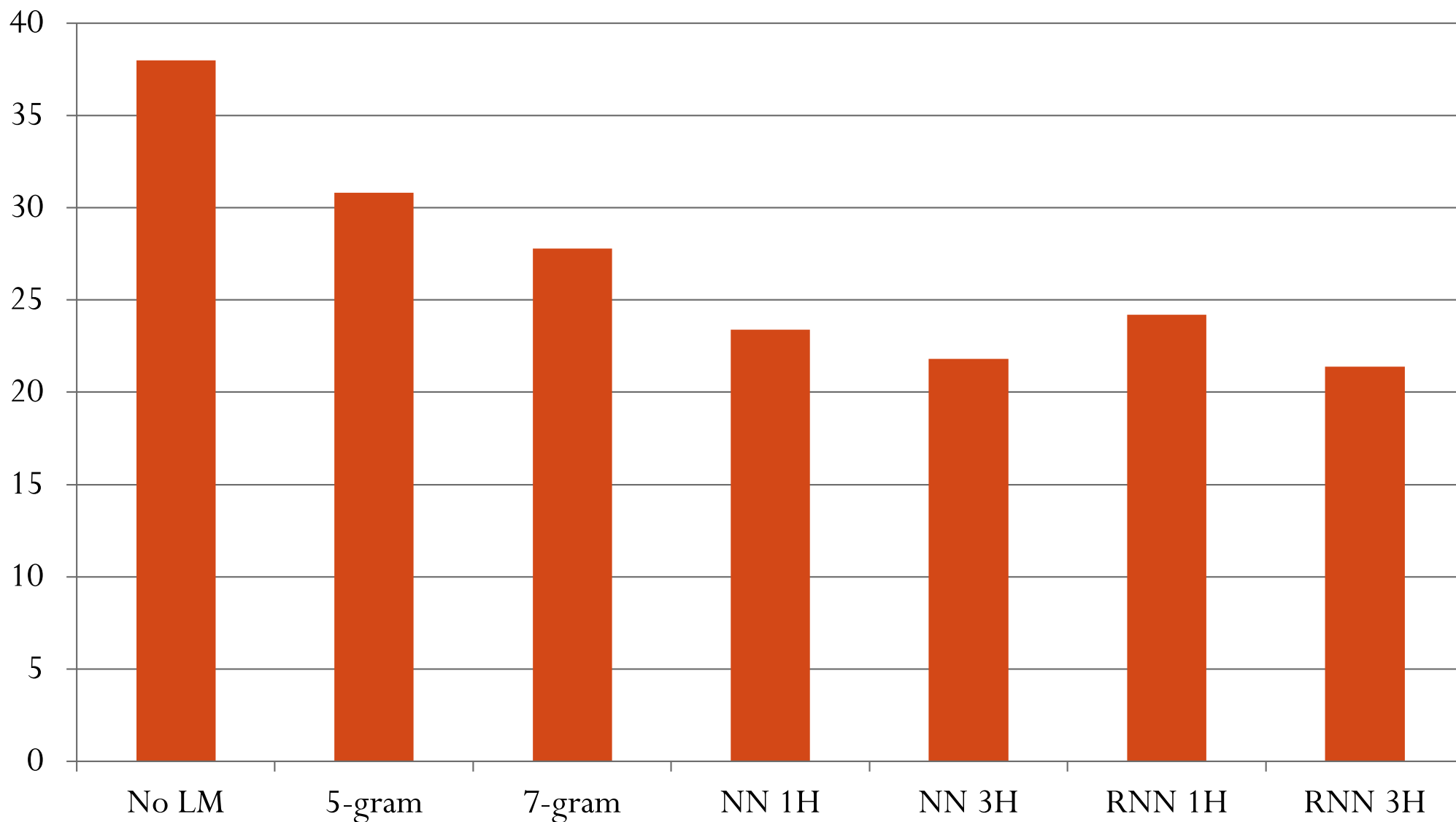
All NN models have 5M total parameters
(Maas*, Xie*, Jurafsky, & Ng. 2015)

