Today's agenda

- Multilingual End-to-end ASR for Incomplete Data (Team Leader: Takaaki Hori)
- 08:30 AM – 09:00 AM Continental Breakfast
- 09:00 AM – 10:30 AM End-to-end speech recognition (Shinji Watanabe)
- 10:30 AM – 10:50 AM Break
- 10:50 AM – 12:10 PM Advanced topics in end-to-end speech recognition (Takaaki Hori)
- 12:10 PM – 01:00 PM Lunch Break
- 01:00 PM – 01:30 PM Computer Setup Time
- 01:30 PM – 03:00 PM Brief introduction of end-to-end speech processing toolkit ESPnet (Shinji Watanabe and Takaaki Hori)
- 03:00 PM – 03:30 PM Coffee Break (ECE lounge)
- 03:30 PM – 05:00 PM Building end-to-end ASR using ESPnet (Shinji Watanabe and Takaaki Hori)

2018 JHU Summer School on Human Language Technology
Wednesday, June 20, 2018

End-to-end speech recognition

Shinji Watanabe

Table of contents

- Preliminaries
- Connectionist Temporal Classification (CTC)
- Attention based encoder-decoder

Notation

Type	Font, case	Latex command	Looks like
Scalar variable	Italic font, lower case	`$x$`	x
Vector variable	Bold font, lower case	`$\mathbf{x}$`	$\mathbf{x}$
Matrix variable	Bold font, upper case	`$\mathbf{X}$`	$\mathbf{X}$

Notation

- Please specify the domain of variables
  - D-dimensional continuous vector: $\mathbf{o} \in \mathbb{R}^D$
  - $(D \times D)$-dimensional matrix: $\mathbf{\Sigma} \in \mathbb{R}^{D \times D}$
  - Word with vocabulary $\mathcal{V}$: $w \in \mathcal{V}$
- Set: calligraphic font, upper case, a set of elements are represented with curly brackets

$$\mathcal{V} = \{\text{"one"}, \text{"two"}, \text{"three"}, \cdots\}$$

- Sequence: italic font, upper case, a sequence of elements are represented with round brackets

$$O = (\mathbf{o}_1, \mathbf{o}_2, \cdots) \qquad O = (\mathbf{o}_t \in \mathbb{R}^D | t = 1, \cdots, T)$$

My recommendation

Speech recognition cases

- T-length speech feature sequence (D-dimensional vector)

$$O = (\mathbf{o}_t \in \mathbb{R}^D | t = 1, \ldots, T)$$

- N-length word sequence with vocabulary $\mathcal{V}$

$$W = (w_n \in \mathcal{V} | n = 1, \ldots, N)$$

Probabilistic rules

- **Product rule**

$$p(x|y)p(y) = p(x, y)$$

- **Sum rule**

$$p(y) = \sum_x p(x, y)$$

- **Conditional independence assumption**

$$p(x|y, z) = p(x|z) \qquad p(x, y|z) = p(x|z)p(y|z)$$

Other rules

- **Bayes rule**

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_x p(y|x)p(x)}$$

- **Probabilistic chain rule**

$$p(x_1, \cdots, x_N) = \prod_{n=1}^{N} p(x_n|x_{1:n-1}) \quad \text{where} \quad p(x_1|x_{1:0}) = p(x_1)$$

- Both are derived with a combination of the product and sum rules

Why we use a probability?

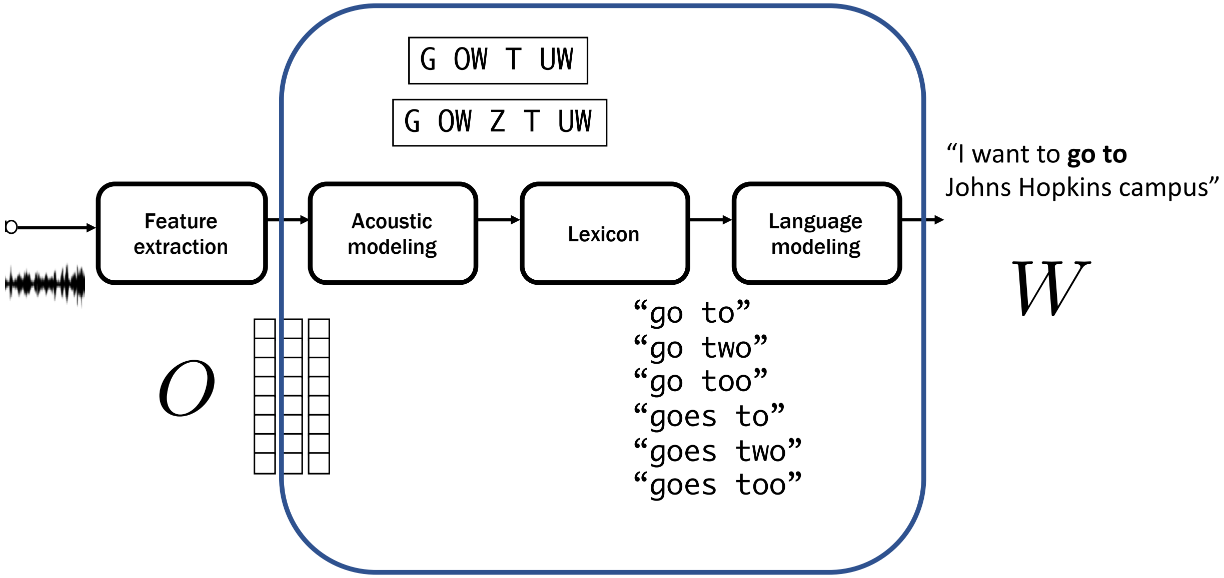
- It is intuitive
- The value is reasonably bounded, e.g.,

$$\sum_x p(x) = 1, \quad p(x) \geq 0 \ \forall x$$

- Easy to formulate. We basically only remember the three rules.

Speech recognition with a probabilistic formulation

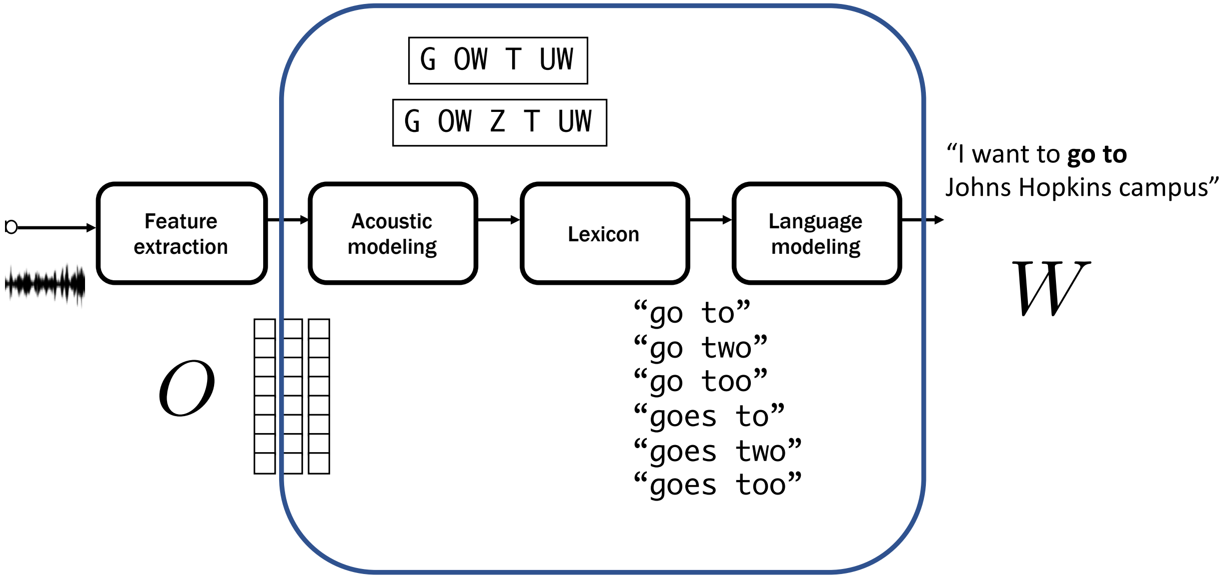
- Let $O = (\mathbf{o}_t \in \mathbb{R}^D | t = 1, \dots, T)$ be a T-length speech feature sequence (D-dimensional vector)
- Let $W = (w_n \in \mathcal{V} | n = 1, \dots, N)$ be a N-length word sequence with vocabulary $\mathcal{V}$



Speech recognition with a probabilistic formulation

- **MAP decision theory**: Estimate the most probable word sequence $\hat{W}$ among all possible word sequences $\mathcal{W}$ (I'll omit the domain sometimes)

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}}\, p(W|O)$$



Sequence to sequence mapping was really difficult problems!!!

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

- Noisy channel model
  - Regarding O as a probabilistic variable (noisy observation)
  - Use the product rule

$$\underset{W}{\operatorname{argmax}}\, p(W|O) = \underset{W}{\operatorname{argmax}} \frac{p(O|W)p(W)}{p(O)}$$

$$= \underset{W}{\operatorname{argmax}}\, p(O|W)p(W)$$

Likelihood Prior

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

- Noisy channel model

$$\underset{W}{\operatorname{argmax}}\, p(W|O) = \underset{W}{\operatorname{argmax}} \frac{p(O|W)p(W)}{p(O)}$$

$$= \underset{W}{\operatorname{argmax}}\, p(O|W)p(W)$$

- Solving generating process of noisy observations!!
- Still difficult to deal with them….

