

CS769 Advanced NLP

# Machine Translation

Junjie Hu



Slides adapted from Austin

<https://junjiehu.github.io/cs769-spring23/>

# Goal for Today

- Background & Parallel Corpus
- **Noisy Channel MT (SMT, non-parametric models)**
  - Lexical Translation
  - Word Alignment
- **Neural Machine Translation (parametric models)**
  - Architecture: LSTM, CNN, Transformer
  - Multilingual NMT
- **Hybrid MT (non-parametric + parametric MT)**
  - Interpolation / Retrieval MT
  - Prompt MT
- Open Research Problems on NMT

One naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: **'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.'**



Warren Weaver to Norbert Wiener, March, 1947

# Parallel Corpus

- We are given a corpus of sentence pairs in two languages to train our machine translation models.
- Source language is also called foreign language, denoted as **f**.
- Conventionally (in earlier studies before NMT) target language is usually referred to English, denoted as **e**.

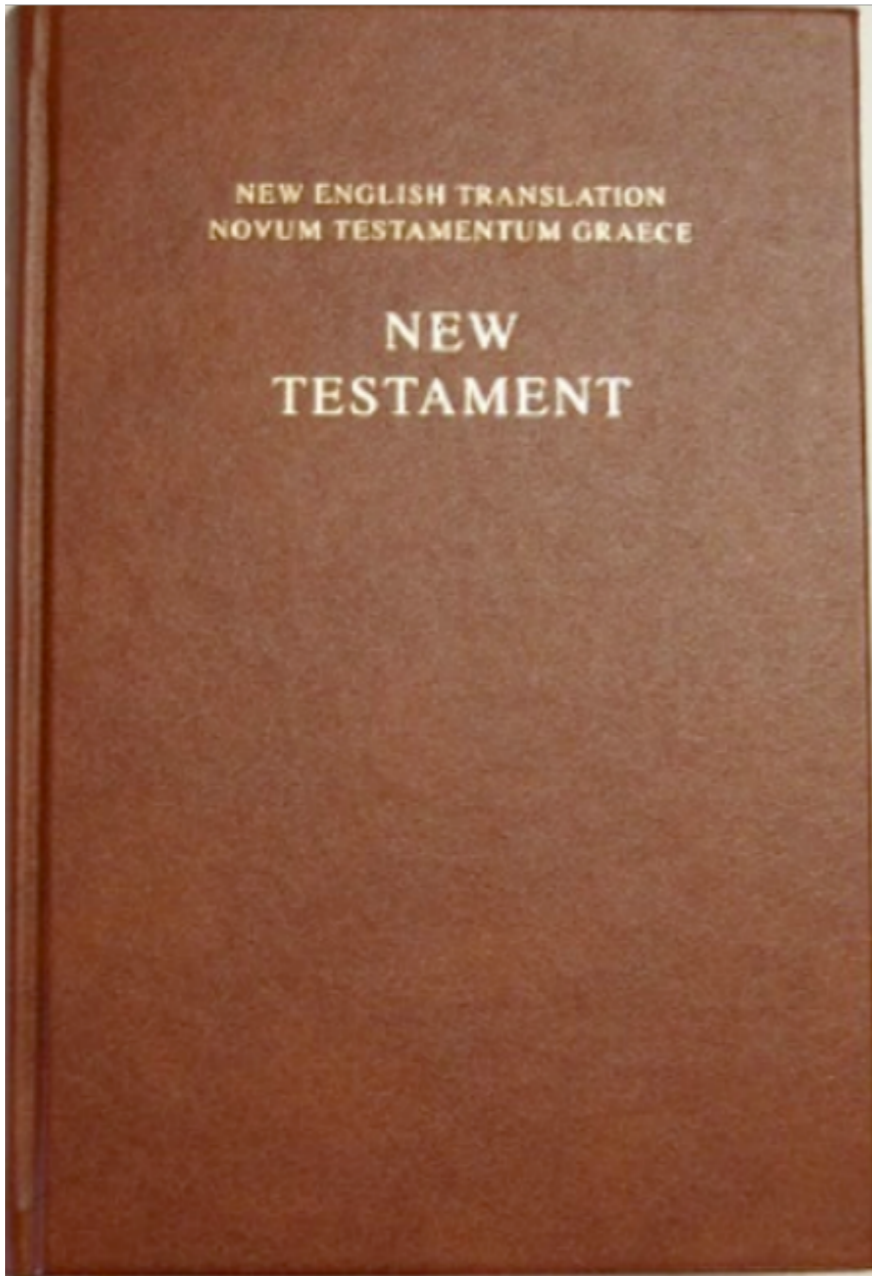
# Parallel Corpus

		<b>CLASSIC SOUPS</b>		Sm.	Lg.
清	燉	雞	湯 57.	House Chicken Soup (Chicken, Celery, Potato, Onion, Carrot) .....	1.50 2.75
雞	飯	湯	58.	Chicken Rice Soup .....	1.85 3.25
雞	麵	湯	59.	Chicken Noodle Soup .....	1.85 3.25
廣	東	雲	吞 60.	Cantonese Wonton Soup.....	1.50 2.75
蕃	茄	蛋	湯 61.	Tomato Clear Egg Drop Soup .....	1.65 2.95
雲	吞	湯	62.	Regular Wonton Soup .....	1.10 2.10
酸	辣	湯	63. ●	Hot & Sour Soup .....	1.10 2.10
蛋	花	湯	64.	Egg Drop Soup.....	1.10 2.10
雲	蛋	湯	65.	Egg Drop Wonton Mix.....	1.10 2.10
豆	腐	菜	湯 66.	Tofu Vegetable Soup .....	NA 3.50
雞	玉	米	湯 67.	Chicken Corn Cream Soup .....	NA 3.50
蟹	肉	玉	米 湯 68.	Crab Meat Corn Cream Soup.....	NA 3.50
海	鮮	湯	69.	Seafood Soup.....	NA 3.50

# Parallel Corpus

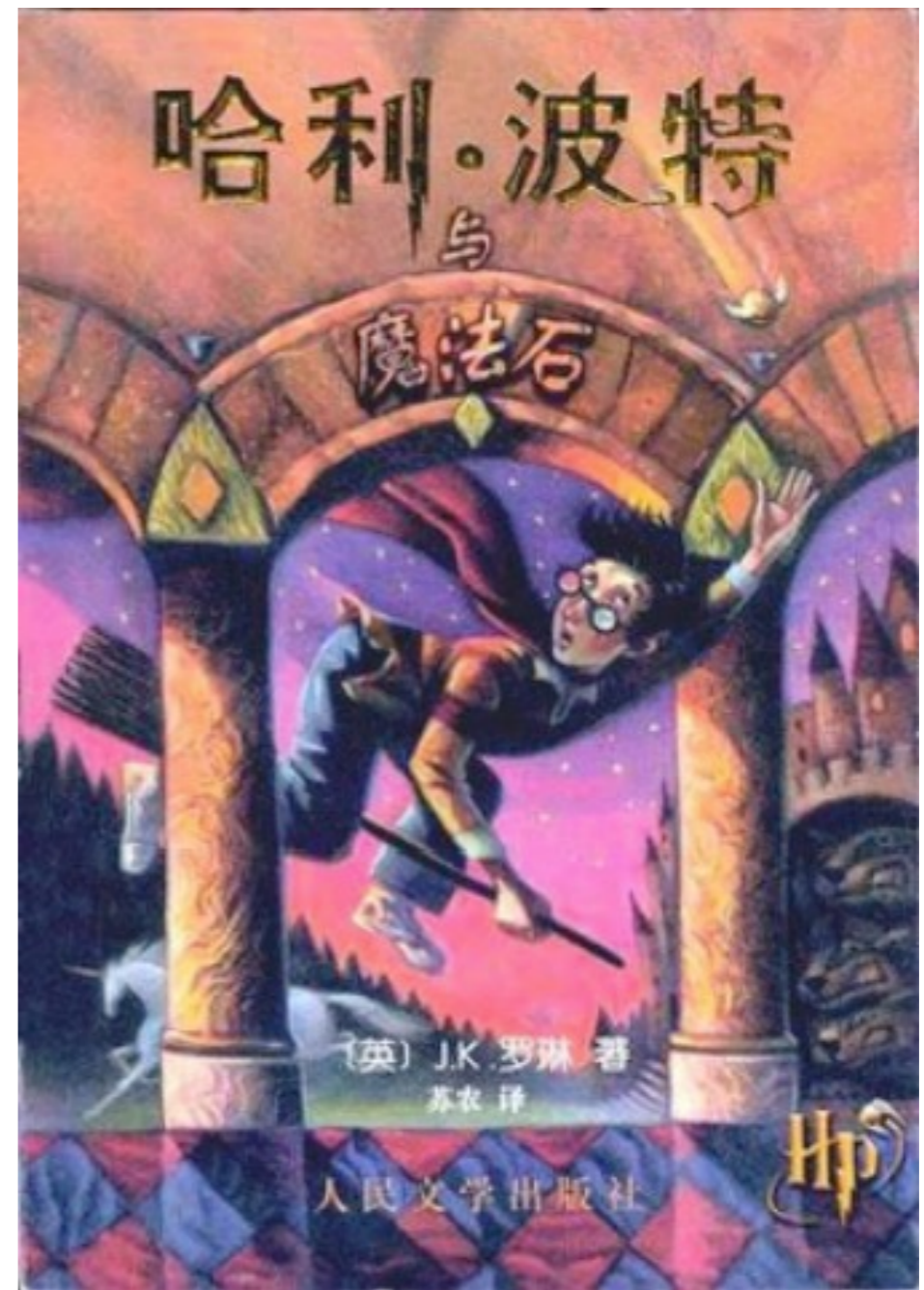
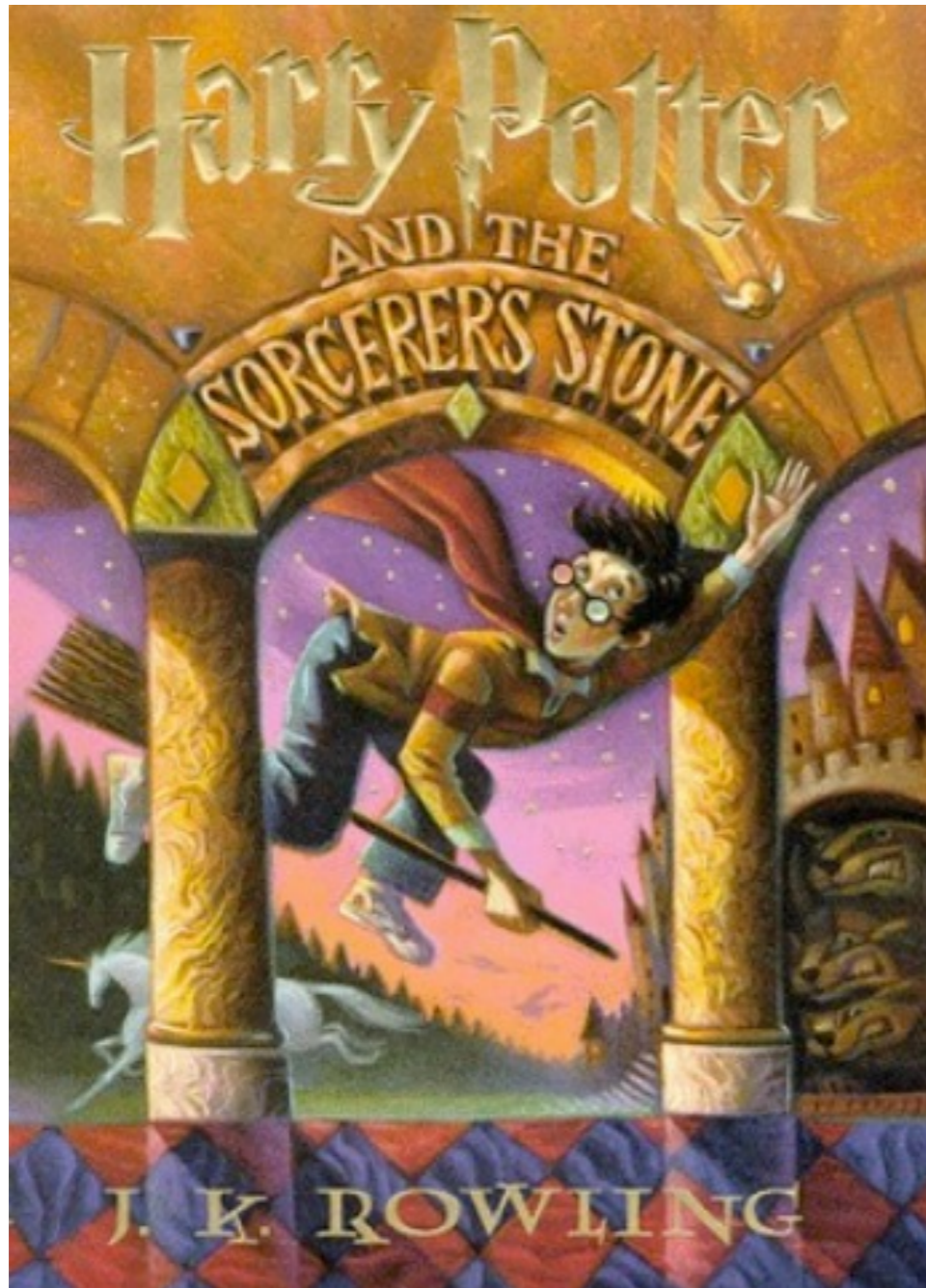


# Parallel Corpus





# Parallel Corpus



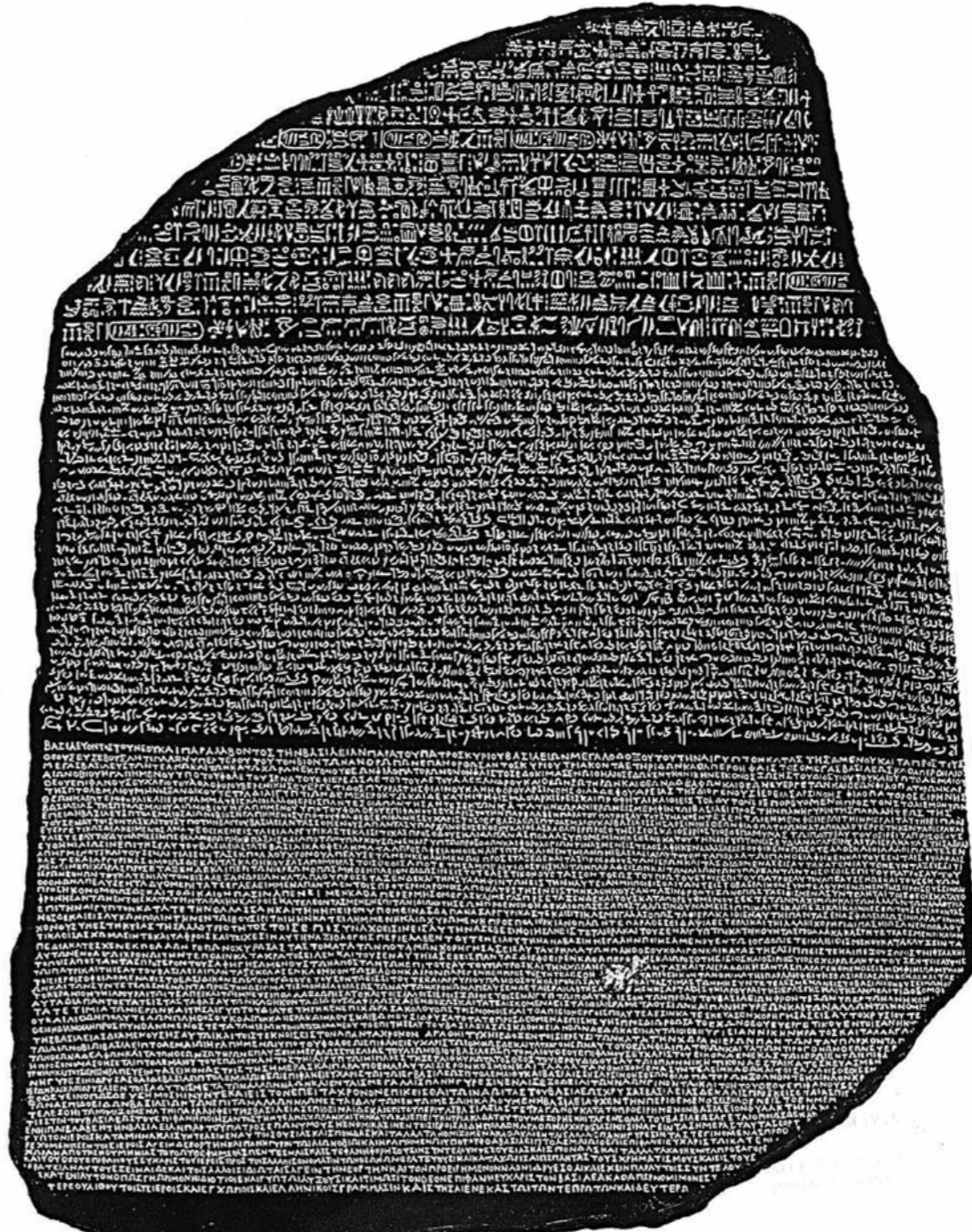


# Parallel Corpus

Egyptian



Greek



# WMT

- Annual conference for Machine Translation (2006-now)
- Many shared tasks:
  - **Translation** tasks: News, Biomedical articles, Translate similar languages, low-resource MT, large-scale multilingual MT, triangular MT, efficiency, terminology, unsupervised MT, lifelong learning
  - **Evaluation** tasks: quality estimation, metrics
  - **Other** tasks: automatic post-editing

