CS11-737 Multilingual NLP

Automatic Speech Recognition

Lei Li

https://lileicc.github.io/course/11737mnlp23fa/

Carnegie Mellon University
Language Technologies Institute

Automatic Speech Recognition (ASR)

Find the text y to maximize the conditional probability

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, p(y \mid x; \theta)$$

The same formulation as translation

Measuring the Performance: WER

- Word error rate: edit distance between reference and candidate

$$WER = \frac{\text{Inserttions} + \text{Subs} + \text{Deletions}}{totalwordsinreference}$$

Ref: pittsburgh is a city of bridge

Candidate: pitts berger is city off bridge

WER = (1+2+1) / 6 = 67%

Overview of ASR Approaches

- Statistical ASR:
    - based on noisy channel model (similar to IBM MT model)

$$P(Y \mid X) = \frac{\sum_Z P(X \mid Z) P(Z \mid Y) P(Y)}{P(X)}$$

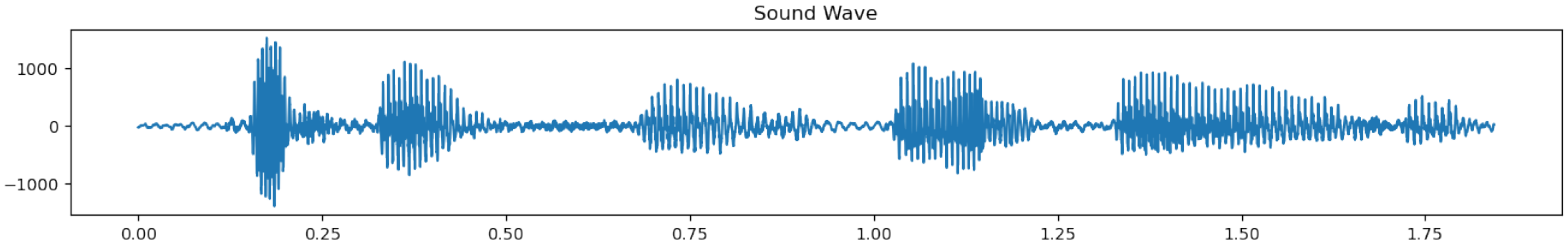
phoneme2audio

g2p

language model

- End-to-end Neural ASR:
    - directly learn mapping from input audio to output text

Statistical ASR in one minute



Feature extraction

MFCC

Gaussian Acoustic Model

Phone probability

HMM

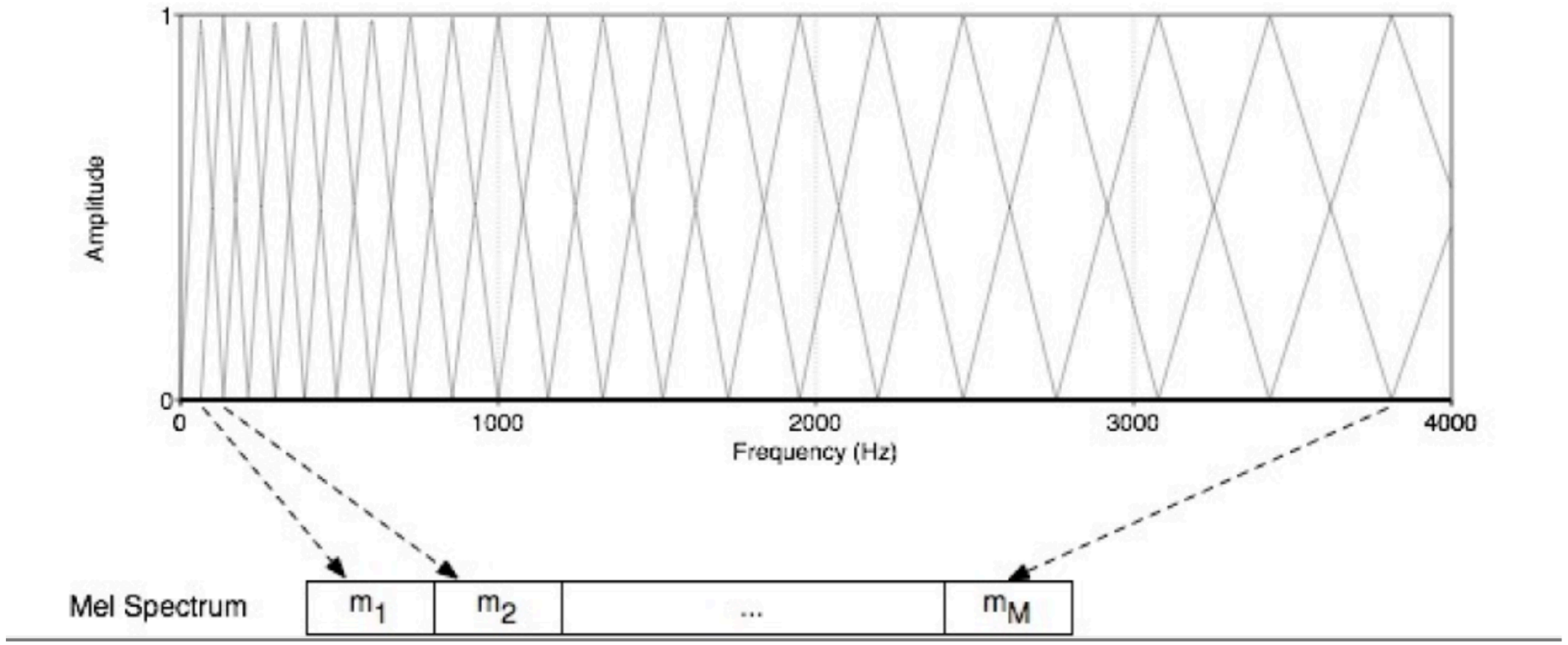
Language Model

Decoding

Feature Extraction for Speech

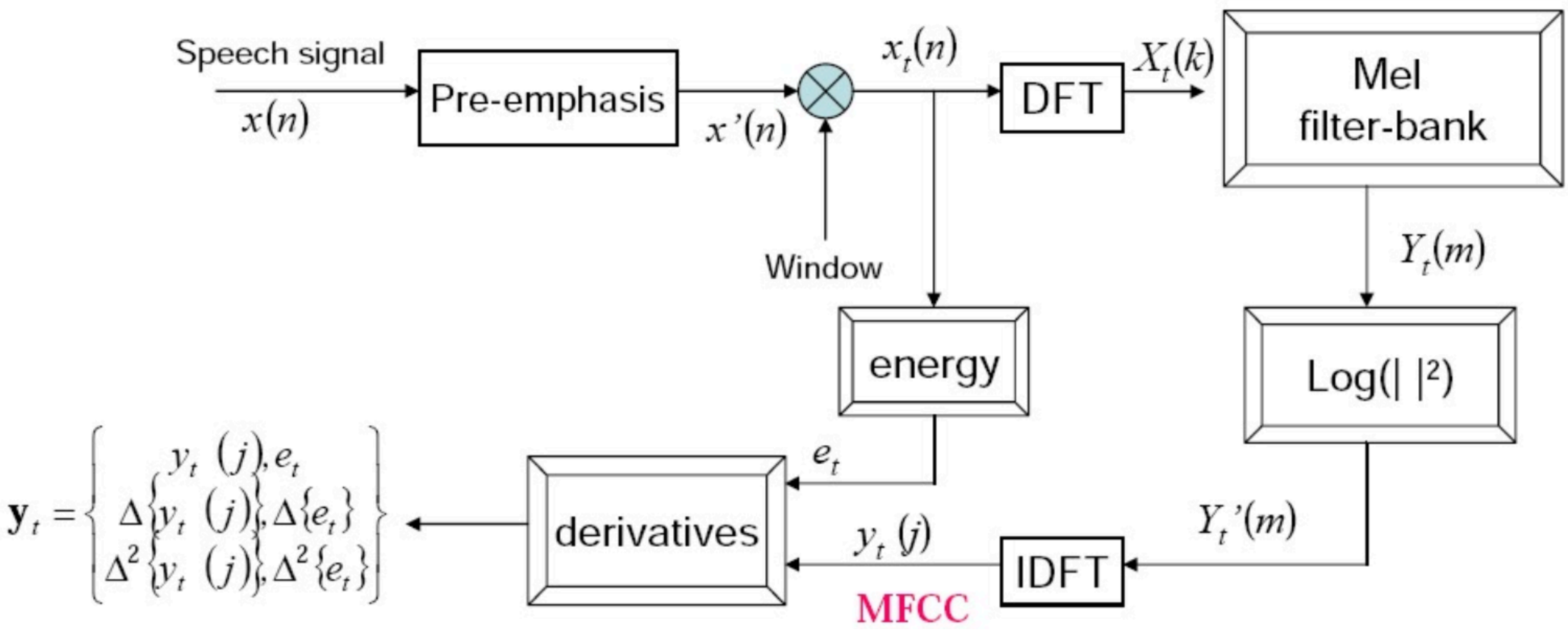
- Human hearing is not equally sensitive to all frequency bands

- Mel Filter Bank:
  - roughly evenly spaced below 1kHz
  - logarithmic scale above 1kHz

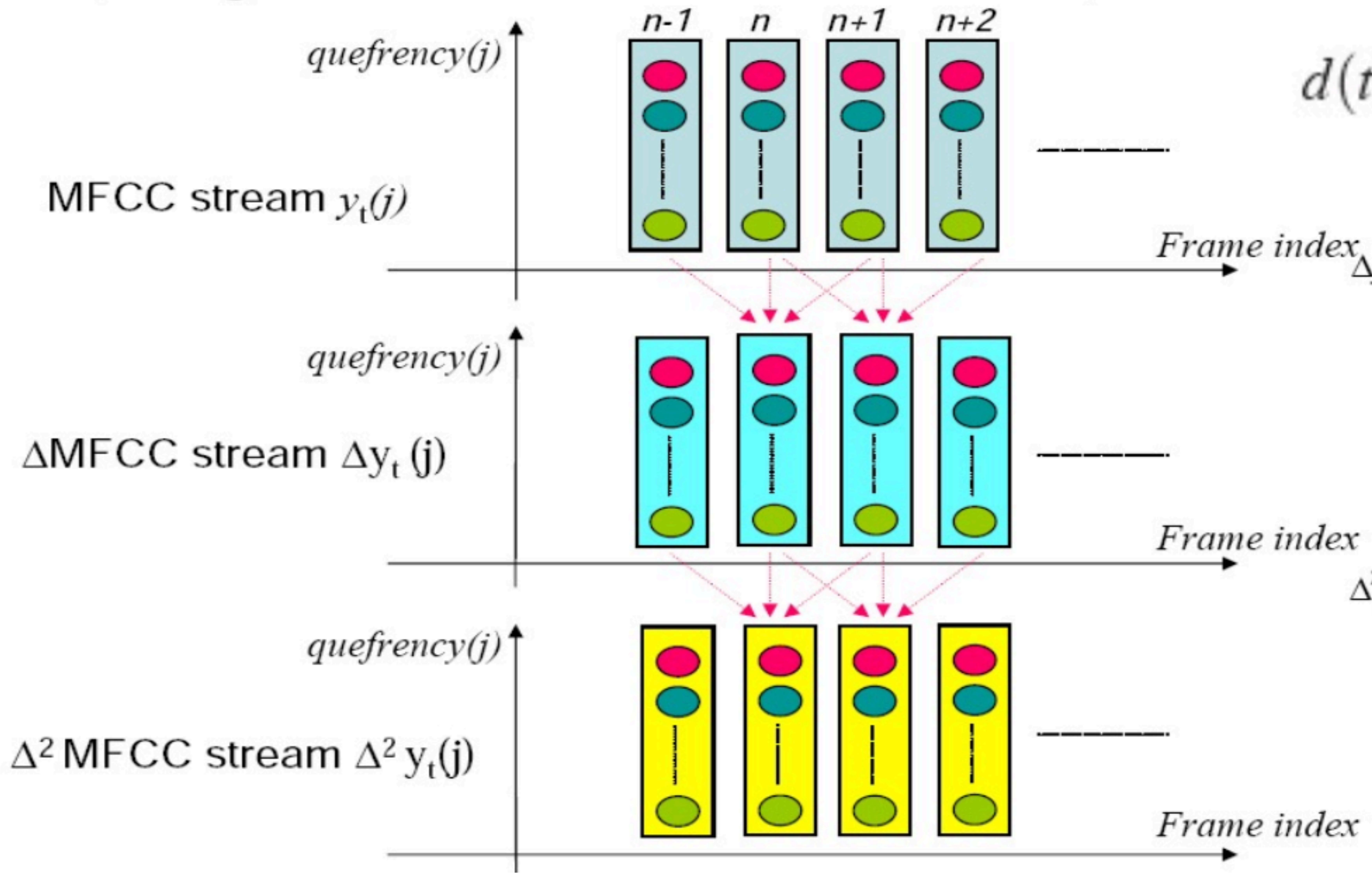


Mel-Frequency Cepstral Coefficient (MFCC)

