# ADVANCING CONNECTIONIST TEMPORAL CLASSIFICATION WITH ATTENTION MODELING

Amit Das, Jinyu Li, Rui Zhao, Yifan Gong

Microsoft AI and Research

# Outline

- Overview: Connectionist Temporal Classification (CTC)
- Issues with CTC
- Proposed Solution: Blend Attention directly into CTC
- Experiments and Results
- Conclusions

# Connectionist Temporal Classification (CTC)

- CTC is a sequence-to-sequence learning method used to map speech waveforms directly to characters, phonemes, or even words

$$L_{\text{CTC}} = -\ln p(\mathbf{z}|\,\mathbf{x})$$

$\mathbf{z}$ -- labels **sequence** $\mathbf{x}$ -- observation **frames**

- CTC paths $\boldsymbol{\pi}$ differ from labels sequences in that:

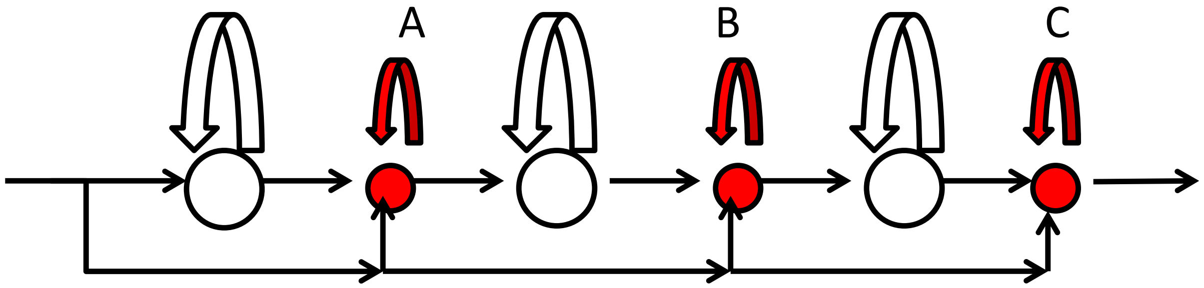
$$p(\mathbf{z}|\mathbf{x}) = \sum_{\boldsymbol{\pi} \in \text{Desired}} p(\boldsymbol{\pi}|\mathbf{x})$$

— Add the blank as an additional label, meaning no (actual) labels are emitted

— Allow repetitions of non-blank/blank labels

A A ε ε B C ε
ε A A B ε C C
ε ε ε A B C ε

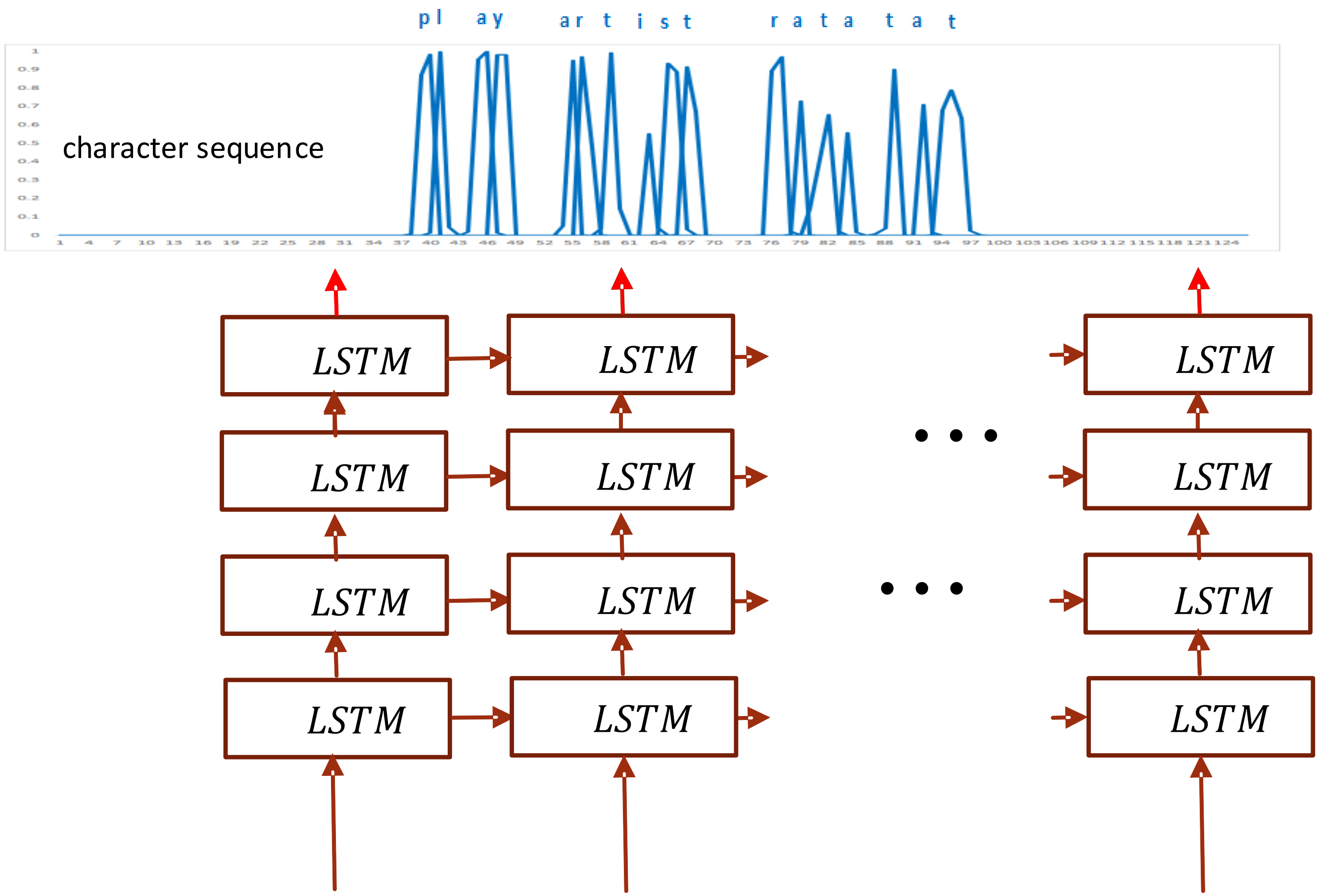
$\xrightarrow{\text{collapse}}$ A B C
$\xleftarrow[\text{expand}]{}$



# End-to-End Modeling with CTC

- Greedy decoding: concatenate the non-blank tokens corresponding to the posterior spikes.

- Neither LM nor complex decoding is involved.