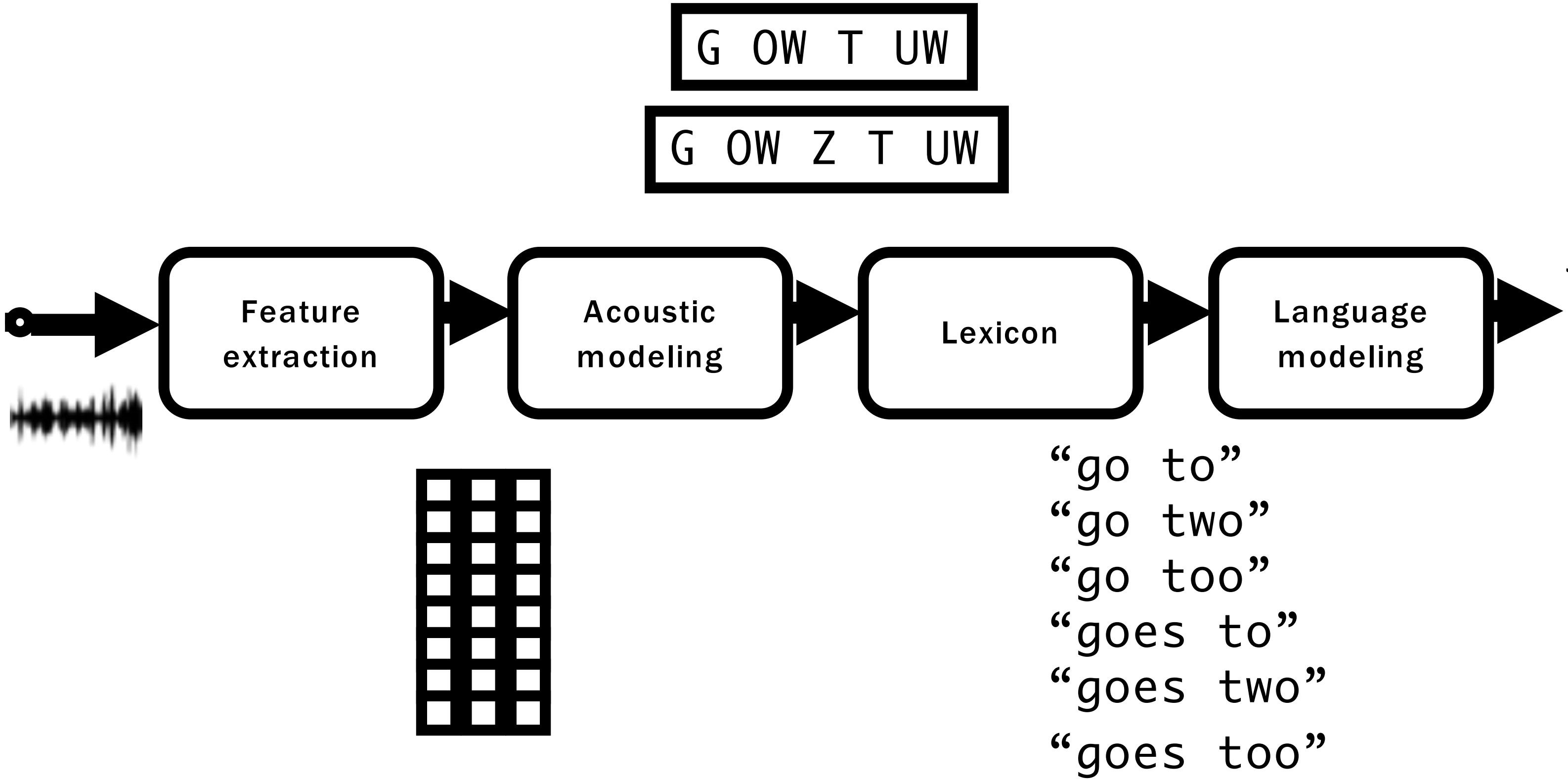
How to obtain the posterior $p(W|O)$

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

$$
\begin{aligned}
\underset{W}{\operatorname{argmax}}\, p(W|O) &= \underset{W}{\operatorname{argmax}} \sum_{L} p(W, L|O) \\
&= \underset{W}{\operatorname{argmax}} \sum_{L} p(O|L, W) p(L, W) \\
&= \underset{W}{\operatorname{argmax}} \sum_{L} p(O|L) p(L|W) p(W)
\end{aligned}
$$

Note: the right hand side does not hold the sum to one constraint

Speech recognition pipeline

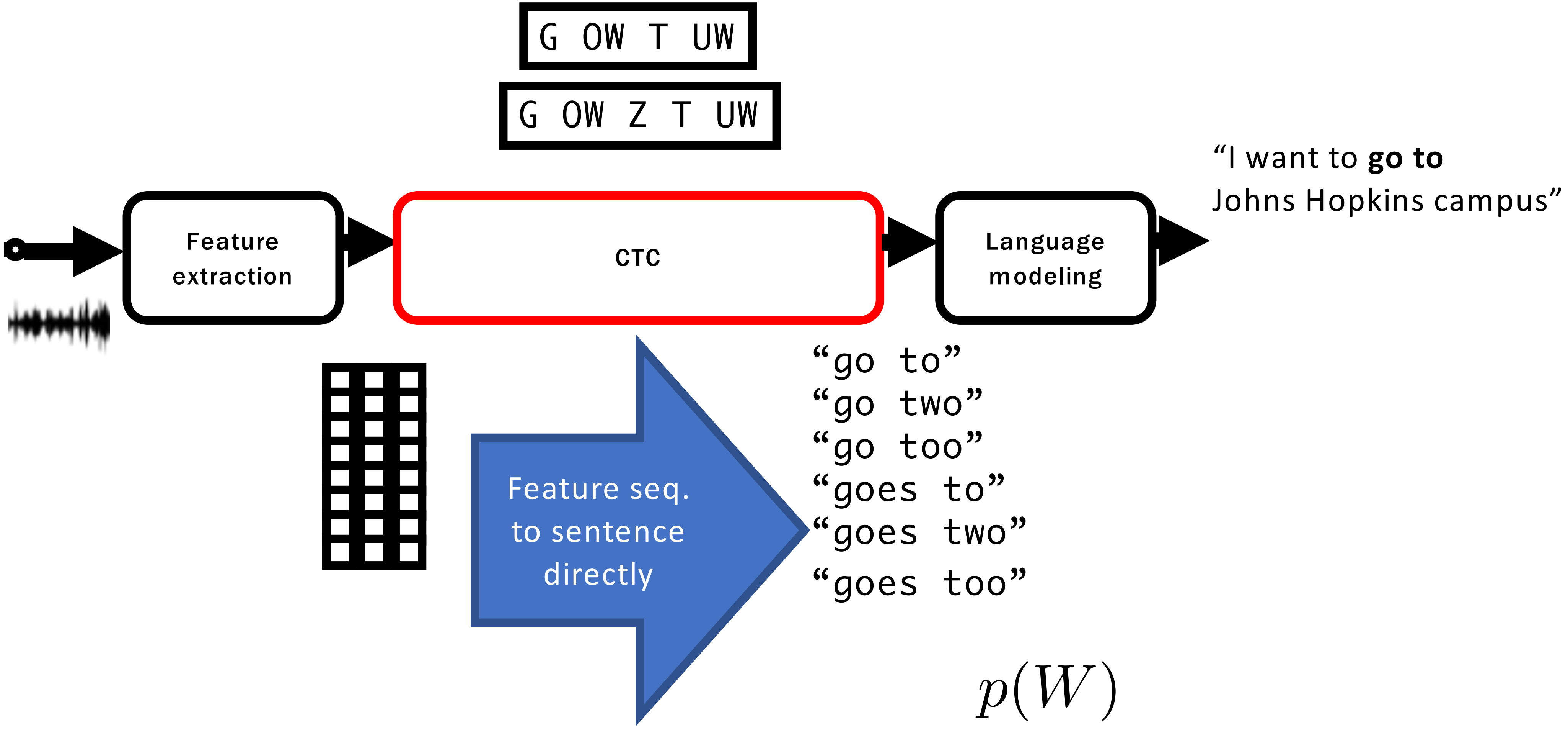


"I want to **go to** Johns Hopkins campus"

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

Table of contents

Speech recognition pipeline



Character seq. vs. word seq.

- Example: “I see”
  - $W = (w_i \in \{\text{“i”}, \text{“see”}, \dots.\} | i = 1, 2)$
  - $C = (c_j \in \mathbb{U} | j = 1, \dots, 5)$, where $\mathbb{U} = \{\text{“a”}, \text{“b”}, \text{“c”}, \text{“d“}, \text{“e”}, \dots\}$
- Low/zero count problem
  - Word “bitcoin” is not appeared in old WSJ sentences, but character seq. can cover it
- Semantic context, lexicon constraint
  - Word unit can handle them, but not in the character unit
- No word unit in some languages
  - Some languages do not have word boundaries (no explicit word units)

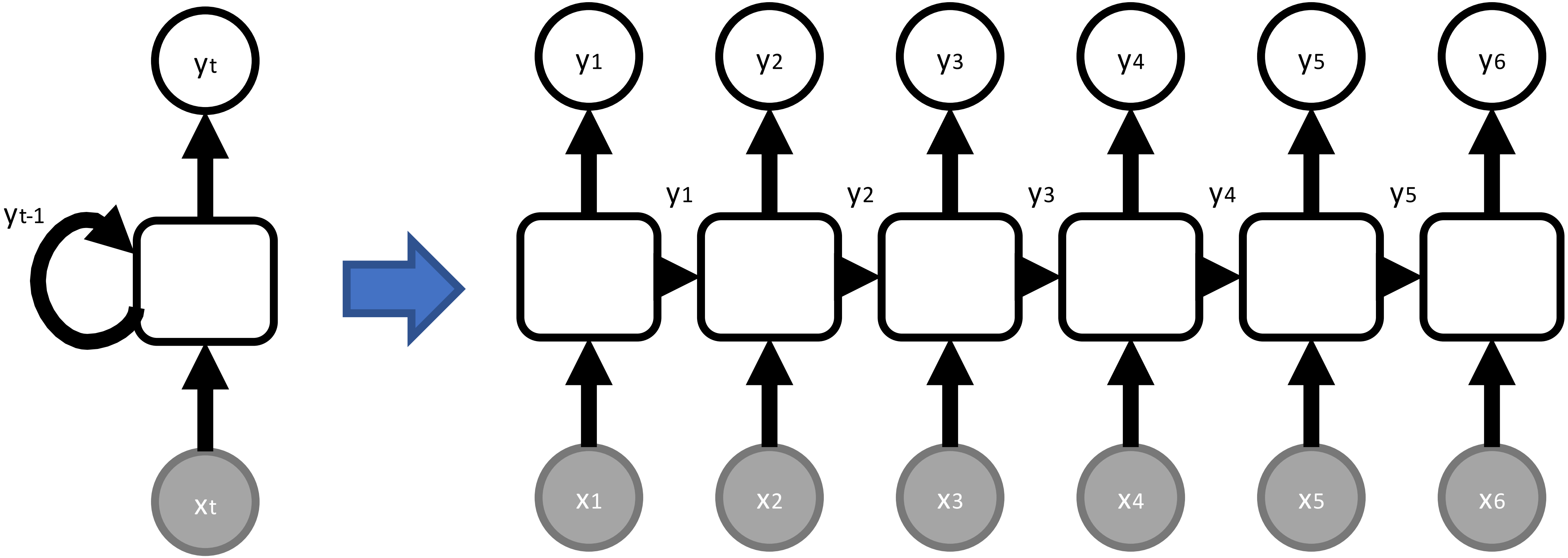
Connectionist temporal classification

- Formulation
  - Let character seq. be $C = (c_t \in \mathbb{U} | j = 1, \dots, J)$ and feature seq. be $O = (\mathbf{o}_t \in \mathbb{R}^D | t = 1, \dots, T)$
  - Focus on the posterior distribution $p(C|O)$, and rewrite it as

$$p(C|O) = \sum_Z p(C|Z,O)p(Z|O)$$
$$\approx \sum_Z \underbrace{p(C|Z)}_{\text{CTC LM}} \underbrace{p(Z|O)}_{\text{CTC AM}}.$$

  - No Bayes theorem, but use conditional independence
  - Introduce latent variable seq. $Z = (z_t \in \{\mathbb{U}, <\mathrm{b}>\} | t = 1, \dots, T)$ that has the same length as input feature seq.

Recurrent neural network



- Input and output are same length in general

Introduction of blank symbol <b>

- First we insert <b> to the character seq.

