CS769 Advanced NLP

# Machine Translation

Junjie Hu

Slides adapted from Austin

https://junjiehu.github.io/cs769-spring22/

# Goal for Today

- Parallel Corpus
- Noisy Channel MT (SMT)
  - Lexical Translation
  - Word Alignment
- Neural Machine Translation
  - Architecture: LSTM, CNN, Transformer
  - Multilingual NMT
- 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

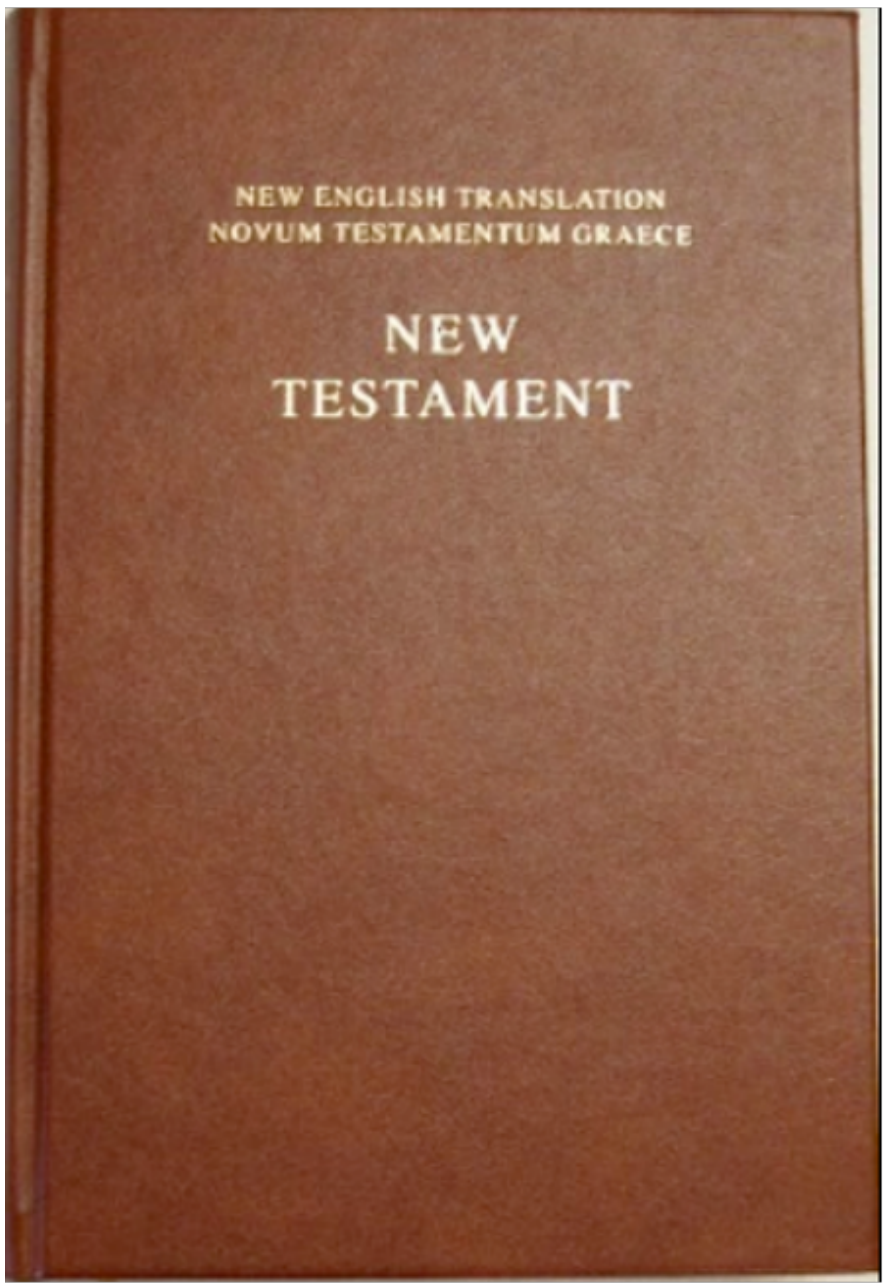
- We are given a corpus of sentence pairs in two languages to train our machine translation models.
- Source language is also called foreign language, denoted as f.
- Conventionally (in earlier studies before NMT) target language is usually referred to English, denoted as e.

# Parallel Corpus

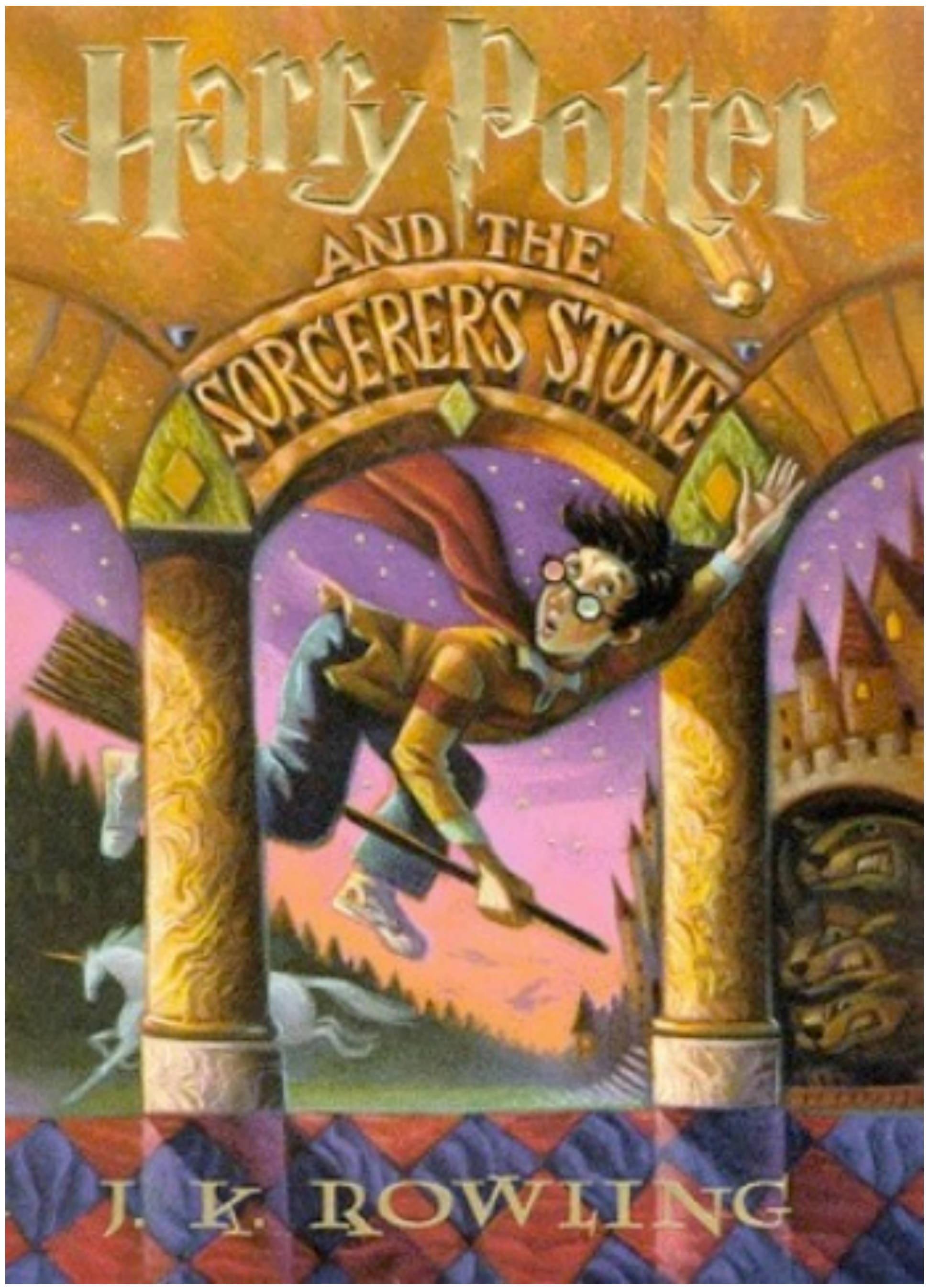
| | | CLASSIC SOUPS | Sm. | Lg. |
|---|---|---|---|---|
| 清燉雞湯 | 57. | House Chicken Soup (Chicken, Celery, Potato, Onion, Carrot) | 1.50 | 2.75 |
| 雞飯湯 | 58. | Chicken Rice Soup | 1.85 | 3.25 |
| 雞麵湯 | 59. | Chicken Noodle Soup | 1.85 | 3.25 |
| 廣東雲吞 | 60. | Cantonese Wonton Soup | 1.50 | 2.75 |
| 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 |