# CTC Issues

**Issues:**

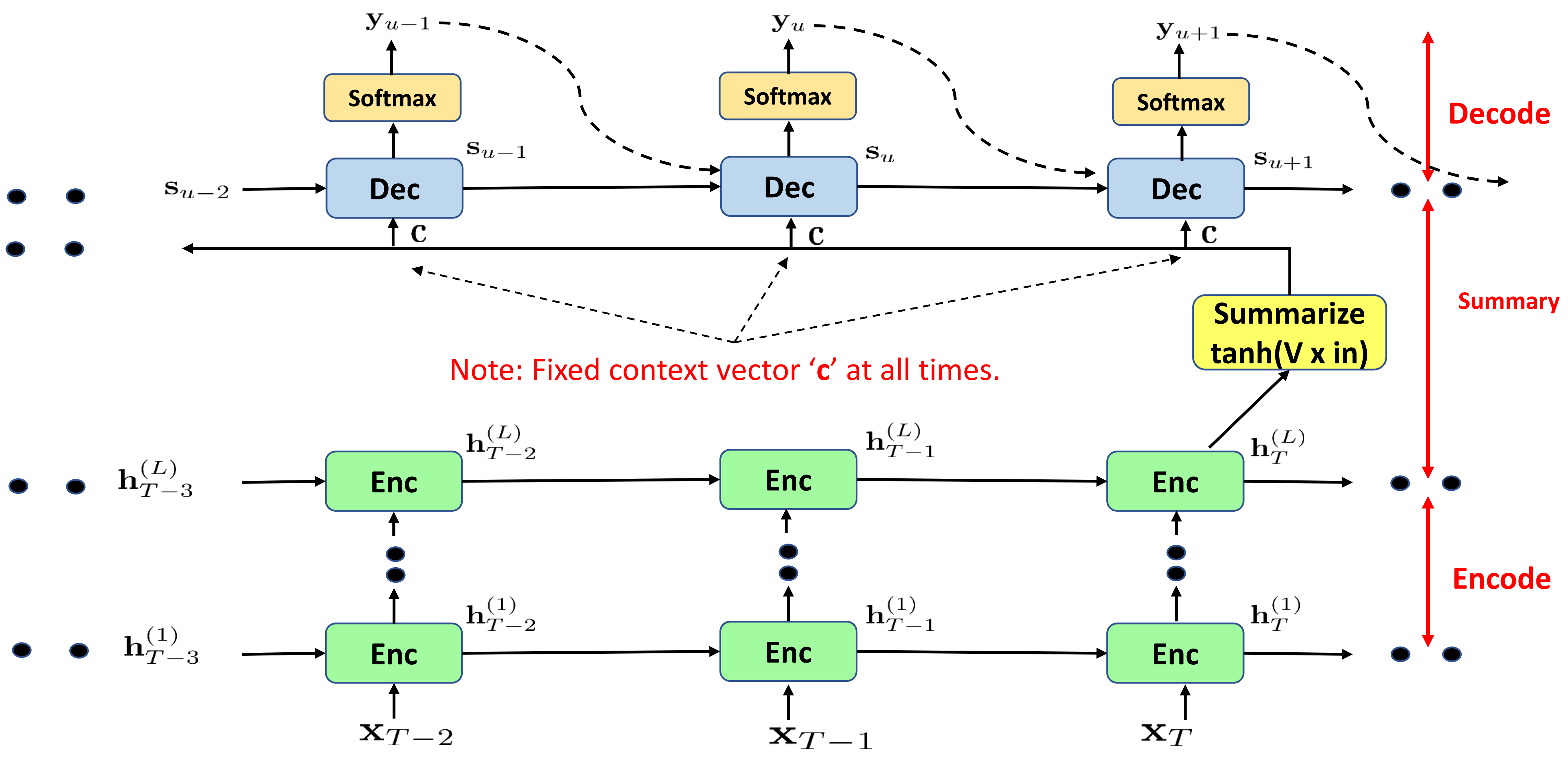
- Assumes conditional independence (CI) between outputs given input. Not true, in general, for sequential tasks like ASR, machine translation, language modeling.

$$p(\boldsymbol{\pi}|\mathbf{x}) \stackrel{\text{CI}}{=} \prod_{t=1}^{T} p(\pi_t|\mathbf{x}) \triangleq \prod_{t=1}^{T} y_t(\pi_t) \qquad \text{CI: } (\pi_t \perp \pi_{\neq t})|\mathbf{x}$$

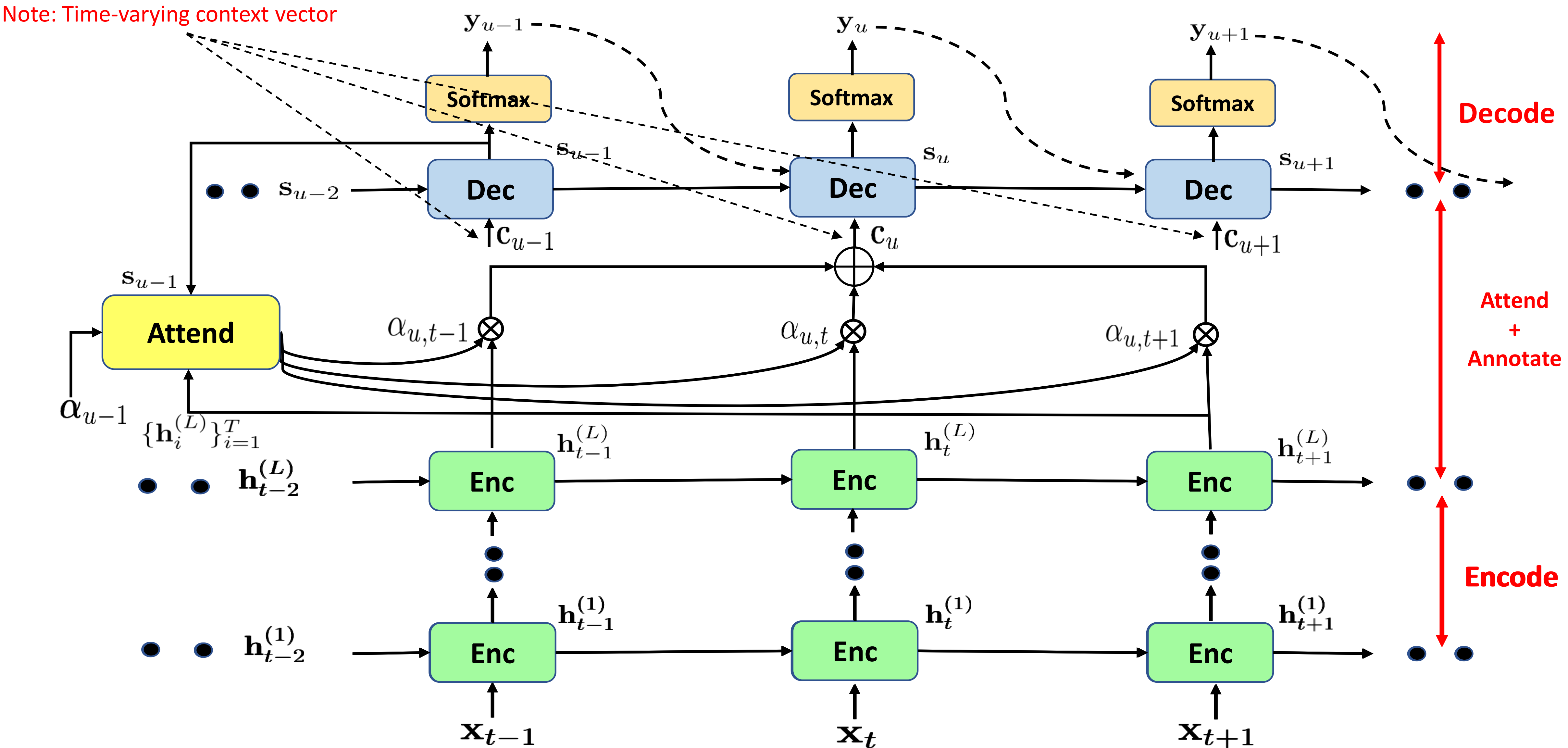
- Assumes hard alignment. Output $y_t$ dependent on input $x_t$ . Not true, in general, since neighboring inputs $x_{<t}, x_{>t}$ also have an influence.

