# TTIC 31210:
# Advanced Natural Language Processing

Kevin Gimpel

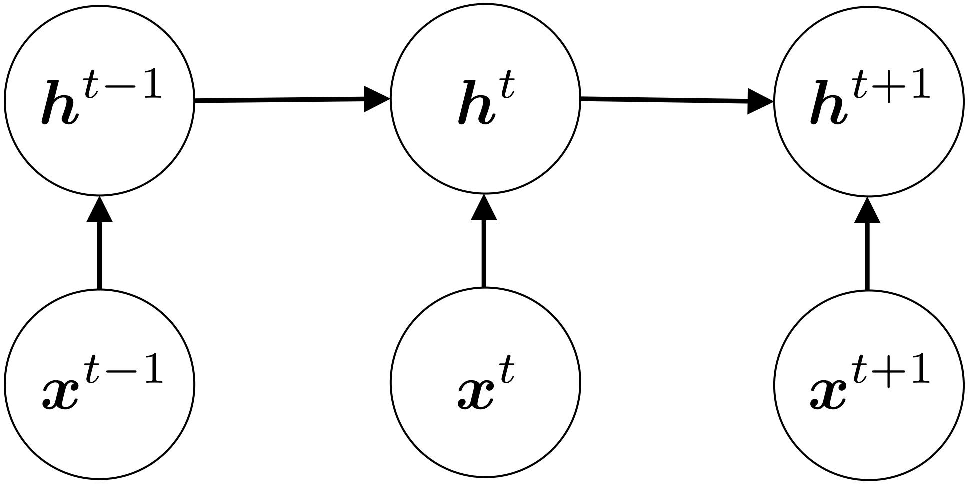
Spring 2017

# Lecture 8:
# Neural Machine Translation

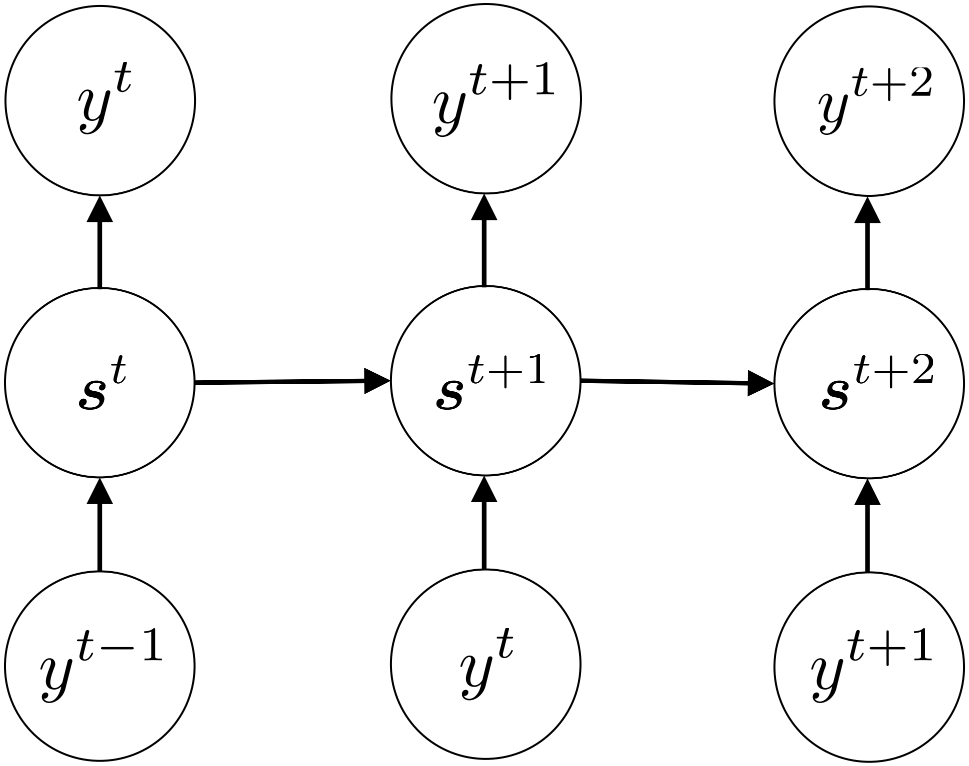
- Neural methods have transformed machine translation
- Neural Machine Translation (NMT) systems are typically based on sequence-to-sequence models with attention
- Today we'll describe a number of enhancements/modifications that improve translation quality

# Input RNN (“Encoder”)

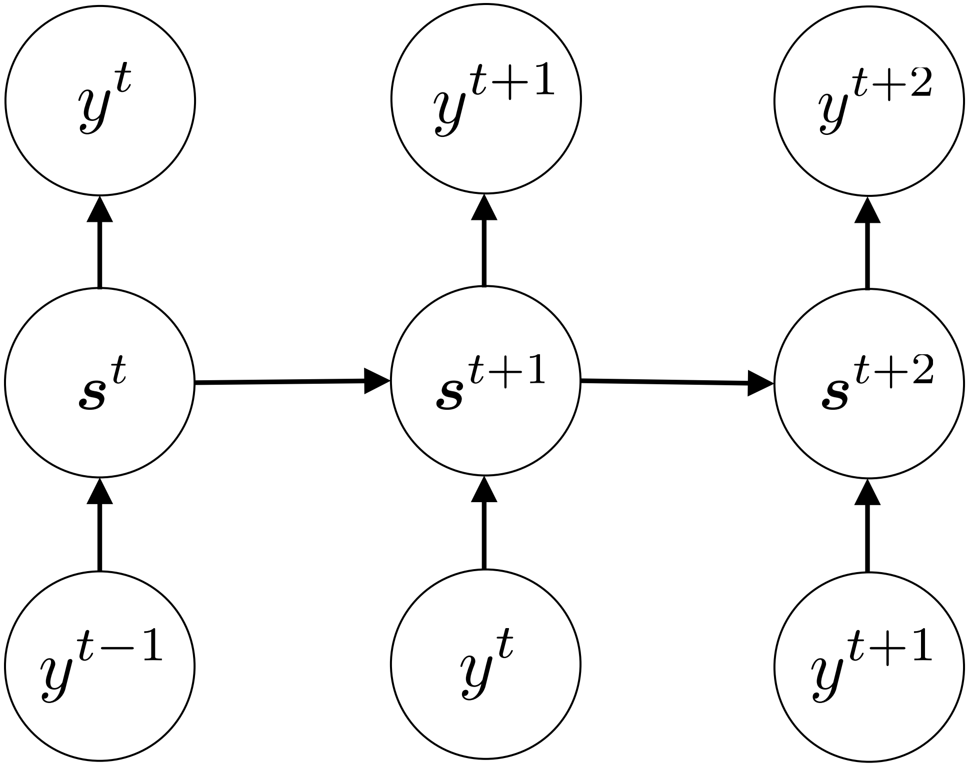
$$\boldsymbol{h}^t = \tanh\left(W^{(x)}\boldsymbol{x}^t + W^{(h)}\boldsymbol{h}^{t-1} + \boldsymbol{b}^{(h)}\right)$$