Stanford CS224S Spring 2017

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

Comparing Alignments
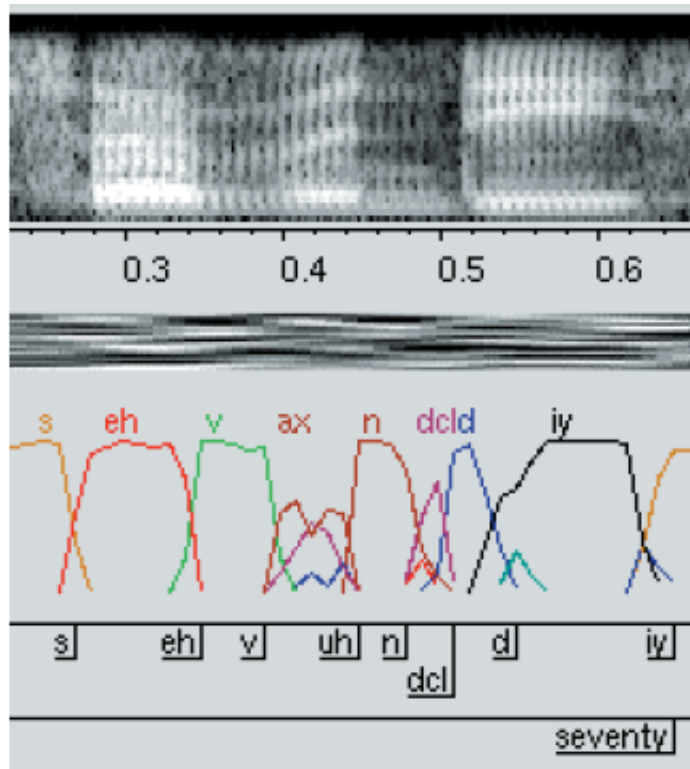

HMM-GMM phone probabilities
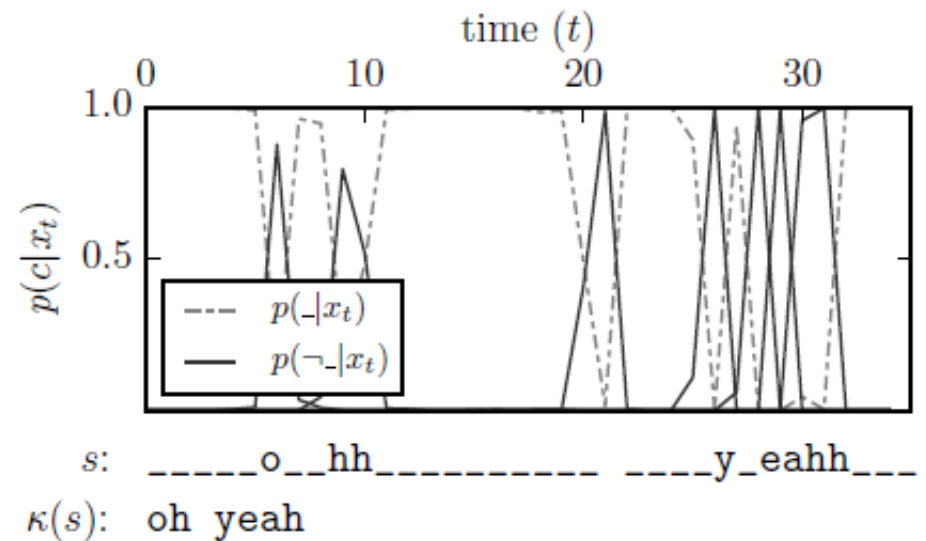
CTC character probabilities

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
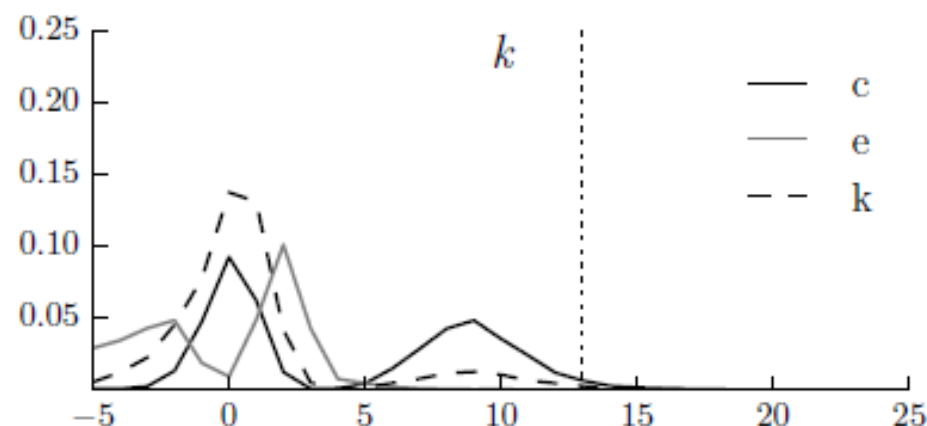
Learning Phonemes and Timing
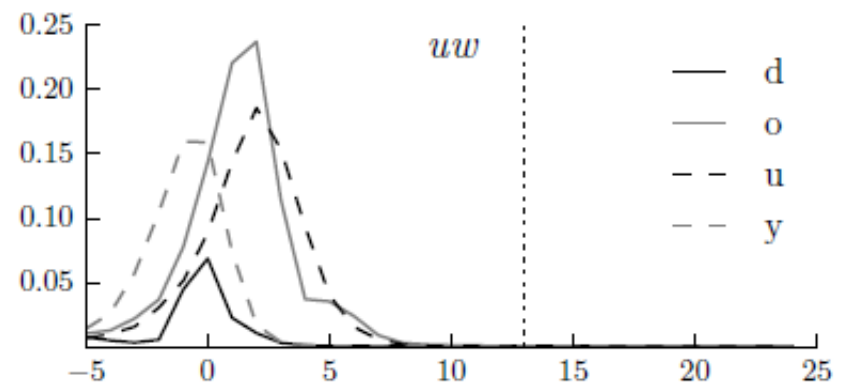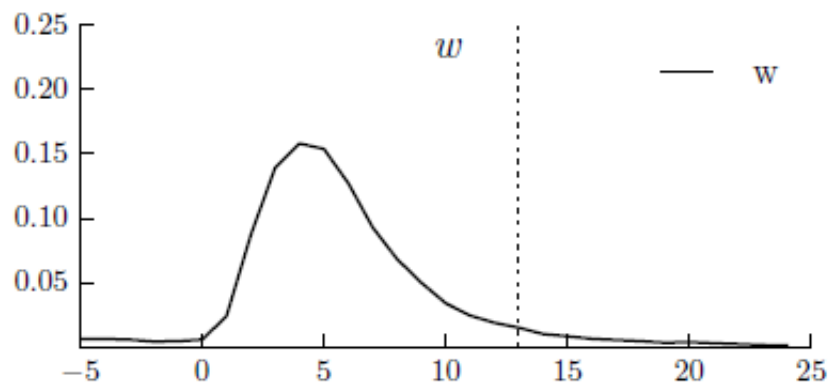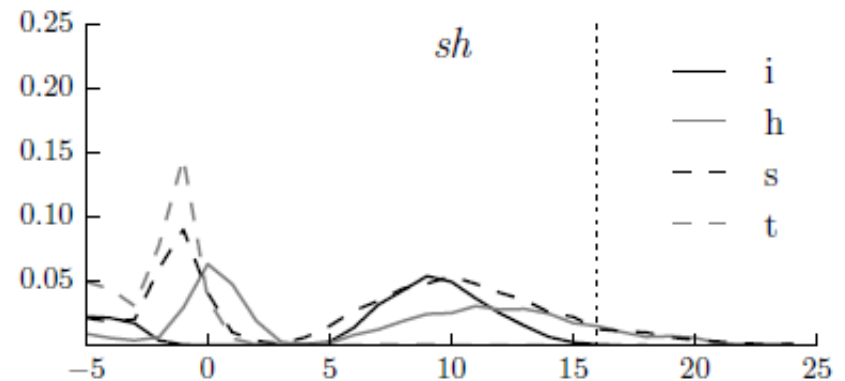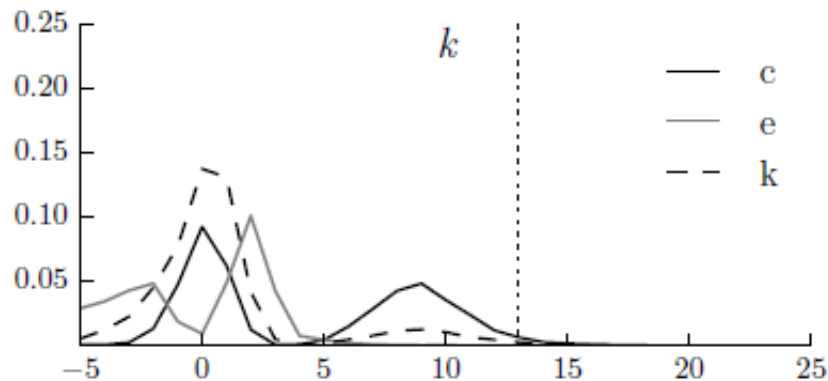
Scaling end to end models: Baidu deep speech

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

Table 2: A summary of the datasets used to train Deep Speech. The Wall Street Journal, Switchboard and Fisher [3] corpora are all published by the Linguistic Data Consortium.