**Solution:** Attention mechanism relaxes hard alignment.

# RNN-Encoder Decoder (No Attention)



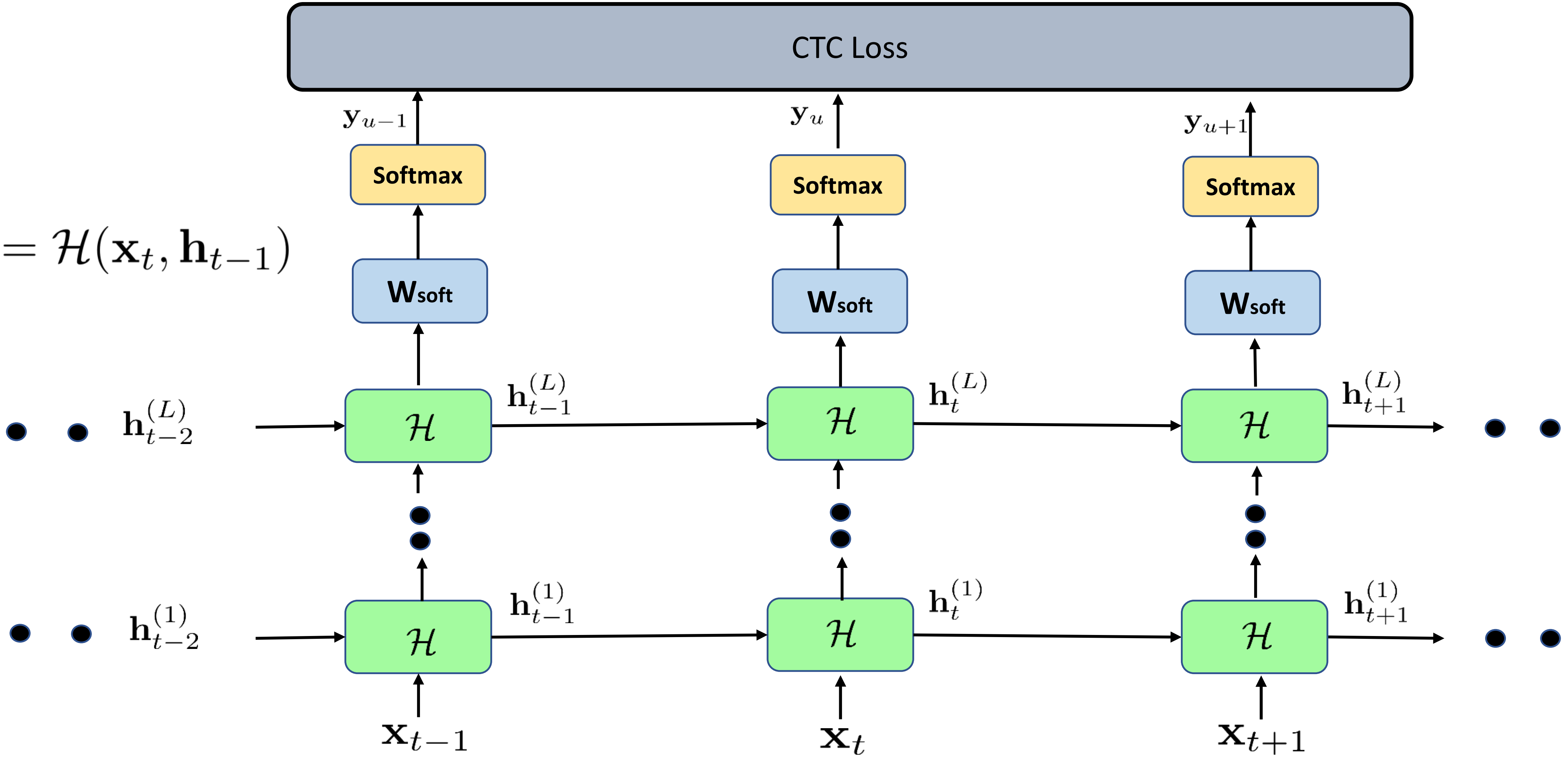
# RNN-Encoder Decoder (Attention)