# Output RNN (“Decoder”)



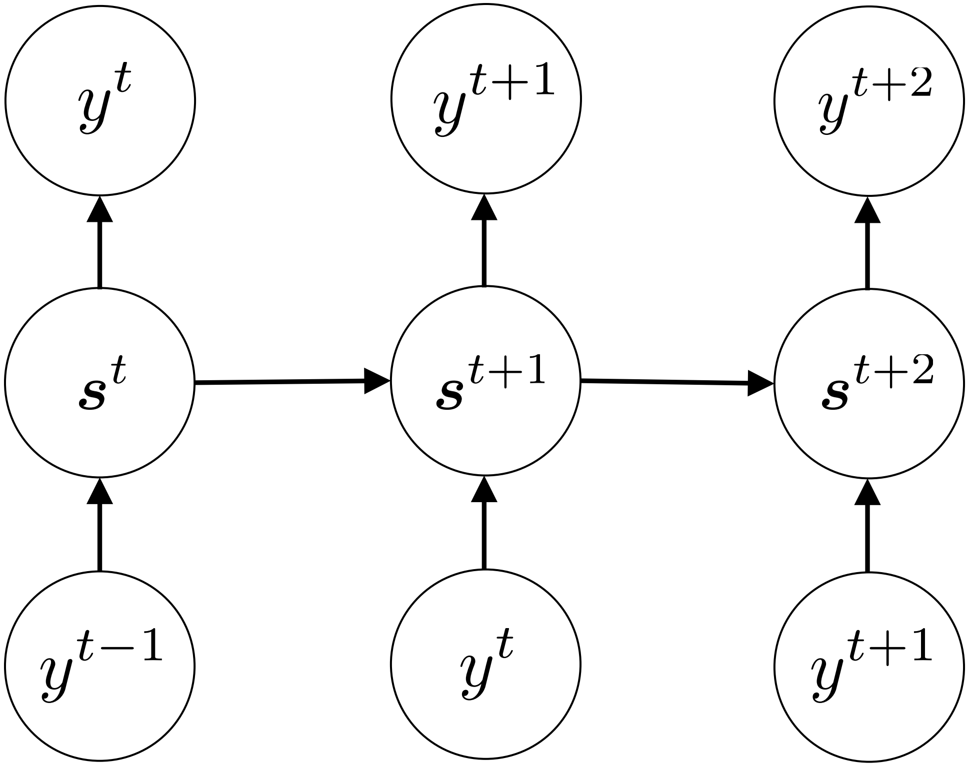
# Output RNN ("Decoder")



$$\boldsymbol{s}^t = \tanh\left(W^{(y)}\boldsymbol{y}^{t-1} + W^{(s)}\boldsymbol{s}^{t-1} + \boldsymbol{b}^{(s)}\right)$$

# Output RNN (“Decoder”)

$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left( emb(y)^\top \boldsymbol{s}^t \right)$$



$$\boldsymbol{s}^t = \tanh\left( W^{(y)} \boldsymbol{y}^{t-1} + W^{(s)} \boldsymbol{s}^{t-1} + \boldsymbol{b}^{(s)} \right)$$

# Attention for NMT

- Luong et al. (2015) simplify & generalize the model of Bahdanau, and compare different ways of defining attention

**"Effective Approaches to Attention-based Neural Machine Translation"**
**Luong et al. (2015)**

# Simplifying Attention for NMT

**Same in both:** $\boldsymbol{c}^t = \sum_{u=1}^{|\boldsymbol{x}|} \alpha^{t,u} \boldsymbol{h}^u$

**Bahdanau et al. (simplified a bit for clarity):**

$$\boldsymbol{s}^t = \tanh\left(W^{(y)}\boldsymbol{y}^{t-1} + W^{(s)}\boldsymbol{s}^{t-1} + W^{(c)}\boldsymbol{c}^t + \boldsymbol{b}^{(s)}\right)$$

$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left(emb(y)^\top [\boldsymbol{s}^t; \boldsymbol{c}^t]\right)$$

**Luong et al.:**

$\boldsymbol{s}^t$ just comes from decoder RNN

**How is this simpler?**

$$\tilde{\boldsymbol{s}}^t = \tanh\left(W^{(c)}[\boldsymbol{c}^t; \boldsymbol{s}^t]\right)$$

$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left(emb(y)^\top \tilde{\boldsymbol{s}}^t\right)$$

