11-737 Multilingual NLP

End-to-End Speech Recognition

Carnegie Mellon University
Language Technologies Institute

Today’s agenda

- Introduction to end-to-end speech recognition
- HMM-based pipeline system
- Connectionist temporal classification (CTC)
- Attention-based encoder decoder
- Joint CTC/attention (Joint C/A)
- RNN transducer (RNN-T)

Noisy channel model (1970s-)

Noisy channel model (1970s-)

$$\arg\max_{W} p(W|O)$$

O: Speech sequence
W: Text sequence

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/,\ /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_W p(W|O) = \arg\max_W \sum_L p(W, L|O)$$

Sum rule

$$= \arg\max_W \sum_L \frac{p(O|W, L)p(L|W)p(W)}{p(O)}$$

Product rule

$$= \arg\max_W \sum_L p(O|W, L)p(L|W)p(W)$$

Ignore $p(O)$ as it does not depend on W

$$= \arg\max_W \sum_L p(O|L)p(L|W)p(W)$$

Conditional independence assumption

Noisy channel model

$$\arg\max_{\mathrm{W}} p(W|O) = \arg\max_{W} p(O|W)p(W)$$
$$\approx \arg\max_{W} \sum_{L} p(O|L)p(L|W)p(W)$$

- **Speech recognition**
  - $p(O|L)$: Acoustic model (Hidden Markov model)
  - $p(L|W)$: Lexicon
  - $p(W)$: Language model (n-gram)

- Factorization
- **Conditional independence (Markov) assumptions, CIA**

Noisy channel model (1970s-)

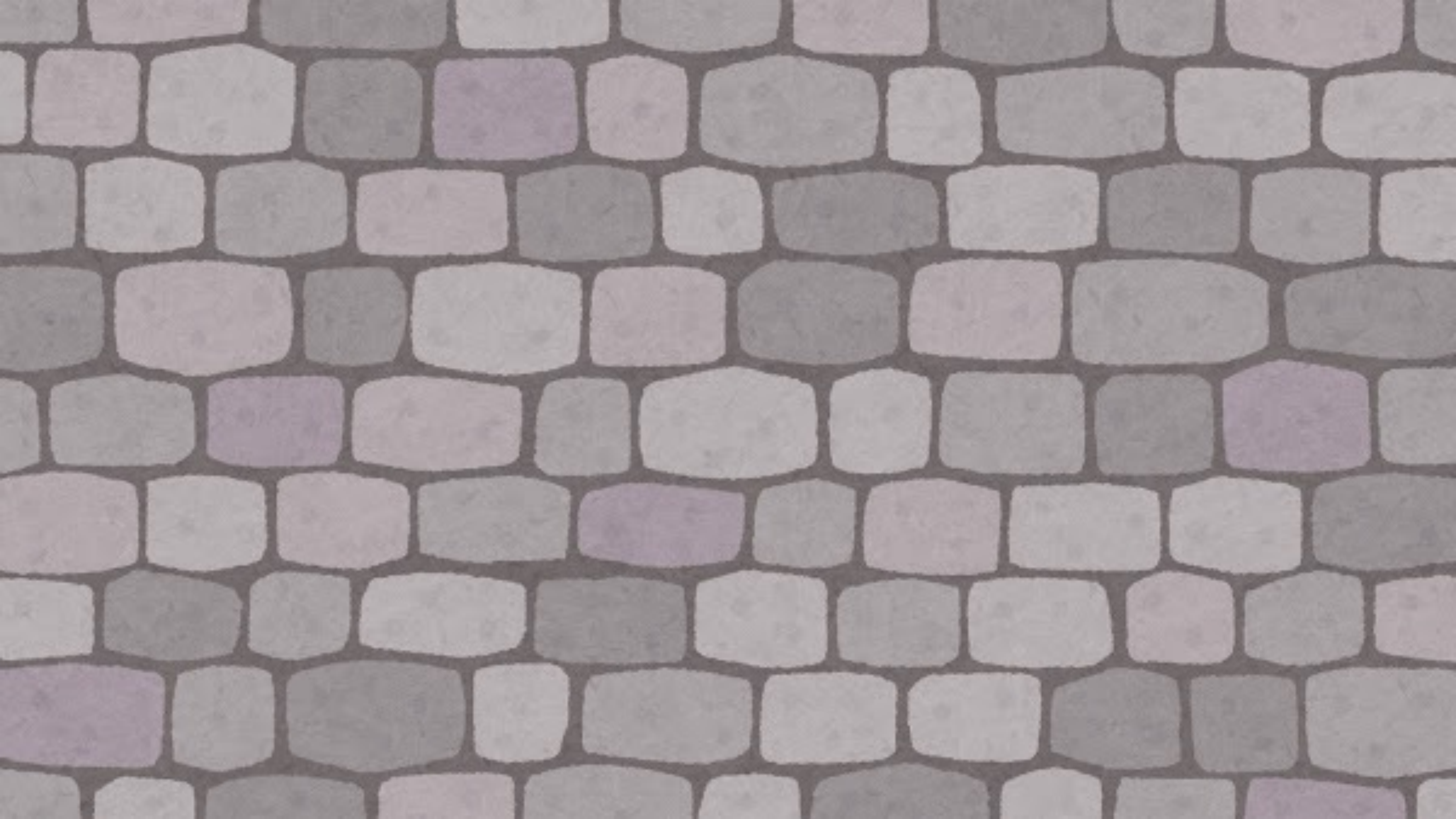
$$\arg\max_{\mathrm{W}} p(W|O) = \arg\max_{W} p(O|W)p(W)$$
$$\approx \arg\max_{W} \sum_{L} p(O|L)p(L|W)p(W)$$

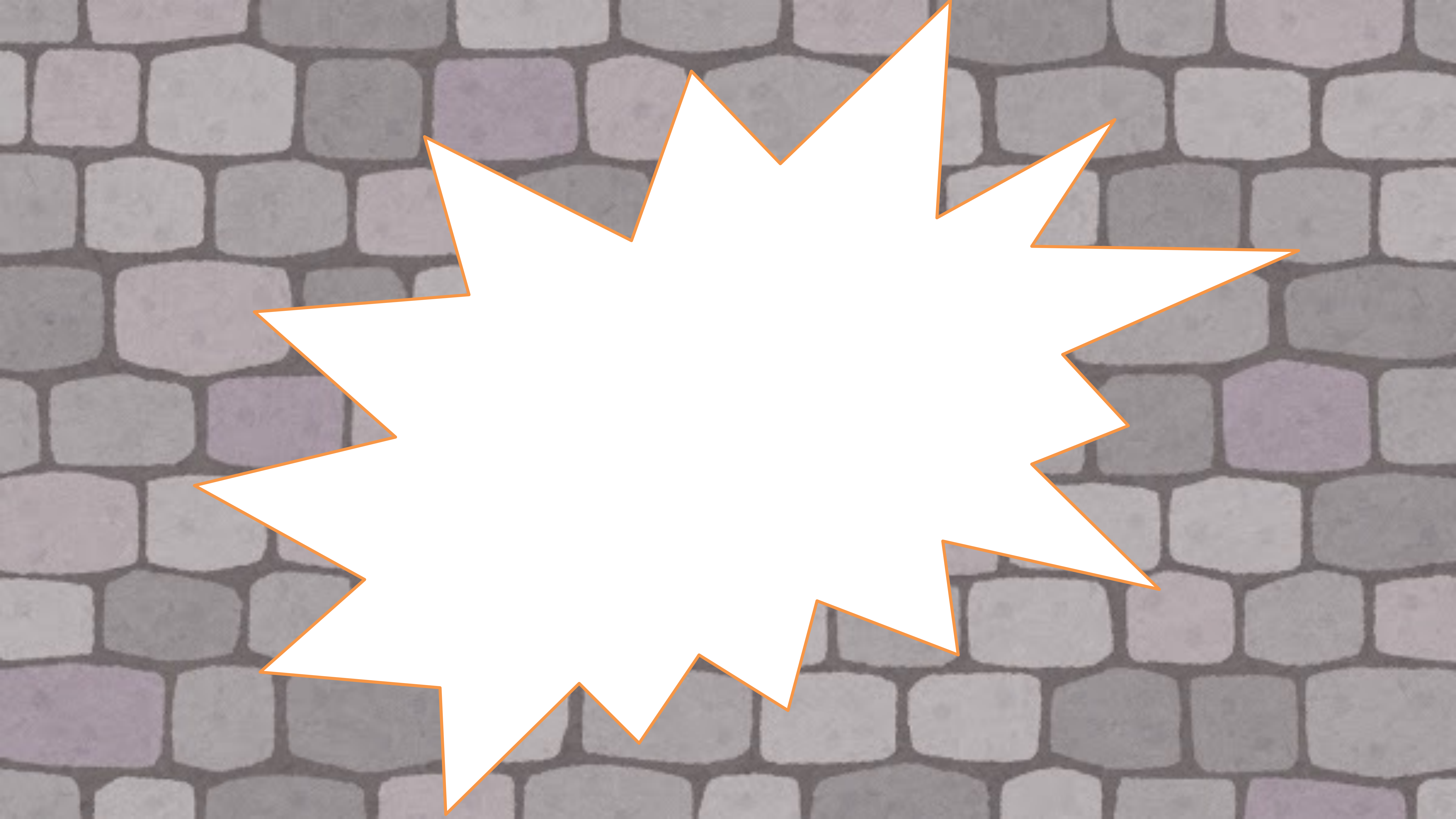
- **Speech recognition**
  - $p(O|L)$: Acoustic model
  - $p(L|W)$: Lexicon
  - $p(W)$: Language model
- Continued 40 years

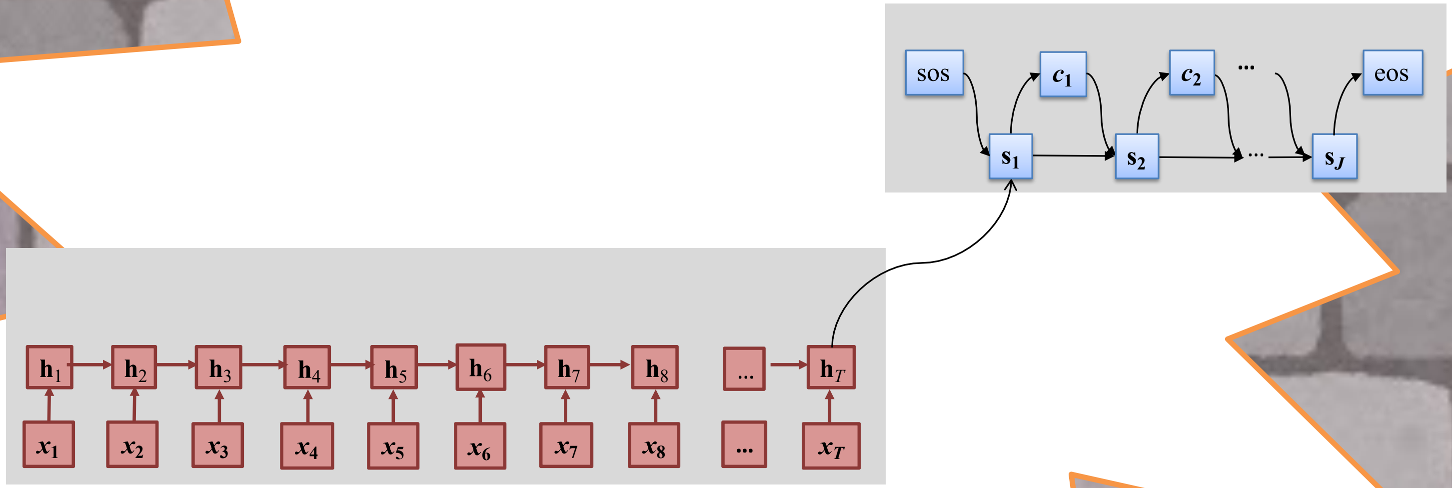
Big barrier:

noisy channel model
HMM
n-gram
etc.