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**

Table 3: Published error rates (%WER) on Switchboard dataset splits. The columns labeled "SWB" and "CH" are respectively the easy and hard subsets of Hub5'00.
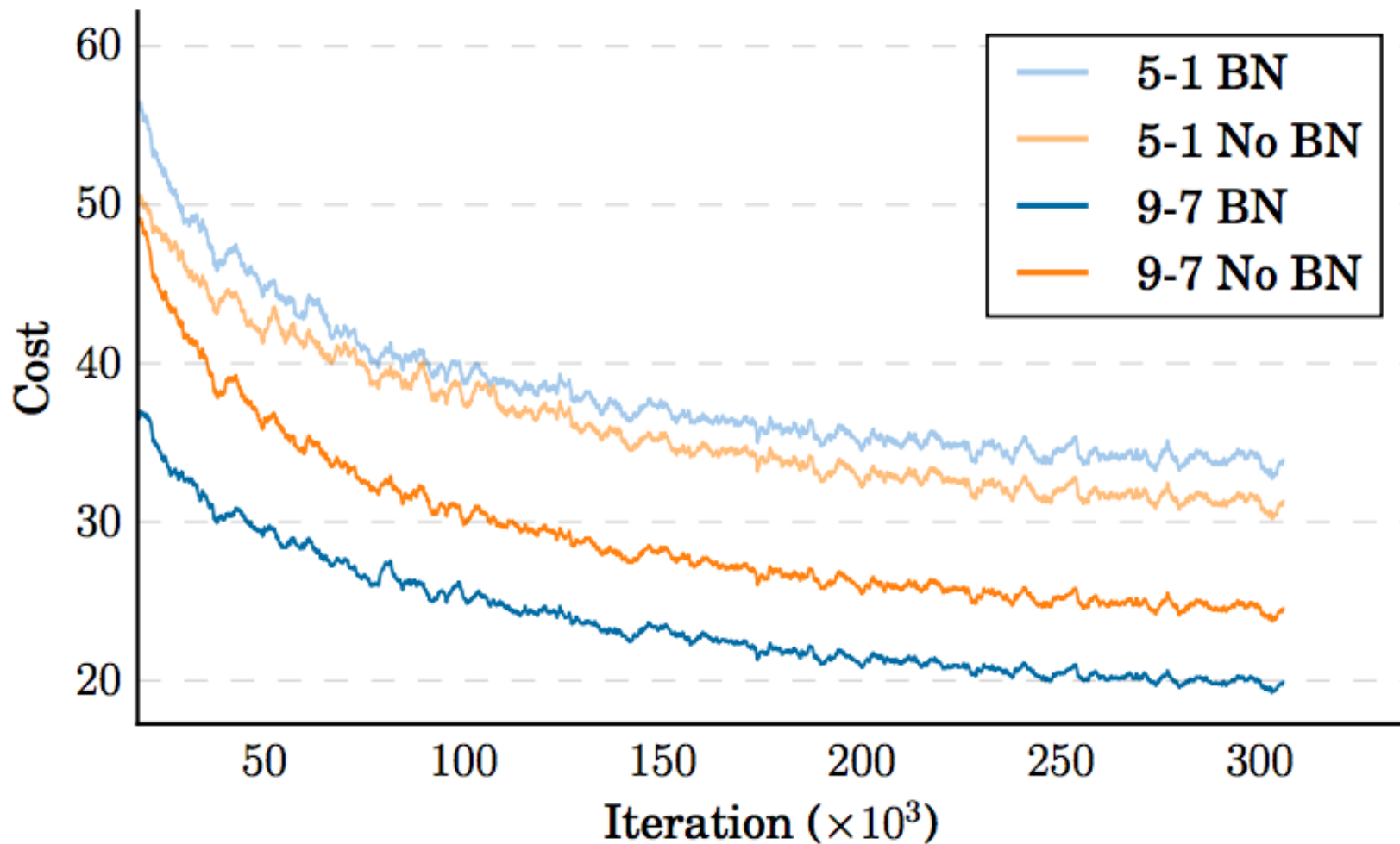
(Hannun *et al.* 2014)