# CTC Attention

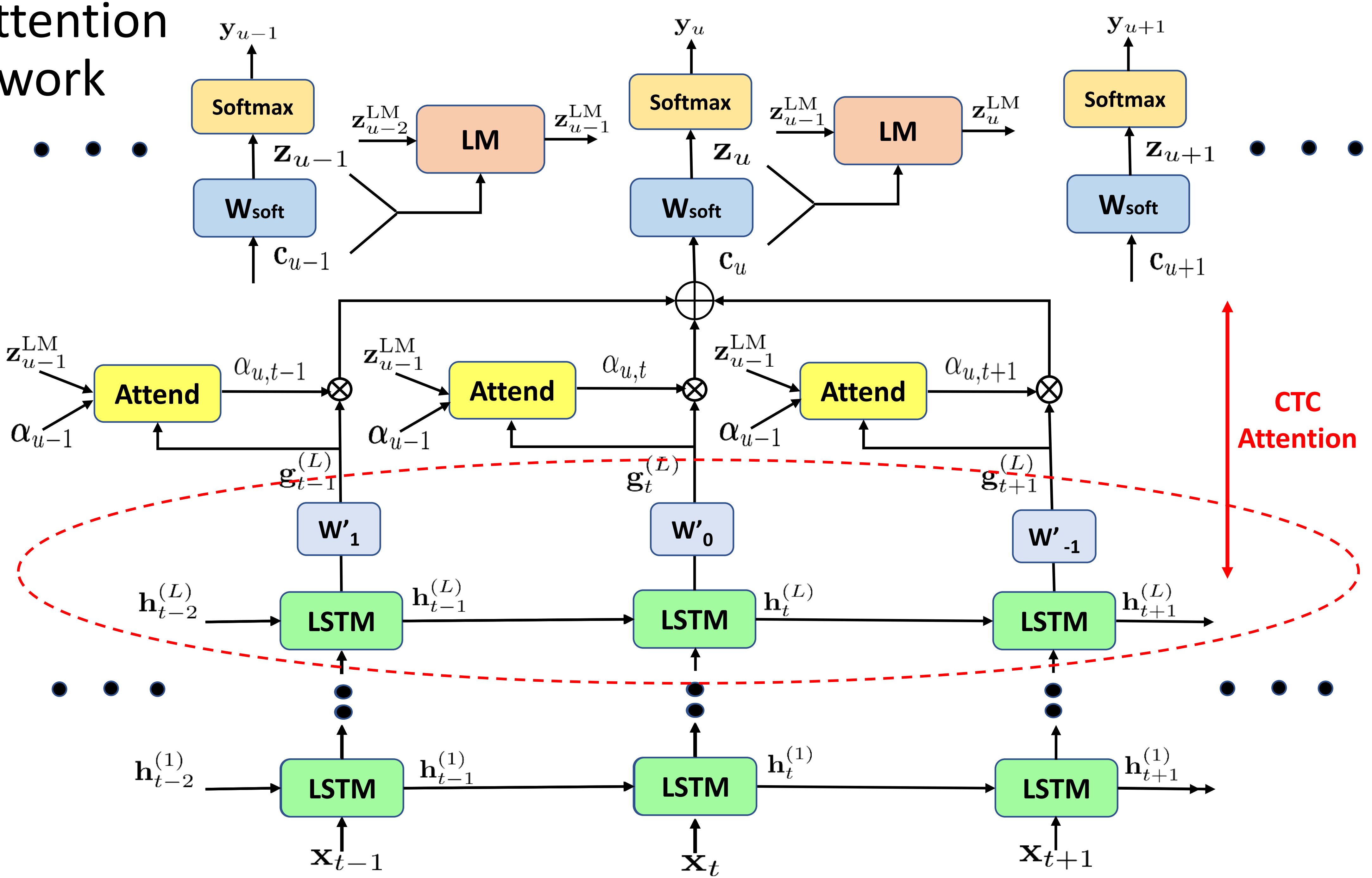
# Baseline CTC Network = RNN + CTC Loss

$$\mathbf{h}_t = \mathcal{H}(\mathbf{x}_t, \mathbf{h}_{t-1})$$



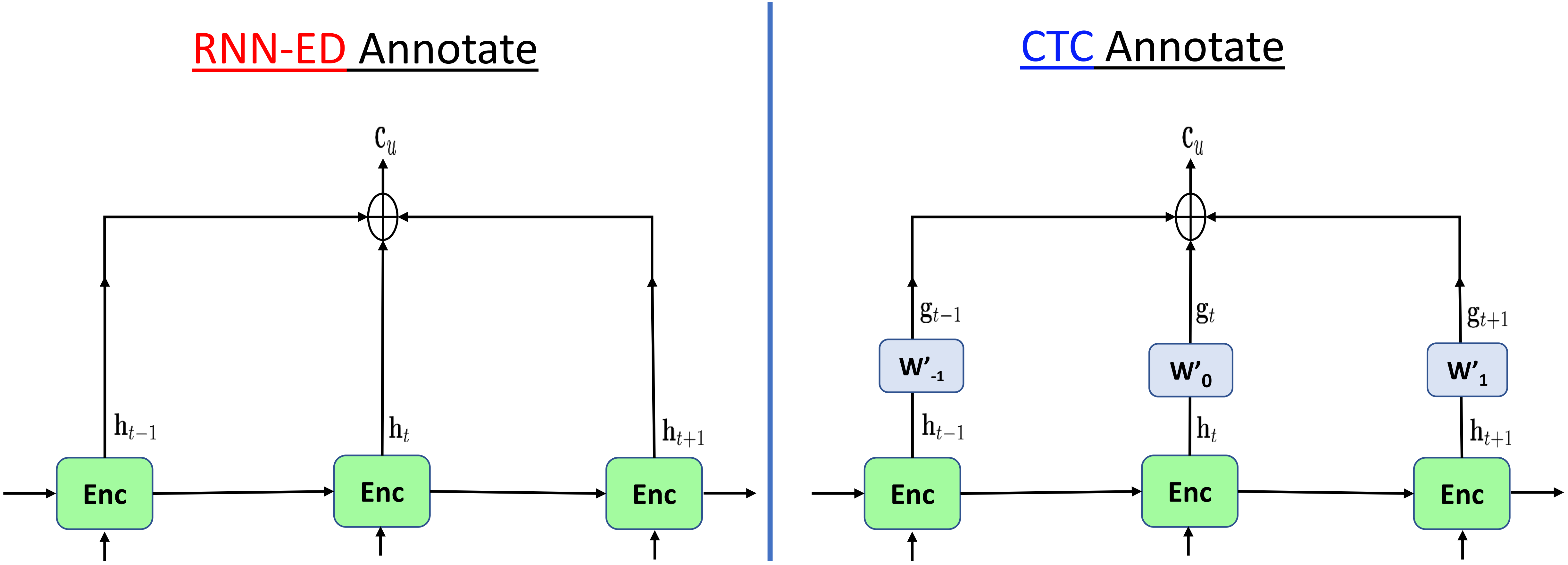

$u = t$ for CTC modeling

# CTC Attention Network



# CTC Annotate

- Key: Compute the context vector $\mathbf{c}_u$ as **time convolved feature**.



# Context Vector As Time Convolved Feature

- Time convolved feature is a special case of context vector with uniform attention.

## RNN-ED Annotate

$$\mathbf{c}_u = \text{Annotate}(\boldsymbol{\alpha}_u, \mathbf{h}) = \sum_{t=1}^{T} \alpha_{u,t} \mathbf{h}_t$$

Non-Uniform attention

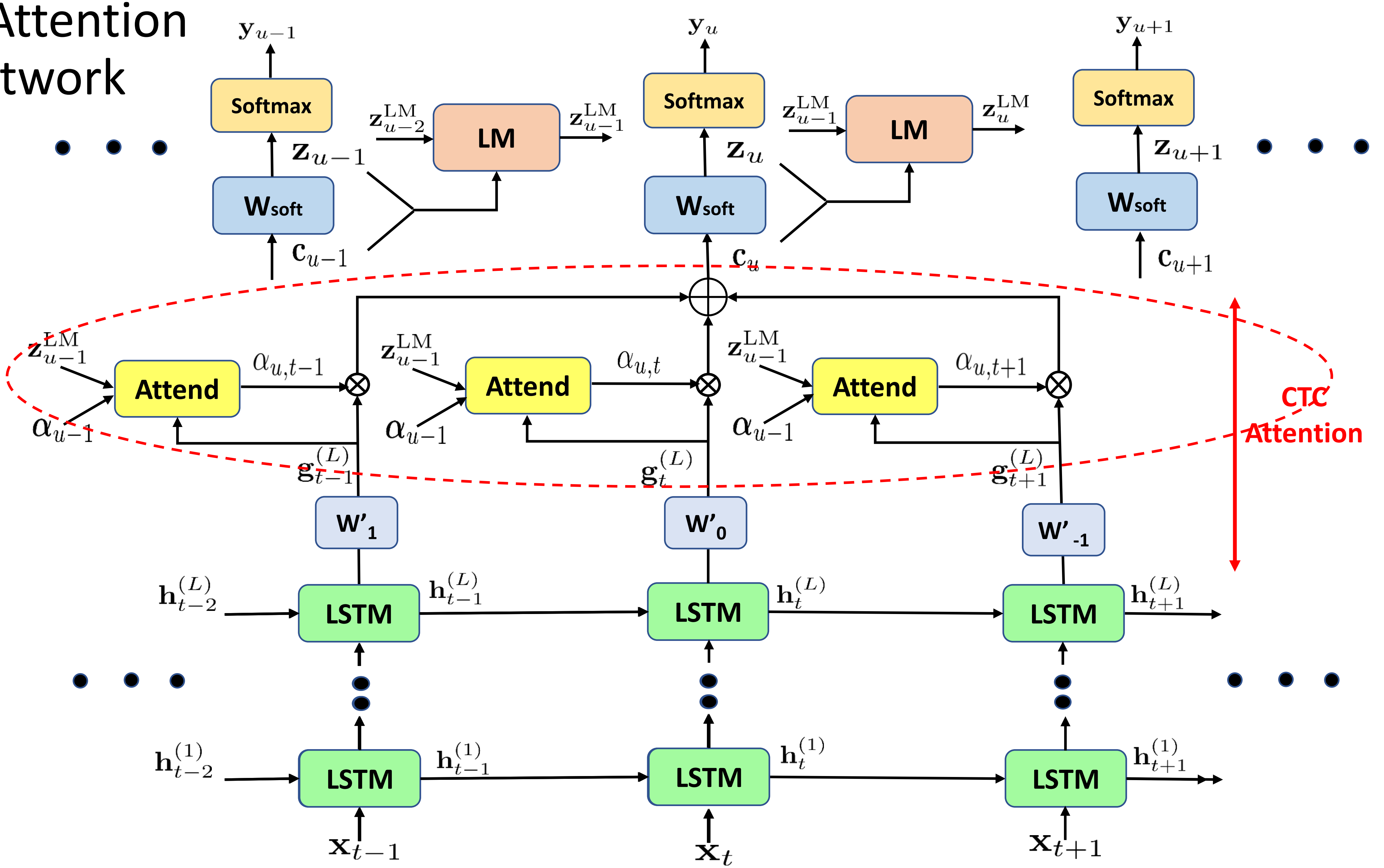
## CTC Annotate

$$\begin{aligned}\mathbf{c}_u &= \mathbf{W}' * \mathbf{h} \\ &= \sum_{t=u-\tau}^{u+\tau} \mathbf{W}'_{u-t} \mathbf{h}_t \\ &\triangleq \sum_{t=u-\tau}^{u+\tau} \mathbf{g}_t \\ &= \gamma \sum_{t=u-\tau}^{u+\tau} \alpha_{u,t} \mathbf{g}_t.\end{aligned}$$

