End-to-End Speech Recognition using Deep LSTMs, CTC Training and WFST Decoding

@ MIT   Dec 07 2015

Yajie Miao

Carnegie Mellon University

Outline

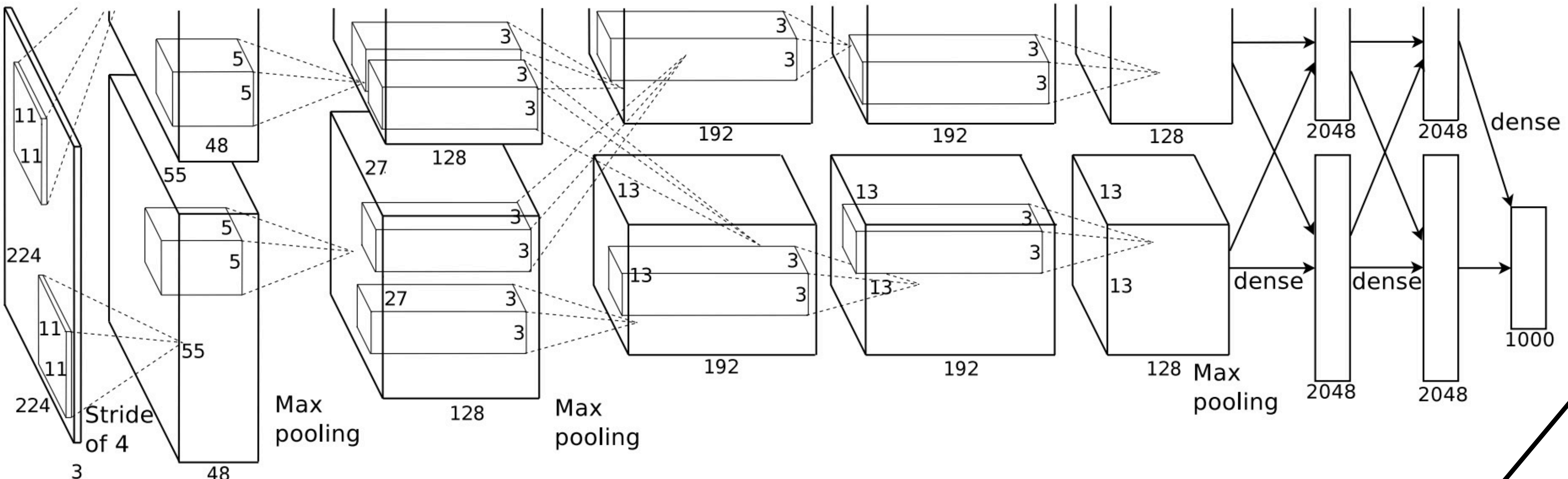
- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

Outline

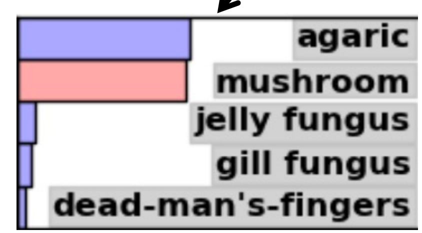
- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

Why End-to-End?

ImageNet Classification with Deep Convolutional Neural Networks. Krizhevsky et al.



End-to-End Image Classification

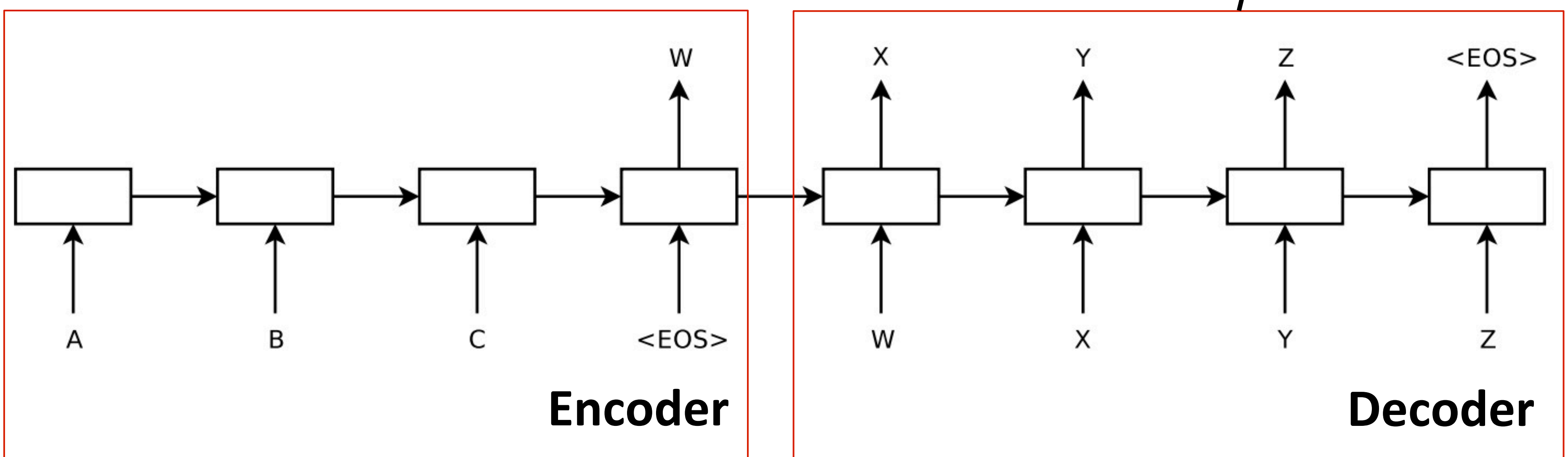


Why End-to-End?

Sequence to Sequence Learning with Neural Networks. Sutskever et al. 2014.

End-to-End Machine Translation

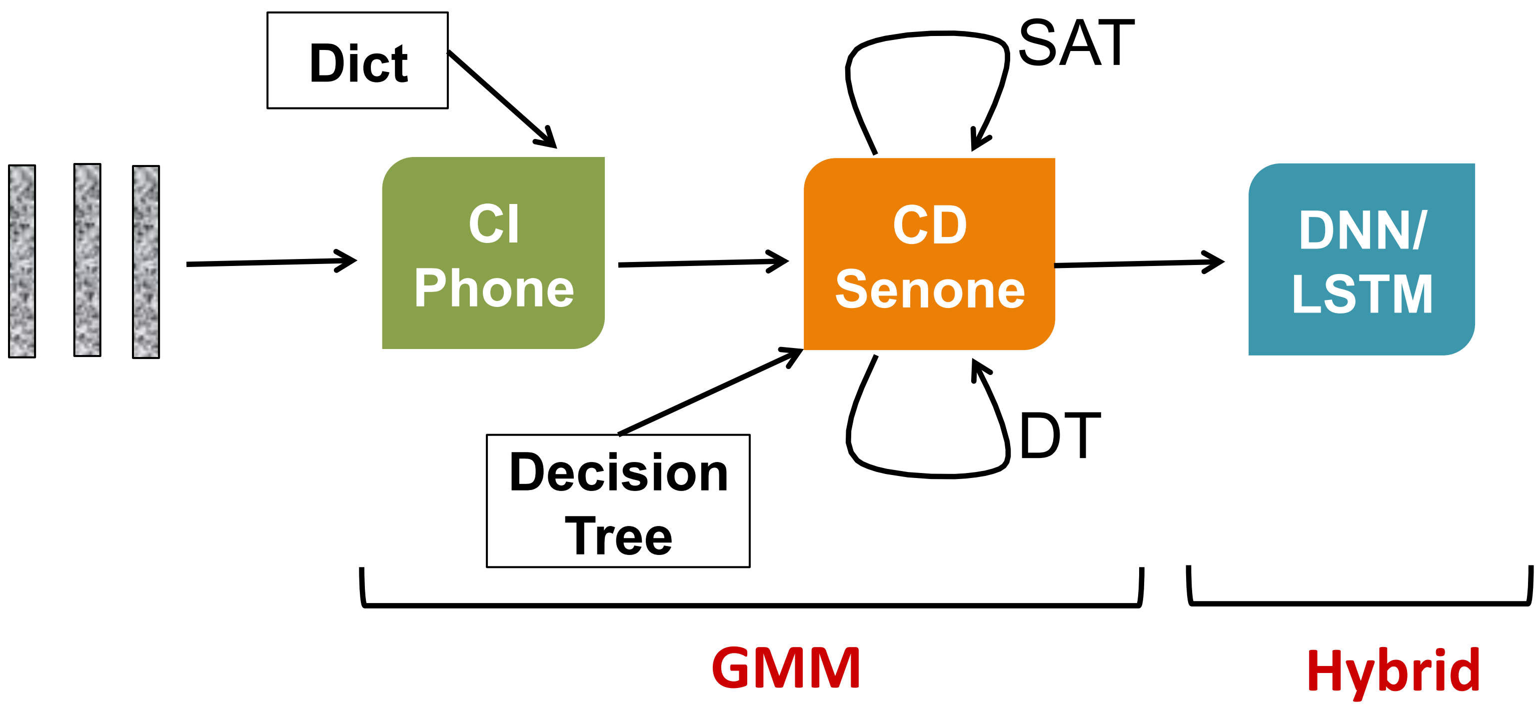
"玛丽爱上了约翰"



