Machine Translation Overview

April 23, 2020

Junjie Hu

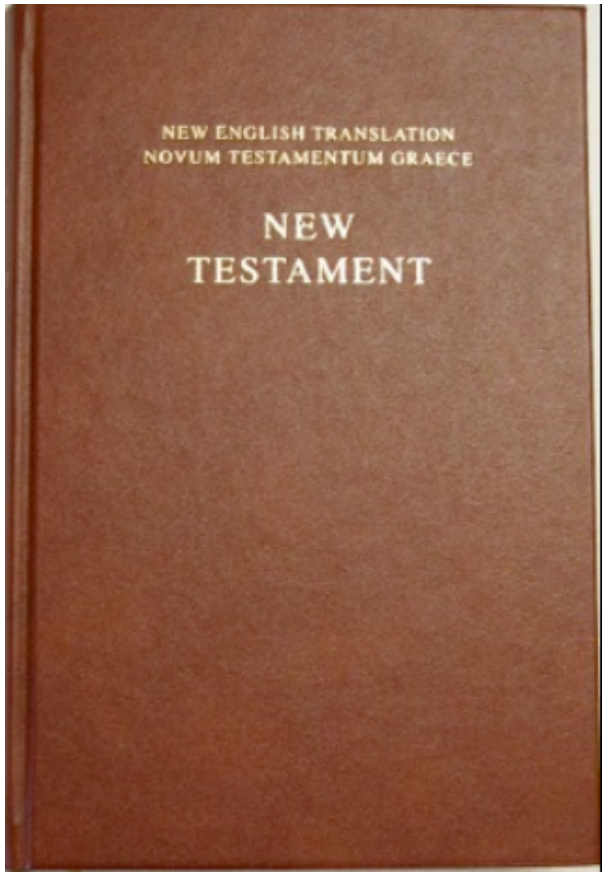
Materials largely borrowed from Austin Matthews

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.'***

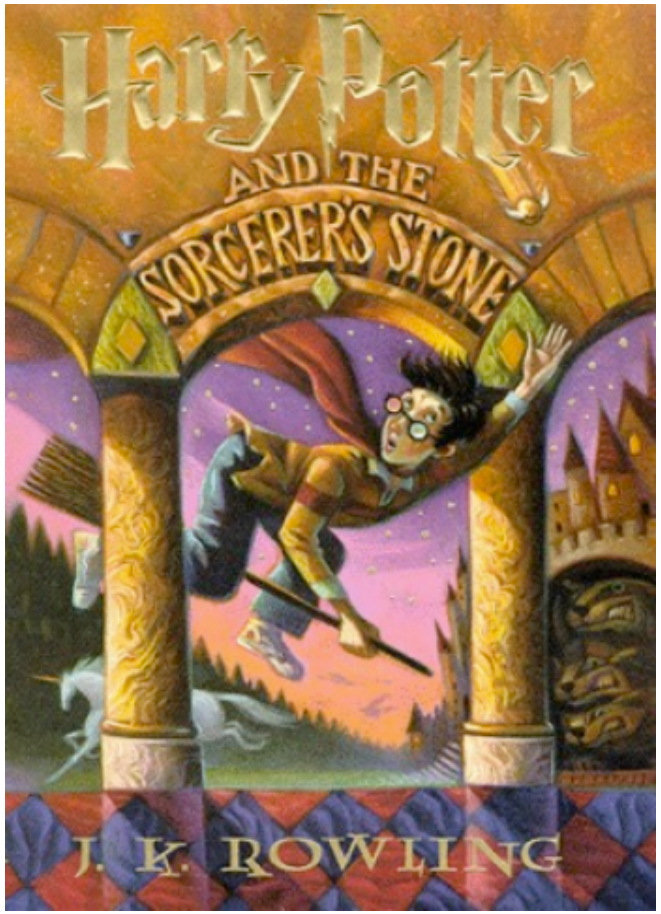
Warren Weaver to Norbert Wiener, March, 1947