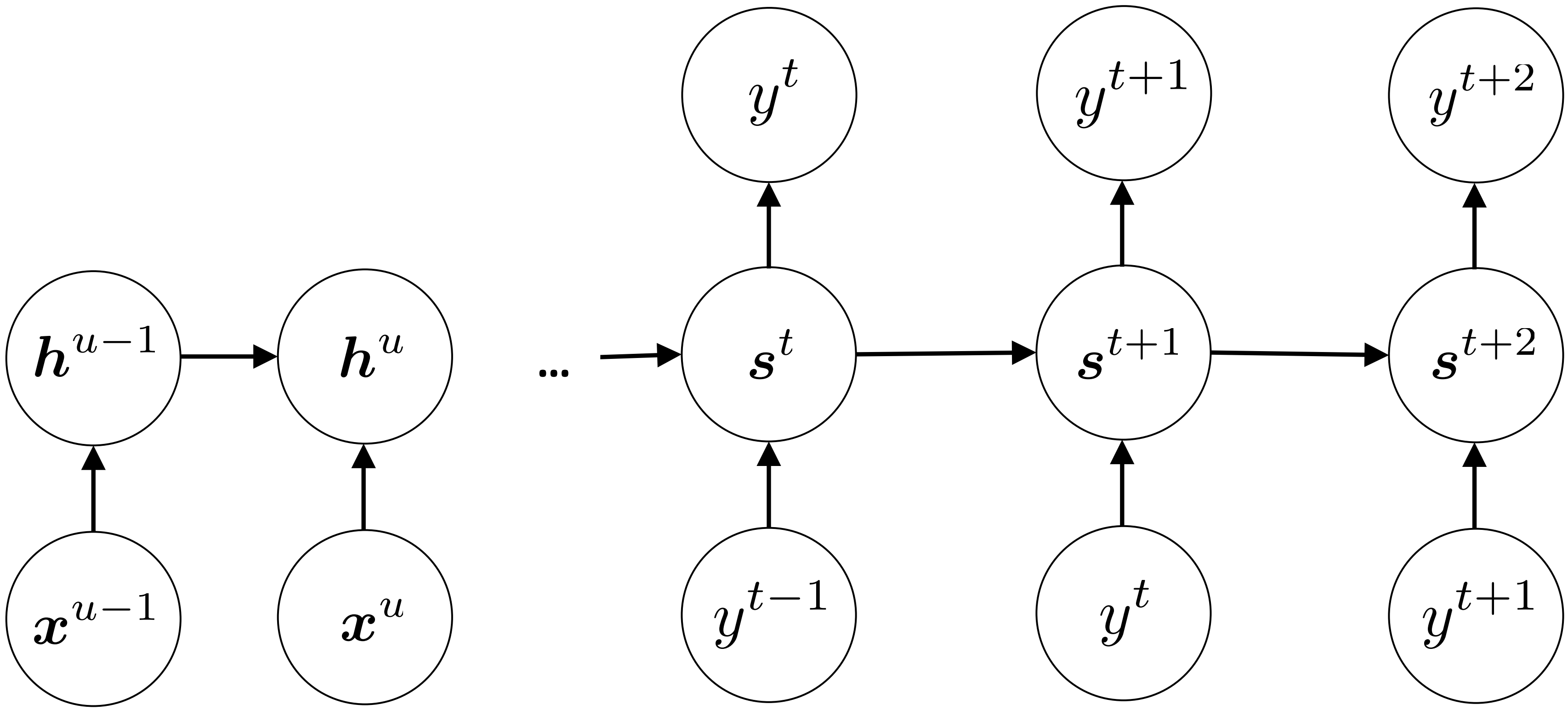
# Attention Functions

**Bahdanau et al.:**

$$\alpha^{t,u} \propto \exp\{att(\boldsymbol{s}^{t-1}, \boldsymbol{h}^u)\}$$

**Luong et al.:**

$$\alpha^{t,u} \propto \exp\{att(\boldsymbol{s}^{t}, \boldsymbol{h}^u)\}$$



# Global Attention

**global** = computed over all hidden vectors of input

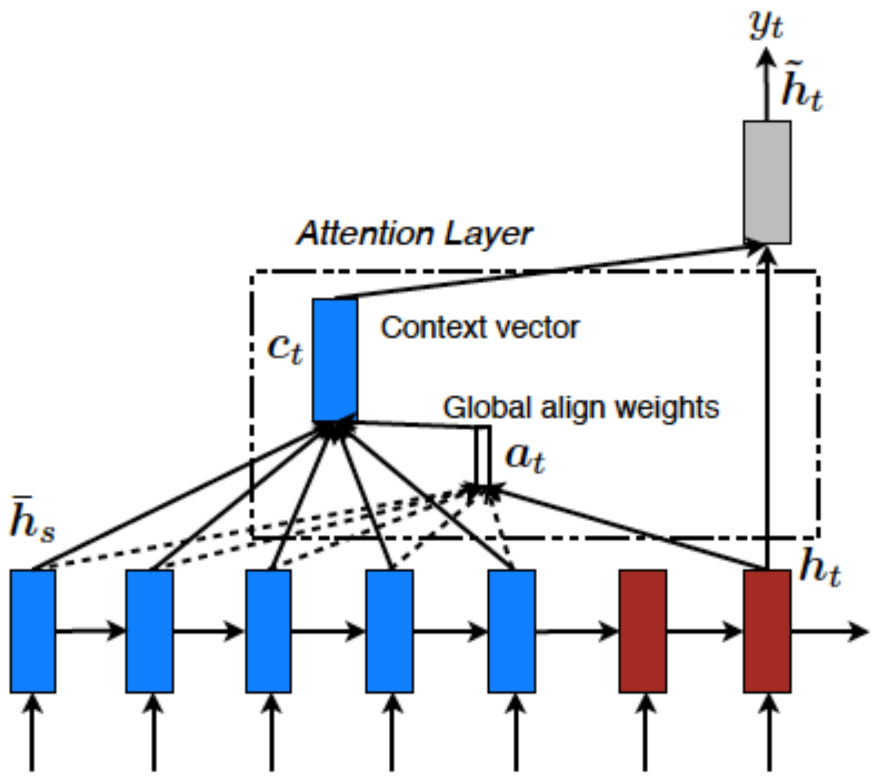


Figure 2: **Global attentional model** – at each time step $t$, the model infers a *variable-length* alignment weight vector $a_t$ based on the current target state $h_t$ and all source states $\bar{h}_s$. A global context vector $c_t$ is then computed as the weighted average, according to $a_t$, over all the source states.

Luong et al. (2015)

# Global Content-Based Attention Functions

**global** = computed over all hidden vectors of input

**content-based** = attention function looks at source hidden vectors

**dot product ("dot"):** $att(\boldsymbol{s}^t, \boldsymbol{h}^u) = {\boldsymbol{s}^t}^\top \boldsymbol{h}^u$

**bilinear ("general"):** $att(\boldsymbol{s}^t, \boldsymbol{h}^u) = {\boldsymbol{s}^t}^\top W^{(a)} \boldsymbol{h}^u$

**feed-forward ("concat"):** $att(\boldsymbol{s}^t, \boldsymbol{h}^u) = {\mathbf{w}^{(a)}}^\top [\boldsymbol{s}^t; \boldsymbol{h}^u]$

**parameter vector**

**Luong et al. (2015)**

# Global **Location-Based** Attention Function

**global** = computed over all hidden vectors of input

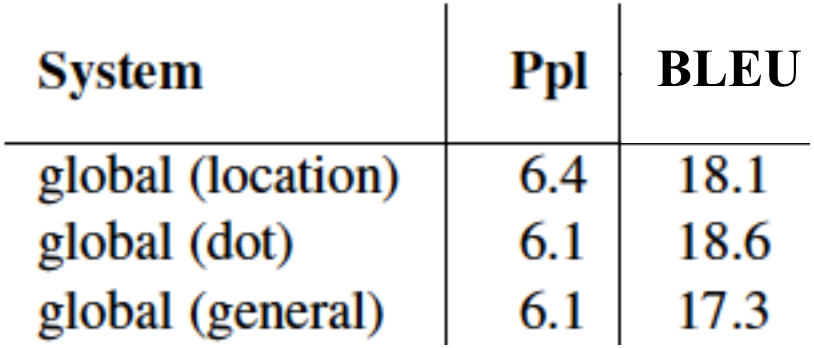
**location-based** = attention function does **not** look at source hidden vectors themselves, just positions:

$$att(\boldsymbol{s}^t, u) = {\boldsymbol{s}^t}^\top \mathbf{w}^{(u)}$$

**parameter vector for position *u* in source sentence**

**Luong et al. (2015)**

# Results



| System | Ppl | BLEU |
|---|---|---|
| global (location) | 6.4 | 18.1 |
| global (dot) | 6.1 | 18.6 |
| global (general) | 6.1 | 17.3 |