# OPUS Parallel Corpus

- OPUS (Tiedemann 2012) is a growing collection of translated texts from the web.
- Preprocessed parallel texts in tmx, mooses format



Search & download resources:     show all versions

Language resources: click on [ tmx | mooses | xces | lang-id ] to download the data! (raw = untokenized, ud = parsed with universal dependencies, alg = word alignments and phrase tables)

corpus	doc's	sent's	de tokens	en tokens	XCES/XML	raw	TMX	Moses	mono	raw	ud	alg	dic	freq	other files		
CCMatrix v1	1	247.5M	3.8G	3.9G	xces de en	de en	tmx	mooses	de en	de en				de en	sample		
WikiMatrix v1	1	6.2M	443.1M	1.0G	xces de en	de en	tmx	mooses	de en	de en				de en	sample		
ParaCrawl v8	364	36.3M	450.7M	478.7M	xces de en	de en	tmx	mooses	de en	de en				de en			
EUbookshop v2	15373	9.6M	337.4M	380.2M	xces de en	de en	tmx	mooses	de en	de en		alg	dic	de en	query	sample	mooses/strict
EuroPat v3	1	12.6M	318.2M	387.8M	xces de en	de en	tmx	mooses	de en	de en				de en	sample		
wikimedia v20210402	1	0.1M	11.0M	349.2M	xces de en	de en	tmx	mooses	de en	de en				de en	sample		
CCAligned v1	1852	15.3M	150.8M	159.5M	xces de en	de en	tmx	mooses	de en	de en				de en	sample		
TildeMODEL v2018	7	4.3M	108.8M	131.4M	xces de en	de en	tmx	mooses	de en	de en		alg smt	dic	de en	sample		
DGT v2019	38675	3.6M	66.0M	73.3M	xces de en	de en	tmx	mooses	de en	de en		alg smt	dic	de en	sample		

<https://opus.nlpl.eu/>

# Noisy Channel MT (Statistic Machine Translation)



# $f$ -to- $e$ Translation

- We want a model of  $p(e | f)$

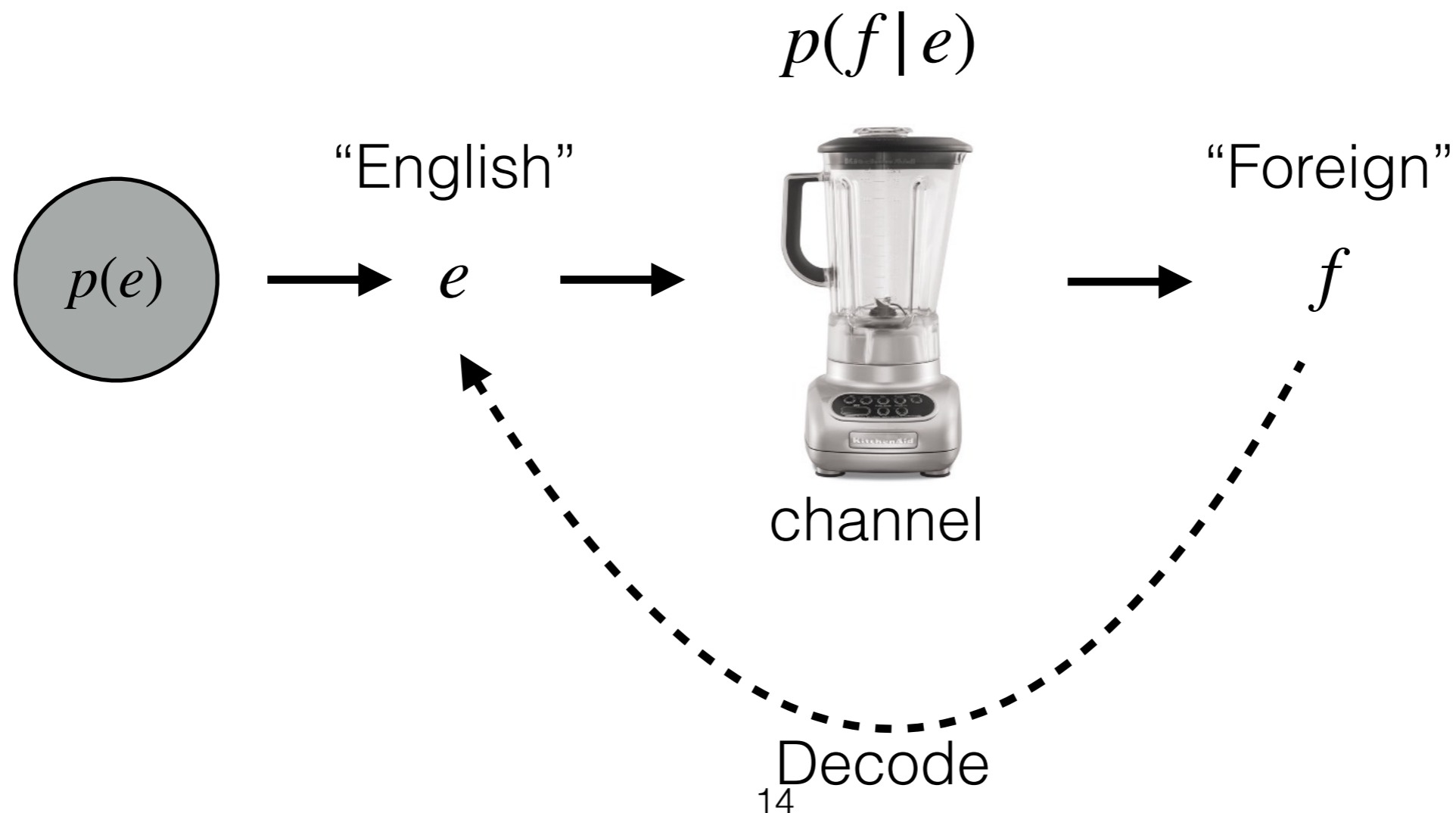
Possible English sentence

Confusing foreign sentence



# Noisy Channel MT

- **Speaker:** Have an English sentence in mind, encrypt it through a noisy channel, and speak the sentence in a foreign language
- **Listener:** Decode what they hear to the original English sentence.



# Noisy Channel MT

(Forward) Translation Model

$$\begin{aligned}\hat{e} &= \arg \max_e \underline{p(e|f)} \\ &= \arg \max_e \frac{p(e) \times p(f|e)}{p(f)} \\ &= \arg \max_e \underline{p(e)} \times \underline{p(f|e)}\end{aligned}$$

Language Model

(Backward) Translation Model  
i.e., Noisy Channel

What's the benefit of the Noisy Channel decomposition instead of modeling the forward translation directly?

# Noisy Channel Division of Labor

- Language model  $p(e)$ 
  - Is the translation fluent, grammatical, and idiomatic?
  - Use any LMs trained on large datasets
- Translation model  $p(f|e)$ 
  - (Backward) translation probability
  - Ensures adequacy of translation



# Training Noisy Channel MT

- Training LMs is simple (refer to the LM lecture)
- Estimating  $p(f|e)$  is a bit harder
  - $f =$  ie voudrais un peu de fromage  $p(f|e)$
  - $e_1 =$  I would like some cheese 0.4
  - $e_2 =$  I would like a little of cheese 0.5
  - $e_3 =$  There is no train to Barcelona  $>0.00001$

# Estimate Channel Translation Model

- How do we parameterize  $p(f | e)$ ?

$$p(f | e) = \frac{\text{count}(f, e)}{\text{count}(e)} \quad ?$$

- There are a lot of possible sentences
  - We can only count the sentences in our training data
  - This won't generalize to new inputs
- Can we break the sentence probability into lexical (word-level) translation probability?

# Lexical Translation

- How do we translate a word? Look it up in a dictionary!
  - e.g., Haus (German): house, home, shell, household

Translation	Count
house	5000
home	2000
shell	100
household	80

Maximum Likelihood Estimation (MLE)

$$\hat{p}_{\text{MLE}}(e \mid \text{Haus}) = \begin{cases} 0.696 & \text{if } e = \text{house} \\ 0.279 & \text{if } e = \text{home} \\ 0.014 & \text{if } e = \text{shell} \\ 0.011 & \text{if } e = \text{household} \\ 0 & \text{otherwise} \end{cases}$$

# Lexical Translation

- Goal: a model  $p(\mathbf{f} | \mathbf{e}, m)$ , where  $\mathbf{e} = \langle e_1, e_2, \dots, e_l \rangle$  and  $\mathbf{f} = \langle f_1, f_2, \dots, f_m \rangle$ , assuming that there is some distribution  $p(m | l)$  that models  $\mathbf{f}$ 's length conditioned on  $\mathbf{e}$ 's length.
- Lexical translation makes the following assumptions:
  1. Each word  $f_i$  is generated from exactly one word in  $\mathbf{e}$
  2. Thus, we have a latent alignment  $a_i$  that indicates which English word  $e_{a_i}$  generates  $f_i$ .
  3. Given the alignments  $\mathbf{a}$ , translation decisions are conditionally independent of each other and depend only on the aligned English word  $e_{a_i}$



# Lexical Translation

- Putting our assumptions together, we have:

$$p(\mathbf{f} | \mathbf{e}, m) = \underbrace{\sum_{\mathbf{a} \in [0, l]^m} p(\mathbf{a} | \mathbf{e}, m)}_{p(\text{Alignment})} \times \underbrace{\prod_{i=1}^m p(f_i | e_{a_i})}_{p(\text{Translation} | \text{Alignment})}$$

where  $\mathbf{a}$  is an  $m$ -dimensional latent vector with each element  $a_i$  in the range of  $[0, l]$

# Word Alignment

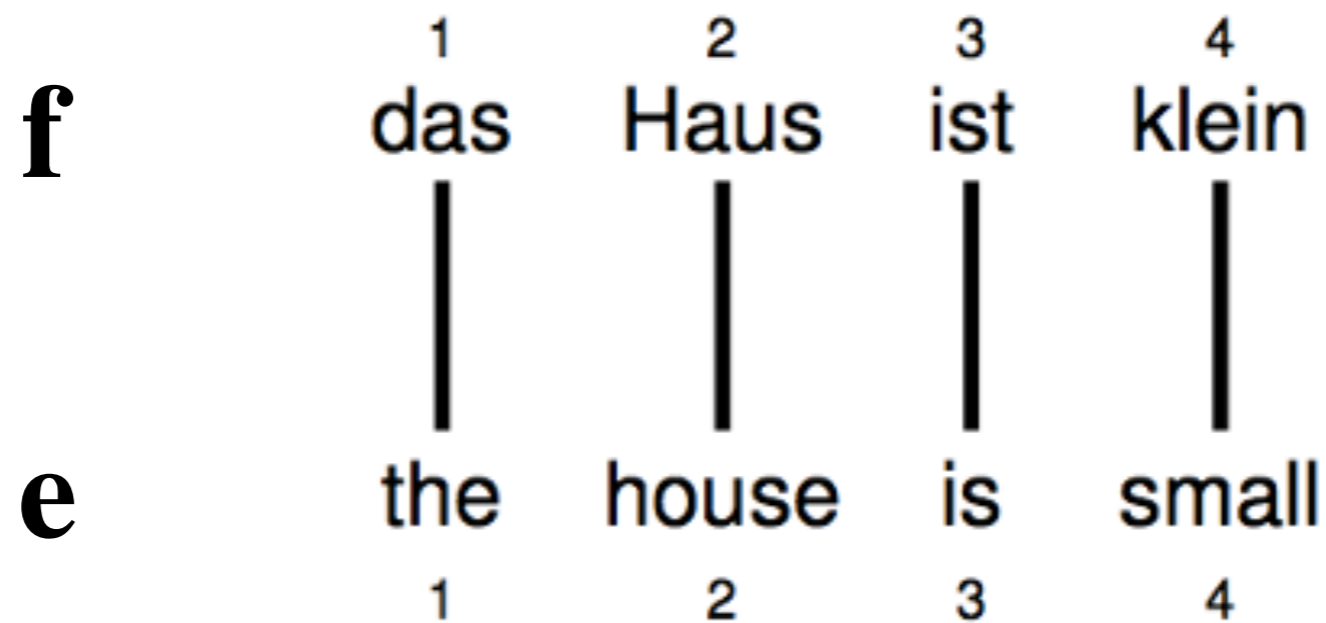
- Most of the research for the first 10 years of SMT was focusing on improving word alignment. Word translations weren't hard (with MLE), but predicting word order was hard.
- E.g. IBM Model 1, 2, 3, Giza++, FastAlign

$$p(\mathbf{a} | \mathbf{e}, m) = \prod_{i=1}^m p(a_i | i, l, m)$$

where  $|\mathbf{e}| = l$ ,  $|\mathbf{f}| = m$ ,  $f_i$  is aligned to  $e_{a_i}$ ,  $a_i \in [0, l]$

# Word Alignment

- Alignments can be visualized by drawing links between two sentences, and they are represented as vectors of positions:

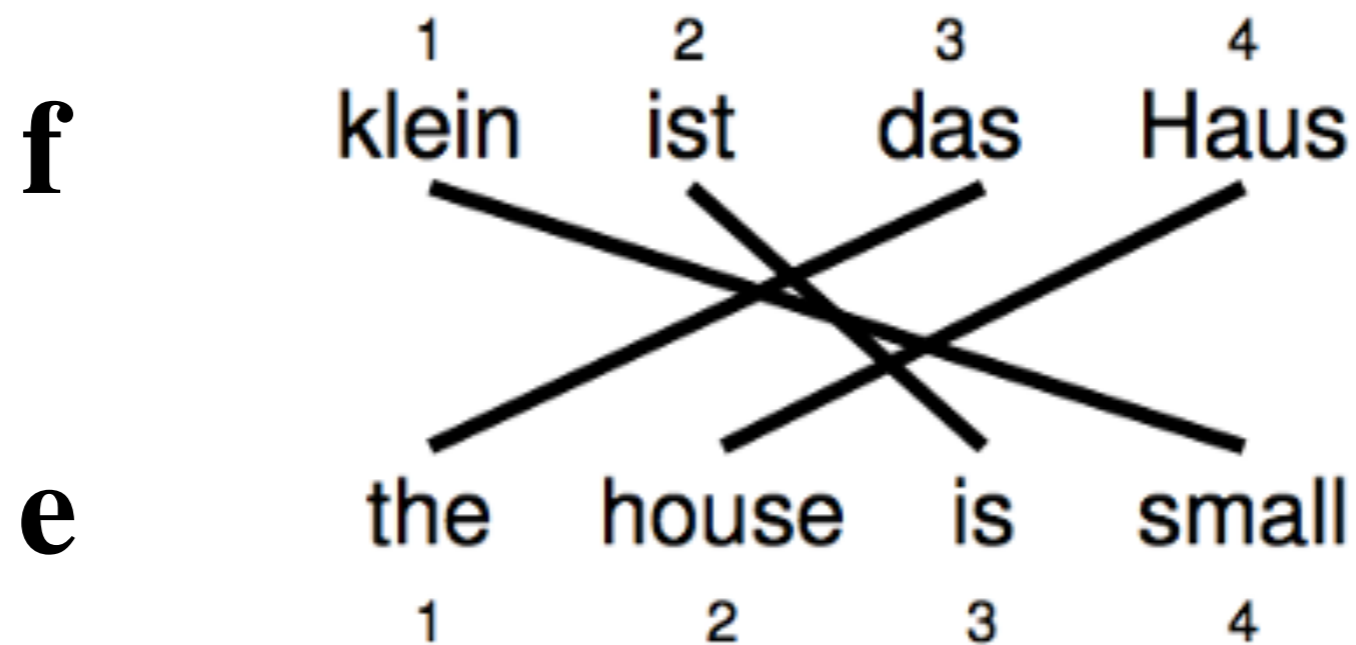


$$\mathbf{a} = (1, 2, 3, 4)^T$$

# Reordering

- Words may be reordered during translation

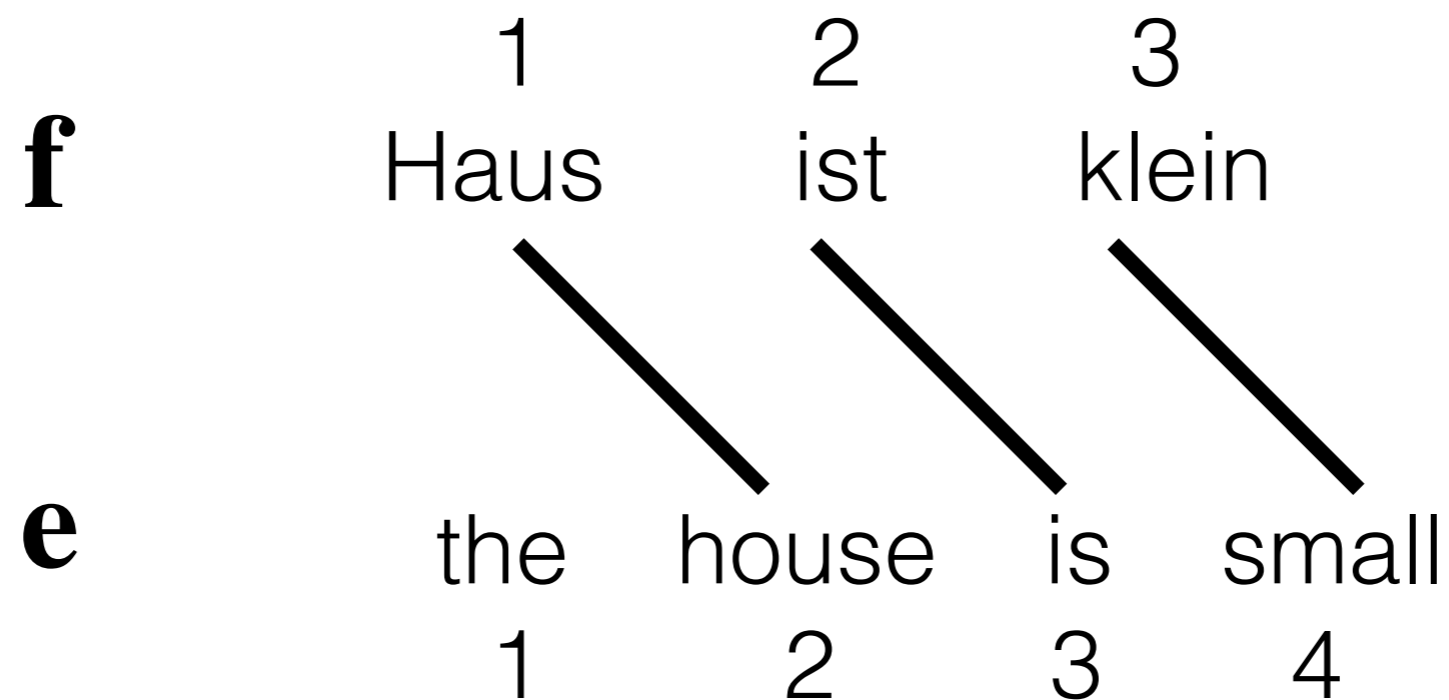
$$\mathbf{a} = (4,3,1,2)^T$$



# Word Dropping

- A source word may not be translated at all

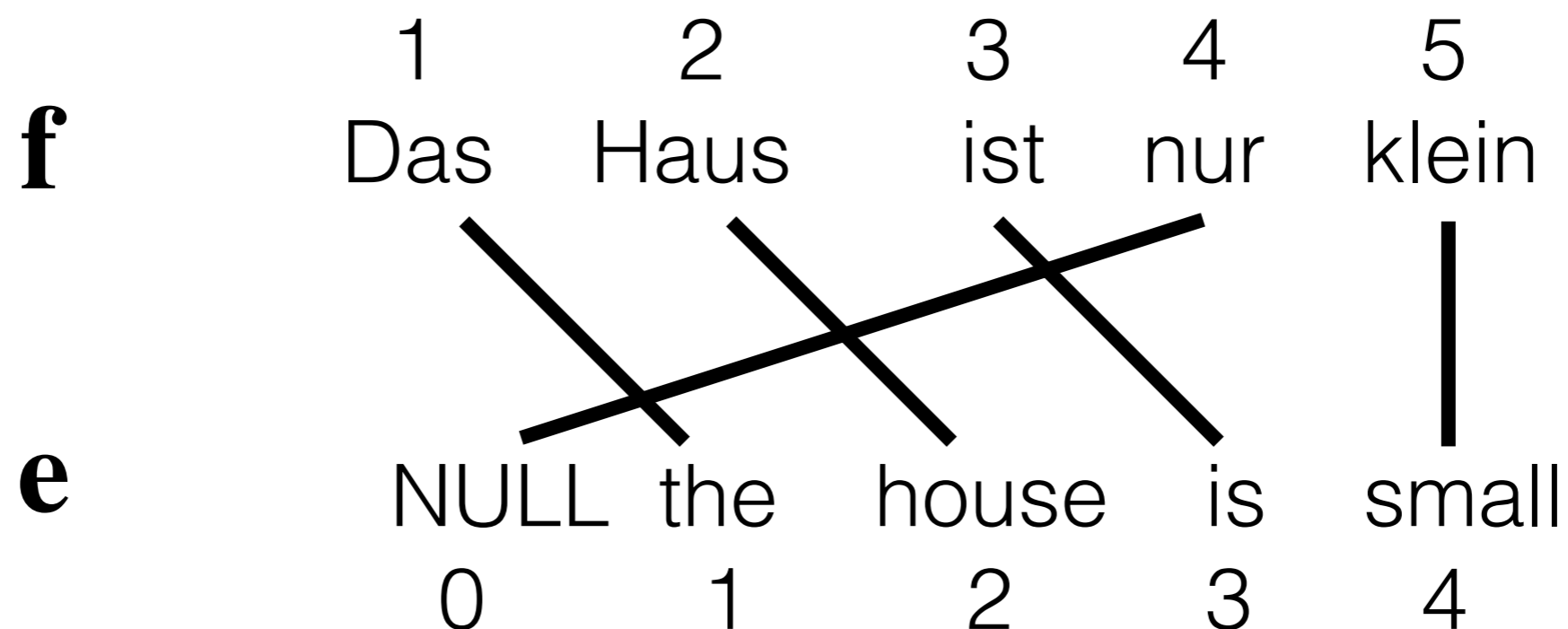
$$\mathbf{a} = (2,3,4)^T$$



# Word Insertion

- Words may be inserted during translation
- e.g., English just does not have an equivalent
- But these words must be explained—we typically assume every source sentence contains a NULL token

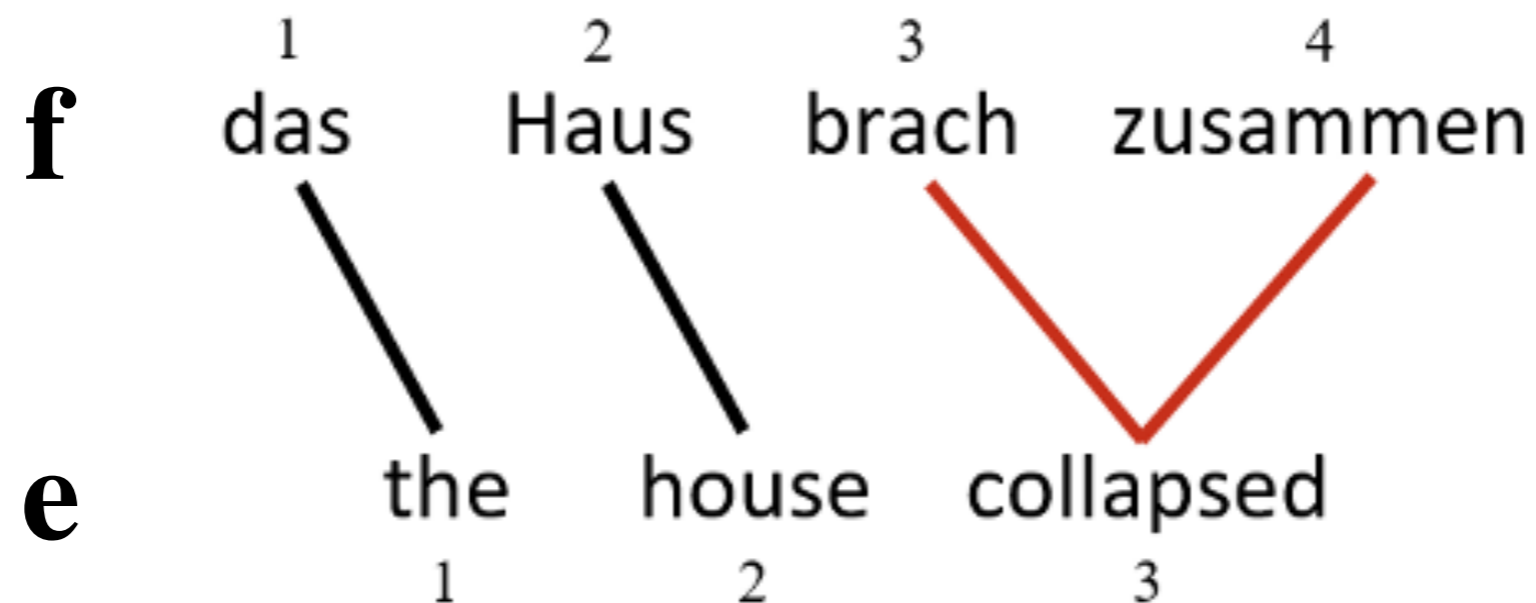
$$\mathbf{a} = (1, 2, 3, 0, 4)^T$$



# One-to-many Translation

- A source word may be translated into more than one target word

$$\mathbf{a} = (1, 2, 3, 3)^T$$

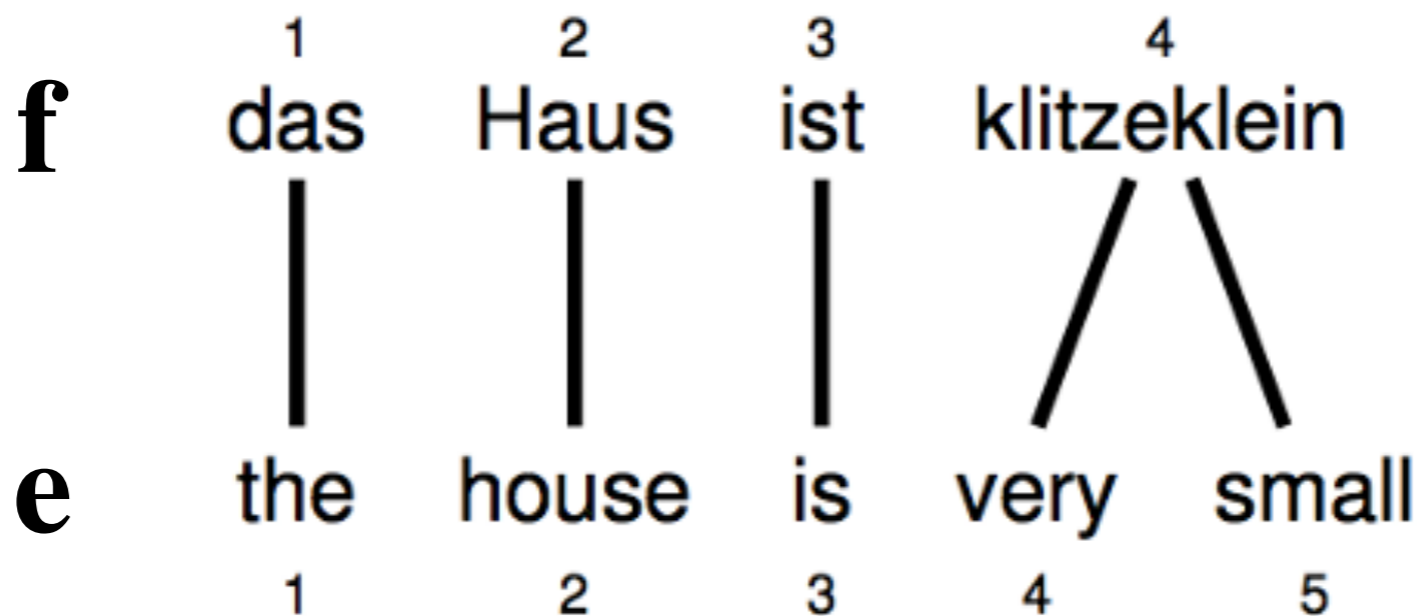




# Many-to-one Translation

- More than one source word may not be translated as a unit in lexical translation

$$\mathbf{a} = ??? \quad \mathbf{a} = (1, 2, 3, (4, 5)^T)^T ?$$



This could be addressed by considering phrase-level alignment instead of word level.

# Learn alignment & translation together

- How do we learn from training corpus of  $(\mathbf{f}, \mathbf{e})$  pairs?

$$\begin{aligned} p(\mathbf{f} | \mathbf{e}, m) &= \sum_{\mathbf{a} \in [0, l]^m} p(\mathbf{a} | \mathbf{e}, m) \times \prod_{i=1}^m p(f_i | e_{a_i}) \\ &= \sum_{\mathbf{a} \in [0, l]^m} \underbrace{\prod_{i=1}^m p(a_i | i, l, m)}_{p(\text{Alignment})} \times \underbrace{p(f_i | e_{a_i})}_{p(\text{Translation} | \text{Alignment})} \end{aligned}$$

- MLE of two probability with the latent alignment

$$p(a_i | i, l, m) = \frac{\text{count}(a_i | i, l, m)}{\text{count}(i, l, m)} \quad p(f_i | e_{a_i}) = \frac{\text{count}(f_i, e_{a_i})}{\text{count}(e_{a_i})}$$

$\text{count}(a_i | i, l, m)$  is the no. time  $f_i$  is aligned to  $e_{a_i}$  in the training set.  $\text{count}(i, l, m)$  is the no. time we see a foreign sentence  $f$  of length  $m$  and an English sentence  $e$  of length  $l$

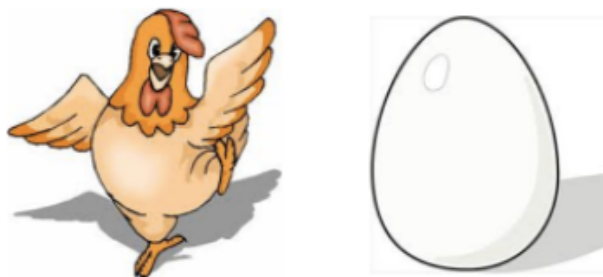
# Learn alignment & translation together

- How do we learn from training corpus of  $(\mathbf{f}, \mathbf{e})$  pairs?
- “Chicken and egg” problem:
  - If we had the alignments, we could estimate the translation probabilities by MLE (i.e., counting)

$$p(f_i | e_{a_i}) = \frac{\text{count}(f_i, e_{a_i})}{\text{count}(e_{a_i})}$$

- If we had the probabilities, we could find the most likely alignments greedily by taking the word pairs with the largest probability

$$a_i = \arg \max_{j \in [0, l]} p(a_i | i, l, m)$$



# Expectation-Maximization (EM) Algorithm

- Pick some random (or uniform) starting parameters (i.e., counts)
- Repeat until converged

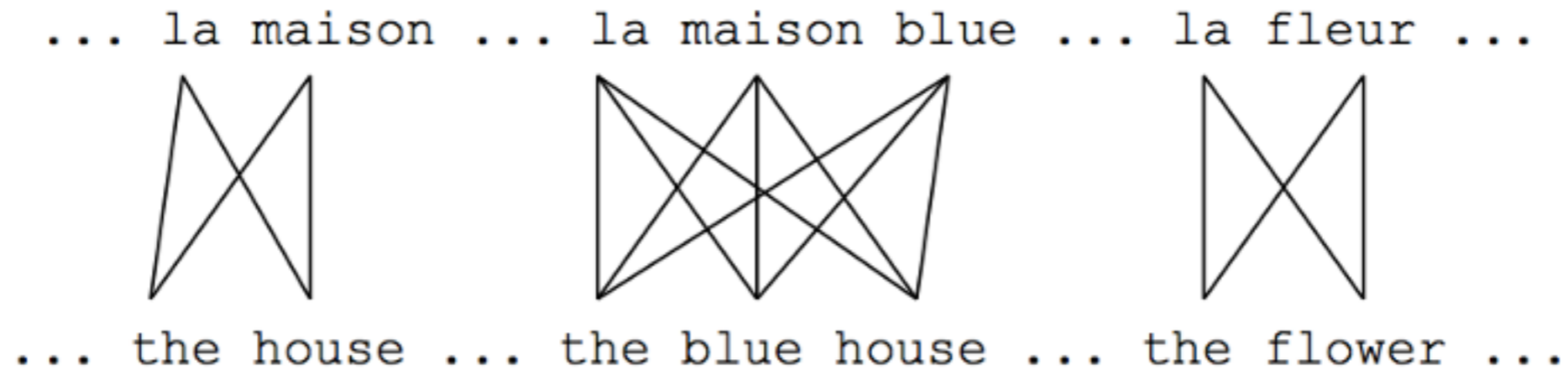
1. **E- Step:** use the current parameters to compute “expected” alignments

2. Update the no. of times  $e_{a_i}$  is translated to  $f_i$  i.e.,  $\text{count}(e_{a_i}, f_i)$ , and keep track of no. of times  $e_{a_i}$  is used in the training corpus  $\text{count}(e_{a_i})$ .