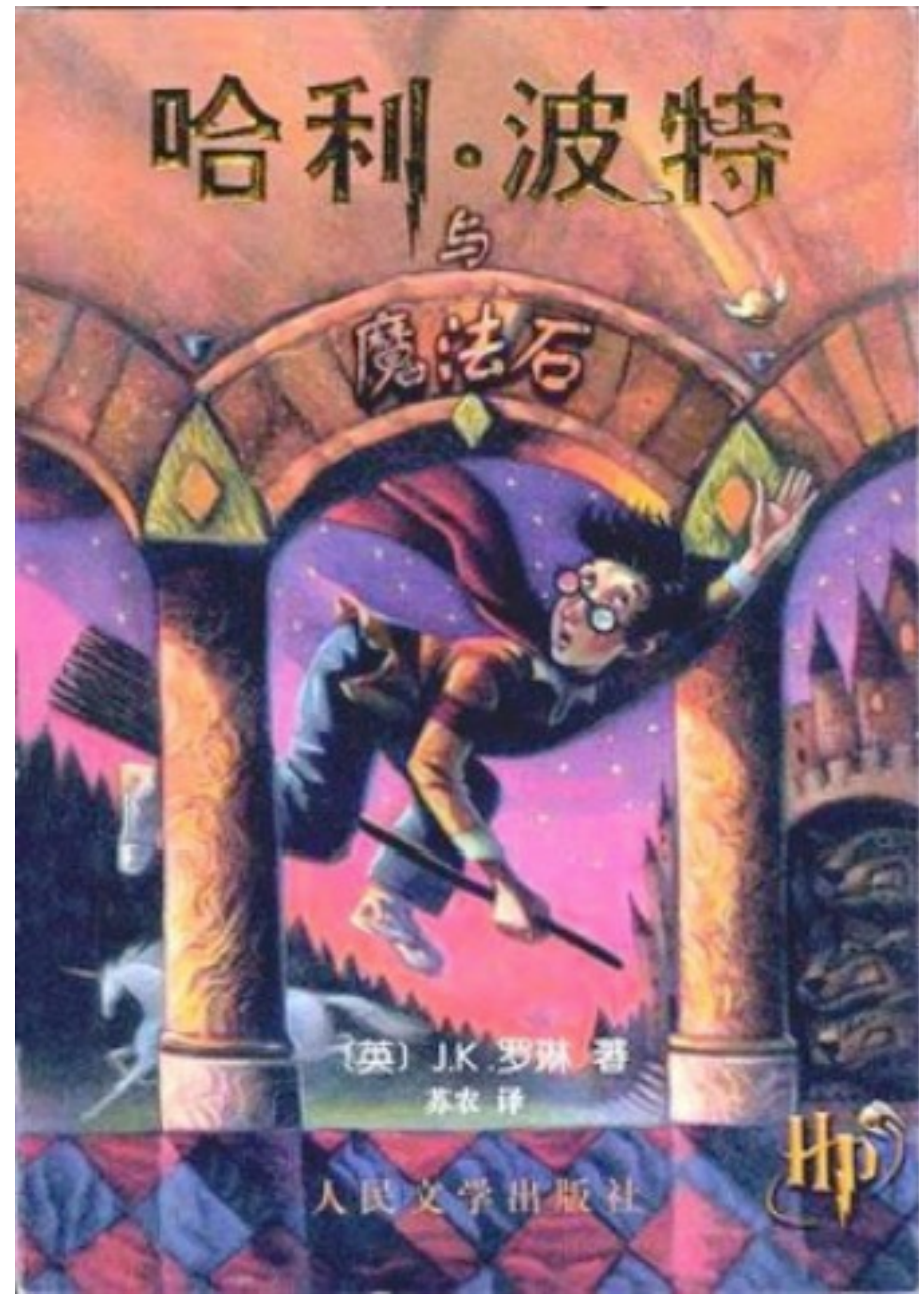
# Parallel Corpus


# Parallel Corpus




# Parallel Corpus

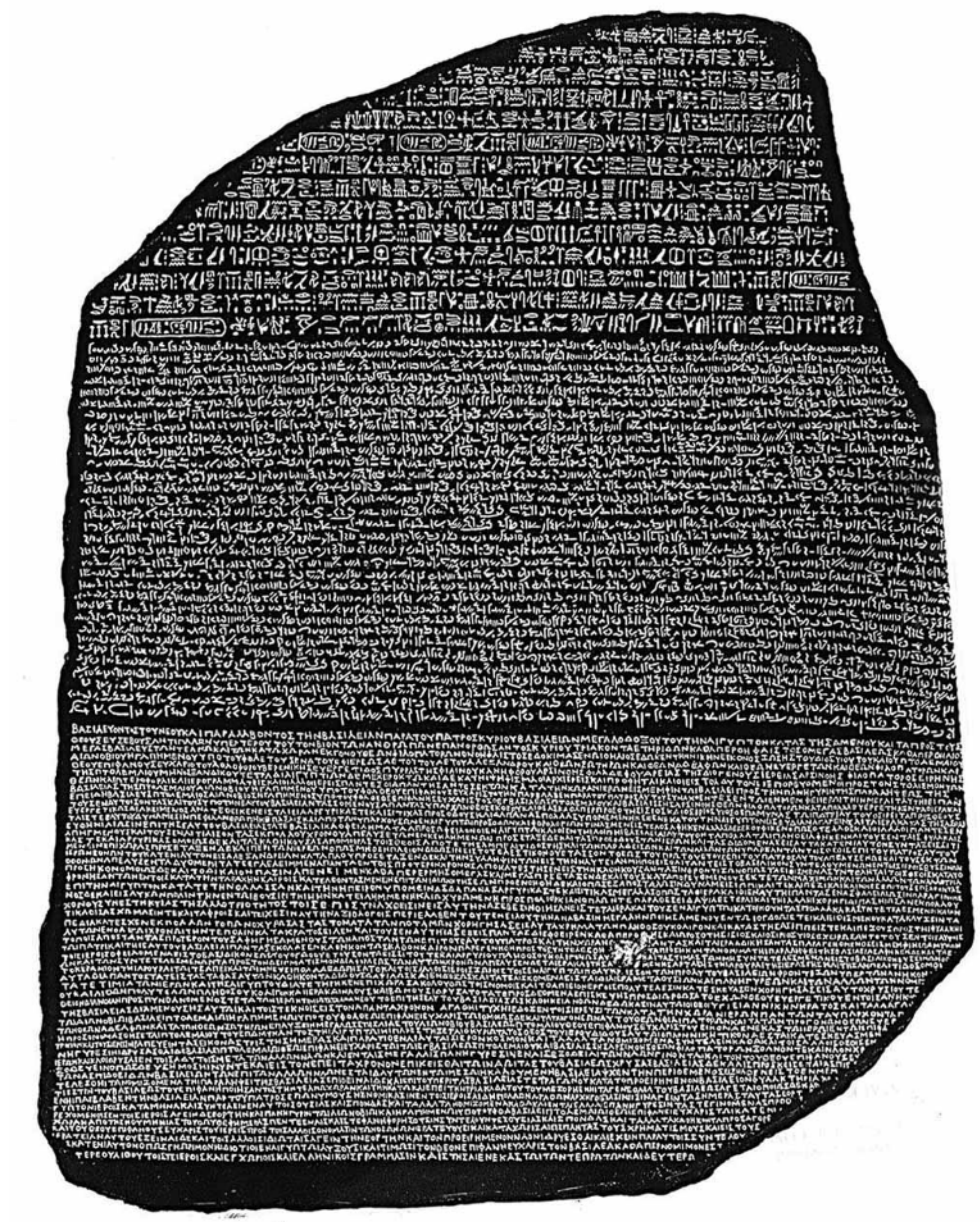


# Parallel Corpus

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

https://www.statmt.org/wmt21/

# OPUS Parallel Corpus

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

OPUS

**Search & download resources:** de (German) | en (English) | all | show all versions

**Language resources:** click on [ tmx | moses | 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 | moses | de en | de en | | | | de en | | sample | |
| **WikiMatrix** v1 | 1 | 6.2M | 443.1M | 1.0G | xces de en | de en | tmx | moses | de en | de en | | | | de en | | sample | |
| **ParaCrawl** v8 | 364 | 36.3M | 450.7M | 478.7M | xces de en | de en | tmx | moses | de en | de en | | | | de en | | | |
| **EUbookshop** v2 | 15373 | 9.6M | 337.4M | 380.2M | xces de en | de en | tmx | moses | de en | de en | | alg | dic | de en | query | sample | moses/strict |
| **EuroPat** v3 | 1 | 12.6M | 318.2M | 387.8M | xces de en | de en | tmx | moses | de en | de en | | | | de en | | sample | |
| **wikimedia** v20210402 | 1 | 0.1M | 11.0M | 349.2M | xces de en | de en | tmx | moses | de en | de en | | | | de en | | sample | |
| **CCAligned** v1 | 1852 | 15.3M | 150.8M | 159.5M | xces de en | de en | tmx | moses | de en | de en | | | | de en | | sample | |
| **TildeMODEL** v2018 | 7 | 4.3M | 108.8M | 131.4M | xces de en | de en | tmx | moses | 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 | moses | de en | de en | | alg smt | dic | de en | | sample | |

https://opus.nlpl.eu/

# Noisy Channel MT
# (Statistic Machine Translation)

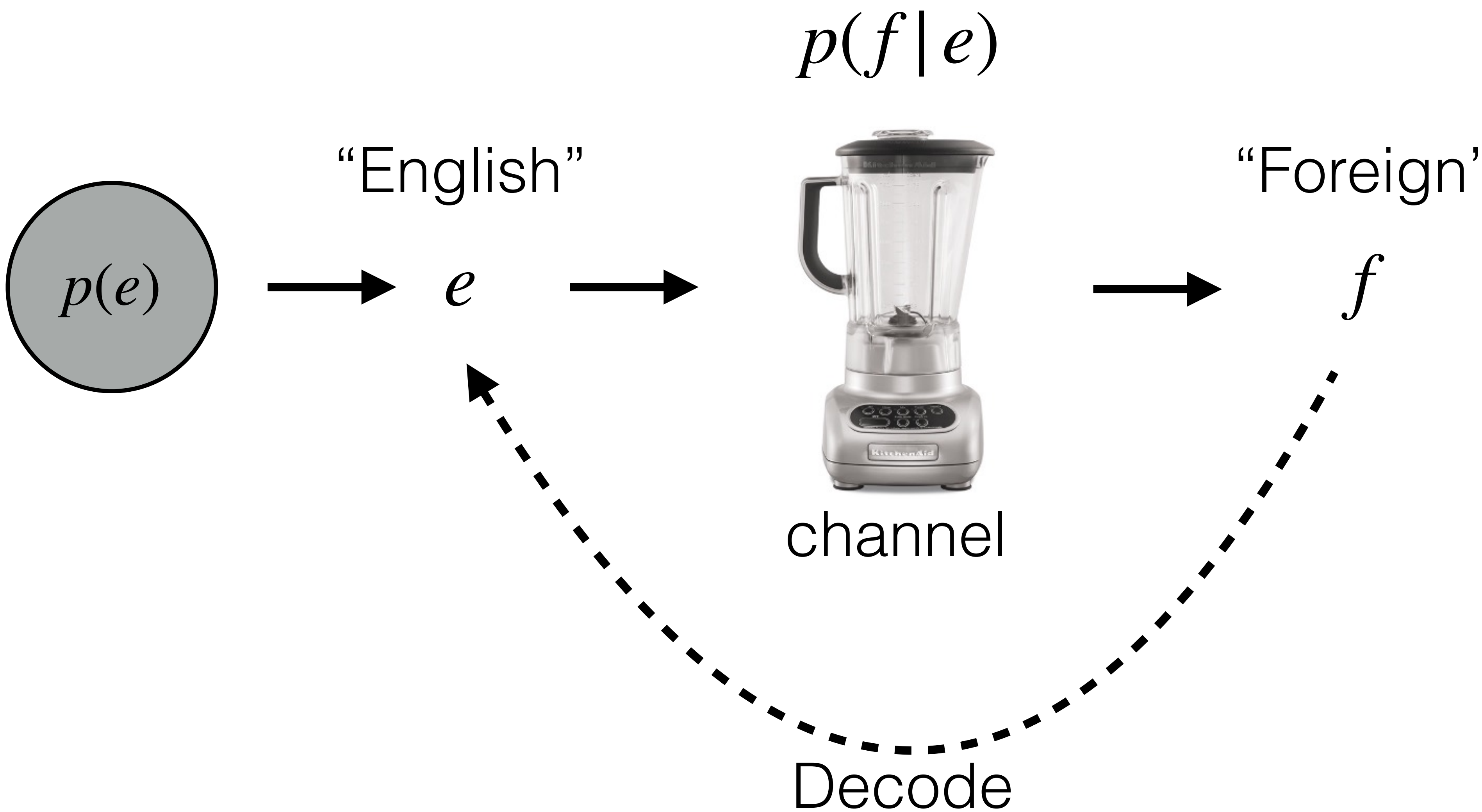
# $f$-to-$e$ Translation

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

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

$$\hat{e} = \arg\max_e \underline{p(e|f)}$$

$$= \arg\max_e \frac{p(e) \times p(f|e)}{p(f)}$$

$$= \arg\max_e \underline{p(e)} \times \underline{p(f|e)}$$

Language Model

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

What's the benefit of the Noisy Channel decomposition in stead 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 $p(f \mid e)$ is a bit harder
  - $f =$ ie voudrais un peu de frommage $\quad p(f \mid e)$
  - $e_1 =$ I would like some cheese $\quad$ 0.4
  - $e_2 =$ I would like a little of cheese $\quad$ 0.5
  - $e_3 =$ There is no train to Barcelona $\quad$ >0.00001

# Estimate Channel Translation Model

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

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

- 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)

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

# Lexical Translation

- Goal: a model $p(\mathbf{f} \mid \mathbf{e}, m)$, where $\mathbf{e} = \langle e_1, e_2, \ldots, e_l \rangle$ and $\mathbf{f} = \langle f_1, f_2, \ldots, f_m \rangle$, assuming that there is some distribution $p(m \mid l)$ 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:

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

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.

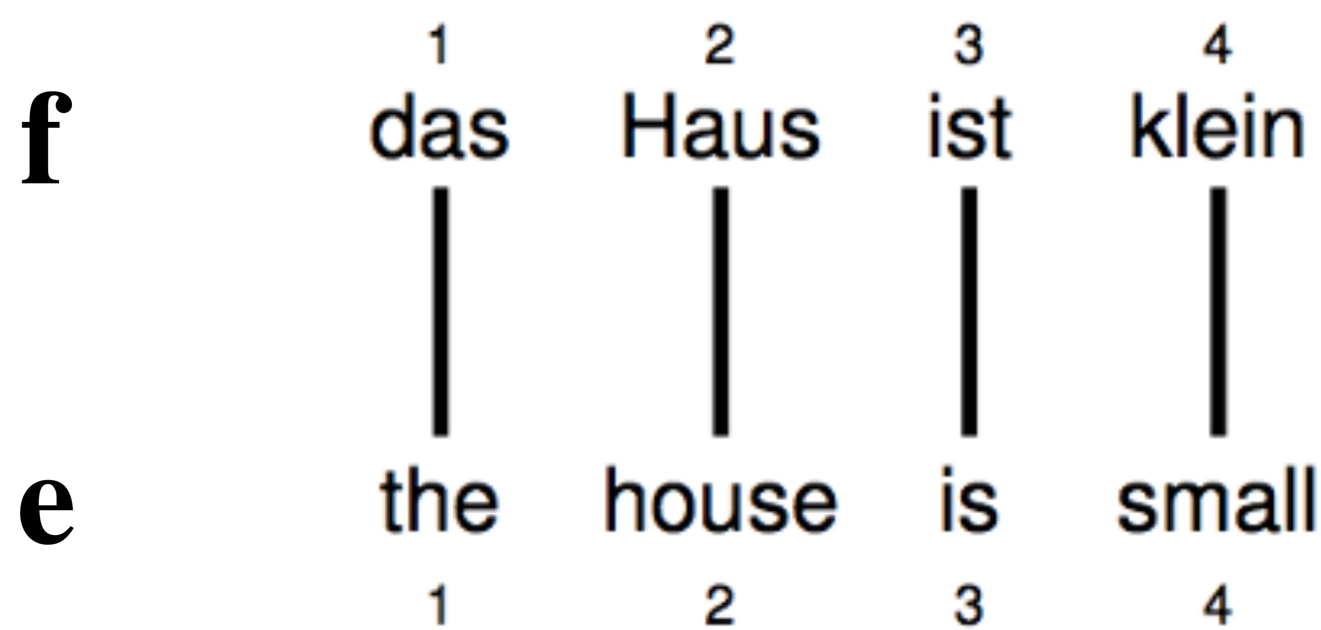
- E.g. IBM Model 1, 2, 3, Giza++, FastAlign

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

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:

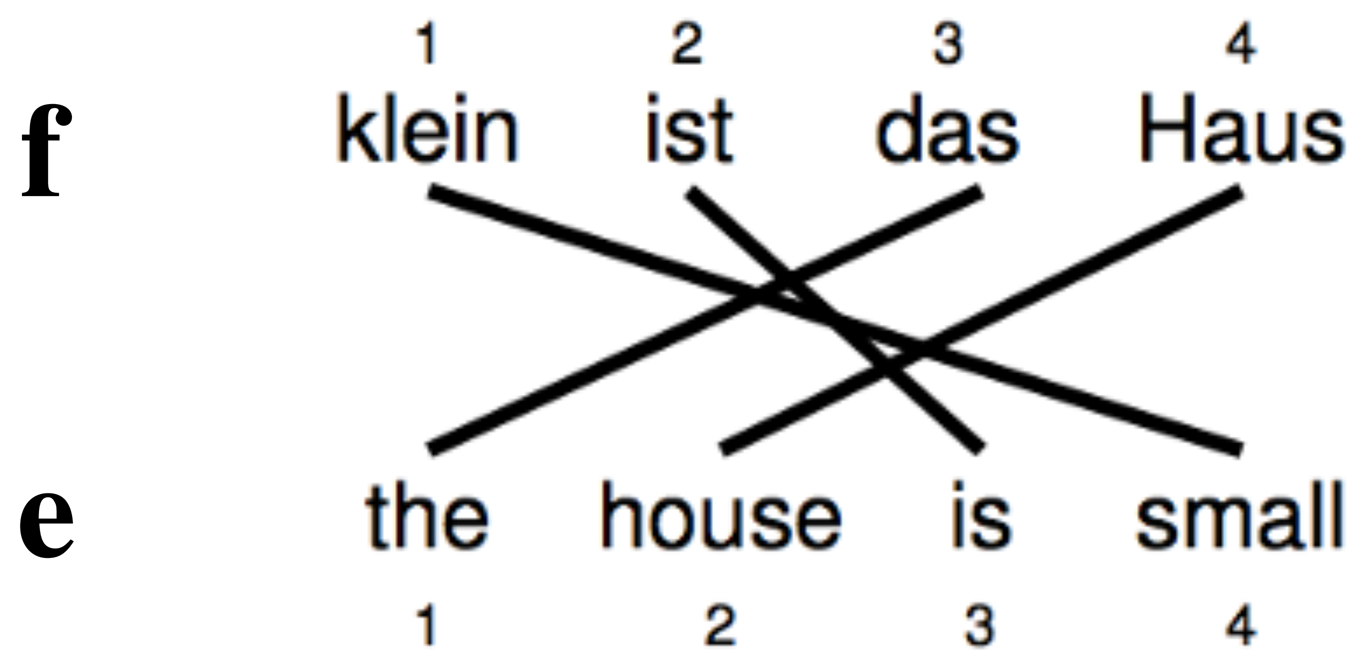



$$\mathbf{a} = (1,2,3,4)^{\mathsf{T}}$$

# Reordering

- Words may be reordered during translation

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



# Word Dropping

- A source word may not be translated at all

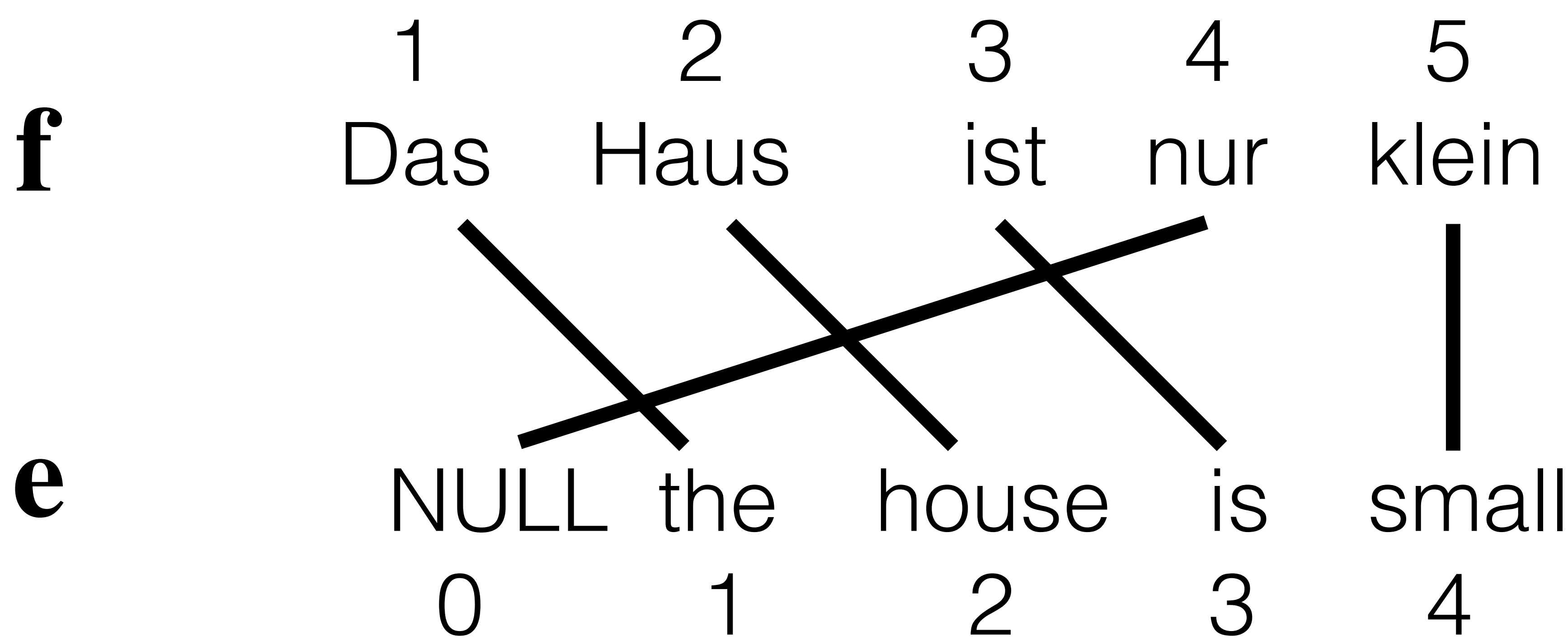
$$\mathbf{a} = (2,3,4)^{\mathsf{T}}$$

| | | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | |
| $\mathbf{f}$ | Haus | ist | klein | |
| $\mathbf{e}$ | the | house | is | small |
| | 1 | 2 | 3 | 4 |