$$\alpha_{u,t} = \frac{1}{C} \longrightarrow \text{Uniform attention}$$

$$\gamma = C$$

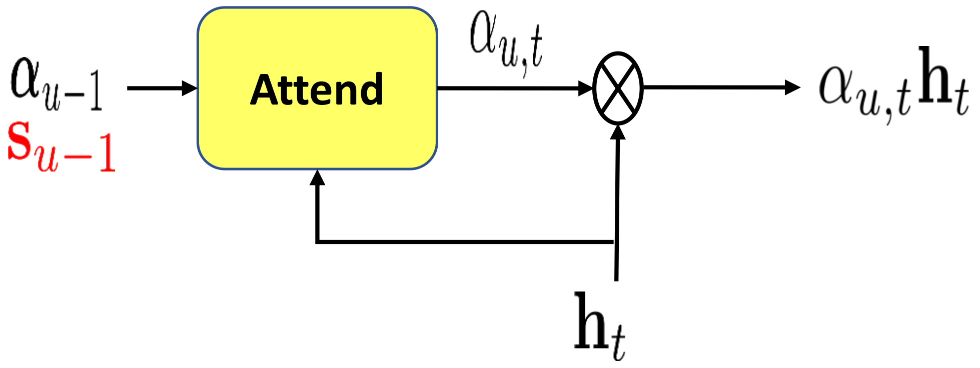
# CTC Attention Network



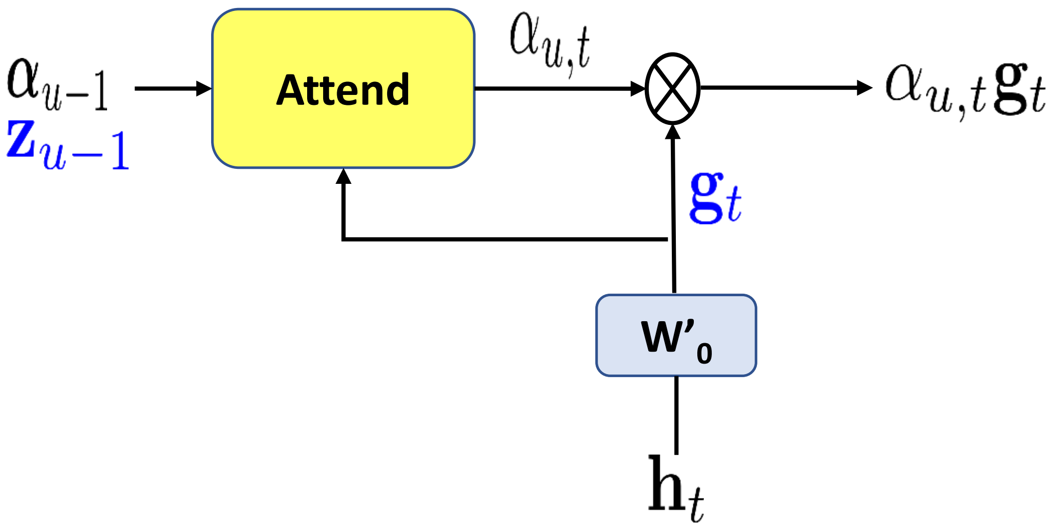
# CTC Attend

- Why need this? Ans: To move from uniform to non-uniform attention.
- In non-uniform attention, we weight the input features distinctively.
- How? Introduce an Attend block. No explicit decoder in CTC network. Replace the decoder state $\mathbf{s}_{u-1}$ in RNN-ED Attend with the logits $\mathbf{z}_{u-1}$ in CTC Attend.

<u>RNN-ED Attend</u>




<u>CTC Attend</u>



# CTC Attend

- Attend block is simply a single layer neural network.
- Scores $e_{u,t}$ are computed using $\mathbf{z}_{u-1}, \mathbf{g}_t$ .
- Softmax over scores computed over a small context window [u – $\tau$ , u + $\tau$ ].

## RNN-ED Attend

$$e_{u,t} = \begin{cases} \mathbf{v}^T \tanh(\mathbf{U}\mathbf{s}_{u-1} + \mathbf{W}\mathbf{h}_t + \mathbf{b}), & \text{(content)} \\ \mathbf{v}^T \tanh(\mathbf{U}\mathbf{s}_{u-1} + \mathbf{W}\mathbf{h}_t + \mathbf{V}\mathbf{f}_{u,t} + \mathbf{b}), & \text{(hybrid)} \end{cases}$$

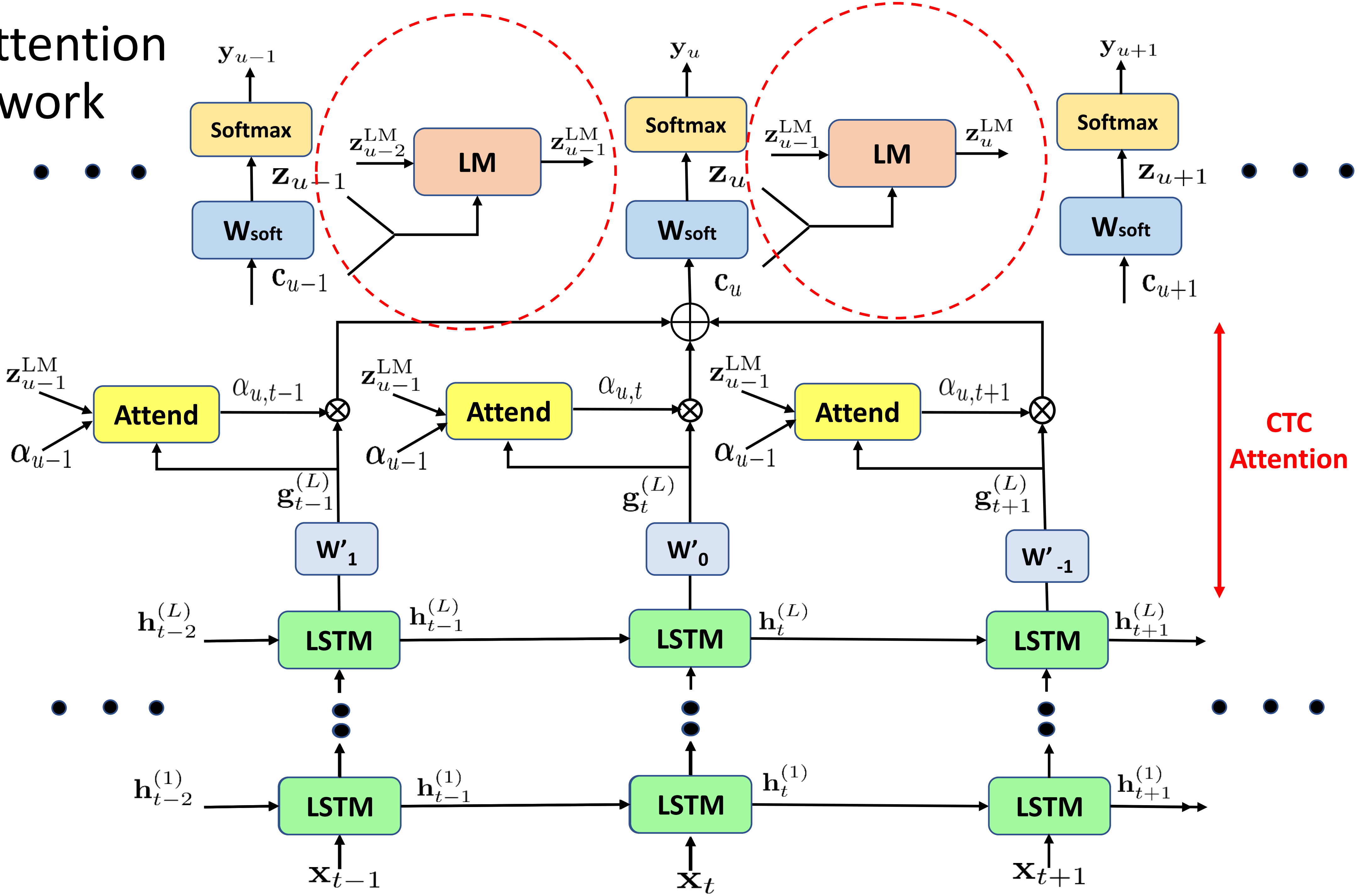
$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=1}^{T} \exp(e_{u,t'})}, \quad t = [1, T]$$