3. **M-Step:** use MLE to update translation probability

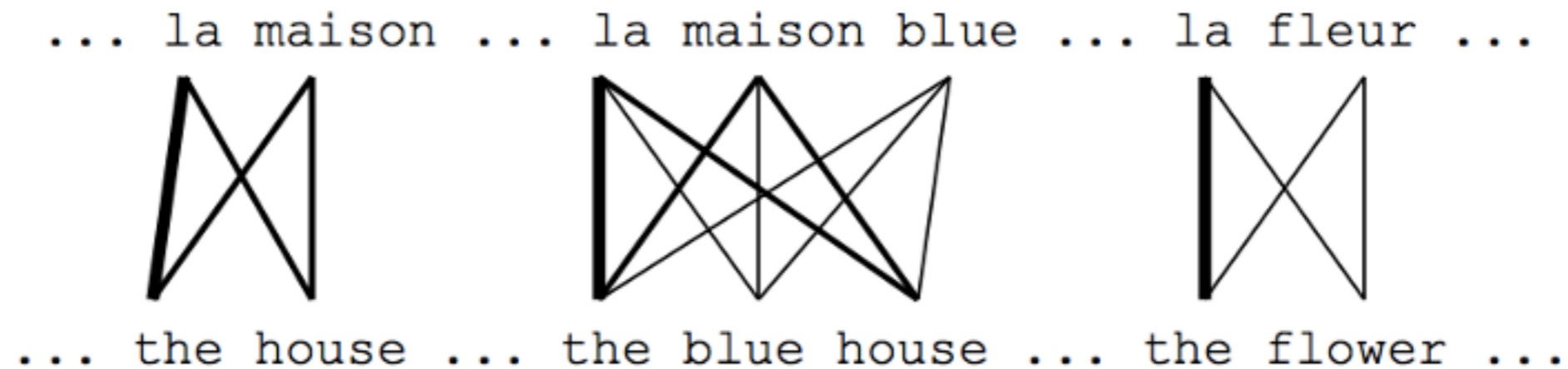
$$p(f_i | e_{a_i}) = \frac{\text{count}(f_i, e_{a_i})}{\text{count}(e_{a_i})}$$

# EM for IBM Model 1



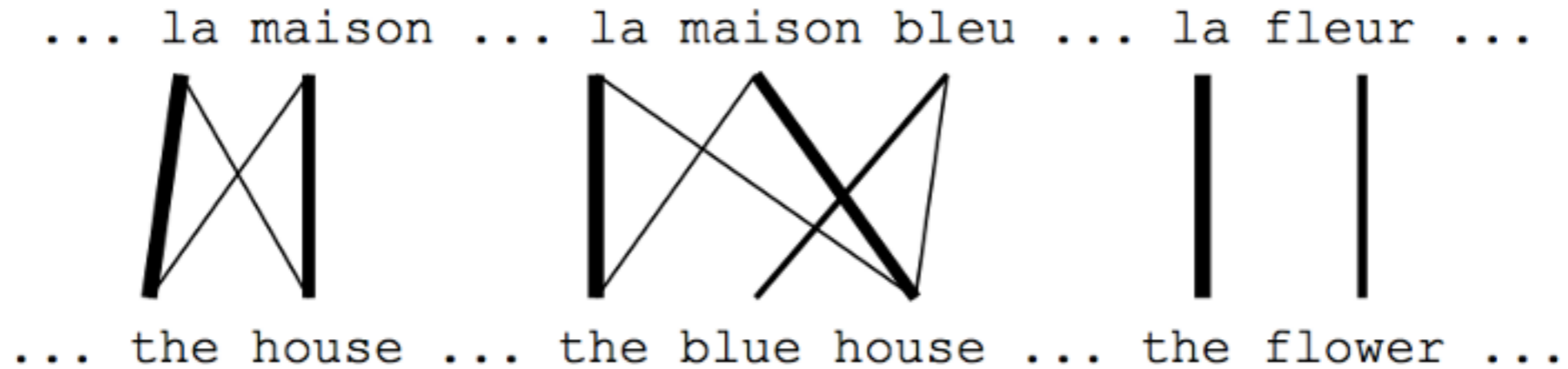
- Initial step: all alignments equally likely
- Model learns that, e.g., **la** is often aligned with **the**

# EM for IBM Model 1



- After one iteration
- Alignments, e.g., between **la** and **the** are more likely

# EM for IBM Model 1



- After another iteration
- It becomes apparent that alignments, e.g., between **fleur** and **flower** are more likely (pigeon hole principle)



# EM for IBM Model 1

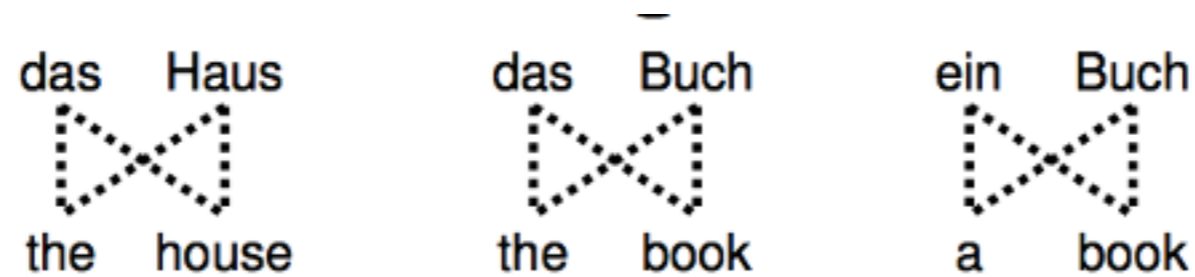
... la maison ... la maison bleu ... la fleur ...  
/ | | X | |  
... the house ... the blue house ... the flower ...



$p(\text{la}|\text{the}) = 0.453$   
 $p(\text{le}|\text{the}) = 0.334$   
 $p(\text{maison}|\text{house}) = 0.876$   
 $p(\text{bleu}|\text{blue}) = 0.563$   
...

- Parameter estimation from the aligned corpus

# Convergence



<i>e</i>	<i>f</i>	initial	1st it.	2nd it.	3rd it.	...	final
the	das	0.25	0.5	0.6364	0.7479	...	1
book	das	0.25	0.25	0.1818	0.1208	...	0
house	das	0.25	0.25	0.1818	0.1313	...	0
the	buch	0.25	0.25	0.1818	0.1208	...	0
book	buch	0.25	0.5	0.6364	0.7479	...	1
a	buch	0.25	0.25	0.1818	0.1313	...	0
book	ein	0.25	0.5	0.4286	0.3466	...	0
a	ein	0.25	0.5	0.5714	0.6534	...	1
the	haus	0.25	0.5	0.4286	0.3466	...	0
house	haus	0.25	0.5	0.5714	0.6534	...	1

# Whole Pipeline of SMT

- Moses (Koehn 2009)

1 Prepare data

2 Run GIZA

3 Align words

4 Lexical translation

5 Extract phrases

6 Score phrases

7 Reordering model

8 Generation model

9 Configuration file

EM algorithms to align  
& translate words

# Extensions: Lexical to Phrase Translation

- Phrase-based MT:
  - Allow multiple words to translate as chunks (including many-to-one)
  - Introduce **another latent variable**, the **source segmentation**



Adapted from Koehn (2006)

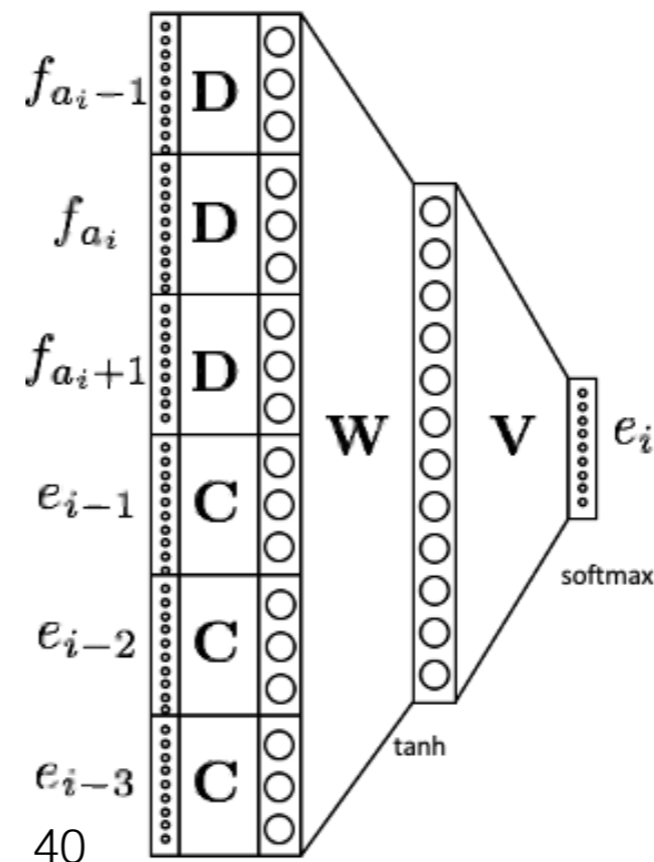
# Neural Machine Translation

# Neural Features for Translation

- Inspired by Neural n-gram LMs, use a conditional model to generate the next English word conditioned on
  - The previous  $n$  English words that have been generated
  - The aligned source word and its  $m$  neighbors

$$p(\mathbf{e} \mid \mathbf{f}, \mathbf{a}) = \prod_{i=1}^{|\mathbf{e}|} p(e_i \mid e_{i-2}, e_{i-1}, f_{a_i-1}, f_{a_i}, f_{a_i+1})$$

$$p(e_i \mid e_{i-2}, e_{i-1}, f_{a_i-1}, f_{a_i}, f_{a_i+1}) =$$



# Neural Features for Translation

- Word alignment is still needed.
- Improves over SMT

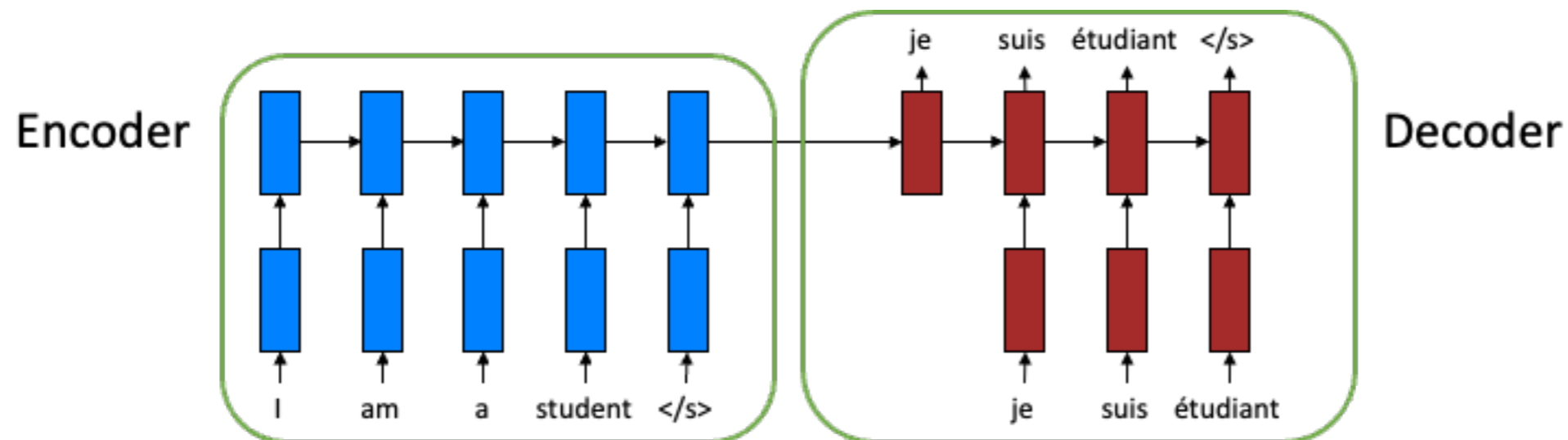
<b>BOLT Test</b>		
	<b>Ar-En</b>	
	<b>BLEU</b>	<b>% Gain</b>
“Simple Hier.” Baseline	33.8	-
S2T/L2R NNJM (Dec)	38.4	100%
Source Window=7	38.3	98%
Source Window=5	38.2	96%
Source Window=3	37.8	87%
Source Window=0	35.3	33%
Layers=384x768x768	38.5	102%
Layers=192x512	38.1	93%
Layers=128x128	37.1	72%
Vocab=64,000	38.5	102%
Vocab=16,000	38.1	93%
Vocab=8,000	37.3	83%
Activation=Rectified Lin.	38.5	102%
Activation=Linear	37.3	76%

Devlin et al. 2014



# Fully Neural Translation

- Fully end-to-end RNN-based MT model
- Encode the source sentence using one RNN
- Generate the target sentence one word at a time using another RNN

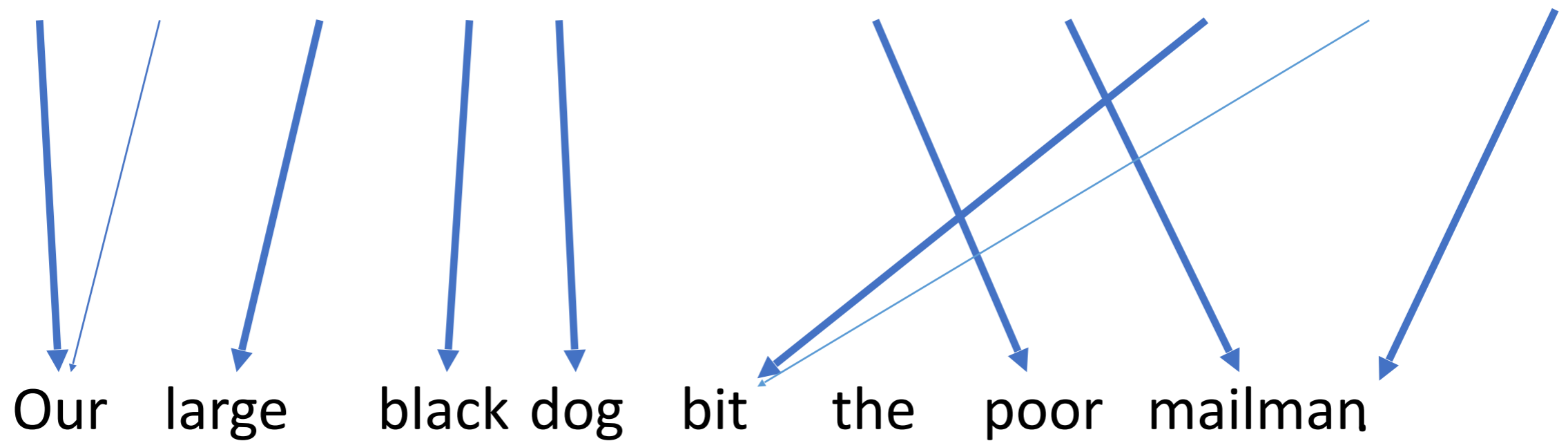


# Attention MT Models

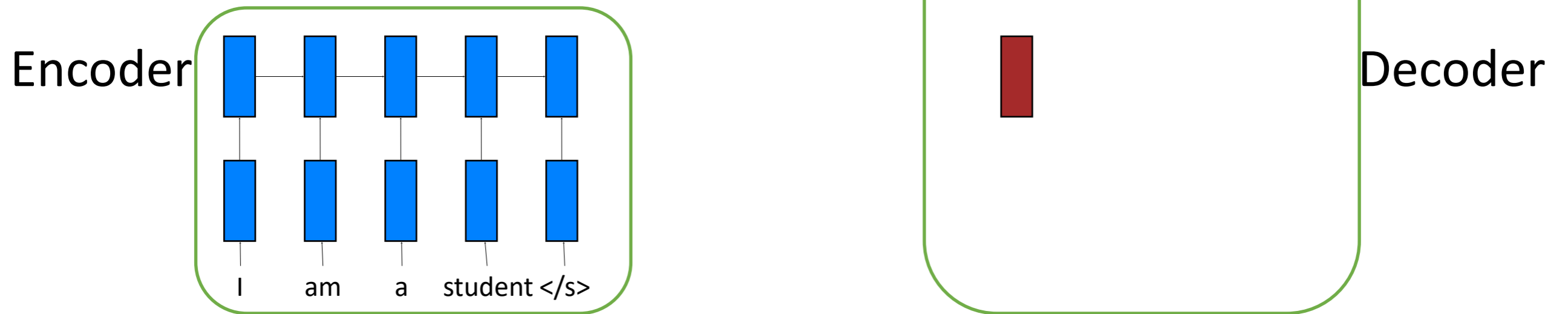
- The encoder-decoder model struggles with long sentences
- An RNN is trying to compress an arbitrarily long sentence into a finite-length word vector
- What if we only look at one (or a few) source words when we generate each output word?