”see”

→ $C = ($“s”, “e”, “e”$)$, where $|C| = J$

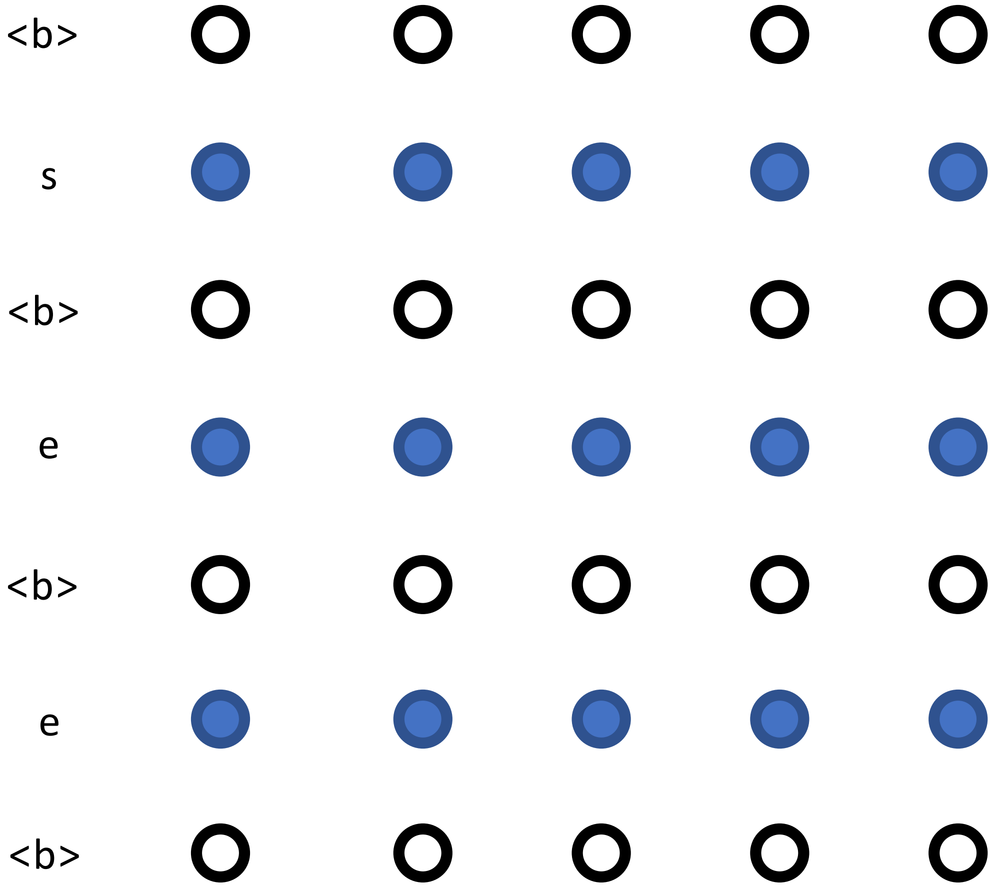
→ $C' = ($“<b>”, “s”, “<b>”, “e”, “<b>”, “e”, “<b>”$)$, where $|C'| = 2J + 1$

- Then, expand C' to the frame length T to form Z
  - All characters can be repeated
  - <b> can be skipped except when it is inserted between repeated character
    - “s”, ”<b>”, ”e”: we can skip <b>
    - “e”, ”<b>”, ”e”: we **cannot** skip <b>

Example of Z

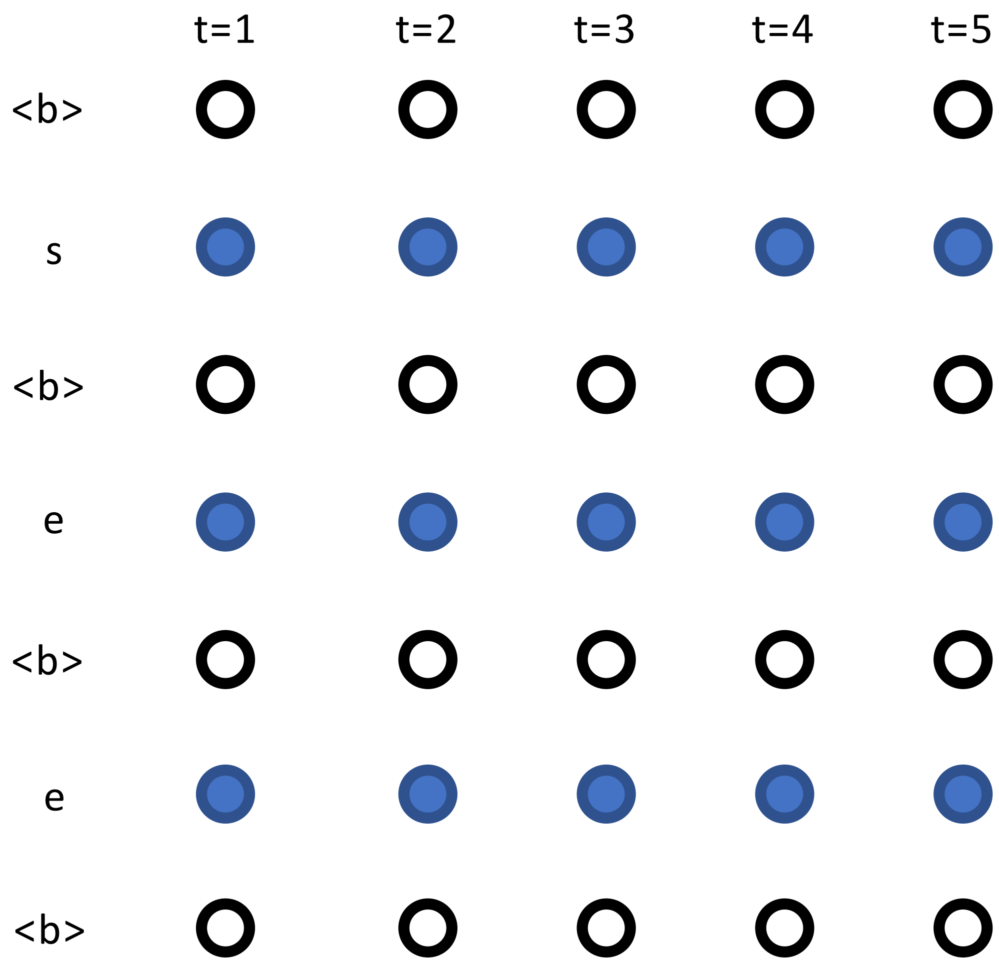
- $C =$ (“s”, “e”, “e”)
- $C' =$ (“<b>”, “s”, “<b>”, “e”, “<b>”, “e”, “<b>”)
- $T = 5$
- $Z =$ (“<b>”, “s”, “e”, “<b>”, “e”), (“s”, “<b>”, “e”, “<b>”, “e”), (“s”, “s”, “e”, “<b>”, “e”),….

Example of Z

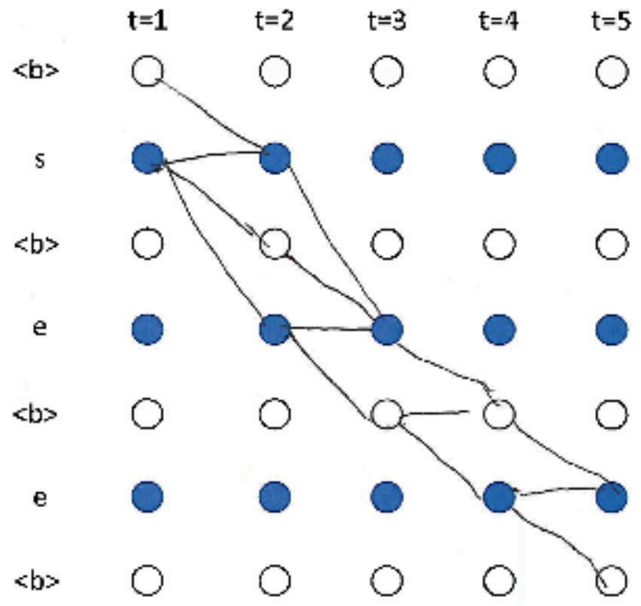


$$p(z_t|z_{t-1}, C) \propto \begin{cases} 1 & z_t = c'_l \text{ and } z_{t-1} = c'_l \text{ for all possible } l \\ 1 & z_t = c'_l \text{ and } z_{t-1} = c'_{l-1} \text{ for all possible } l \\ 1 & z_t = c'_l \text{ and } z_{t-1} = c'_{l-2} \text{ for all possible even } l \\ 0 & \text{otherwise} \end{cases}$$

Example of Z



Example of Z



10

CTC Formulation

- CTC acoustic model

$$p(Z|O) = \prod_{t=1}^{T} p(z_t|z_1, \cdots, z_{t-1}, O)$$
$$\approx \prod_{t=1}^{T} p(z_t|O).$$

  - Using conditional independence assumption to factorize the posterior $p(Z|O)$ but this is not bad assumption compared with HMM
  - This can be realized by Bidirectional LSTM

$$p(z_t = j|O) = [\mathrm{softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b})]_j \,,$$
$$\mathbf{h}_t = \mathrm{BLSTM}(O) \text{ for } t = 1, \ldots, T.$$