"Mary is in love with John"

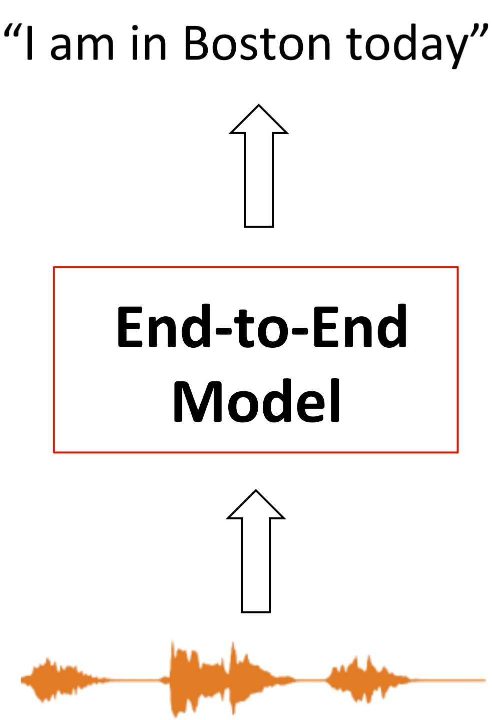
Complexity of ASR

- The HMM/GMM or HMM/DNN pipelines are highly complex
  - Multiple training stages: CI phone, CD senones, …
  - Various resources: dictionaries, decision trees, …
  - Many super-parameters: number of senones, number of Gaussians, …



End-to-End ASR!

- ASR is a sequence-to-sequence learning problem
- A simpler paradigm with a single model (and training stage)



Outline

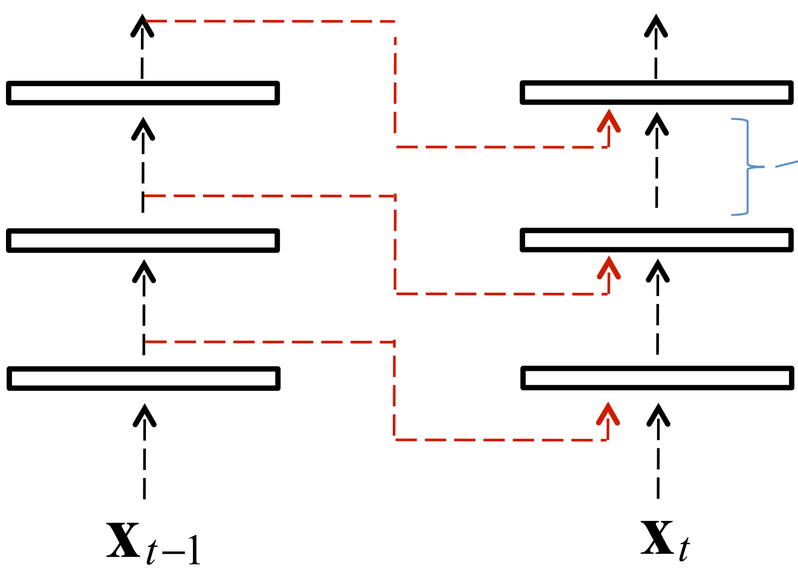
- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

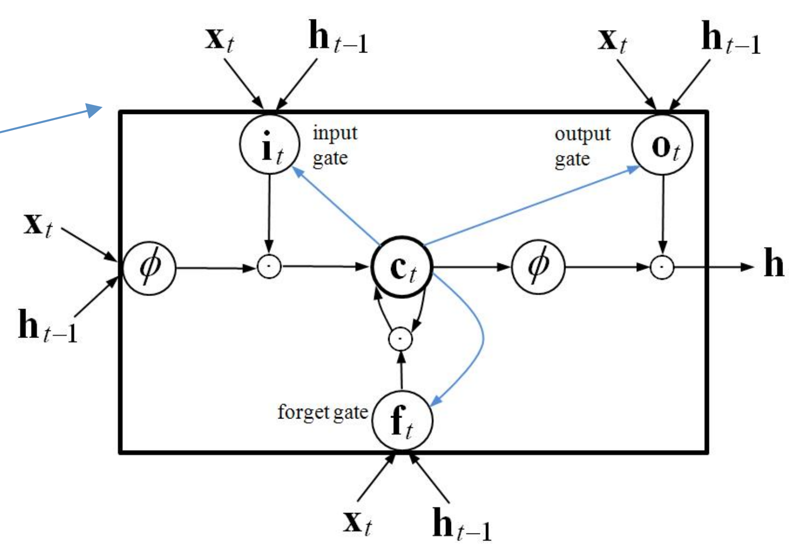
Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

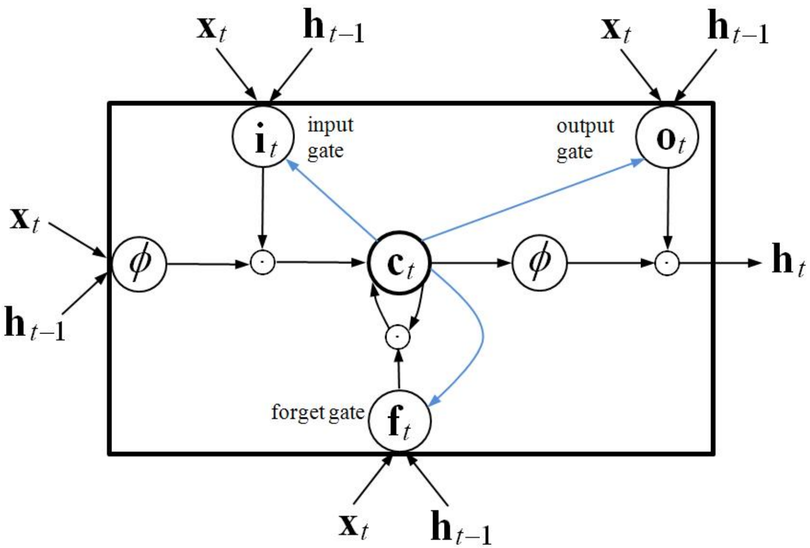
LSTM Models

- RNNs model temporal dependency across speech frames.
- Long short-term memory (LSTM) units.
  - Memory cells store the history information.
  - Various gates control the information flow inside the LSTM.
  - Advantageous in learning long-term temporal dependency.