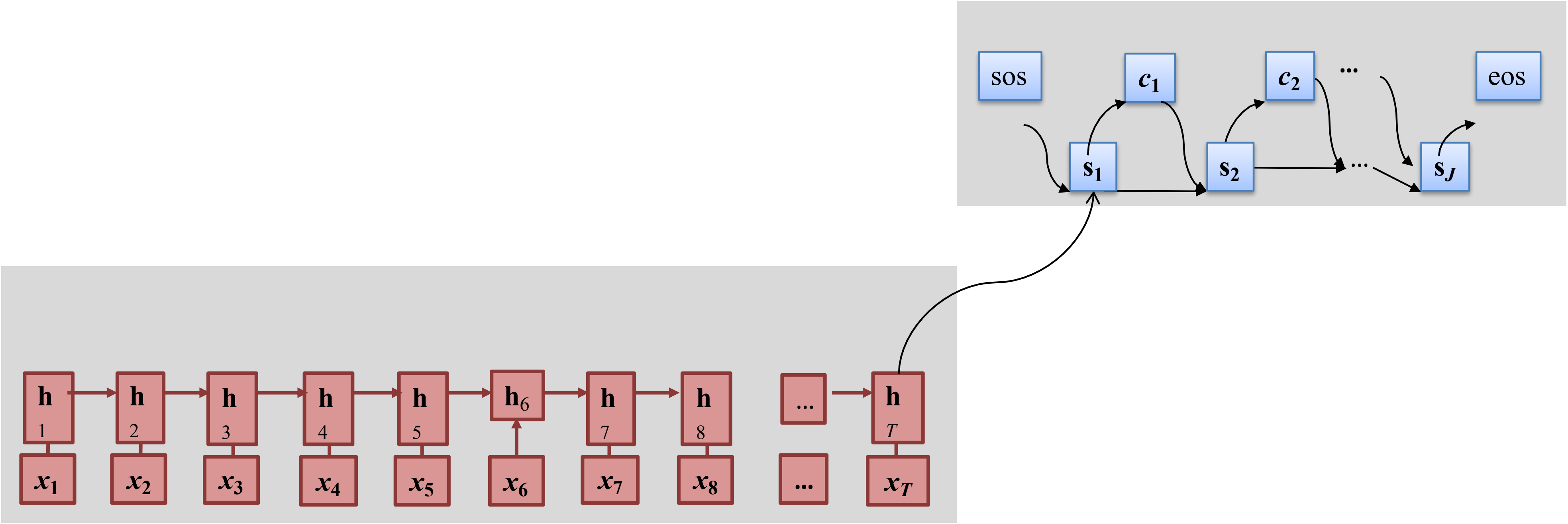
However,






sos
c_1
c_2
...
eos
$\mathbf{s}_1$
$\mathbf{s}_2$
...
$\mathbf{s}_J$
$\mathbf{h}_1$ $\mathbf{h}_2$ $\mathbf{h}_3$ $\mathbf{h}_4$ $\mathbf{h}_5$ $\mathbf{h}_6$ $\mathbf{h}_7$ $\mathbf{h}_8$ ... $\mathbf{h}_T$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 ... x_T

“End-to-End” Processing Using Sequence to Sequence



- Directly model $p(W|O)$ with a **single neural network**
- Great success in neural machine translation

Today's agenda

- Introduction to end-to-end speech recognition
- HMM-based pipeline system
- Connectionist temporal classification (CTC)
- Attention-based encoder decoder
- Joint CTC/attention (Joint C/A)
- RNN transducer (RNN-T)

Seq2seq end-to-end ASR

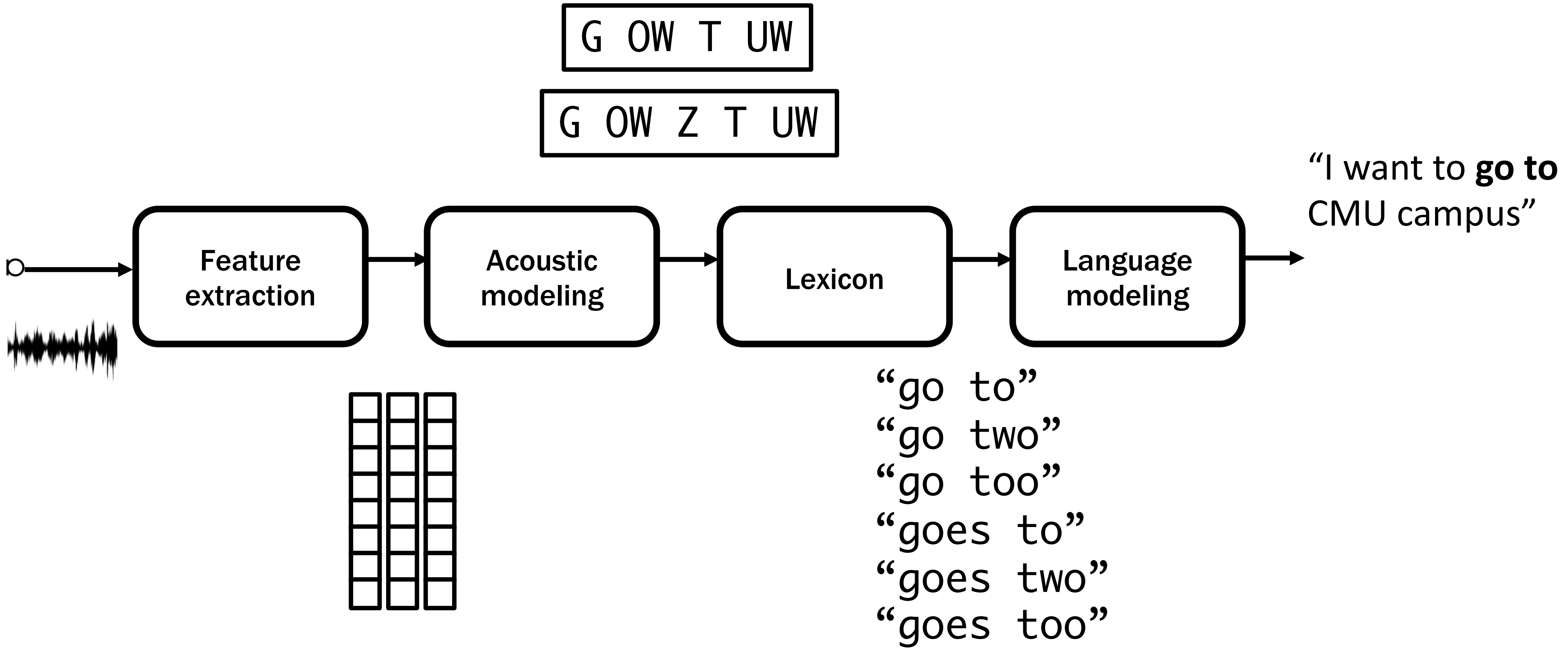
$$\boxed{X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N)}$$

$$f(\cdot)$$

Direct seq2seq mapping function

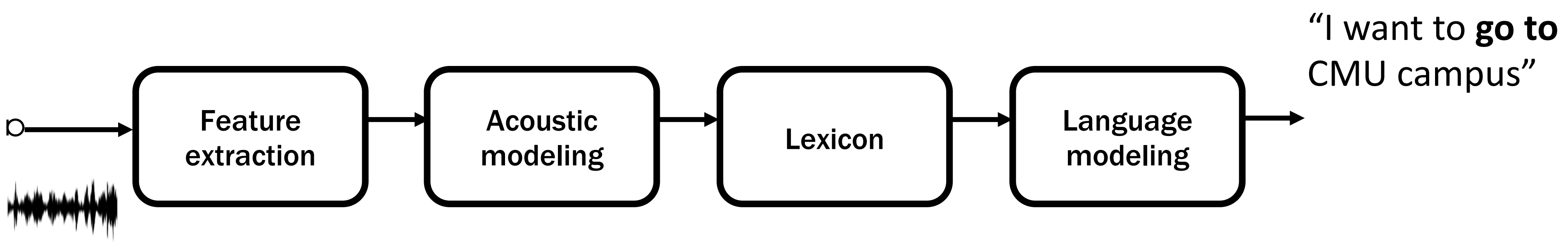
1. HMM-based pipeline system
2. Connectionist temporal classification (CTC)
3. Attention-based encoder decoder
4. Joint CTC/attention (Joint C/A)
5. RNN transducer (RNN-T)

HMM-based speech recognition pipeline



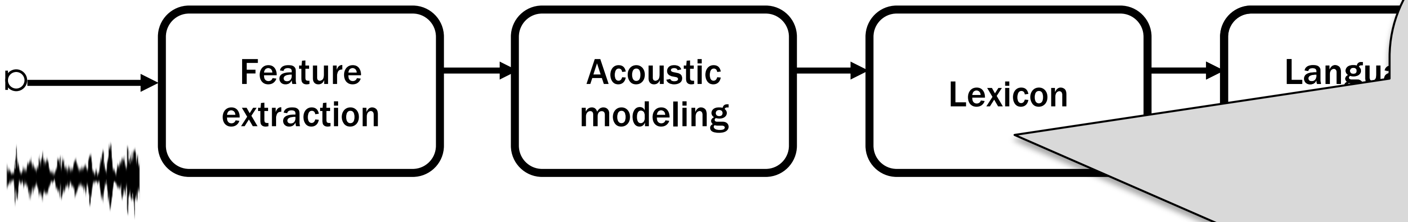
$p(O|L)$ $p(L|W)$ $p(W)$

HMM-based speech recognition pipeline



- Require **a lot of development** for an acoustic model, a pronunciation lexicon, a language model, and finite-state-transducer decoding
- Require linguistic resources
- Difficult to build ASR systems for non-experts

HMM-based speech recognition pipeline



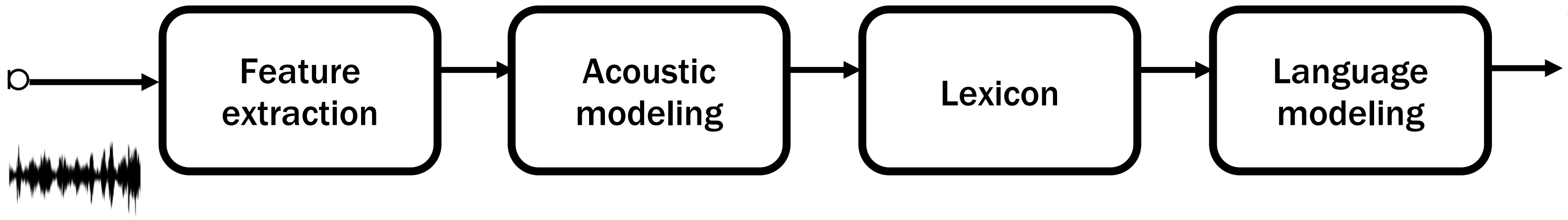
Pronunciation lexion

A	AH
A'S	EY Z
A(2)	EY
A.	EY
A.'S	EY Z
A.S	EY Z
AAA	T R IH P AH L EY
AABERG	AA B ER G
AACHEN	AA K AH N
AACHENER	AA K AH N ER
AAKER	AA K ER
AALSETH	AA L S EH TH
AAMODT	AA M AH T
AANCOR	AA N K AO R
AARDEMA	AA R D EH M AH
AARDVARK	AA R D V AA R K
AARON	EH R AH N
AARON'S	EH R AH N Z
AARONS	EH R AH N Z
…	

100K~1M words!

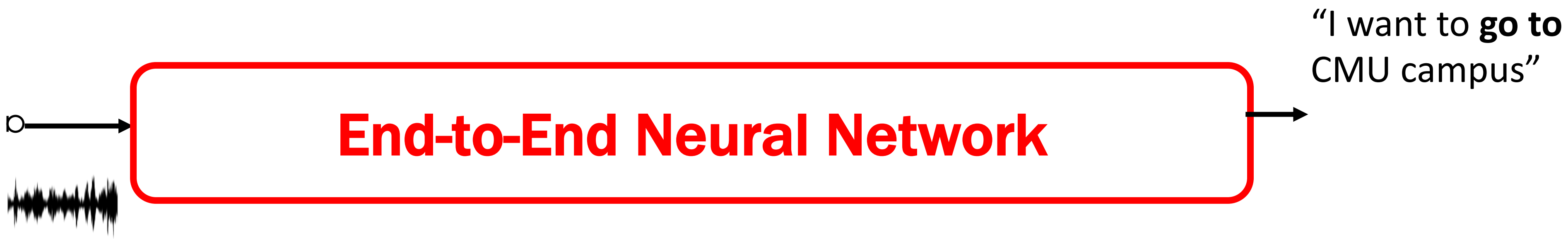
- Require **a lot of development** for an acoustic lexicon, a language model, and finite-state-tr
- Require linguistic resources
- Difficult to build ASR systems for non-experts

HMM-based speech recognition pipeline



- Require **a lot of development** for an acoustic model, a pronunciation lexicon, a language model, and finite-state-transducer decoding
- Require linguistic resources
- Difficult to build ASR systems for **non-experts**

From pipeline to integrated architecture



- Train a deep network that directly maps speech signal to the target letter/word sequence
- Greatly simplify the complicated model-building/decoding process
- Easy to build ASR systems for new tasks **without expert knowledge**
- Potential to outperform conventional ASR by **optimizing the entire network** with a single objective function

Note that all items have pros and cos

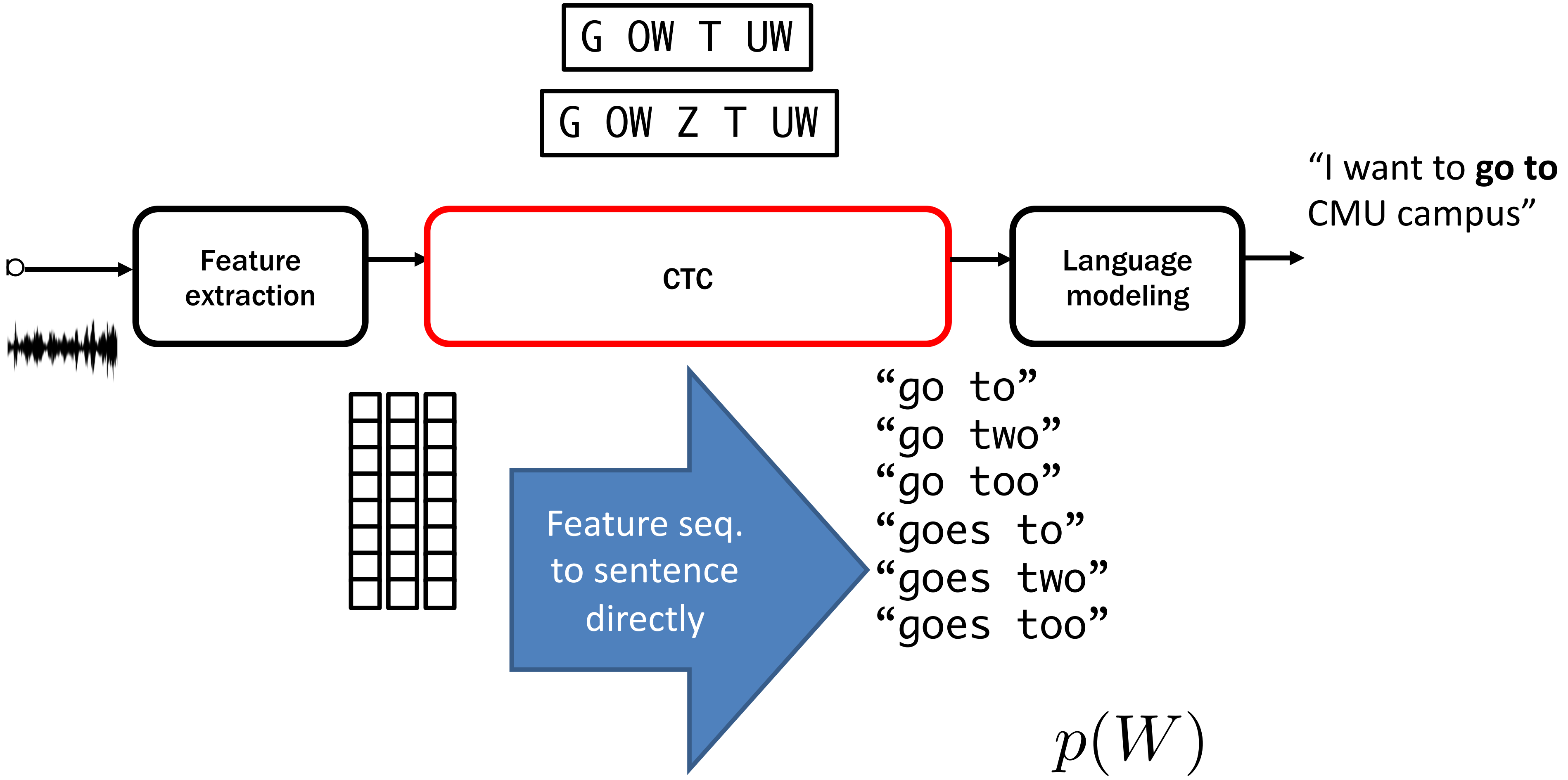
Speech recognition pipeline

- We will skip the feature extraction in most cases

Today’s agenda

- Introduction to end-to-end speech recognition
- HMM-based pipeline system
- Connectionist temporal classification (CTC)
- Attention-based encoder decoder
- Joint CTC/attention (Joint C/A)
- RNN transducer (RNN-T)