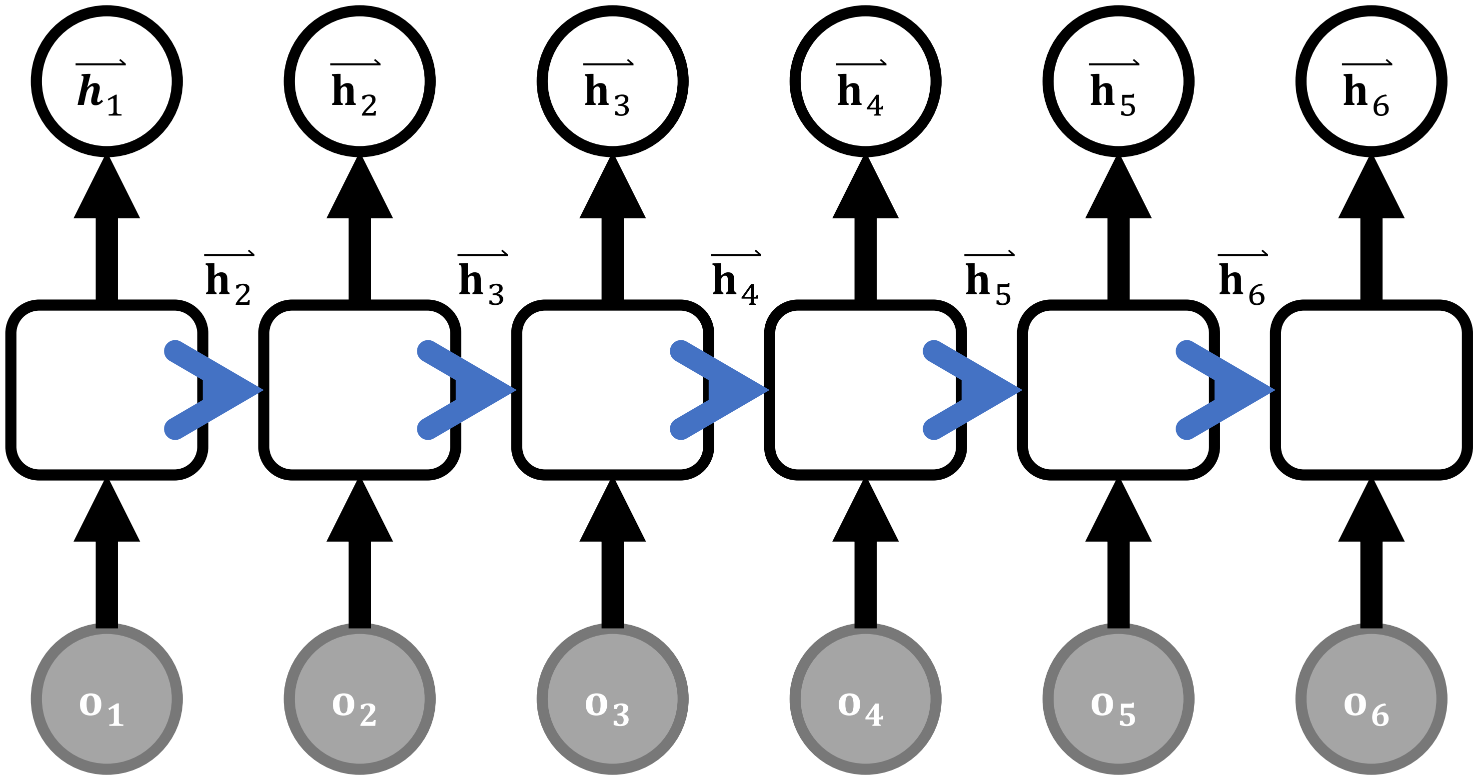
Forward directional RNN

- $\overrightarrow{\mathbf{h}_t} = f(\mathbf{o}_1, \ldots, \mathbf{o}_t)$

Then,

- $p(z_t | \mathbf{o}_1, \ldots, \mathbf{o}_t)$

  $\approx p(z_t | \overrightarrow{\mathbf{h}_t})$



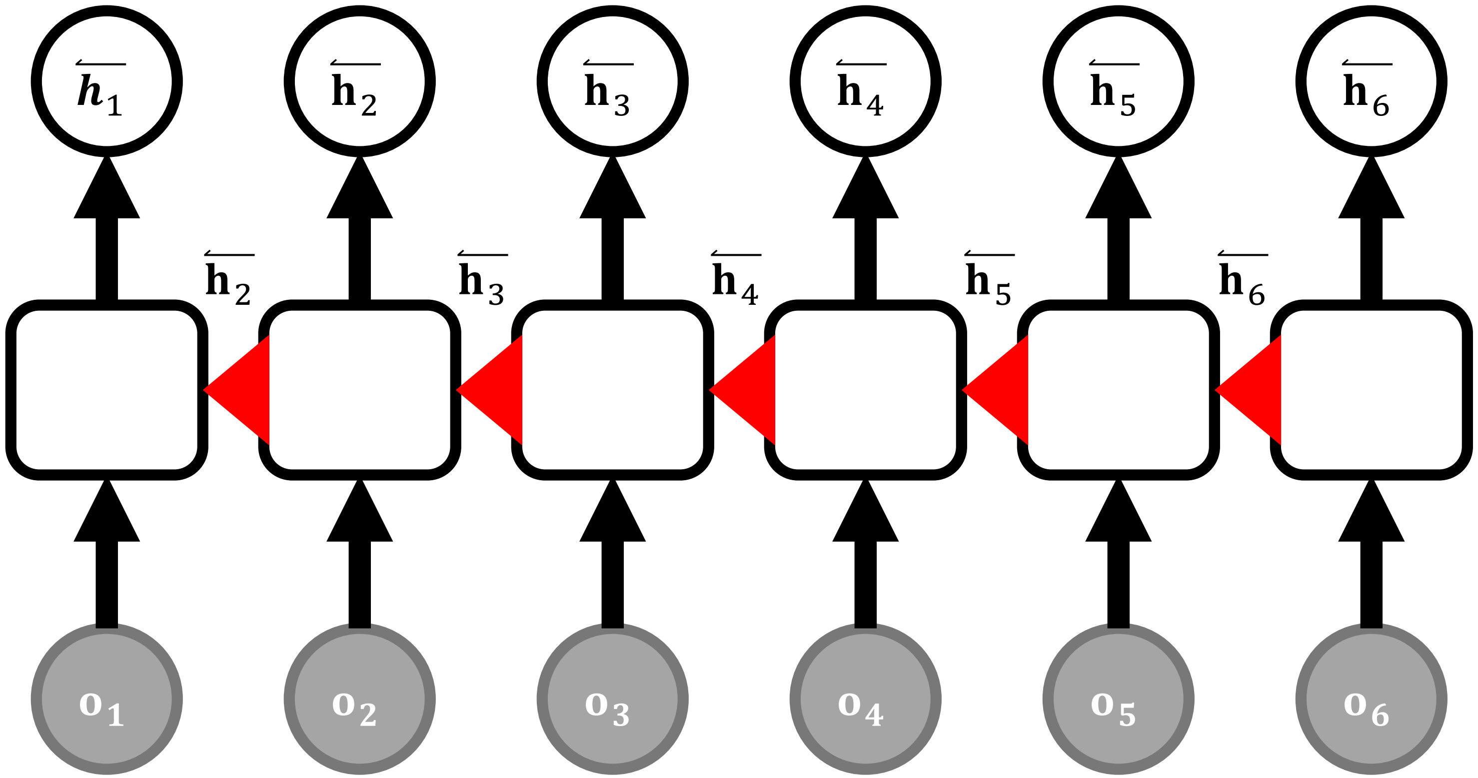
Backward directional RNN

- $\overleftarrow{\mathbf{h}_t} = f(\mathbf{o}_t, \dots, \mathbf{o}_T)$

Then,

- $p(z_t | \mathbf{o}_t, \dots, \mathbf{o}_T)$

  $\approx p(z_t | \overleftarrow{\mathbf{h}_t})$

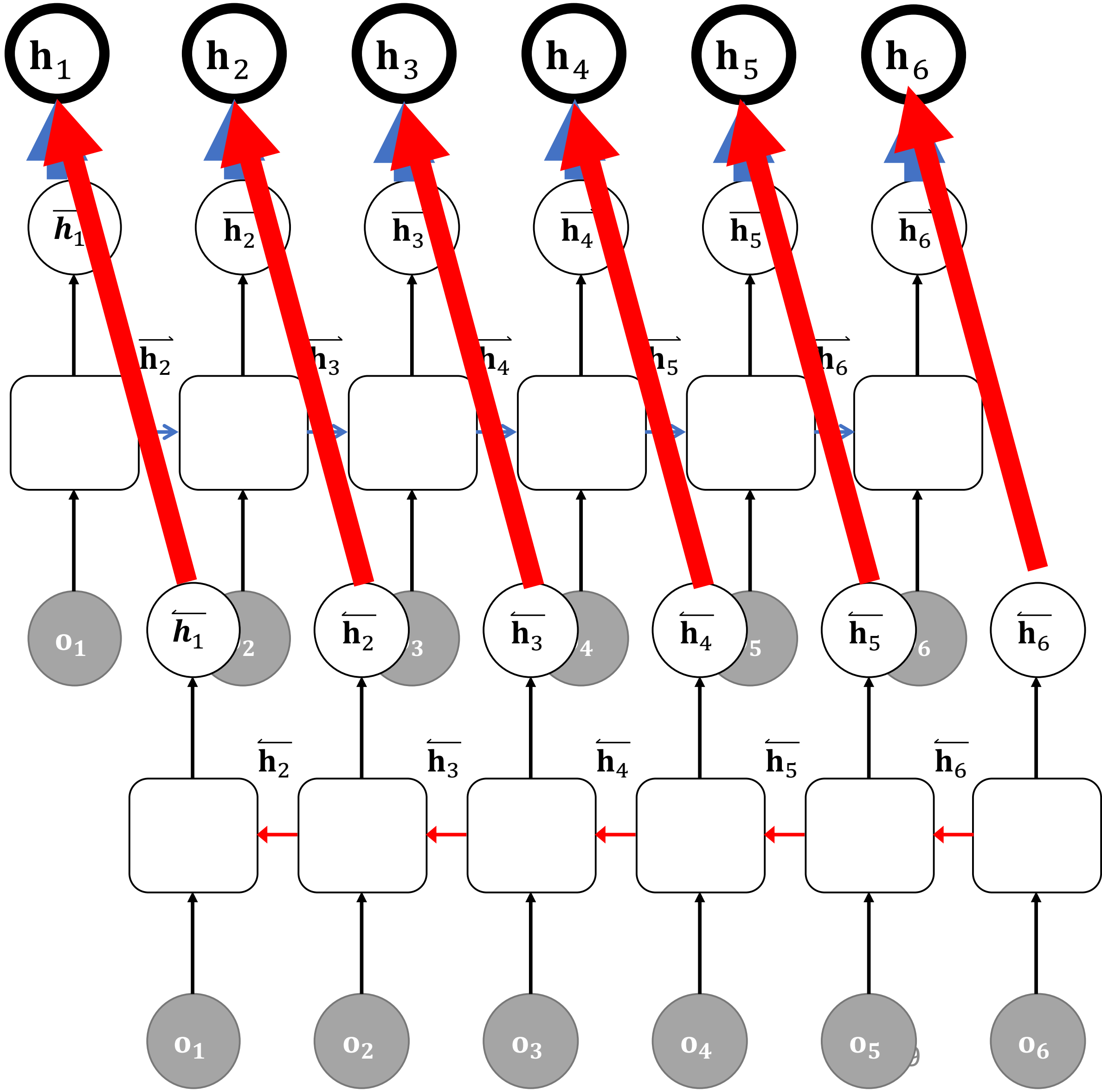


Bidirectional RNN

- $\mathbf{h}_t = \begin{bmatrix} \overrightarrow{\mathbf{h}_t} \\ \overleftarrow{\mathbf{h}_t} \end{bmatrix} = f(O = (\mathbf{o}_1, \ldots, \mathbf{o}_T))$

Then,

- $p(z_t|O)$
  $\approx p(z_t|\mathbf{h_t})$



CTC Formulation

- CTC Language model

$$p(C|Z) = \frac{p(Z|C)p(C)}{p(Z)}$$

$$= \prod_{t=1}^{T} p(z_t|z_1, \cdots, z_{t-1}, C)\frac{p(C)}{p(Z)}$$

$$\approx \prod_{t=1}^{T} p(z_t|z_{t-1}, C)\frac{p(C)}{p(Z)},$$

  - Using conditional independence assumption (1st order Markov) to factorize the posterior, same as the HMM
  - $p(C)$: Letter language model (we can also combine the word language model)
  - $p(Z)$: Prior probability for the state sequence

Summary of CTC formulation

- $p(C|O)$ is rewritten as follows

$$p(C|O) \approx \sum_{Z} \prod_{t=1}^{T} p(z_t|z_{t-1}, C) p(z_t|O) \frac{p(C)}{p(Z)}.$$

- In general, prior probabilities $p(C)$ and $p(Z)$ are separately obtained (not fully end-to-end)
- We can further eliminate the prior probabilities by assuming the uniform distributions as follows ($\mathcal{Z}(C)$ denotes all possible CTC paths given C):

$$p(C|O) \approx \underbrace{\sum_{Z \in \mathcal{Z}(C)} \prod_{t=1}^{T} p(z_t|O)}_{\triangleq p_{\mathrm{ctc}}(C|O)}$$

  - Basically, we can use **a forward-backward algorithm** to estimate the parameter

Baidu CTC
[Amodei+(2015)]

- Optimization of computational cost of CTC dynamic programming
- Multiple GPUs
- Architecture optimization (BLSTM -> GRU, use of CNN)
- Use 12,000 hours of data for training
- Data augmentation (noise)

Read Speech			
Test set	DS1	DS2	Human
WSJ eval'92	4.94	3.60	5.03
WSJ eval'93	6.94	4.98	8.08
LibriSpeech test-clean	7.89	5.33	5.83
LibriSpeech test-other	21.74	13.25	12.69

Google CTC
[Soltau+(2016)]

- Word-level CTC, conventional BLSTM
- No language model
- 125,000 hours of training data (!) from Youtube

Model	Layers	Outputs	Params	Vocab	OOV(%)	Spoken WER(%) w/ LM	Spoken WER(%) w/o LM
CTC CD phone	7x1000	6400	43m	500000	0.24	12.3	—
CTC spoken words	7x1000	82473	116m	82473	0.63	11.6	12.0