- Most widely used feature representation in ASR



Higher-order information



$$d(t) = \frac{c(t+1) - c(t-1)}{2}$$

$$\Delta y_t(j) = \frac{\sum_{m=-p}^{p} m \bullet y_{t-m}(j)}{\sum_{m=-p}^{p} m^2}$$

$$\Delta^2 y_t(j) = \frac{\sum_{m=-p}^{p} m \bullet \Delta y_{t-m}(j)}{\sum_{m=-p}^{p} m^2}$$

MFCC

- Window size: 25ms
  Window shift: 10ms
  Pre-emphasis coefficient: 0.97

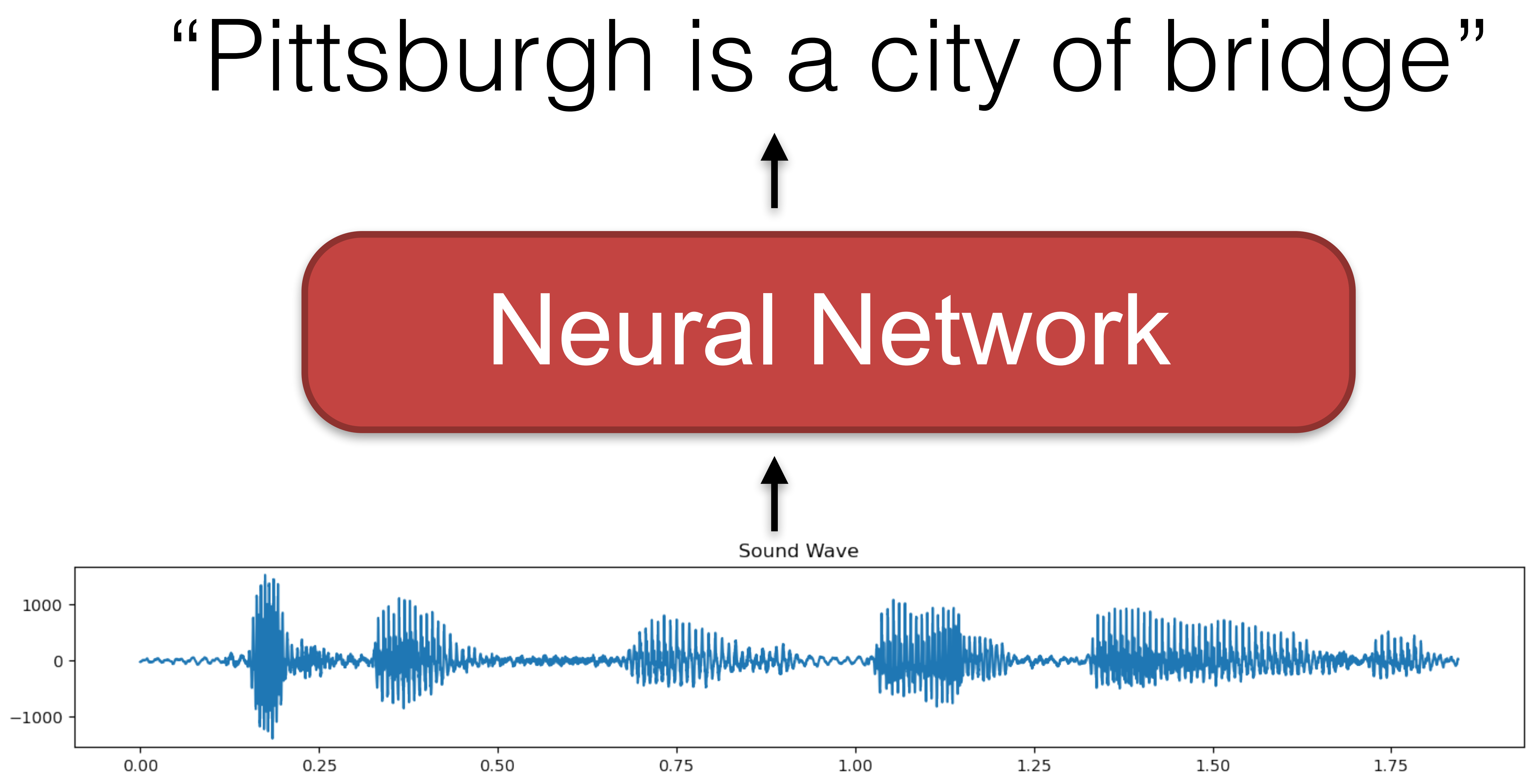
- MFCC:
  - 12 MFCC (mel frequency cepstral coefficients)
  - 1 energy feature
  - 12 delta MFCC features
  - 12 double-delta MFCC features
  - 1 delta energy feature
  - 1 double-delta energy feature

- Total 39-dimensional features

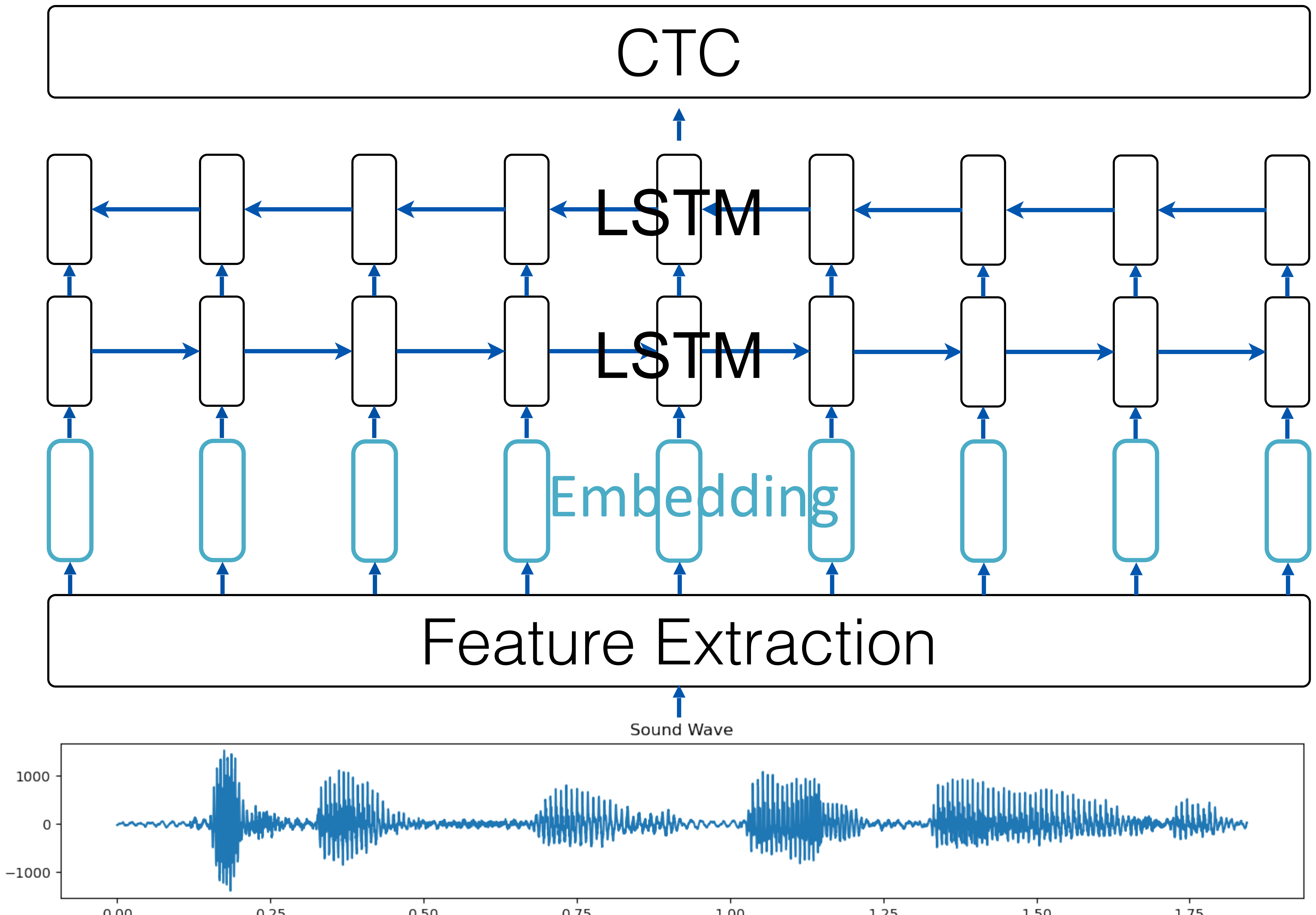
End-to-end ASR

- Train a deep network that directly maps speech signal to the target letter/word sequence

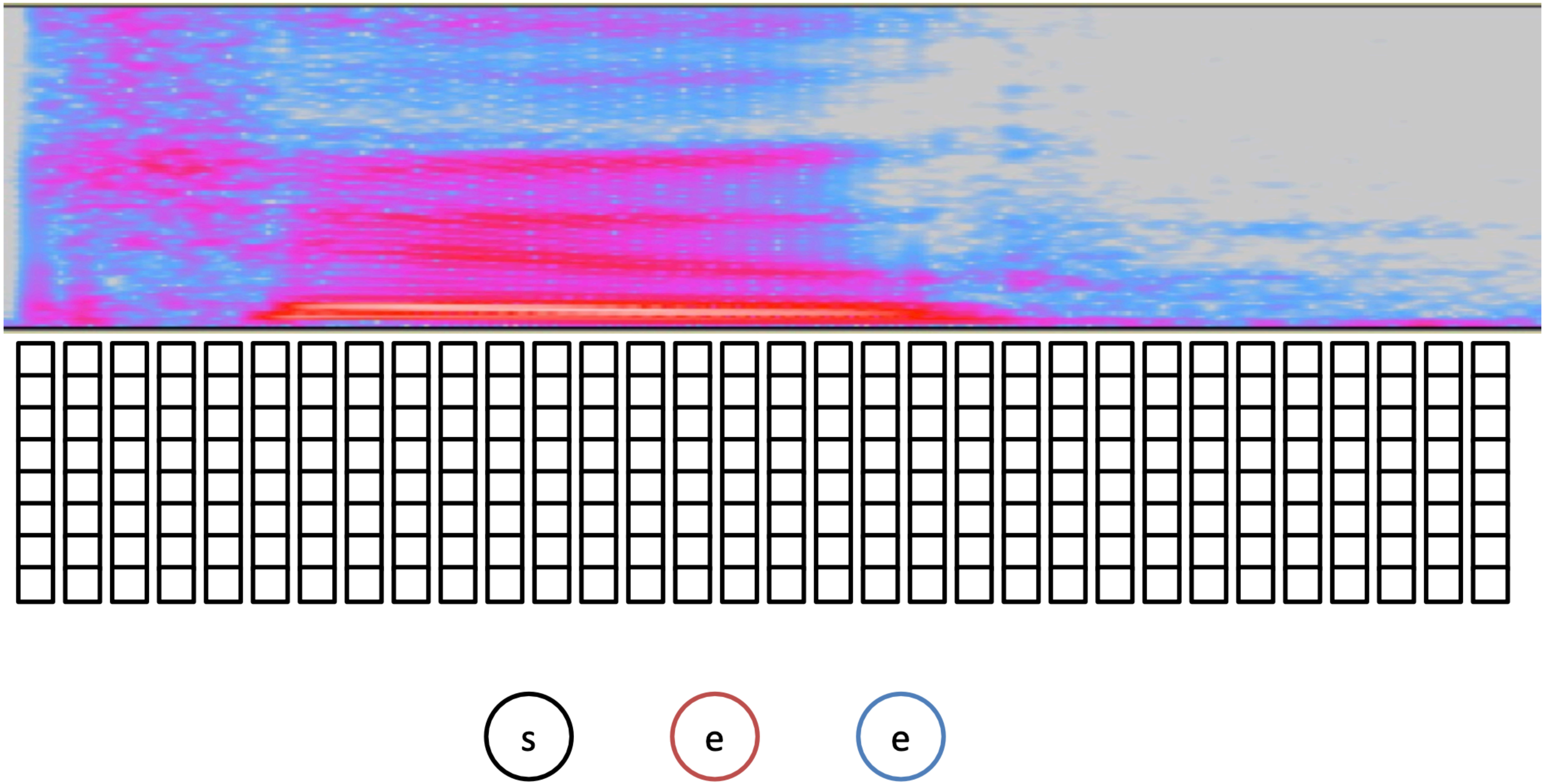
- Easy to build ASR systems for new tasks without expert knowledge