Speech recognition pipeline



Connectionist temporal classification (CTC)

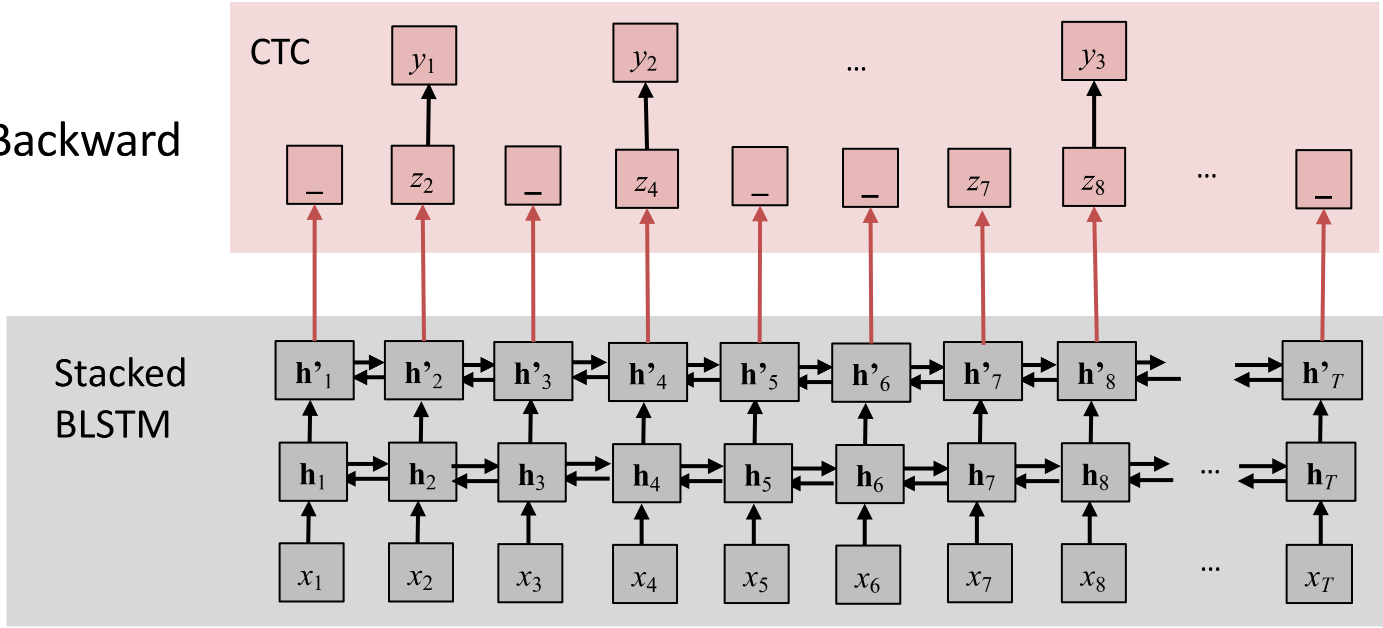
[Graves+ 2006, Graves+ 2014, Miao+ 2015]

- Use bidirectional RNNs (later self-attention) to predict frame-based labels including blanks
- Find alignments between *X* and *Y* using dynamic programming

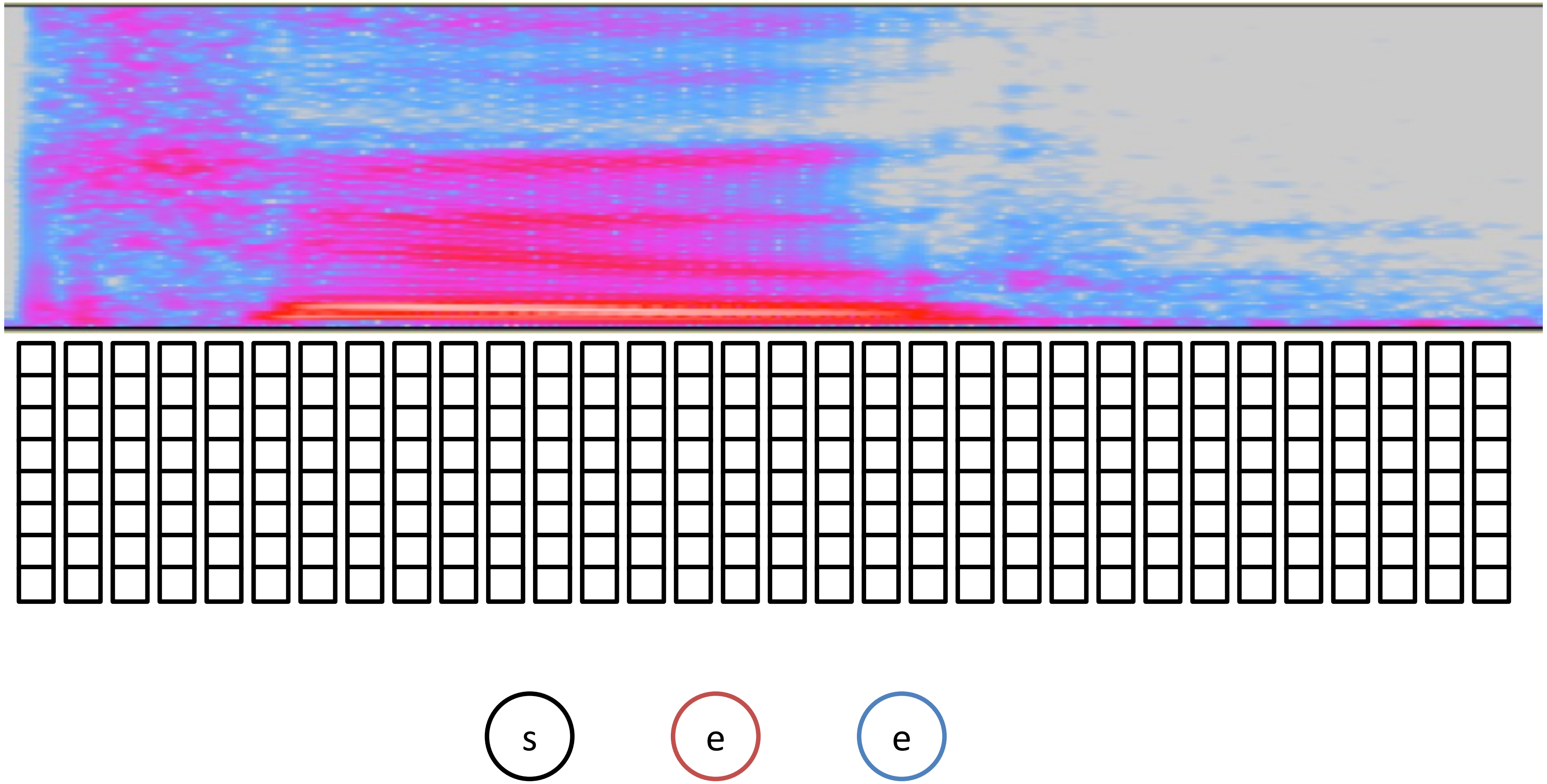
😄 Simple implementation (built-in & cudnn), on-line, fast