# 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

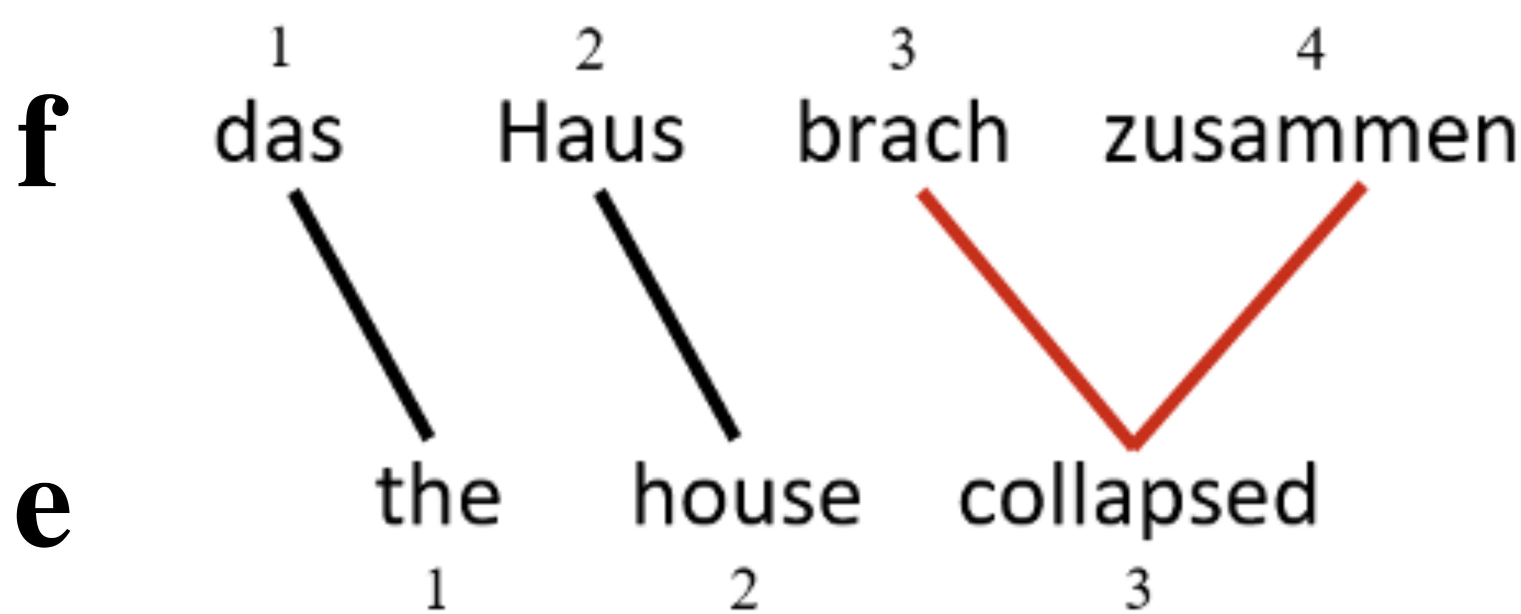
$$\mathbf{a} = (1,2,3,0,4)^{\mathsf{T}}$$



# One-to-many Translation

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

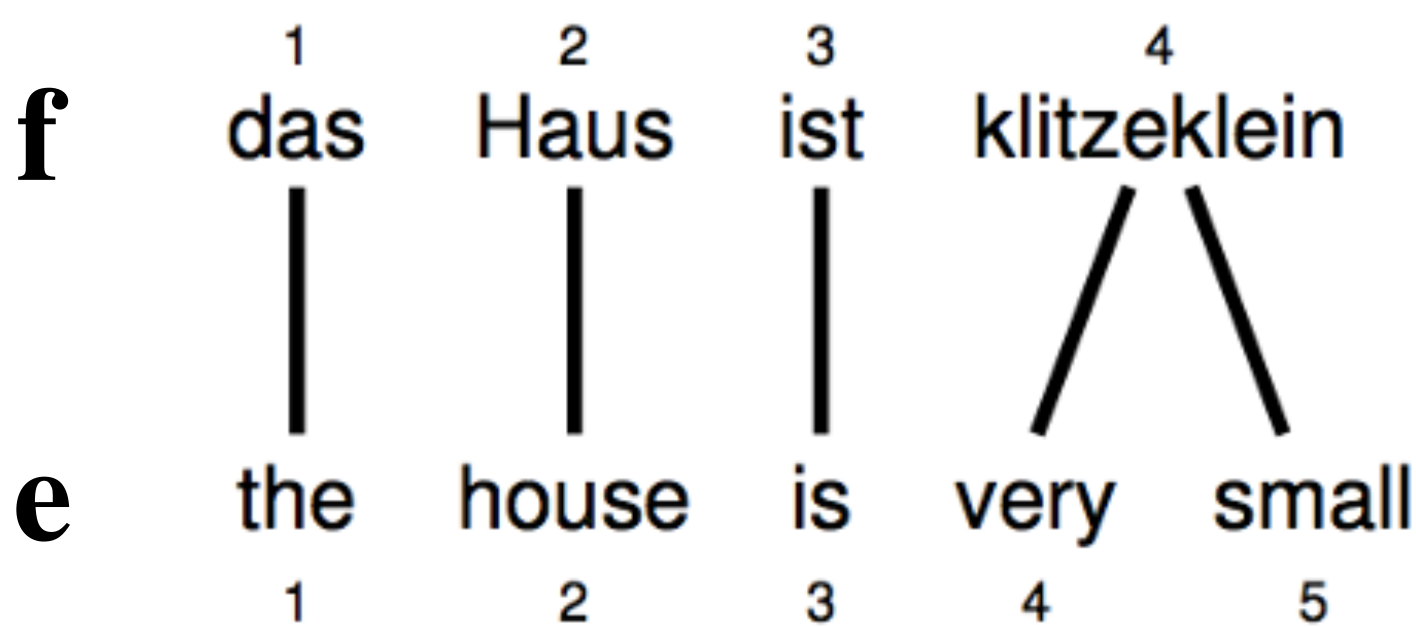
$$\mathbf{a} = (1,2,3,3)^{\mathsf{T}}$$



# Many-to-one Translation

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

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




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?

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

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

- MLE of two probability with the latent alignment

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

count$(a_i \mid i, l, m)$ is the no. time $f_i$ is aligned to $e_{a_i}$ in the training set. 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)

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

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

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

# Expectation-Maximization (EM) Algorithm

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

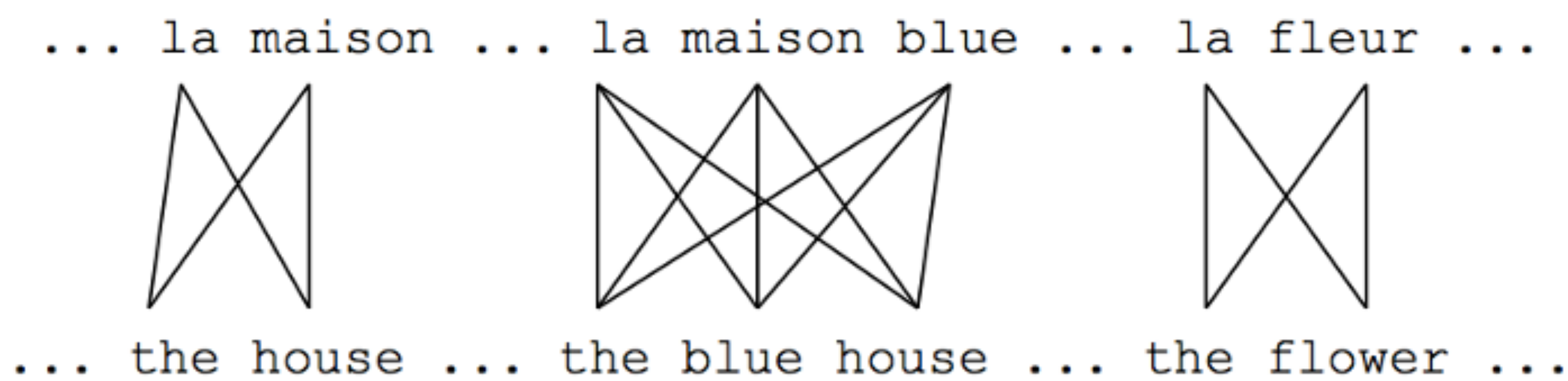
1. **E- Step**: use the current parameters to compute "expected" alignments

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

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

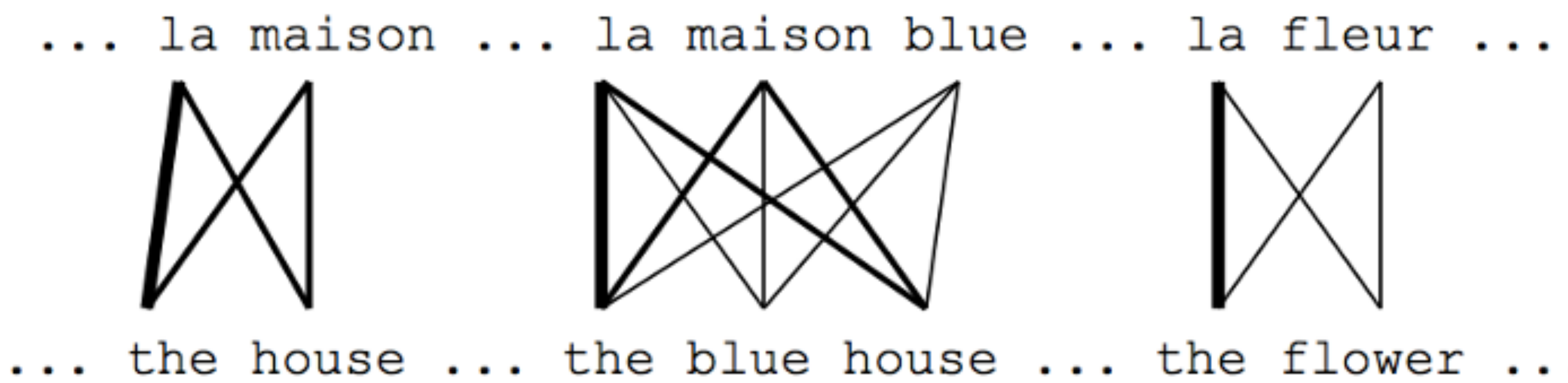
$$p(f_i \mid e_{a_i}) = \frac{\text{count}(f_i, e_{a_i})}{\text{count}(e_{a_i})}$$

# EM for IBM Model 1



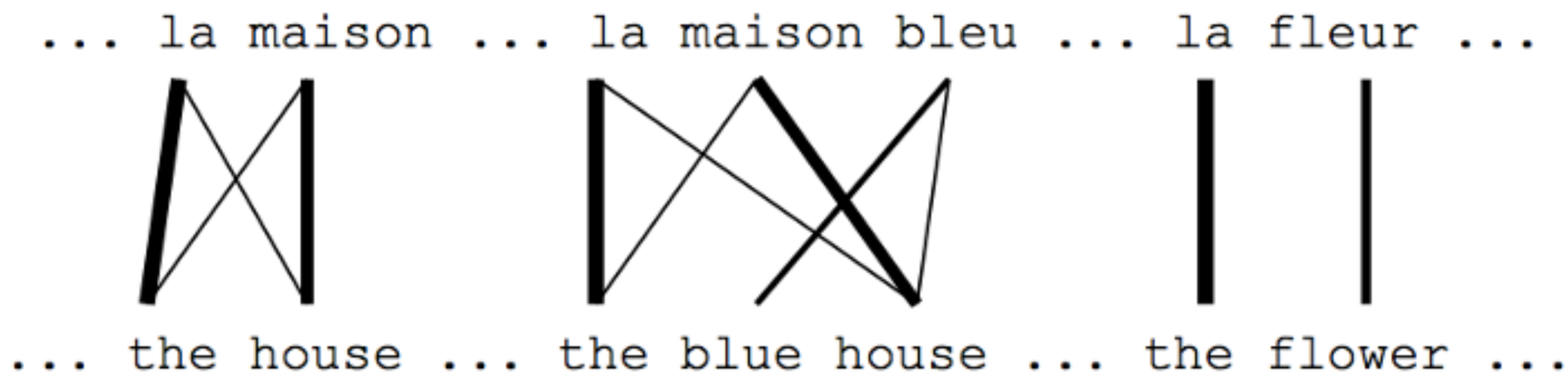

- Initial step: all alignments equally likely
- Model learns that, e.g., la is often aligned with the