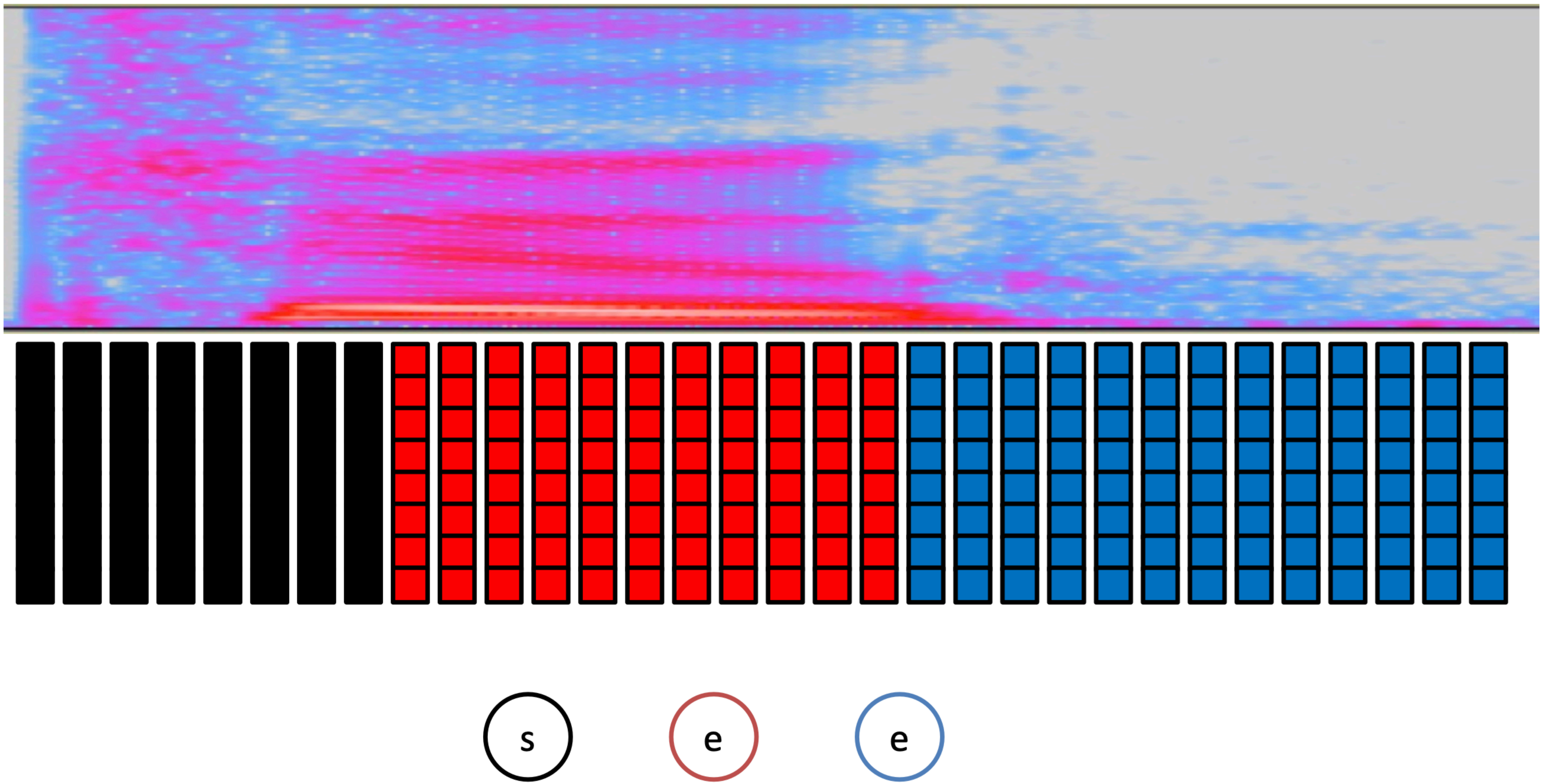
End-to-end ASR Network Architecture (LSTM)



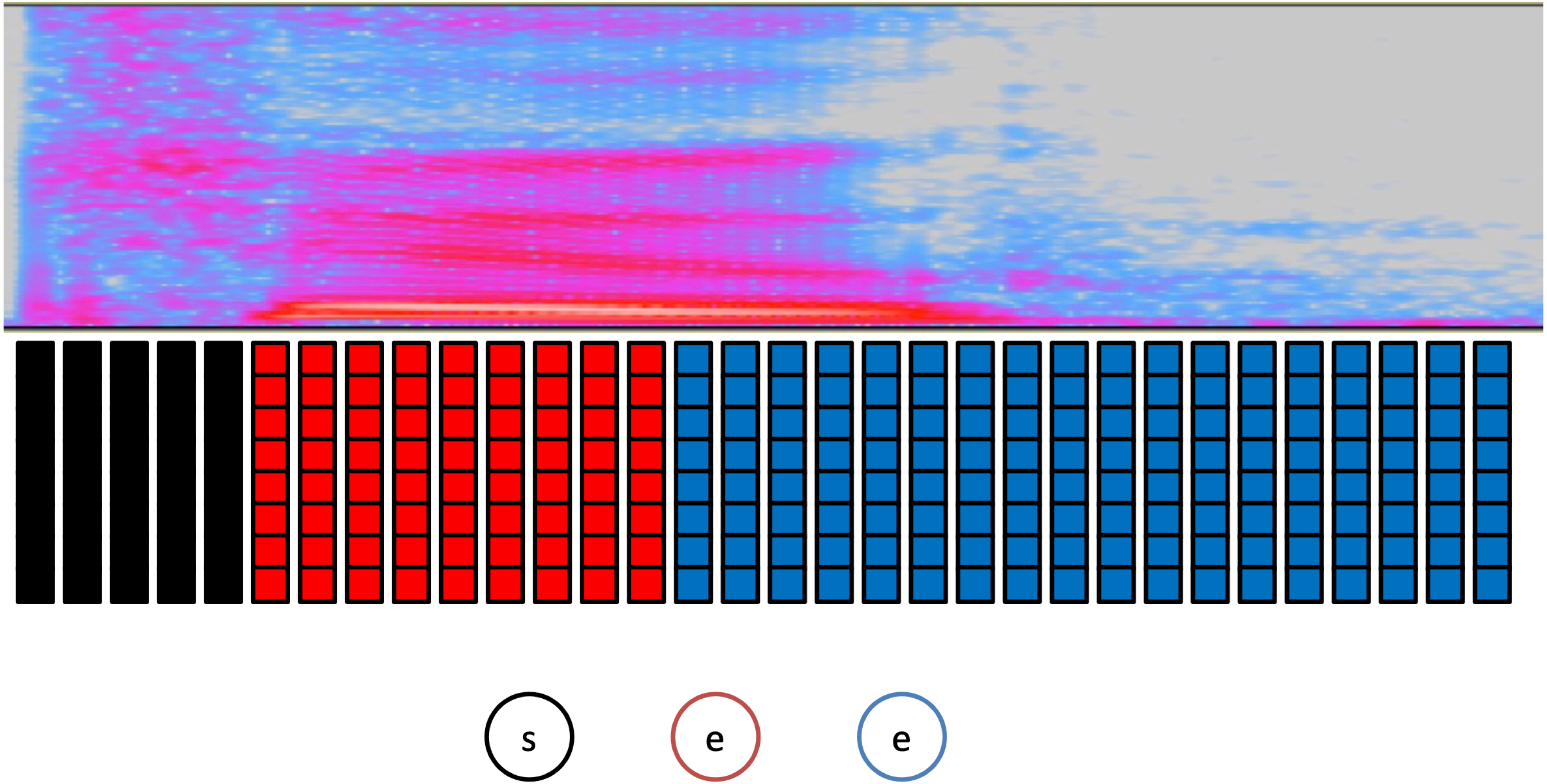
Alignment Problem



Alignment Problem



Alignment Problem



Possible Frame-Phoneme Alignment

input (NN feature vector for each frame)	x_1	x_2	x_3	x_4	x_5	x_6	x_7
per frame prediction (include blank)	s	_	e	e	_	e	_
output	s		e			e	

Possible Frame-Phoneme Alignment

input (NN feature vector for each frame)

x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8

per frame prediction (include blank)

s _ e e _ e _

s s _ _ e e e _ e e _

s _ e e e _

output

s e e

Connectionist Temporal Classification (CTC)

$$p(Y \ X) = \sum_{\text{all valid alignment a for} Y} p(a \ X)$$

x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8

s s _ _ e e e _ e e _

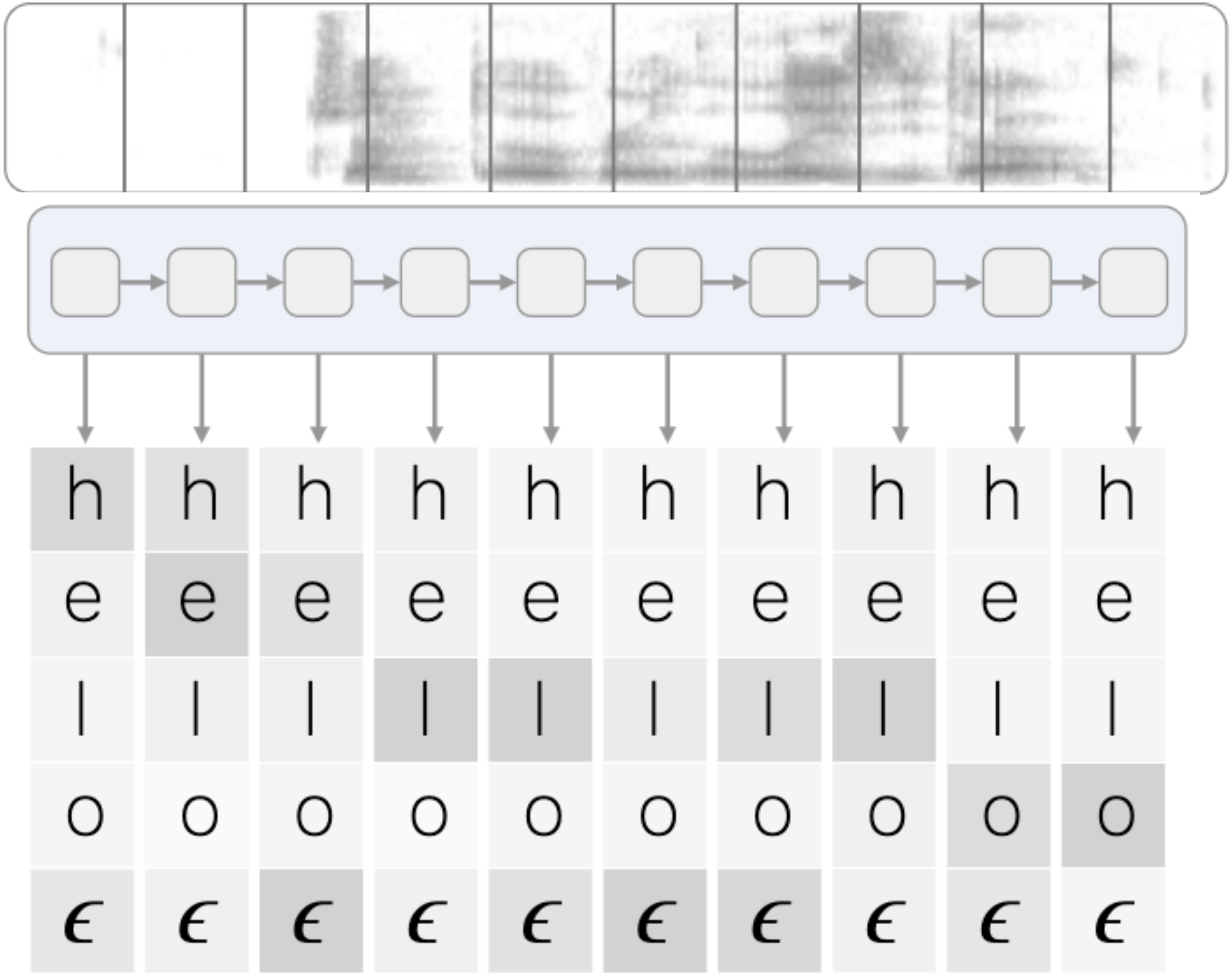
s _ e e _ e _

Connectionist Temporal Classification (CTC)