## CTC Attend

$$e_{u,t} = \begin{cases} \mathbf{v}^T \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{b}), & \text{(content)} \\ \mathbf{v}^T \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{V}\mathbf{f}_{u,t} + \mathbf{b}), & \text{(hybrid)} \end{cases}$$

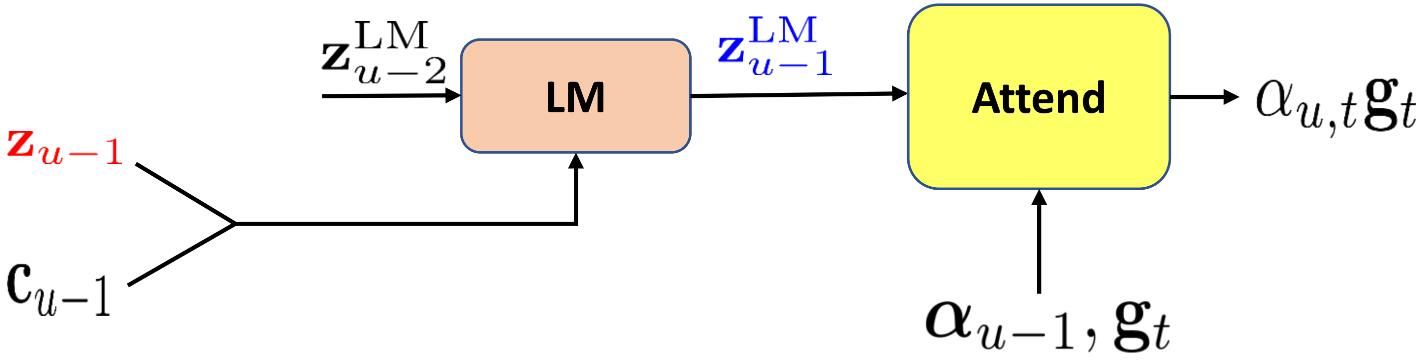
$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=u-\tau}^{u+\tau} \exp(e_{u,t'})}, \quad t = [u - \tau, u + \tau]$$

# CTC Attention Network



# Integration with Language Model (LM)

- Instead of using $\mathbf{z}_{u-1}$ in Attend(.), use $\mathbf{z}_{u-1}^{\text{LM}}$ obtained from another RNN/LSTM $\mathcal{H}(.)$ modeling a pseudo-LM.




- $$\boldsymbol{\alpha}_u = \text{Attend}(\mathbf{z}_{u-1}^{\text{LM}}, \boldsymbol{\alpha}_{u-1}, \mathbf{g})$$

$$\mathbf{z}_{u-1}^{\text{LM}} = \mathcal{H}(\mathbf{x}_{u-1}^{\text{LM}}, \mathbf{z}_{u-2}^{\text{LM}}), \quad \mathbf{x}_{u-1}^{\text{LM}} = \begin{bmatrix} \mathbf{z}_{u-1} \\ \mathbf{c}_{u-1} \end{bmatrix}$$

- The term $\mathbf{z}_{u-1}^{\text{LM}}$ captures long-term language information (n-gram like).
- However, because of blanks in CTC, it is only a pseudo-LM.

# Component-wise Attention (COMA)

- Instead of a single score per vector $\mathbf{g}_t$, we obtain a score for every component of $\mathbf{g}_t$ .

## CTC Attend (w/o COMA)

$$e_{u,t} = \begin{cases} \mathbf{v}^T \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{b}), & \text{(content)} \\ \mathbf{v}^T \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{V}\mathbf{f}_{u,t} + \mathbf{b}), & \text{(hybrid)} \end{cases}$$

$$e_{u,t} \in \mathbb{R}$$

$$t = [u - \tau, u + \tau]$$

## CTC Attend (w/ COMA)

$$\mathbf{e}_{u,t} = \begin{cases} \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{b}), & \text{(content)} \\ \tanh(\mathbf{U}\mathbf{z}_{u-1} + \mathbf{W}\mathbf{g}_t + \mathbf{V}\mathbf{f}_{u,t} + \mathbf{b}), & \text{(hybrid)} \end{cases}$$

$$\mathbf{e}_{u,t} \in \mathbb{R}^n$$

$$t = [u - \tau, u + \tau]$$

# Component-wise Attention (COMA)

- Keeping component fixed, take softmax across all time steps to get the COMA weights.

## CTC Attend (w/o COMA)

$$e_{u,t} \in \mathbb{R}$$

$$\begin{bmatrix} e_{u,t-\tau} & e_{u,t-\tau+1} & \cdots & e_{u,t} & \cdots & e_{u,t+\tau} \end{bmatrix}$$

softmax

## CTC Attend (w/ COMA)

$$\mathbf{e}_{u,t} \in \mathbb{R}^n$$

$$\begin{array}{c|cccccc} & t-\tau & t-\tau+1 & \ldots & t & \ldots & t+\tau \\ \hline 1 & e_{u,t-\tau}(1) & e_{u,t-\tau+1}(1) & \ldots & e_{u,t}(1) & \ldots & e_{u,t+\tau}(1) \\ 2 & e_{u,t-\tau}(2) & e_{u,t-\tau+1}(2) & \ldots & e_{u,t}(2) & \ldots & e_{u,t+\tau}(2) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ n & e_{u,t-\tau}(n) & e_{u,t-\tau+1}(n) & \ldots & e_{u,t}(n) & \ldots & e_{u,t+\tau}(n) \end{array}$$

softmax

# Component-wise Attention (COMA)

- Keeping component fixed, take softmax across all time steps to get the COMA weights.