# EM for IBM Model 1




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

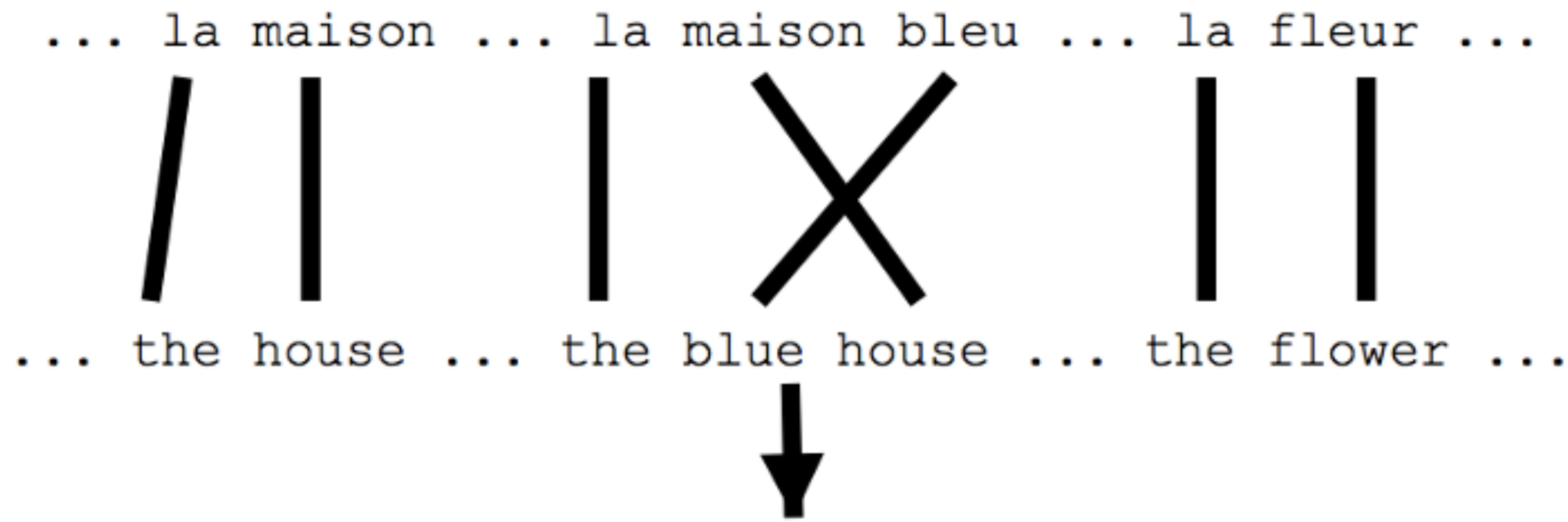
# EM for IBM Model 1




- After another iteration

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

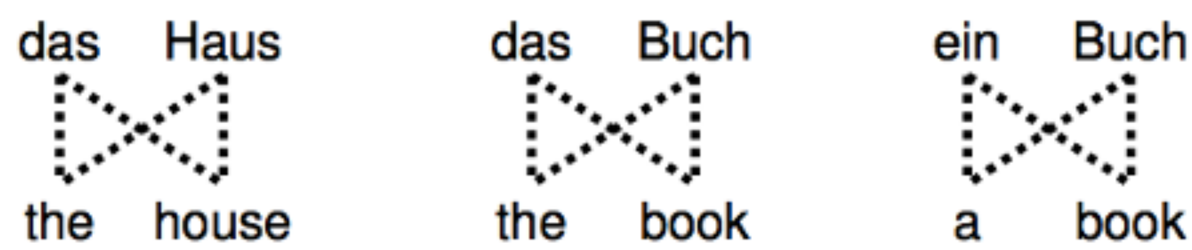
# EM for IBM Model 1




```
p(la|the) = 0.453
p(le|the) = 0.334
p(maison|house) = 0.876
p(bleu|blue) = 0.563
...
```

- Parameter estimation from the aligned corpus

# Convergence




| $e$ | $f$ | 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

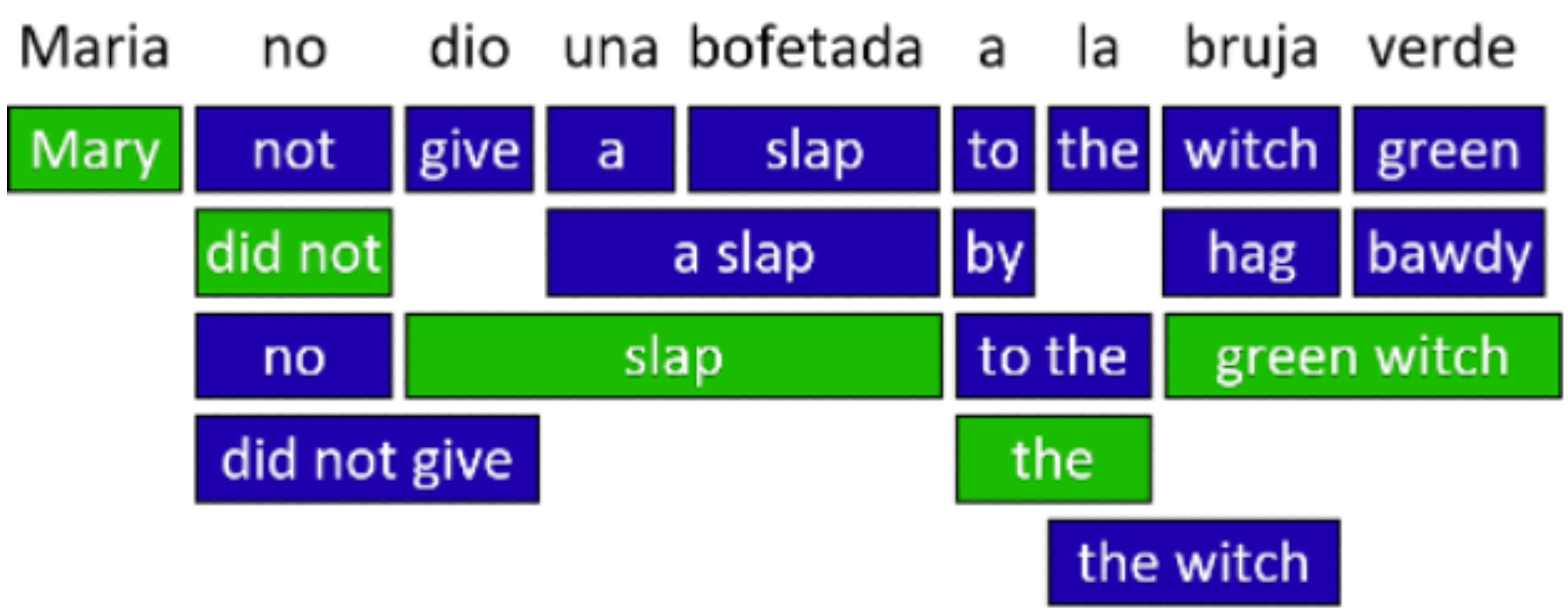
- Moses (Koehn 2009)

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

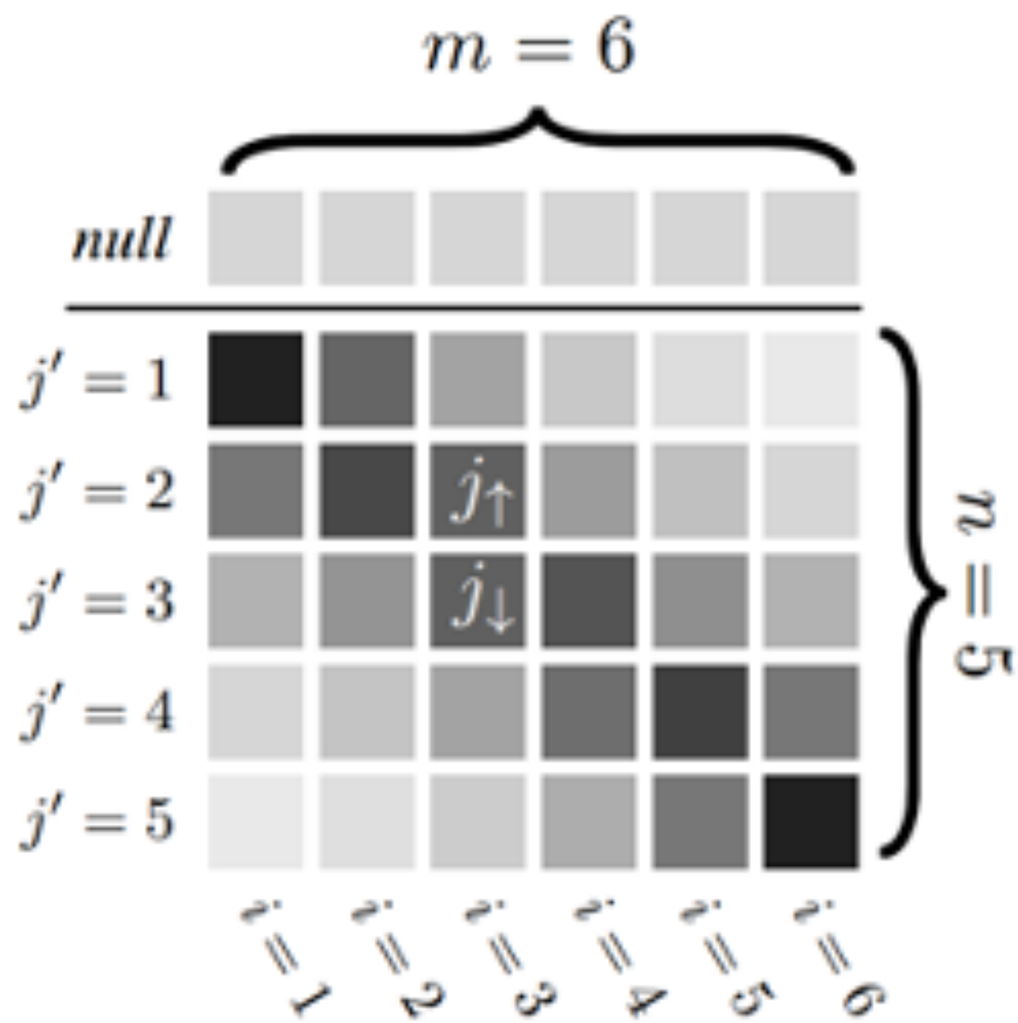
- Phrase-based MT:
  - Allow multiple words to translate as chunks (including many-to-one)
  - Introduce **another latent variable**, the **source segmentation**




Adapted from Koehn (2006)

# Extensions: Alignment Heuristics

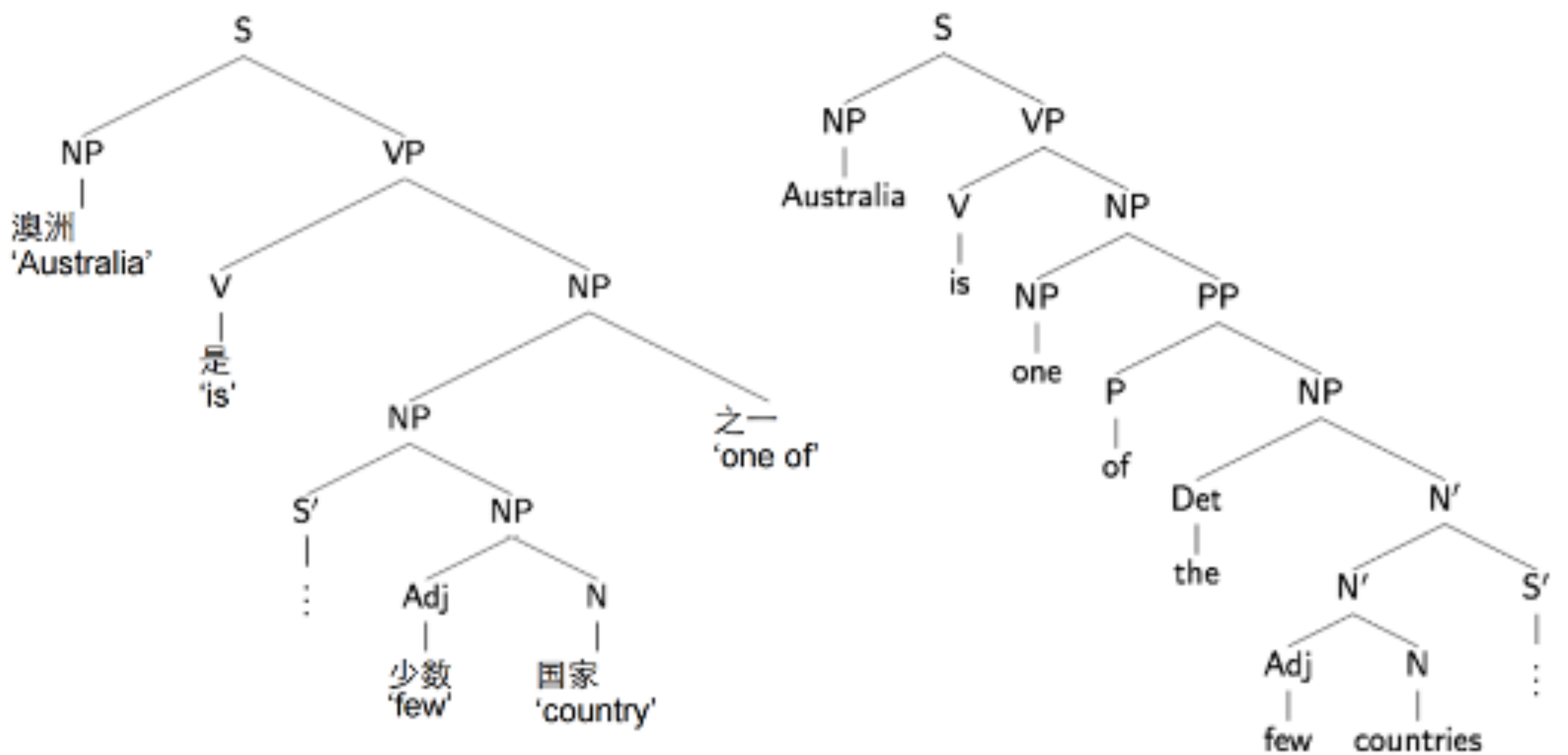
- Alignment Priors:
  - Instead of assuming the alignment decisions are uniform, impose (or learn) a prior over alignment grids




Chahuneau et al. 2013

# Extensions: Hierarchical Phrase-based MT

- Syntactics structure
- Instead of extracting **parallel phrases**, extract **translation rules** of the form: X 之一 → one of the X




Chang 2005, Galley et al. 2006

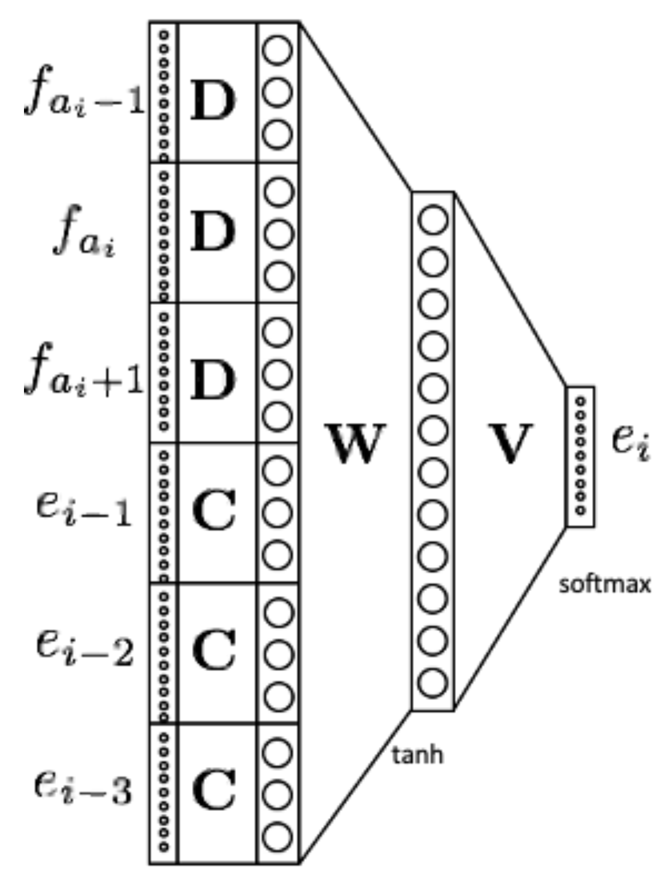
# Neural Machine Translation

# 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

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

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




Devlin et al. 2014

# Neural Features for Translation

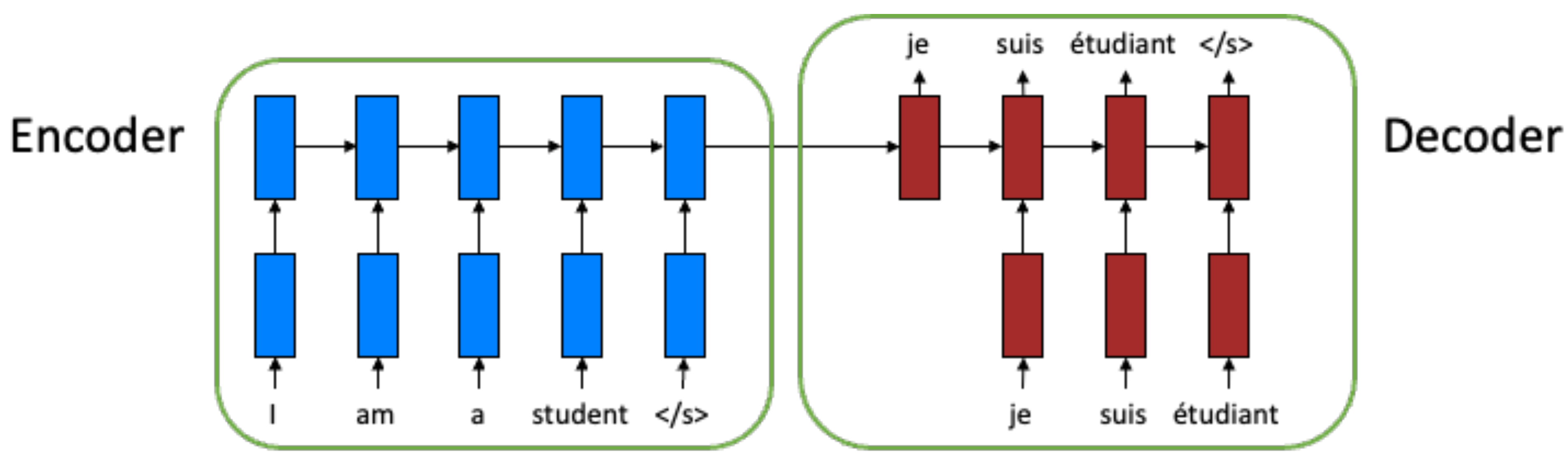
- Word alignment is still needed.
- Improves over SMT

| BOLT Test | | |
|---|---|---|
| | Ar-En | |
| | **BLEU** | **% Gain** |
| "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