# Intuition

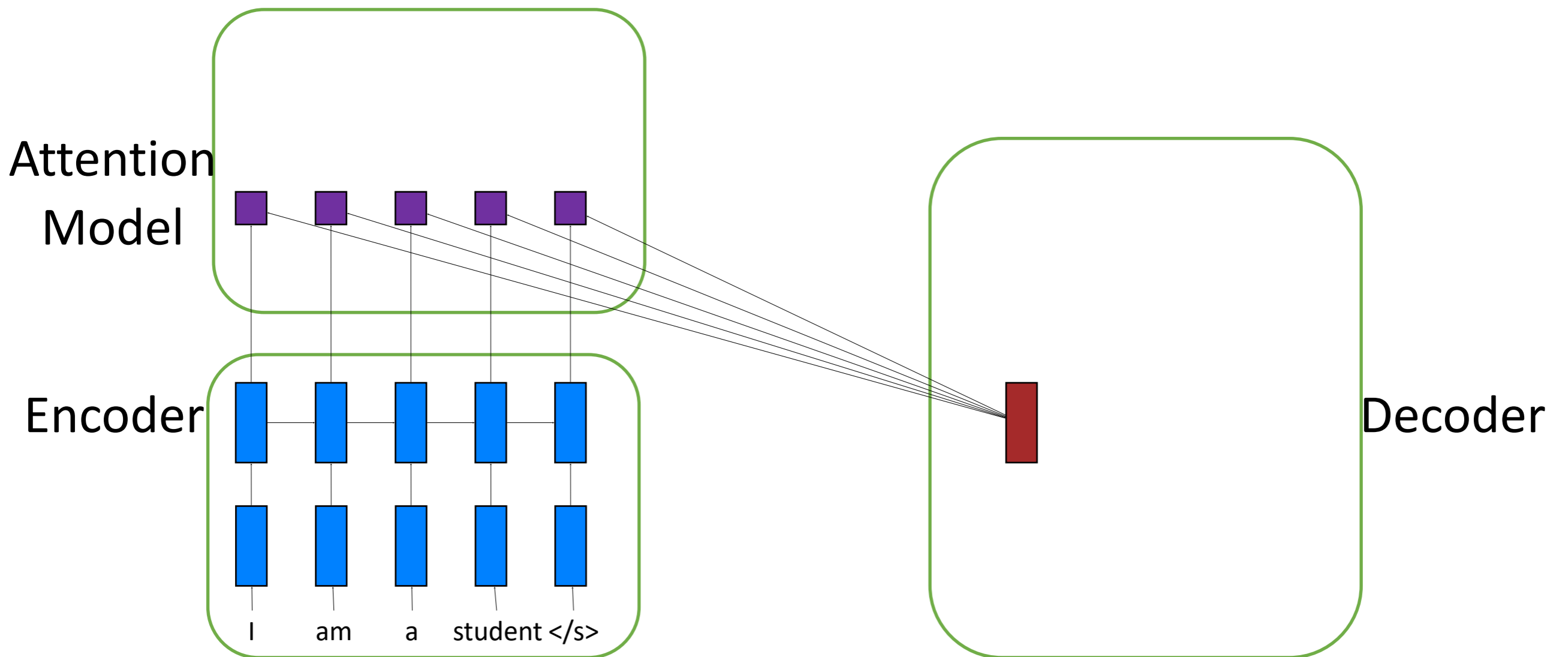
うち の 大きな黒い犬 が 可哀想な郵便屋に 噛み ついた。



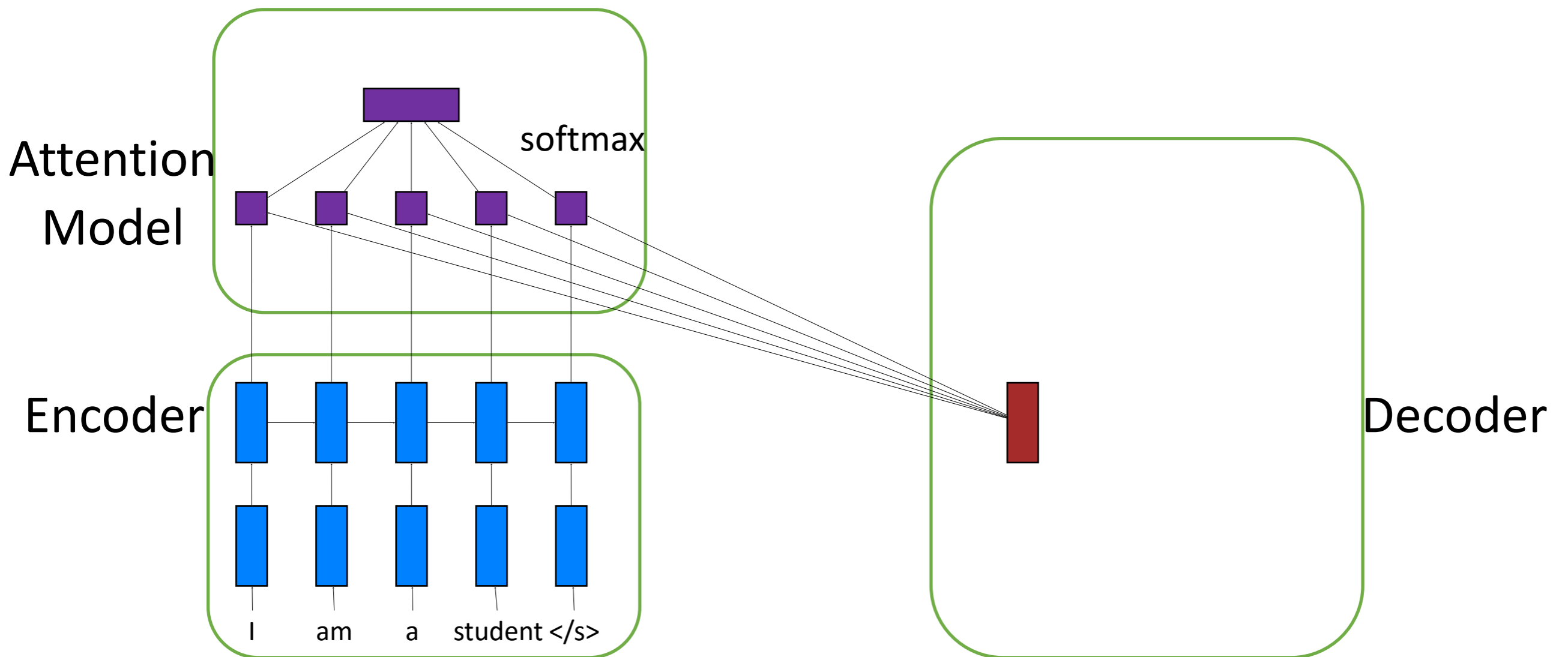
# Attention MT Models



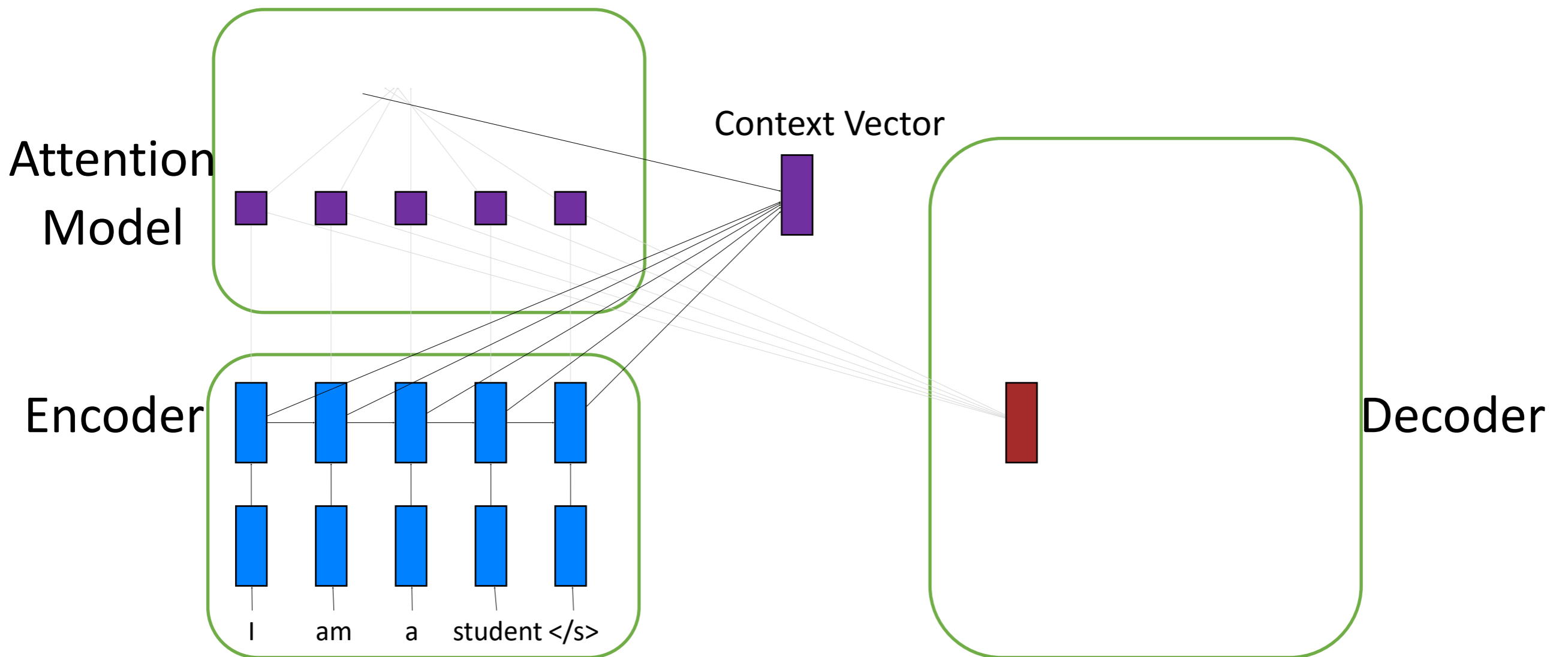
# Attention MT Models



# Attention MT Models

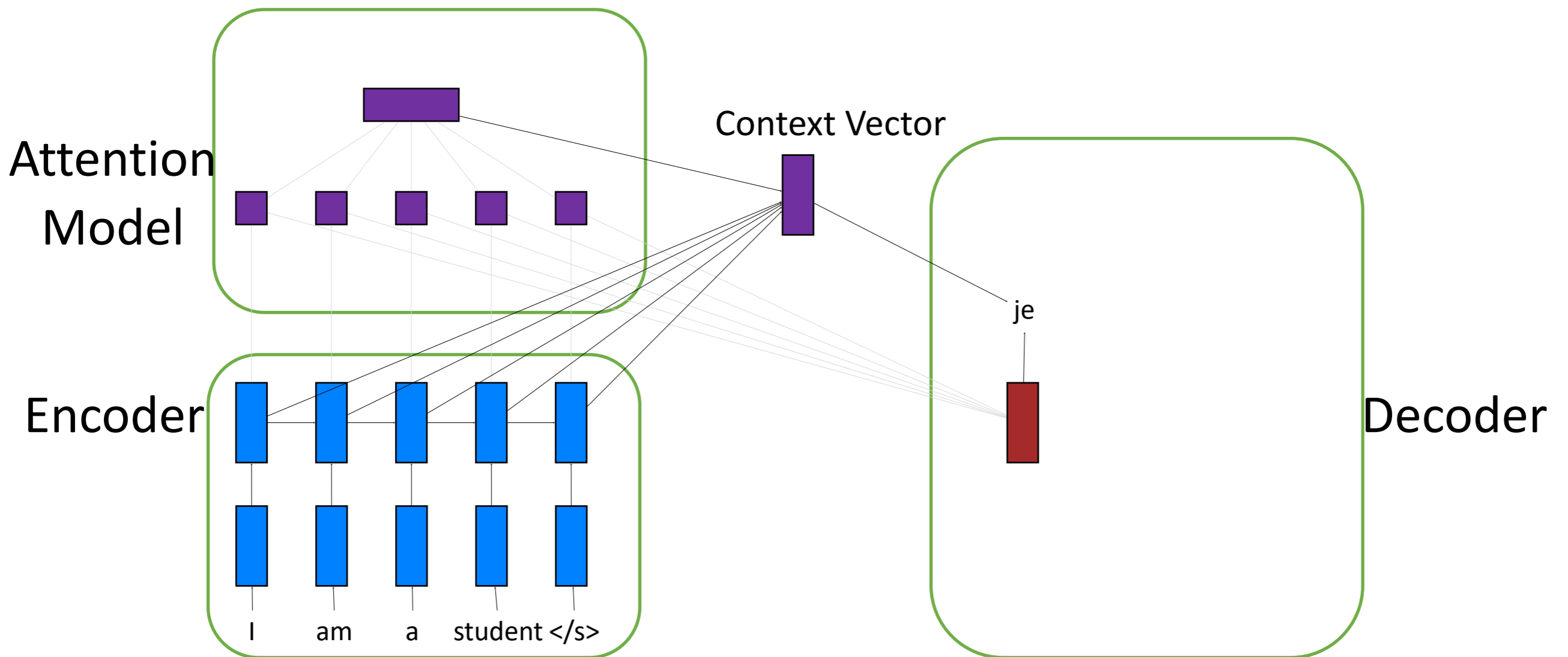


# Attention MT Models

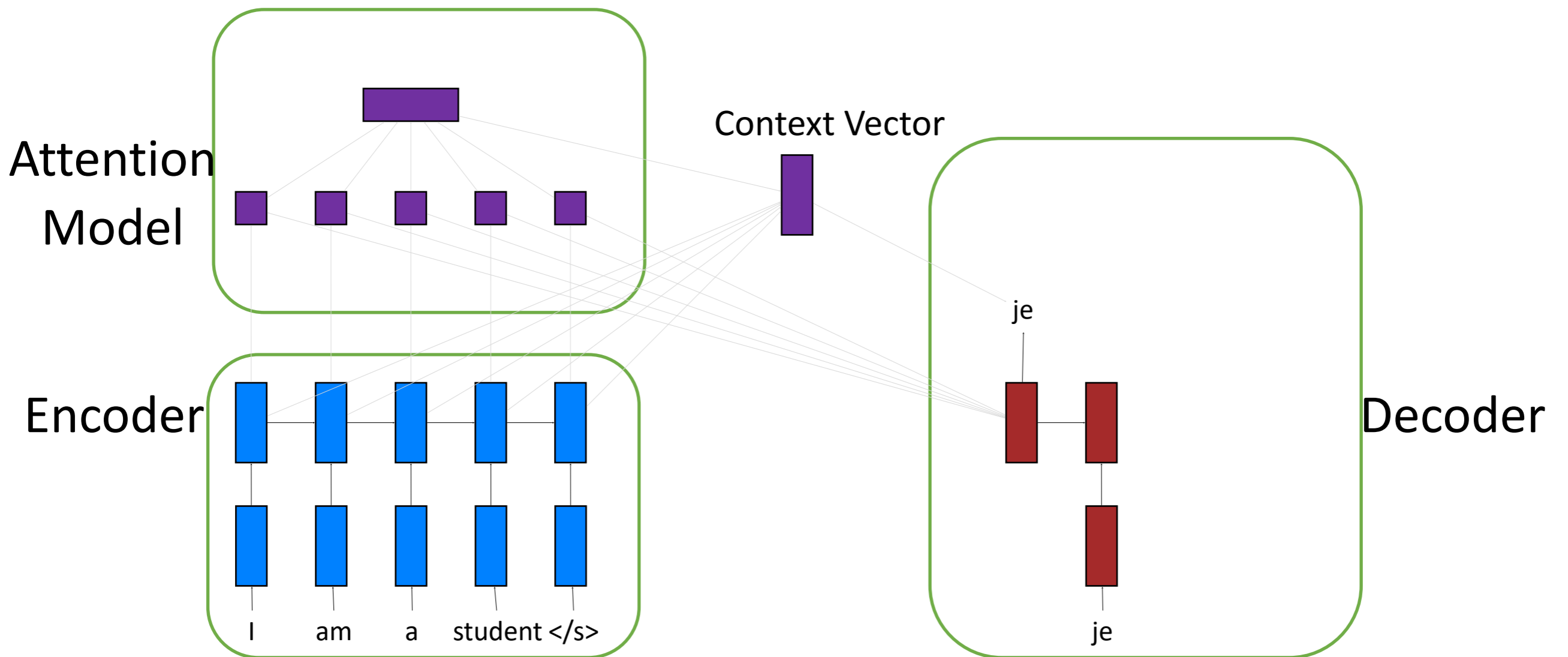




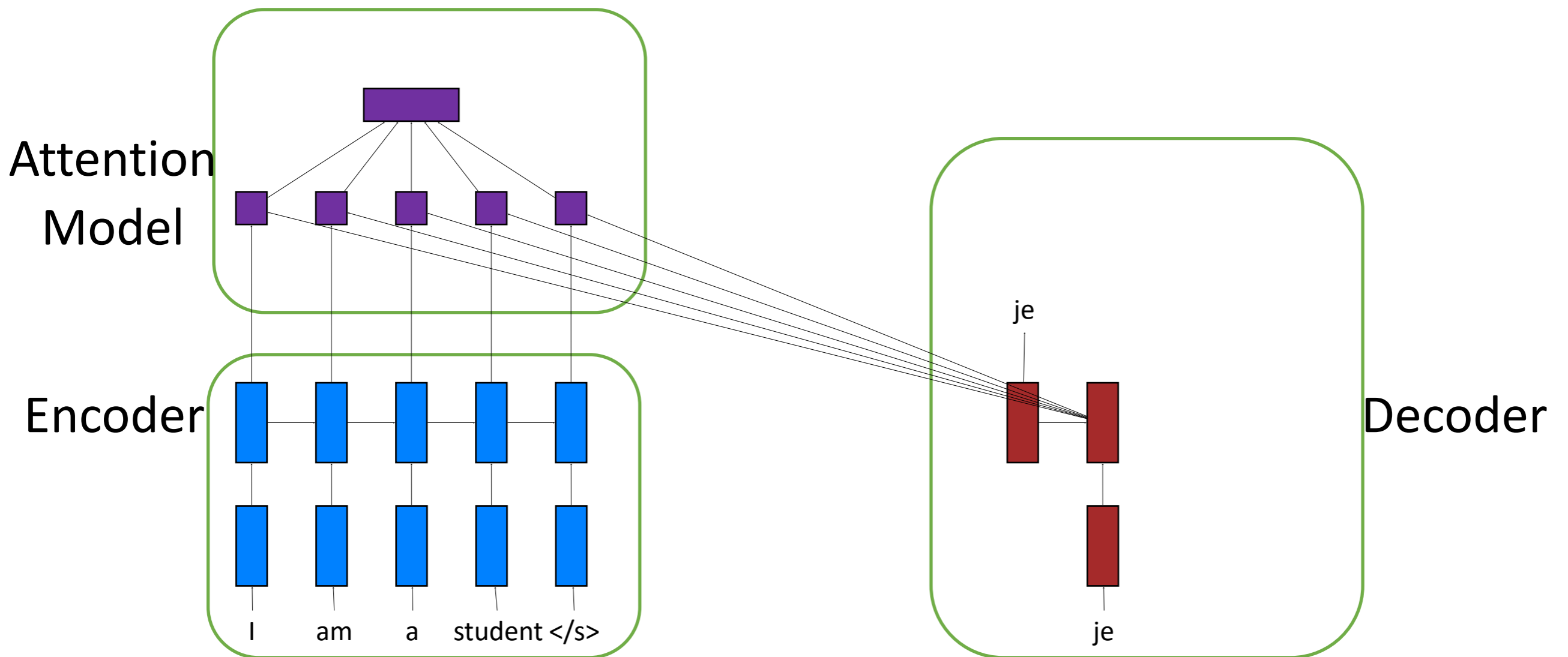
# Attention MT Models



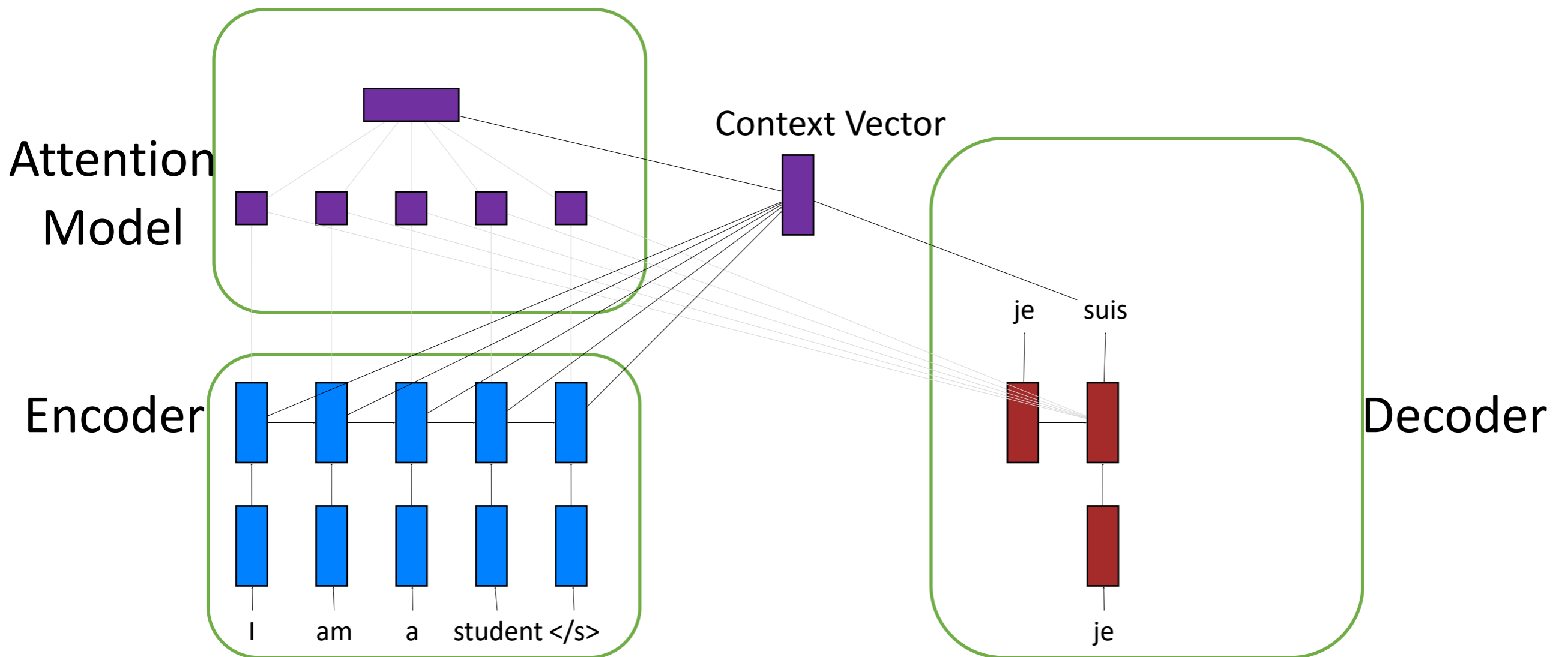
# Attention MT Models



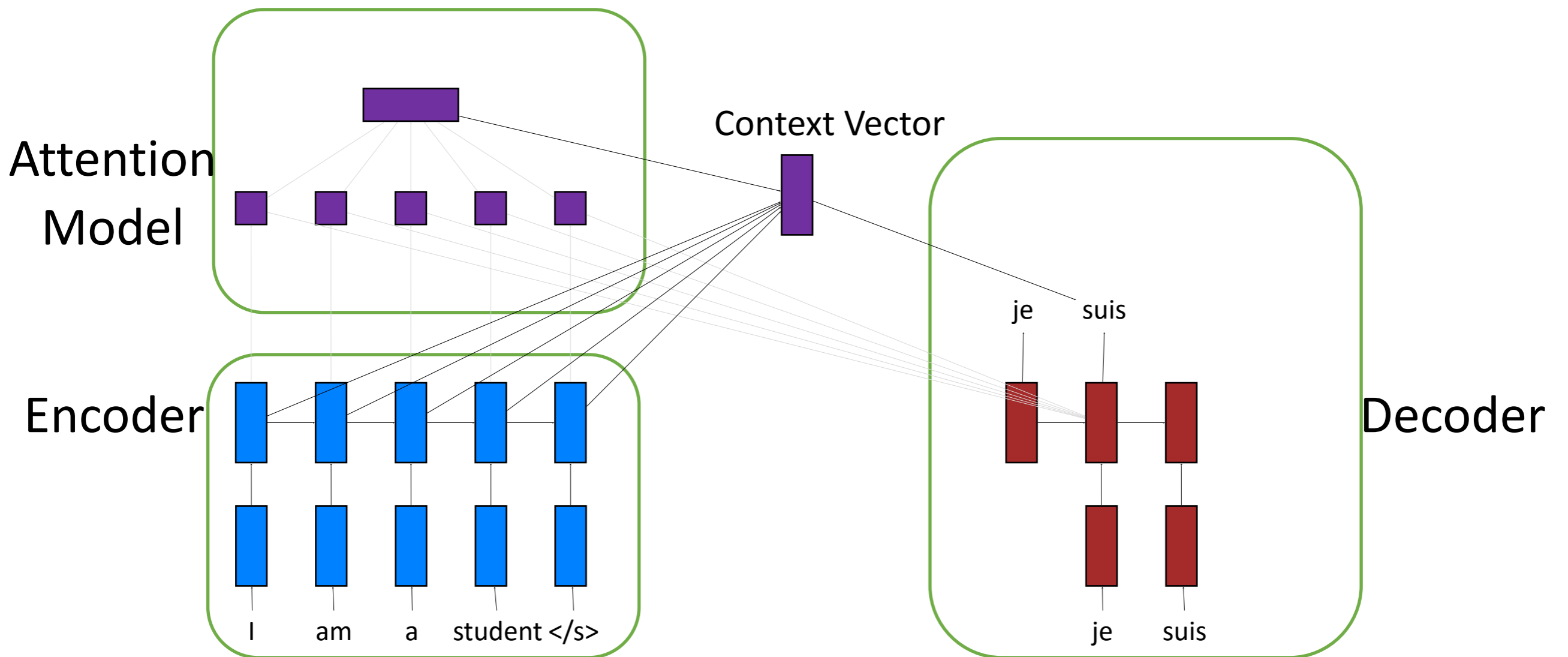
# Attention MT Models



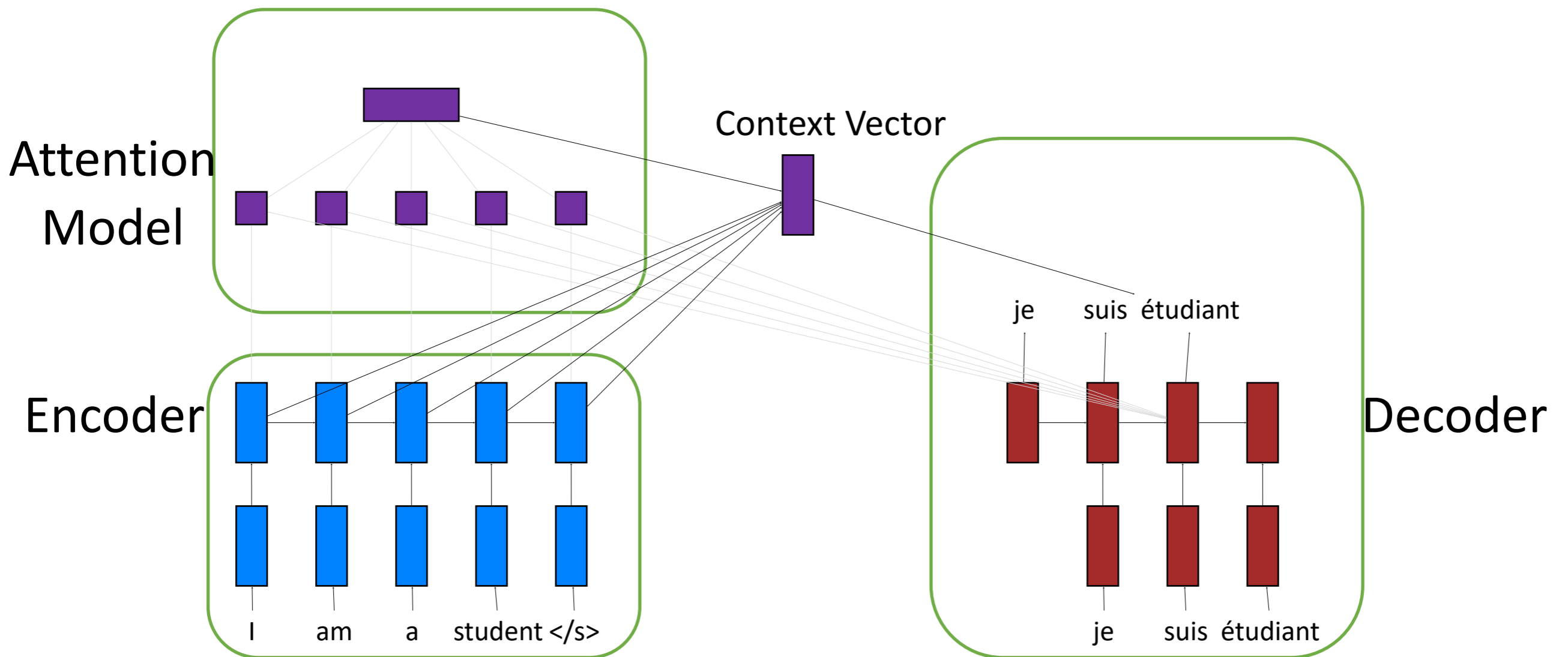
# Attention MT Models



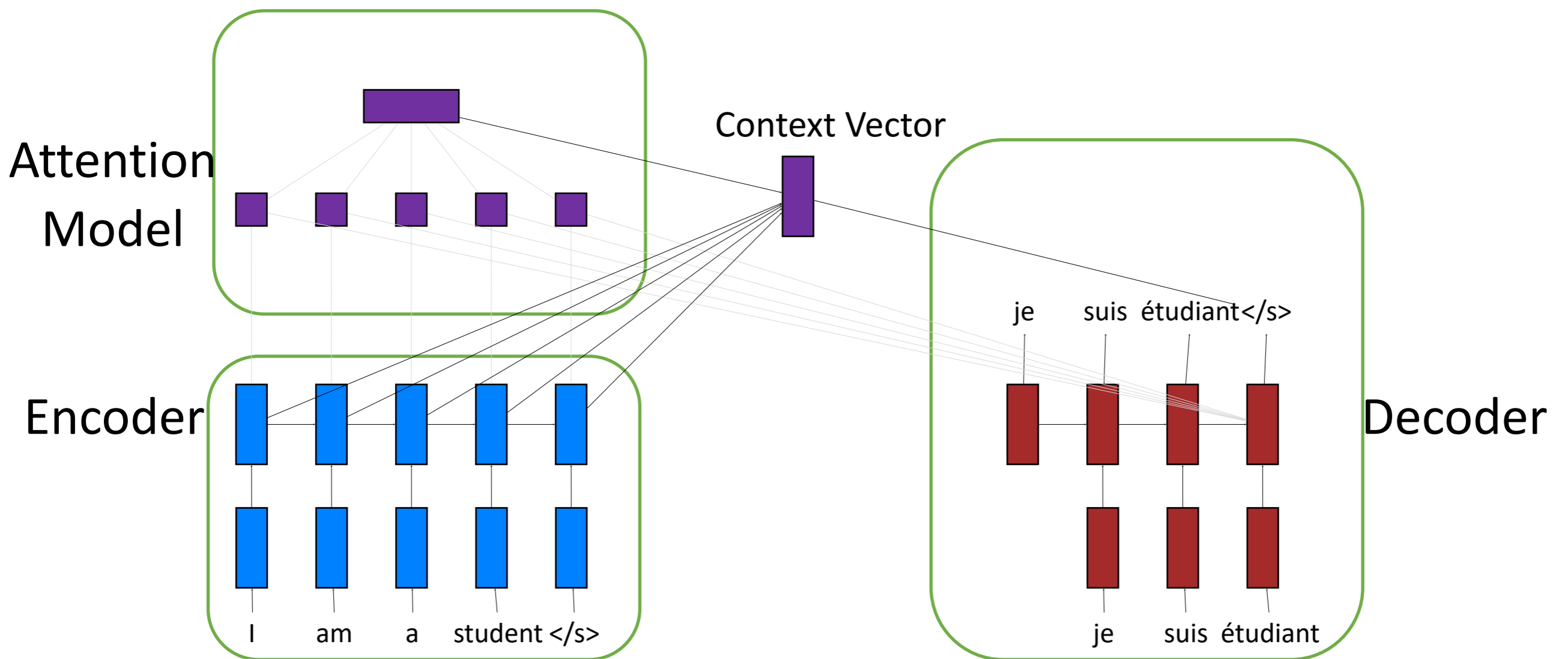
# Attention MT Models



# Attention MT Models

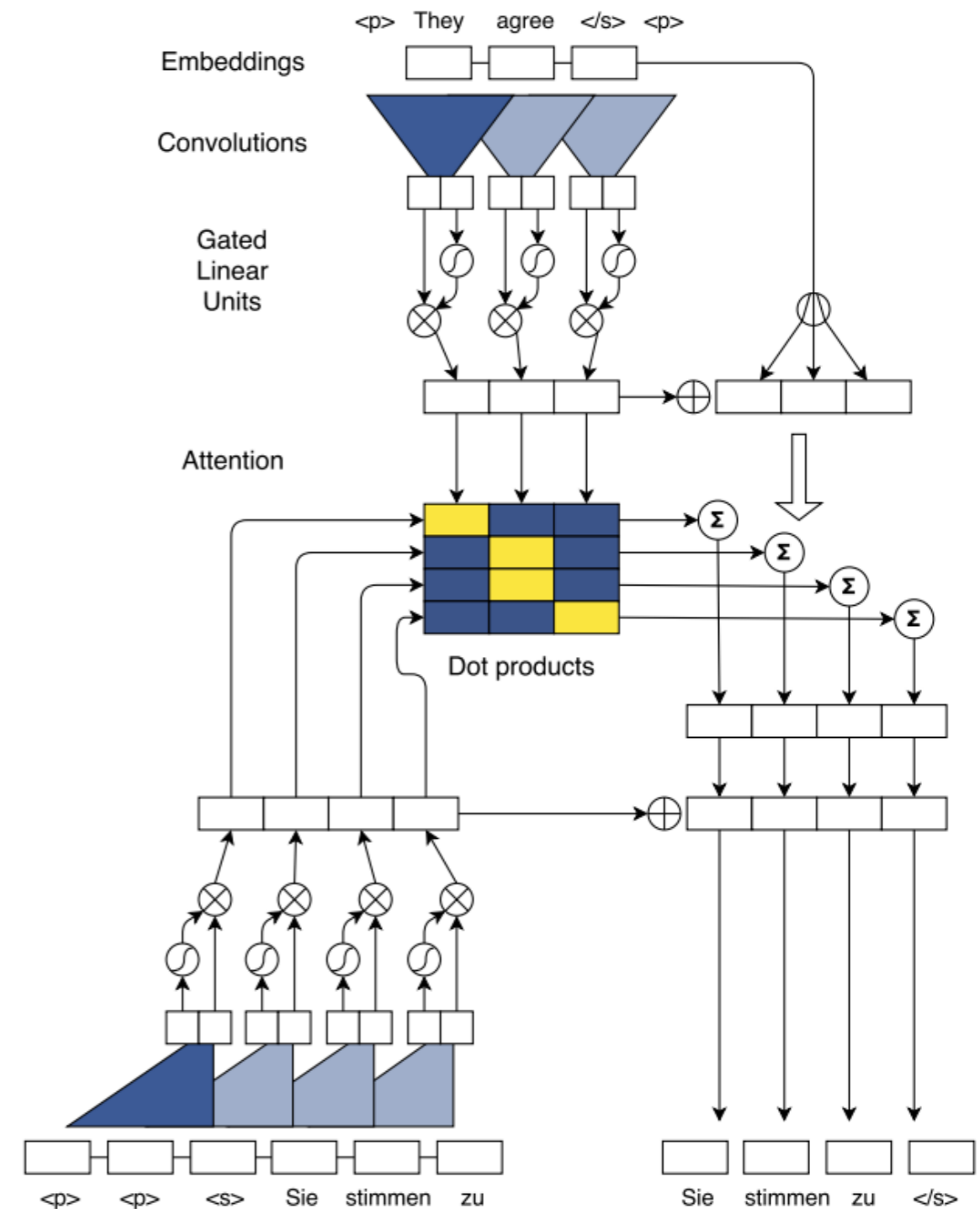


# Attention MT Models



# Convolutional Encoder-Decoder

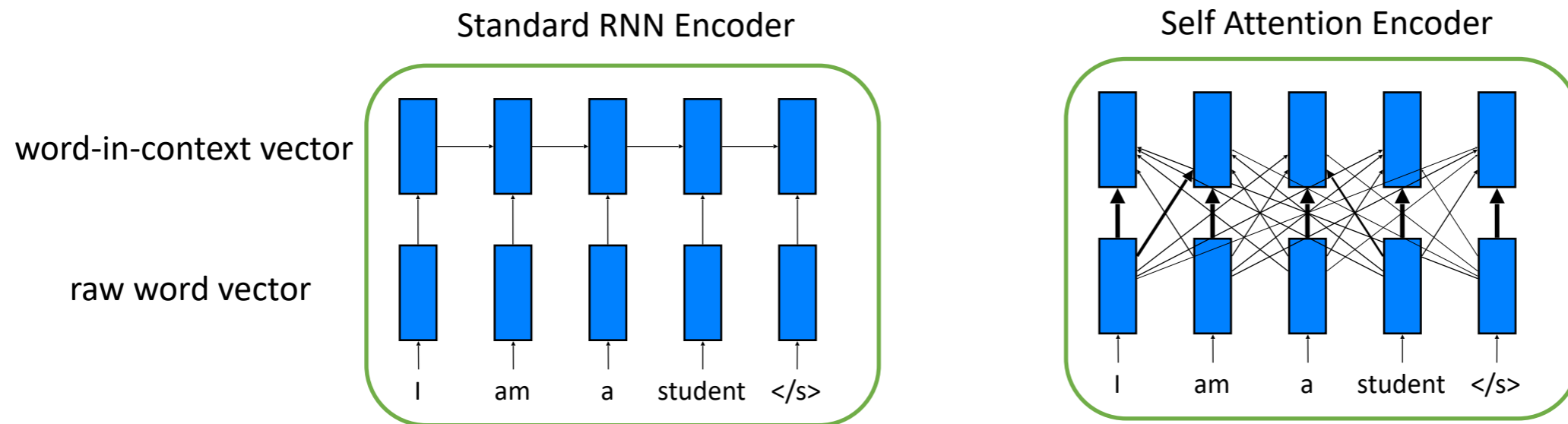
- CNN:
  - encodes words within a fixed size window
  - Parallel computation
  - Shortest path to cover a wider range of words
- RNN:
  - sequentially encode a sentence from left to right
  - Hard to parallelize





# Transformer

- Idea: Instead of using an RNN to encode the source sentence and the partial target sentence, use self-attention!



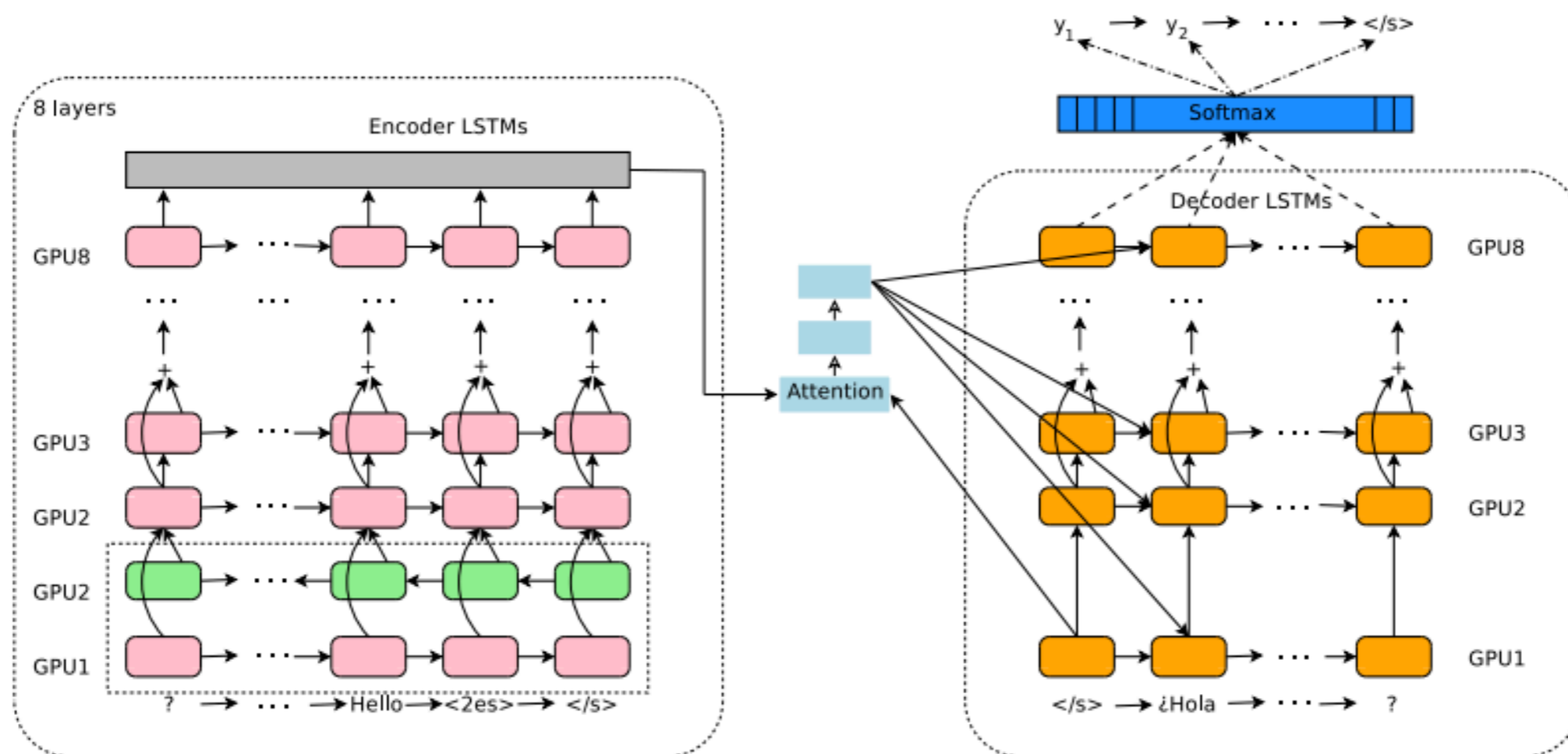
# Transformer

- Computation is easily parallelizable
- Shorter path from each target word to each source word -> stronger gradient signals