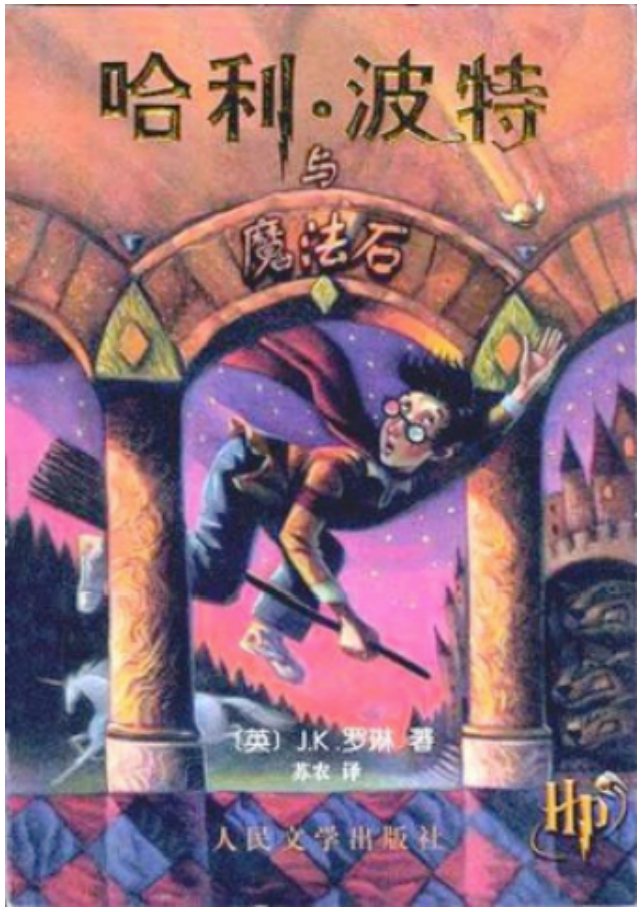

www.un.org
http://www.un.org/english/
Google
We the peoples
Daily Briefing | Radio, TV, Photo | Documents, Maps | Publications, Stamps, Databases | UN Works | Search
Peace & Security | Economic & Social Development | Human Rights | Humanitarian Affairs | International Law
Welcome to the
United Nations
UN Millennium Development Goals
United Nations News Centre
About the United Nations
Main Bodies
Conferences & Events
Member States
General Assembly President
Secretary-General
Situation in Iraq
Mideast Roadmap
Renewing the UN
UN Action against Terrorism
Issues on the UN Agenda
Civil Society / Business
UN Webcast
CyberSchoolBus
8 September 2005 >>
Home | Recent Additions | Employment | UN Procurement | Comments | Q & A | UN System Sites | Index
عربي | 中文 | English | Français | Русский | Español
Copyright, United Nations, 2000-2005 | Use of UN60 Logo | Terms of Use | Privacy Notice | Help
[Text version]
Live and On-Demand Webcasts, 24 Hours a Day: Click on UN Webcas


联合国主页
http://www.un.org/chinese/
Google
我们人民
每日简报 | 多媒体 | 文件与地图 | 出版物 邮票 数据库 | 服务全球 | 网址搜索
和平与安全 | 经济与社会发展 | 人权 | 人道主义事务 | 国际法
欢迎来到
联合国
联合国千年发展目标
联合国新闻
联合国概况
联合国主要机关
会议与活动
联合国会员国
联合国大会主席
联合国秘书长
伊拉克局势
中东路线图
更新联合国
反恐怖主义
联合国日常议题
民间团体/商业
联合国网络直播
空中校车
联大第60届会议一般性辩论
新增内容 | 工作机会 | 联合国采购 | 建议 | 问题与解答 | 其他网址 | 网址索引
عربي | 中文 | English | Français | Русский | Español
联合国2000-2005年版权|联合国60周年徽标使用准则|使用条件|隐私通告|帮助
[纯文字版]