LSTM Models



- LSTMs outperform DNNs in the hybrid approach [Sainath et al., Miao et al.]
- This is uni-directional LSTM, i.e., forward LSTM.

$$\mathbf{i}_t = \sigma(\mathbf{W}_{ix}\mathbf{x}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ic}\mathbf{c}_{t-1} + \mathbf{b}_i)$$ input gate

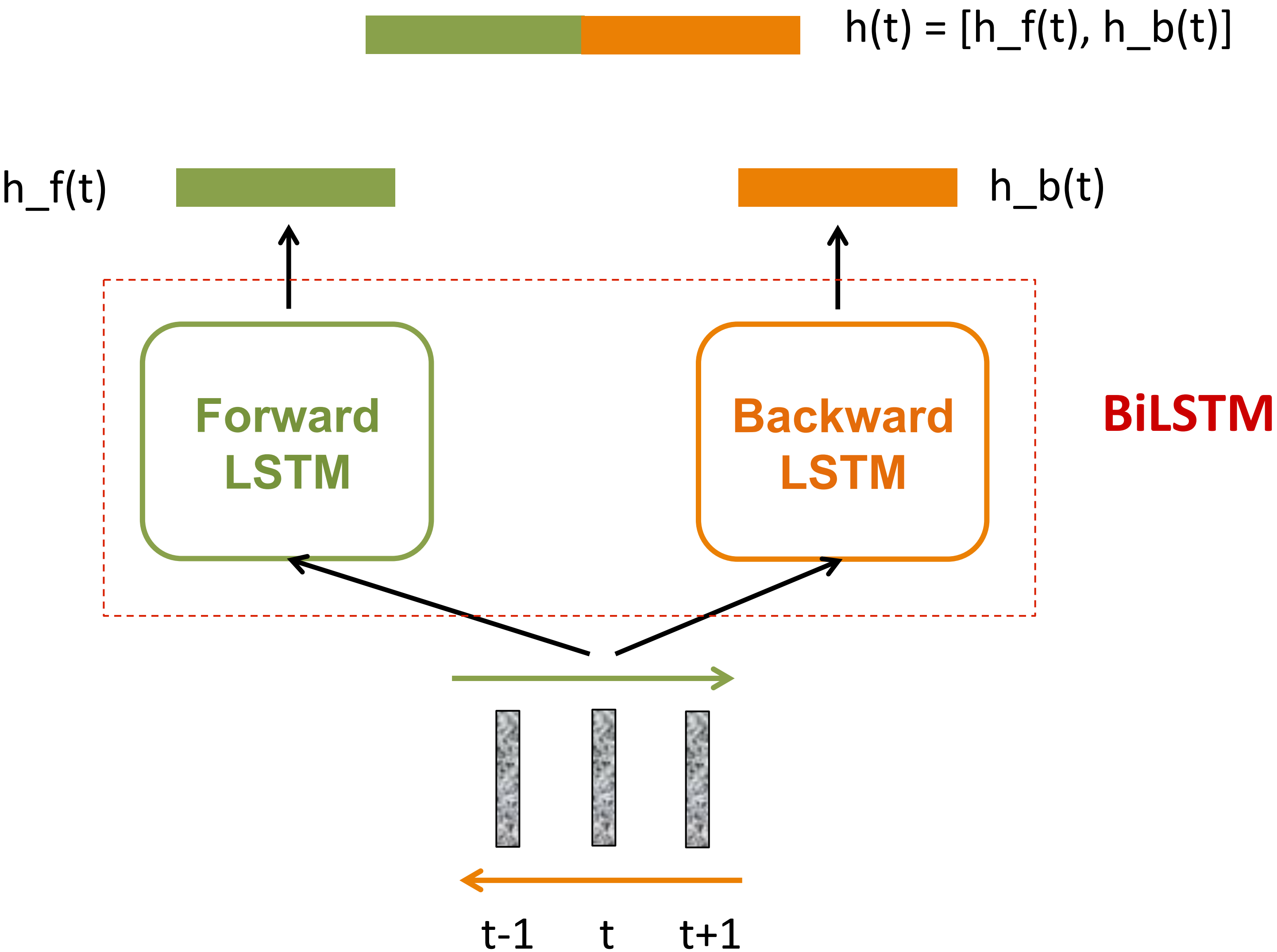
$$\mathbf{f}_t = \sigma(\mathbf{W}_{fx}\mathbf{x}_t + \mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fc}\mathbf{c}_{t-1} + \mathbf{b}_f)$$ forget gate

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \phi(\mathbf{W}_{cx}\mathbf{x}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c)$$ memory cell

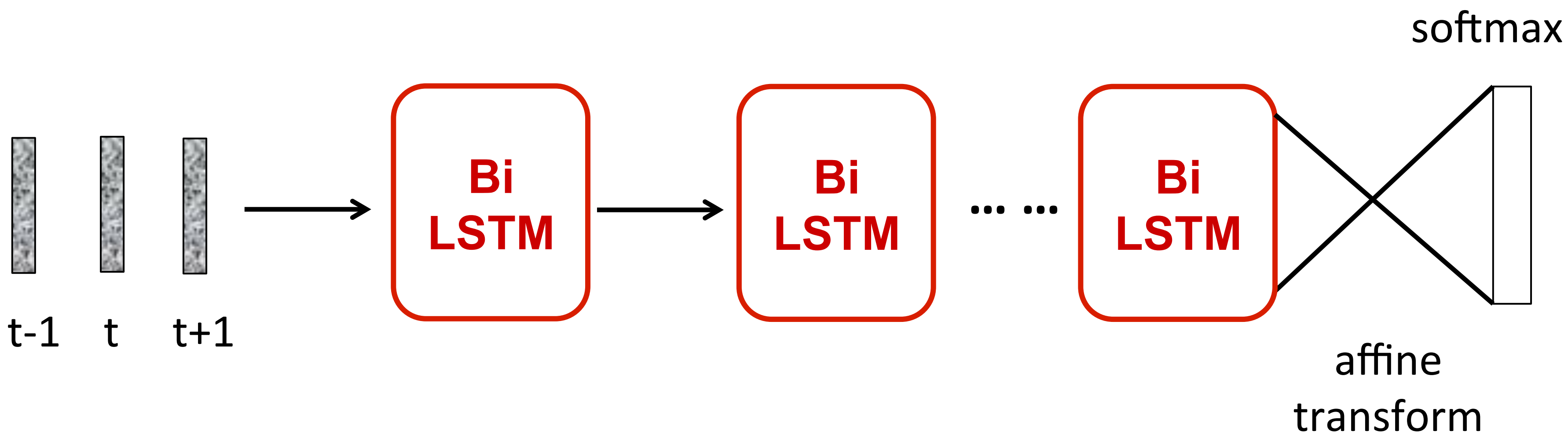
$$\mathbf{o}_t = \sigma(\mathbf{W}_{ox}\mathbf{x}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{oc}\mathbf{c}_t + \mathbf{b}_o)$$ output gate

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t)$$

Bi-directional LSTMs



Deep BiLSTM Model



Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

Connectionist Temporal Classification

- CTC is a sequence-to-sequence learning technique [Graves et al.]

$$L_{CTC} = \ln Pr(\mathbf{z} \mid \mathbf{X})$$

label sequence

"A B C"

observations

$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$

Connectionist Temporal Classification

- CTC is a sequence-to-sequence learning technique [Graves et al.]

$$L_{CTC} = \ln Pr(\mathbf{z} \mid \mathbf{X})$$

label sequence

"A B C"

observations

$$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$$

$$\Downarrow$$

$$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_T)$$

CTC Paths

- CTC is a sequence-to-sequence learning technique [Graves et al.]

$$L_{CTC} = \ln Pr(\mathbf{z} \mid \mathbf{X})$$

label sequence

"A B C"

$\boxed{\text{CTC Path}}$

observations

$$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$$

$$\Downarrow$$

$$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_T)$$

- CTC paths bridge frame-level labels with the label sequence
  - A CTC path is a sequence of labels on the frame level $\mathbf{p} = [p_1, \ldots, p_T]$
  - The likelihood of a CTC path is decomposed onto the frames:

$$Pr(\mathbf{p} \mid \mathbf{X}) = \prod_{t=1}^{T} y_t^{p_t}$$

CTC Paths

- CTC paths differ from labels sequences in that:
  - Add the blank as an additional label, meaning no (actual) labels are emitted
  - Allow repetitions of non-blank labels

$$\left.\begin{array}{ccccccc} A & A & \varnothing & \varnothing & B & C & \varnothing \\ \varnothing & A & A & B & \varnothing & C & C \\ \varnothing & \varnothing & \varnothing & A & B & C & \varnothing \end{array}\right] \underset{\text{expand}}{\overset{\text{collapse}}{\rightleftarrows}} A\ B\ C$$