Deep Speech – Deep RNN
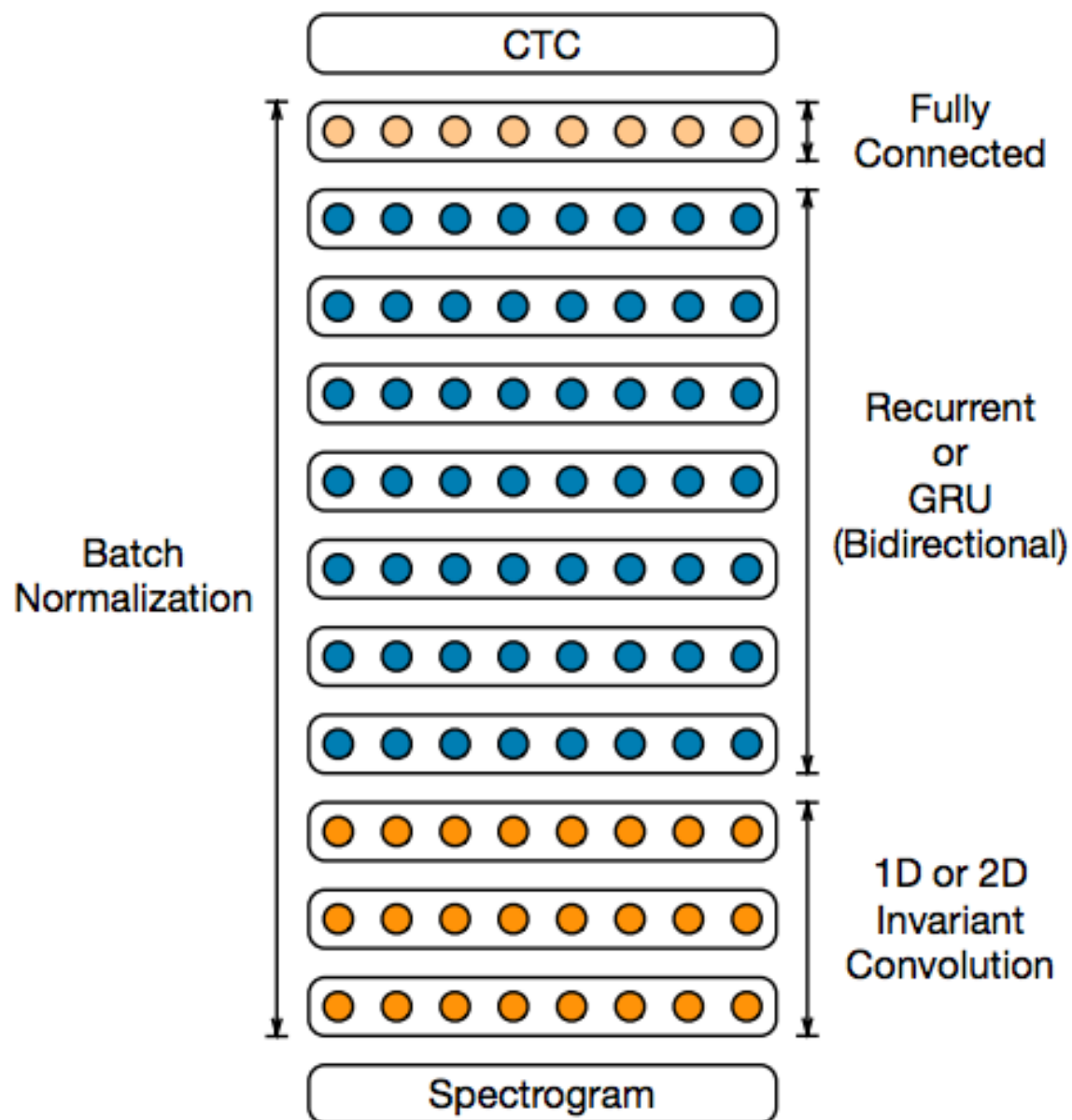
Deep Speech – Batch Norm for RNNs
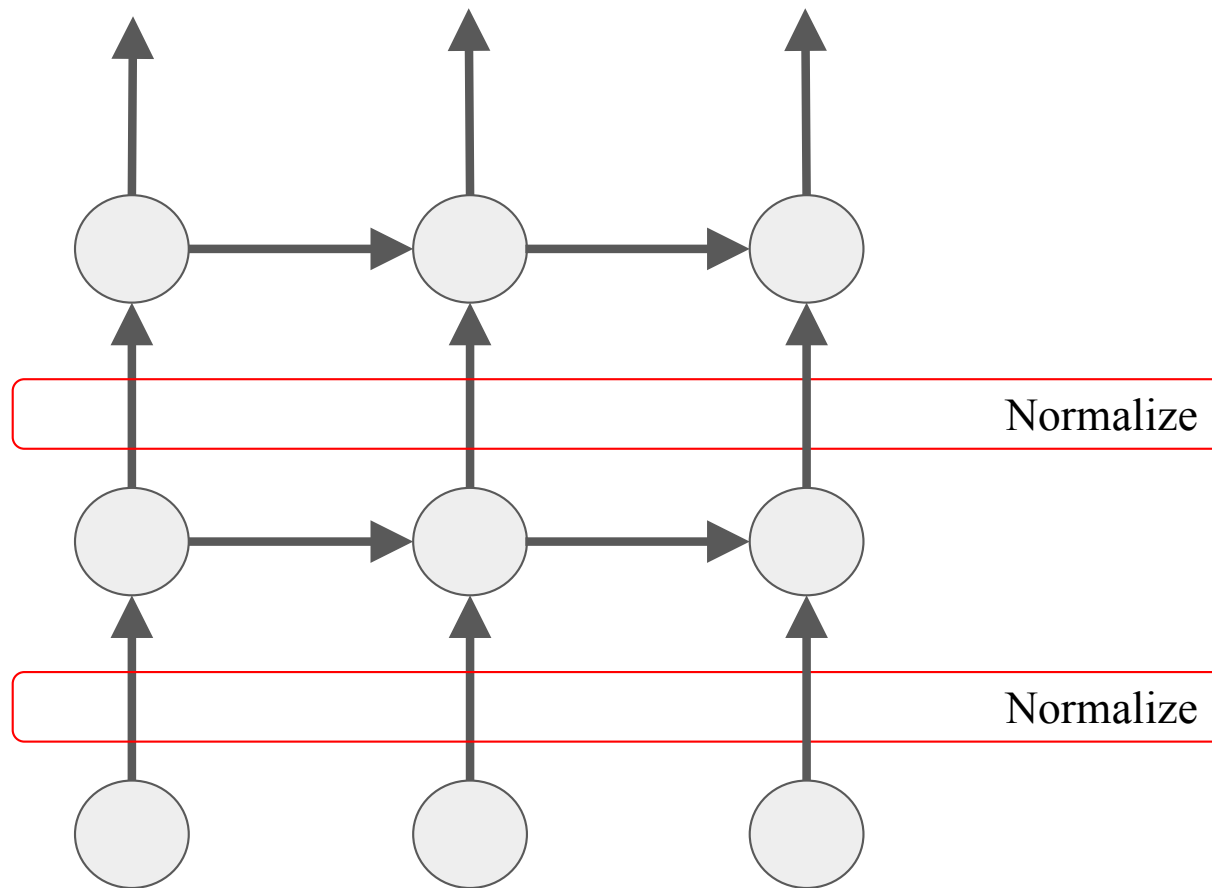
Deep Speech – Batch Norm for RNNs

Deep Speech - Hours of speech data

Language	Hours
English	12,000
Mandarin	10,000

Where does the data come from?
- Public benchmarks (English)

- Internal manually labelled data (English and Mandarin)

- Captioned videos (English and Mandarin)