MFCC feature sequence

Neural network

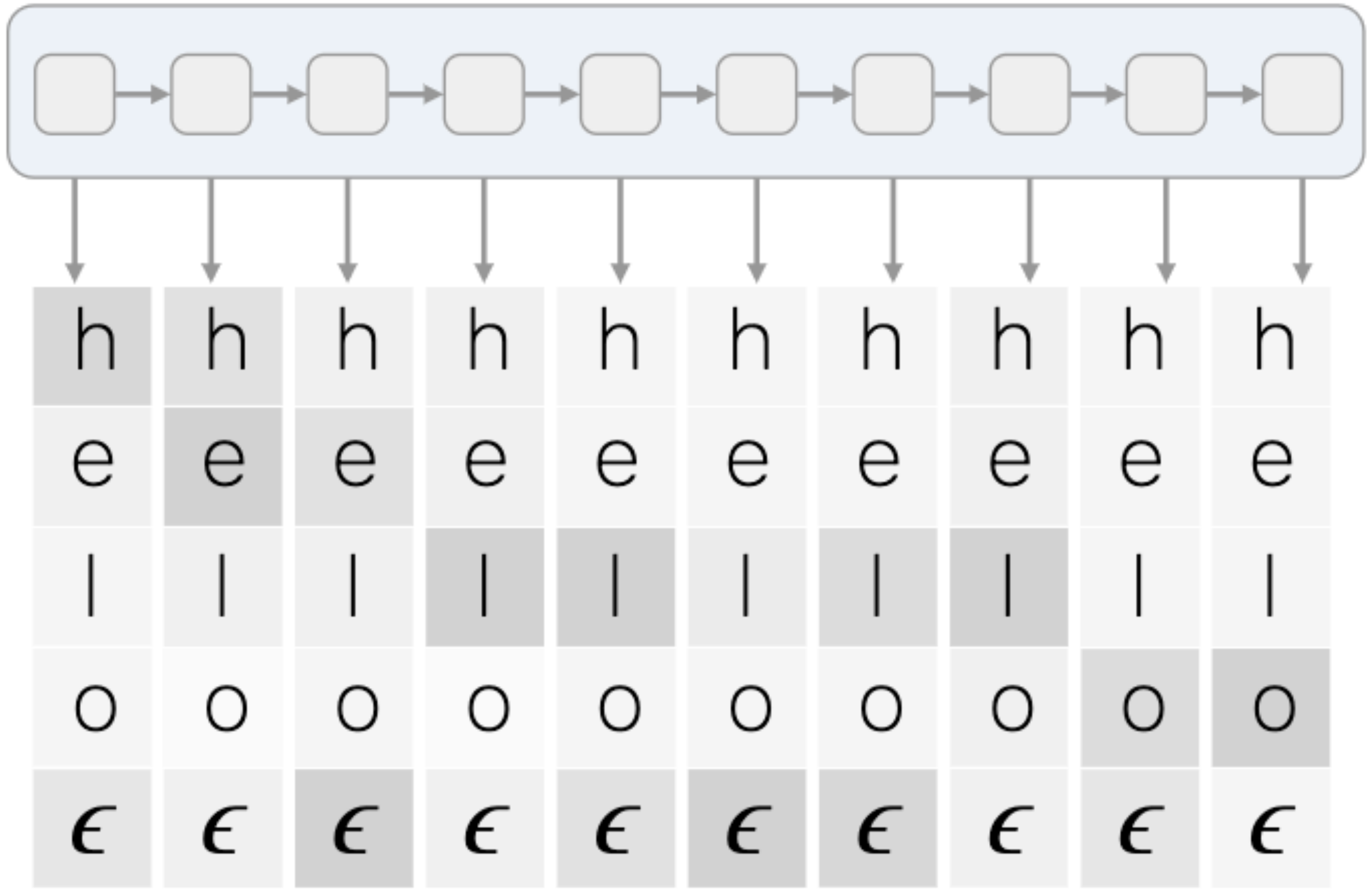
NN computes probabilities of token per frame



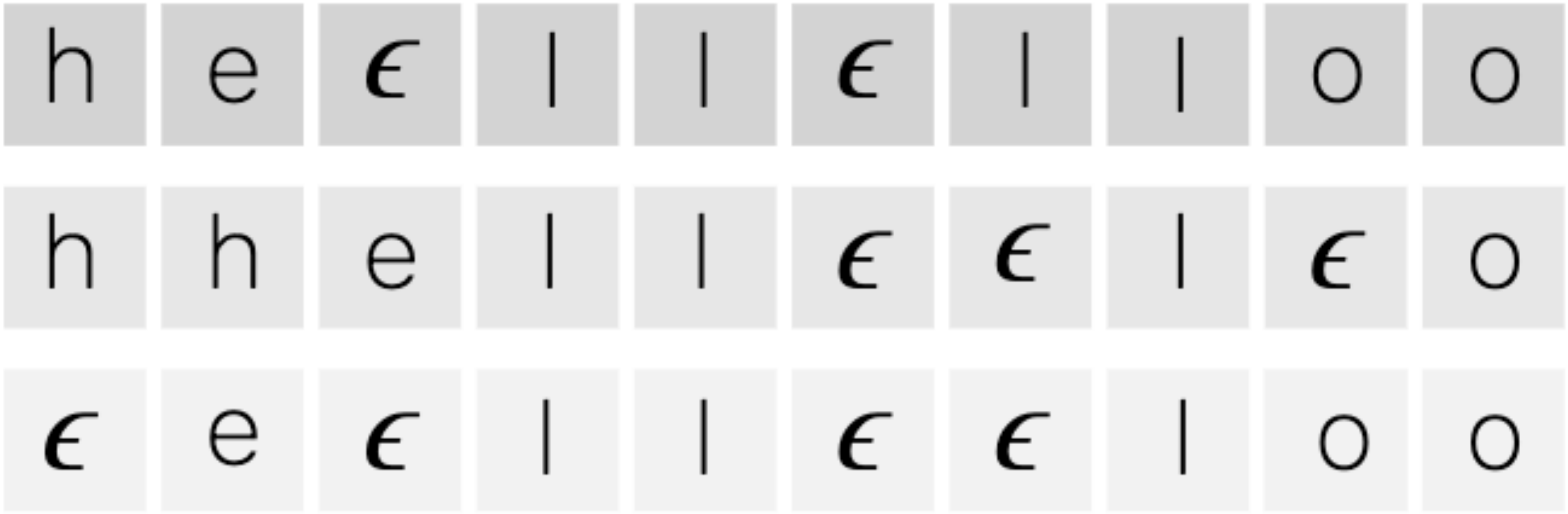
Connectionist Temporal Classification (CTC)

Neural network

NN computes probabilities of token per frame



compute sequence probability



Connectionist Temporal Classification (CTC)

compute sequence probability

h	e	ϵ	l	l	ϵ	l	l	o	o
h	h	e	l	l	ϵ	ϵ	l	ϵ	o
ϵ	e	ϵ	l	l	ϵ	ϵ	l	o	o

compute possible output, marginalize over alignments

h	e	l	l	o
e	l	l	o	
h	e	l	o	

Connectionist Temporal Classification (CTC)

$$p(Y \;\; X) = \sum_{\text{valid alignment a for} Y} \prod_{t=1}^{T} p(a_t \;\; X)$$

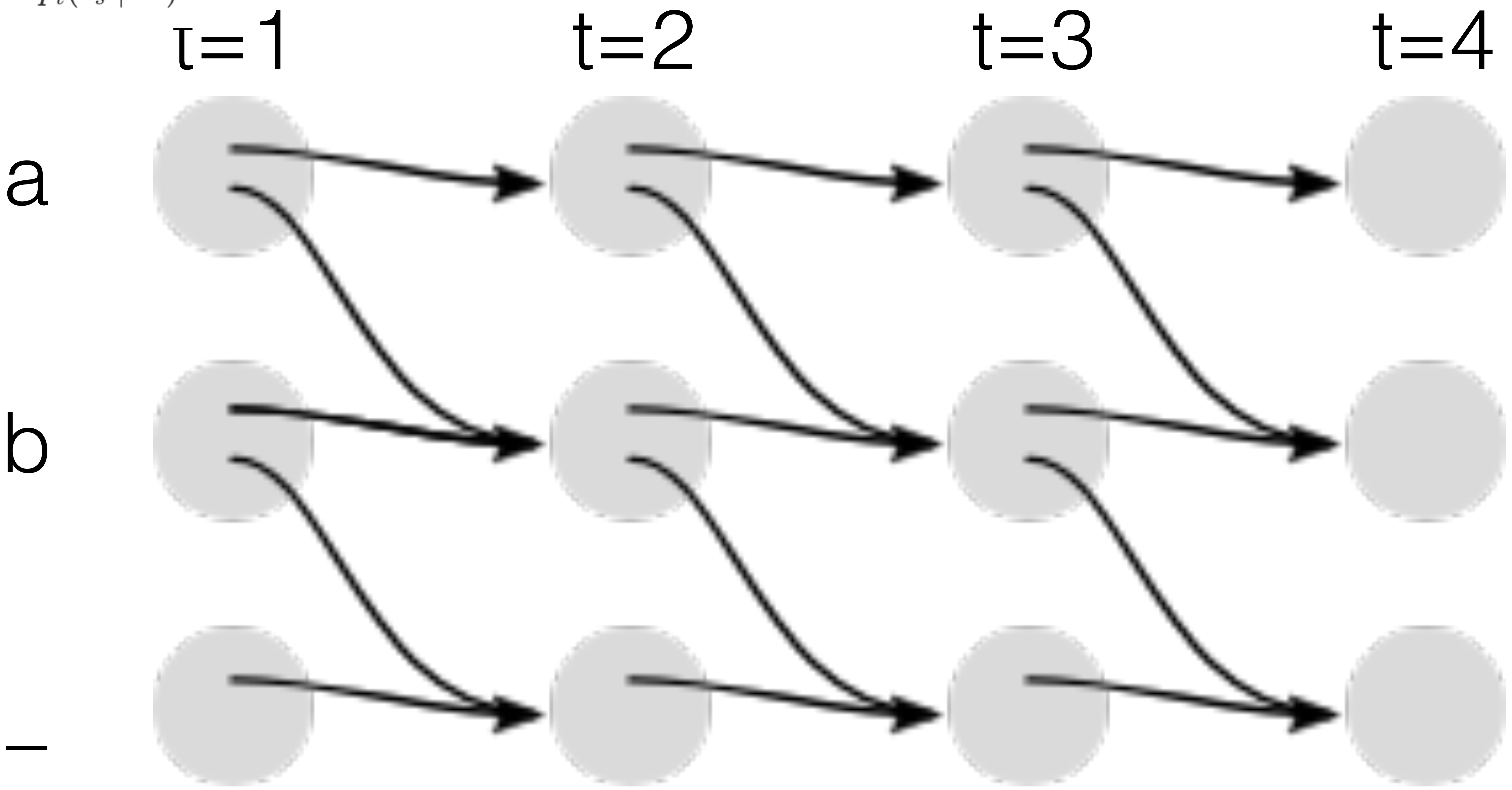
Direct summing over all alignments can be expensive, instead we use dynamic programming to efficiently compute the probability