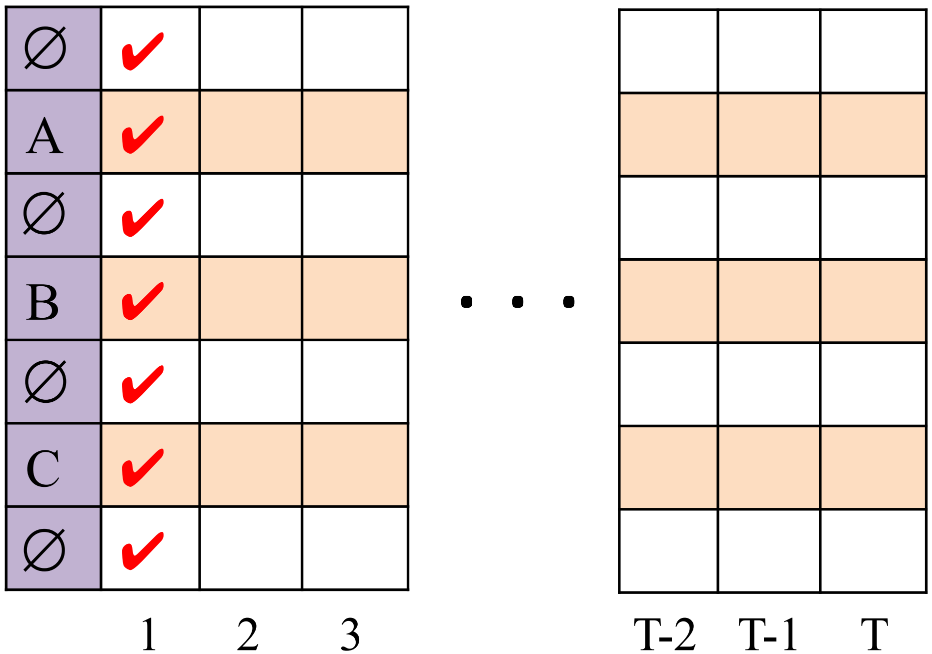
- Many-to-one mapping from CTC paths $\mathbf{\Phi}(\mathbf{z})$ to the label sequence $\mathbf{z}$

$$Pr(\mathbf{z} \mid \mathbf{X}) = \sum_{p \in \Phi(\mathbf{z})} Pr(\mathbf{p} \mid \mathbf{X})$$

Computationally Intractable !!

Forward-Backward Algorithm

expanded labels l

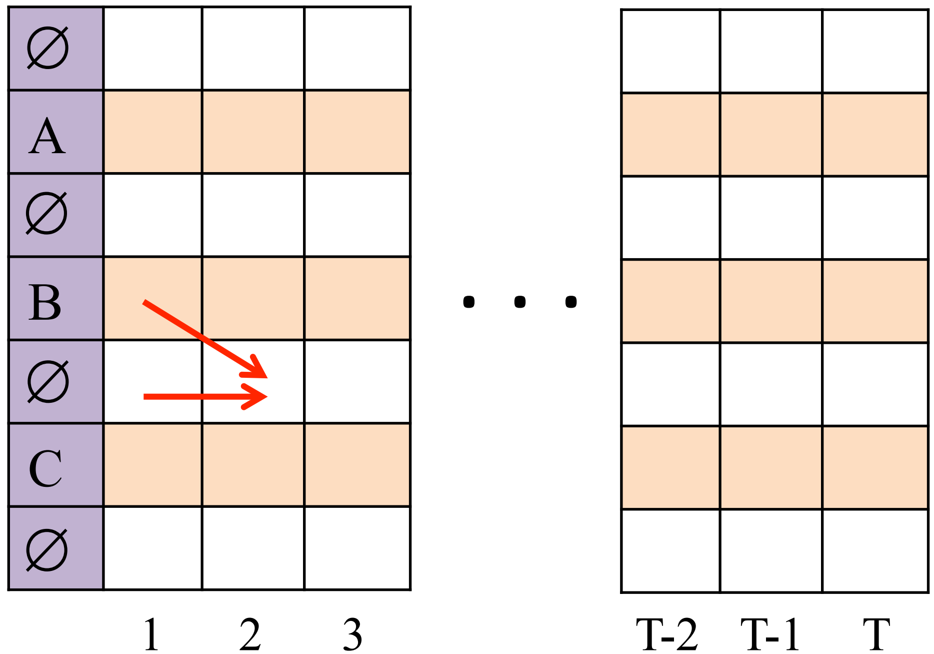


$\alpha_t(s)$

summed probability of all the CTC paths ending at t with s

$$\alpha_1(\varnothing) = y^1_\varnothing \qquad \alpha_1(A) = y^1_A \qquad \alpha_1(s) = 0,\ \forall s > 2$$

Forward Computation



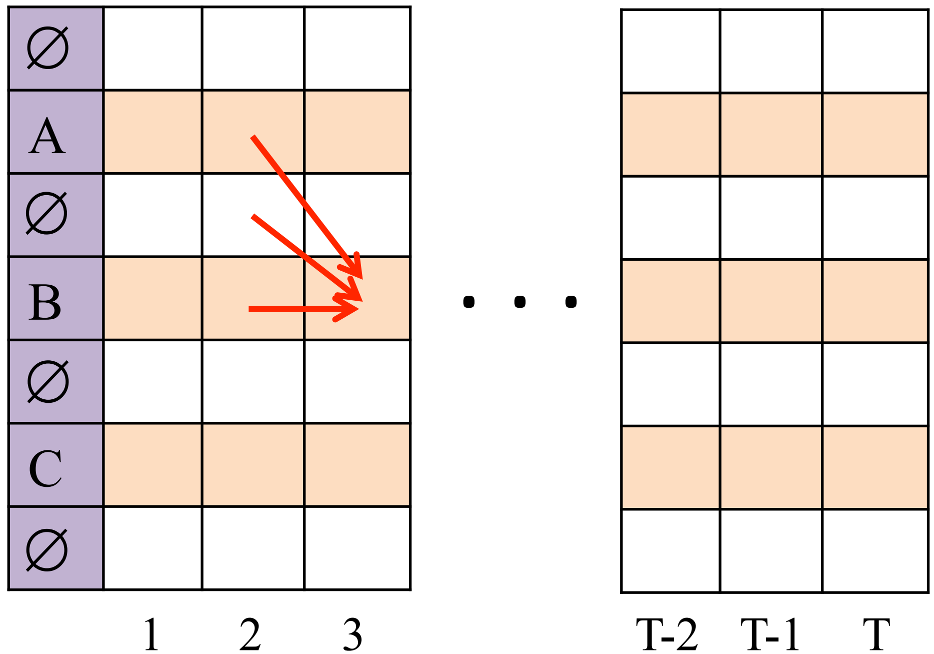
expanded labels
l

$\alpha_t(s)$
summed probability of all the CTC paths ending at t with s

$$\alpha_t(s) = \begin{cases} y^t_{l_s}\left(\alpha_{t-1}(s) + \alpha_{t-1}(s-1)\right) & \text{if } l_s = \varnothing \text{ or } l_s = l_{s-2} \\ \end{cases}$$

Forward Computation

expanded labels l



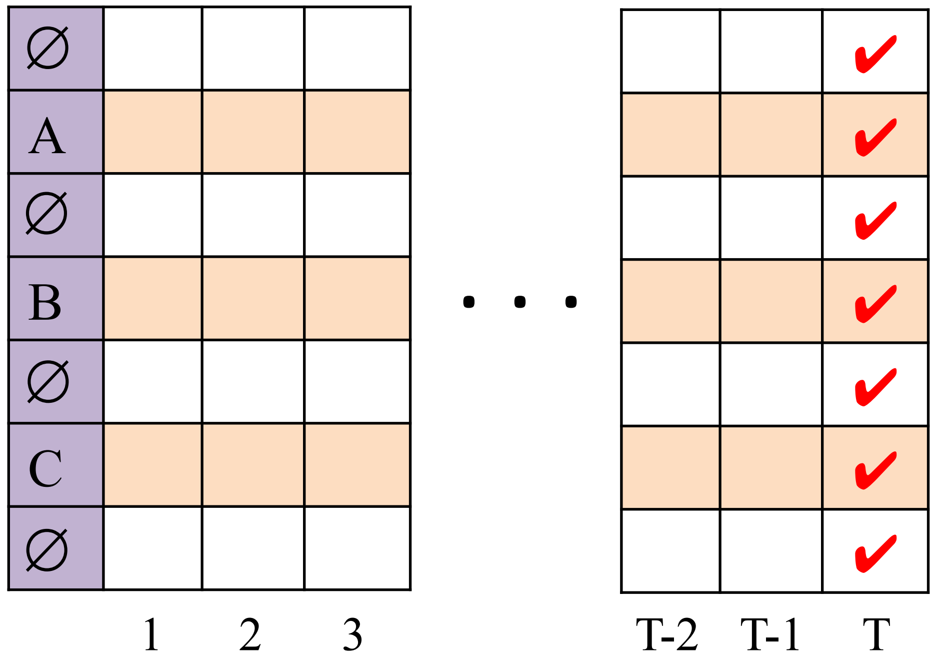
$\alpha_t(s)$

summed probability of all the CTC paths ending at t with s

$$\alpha_t(s) = \begin{cases} y^t_{l_s}\left(\alpha_{t-1}(s) + \alpha_{t-1}(s-1)\right) & \text{if } l_s = \varnothing \text{ or } l_s = l_{s-2} \\ y^t_{l_s}\left(\alpha_{t-1}(s) + \alpha_{t-1}(s-1) + \alpha_{t-1}(s-2)\right) & \text{otherwise} \end{cases}$$