Deep Speech - Captioned Video Data Pipeline

1. Download publicly available video + captions.

1. Align caption to video with CTC Model

1. Segment at regions of silence

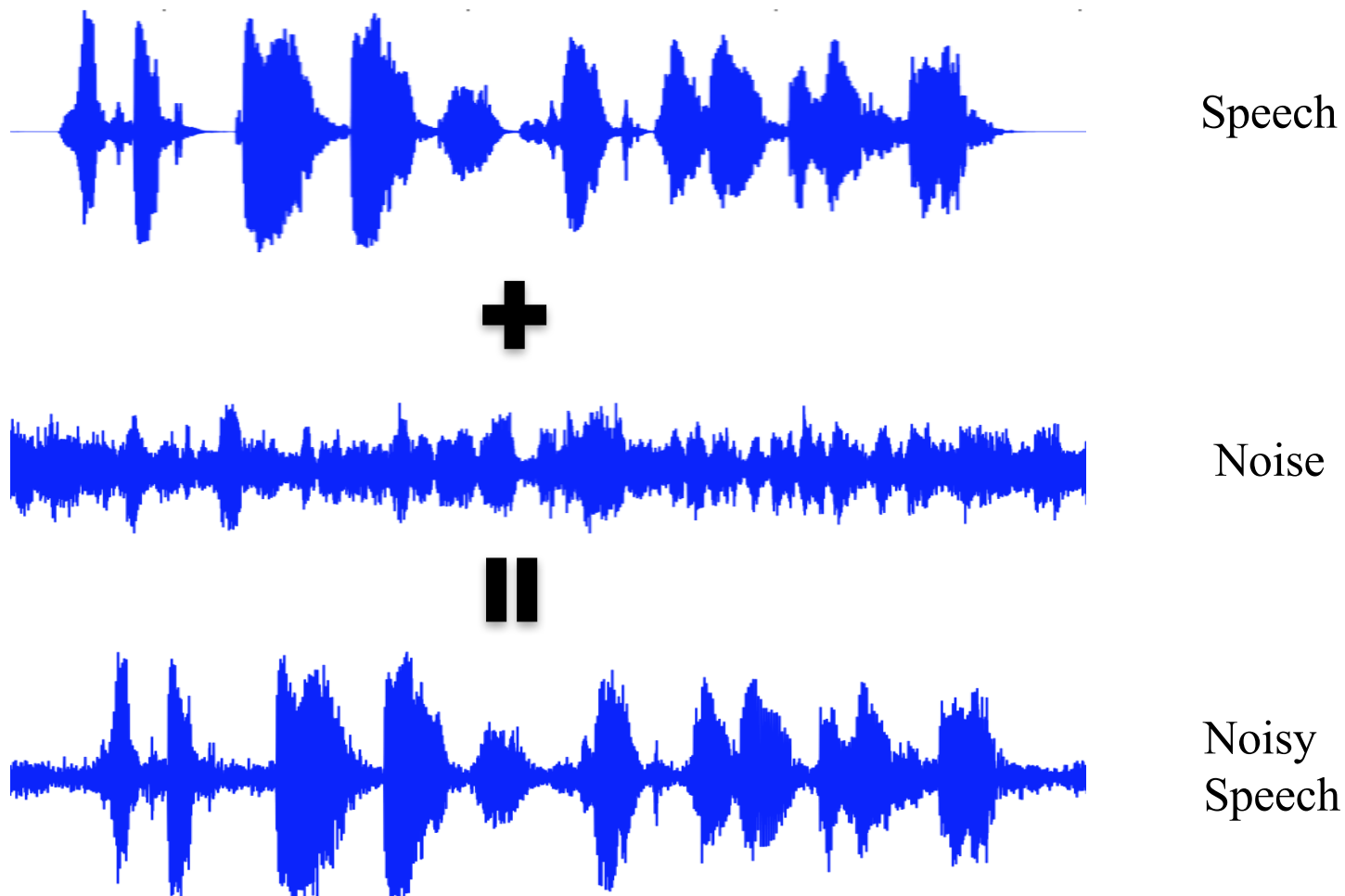1. Use simple classifier to throw out very noisy samples.

Deep Speech - Captioned Video Data Pipeline

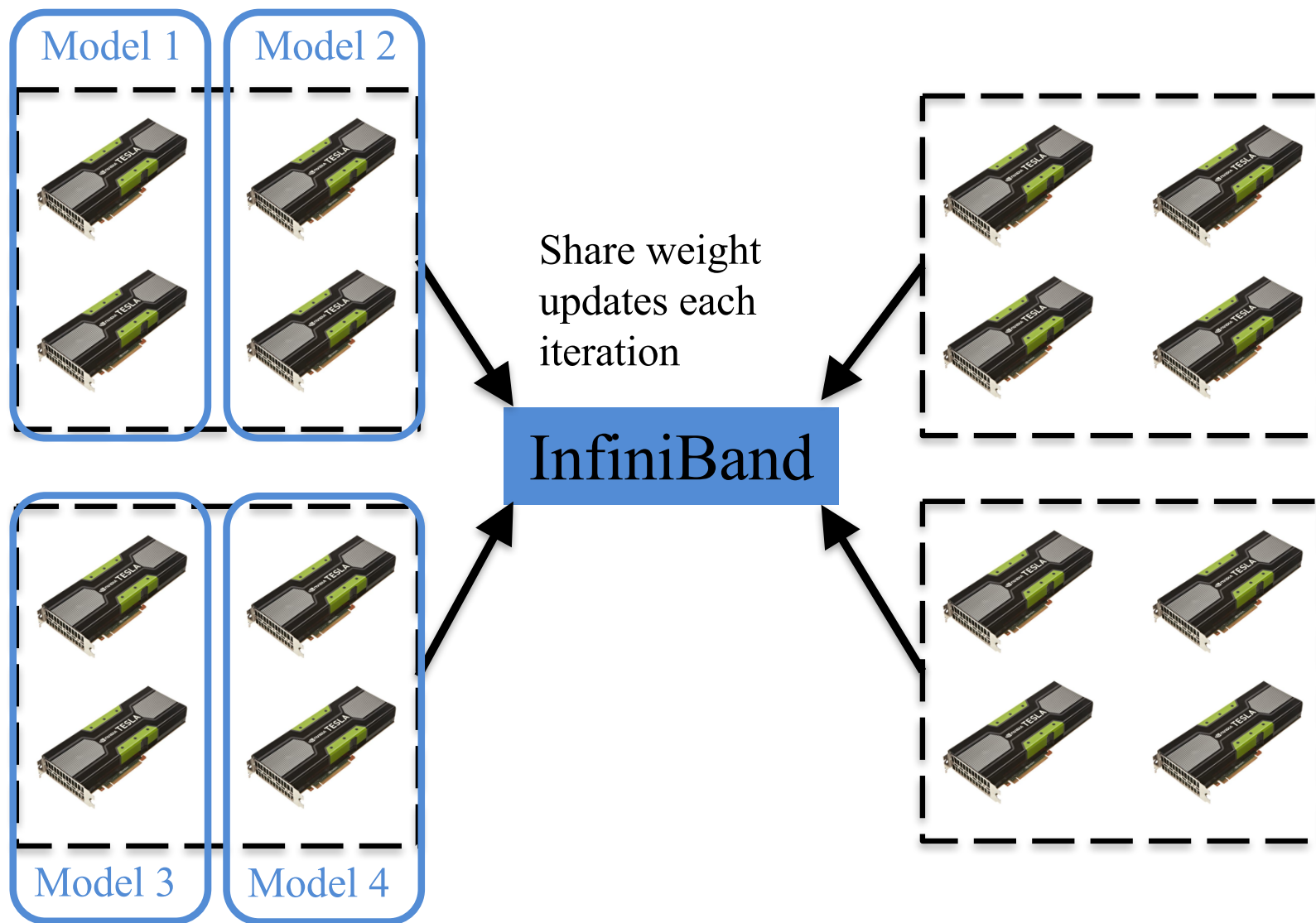Align with a model trained with CTC?