😓 Poor performance (conditional independence assumptions), limited applications

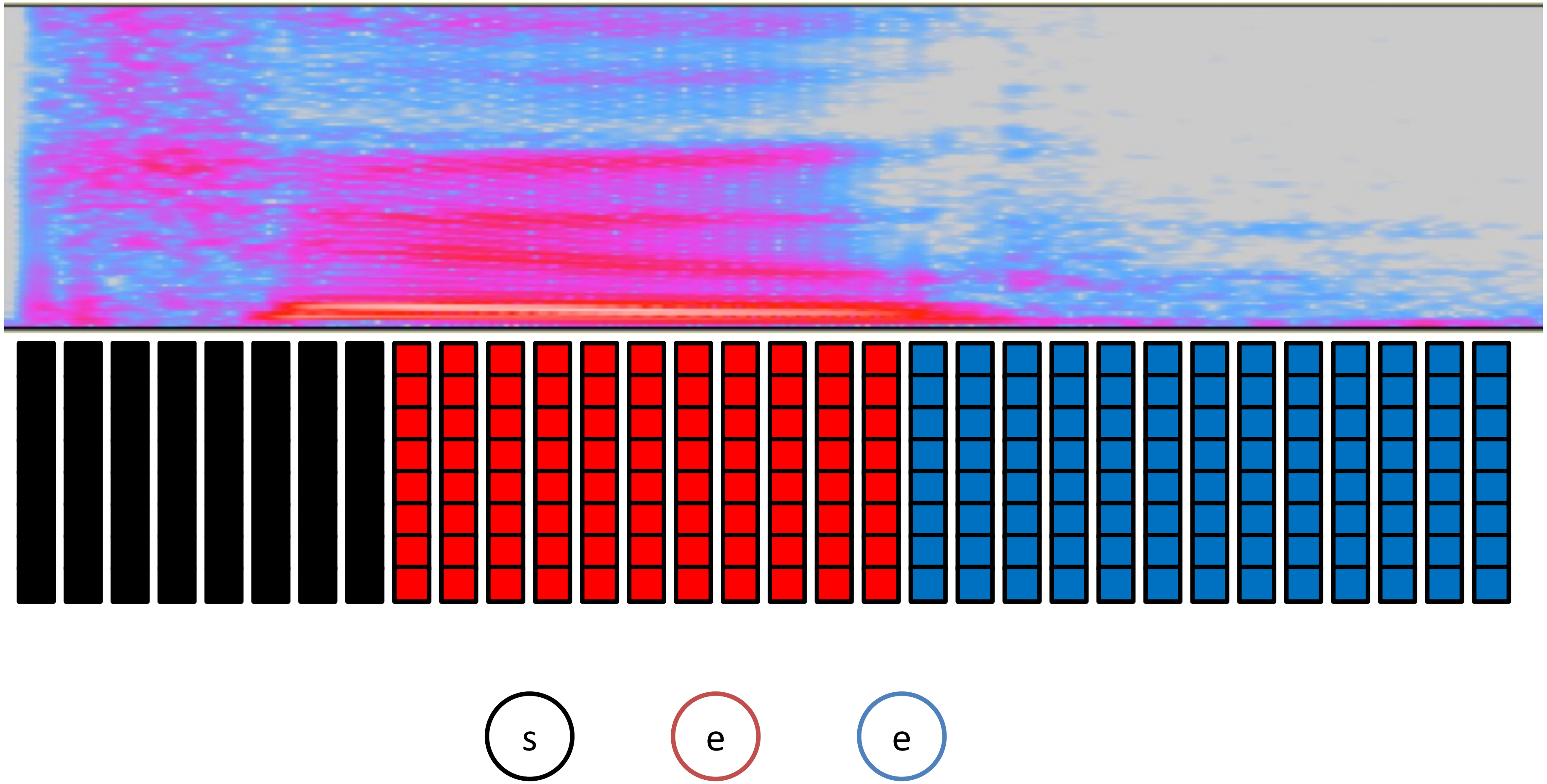
Forward-Backward algorithm



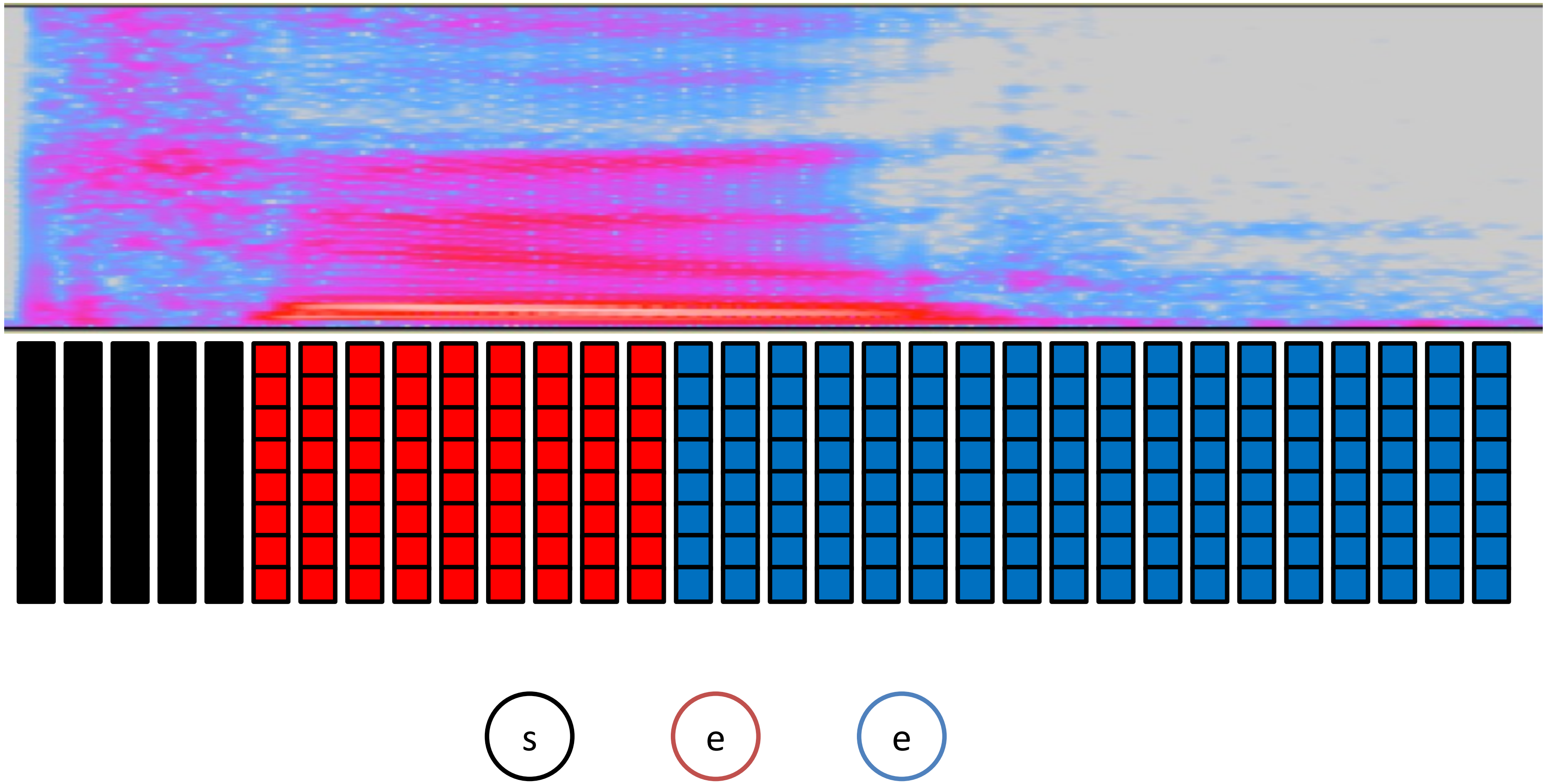
Alignments



Alignments

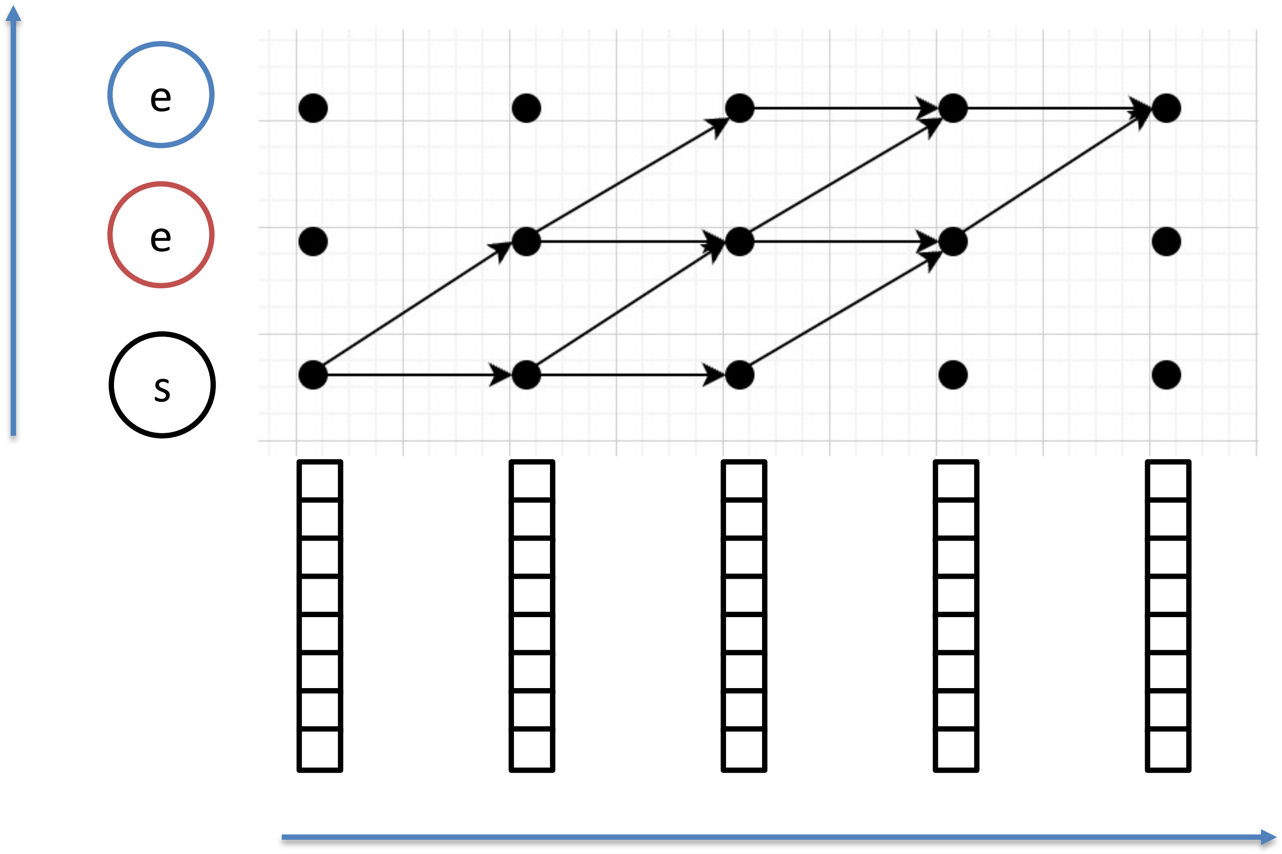


Alignments



How to represent all possible alignments?

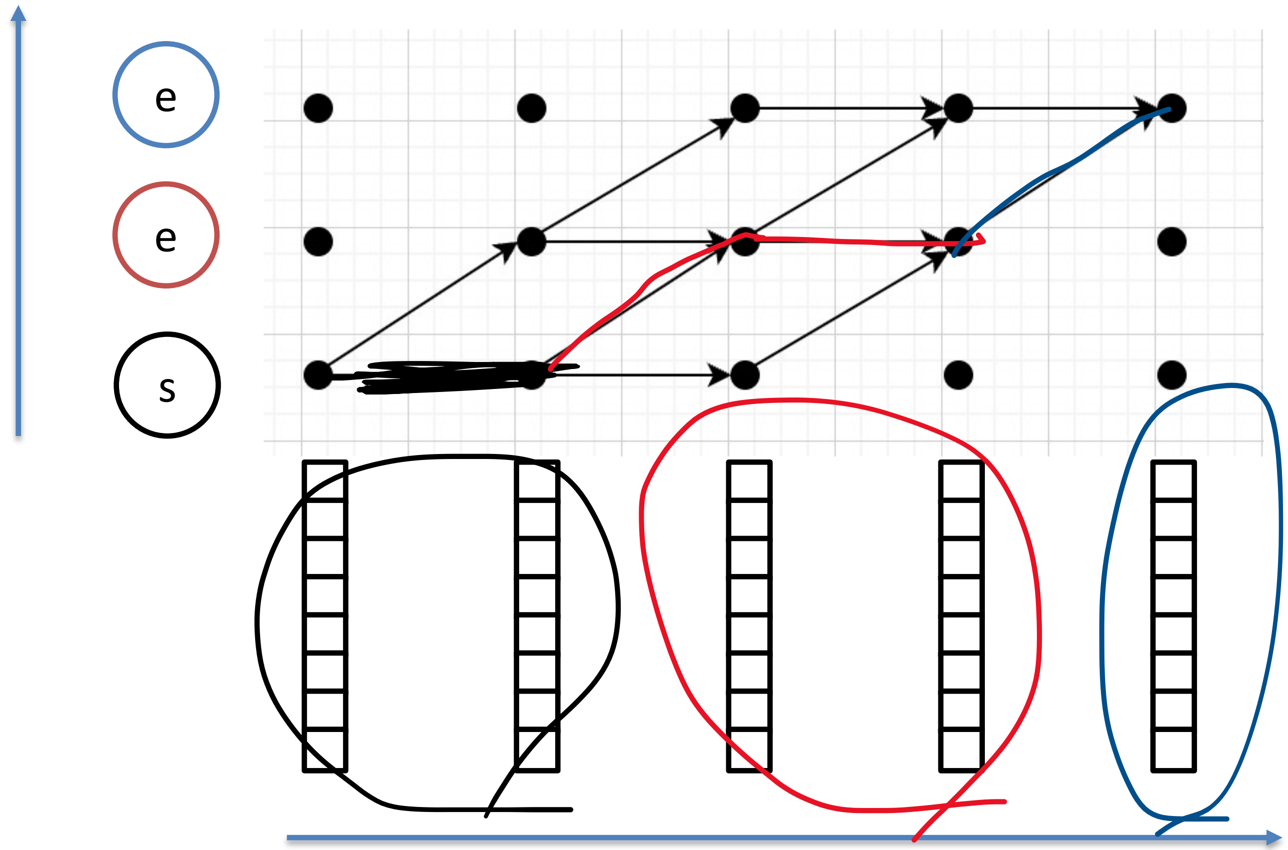
- Trellis



$p(\text{“}s\ e\ e\text{”} | o_1, o_2, o_3, o_4, o_5)$

How to represent all possible alignments?

- Trellis



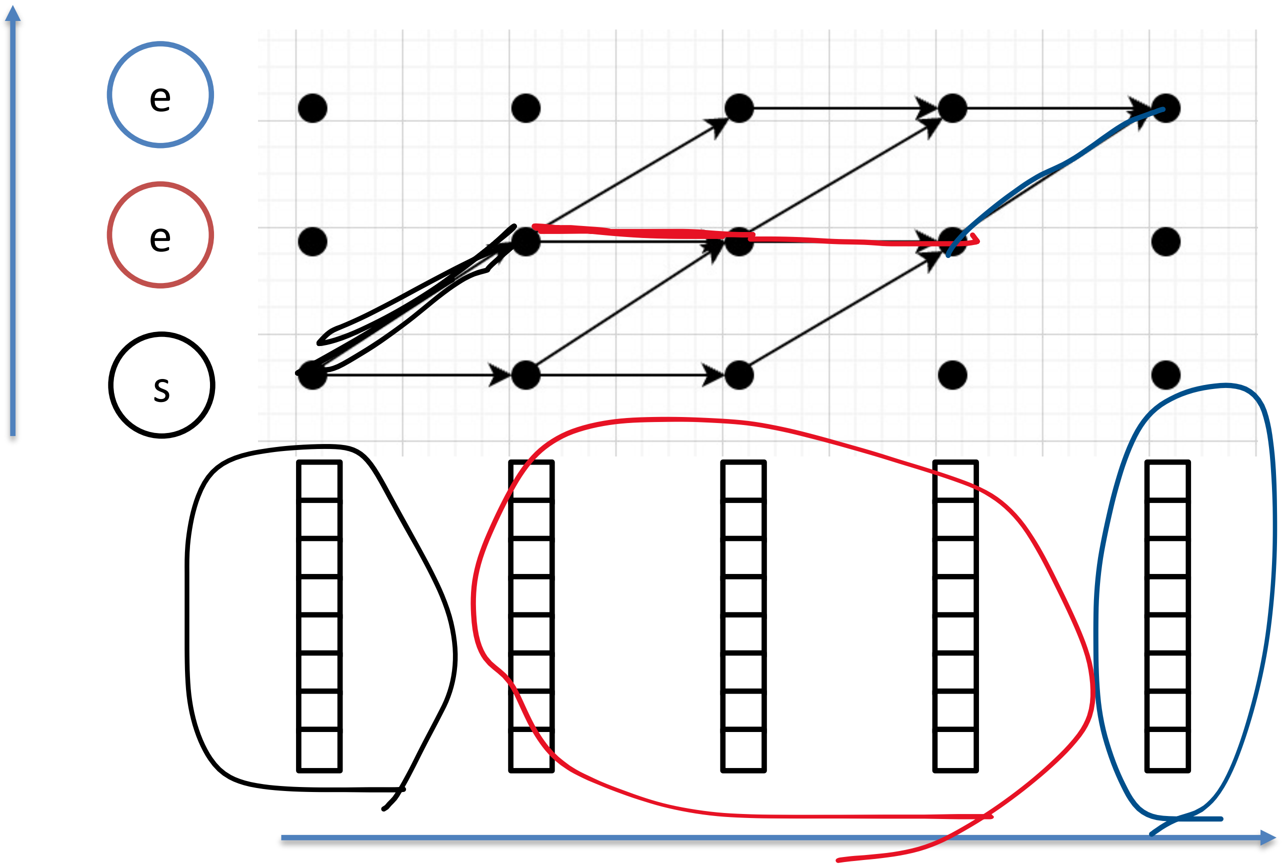
$$p(\text{“}s\ e\ e\text{”}|o_1, o_2, o_3, o_4, o_5)$$

$$p(\text{“}s\text{”}|o_1, o_2)p(\text{“}e\text{”}|o_3, o_4)p(\text{“}e\text{”}|o_5)$$

- This is derived by using the conditional independence assumptions
- To compute the factorized probability, we also introduce a blank symbol

How to represent all possible alignments?