Backward Computation

expanded labels
l

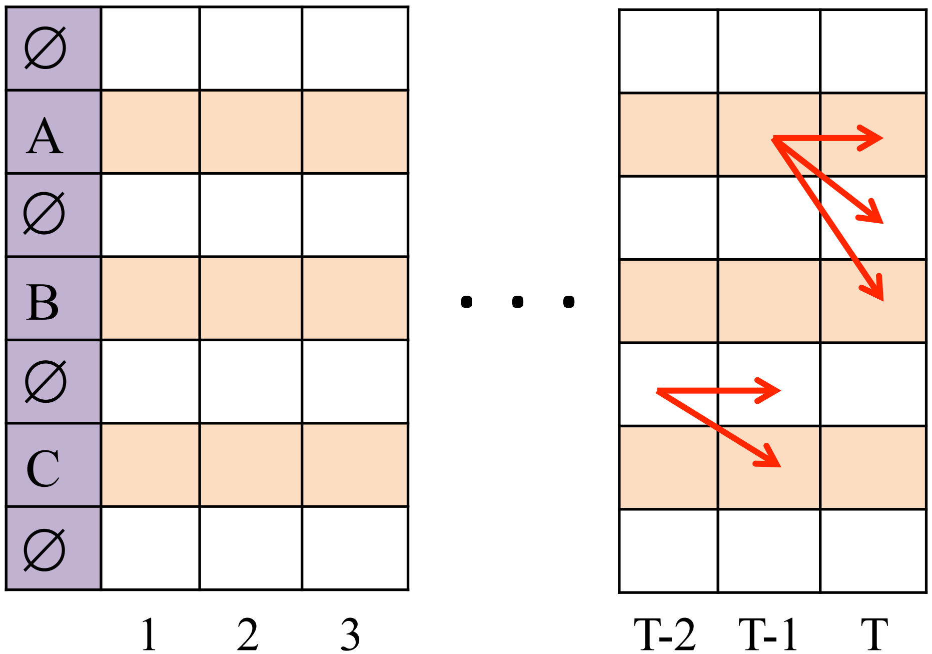


$\beta_t(s)$

summed probability of all the CTC paths starting at t with s

$$\beta_T(\varnothing) = y_{\varnothing}^{T} \qquad \beta_T(C) = y_{C}^{T} \qquad \beta_T(s) = 0,\ \forall s < |l| - 1$$

Backward Computation



expanded labels
l

$\beta_t(s)$

summed probability of all CTC paths starting at t with s

$$\beta_t(s) = \begin{cases} y^t_{l_s}\left(\beta_{t+1}(s) + \beta_{t+1}(s+1)\right) & \text{if } l_s = \varnothing \text{ or } l_s = l_{s+2} \\ y^t_{l_s}\left(\beta_{t+1}(s) + \beta_{t+1}(s+1) + \beta_{t+1}(s+2)\right) & \text{otherwise} \end{cases}$$

CTC Training

- Evaluation of the objective $Pr(\mathbf{z} \mid \mathbf{X})$

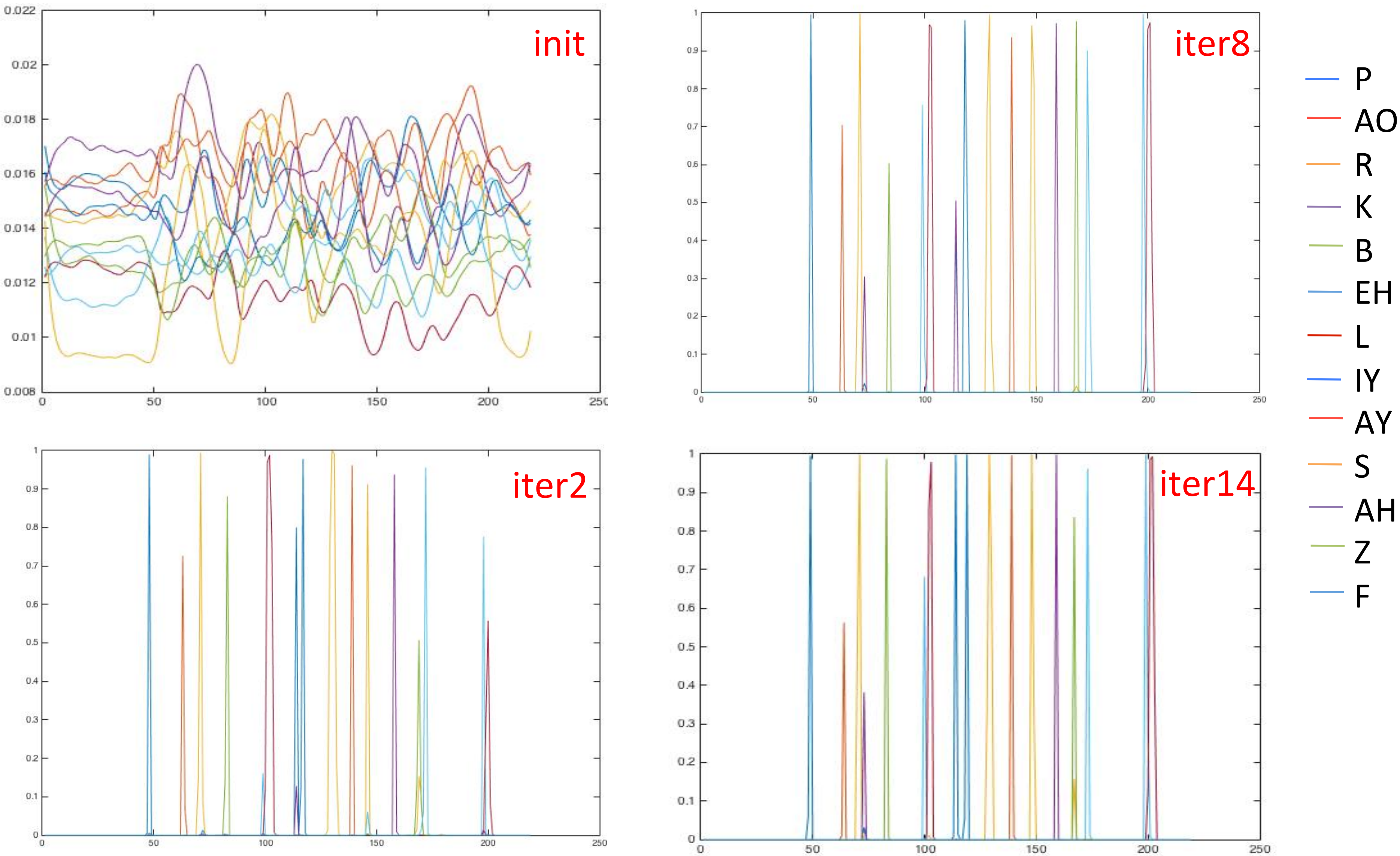
$$Pr(\mathbf{z} \mid \mathbf{X}) = \sum_{s=1}^{|\mathbf{l}|} \alpha_t(s)\beta_t(s) \qquad L_{CTC} = \ln Pr(\mathbf{z} \mid \mathbf{X})$$

- Gradients w.r.t. the pre-softmax network outputs:

CTC $$\frac{\partial L_{CTC}}{\partial a_k^t} = y_k^t - \boxed{\frac{1}{Pr(\mathbf{z} \mid \mathbf{X})} \sum_{s \in B(\mathbf{l}, k)} \alpha_t(s)\beta_t(s)}$$ soft labels

CE $$\frac{\partial L_{CE}}{\partial a_k^t} = y_k^t - \boxed{g_k^t}$$ hard labels

What Happens during CTC Training?