CTC

$$Z = [\_, y_1, \_, y_2, \_, y_3, \_, \ldots, y_n, \_]$$

Each node represents a partial sum of score

$$\alpha_{s,t} = (\alpha_{s-1,t-1} + \alpha_{s,t-1}) \cdot p_t(z_s \mid X)$$



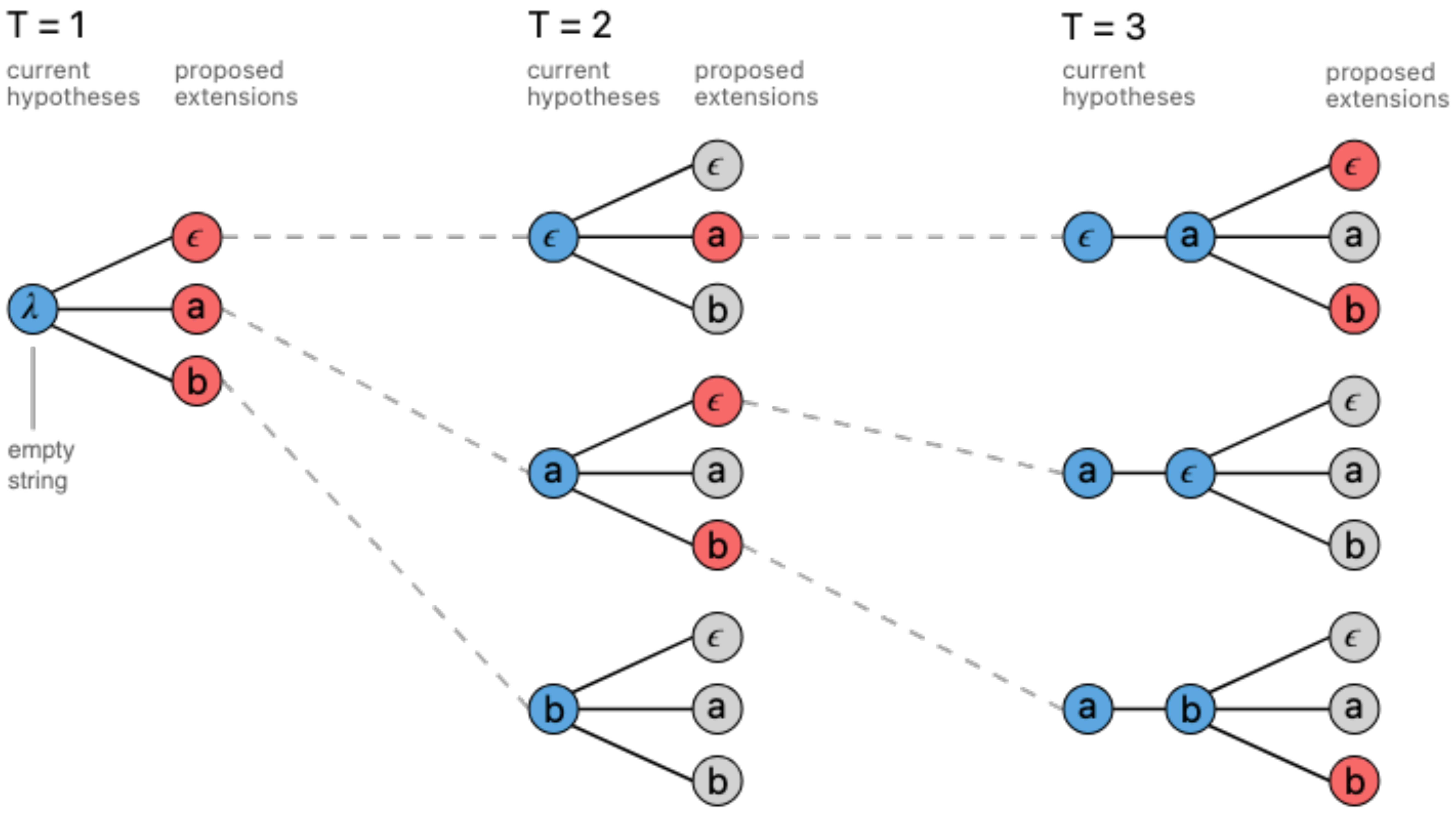
CTC training

- Once computed the probability

$$\min \sum_{(X,Y)\in\mathscr{D}} -\log p(Y \quad X)$$

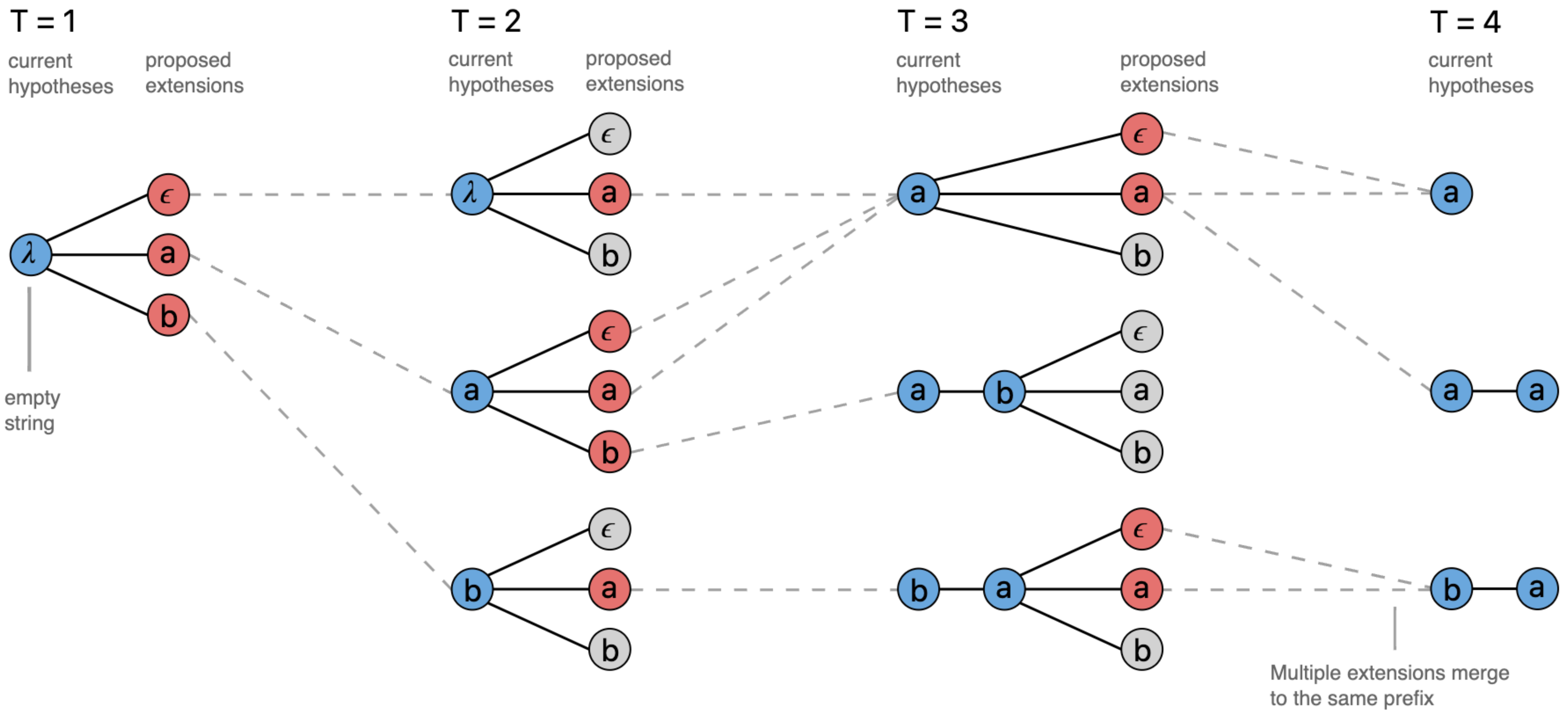
CTC Inference

- Beam search with CTC



CTC Inference

- Beam search with CTC



Demo

- https://distill.pub/2017/ctc/

Software support

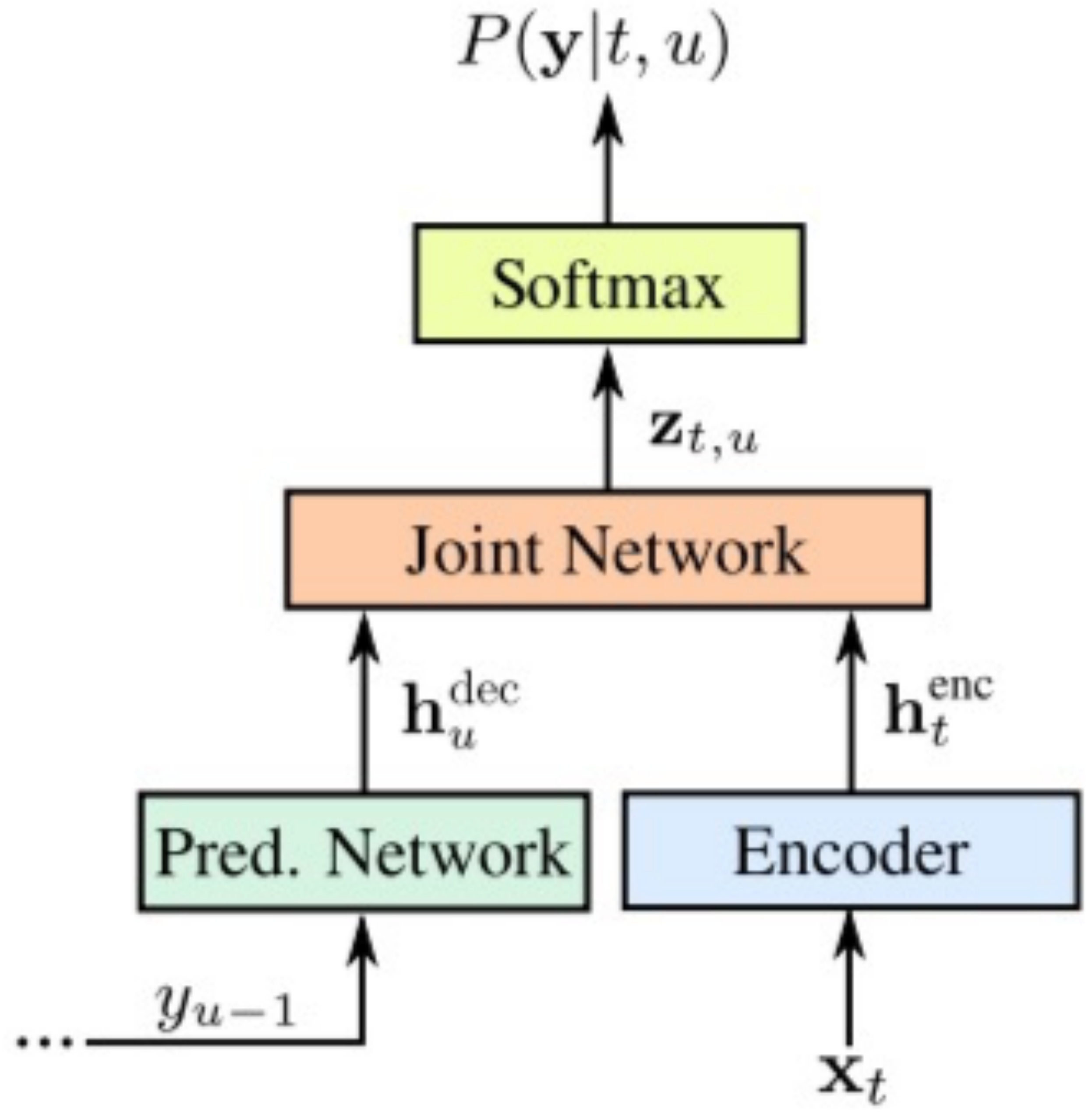
- CTC loss is supported in all major DL library
- wart-ctc: open source implementation of a fast CTC in CUDA and C++
-