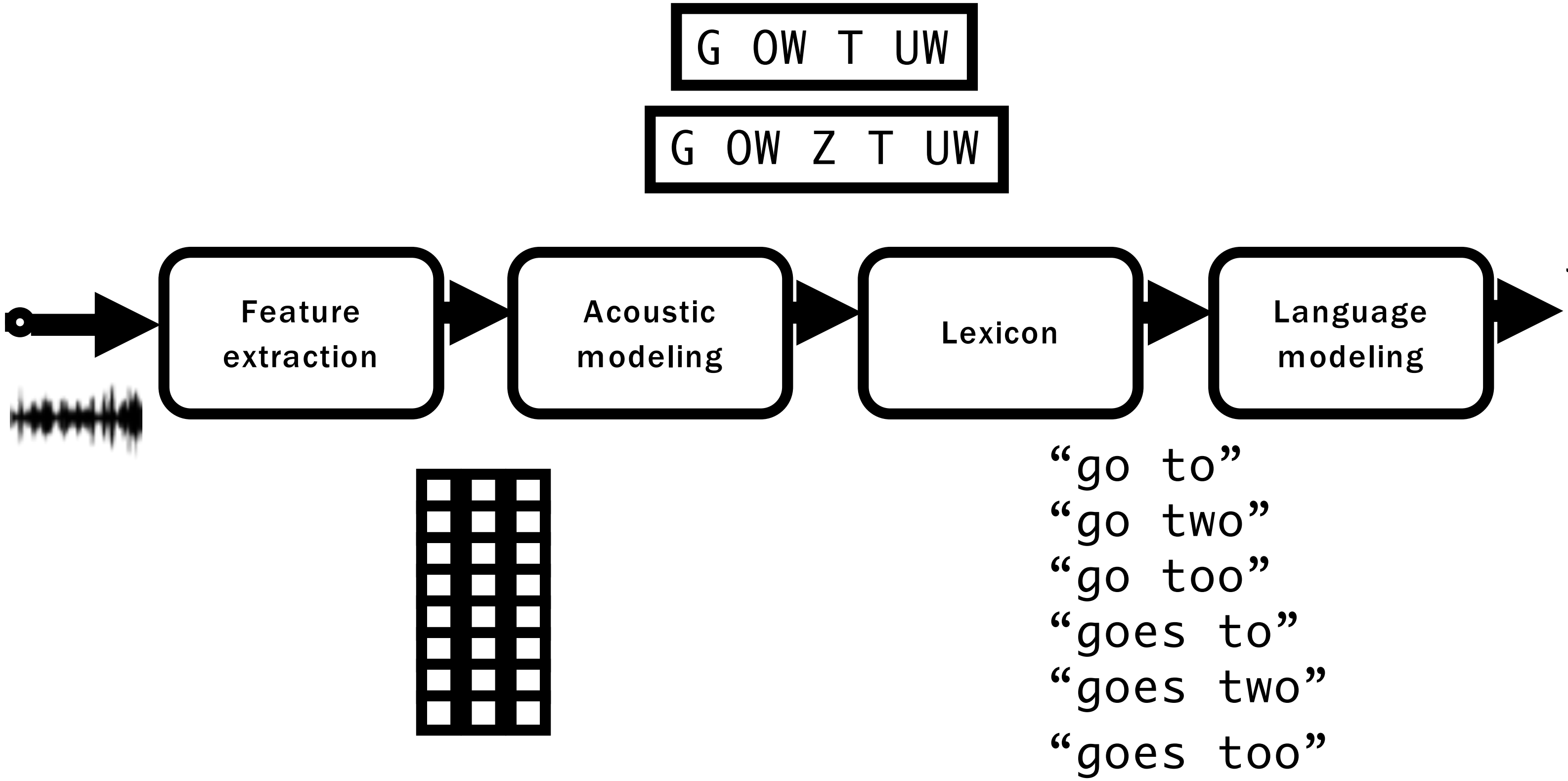
- Word-level CTC obtains comparable performance (even without LM)

Summary

- CTC
    - One promising direction of end-to-end
    - No language model
    - Still based on conditional independence assumptions and Markov assumptions
    - CTC is really end-to-end?
- Attention
    - Another end-to-end

Table of contents

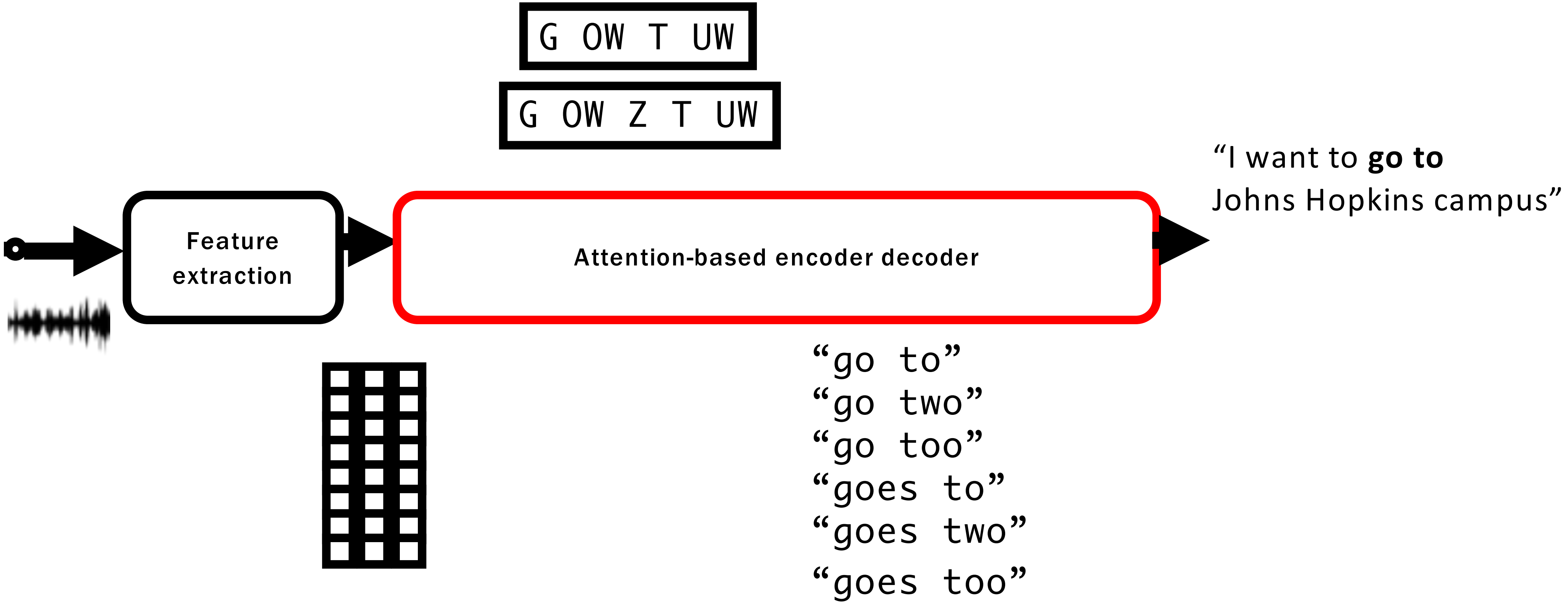
Speech recognition pipeline



"I want to **go to** Johns Hopkins campus"

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

Speech recognition pipeline



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

Attention based encoder-decoder

- Let $C = (c_j \in \mathbb{U} | j = 1, \ldots, J)$, be a character sequence
  - $\mathbb{U}$: set of characters
- Let $O = (\mathbf{o}_t \in \mathbb{R}^D | t = 1, \ldots, T)$, be a sequence of D dimensional feature vectors

$$\hat{C} = \mathrm{argmax}_C p(C|O)$$

- Problem: T and J are different, and we cannot use normal neural networks
- Sequence to sequence is a solution to deal with it

Sequence to sequence

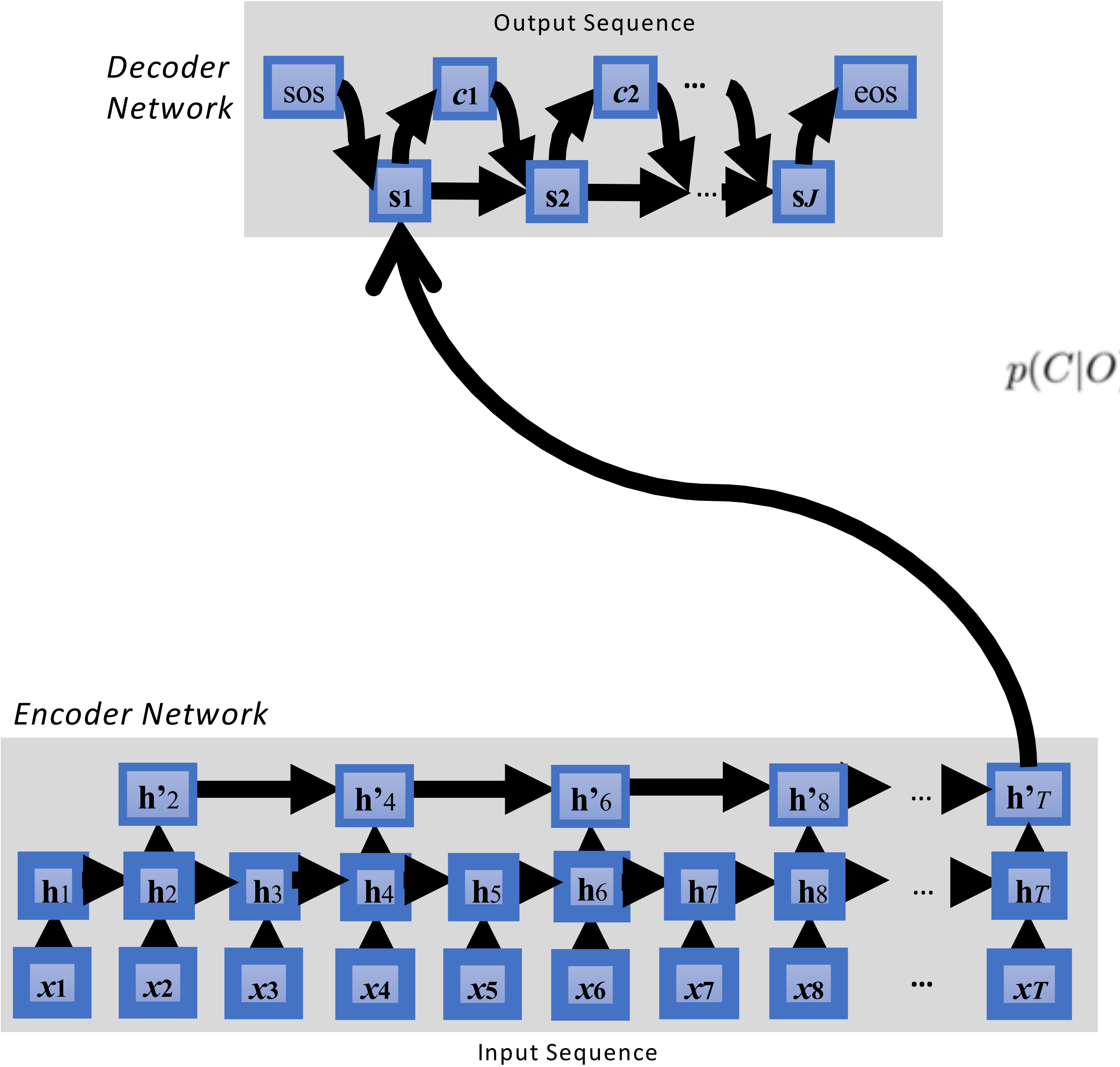
- We only use a probabilistic chain rule

$$p(C|O) = \prod_{j=1}^{J} p(c_j|C_{1:j-1}, O)$$

- Encoder-decoder architecture
  - Taking a final LSTM vector as an initial vector of a decoder network

$$\begin{aligned} p(C|O) &= \prod_{j=1}^{J} p(c_j|C_{1:j-1}, O) \\ &\approx \prod_{j=1}^{J} p(c_j|C_{1:j-1}, \mathbf{h}'_T = \mathrm{LSTM}(O)) \end{aligned}$$