- Trellis

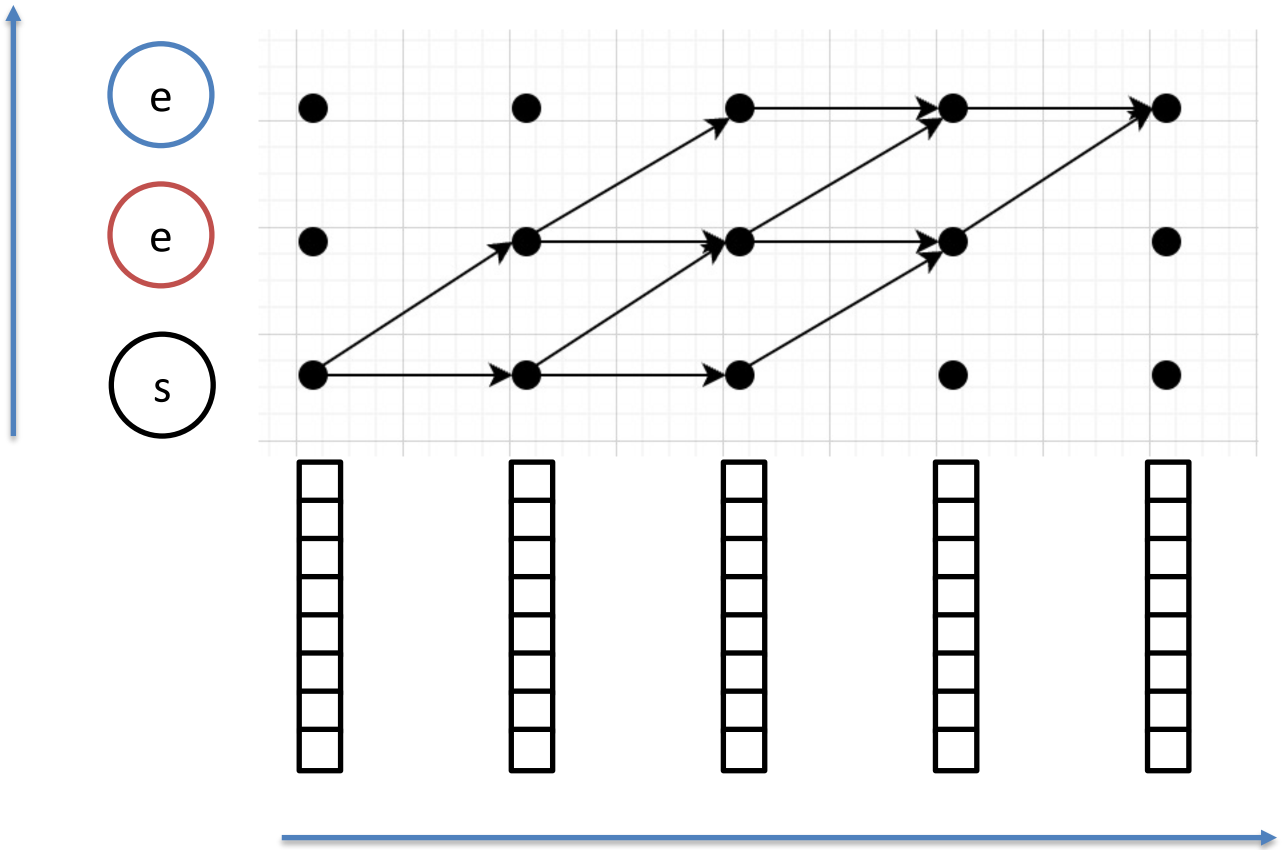


$p(\text{“}s\ e\ e\text{”} | o_1, o_2, o_3, o_4, o_5)$

$p(\text{“}s\text{”}|o_1)p(\text{“}e\text{”}|o_2, o_3, o_4)p(\text{“}e\text{”}|o_5)$

How to represent all possible alignments?

- Trellis



$$p(\text{“}s\ e\ e\text{”}|o_1, o_2, o_3, o_4, o_5)$$

We can compute the probability with all possible paths based on **dynamic programming**

HMM vs. CTC

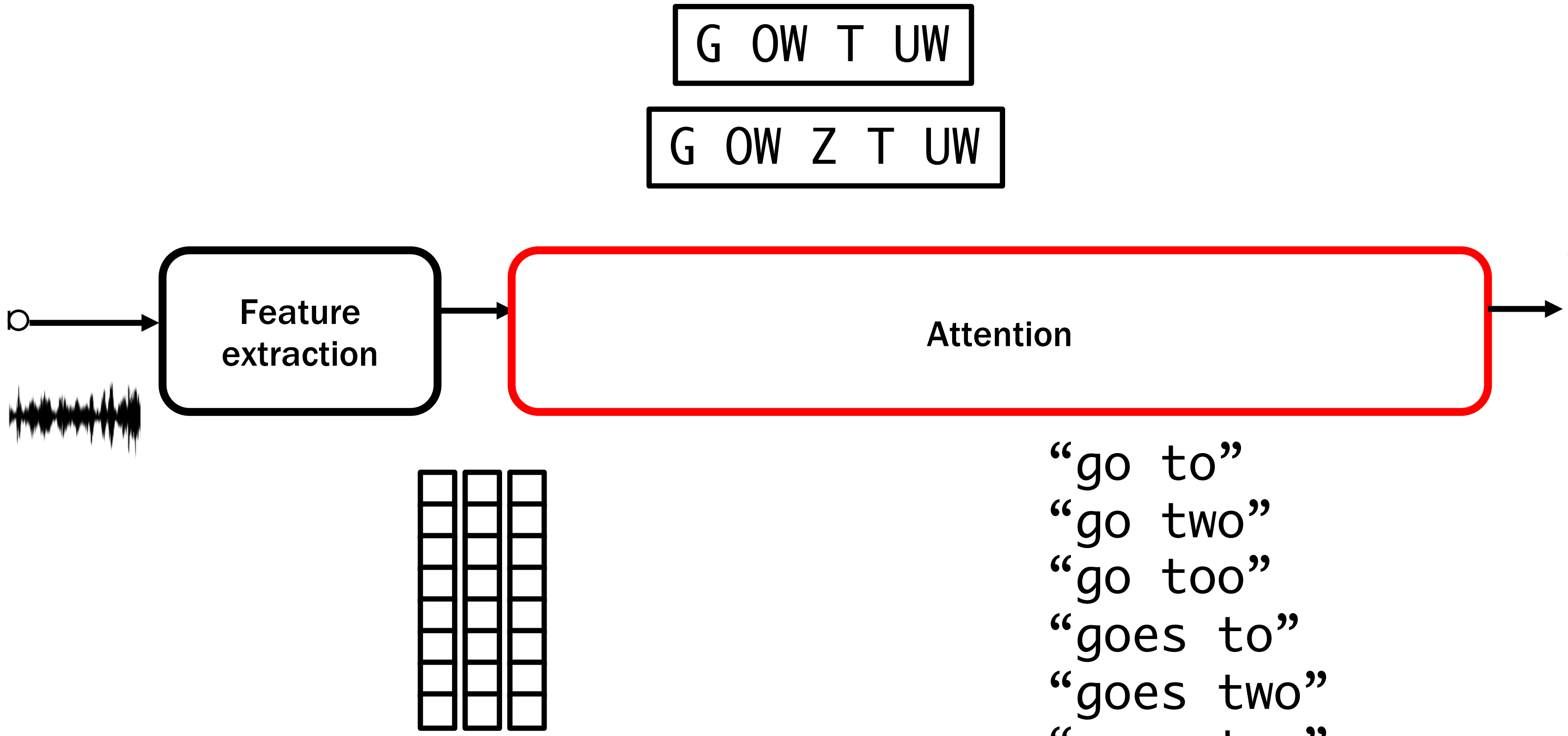
- Conditional independence assumptions
- Language models
- Use of pronunciation lexicon information
- Implementation

Let’s discuss the difference

Today's agenda

- Introduction to end-to-end speech recognition
- HMM-based pipeline system
- Connectionist temporal classification (CTC)
- Attention-based encoder decoder
- Joint CTC/attention (Joint C/A)
- RNN transducer (RNN-T)

Speech recognition pipeline



"I want to **go to** CMU campus"

"go to"
"go two"
"go too"
"goes to"
"goes two"
"goes too"

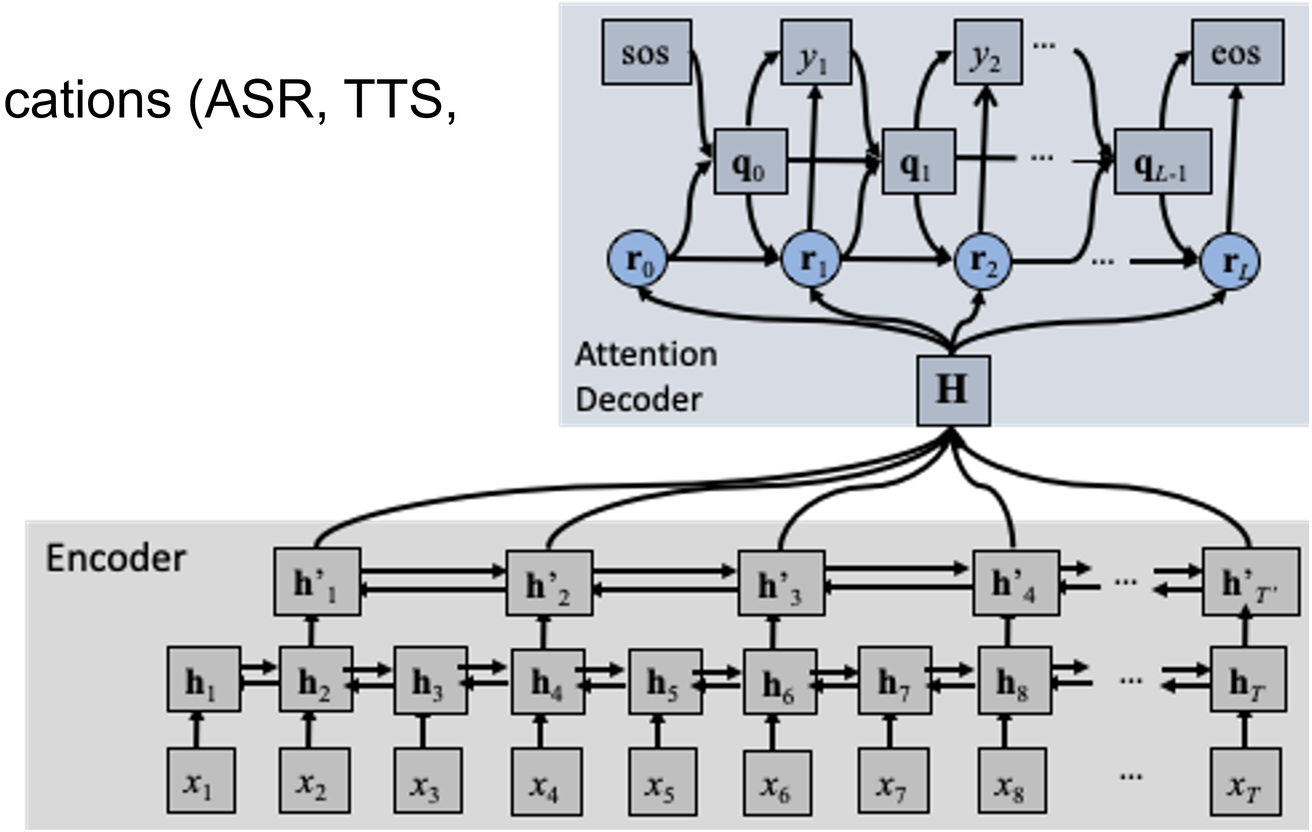
$$p(W)$$

Attention-based encoder decoder [Chorowski+ 2015, Chan+ 2016]

- Encoder: acoustic model, decoder: RNN language model, attention: align input and output labels
- Later transformer
- No conditional independence assumption

😄 Good performance but, a lot of applications (ASR, TTS, NMT)