## CTC Attend (w/o COMA)

$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=u-\tau}^{u+\tau} \exp(e_{u,t'})}$$

$$\mathbf{c}_u = \gamma \sum_{t=u-\tau}^{u+\tau} \alpha_{u,t} \mathbf{g}_t$$

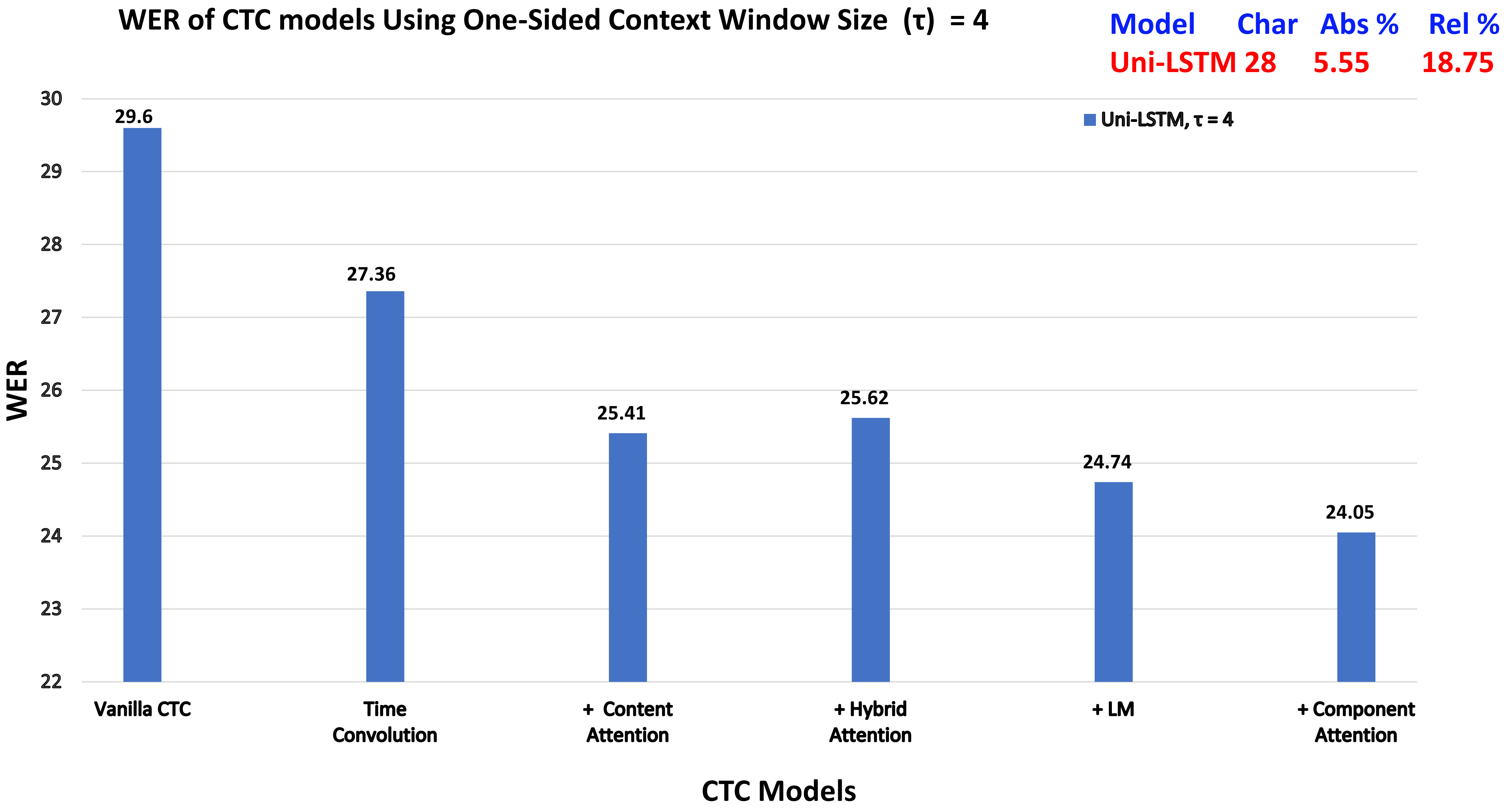
## CTC Attend (w/ COMA)

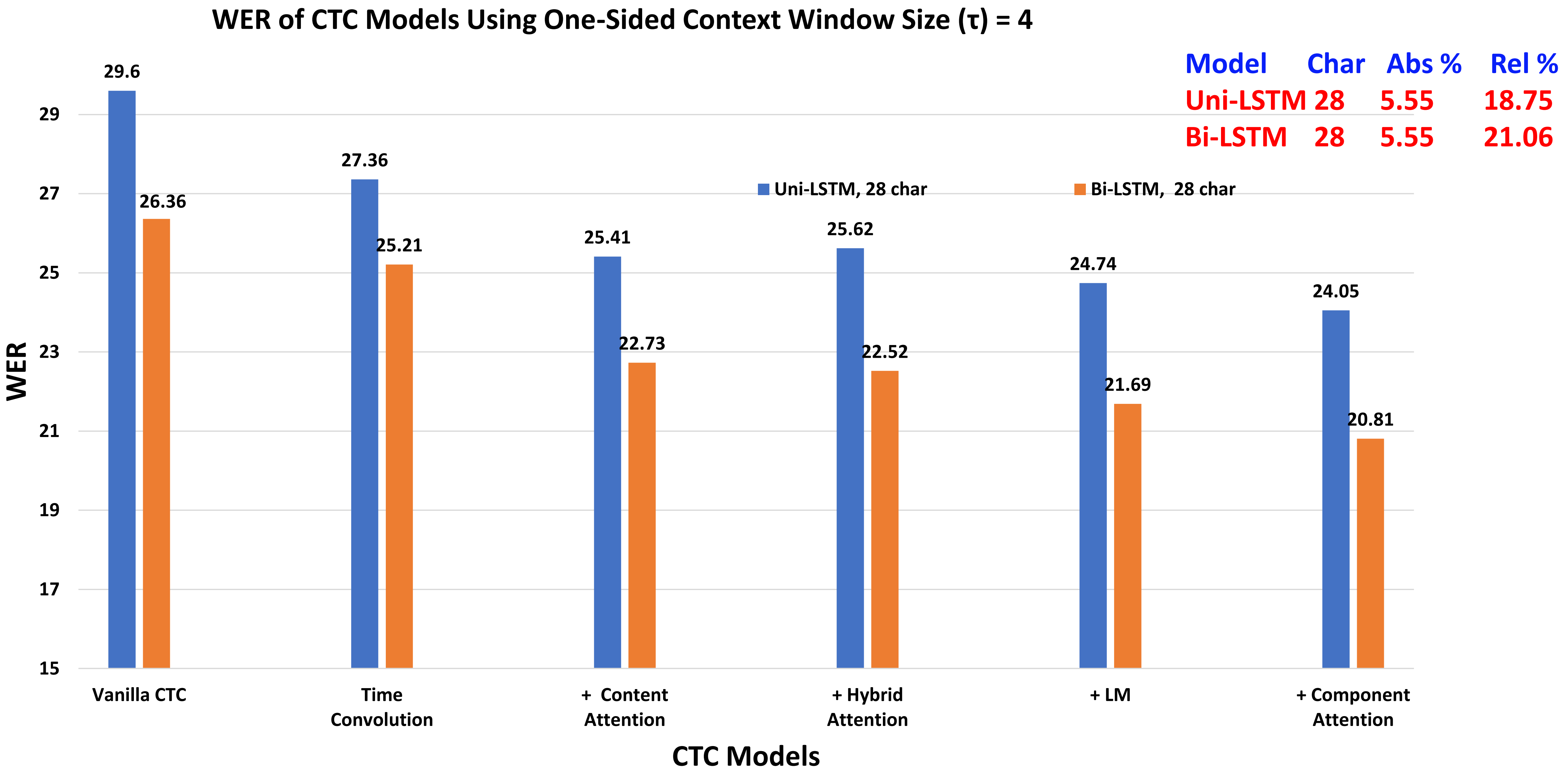
$$\alpha_{u,t,j} = \frac{\exp(e_{u,t,j})}{\sum_{t'=u-\tau}^{u+\tau} \exp(e_{u,t',j})}, \quad j = 1, \cdots, n$$