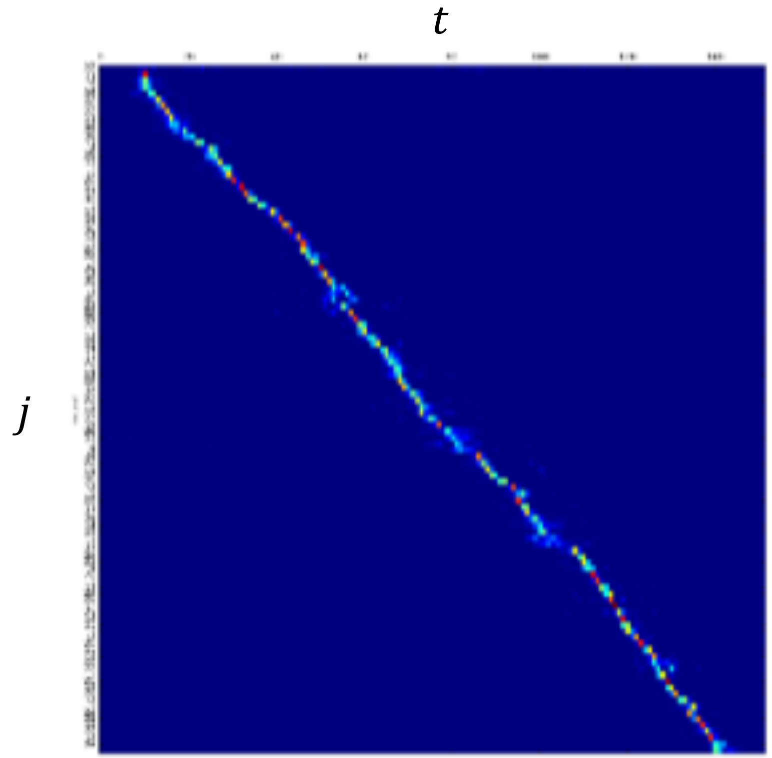
  - RNNLM-style text generation given summarized acoustic information $\mathbf{h}'_T$



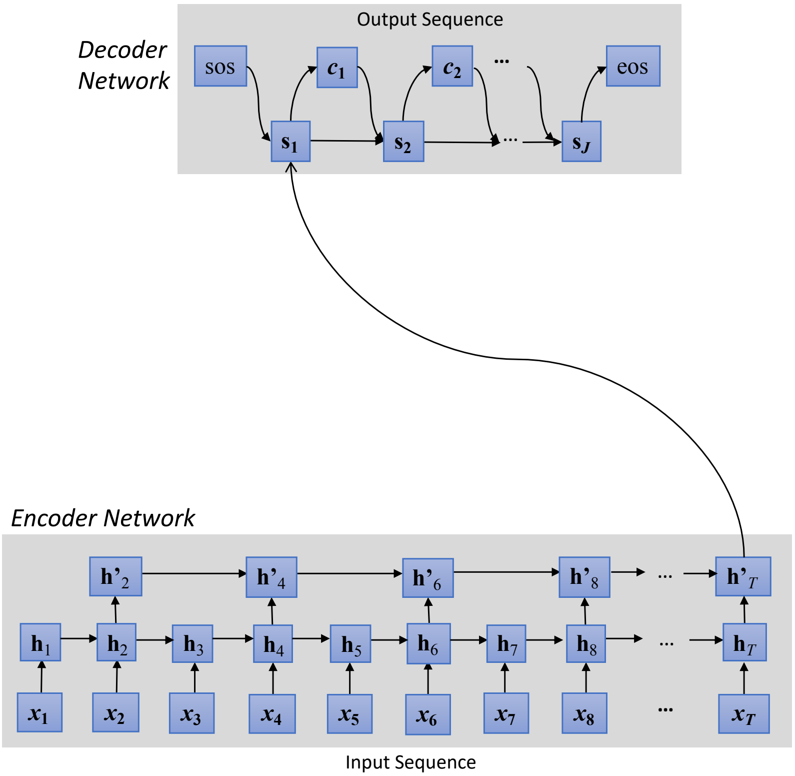
$$
\begin{aligned}
p(C|O) &= \prod_{j=1}^{J} p(c_j | C_{1:j-1}, O) \\
&\approx \prod_{j=1}^{J} p(c_j | C_{1:j-1}, \mathbf{h}'_T = \mathrm{LSTM}(O))
\end{aligned}
$$

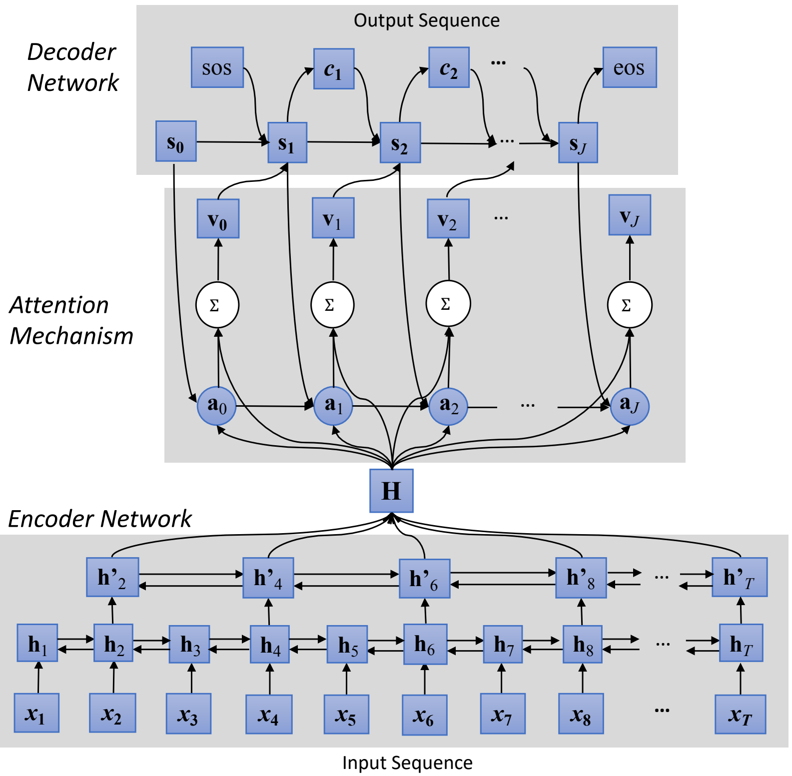
Problem of encoder-decoder architecture

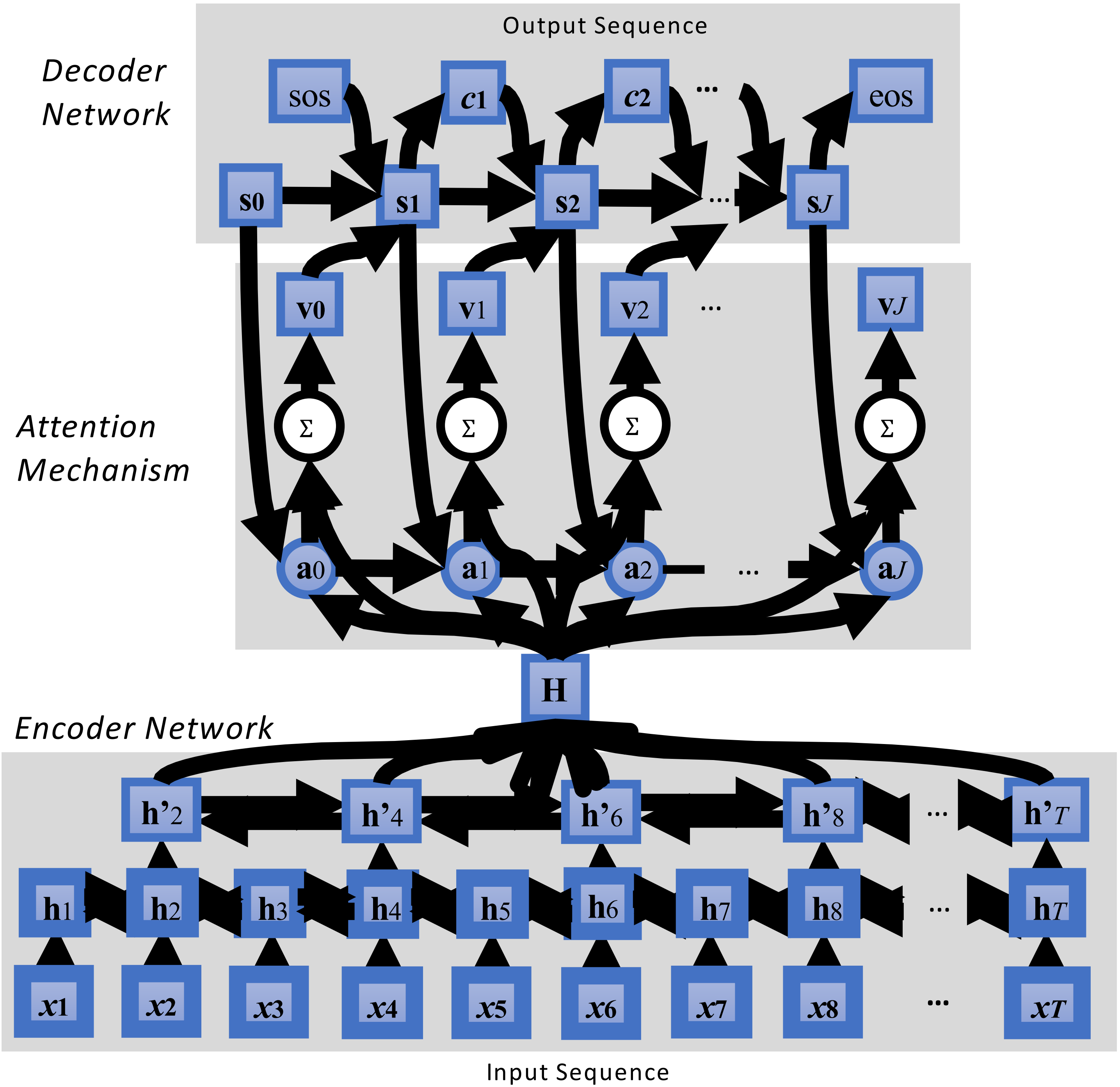
- We cannot explicitly have an alignment property
  - No connection between frame-level activations $\mathbf{h}'_t$ and output labels y_j
  - Long sentence would have issues
- Attention mechanism
  - Compute the assignment probability for each output j from a neural network
  - $\mathbf{a}_j = \{a_{jt} | t = 1, \dots, T\} \in \mathbb{R}^T, 0 < a_{jt} < 1, \sum_{t=1}^{T} a_{jt} = 1$
  - Obtain the context vector $\mathbf{v}_j = \sum_{t=1}^{T} a_{jt}\, \mathbf{h}'_t$, which is feeded to the RNNLM generator