$$\sum_{\ell \in \text{Align}(x,y)} \prod_{t}^{T} p_{\text{ctc}}(\ell_t | x; \theta)$$

$$\downarrow$$

$$\underset{\ell \in \text{Align}(x,y)}{\arg\max} \prod_{t}^{T} p_{\text{ctc}}(\ell_t | x; \theta)$$

Deep Speech - Even more data!

Augmentation: noise synthesis, reverb, time-stretching, pitch-shifting,...

Speech

+

Noise

=

Noisy
Speech

Deep Speech – Data Parallel GPU Scaling

Share weight updates each iteration

InfiniBand

Model 1   Model 2

Model 3   Model 4
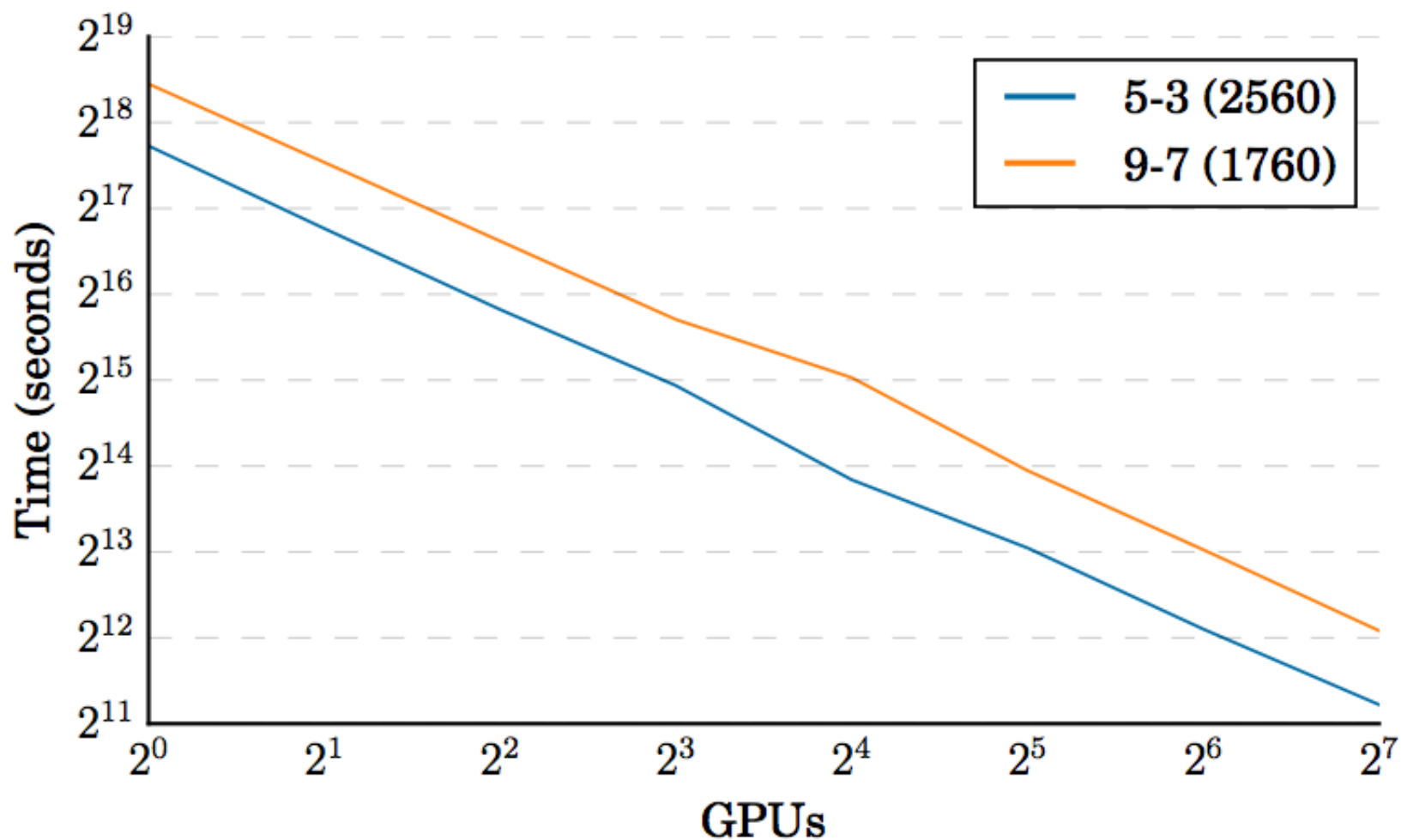
Stanford CS224S Spring 2017

Deep Speech – Data Parallel GPU Scaling

Custom Ring Reduce avoids extraneous copies to CPU memory.

# GPUs	OpenMPI All-reduce (s)*	Custom All-reduce (s)*	Factor Speedup
4	55359	2587	21.4
8	48881	2470	19.8
16	21562	1393	15.5

*Measures time spent in all-reduce for a single epoch.