😓 Too flexible alignment, off-line



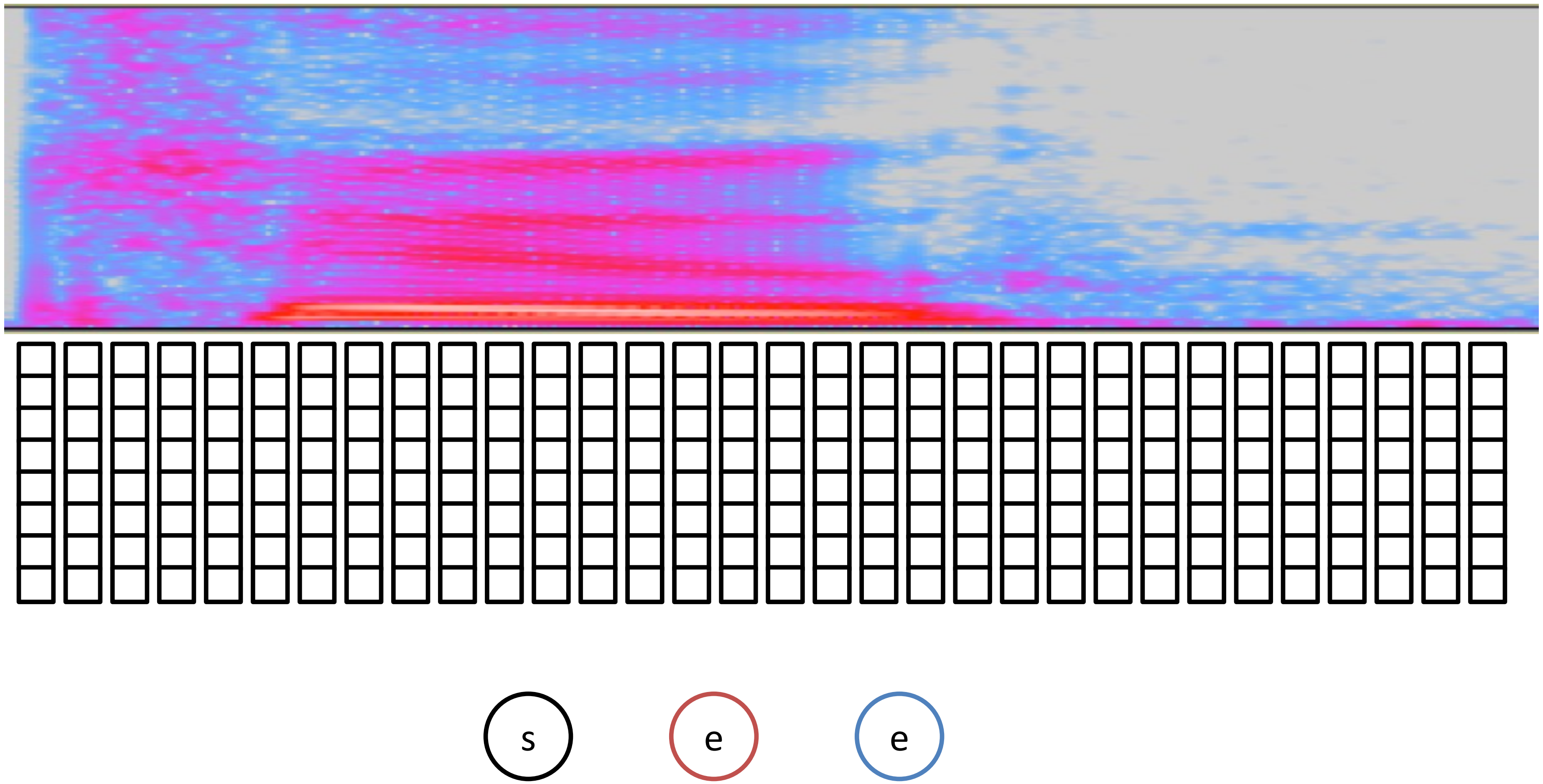
Source-target attention

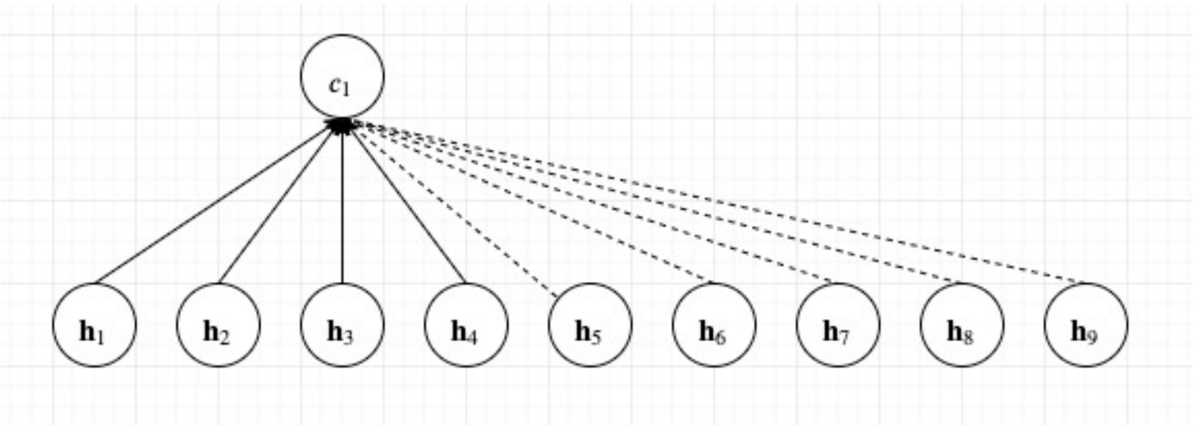
- Adjust different-length sequences based on the attention mechanism
  - If the encoder state at input frame t is $\boldsymbol{h}_t$, and we can compute a hidden state value in token i based on the following equation

$$\boldsymbol{c}_i = \sum_{t=1}^{T} a_{it} \boldsymbol{h}_t$$

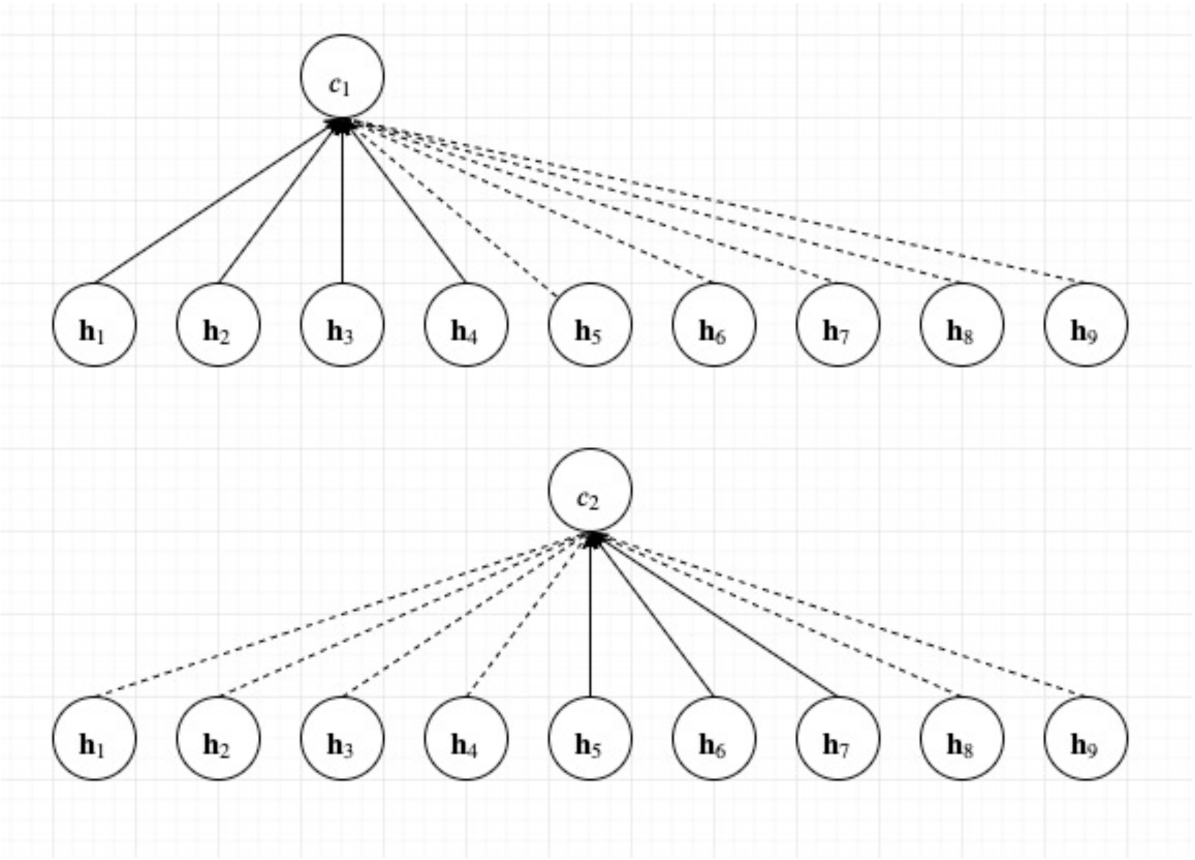
  - a_{it}: attention weight obtained by a neural network
- Widely used in machine translation and other sequence-to-sequence applications in NLP

Alignments

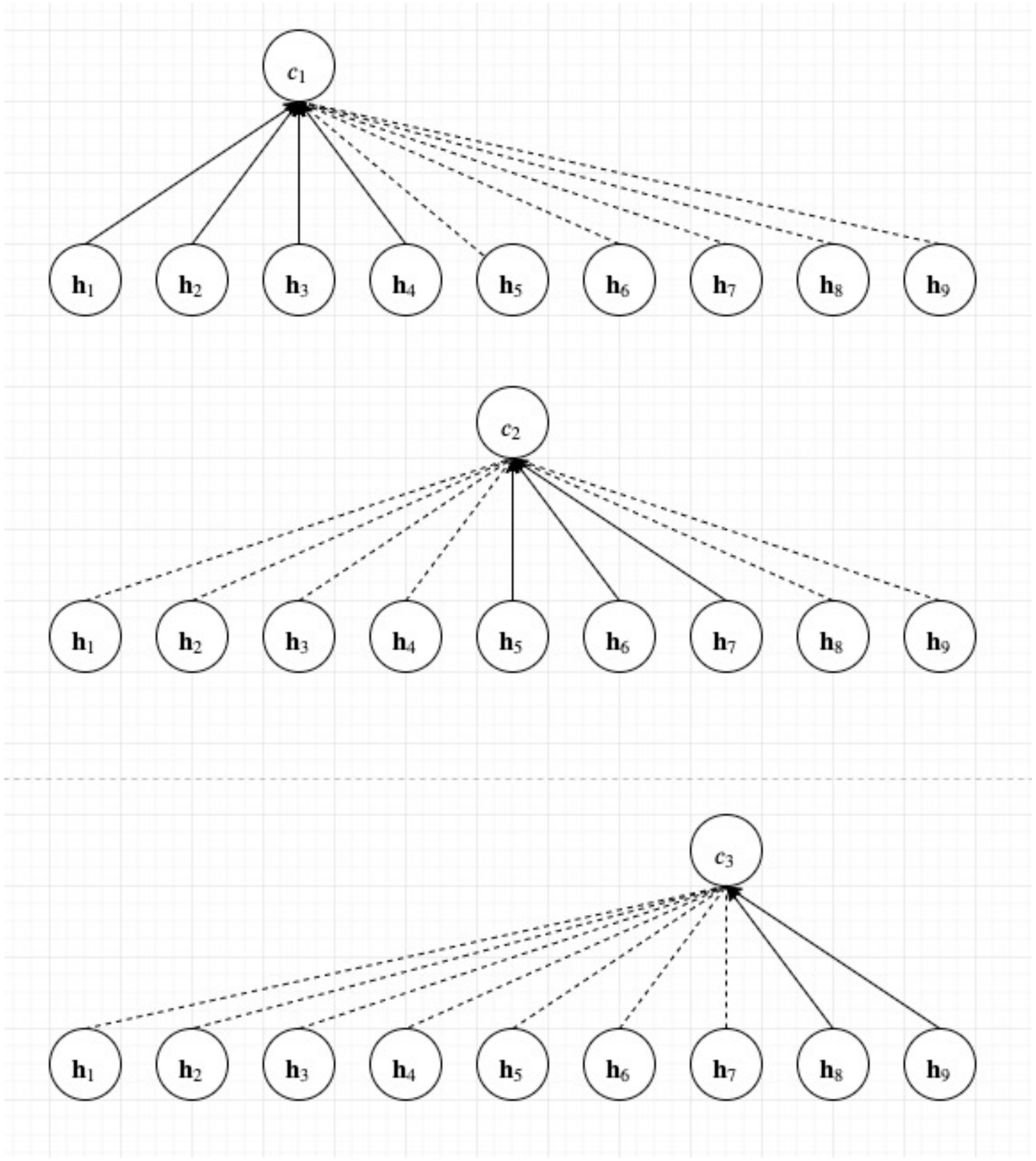




Normal arrow: high probability
Dashed arrow: low probability

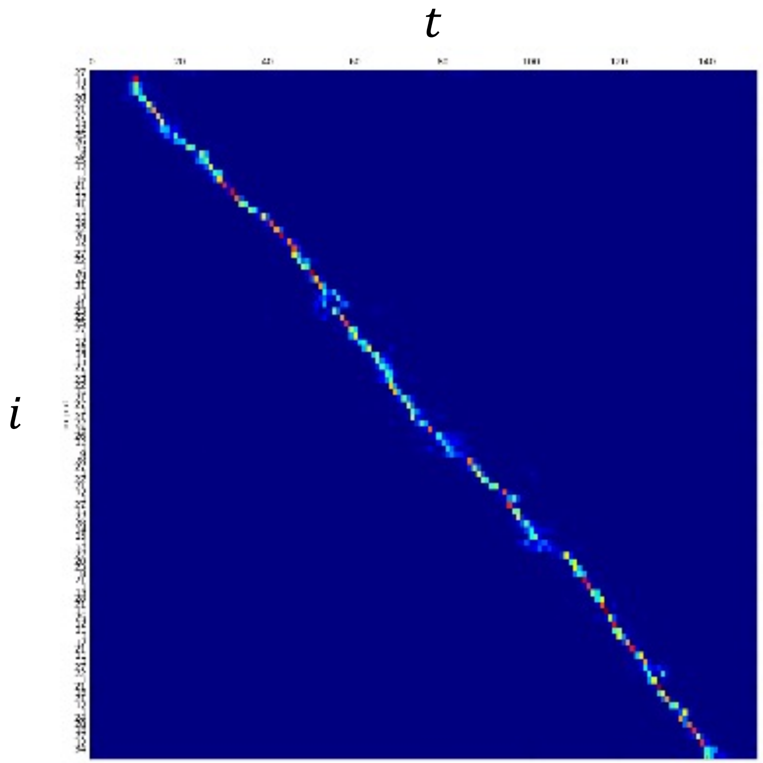


Normal arrow: high probability
Dashed arrow: low probability



Normal arrow: high probability
Dashed arrow: low probability

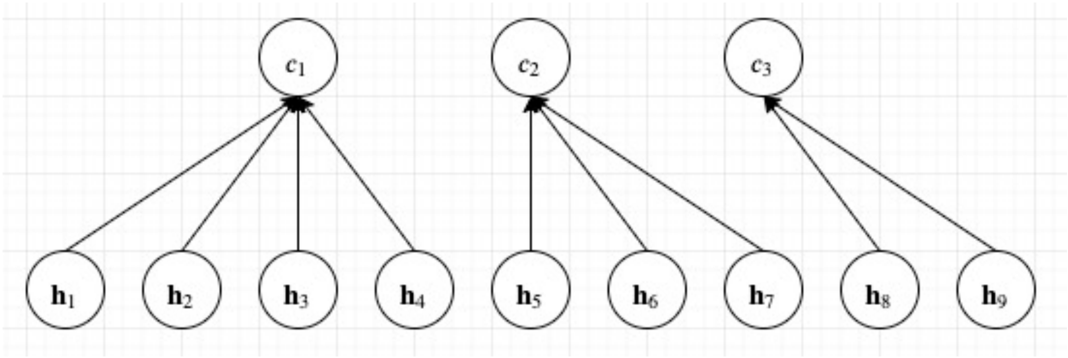
The attention mechanism performs a soft alignment