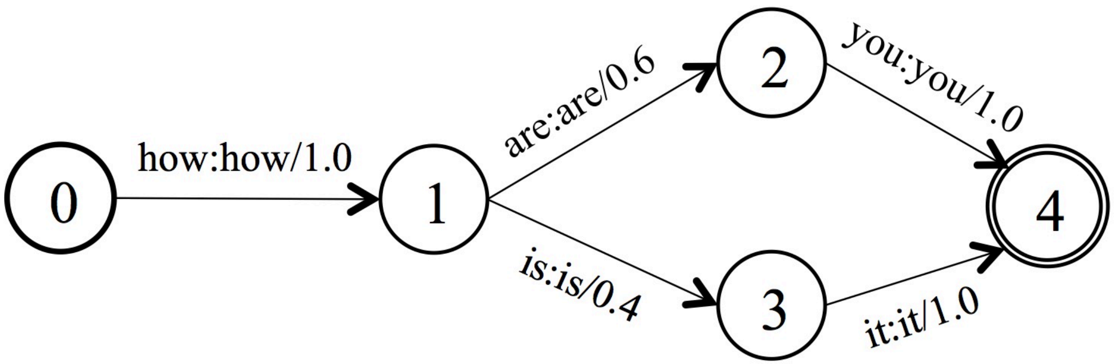


Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

CTC Decoding

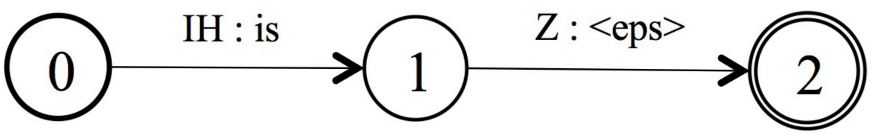
- Difficulty of CTC decoding
  - Previous work proposed beam search for CTC [Graves et al, Hannun et al.]
  - Incorporating word language models efficiently was difficult
  - It was challenging to deal with the behaviors of blanks
- WFST-based Decoding
  - 3 WFSTs encode 3 components required in decoding
- The language model WFST G



how are you
how is it

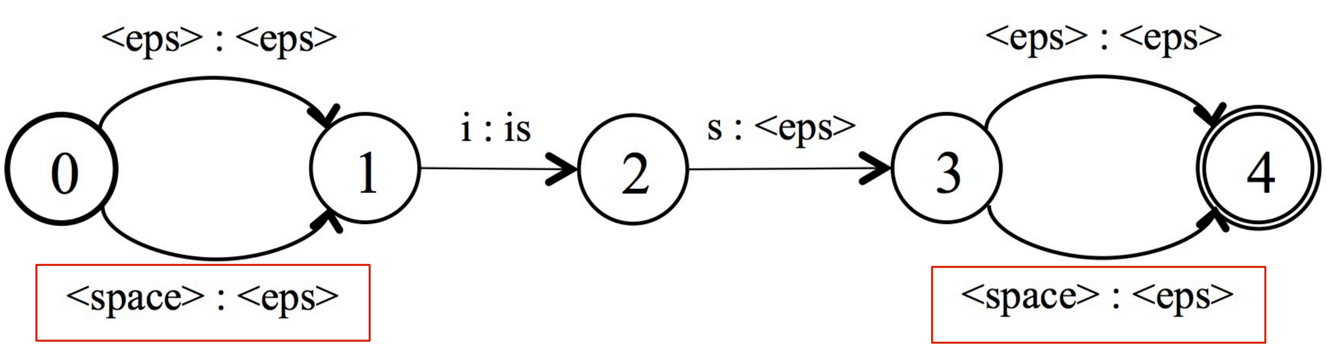
Lexicon -- L

- Maps sequences of lexicon units to words.
- Phonemes as CTC labels: the standard dictionary WFST



is IH Z

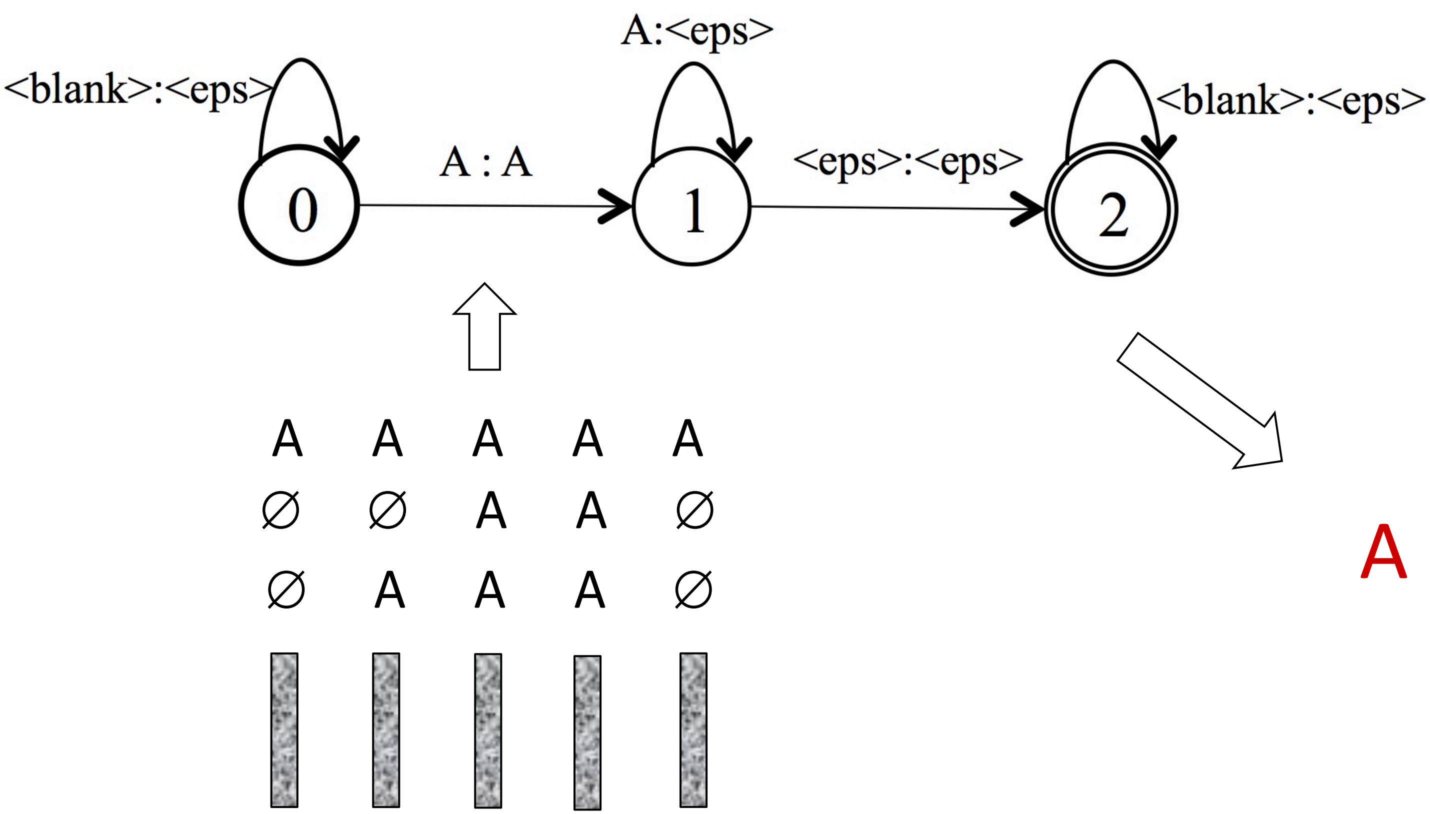
- Characters as CTC labels
  - Contains word spellings, easy to include OOV words
  - The space <space> between each pair of words is taken as a label
  - Allows <space> to appear optionally at the beginning and end of the word



is i s

Token -- T

- Maps a (segment of) CTC path to a lexicon unit (phonemes or characters)
- Allows occurrences of blanks and repetitions of non-blank labels



Search Graph & Posterior Scaling

- The 3 WFSTs are composed into a search graph

$$S = T \circ min(det(L \circ G))$$

$\circ$ – composition det – determinization min – minimization

- The search graph encodes the mapping from a sequence of frame-level CTC labels to a sequence of words.