Sutskever et al. 2014

# Attention MT Models

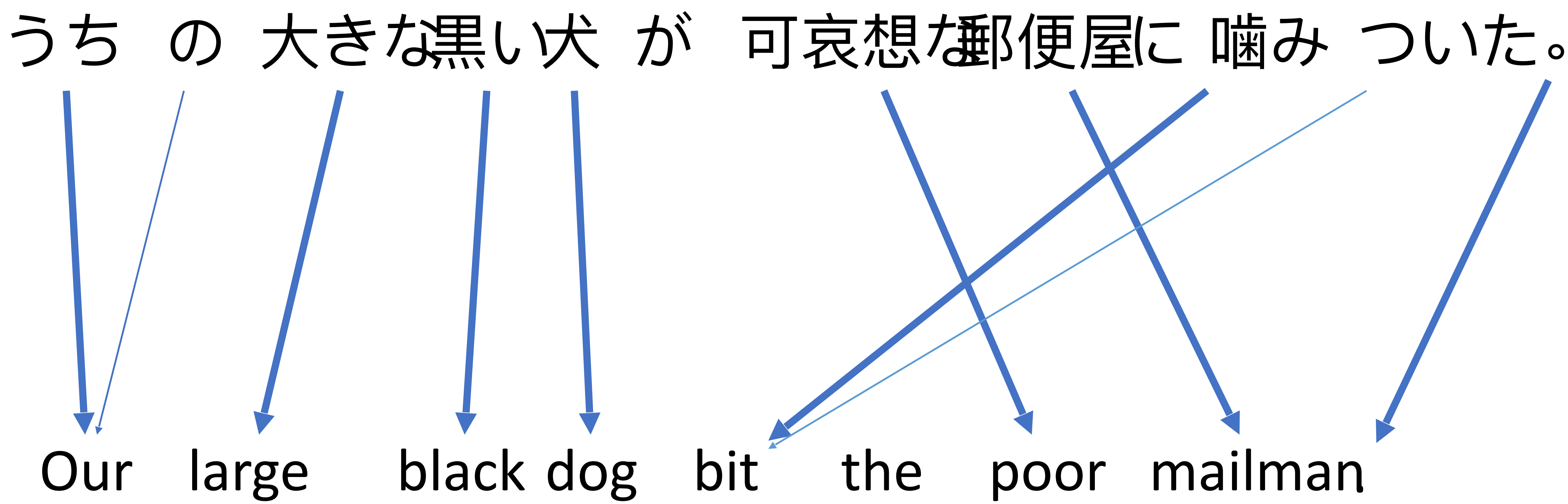
- The encoder-decoder model struggles with long sentences

- 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?