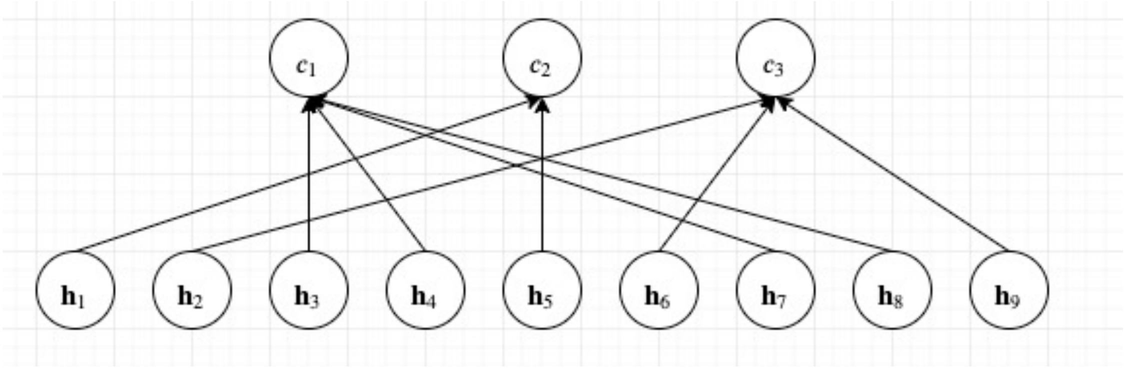
- $\boldsymbol{c}_i = \sum_{t=1}^{T} a_{it}\,\mathbf{h}_t$
- Attention weight a_{it} determines whether encoder $\mathbf{h}_t$ is assigned to a character c_i or not
  - $a_{it} \approx 0$: no assignment
  - $a_{it} \neq 0$: assigned

The attention mechanism performs a soft alignment

- There is no constraint for the alignment
- The order can be changed (good for machine translation, but it does not happen in speech recognition)

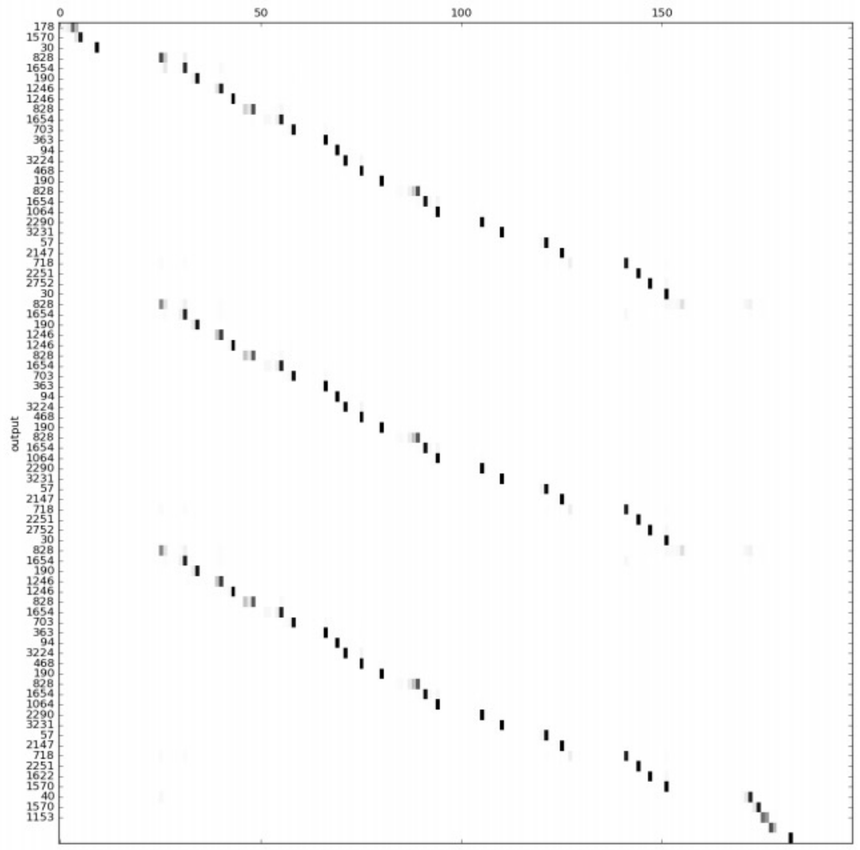


Monotonic



Non monotonic

Examples of wrong alignments



id: (20040717_152947_A010409_B010408-A-057045-057837)

Reference

但是如果你想想如果回到了过去你如果带着这个现在的记忆是不是很痛苦啊

MTL

Scores: (#Correctness #Substitution #Deletion #Insertion) 28 2 3 45

但是如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的机是不是很

. . .

We can avoid it by setting the constraint for the output length

HMM vs. CTC vs. Attention

- Conditional independence assumptions

- Language models

- Use of pronunciation lexicon information

- Implementation

Let’s discuss the difference

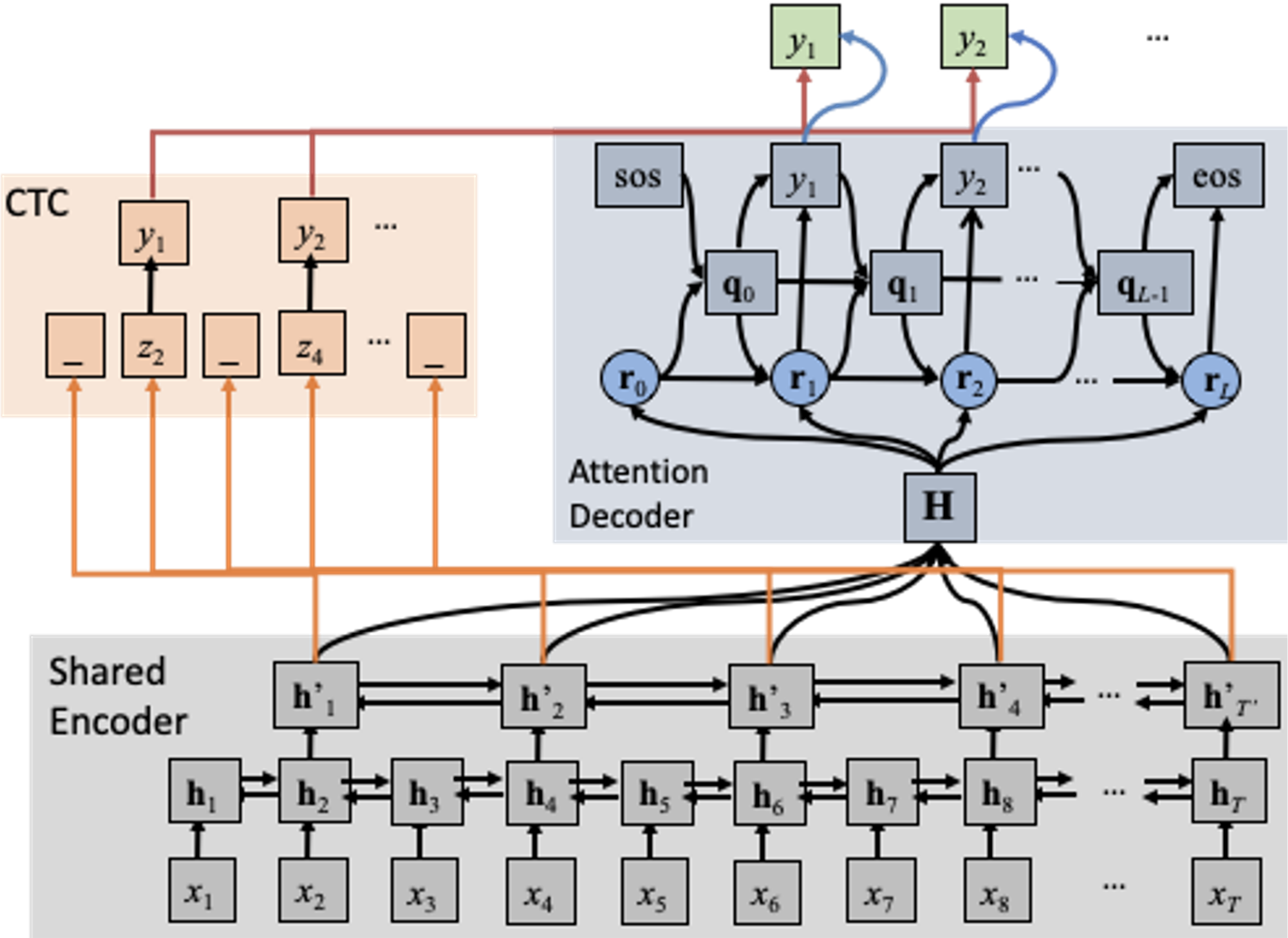
Joint CTC/attention (Joint C/A) [Kim+ 2017, Hori+ 2017]

- Combine CTC and attention during
  - training based on multi-task learning
  - inference based on score combination

😄 Very good performance with reasonable alignment

😓 Complicated implementation, off-line, limited applications

ESPnet uses joint CTC/attention since it does not have a tuning parameter during inference



Example of recovering insertion errors (HKUST)

id: (20040717_152947_A010409_B010408-A-057045-057837)

Reference

但是如果你想想如果回到了过去你如果带着这个现在的记忆是不是很痛苦啊

Hybrid CTC/attention (w/o joint decoding)

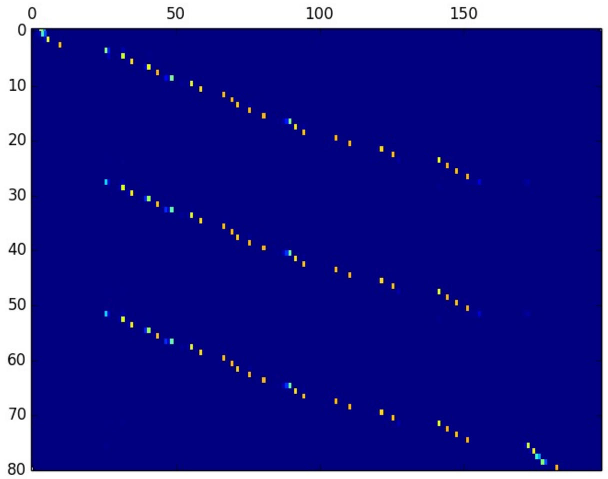
Scores: (#Correctness #Substitution #Deletion #Insertion) 28 2 3 45

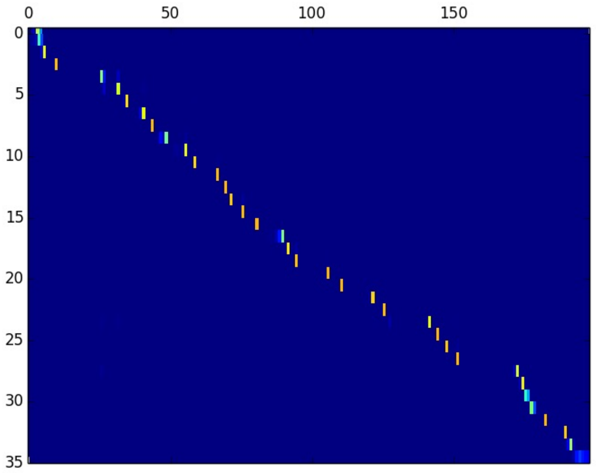
但是如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的机是不是很・・・

w/ Joint decoding

Scores: (#Correctness #Substitution #Deletion #Insertion) 31 1 1 0

HYP: 但是如果你想想如果回到了过去你如果带着这个现在的・机是不是很痛苦啊





Example of recovering deletion errors (CSJ)

id: (A01F0001_0844951_0854386)

Reference

ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る 為 に 超 小 型 マ イ
ク ロ ホ ン お よ び 生 体 ア ン プ を コ ウ モ リ に 搭 載 す る こ と を 考 え て お り ま す そ う す
る こ と に よ っ て

Hybrid CTC/attention (w/o joint decoding)

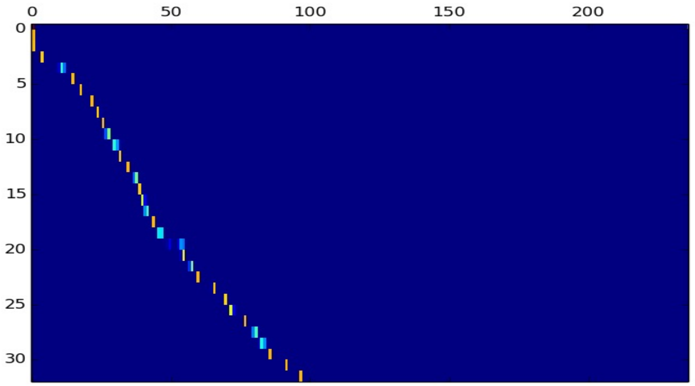
Scores: (#Correctness #Substitution #Deletion #Insertion) 30 0 47 0