**feed-forward ("concat") did not work well!**

**Luong et al. (2015)**

# Local Attention

**local** = computed over a subset of input hidden vectors

at decoder step $t$,
find position $p_t$ in source sentence,

compute attention over a window centered at that position in the source sentence

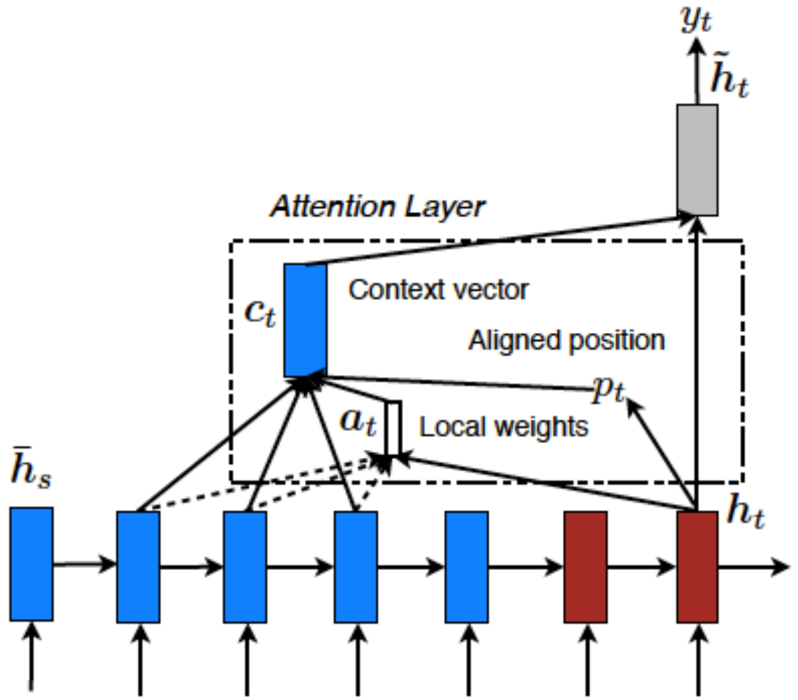
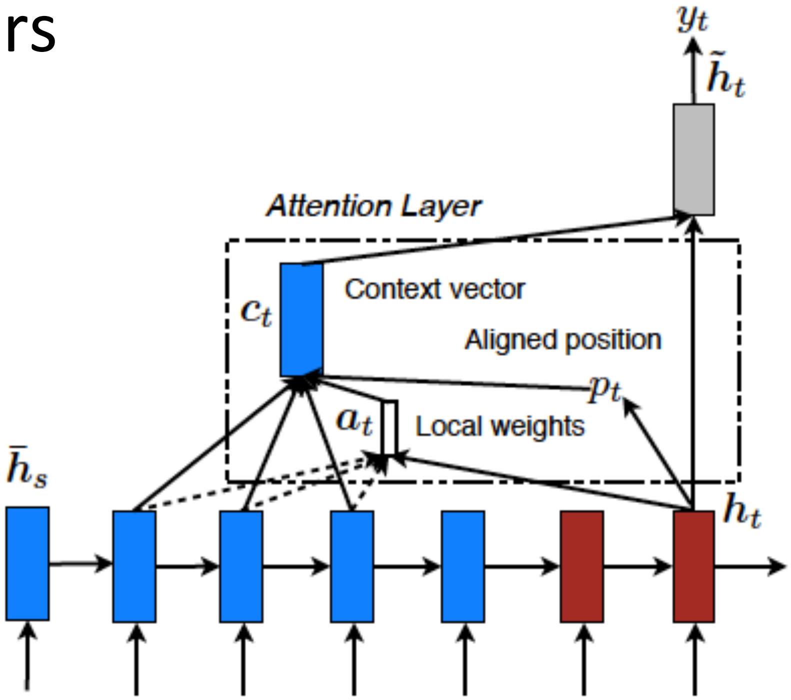


Figure 3: **Local attention model** – the model first predicts a single aligned position $p_t$ for the current target word. A window centered around the source position $p_t$ is then used to compute a context vector $c_t$, a weighted average of the source hidden states in the window. The weights $a_t$ are inferred from the current target state $h_t$ and those source states $\bar{h}_s$ in the window.

# Local Attention

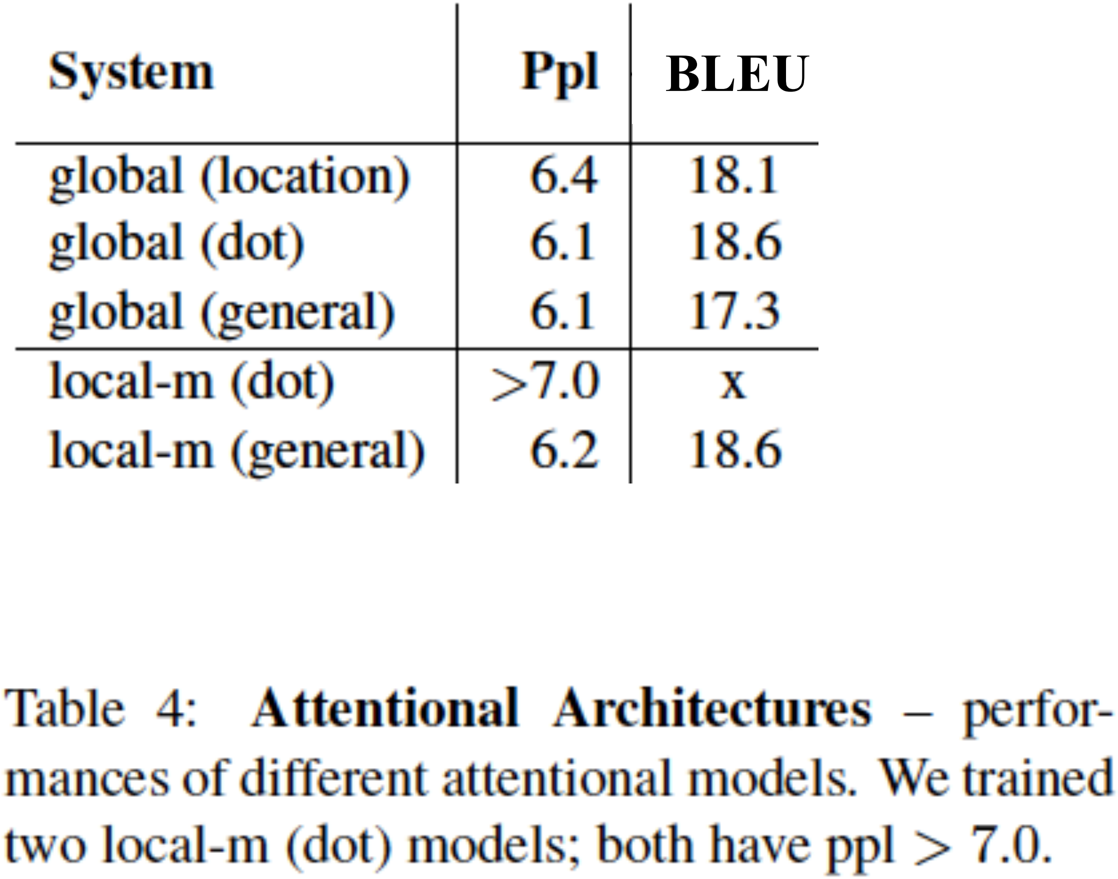
**local-m**: set $p_t = t$, assumes roughly monotonic alignment between decoder positions and source sentence positions