- During decoding, scale the label posteriors with their priors

$$p(\mathbf{x}_t \mid k) \propto p(k \mid \mathbf{x}_t) / p(k)$$

where the prior $p(k)$ is estimated from the expanded label sequences ($\varnothing$ A $\varnothing$ B $\varnothing$ C $\varnothing$) from the training set by simple counting.

Eesen Recipe - Switchboard

https://github.com/yajiemiao/eesen

Data Prep and FST Composition

```
# Use the same datap prepatation script from Kaldi
local/swbd1_data_prep.sh $swbd  || exit 1;

# Construct the phoneme-based lexicon
local/swbd1_prepare_phn_dict.sh || exit 1;

# Compile the lexicon and token FSTs
utils/ctc_compile_dict_token.sh data/local/dict_phn data/local/lang_phn_tmp data/lang_phn || exit 1;

# Train and compile LMs.
local/swbd1_train_lms.sh data/local/train/text data/local/dict_phn/lexicon.txt data/local/lm $fisher_dirs

# Compile the language-model FST and the final decoding graph TLG.fst
local/swbd1_decode_graph.sh data/lang_phn data/local/dict_phn/lexicon.txt || exit 1;

# Data preparation for the eval2000 set
local/eval2000_data_prep.sh $eval2000_dirs
```

Feature Generation

```
# Generate the fbank features; by default 40-dimensional fbanks on each frame
steps/make_fbank.sh --cmd "$train_cmd" --nj 32 data/train exp/make_fbank/train $fbankdir || exit 1;
utils/fix_data_dir.sh data/train || exit;
steps/compute_cmvn_stats.sh data/train exp/make_fbank/train $fbankdir || exit 1;

steps/make_fbank.sh --cmd "$train_cmd" --nj 10 data/eval2000 exp/make_fbank/eval2000 $fbankdir || exit 1;
utils/fix_data_dir.sh data/eval2000 || exit;
steps/compute_cmvn_stats.sh data/eval2000 exp/make_fbank/eval2000 $fbankdir || exit 1;
```

Eesen Recipe - Switchboard

https://github.com/yajiemiao/eesen

Model Training

```
# Specify network structure and generate the network topology
input_feat_dim=120   # dimension of the input features
lstm_layer_num=4     # number of LSTM layers
lstm_cell_dim=320    # number of memory cells in every LSTM layer

dir=exp_110h/train_phn_l${lstm_layer_num}_c${lstm_cell_dim}
mkdir -p $dir

# Output the network topology
utils/model_topo.py --input-feat-dim $input_feat_dim --lstm-layer-num $lstm_layer_num \
  --lstm-cell-dim $lstm_cell_dim --target-num $target_num \
  --fgate-bias-init 1.0 > $dir/nnet.proto || exit 1;

# Label sequences; simply convert words into their label indices
utils/prep_ctc_trans.py data/lang_phn/lexicon_numbers.txt data/train_100k_nodup/text "<unk>" | gzip -c -
utils/prep_ctc_trans.py data/lang_phn/lexicon_numbers.txt data/train_dev/text "<unk>" | gzip -c - > $dir/

# Train the network with CTC. Refer to the script for details about the arguments
steps/train_ctc_parallel.sh --add-deltas true --num-sequence 10 --frame-num-limit 25000 \
  --learn-rate 0.00004 --report-step 1000 --halving-after-epoch 12 \
  data/train_100k_nodup data/train_dev $dir || exit 1;
```

Decoding

```
# decoding
for lm_suffix in sw1_tg sw1_fsh_tgpr; do
  steps/decode_ctc_lat.sh --cmd "$decode_cmd" --nj 20 --beam 17.0 --lattice_beam 8.0 --max-active 5000 --acwt
    data/lang_phn_${lm_suffix} data/eval2000 $dir/decode_eval2000_${lm_suffix} || exit 1;
done
```

Eesen Recipe - Switchboard

https://github.com/yajiemiao/eesen

```
# Specify network structure and generate the network topology
input_feat_dim=120    # dimension of the input features
lstm_layer_num=4      #
lstm_cell_dim=320     #

dir=exp_110h/train_phn_
mkdir -p $dir

# Output the network to
utils/model_topo.py --i                                                        er_num \
  --lstm-cell-dim $lstm
  --fgate-bias-init 1.0

# Label sequences; simp
utils/prep_ctc_trans.py                                                        /text "<unk>" | gzip -c -
utils/prep_ctc_trans.py                                                        unk>" | gzip -c - > $dir/

# Train the network wit                                                        s
steps/train_ctc_paralle                                                        t 25000 \
  --learn-rate 0.00004
  data/train_100k_nodup data/train_dev $dir || exit 1;
```

odel Training

NO

- HMM
- GMM
- Phonetic decision trees
- Multiple training stages
- Dictionaries, if characters are CTC labels
- … …

Decoding

```
# decoding
for lm_suffix in sw1_tg sw1_fsh_tgpr; do
  steps/decode_ctc_lat.sh --cmd "$decode_cmd" --nj 20 --beam 17.0 --lattice_beam 8.0 --max-active 5000 --acwt
    data/lang_phn_${lm_suffix} data/eval2000 $dir/decode_eval2000_${lm_suffix} || exit 1;
done
```

Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

Results on WSJ

- Experimental Setup
  - 4 bi-directional LSTM layers, each with 640 memory cells
  - 40-dimemsional log-mel filterbank features, plus Δ and $\Delta\Delta$
  - Training utterances are sorted by their lengths, and 10 utterances are processed in a batch each time
  - WERs on the eval92 testing set
- Phoneme-based Systems
  - The CMU dictionary as the lexicon
  - 72 labels including phonemes, noise marks and the blank
- Character-based Systems
  - 59 labels including letters, digits, punctuation marks and the blank.

Results on WSJ

	Models	Vocabulary	LM	WER%
Phone	CTC	Original	NIST	7.87
	Hybrid DNN	Original	NIST	7.14

Results on WSJ

	Models	Vocabulary	LM	WER%
Phone	CTC	Original	NIST	7.87
	Hybrid DNN	Original	NIST	7.14
Char	CTC	Original	NIST	9.07
	CTC	Expanded	Retrained	7.34
	Graves et al.	Expanded	Retrained	8.7
	Hannun et al.	Original	Unknown	14.1

Results on Switchboard

- Experimental Setup
  - 300 hours of training speech; tested on the SWBD part of Hub5'00
  - 5 bi-directional LSTM layers, each with 640 memory cells
  - CTC labels: 46 phonemes (including the blank)
- Initialization of Forget-gate Bias
  - Initializing the forget gate bias to a larger value helps LSTM learn long-term dependency [Jozefowicz et al.]
  - The bias of the forget gates is initialized to 1.0

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{ix}\mathbf{x}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ic}\mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{fx}\mathbf{x}_t + \mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fc}\mathbf{c}_{t-1} + \boxed{\mathbf{b}_f)} \quad = 1.0 \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \phi(\mathbf{W}_{cx}\mathbf{x}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{ox}\mathbf{x}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{oc}\mathbf{c}_t + \mathbf{b}_o)
\end{aligned}$$

Results on Switchboard

Models	FG Bias	WER%
CTC	0	15.7
CTC	1.0	15.0

Results on Switchboard

Models	#Model Param	WER%
CTC	11M	**15.0**
Hybrid DNN	40M	16.9
Hybrid LSTM	12M	15.8

For fairness, all the systems use the filterbank features

Results on Switchboard

300-Hour Training

Models	#Model Param	WER%
CTC	11M	**15.0**
Hybrid DNN	40M	16.9
Hybrid LSTM	12M	15.8