NEW ENGLISH TRANSLATION
NOVUM TESTAMENTUM GRAECE
NEW
TESTAMENT


聖經
■
新譯本


Harry Potter
AND THE
SORCERER'S STONE
J. K. ROWLING


哈利·波特
与
魔法石
(英) J.K.罗琳 著
苏农 译
人民文学出版社
Hp

Parallel corpus

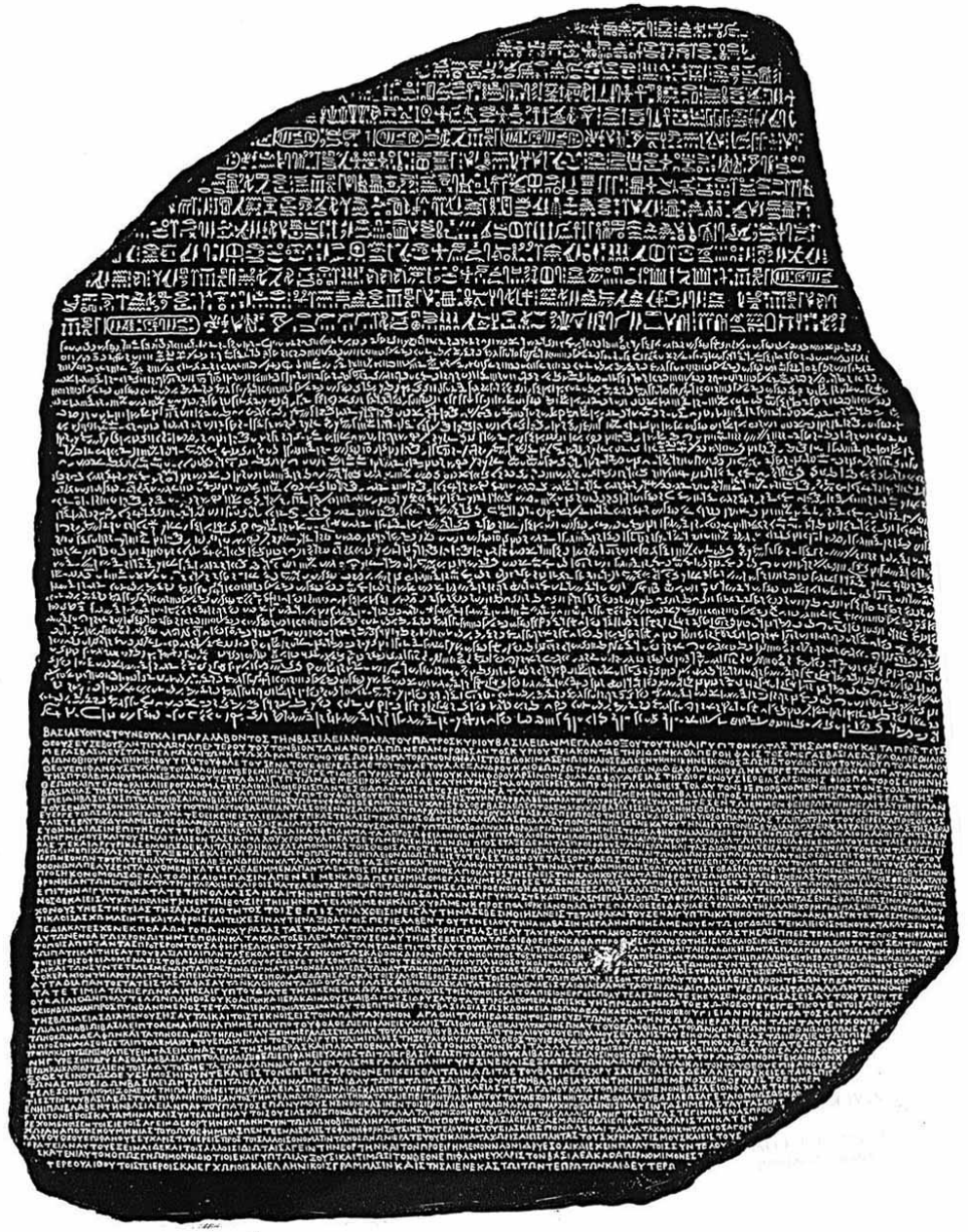
- We are given a corpus of sentence pairs in two languages to train our machine translation models.
- Source language is also called foreign language, denoted as f.
- Conventionally target language is usually referred to English.