From seq2seq to attention encoder-decoder

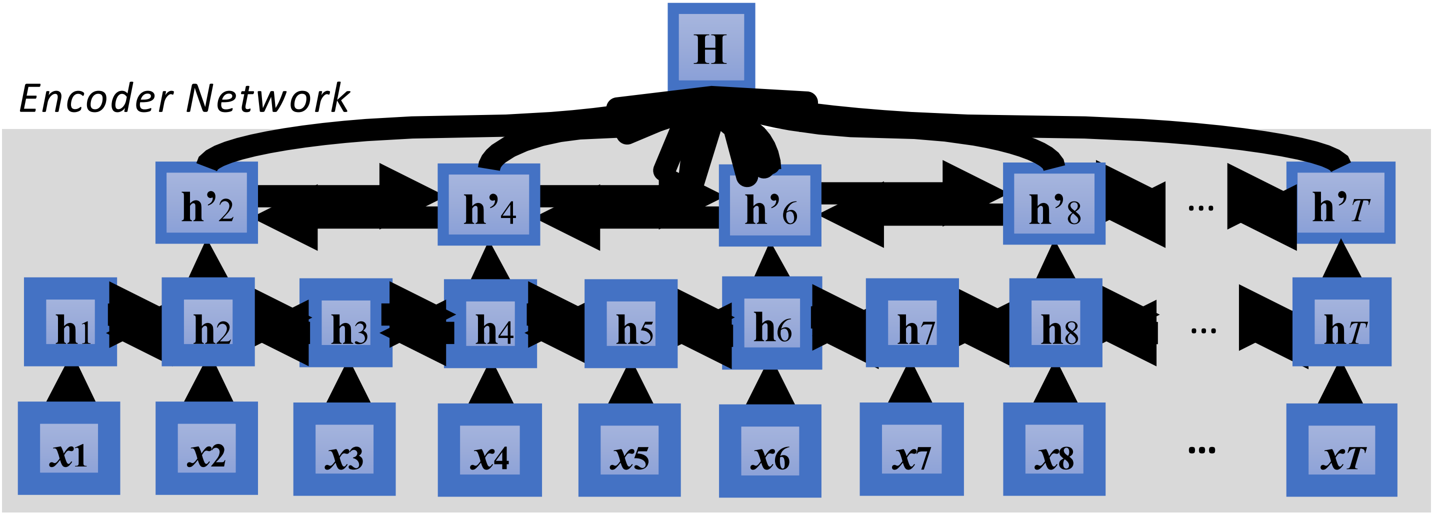






Output Sequence
Decoder Network
sos
c1
c2
…
eos
s0
s1
s2
…
sJ
Attention Mechanism
v0
v1
v2
…
vJ
Σ
Σ
Σ
Σ
a0
a1
a2
…
aJ
H
Encoder Network
h'2
h'4
h'6
h'8
…
h'T
h1
h2
h3
h4
h5
h6
h7
h8
…
hT
x1
x2
x3
x4
x5
x6
x7
x8
…
xT
Input Sequence

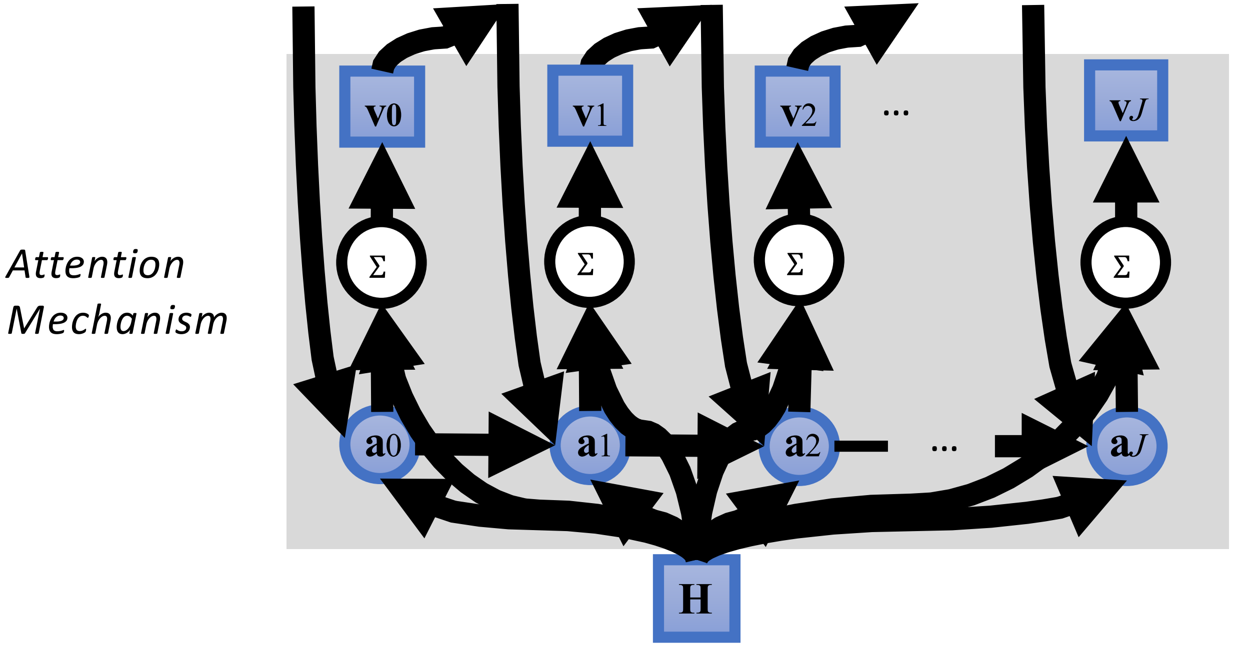
Encoder network



Input Sequence

- The encoder network to extract high-level features $\mathbf{H} = (\mathbf{h}'_t | t = 1, \dots, T)$ from BLSTM, i.e., $\mathbf{H} = \mathrm{BLSTM}(O)$
- Subsampling
  - Reduces computational cost
  - Input and output lengths similar

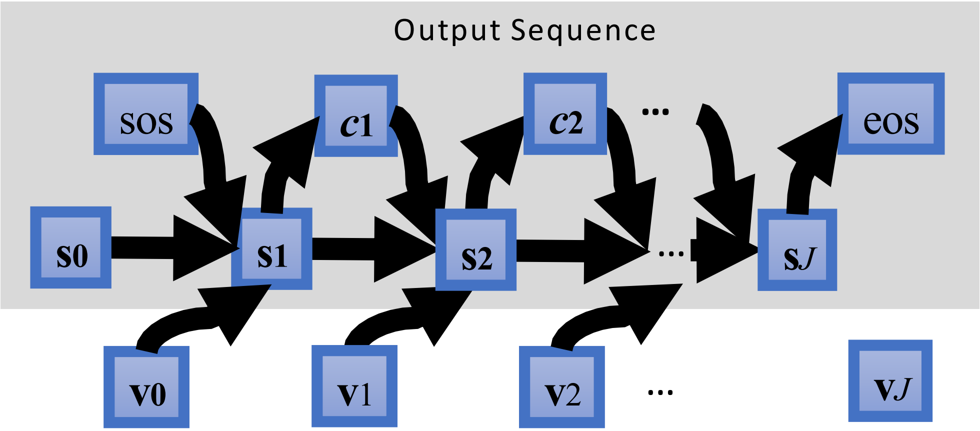
Attention mechanism



- Compute the attention weights $\mathbf{a}_j = \{a_{jt} | t = 1, \dots, T\}$
- Compute the context vector $\mathbf{v}_j = \sum_{t=1}^{T} a_{jt}\, \mathbf{h}'_t$

Decoder network

Decoder Network



- RNNLM generator given the context vector $\mathbf{v}_j$

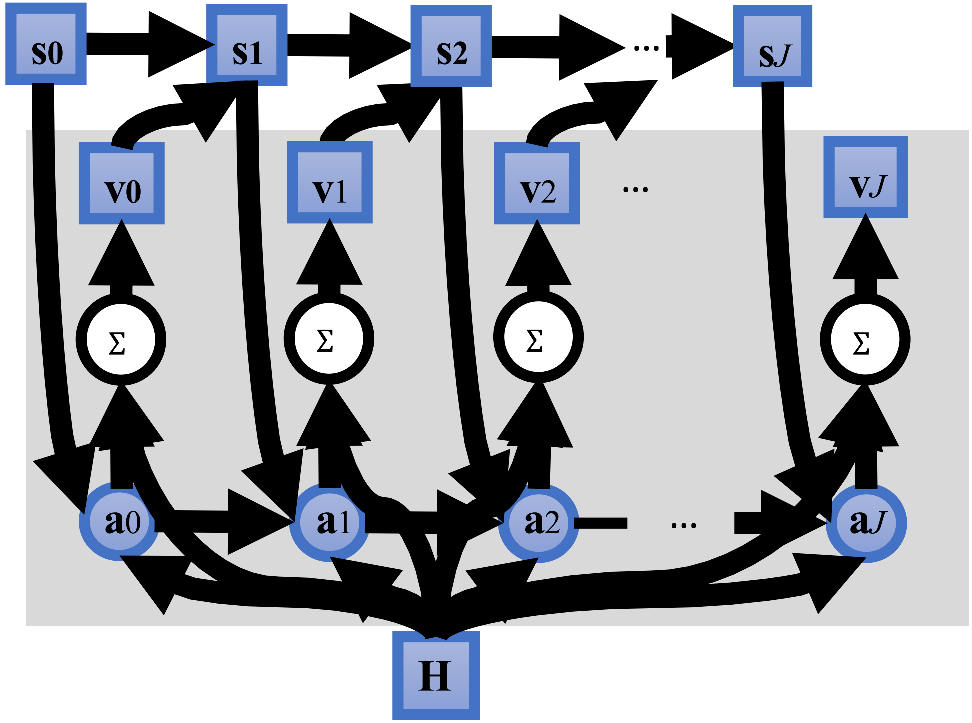
$$p(C|O) = \prod_{j=1}^{J} p(c_j|C_{1:j-1}, \mathbf{v}_j)$$

- Consider the acoustic information through the context vector

$$p(C|O) = \prod_{j=1}^{J} p(c_j|\mathbf{s}_j(C_{1:j-1}, \mathbf{v}_{j-1}))$$

How to compute the attention?

Attention Mechanism



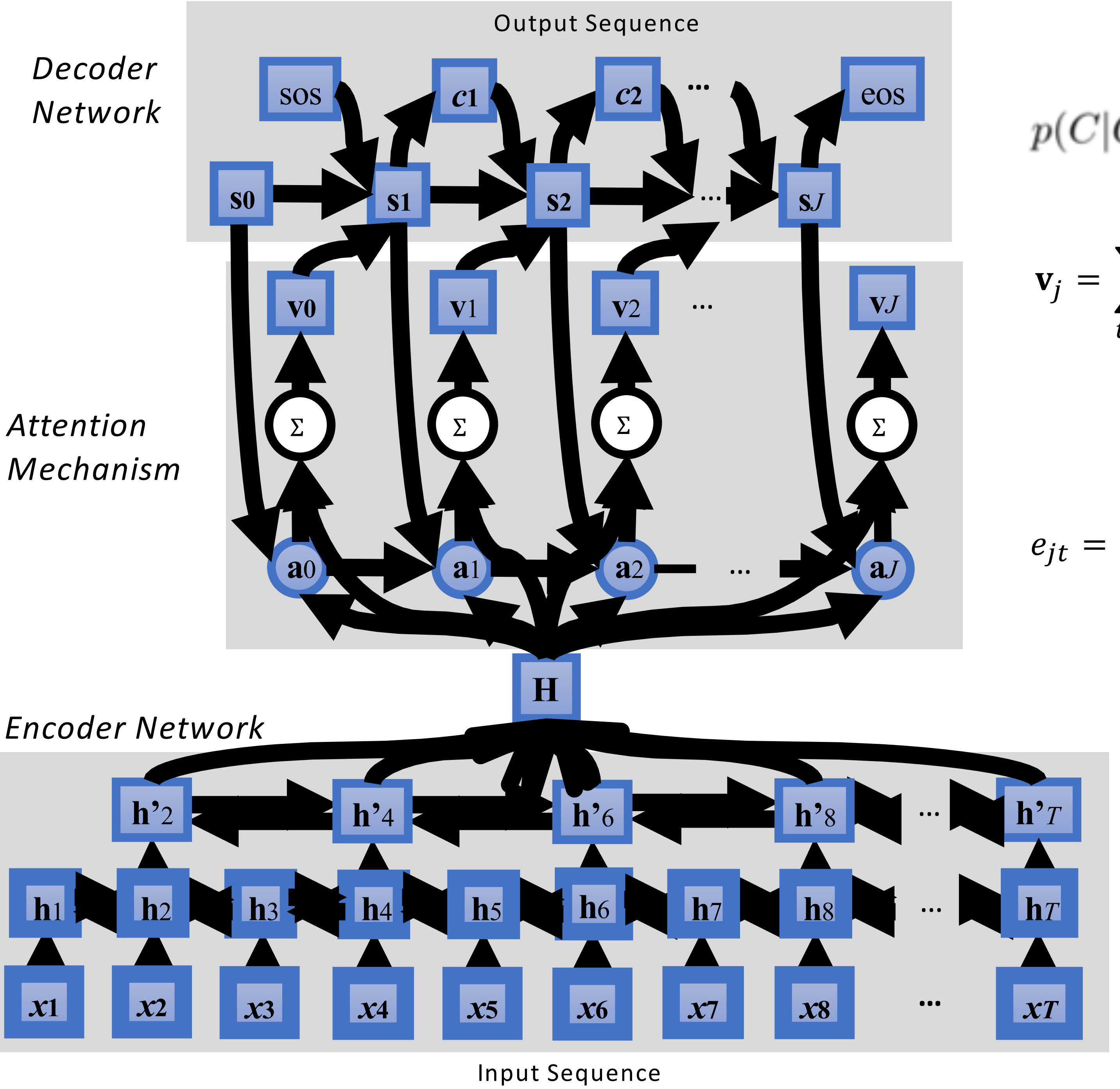
- Use the decode state $\mathbf{s}_j \in \mathbb{R}^{D_{out}}$ and encoder output $\mathbf{H} \in \mathbb{R}^{D_{in} \times T}$ to compute $\mathbf{a}_j \in \mathbb{R}^T$
- Dot product attention