**local-p:** predict $p_t$ based on decoder hidden state and some additional parameters



Luong et al. (2015)

# Results

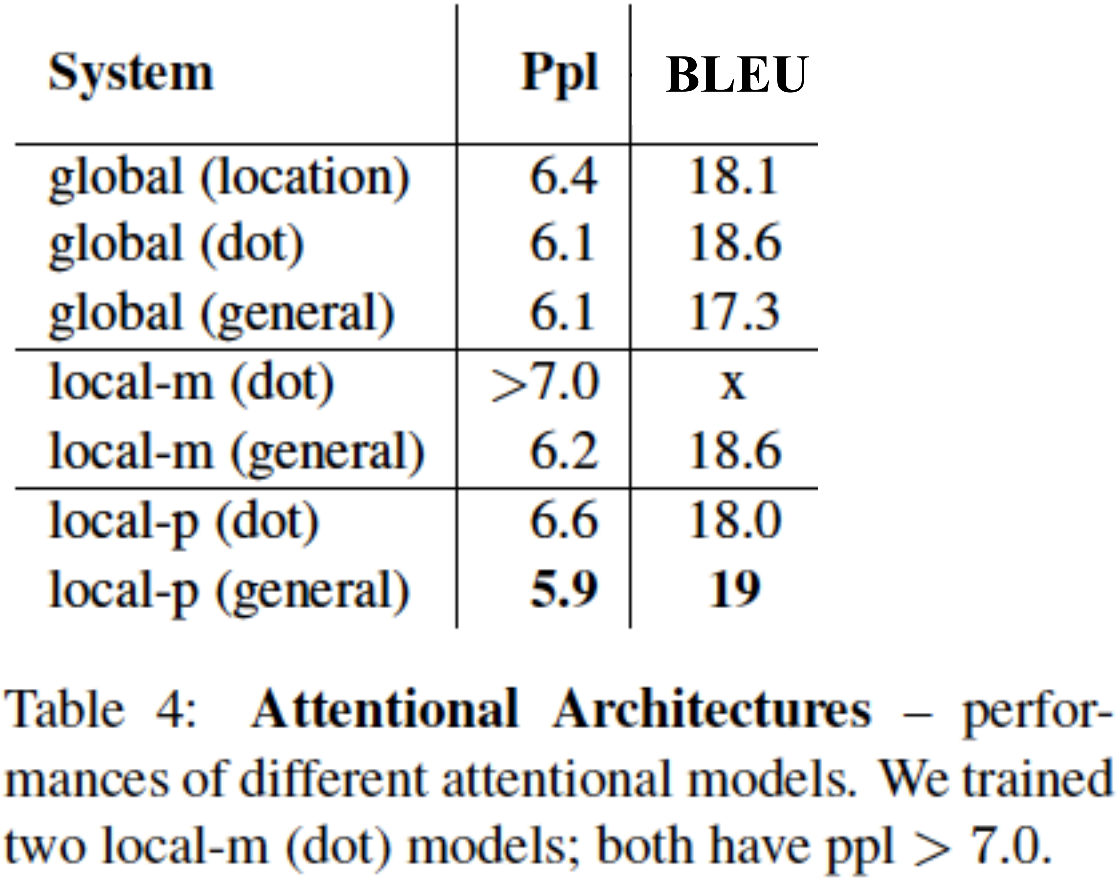


| System | Ppl | BLEU |
|---|---|---|
| global (location) | 6.4 | 18.1 |
| global (dot) | 6.1 | 18.6 |
| global (general) | 6.1 | 17.3 |
| local-m (dot) | >7.0 | x |
| local-m (general) | 6.2 | 18.6 |

Table 4: **Attentional Architectures** – performances of different attentional models. We trained two local-m (dot) models; both have ppl $> 7.0$.

“Effective Approaches to Attention-based Neural Machine Translation”
Luong et al. (2015)

# Results



| System | Ppl | BLEU |
|---|---|---|
| global (location) | 6.4 | 18.1 |
| global (dot) | 6.1 | 18.6 |
| global (general) | 6.1 | 17.3 |
| local-m (dot) | >7.0 | x |
| local-m (general) | 6.2 | 18.6 |
| local-p (dot) | 6.6 | 18.0 |
| local-p (general) | **5.9** | **19** |

Table 4: **Attentional Architectures** – performances of different attentional models. We trained two local-m (dot) models; both have ppl > 7.0.

"Effective Approaches to Attention-based Neural Machine Translation"
Luong et al. (2015)