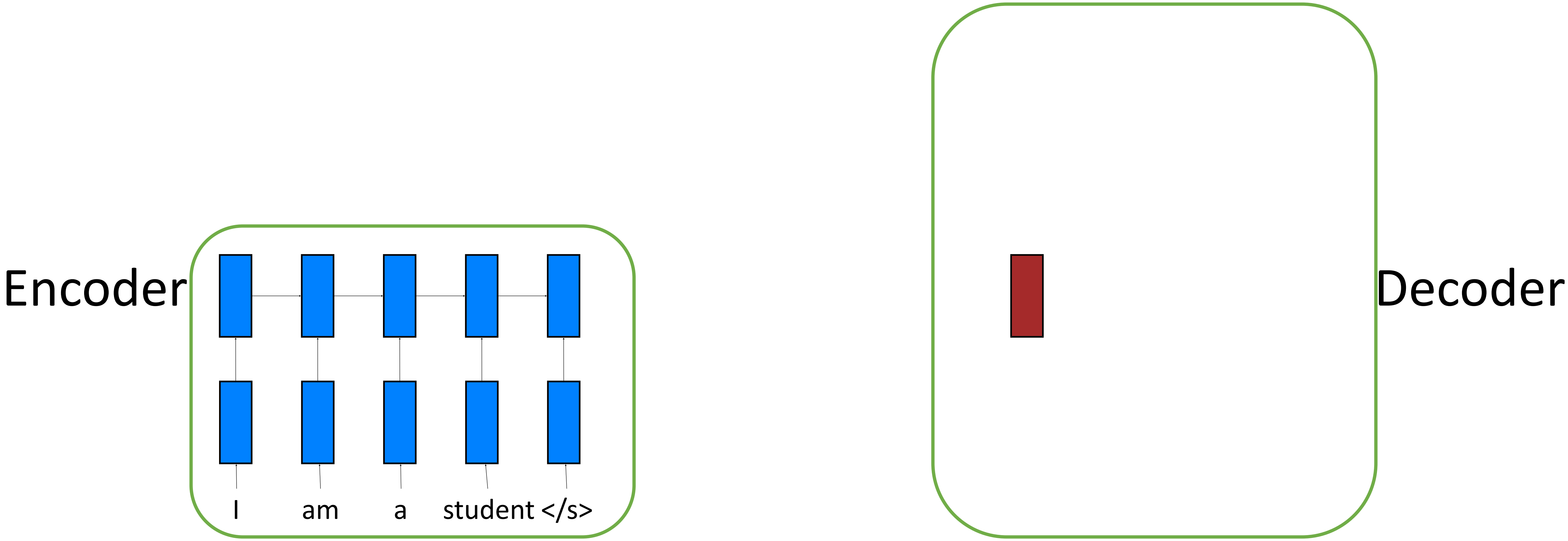
Bahdanau et al. 2014

# Intuition



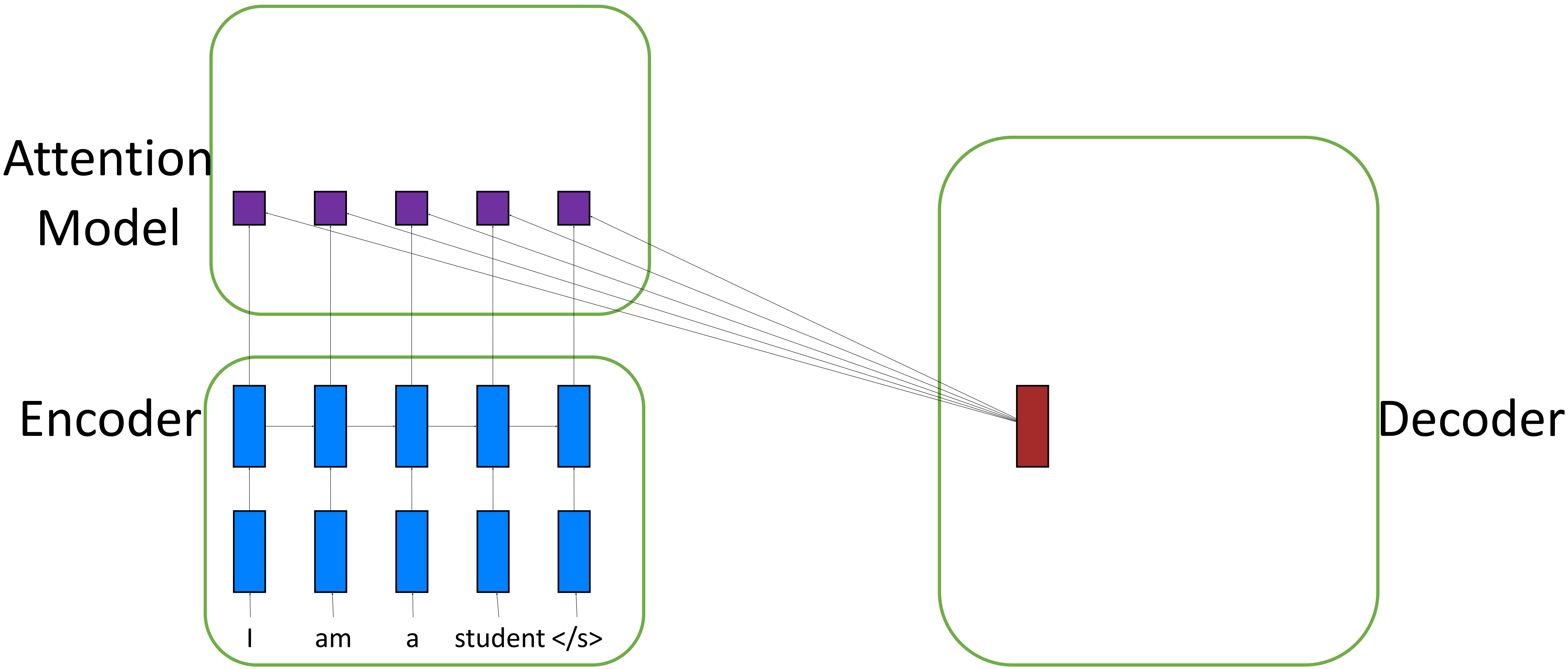

Bahdanau et al. 2014

# Attention MT Models



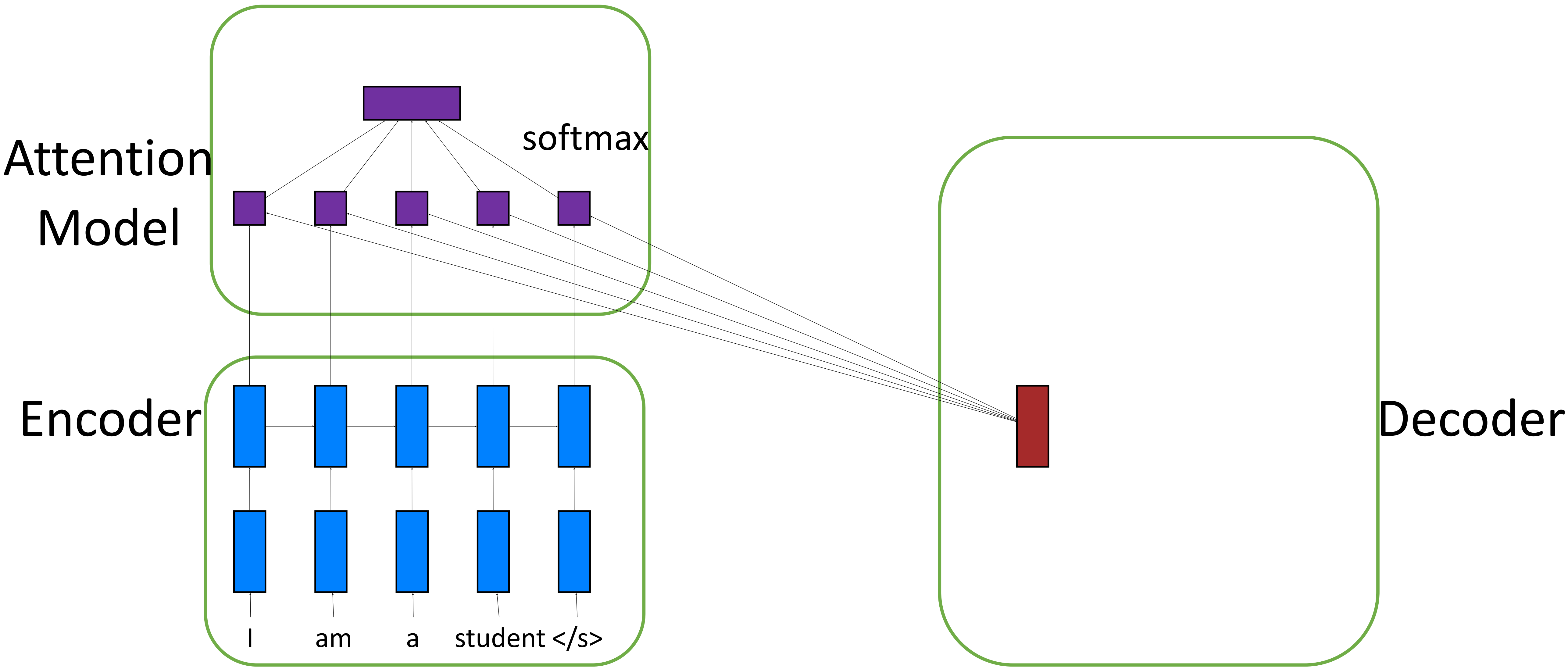

Bahdanau et al. 2014

# Attention MT Models



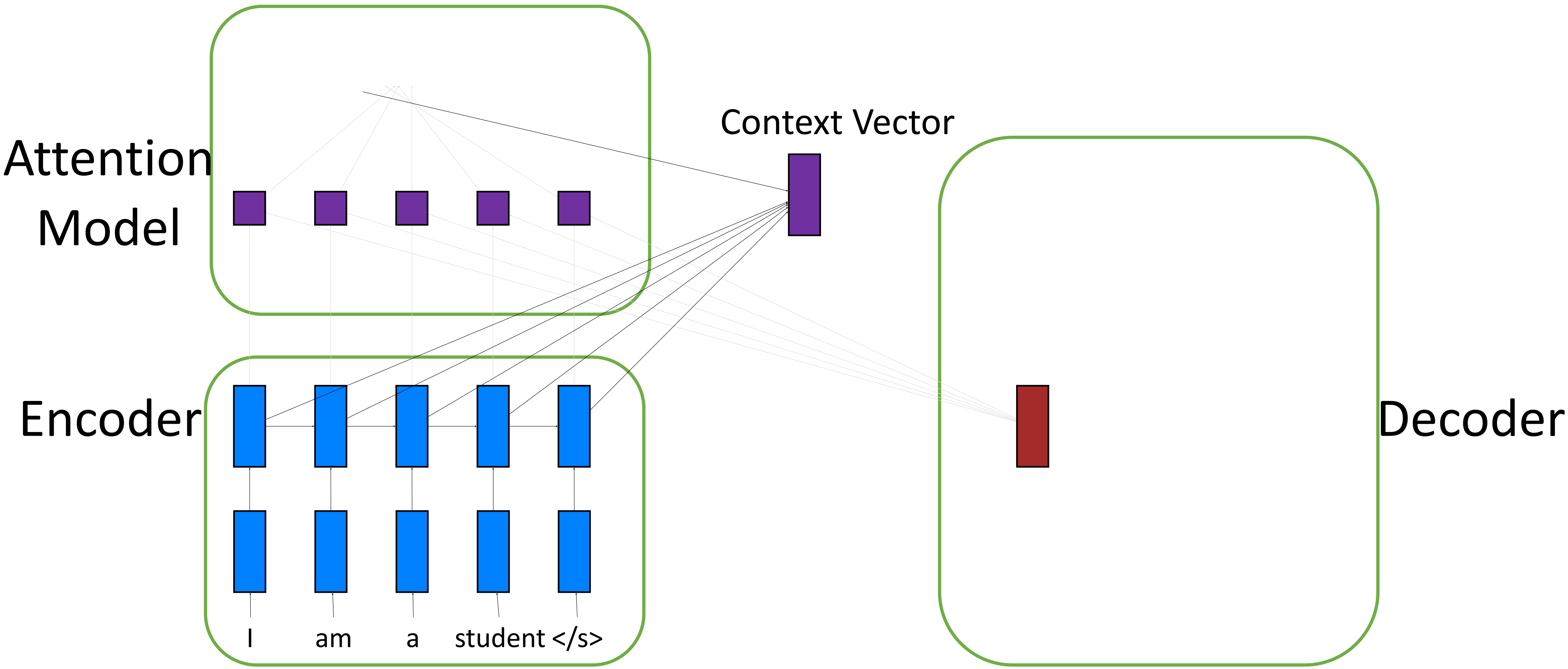

Bahdanau et al. 2014

# Attention MT Models



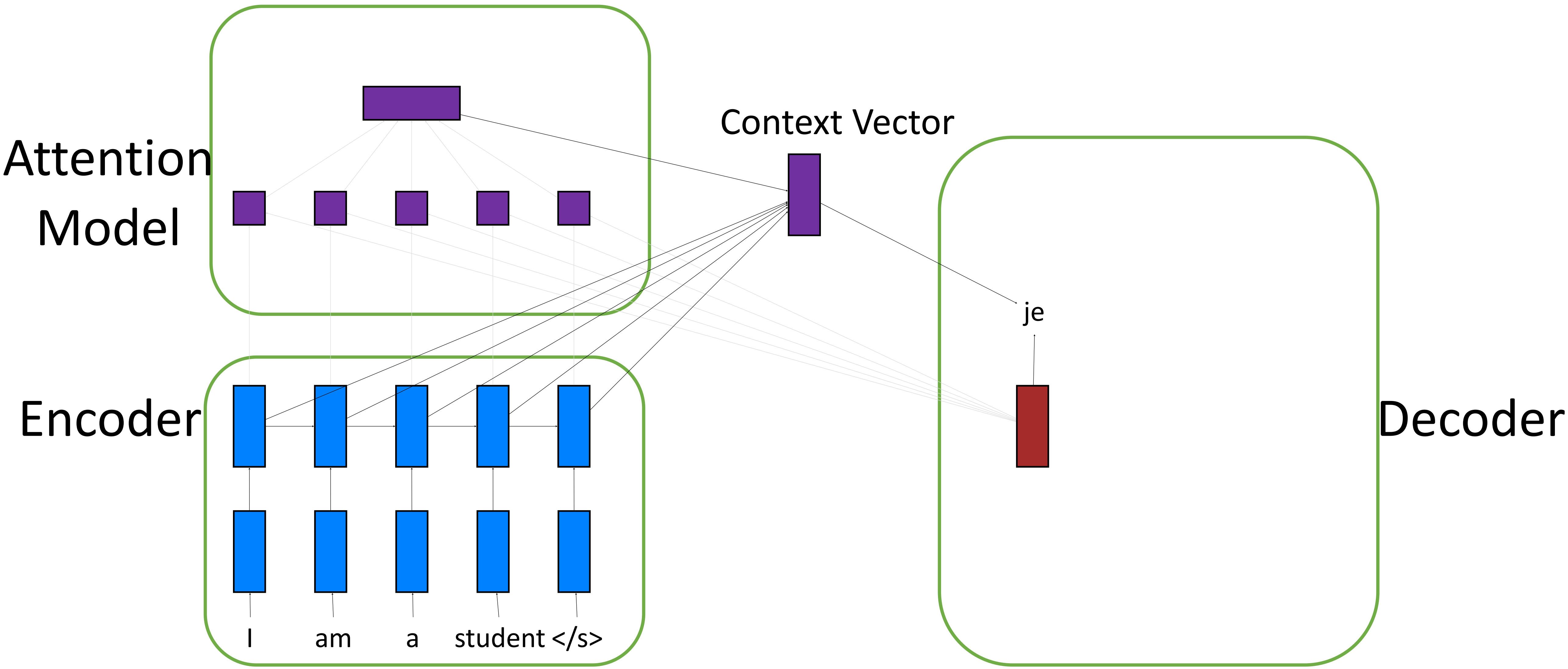

Bahdanau et al. 2014

# Attention MT Models



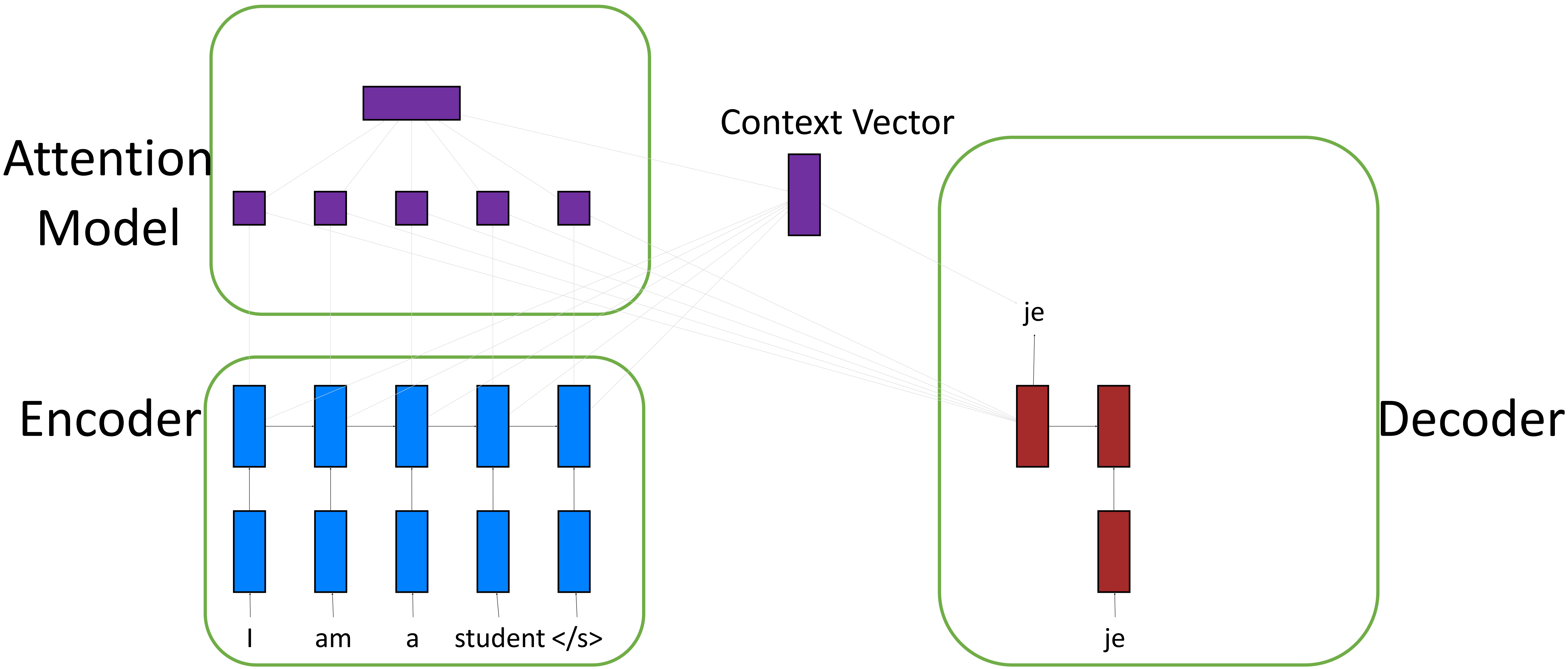

Bahdanau et al. 2014

# Attention MT Models



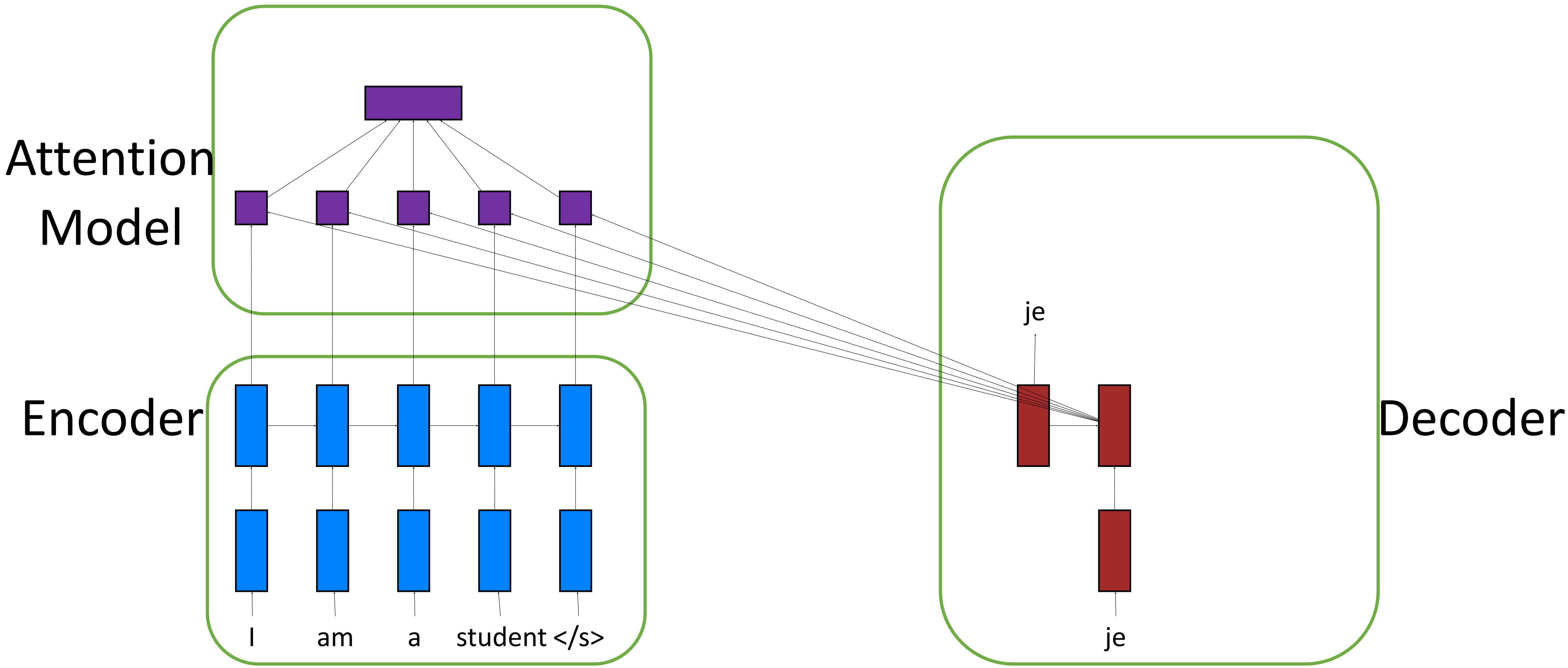

Bahdanau et al. 2014

# Attention MT Models



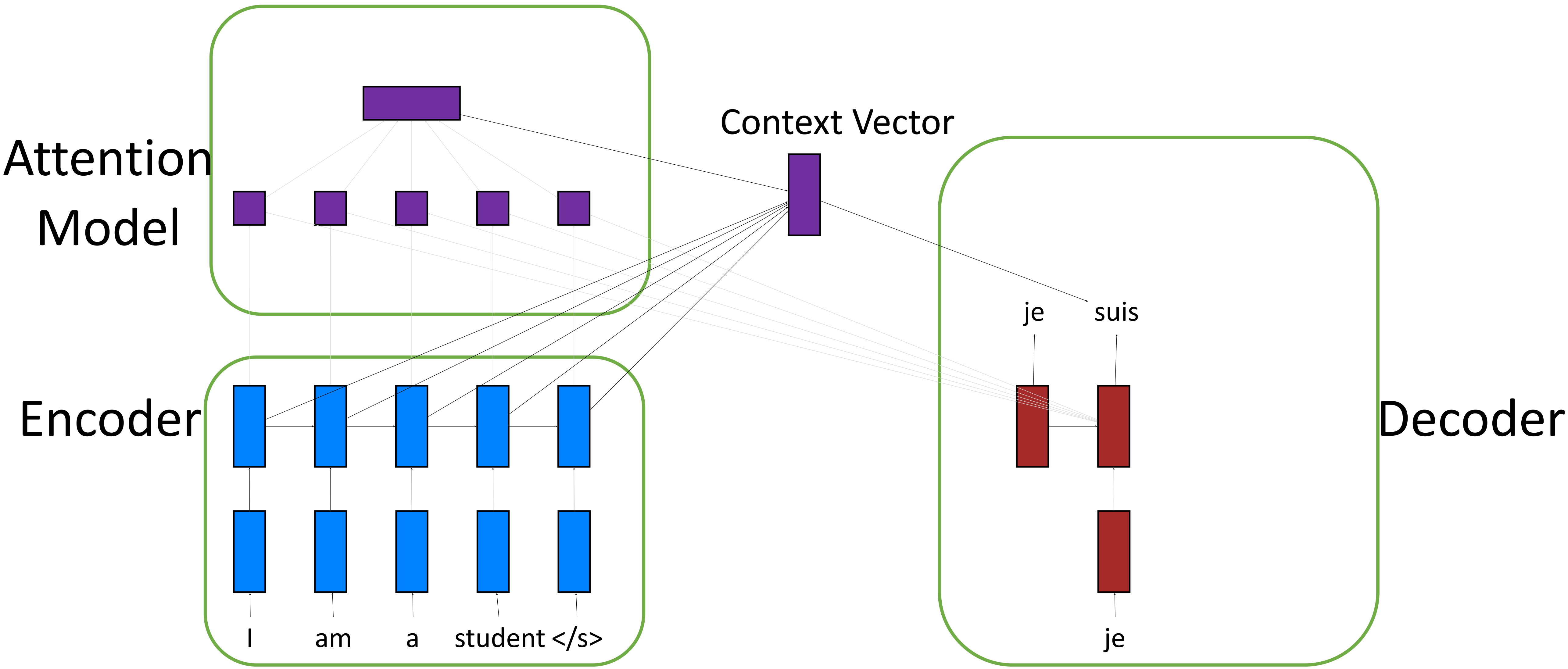

Bahdanau et al. 2014