$$e_{jt} = \sum_{d_{in} d_{out}} w_{d_{in} d_{out}} s_{j d_{out}} h'_{d_{in} t}$$
$$= \left[\mathbf{H}^T \mathbf{W} \mathbf{s}_j\right]_{jt}$$

- Energy based attention

$$e_{jt} = \mathbf{v}^T \tanh(\mathbf{W}^s \mathbf{s}_j + \mathbf{W}^h \mathbf{h}_t + \mathbf{b})$$
$$\mathbf{v} \in \mathbb{R}^{D_{att}}, \mathbf{W}^s \in \mathbb{R}^{D_{att} \times D_{out}}, \mathbf{W}^h \in \mathbb{R}^{D_{att} \times D_{in}}$$

- There are a lot of variations
- Softmax operation $\mathbf{a}_j = \text{softmax}(\mathbf{e}_j)$

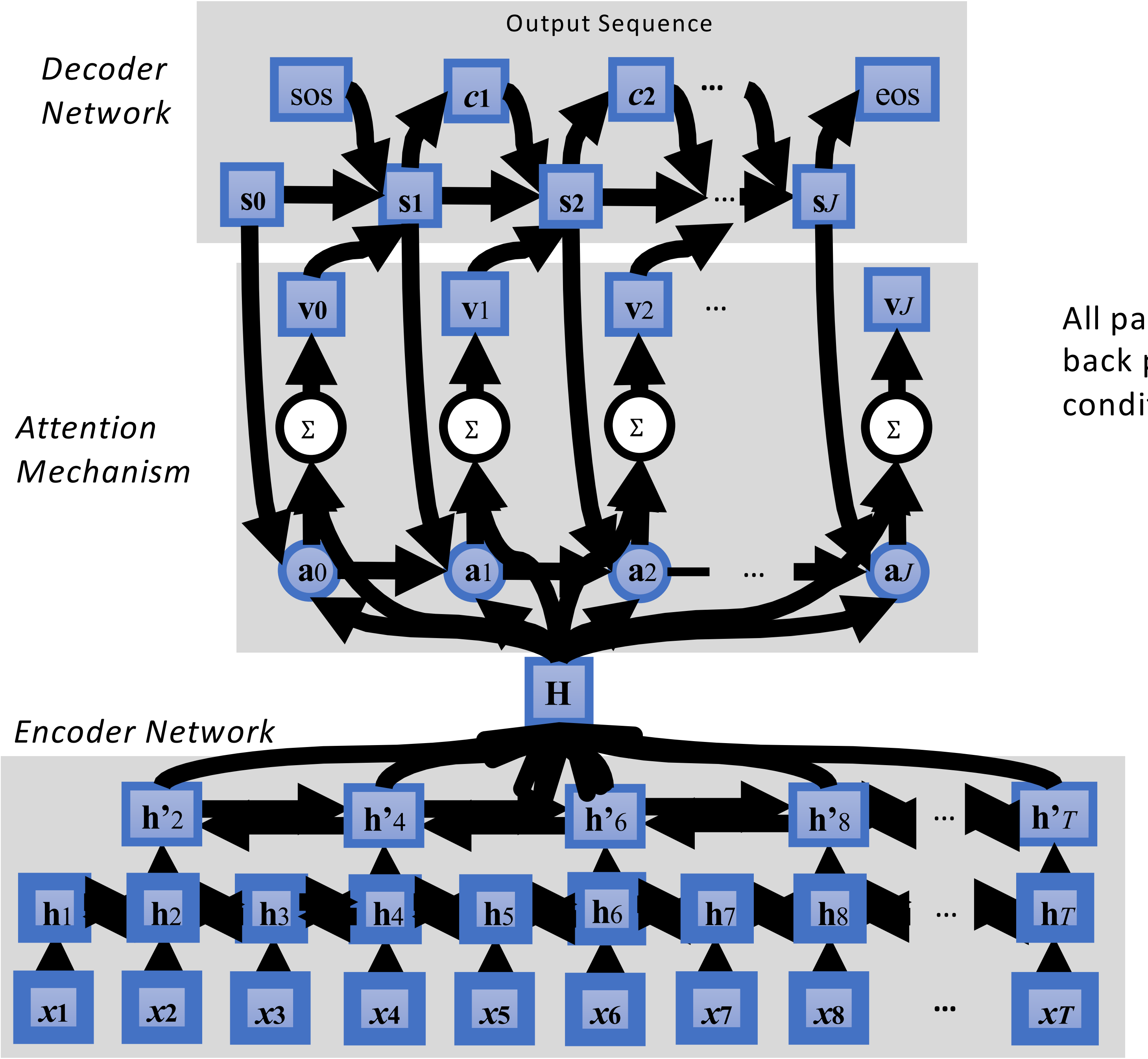


$$p(C|O) = \prod_{j=1}^{J} p(c_j | \mathbf{s}_j(C_{1:j-1}, \mathbf{v}_{j-1}))$$

$$\mathbf{v}_j = \sum_{t=1}^{T} a_{jt}\, \mathbf{h}'_t$$

$$e_{jt} = \mathbf{v}^T \tanh\left(\mathbf{W}^s \mathbf{s}_j + \mathbf{W}^h \mathbf{h}_t + \mathbf{b}\right)$$
$$\mathbf{a}_j = \mathrm{softmax}(\mathbf{e}_j)$$

$$\mathbf{H} = \mathrm{BLSTM}(O)$$

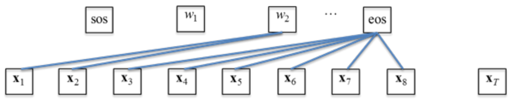


All parameters are jointly optimized by back propagation to maximize a conditional likelihood

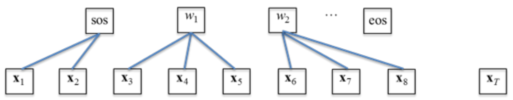
$$\hat{\Theta} = \operatorname{argmax}_{\Theta} p(C|O)$$

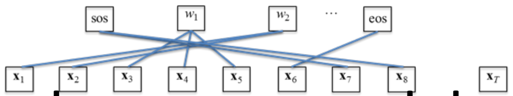
Summary of attention encoder-decoder

- No conditional independence assumption
  - No need for pronunciation lexicon
  - Attention & Encoder: acoustic model
  - Decoder: language model
  - Combine acoustic and language models with single network
- Attention model is too flexible for alignment issues

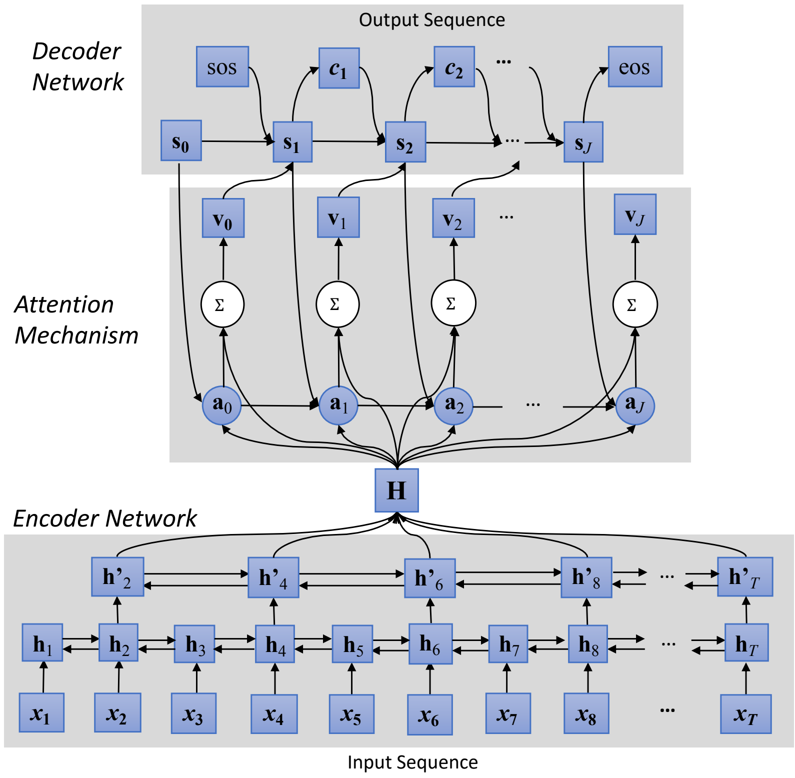


HMM or CTC case





- Not easy to combine the language model trained with a bunch of text data



Experiments (Google, *slides from T. Sainath)

Exp ID	Model	WER - VS	WERR
E1	Grapheme	9.2	-
E2	WPM	9.0	2.2%
E3	+MHA	8.0	11.1%
E4	+Optimization*	6.7	16%
E5	MWER	5.8	**13.4%**

- WPM: Word piece model, MHA: Multihead attention, MWER, minimum word error rate
- Hybrid DNN/HMM system 6.7%

Experiments (MERL)

- Hybrid CTC/attention
  - Combine CTC and attention encoder-decoder networks
- Corpus of spontaneous Japanese (CSJ)

	Eval1	Eval2	Eval3
End-to-end	**7.9**	**5.8**	**6.7**
Hybrid DNN/HMM	8.4	6.9	7.1

- HKUST Chinese Telephone Conversation

	Test
End-to-end	**28.0**
Hybrid DNN/HMM	28.2

Summary

- Attention encoder-decoder
  - Another possible direction for end-to-end ASR
  - Single neural network to have acoustic, linguistic, and language modeling
  - Several reports that achieve better performance from conventional hybrid DNN/HMM
- Connection to NLP
- No need for pronunciation lexicon
  - Easily applied to multilingual ASR
- Opensource

  ESPnet https://github.com/espnet/espnet

  Today's lab