# “Input Feeding” of Decoder Hidden States

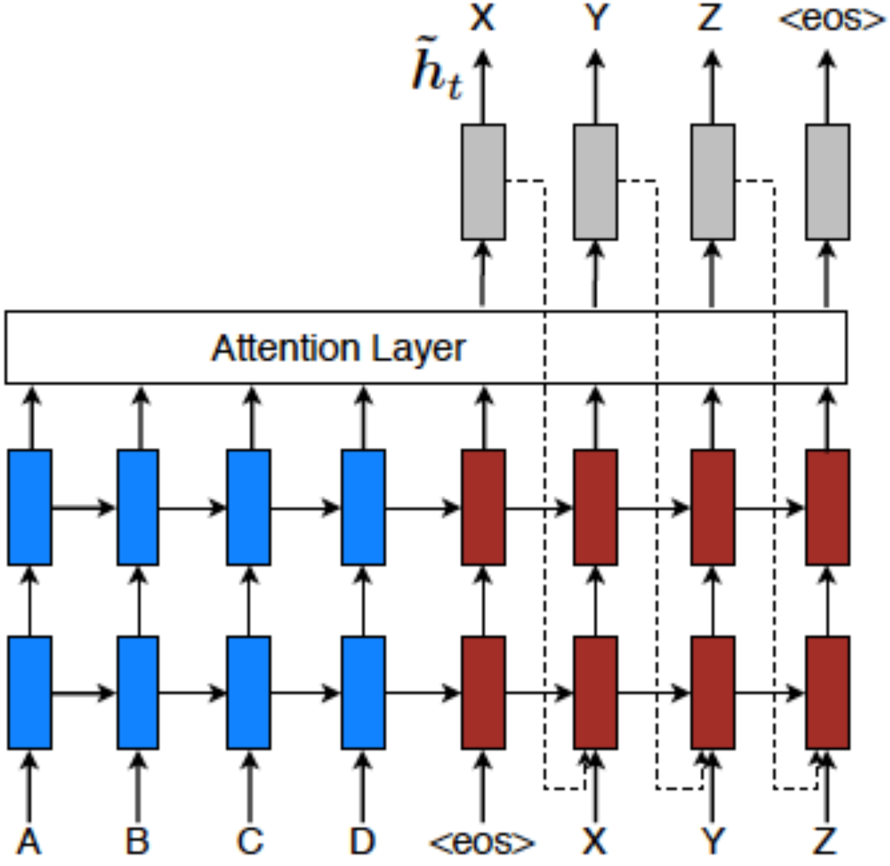


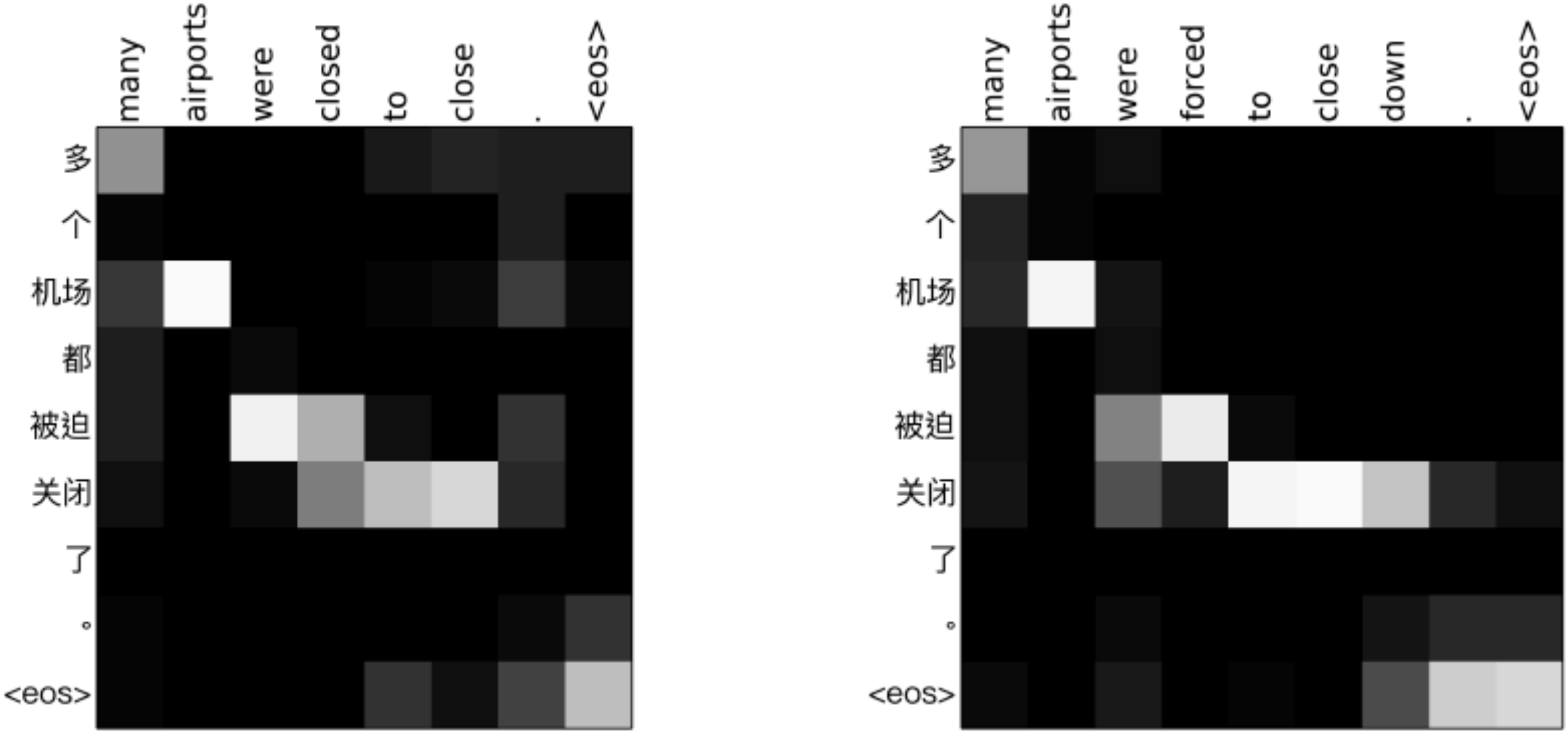
Figure 4: **Input-feeding approach** – Attentional vectors $\tilde{h}_t$ are fed as inputs to the next time steps to inform the model about past alignment decisions.

**Luong et al. (2015)**

# Modeling Coverage

- NMT sometimes doesn't translate all source words, or translates them multiple times

**"Modeling Coverage for Neural Machine Translation"**
**Tu et al. (2016)**



(a) Over-translation and under-translation generated by NMT.

(b) Coverage model alleviates the problems of over-translation and under-translation.

Figure 1: Example translations of (a) NMT without coverage, and (b) NMT with coverage. In conventional NMT without coverage, the Chinese word "*guānbì*" is over translated to "*close*(*d*)" twice, while "*bèipò*" (means "*be forced to*") is mistakenly untranslated. Coverage model alleviates these problems by tracking the "coverage" of source words.