- Empirically stronger translation performance
- Empirically trains substantially faster than more serial models

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [17]	23.75			
Deep-Att + PosUnk [37]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [36]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [31]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [37]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [36]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	<b>41.29</b>	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	<b><math>3.3 \cdot 10^{18}</math></b>	
Transformer (big)	<b>28.4</b>	<b>41.0</b>	$2.3 \cdot 10^{19}$	

# Google's Multilingual NMT

- Stack 8-layers of LSTM encoder, and 8-layers of LSTM decoder
- Only use the last layer of encoder LSTM to perform target-to-source attention -> Re-use the context vector for each decoder layer
- Use the language code to indicate which target language to translate



Johnson et al. 2016

# Google's Multilingual NMT

- Add the target language code to the start of the source sentence, which enables sharing parameters for different language pairs (**many-to-one, one-to-many, zero-shot translation**)

Hello, how are you? -> Hola, ¿cómo estás?



<2es> Hello, how are you? -> Hola, ¿cómo estás?

Table 5: Portuguese→Spanish BLEU scores using various models.

	Model	Zero-shot	BLEU
(a)	PBMT bridged	no	28.99
(b)	NMT bridged	no	30.91
(c)	NMT Pt→Es	no	31.50
(d)	Model 1 (Pt→En, En→Es)	yes	21.62
(e)	Model 2 (En↔{Es, Pt})	yes	24.75
(f)	Model 2 + incremental training	no	31.77

# Google's Multilingual NMT

- Interpolate the language code embeddings

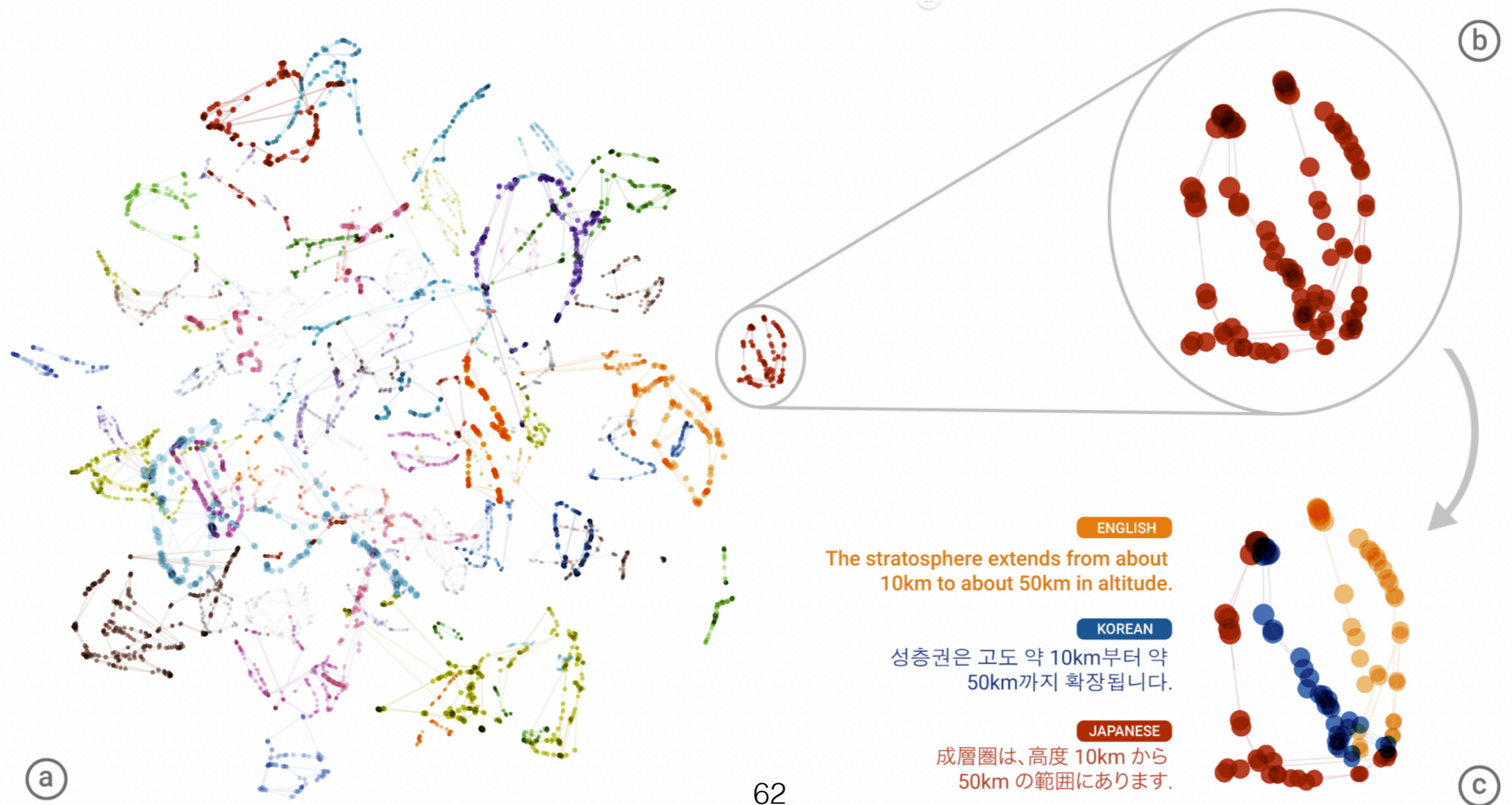
$$(1 - w)\langle 2ja \rangle + w\langle 2ko \rangle$$

Russian/Belarusian:	I wonder what they'll do next!
$w_{be} = 0.00$	Интересно, что они сделают дальше!
$w_{be} = 0.20$	Интересно, что они сделают дальше!
$w_{be} = 0.30$	<u>Цікаво</u> , что они будут делать дальше!
$w_{be} = 0.44$	<u>Цікаво</u> , што вони будуть робити далі!
$w_{be} = 0.46$	<u>Цікаво</u> , што вони будуть робити далі!
$w_{be} = 0.48$	<u>Цікаво</u> , што яны зробіць далей!
$w_{be} = 0.50$	Цікава, што яны будуць рабіць далей!
$w_{be} = 1.00$	Цікава, што яны будуць рабіць далей!
Japanese/Korean:	I must be getting somewhere near the centre of the earth.
$w_{ko} = 0.00$	私は地球の中心の近くにどこかに行っているに違いない。
$w_{ko} = 0.40$	私は地球の中心近くのどこかに着いているに違いない。
$w_{ko} = 0.56$	私は地球の中心の近くのどこかになっているに違いない。
$w_{ko} = 0.58$	私は 지구 중심의 가까이에 어딘가에도 착하고 있어야 한다.
$w_{ko} = 0.60$	나는 지구 중심의 가까이에 어딘가에도 착하고 있어야 한다.
$w_{ko} = 0.70$	나는 지구 중심 근처 어딘가에도 착해야 합니다.
$w_{ko} = 0.90$	나는 어딘가 지구의 중심 근처에도 착해야 합니다.
$w_{ko} = 1.00$	나는 어딘가 지구의 중심 근처에도 착해야 합니다.