110-Hour Training

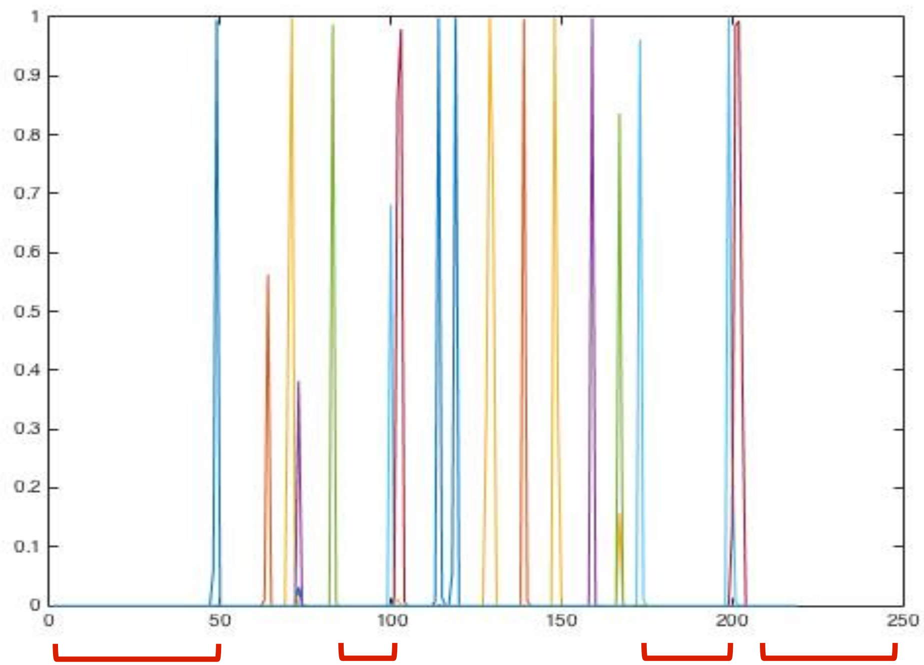
Models	#Model Param	WER%
CTC	8M	19.9
Hybrid DNN	12M	20.2
Hybrid LSTM	8M	**19.2**

Decoding Efficiency

Models	Decoding Graph	Graph Size	RTF*
CTC	TLG	123M	0.71
Hybrid DNN	HCLG	216M	1.43
Hybrid LSTM	HCLG	216M	1.12

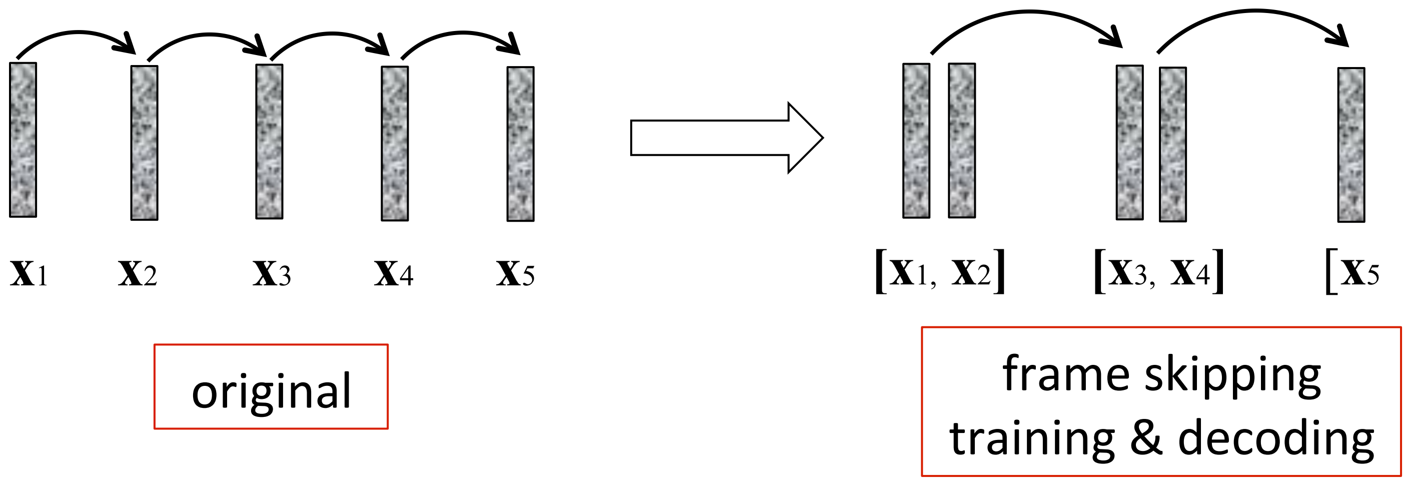
* RTF – Real time factor

Frame Skipping



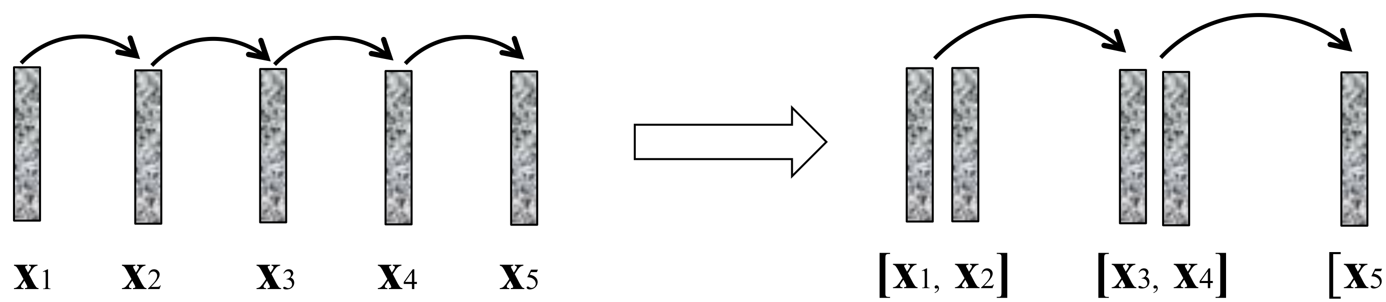
blanks

Frame Skipping



#Frames Skipped	Decoding RTF	WER%
0	0.71	15.0
1	0.38	15.8
2	0.25	16.5

Frame Skipping



original

frame skipping
training & decoding

	#Frames Skipped	Decoding RTF	WER%
	0	0.71	15.0
	1	0.38	15.8
< 100Hrs →	2	0.25	16.5
110Hrs →	0	----	19.9

Results on HKUST Mandarin

- Experimental Setup
  - Chinese Mandarin conversational telephone speech [Liu et al.]
  - 175 hours of training
  - CTC labels: 3600+ Chinese characters

Models	Features	CER%
Hybrid DNN	FBank	39.42
CTC	FBank	39.70
CTC	FBank+Pitch	38.67

Results on Multilingual CTC

- Experimental Setup
  - BABEL multilingual corpora
  - Around 80 hours of training speech per language
  - Features: filterbank + pitch

Models	Training Hrs	#Labels	CTC WER%
Tagalog	84.5	48	51.5
Turkish	77.2	51	54.5
Pashto	78.4	54	56.6

Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions

Conclusions

- Conclusions
  - We have presented a complete end-to-end ASR framework
  - CTC models achieve comparable performance to the hybrid approach
  - There are still a lot of unknowns about CTC
- Open Questions
  - How can we perform Keyword Search (KWS) over CTC models?
  - How can we add context-dependency to CTC labels?
  - How can we estimate i-vectors [Dehak et al.]?
  - How can we perform pre-training of the deep BiLSTM models?
  - … …

References

- Y. Miao, M. Gowayyed, and F. Metze, “EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding,” in *Proc. ASRU.* IEEE, 2015.
- T. N. Sainath, O. Vinyals, A. Senior, H. Sak, “Convolutional, long short-term memory, fully connected deep neural networks,” in *Proc. ICASSP*. IEEE, 2015.
- R. Jozefowicz, W. Zaremba, and I. Sutskever, “An empirical exploration of recurrent network architectures,” in *Proc ICML*, 2015.
- A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks,” in *Proc. ICML*, 2014.
- A. Y. Hannun, A. L. Maas, D. Jurafsky, and A. Y. Ng, “First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNs,” *arXiv preprint arXiv:1408.2873*, 2014.
- Y. Miao and F. Metze, “On speaker adaptation of long short-term memory recurrent neural networks,” in *Proc. INTERSPEECH*. ISCA, 2015.
- Y. Liu, P. Fung, Y. Yang, C. Cieri, S. Huang, and D. Graff, “HKUST/MTS: a very large scale Mandarin telephone speech corpus,” in *Chinese Spoken Language Processing*, 2006.
- H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition,” *arXiv preprint arXiv:1507.06947*, 2015.
- N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, “Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification,” in *Proc. Interspeech,* 2009.

Questions?

@ MIT   Dec 07 2015

Yajie Miao
Carnegie Mellon University