# Attention MT Models



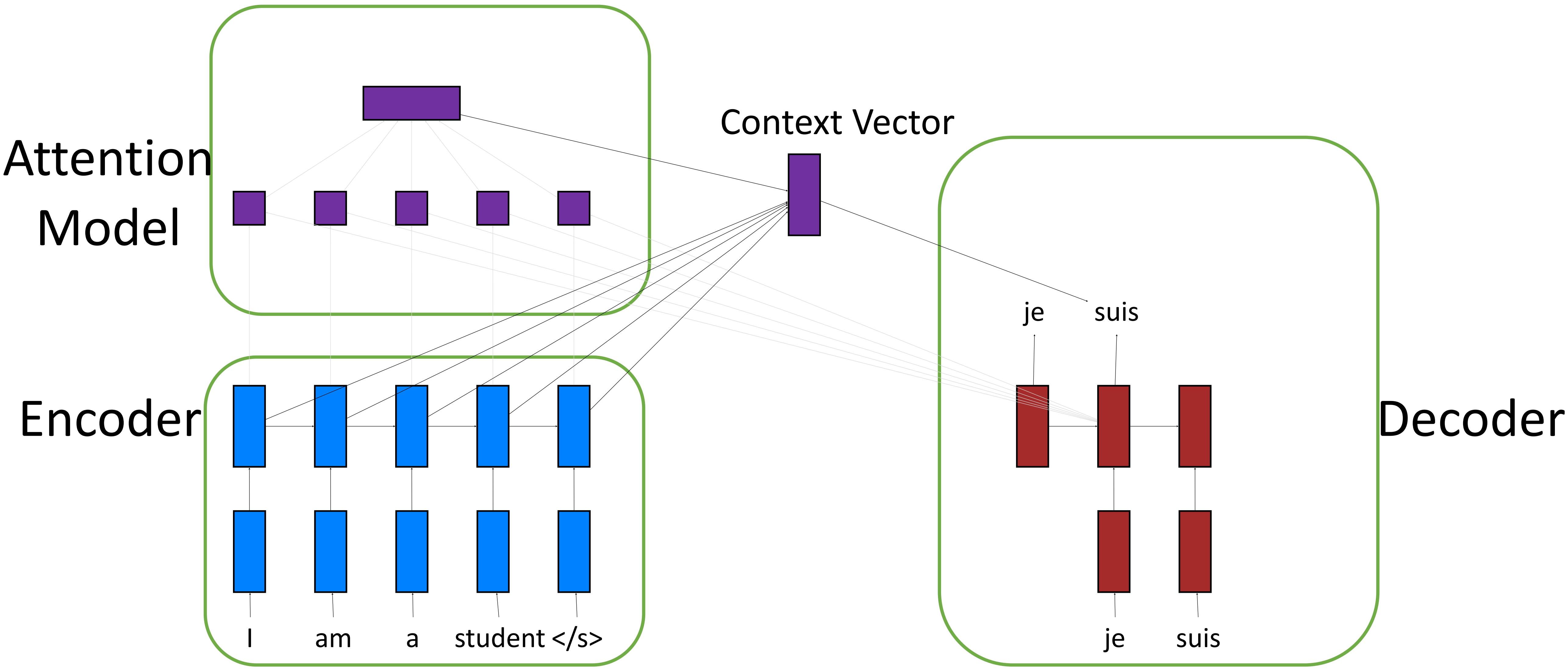

Bahdanau et al. 2014

# Attention MT Models



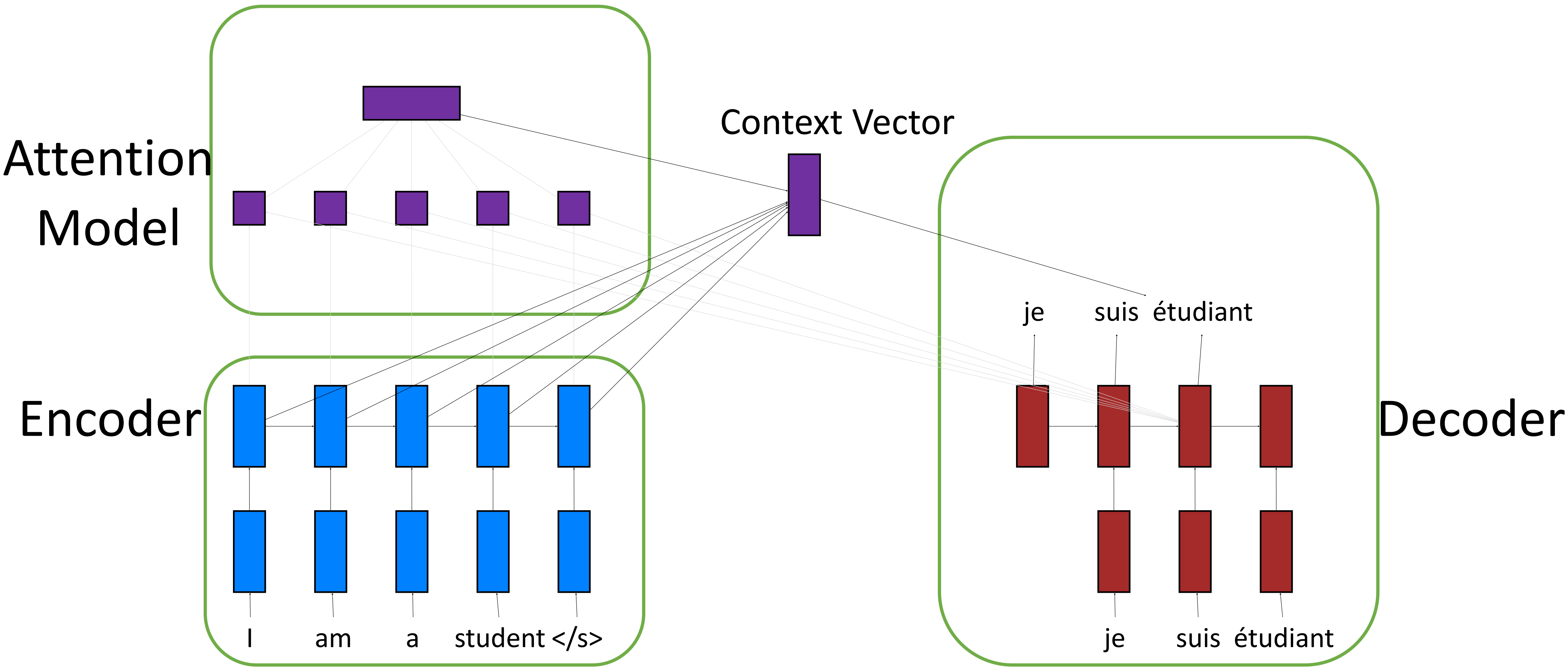

Bahdanau et al. 2014

# Attention MT Models



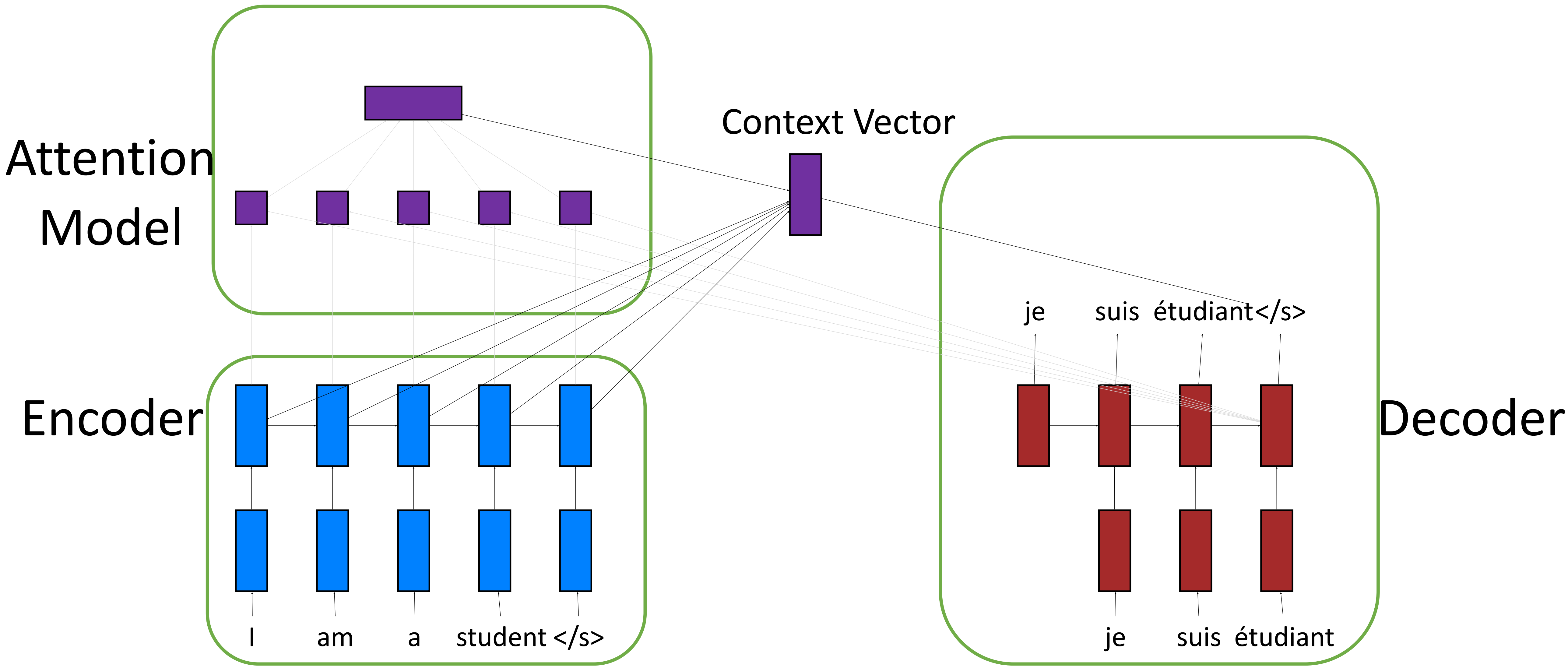

Bahdanau et al. 2014

# Attention MT Models




Bahdanau et al. 2014

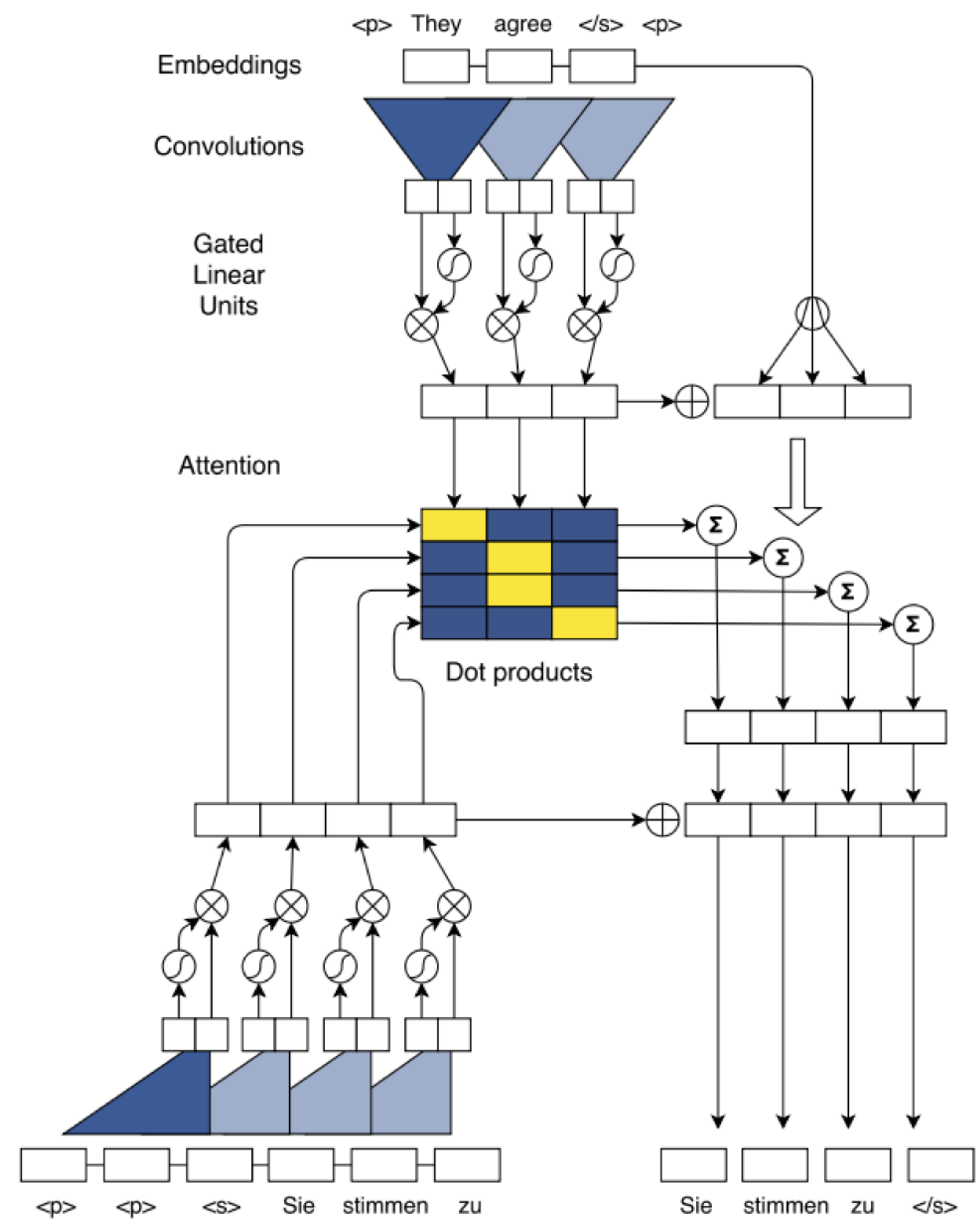
# Attention MT Models




Bahdanau et al. 2014

# Convolutional Encoder-Decoder

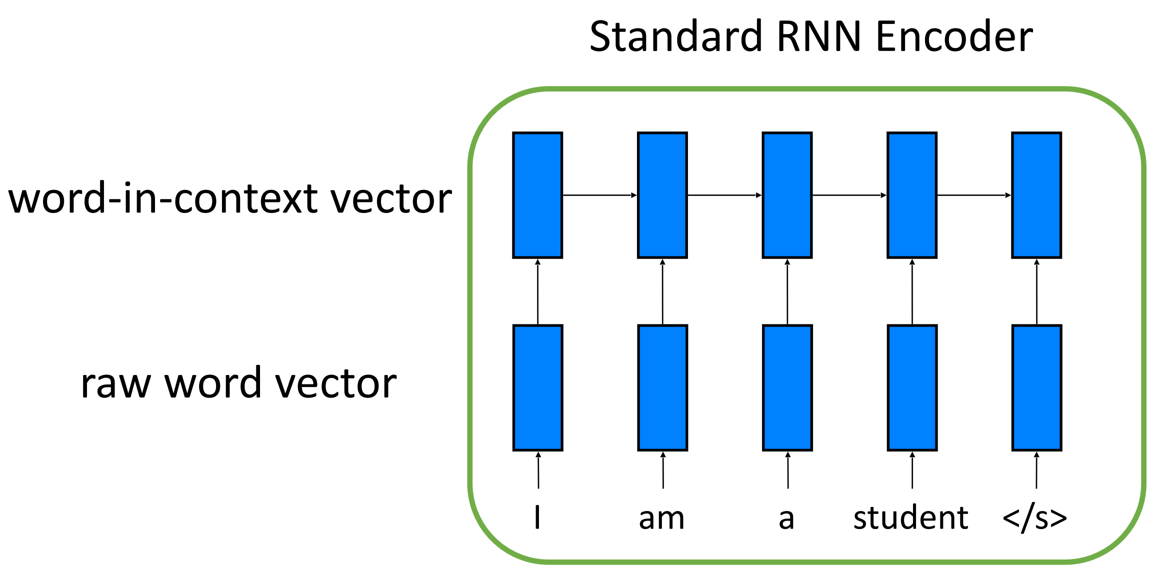
- CNN:
  - encodes words within a fixed size window
  - Parallel computation
  - Shortest path to cover a wider range of words
- RNN:
  - sequentially encode a sentence from left to right
  - Hard to parallelize