Advanced End-to-end ASR

- RNN Transducer
  - Combining CTC and Language Model
- Conformer

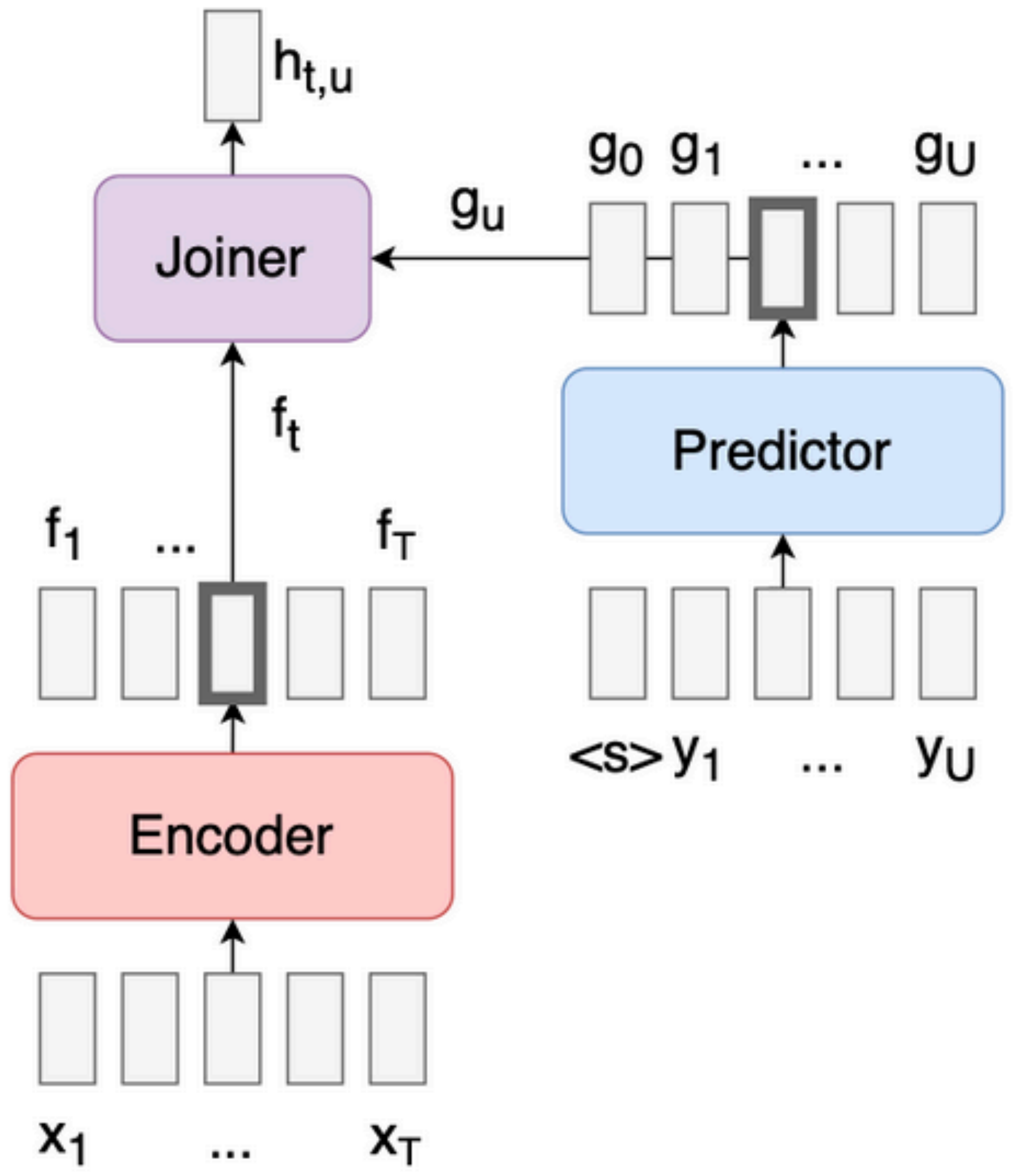
RNN Transducer

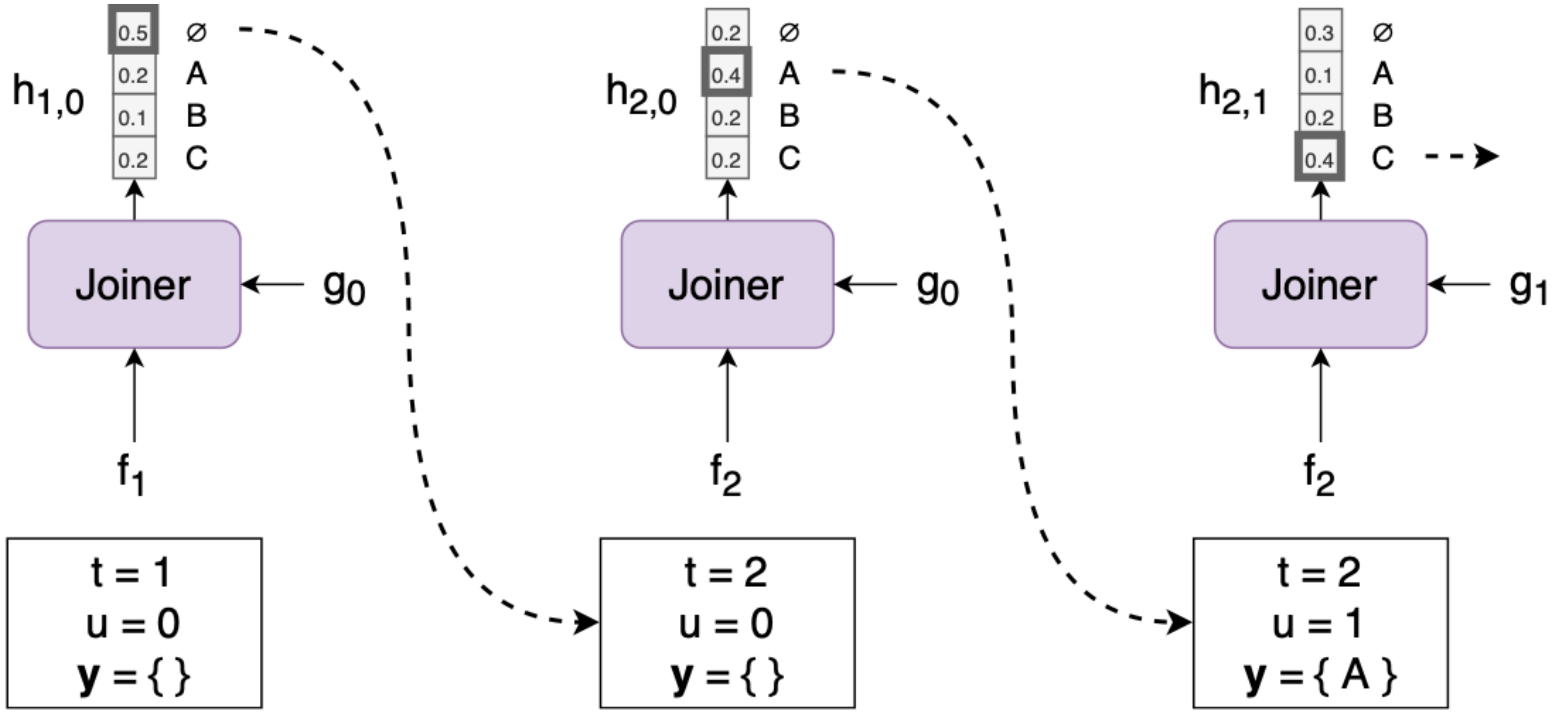
- Directly optimizes target word sequence as correct label
    - Graphemes (letters) or word parts (10k-50k) used in practice
- Learned combination of acoustic + language model pieces
- Conditions on sequence output so far (y_{t-1})



RNN Transducer

- Autoregressive Generation
- Predictor inputs are only non-black tokens (y)
- Do not increment t if non-blank token is output