Deep Speech – Data Parallel GPU Scaling

Deep Speech – Some results

Architecture	English (WER)	Mandarin (WER)
5-layer 1-RNN	13.55	15.41
5-layer 3-RNN	11.61	11.85
5-layer 3-RNN + BatchNorm	10.56	9.39
9-layer 7-RNN + BatchNorm + Frequency Convolution	9.52	7.93

Deep Speech – Deployment

- Bi-directional models give almost 10% relative boost … but we can't deploy them.

- ASR latencies for voice search <50ms

- For 3 second audio would need to decode 60x faster than realtime!

Deep Speech – Lookahead convolution
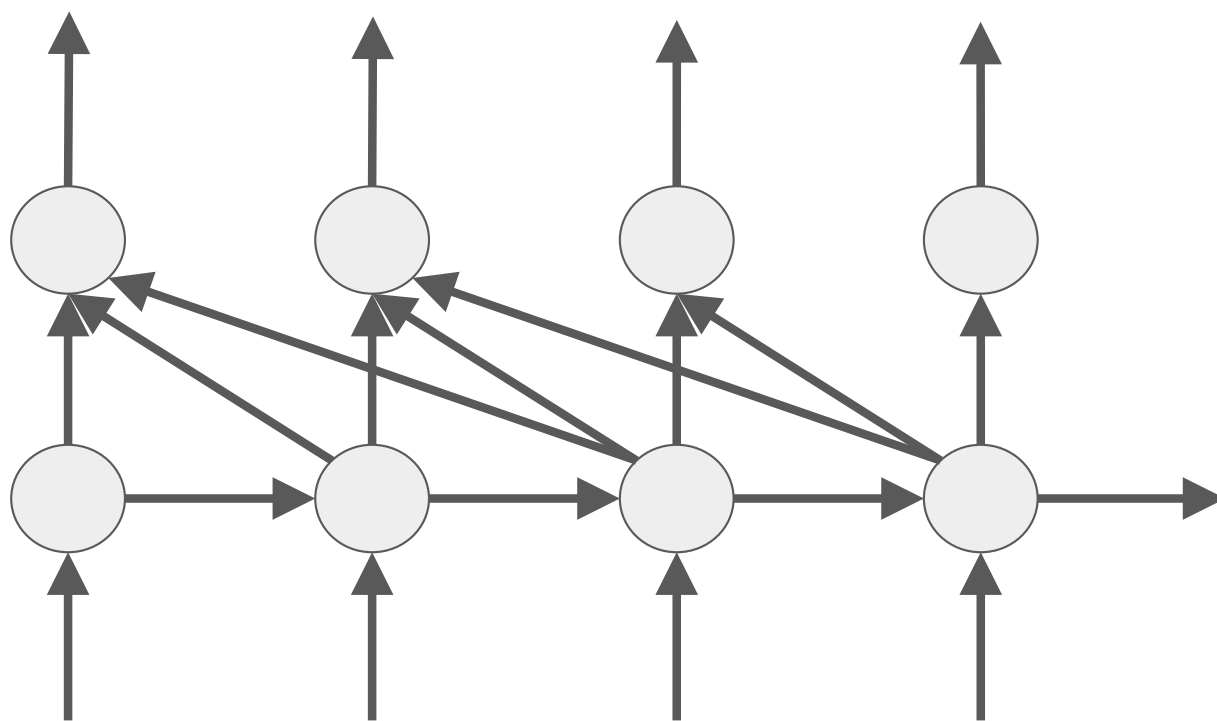
$$\mathbf{h}_t = \sum_{j=1}^{\tau+1} \mathbf{w}_j \odot \mathbf{x}_{t+j-1}$$

Deep Speech – Lookahead convolution

For a lookahead of 20 time-steps (about 800ms in the future)

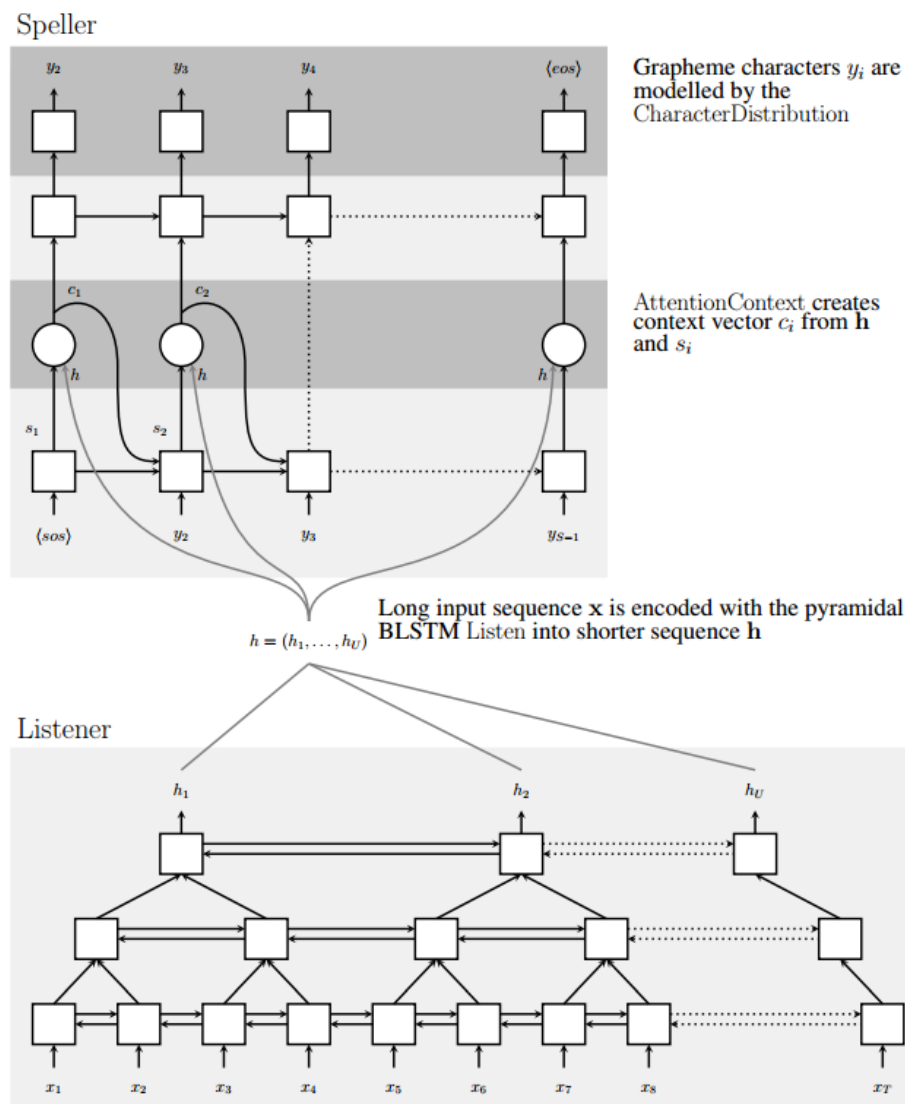| Model	English (WER)	Chinese (WER)
Forward only	18.8	15.7
Forward + Lookahead (+50k params)	16.8	13.5
Bidirectional (+12M params)	15.4	12.8