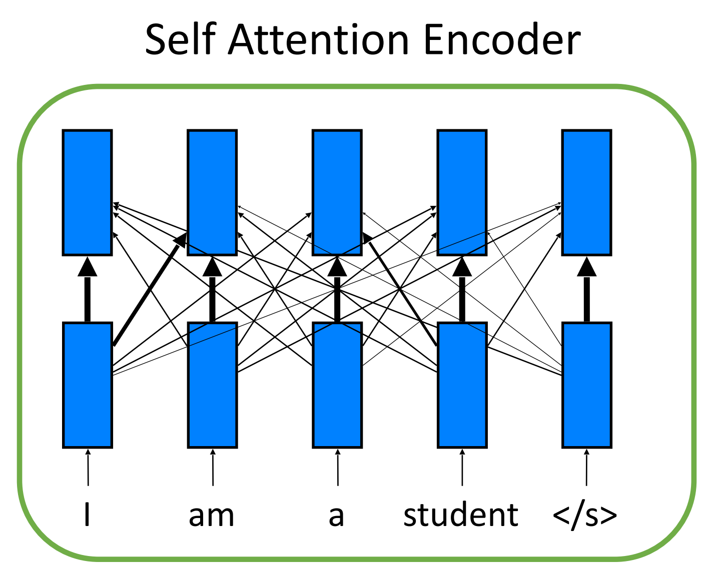



Gehring et al. 2014

# Transformer

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




Vaswani et al. 2017

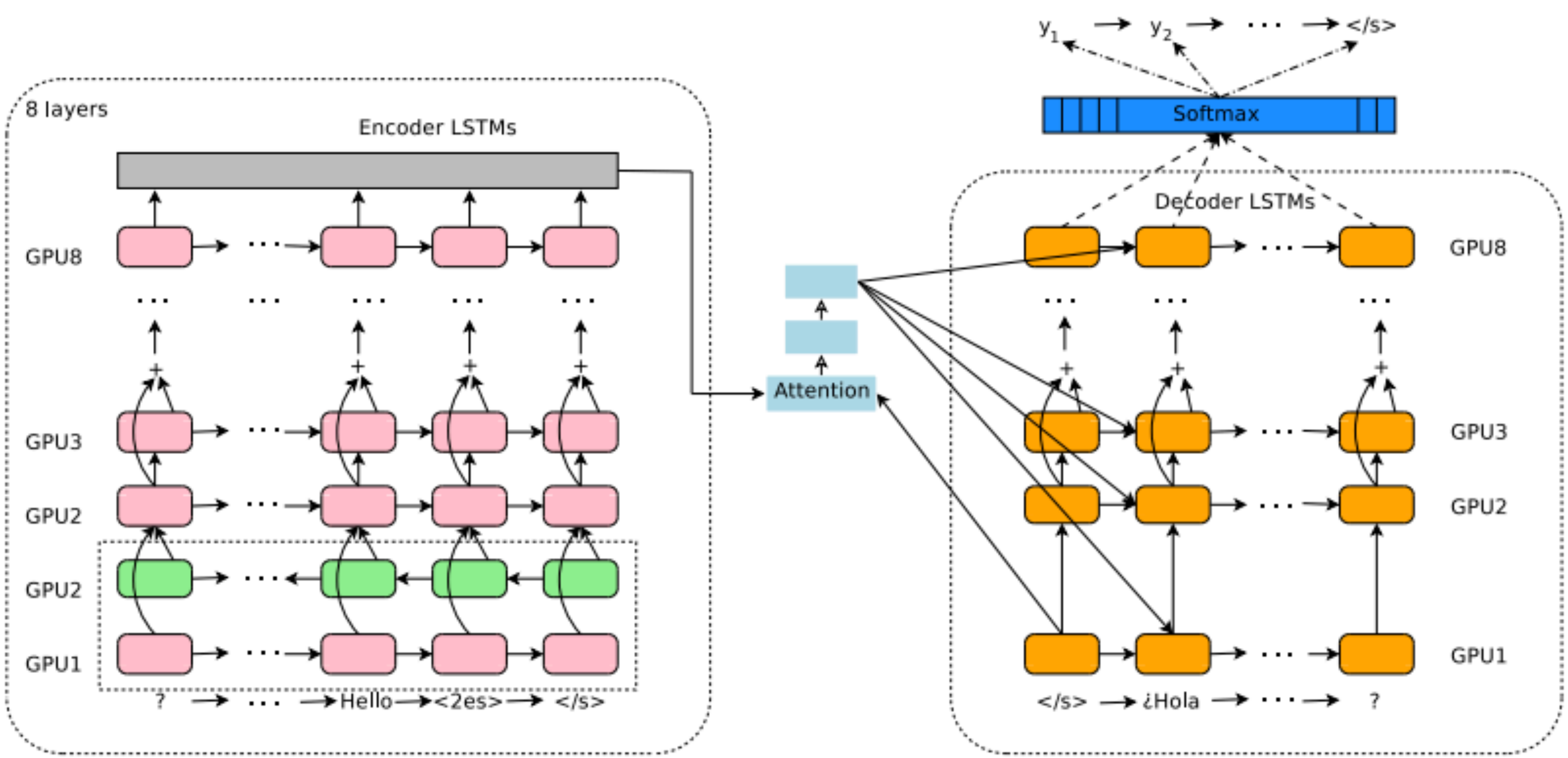
# Transformer

- 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 | **41.29** | $7.7 \cdot 10^{19}$ | $1.2 \cdot 10^{21}$ |
| Transformer (base model) | 27.3 | 38.1 | $\mathbf{3.3 \cdot 10^{18}}$ | |
| Transformer (big) | **28.4** | **41.0** | $2.3 \cdot 10^{19}$ | |

# 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$\rightarrow$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$\rightarrow$Es | no | 31.50 |
| (d) | Model 1 (Pt$\rightarrow$En, En$\rightarrow$Es) | yes | 21.62 |
| (e) | Model 2 (En$\leftrightarrow$\{Es, Pt\}) | yes | 24.75 |
| (f) | Model 2 + incremental training | no | 31.77 |

Johnson et al. 2016

# Google's Multilingual NMT

- Interpolate the language code embeddings

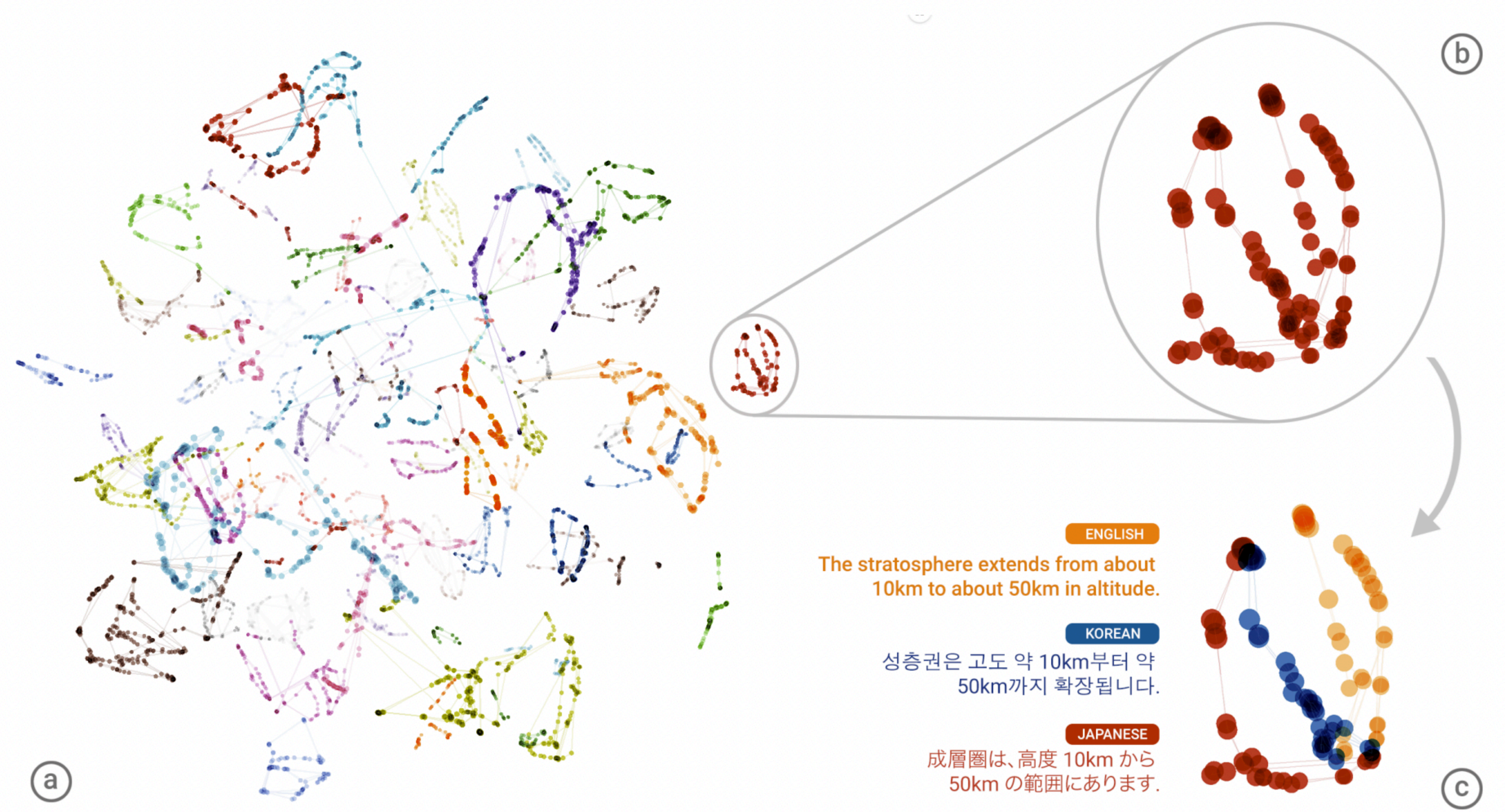
$$(1-w)\text{<2ja>} + w\text{<2ko>}$$

| 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! |
| $w_{be} = 0.30$ | Цікаво, что они будут делать дальше! |
| $w_{be} = 0.44$ | Цікаво, що вони будуть робити далі! |
| $w_{be} = 0.46$ | Цікаво, що вони будуть робити далі! |
| $w_{be} = 0.48$ | Цікаво, што яны зробяць 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の中心의가까이에어딘가에도착하고있어야한다。 |
| $w_{ko} = 0.60$ | 나는지구의센터의가까이에어딘가에도착하고있어야한다。 |
| $w_{ko} = 0.70$ | 나는지구의중심근처어딘가에도착해야합니다。 |
| $w_{ko} = 0.90$ | 나는어딘가지구의중심근처에도착해야합니다。 |
| $w_{ko} = 1.00$ | 나는어딘가지구의중심근처에도착해야합니다。 |

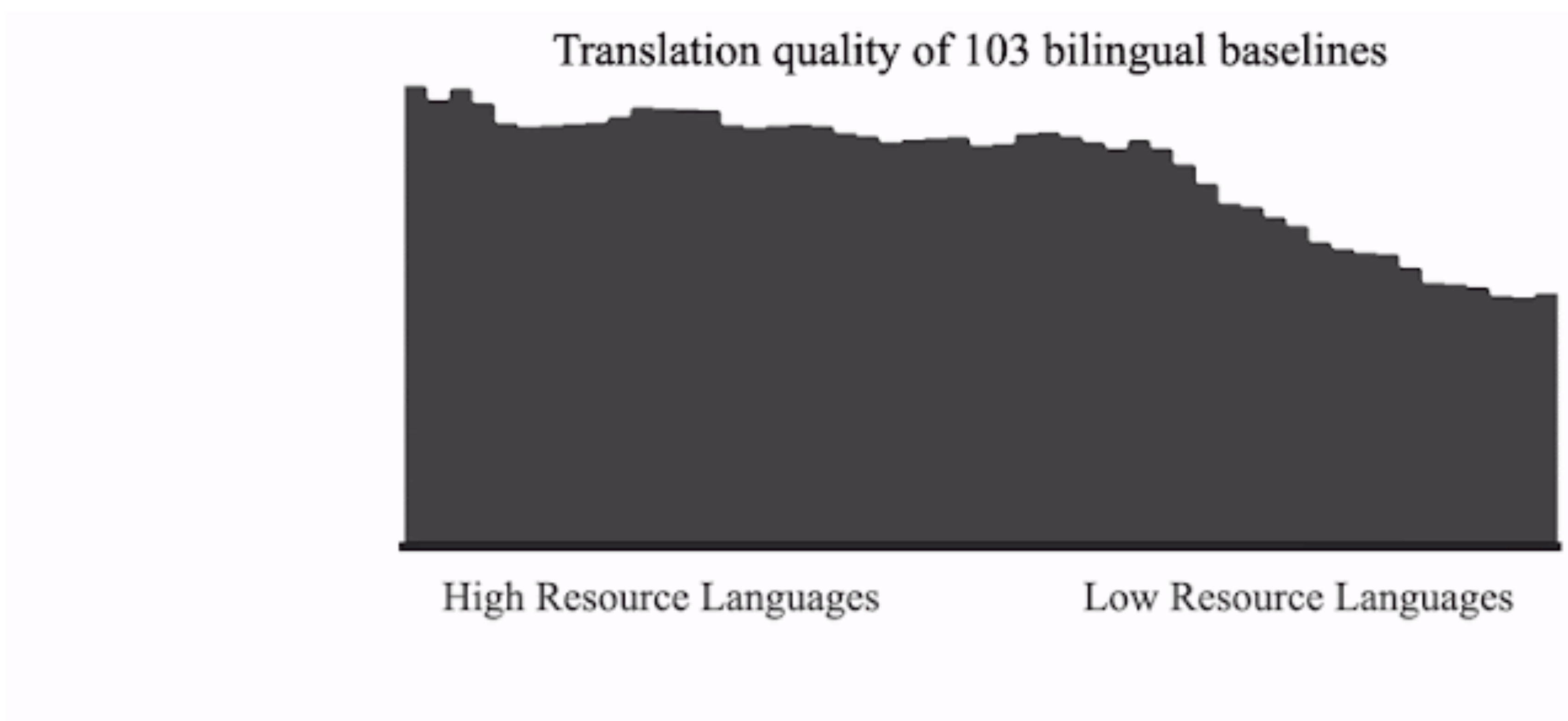
# Google's Multilingual NMT

- Sentence embeddings learned by MNMT are clustered by languages



# Multilingual vs Bilingual

- Multilingual NMT especially improves low-resource language translation



# Future Research of NMT

# 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.

Koehn & Knowles 2016

# 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)

Arivazhagan et al. 2019

# Questions?