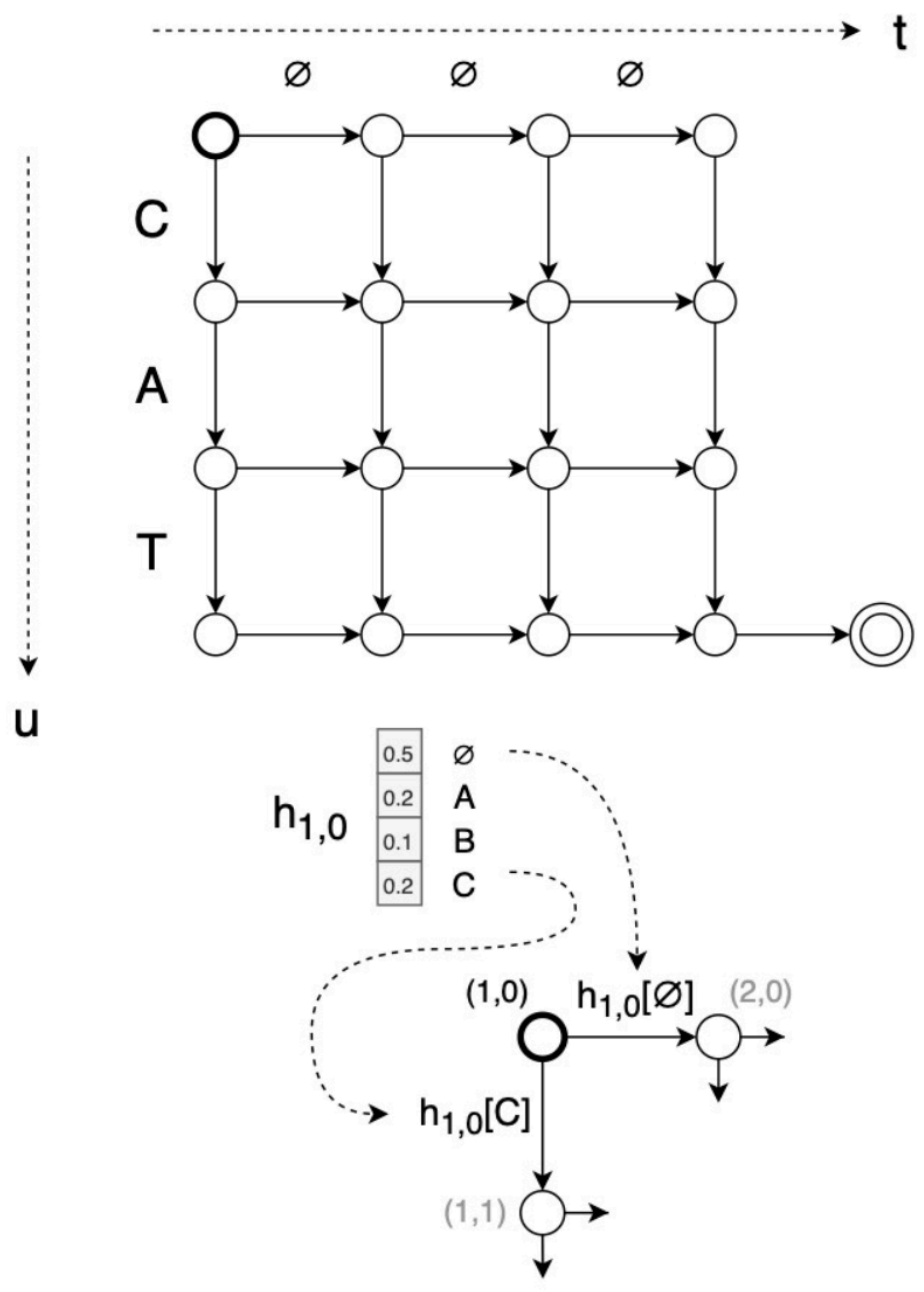


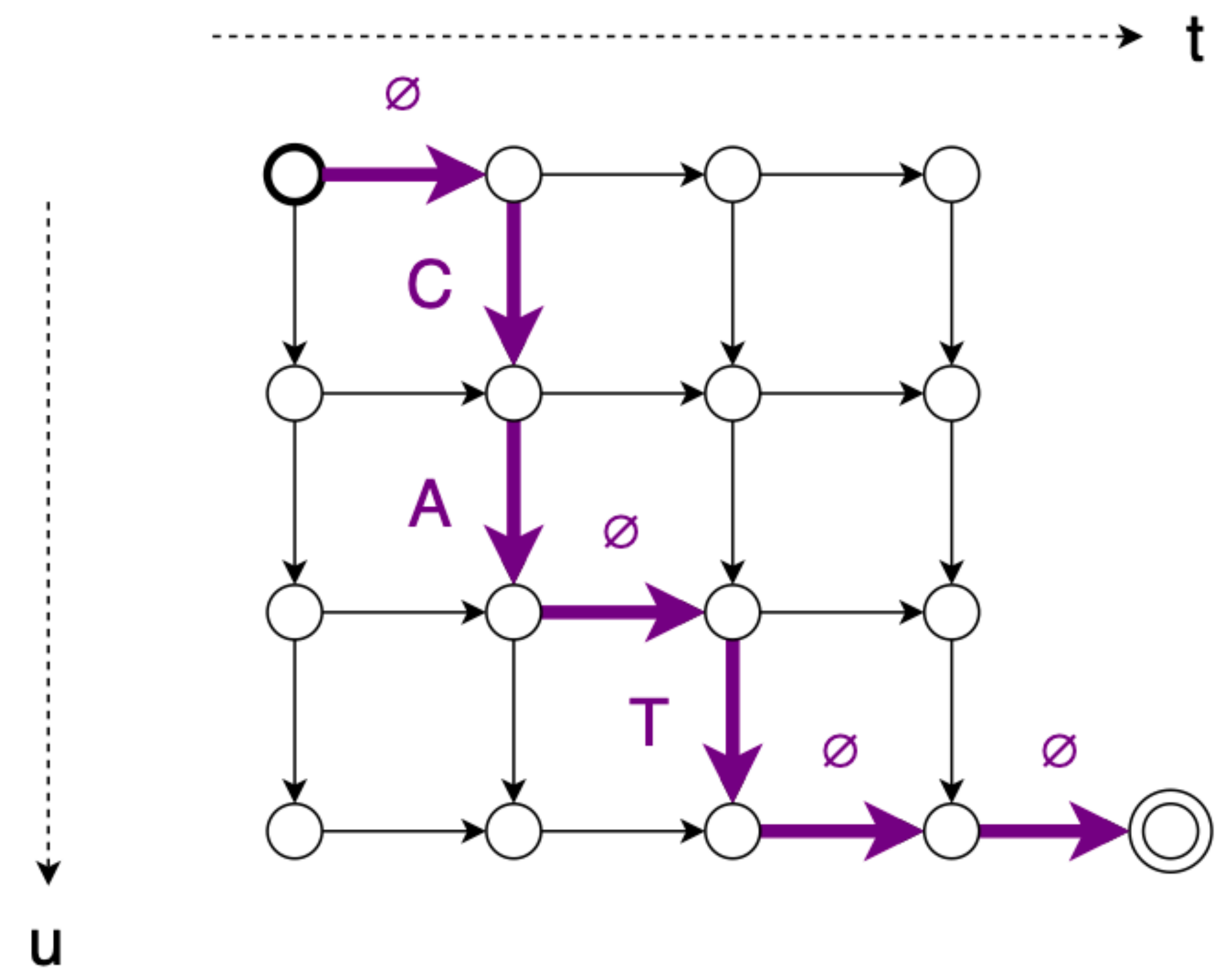


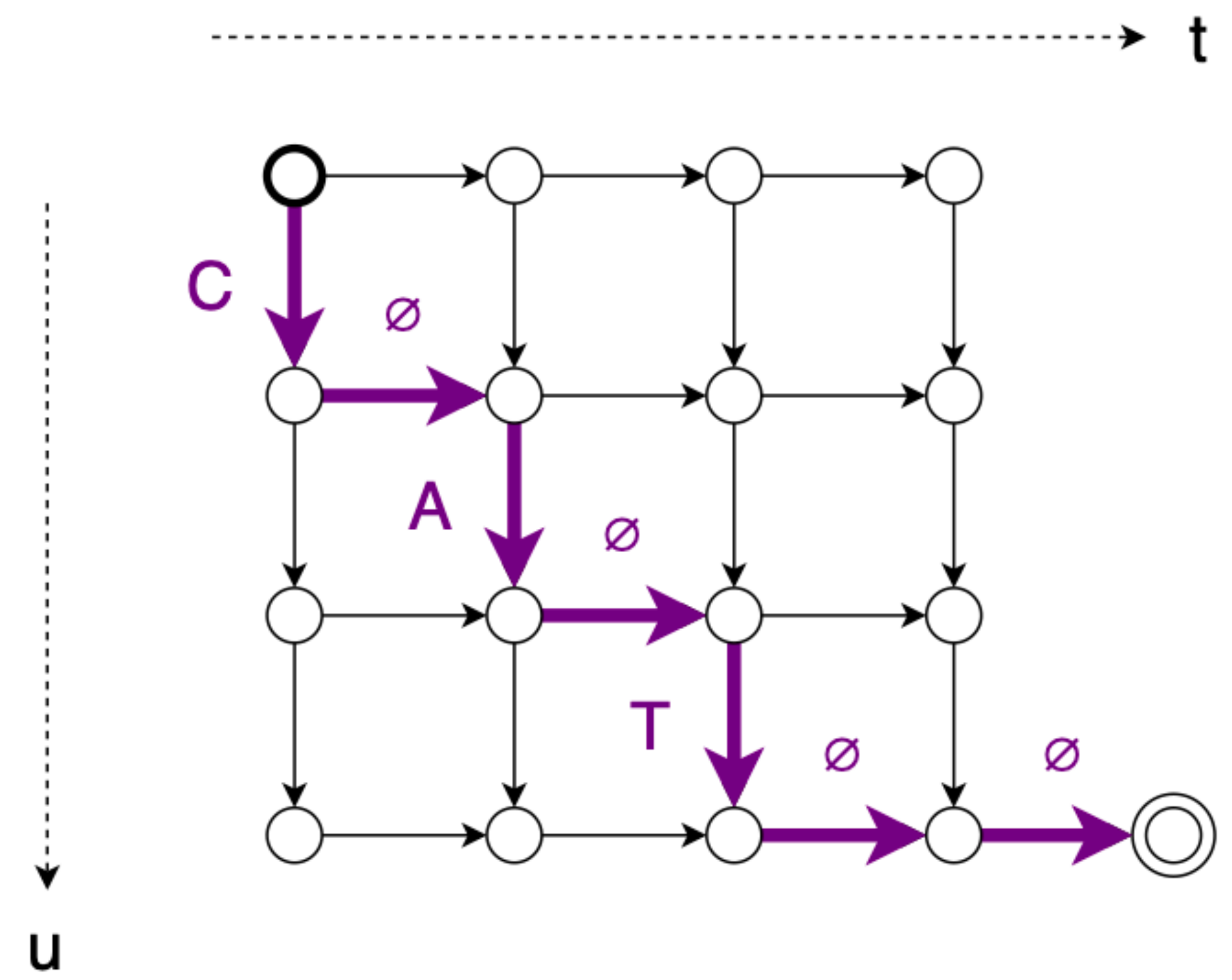
RNN Transducer Loss

- Many alignments are consistent with groundtruth

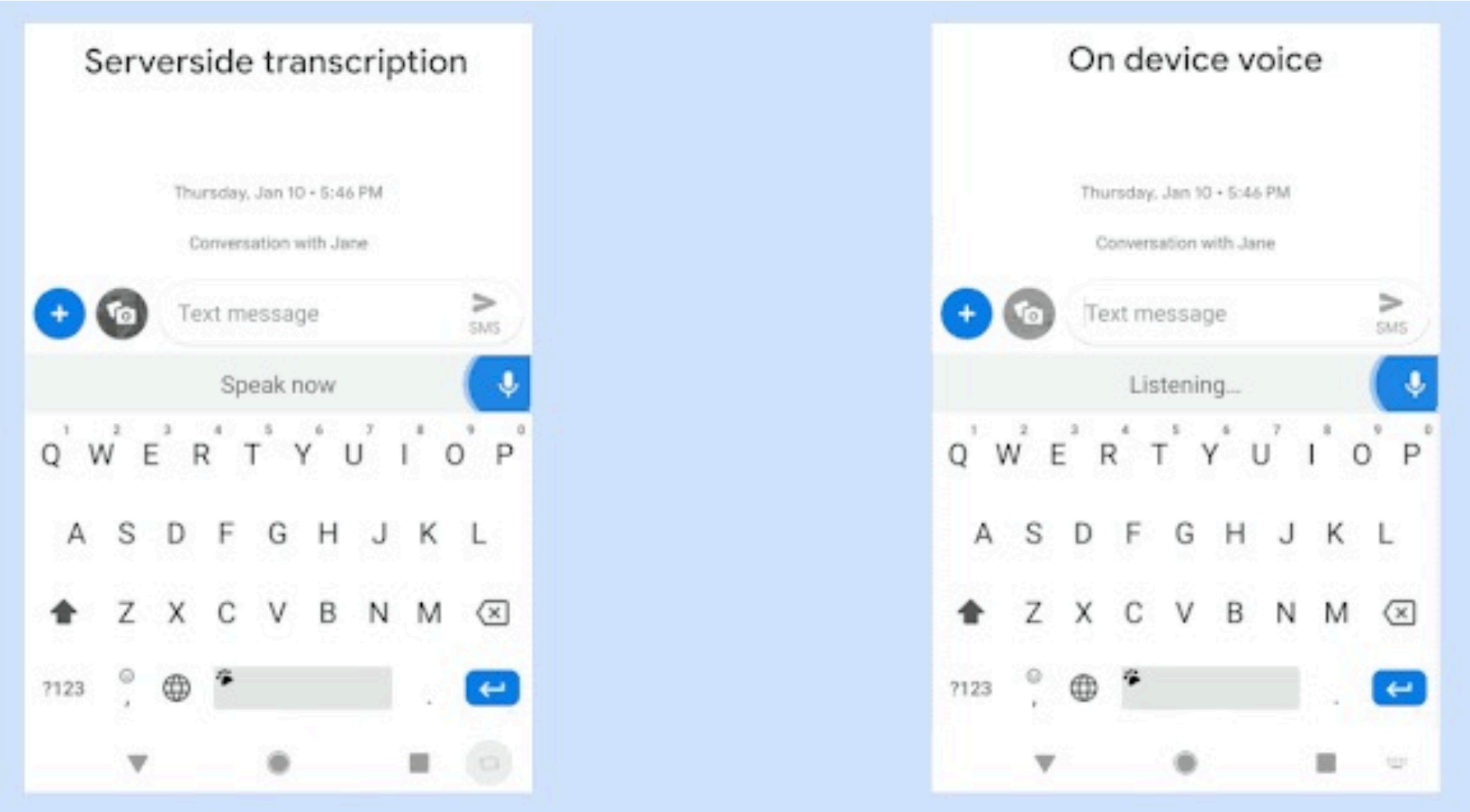
- CTC style dynamic programming







Google on-device ASR enabled by RNN-T



Key Techniques for on-device ASR

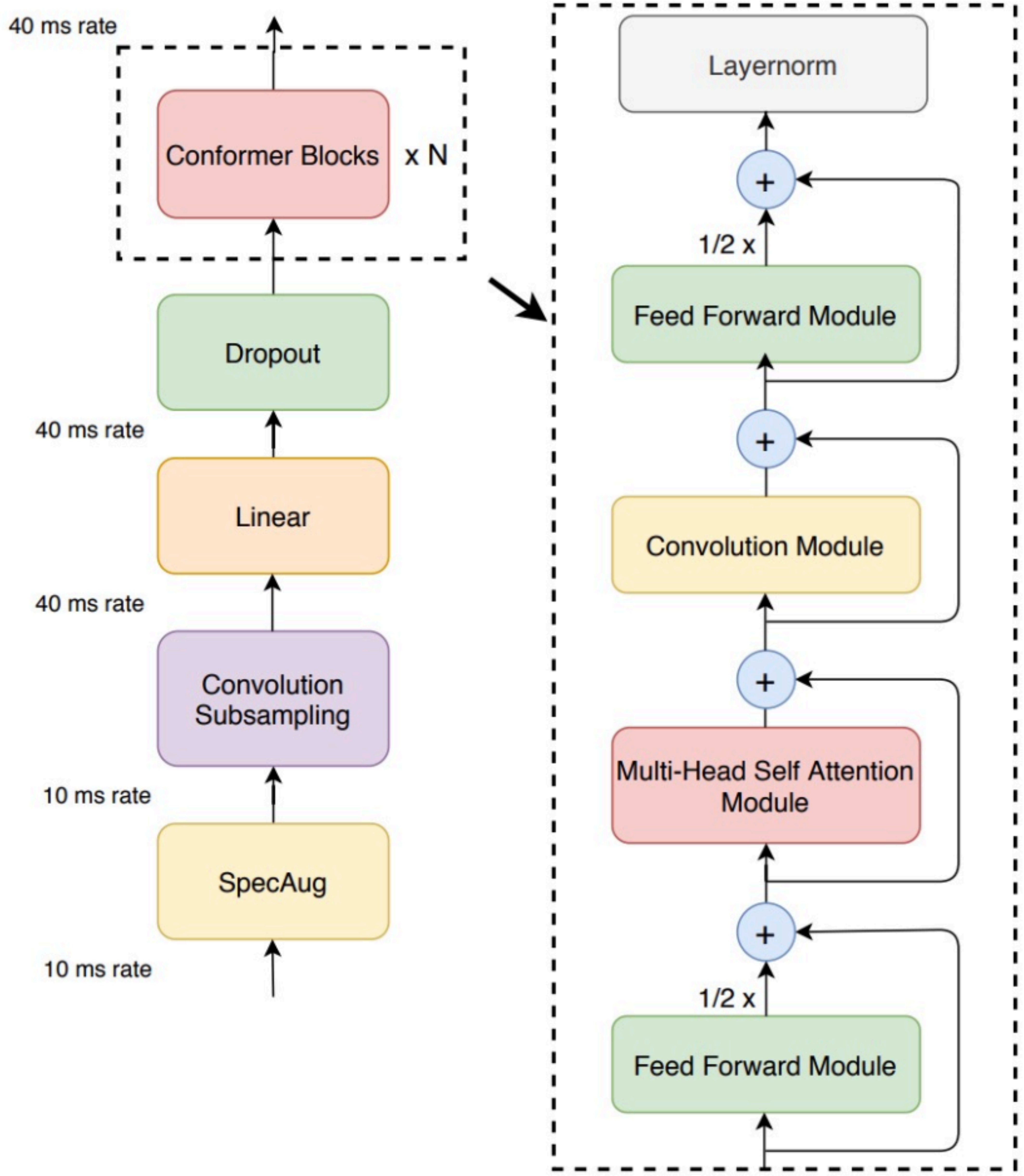
- RNN Transducer architecture
- Scaling up training with parallel RNN-T.
- Decoding: Beam search with a single NN instead of weighted finite state transducer decoding machinery
- NN parameter quantization.
  4x model size compression. 4x runtime speed improvement
- LM contextual biasing. User-specialized LM to upweight common requests / inputs