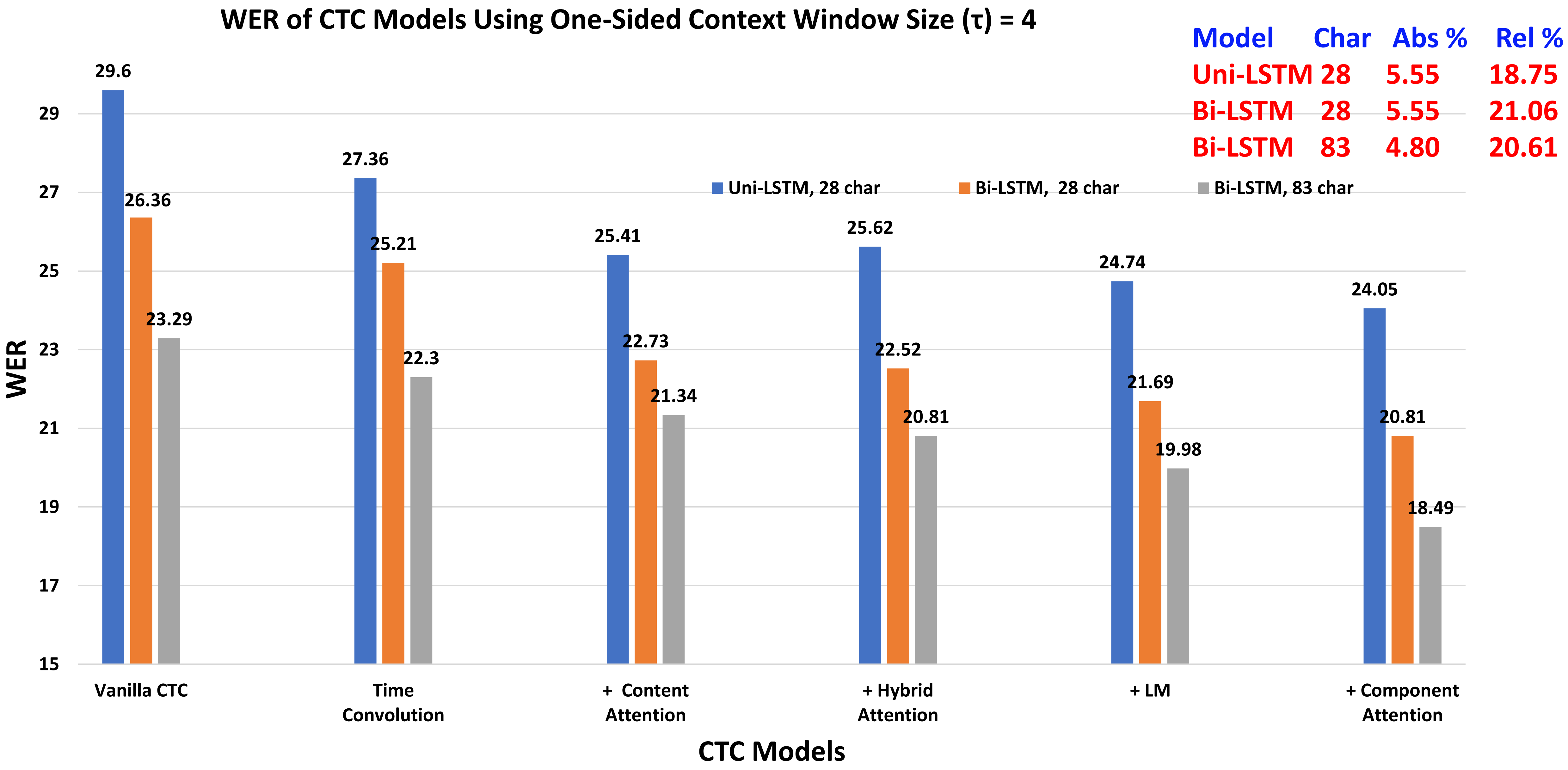
$$\mathbf{c}_u = \gamma \sum_{t=u-\tau}^{u+\tau} \boldsymbol{\alpha}_{u,t} \odot \mathbf{g}_t$$

# Experimental Set-Up

- Training Data: Cortana (Microsoft Voice Assistant)
  - 3400 hours (3.3 million utterances)
- Test Data: Cortana
  - 6 hours (5600 utterances)
- Model :
  - **Letter CTC (28 or 83 characters)**
  - 5 layers Uni-LSTM with 1024 memory cells or Bi-LSTM with 512 memory cells in each direction. Layer output is linearly projected to 512 dimensions.
- Greedy decoding
  - **No lexicon, No LM.** (Purest E2E)
- Log Mel Filterbank Energy (LMFE) Features:
  - base frame: 10 ms, Dim = 80
  - Input for Uni-LSTM: 8 base frames, shift = 3 base frames, Dim = 640
  - Input for Bi-LSTM: 3 base frames, shift = 3 base frames, Dim = 240


WER of CTC models Using One-Sided Context Window Size (τ) = 4
Model Char Abs % Rel %
Uni-LSTM 28 5.55 18.75
Uni-LSTM, τ = 4
WER
30
29
28
27
26
25
24
23
22
29.6
27.36
25.41
25.62
24.74
24.05
Vanilla CTC
Time Convolution
+ Content Attention
+ Hybrid Attention
+ LM
+ Component Attention
CTC Models


WER of CTC Models Using One-Sided Context Window Size (τ) = 4
Model Char Abs % Rel %
Uni-LSTM 28 5.55 18.75
Bi-LSTM 28 5.55 21.06
Uni-LSTM, 28 char
Bi-LSTM, 28 char
WER
29.6
26.36
27.36
25.21
25.41
22.73
25.62
22.52
24.74
21.69
24.05
20.81
29
27
25
23
21
19
17
15
Vanilla CTC
Time Convolution
+ Content Attention
+ Hybrid Attention
+ LM
+ Component Attention
CTC Models


WER of CTC Models Using One-Sided Context Window Size (τ) = 4
Model Char Abs % Rel %
Uni-LSTM 28 5.55 18.75
Bi-LSTM 28 5.55 21.06
Bi-LSTM 83 4.80 20.61
Uni-LSTM, 28 char
Bi-LSTM, 28 char
Bi-LSTM, 83 char
WER
29.6
26.36
23.29
27.36
25.21
22.3
25.41
22.73
21.34
25.62
22.52
20.81
24.74
21.69
19.98
24.05
20.81
18.49
29
27
25
23
21
19
17
15
Vanilla CTC
Time Convolution
+ Content Attention
+ Hybrid Attention
+ LM
+ Component Attention
CTC Models

# Gain on Larger Units

| CTC Models | WER |
|---|---|
| CD-phone CTC (with LM) | 9.28 |
| E2E CTC with 3-letter units | 13.28 |
| E2E CTC with 3-letter units + Attention | 11.36 |
| E2E CTC with mixed units (word + 3-letter) | **9.32** |
| E2E CTC with mixed units + **Attention** | **8.65** |

- More details in “Advancing Acoustic-to-Word CTC Model” at Friday’s End-to-End Speech Recognition II session, 13:30-15:30.

# Conclusions

- Soft-alignment training in CTC using
  - Time Convolution
  - Hybrid Attention
  - Implicit LM
  - Component Attention

- Reduction in WER:
  - 3400 hrs: ~ 20% relative with single letter unit. Significant gain with larger unit.
  - Similar improvement no matter whether we used weaker (Uni-LSTM CTC) or stronger baseline (Bi-LSTM CTC).

Thank You