**"Modeling Coverage for Neural Machine Translation" Tu et al. (2016)**

# Results: Modeling Coverage

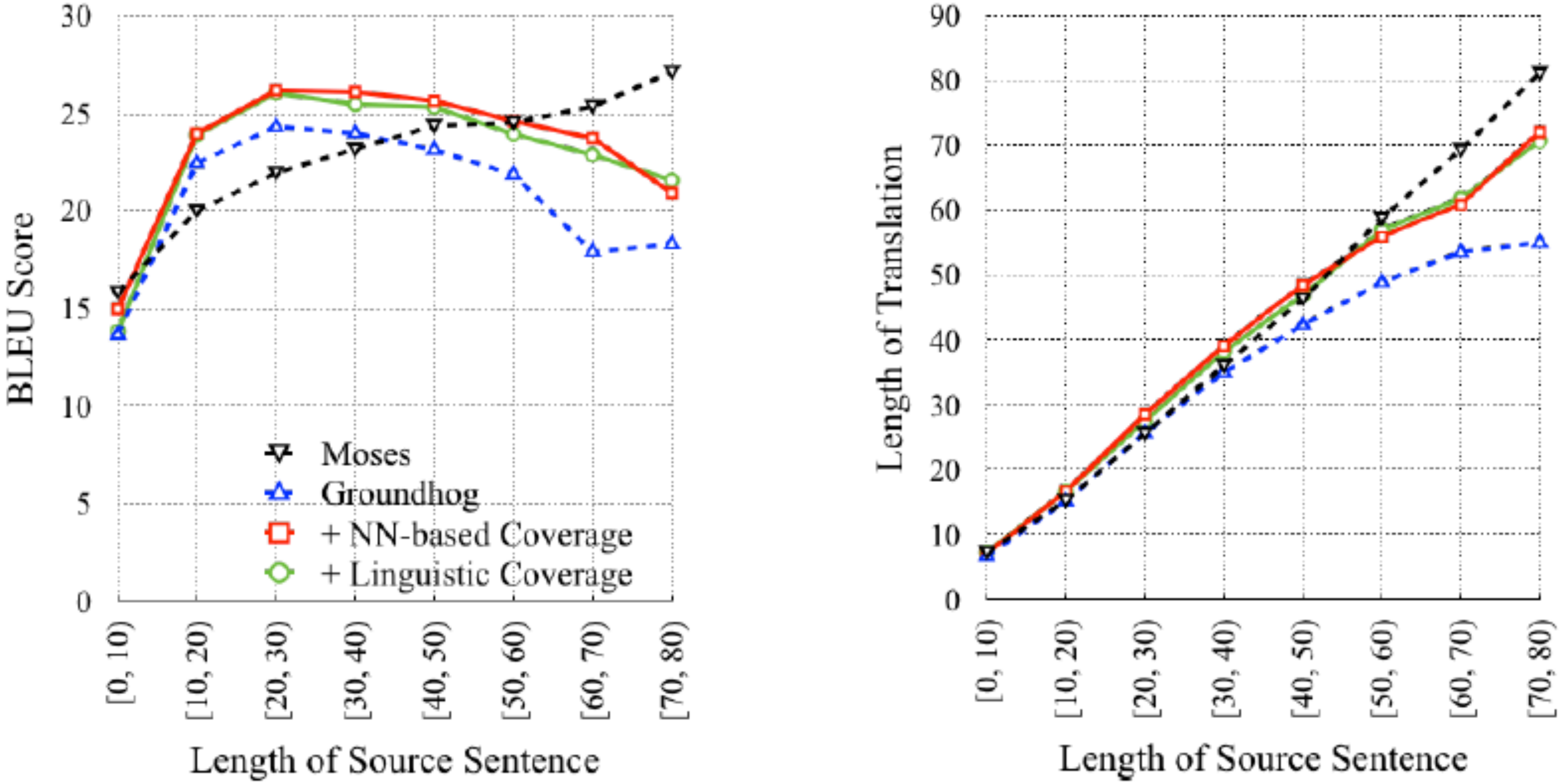


Figure 6: Performance of the generated translations with respect to the lengths of the input sentences. Coverage models alleviate under-translation by producing longer translations on long sentences.

**"Modeling Coverage for Neural Machine Translation"**
**Tu et al. (2016)**

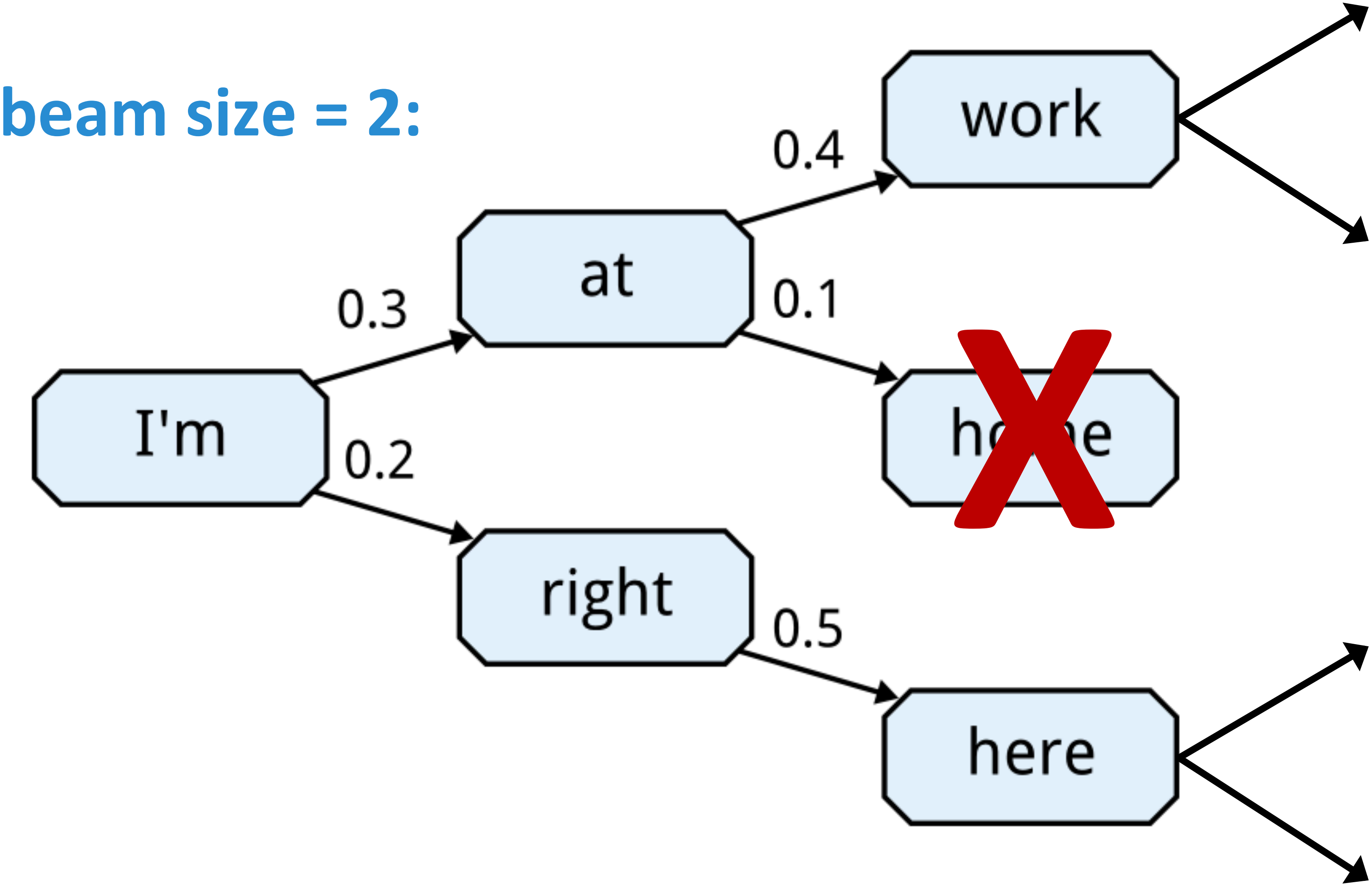
# Inference

# Beam Search

- to find a translation, greedy search just picks most-probable word at each position
- but does this give us any guarantees about the entire translation?
- beam search can be used to approximately find the most-probable complete translation