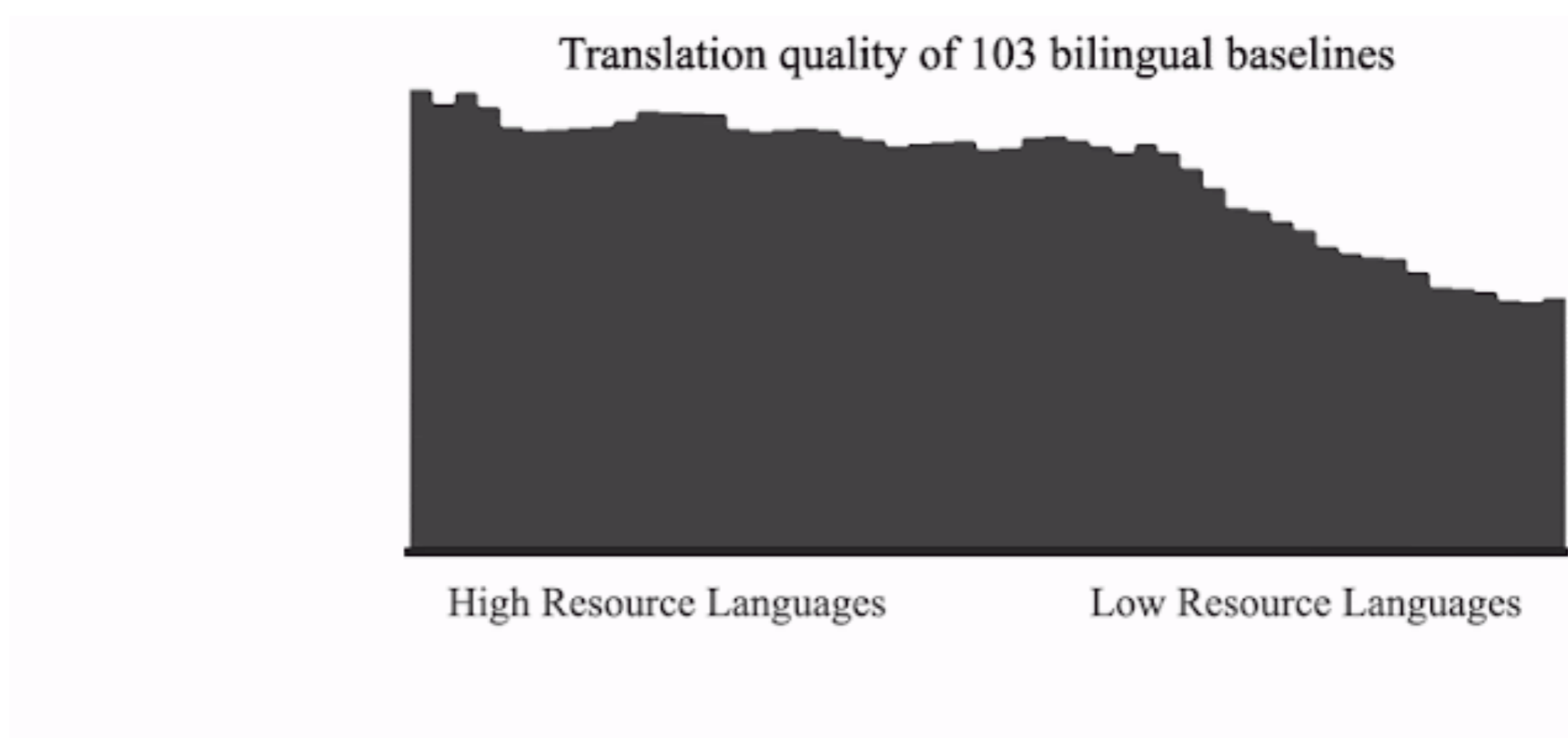
# Google's Multilingual NMT

- Sentence embeddings learned by MNMT are clustered by languages



# Multilingual vs Bilingual

- Multilingual NMT especially improves low-resource language translation



# Hybrid MT



# Incorporating Discrete Translation Lexicons in Neural Machine Translation

- Estimate the probability of translation lexicons (a.k.a translation phrases)  $p_l(\cdot)$
- Integrate this lexicon probability  $p_l(\cdot)$  with NMT probability  $p_m(\cdot)$ 
  - Add them as a bias to the NMT's softmax output

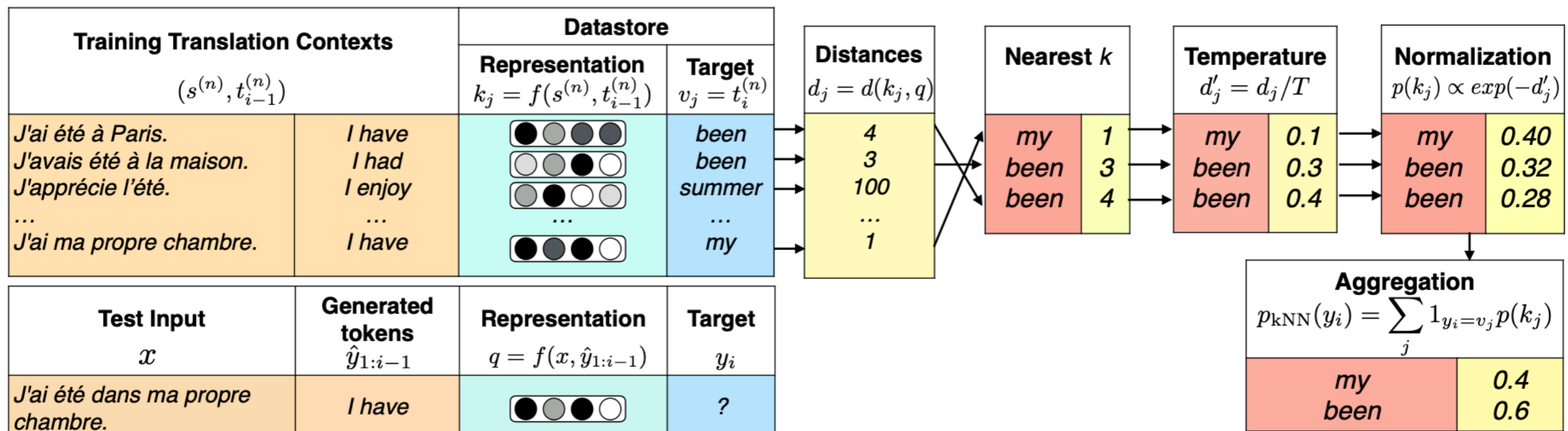
$$p_b(e_i|F, e_1^{i-1}) = \text{softmax}(W_s \eta_i + b_s + \log(p_l(e_i|F, e_1^{i-1}) + \epsilon)).$$

- Linear interpolation

$$p_o(e_i|F, e_1^{i-1}) = \lambda p_l(e_i|F, e_1^{i-1}) + (1 - \lambda) p_m(e_i|F, e_1^{i-1})$$

# KNN-MT

- Construct a datastore (e.g., phrase table in SMT), and perform KNN retrieval to get related phrases for a test input



$$p(y_i | x, \hat{y}_{1:i-1}) = \lambda p_{\text{KNN}}(y_i | x, \hat{y}_{1:i-1}) + (1 - \lambda) p_{\text{MT}}(y_i | x, \hat{y}_{1:i-1})$$

# Prompt-based MT

- Given a **translation instruction** and **few-shot examples**, ask a LLM to perform MT

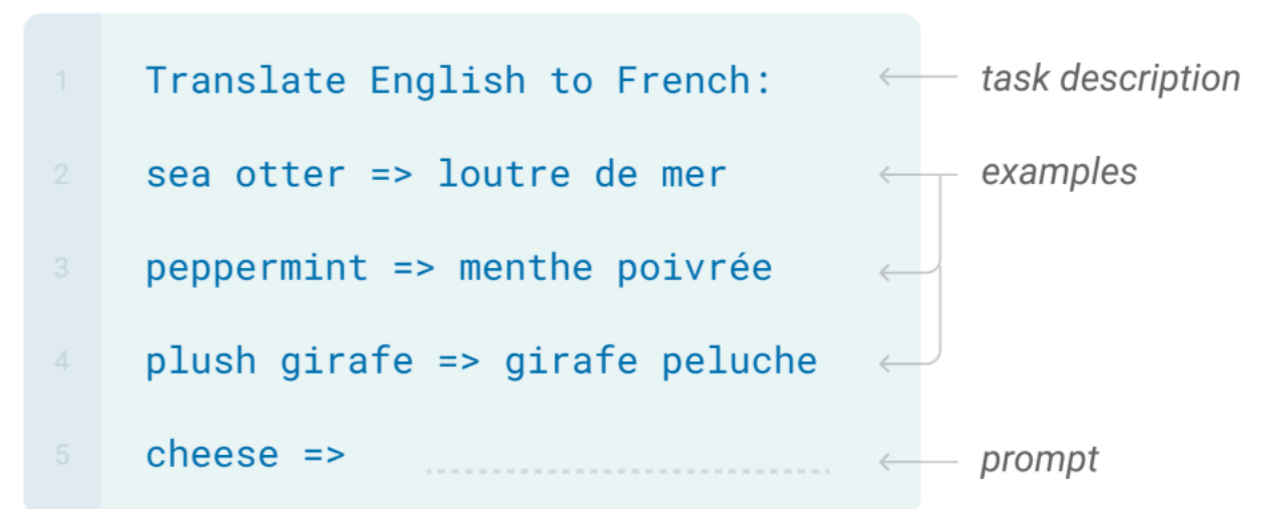
## Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



## Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



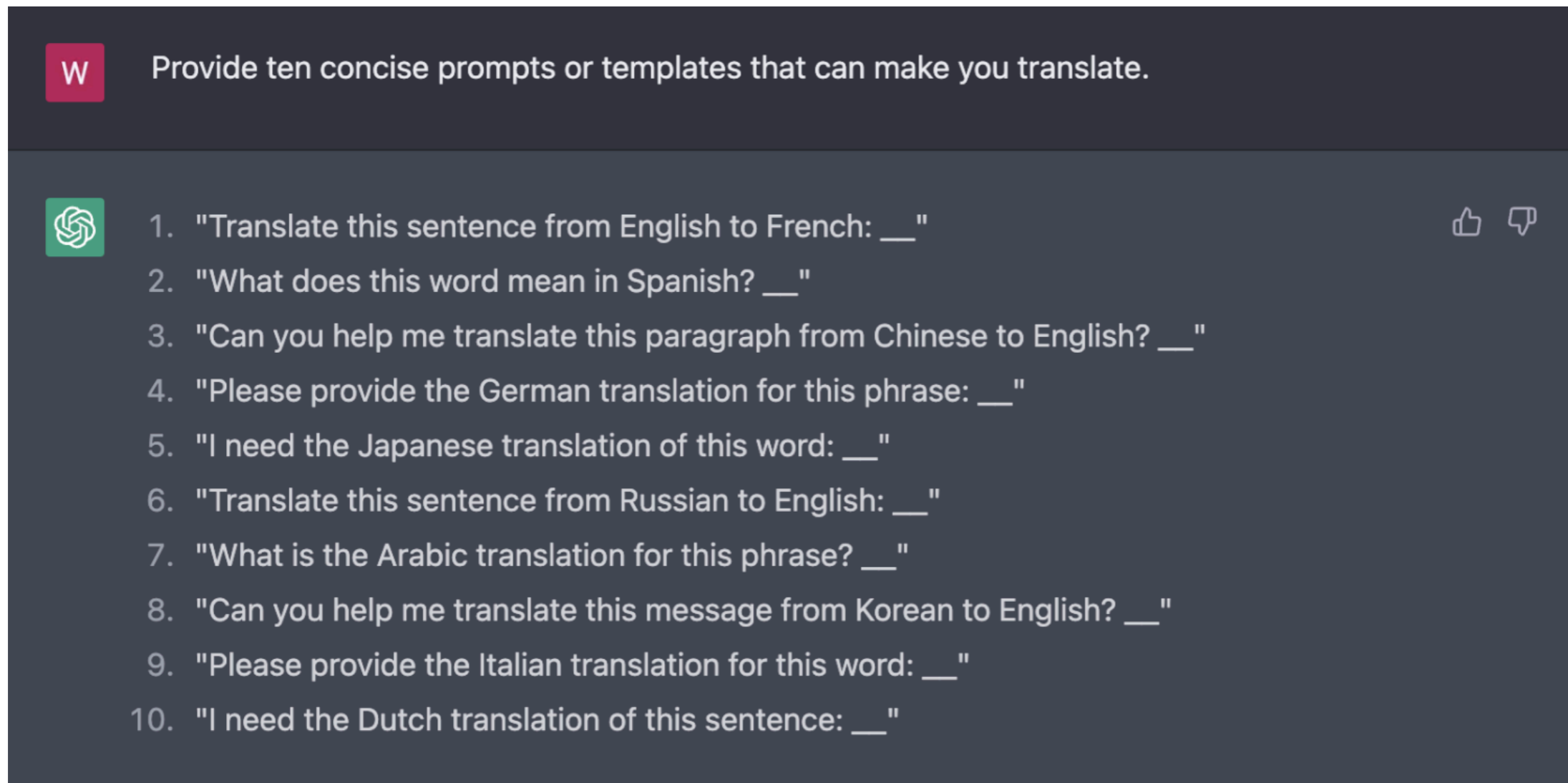
# Prompt-based MT

- Perform comparable or even better at XX-English translation, but still far behind at English-XX translation
- GPT-3's training data (93% by word count) is still English

Setting	En→Fr	Fr→En	En→De	De→En	En→Ro	Ro→En
SOTA (Supervised)	<b>45.6<sup>a</sup></b>	35.0 <sup>b</sup>	<b>41.2<sup>c</sup></b>	40.2 <sup>d</sup>	<b>38.5<sup>e</sup></b>	<b>39.9<sup>e</sup></b>
XLM [LC19]	33.4	33.3	26.4	34.3	33.3	31.8
MASS [STQ <sup>+</sup> 19]	<u>37.5</u>	34.9	28.3	35.2	<u>35.2</u>	33.1
mBART [LGG <sup>+</sup> 20]	-	-	<u>29.8</u>	34.0	35.0	30.5
GPT-3 Zero-Shot	25.2	21.2	24.6	27.2	14.1	19.9
GPT-3 One-Shot	28.3	33.7	26.2	30.4	20.6	38.6
GPT-3 Few-Shot	32.6	<u>39.2</u>	29.7	<u>40.6</u>	21.0	<u>39.5</u>

# Prompt-based MT

- Ask ChatGPT to create prompts for MT



The screenshot shows a chat interface with a dark background. At the top, a user prompt (indicated by a pink 'W' icon) asks for ten concise prompts or templates for translation. Below, the ChatGPT response (indicated by a green icon) lists ten numbered prompts for various language translations. To the right of the list are thumbs-up and thumbs-down icons.

**W** Provide ten concise prompts or templates that can make you translate.

**1.** "Translate this sentence from English to French: \_\_\_"

**2.** "What does this word mean in Spanish? \_\_\_"

**3.** "Can you help me translate this paragraph from Chinese to English? \_\_\_"

**4.** "Please provide the German translation for this phrase: \_\_\_"

**5.** "I need the Japanese translation of this word: \_\_\_"

**6.** "Translate this sentence from Russian to English: \_\_\_"

**7.** "What is the Arabic translation for this phrase? \_\_\_"

**8.** "Can you help me translate this message from Korean to English? \_\_\_"

**9.** "Please provide the Italian translation for this word: \_\_\_"

**10.** "I need the Dutch translation of this sentence: \_\_\_"

# Future Research of MT

# Six Challenges of NMT

1. NMT works poorly on **out-of-domain** sentences
2. Works better in high-resource languages, not in **low-resource** languages.
3. Weakness in **low-frequency words** w.r.t. SMT
4. Bad at **very long sentences**
5. Attentions **do not always fulfill** the role of a **word alignment**
6. Beam search decoding only works with **a smaller beam size**, and deteriorates when exposed to a larger search space.

# Massively Multilingual NMT in the Wild

- **Data and supervision:** learn from **monolingual** data for most low-resourced languages (e.g., pre-training, data augmentation such as back-translation (Sennrich et al. 2015), language model fusion (Gulcehre et al. 2015), unsupervised NMT (Lample et al. 2017))
- **Multitask training:** cross-lingual transfer (Neubig, Hu 2018), meta learning (Nichol et al. 2018), curriculum learning (Graves et al. 2017)
- **Increasing Capacity:** train on more languages, efficiency
- **Architecture & Vocabulary:** character NMT (Lee et al. 2017), byte-based NMT (Gillick et al. 2015)



# Comparison between SMT and NMT

- **SMT** (non-parametric model)
  - Explicitly store translation phrase pairs
  - High precision, but low coverage, thus poor generation
  - Easy manipulation (insert/delete/update pairs in the translation phrase table)
- **NMT** (over-parametric model)
  - Implicitly store translation phrase pairs in model parameters
  - Better generation, but data-hungry
  - Hard manipulation (catastrophic forgetting after fine-tuning, domain adaptation issues)
- **Hybrid MT** (e.g., Retrieval-MT, Prompt-based MT)
  - Leverage both neural network parameters and retrieved translation phrase pairs
  - Perform better than NMT on domain adaptation when providing new domain information from retrieval.

Questions?