- Improved text normalization + sub-word output units

Conformer

- Convolution + Transformer

- RNN-T loss

Conformer Encoder

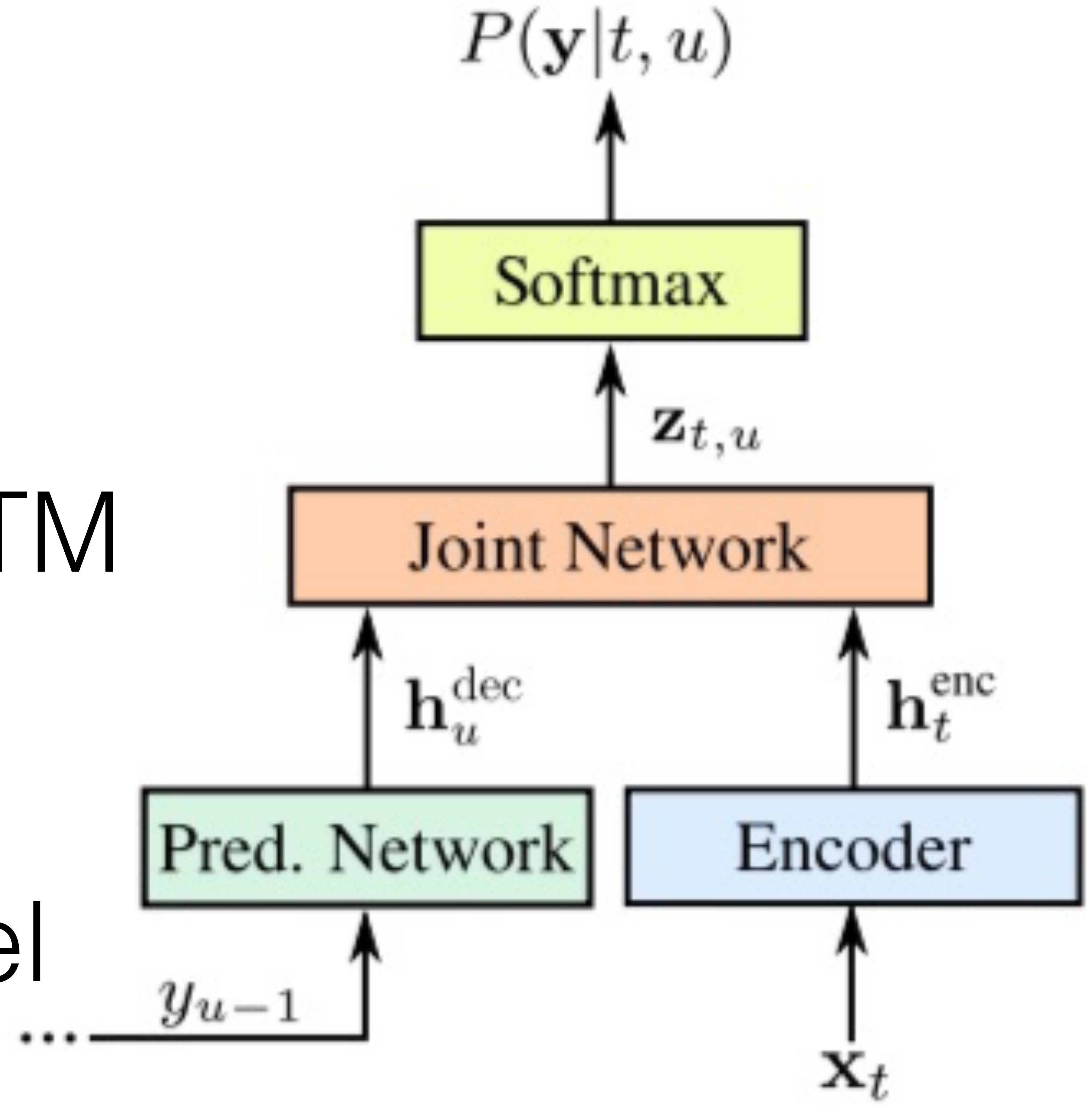


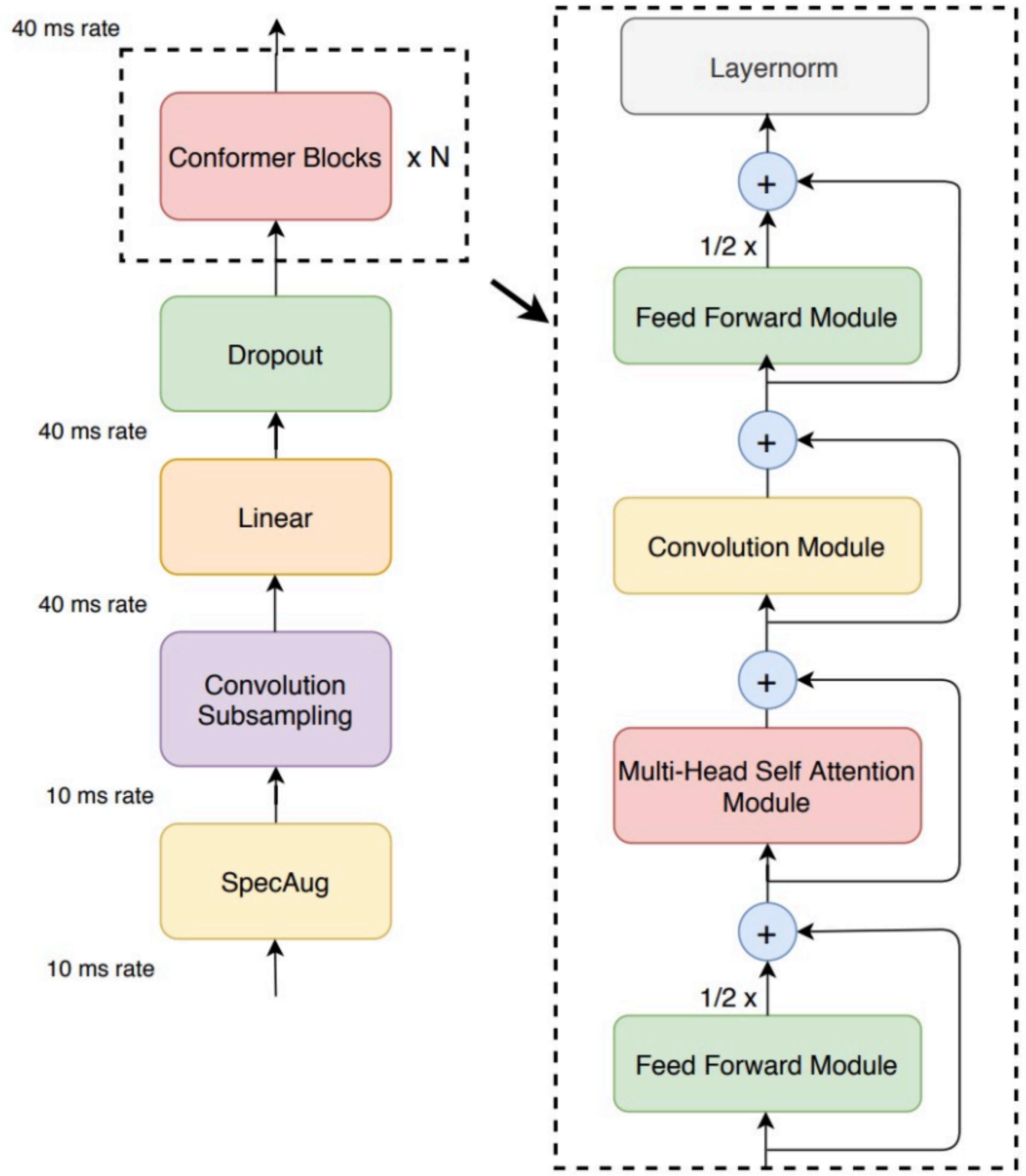
Conformer

- Conformer Encoder + RNN Transducer

1 layer LSTM

3 layer LSTM language model





Conformer Performance

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**

Summary

- Measuring Performance
  - Word Error Rate: edit distance between reference and candidate
- Audio Feature Extraction: MFCC
- End-to-end ASR model
  - CTC loss to sum all valid alignments
- RNN Transducer: CTC+Language Model
- Conformer: Convolution + Transformer + RNN-T

Language in 10

Homework 2

- ASR