# Beam Search

**Let beam size = 2:**



**“Chat Smarter with Allo”**
**Pranav Khaitan, Google Research Blog**

# Learning

# Concern

- there’s a mismatch between training and test!
- (what is it?)

# Scheduled Sampling

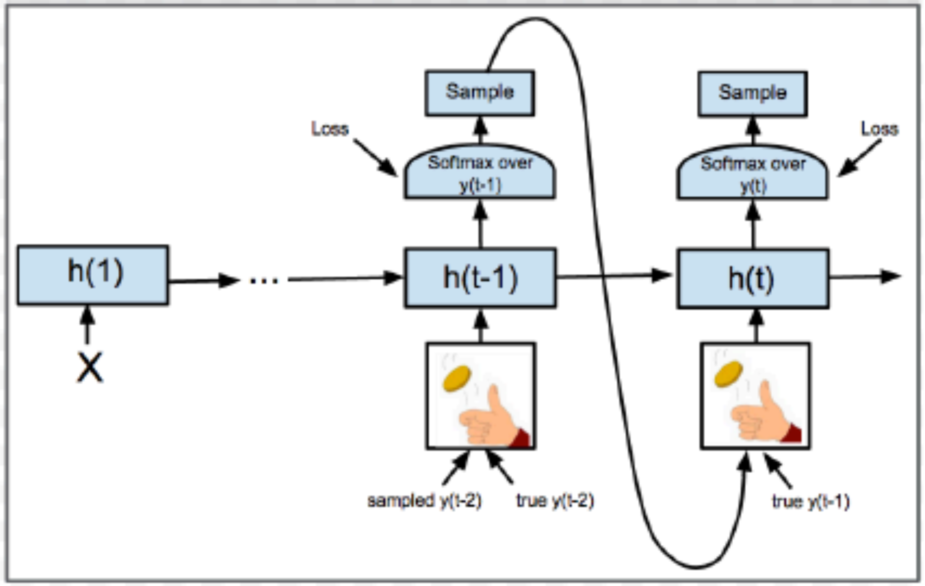


Figure 1: Illustration of the Scheduled Sampling approach, where one flips a coin at every time step to decide to use the true previous token or one sampled from the model itself.

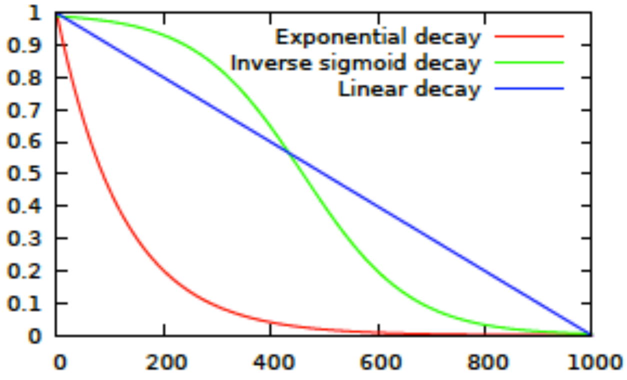


Figure 2: Examples of decay schedules.

**"Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks"**
**Bengio et al. (2015)**

# Scheduled Sampling Results

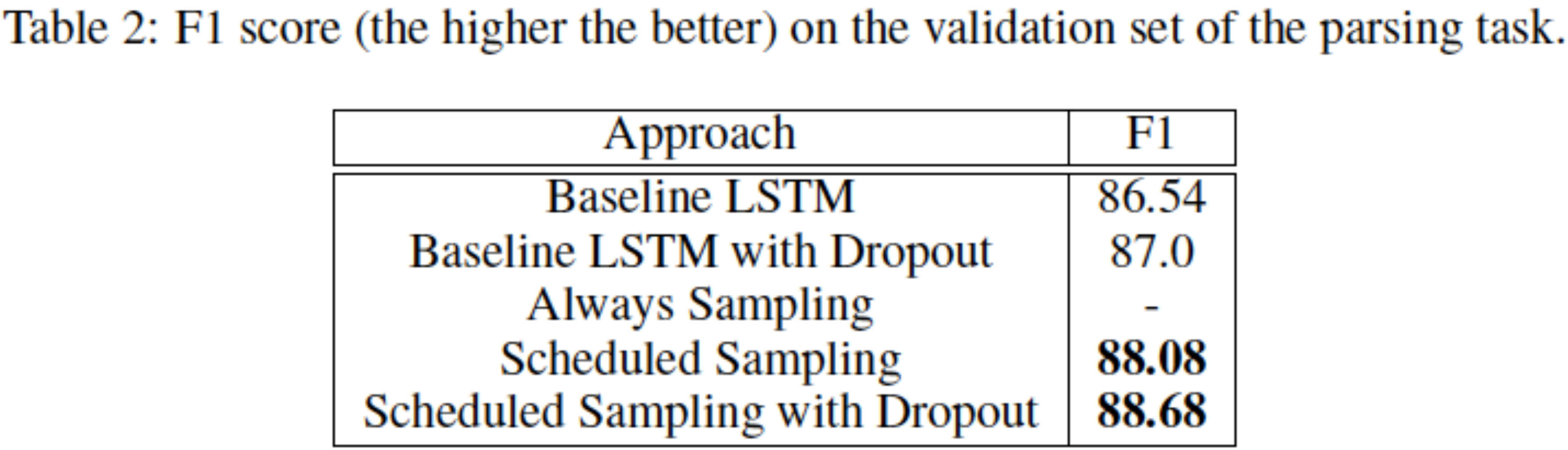


Table 2: F1 score (the higher the better) on the validation set of the parsing task.

| Approach | F1 |
|---|---|
| Baseline LSTM | 86.54 |
| Baseline LSTM with Dropout | 87.0 |
| Always Sampling | - |
| Scheduled Sampling | **88.08** |
| Scheduled Sampling with Dropout | **88.68** |

**“Always Sampling” did not work well!**

**“Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks”**
**Bengio et al. (2015)**