Listen, Attend, and Spell

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

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

Listen, Attend, and Spell

Alignment between the Characters and Audio
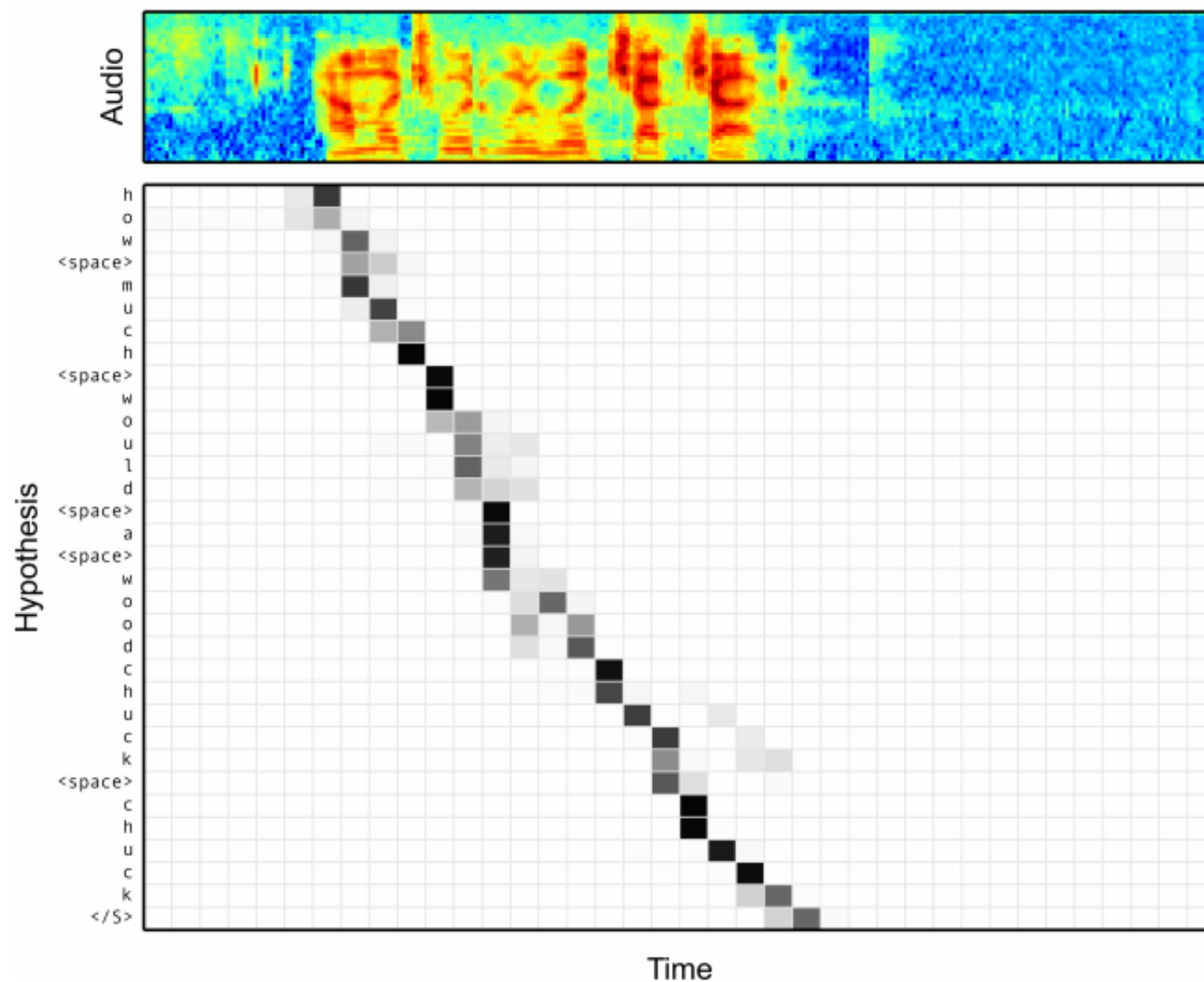
Listen, Attend, and Spell

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)

Attention-based sequence generation

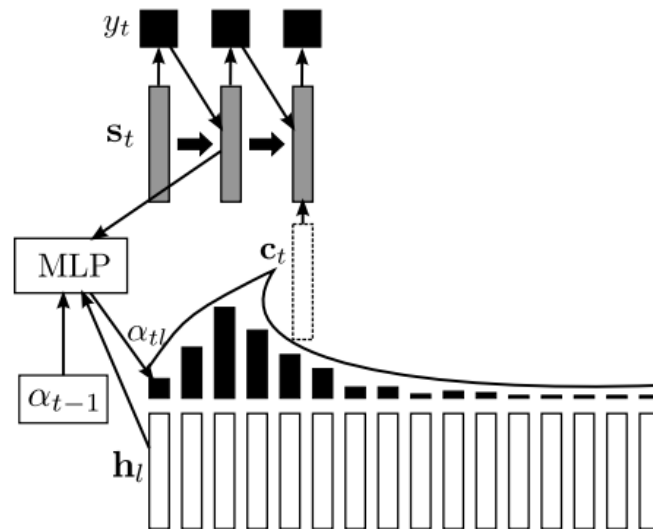- Maximum likelihood conditional language model given the aud $p(y_1) \prod_{t=2}^{T} p(y_t | y_1, \cdots, y_{t-1})$

Fig. 3. Schematic representation of the Attention-based Recurrent Sequence Generator. At each time step t, an MLP combines the hidden state s_{t-1} with all the input vectors h_l to compute the attention weights α_{tl}. Subsequently, the new hidden state s_t and prediction for output label y_t can be computed.

(Bahdanau *et al.* 2016)