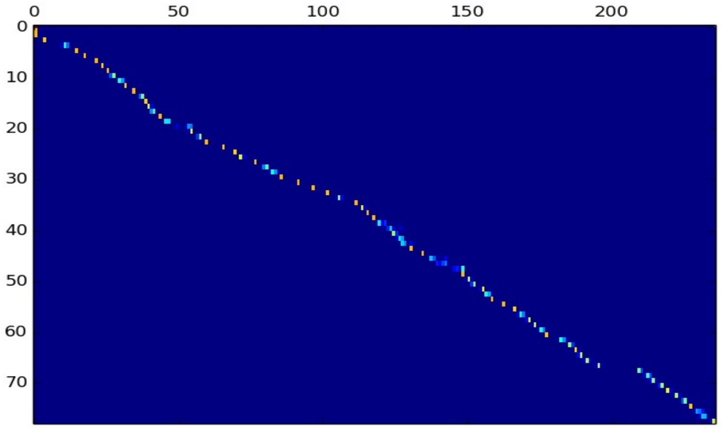
ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る
為 ・
・ ・ ・ ・ ・ ・ ・ ・ ・ ・ に ・ ・ ・

w/ Joint decoding

Scores: (#Correctness #Substitution #Deletion #Insertion) 67 9 1 0

ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る 為 に 長 国 型 マ イ
ク ロ ホ ン お ・ い く 声 単 位 方 を コ ウ モ リ に 登 載 す る こ と を 考 え て お り ま す そ う す
る こ と に よ っ て





Today’s agenda

- Introduction to end-to-end speech recognition
- HMM-based pipeline system
- Connectionist temporal classification (CTC)
- Attention-based encoder decoder
- Joint CTC/attention (Joint C/A)
- RNN transducer (RNN-T)

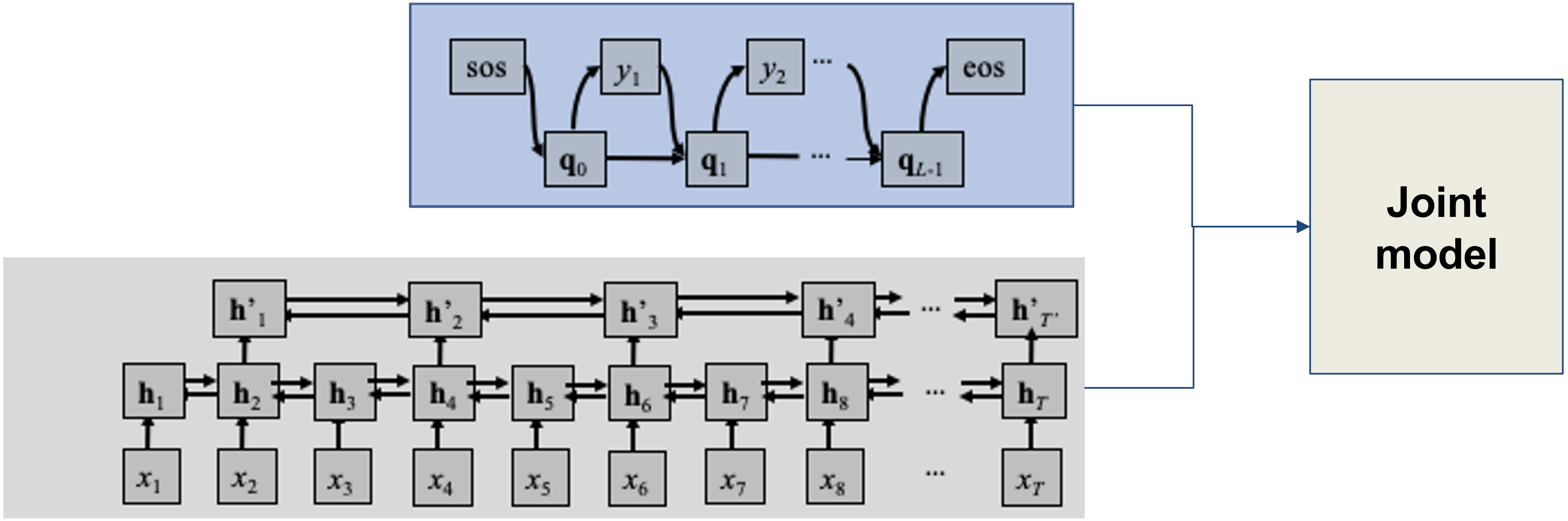
RNN-transducer [Graves+ 2013]

- Extension of CTC by considering previous output dependency
- Combine input RNN and auto-regressive output RNN to provide a joint distribution
    - Joint model can handle this combination

😄 Good performance with reasonable alignment, on-line

😓 lower performance than attention, limited applications

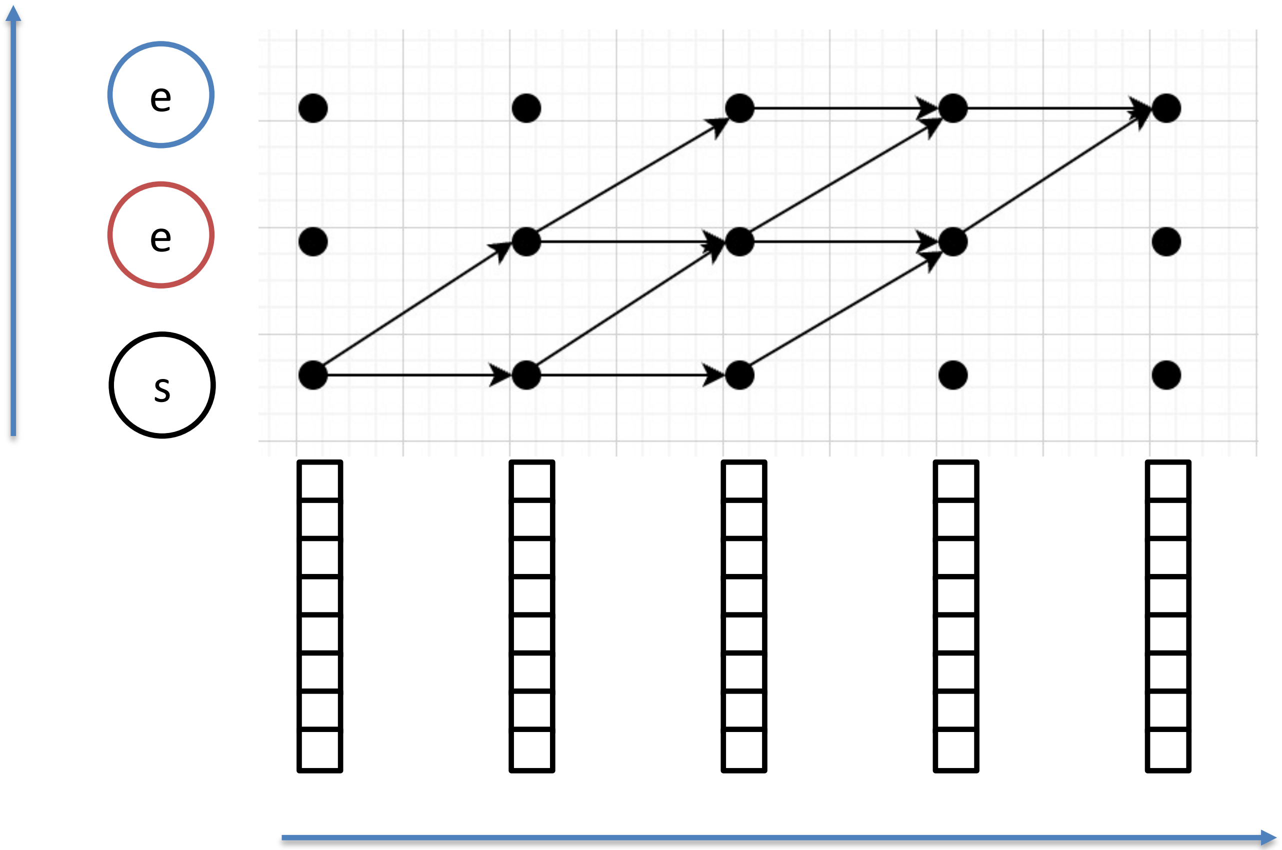
Now, widely used especially in industry



$$p(y_n | \mathbf{x}_t, y_{n-1})$$

How to represent all possible alignments?

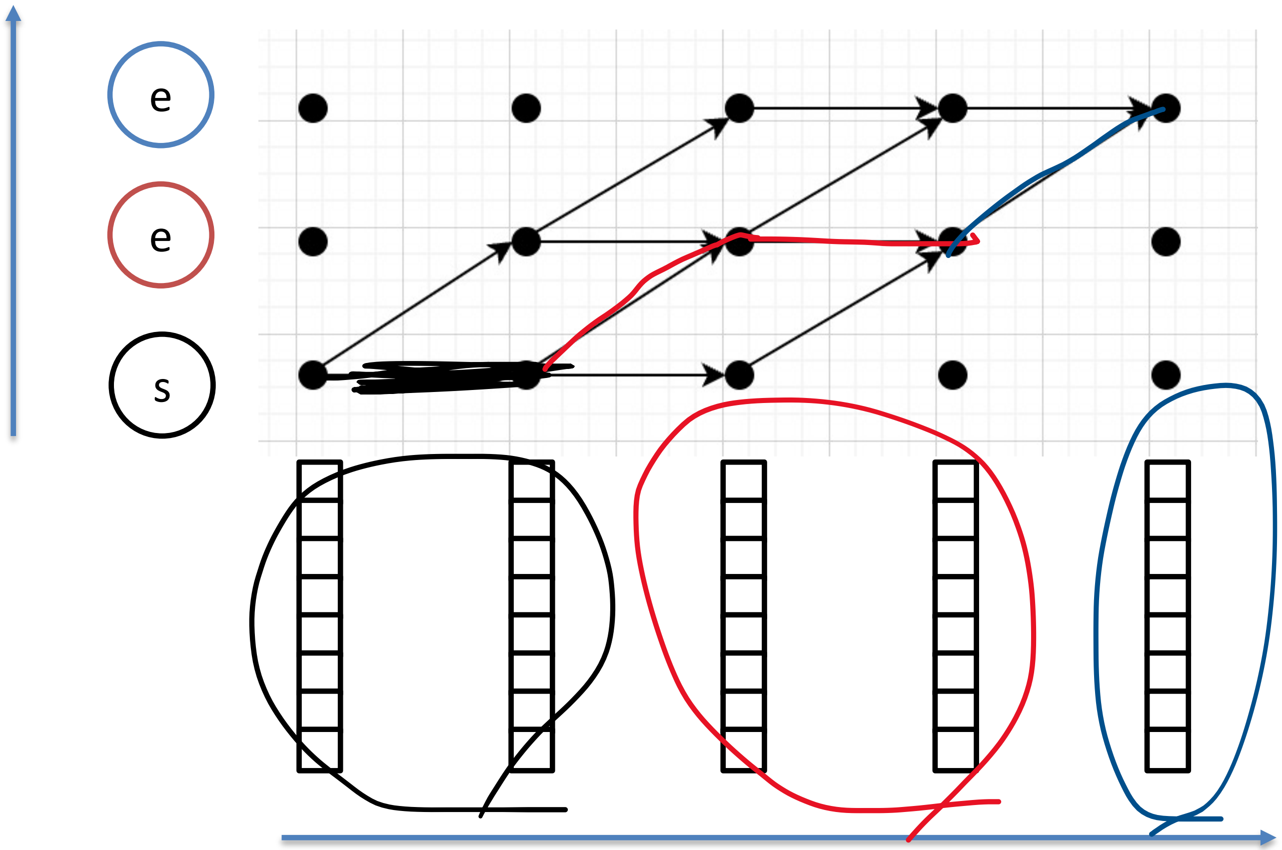
- Trellis



$$p(\text{“}s\ e\ e\text{”}|o_1, o_2, o_3, o_4, o_5)$$

How to represent all possible alignments?

- Trellis



$$p(\text{"}s\ e\ e\text{"}|o_1, o_2, o_3, o_4, o_5)$$

$$p(\text{"}s\text{"}|o_1, o_2)p(\text{"}e\text{"}|o_3, o_4)p(\text{"}e\text{"}|o_5)$$

$$p(\text{"}s\text{"}|o_1, o_2)$$
$$p(\text{"}e\text{"}|\text{"}s\text{"}, o_3, o_4)p(\text{"}e\text{"}|\text{"}s\text{"}, \text{"}e\text{"}, o_5)$$

- We consider the history (relax the conditional independence assumptions)
- We can still compute it by using dynamic programming

HMM vs. CTC vs. Attention vs. RNN-T

- Conditional independence assumptions
- Language models
- Use of pronunciation lexicon information
- Implementation

Let’s discuss the difference

Seq2seq end-to-end ASR

$$\boxed{X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N)}$$

$$f(\cdot)$$

Direct seq2seq mapping function

1. HMM-based pipeline system
2. Connectionist temporal classification (CTC)
3. Attention-based encoder decoder
4. Joint CTC/attention (Joint C/A)
5. RNN transducer (RNN-T)

ESPnet

References

- **CTC:** 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.
- **Attention:** Chorowski, Jan K., et al. "Attention-based models for speech recognition." Advances in neural information processing systems 28 (2015).
- **Joint CTC/Attention:** Watanabe, Shinji, et al. "Hybrid CTC/attention architecture for end-to-end speech recognition." IEEE Journal of Selected Topics in Signal Processing 11.8 (2017): 1240-1253.
- **RNN transducer:** A. Graves, “Sequence transduction with recurrent neural networks,” in ICML Representation Learning Workshop, 2012.

Discussion

Please discuss your current status of assignment 3. Please pick up one or two items from the following items.

- Which language did you choose, and why?
  - Please share the information of how many hours of training data? What kind of scripts are used? What kind of text/audio pre-processing you're performing? etc.
- What is your computing environment?
  - Using AWS? Your Lab's computing resources?
  - OS, GPU types, cudnn versions, python version, pytorch version, etc.
- Which stage did you finish?
  - What were the difficulties and what were the things that should be good to be shared with the others?
  - What issues are you currently facing on?
- What is the role in your team, if your team member is also in the discussion group?
- Any other issues, status, and TIPS that you want to report