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

Egyptian

Greek



Noisy Channel MT

We want a model of $p(\boldsymbol{e}|\boldsymbol{f})$

Noisy Channel MT

We want a model of $p(\boldsymbol{e}|\boldsymbol{f})$

Confusing foreign sentence

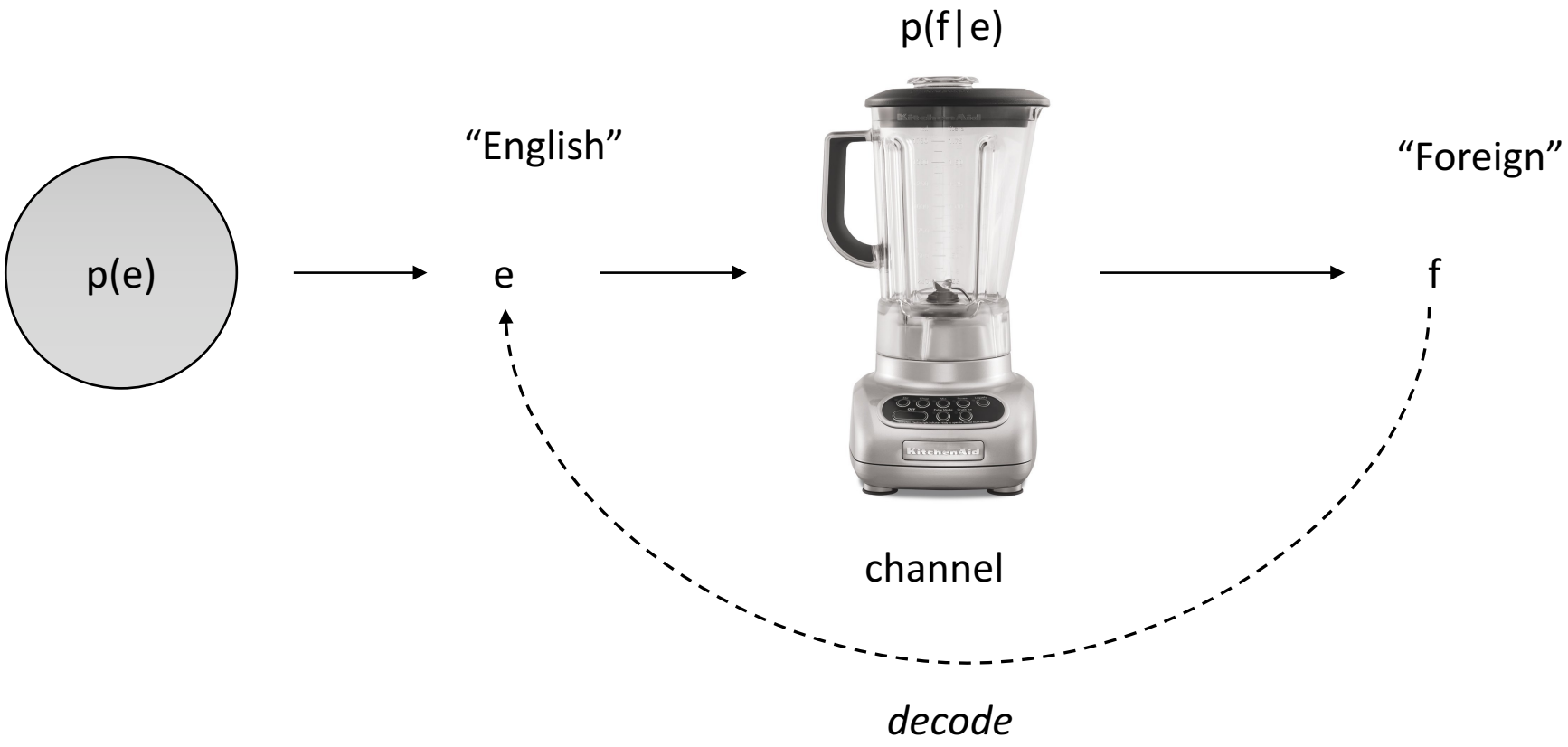
Noisy Channel MT

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

Confusing foreign sentence

Possible English translation

Noisy Channel MT



Noisy Channel MT

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

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

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

“Language Model” **“Translation Model”**

Noisy Channel Division of Labor

- Language model – $p(\mathbf{e})$
  - is the translation fluent, grammatical, and idiomatic?
  - use any model of $p(\mathbf{e})$ – typically an *n*-gram model
- Translation model – $p(\mathbf{f} | \mathbf{e})$
  - "reverse" translation probability
  - ensures adequacy of translation

Language Model Failure

My legal name is Alexander Perchov.

Language Model Failure

My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always spleening her.

Language Model Failure

My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always spleening her. If you want to know why I am always spleening her, it is because I am always elsewhere with friends, and disseminating so much currency, and performing so many things that can spleen a mother.

Translation Model

- $p(\boldsymbol{f}|\boldsymbol{e})$ gives the channel probability – the probability of translating an English sentence into a foreign sentence
- $\boldsymbol{f}$ = je voudrais un peu de frommage — $p(\boldsymbol{f}|\boldsymbol{e})$
- $\mathbf{e}_1$ = I would like some cheese — 0.4

  $\mathbf{e}_2$ = I would like a little of cheese — 0.5

  $\mathbf{e}_3$ = There is no train to Barcelona — >0.00001

Translation Model

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

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

- There are a lot of possible sentences (closed to infinite number):
  - We can only count the sentences in our training data
  - this won't generalize to new inputs

Lexical Translation

- How do we translate a word? Look it up in a dictionary!

  Haus: house, home, shell, household

- Multiple translations
  - Different word senses, different registers, different inflections
  - *house, home* are common
  - *shell* is specialized (the Haus of a snail is its shell)

How common is each translation?

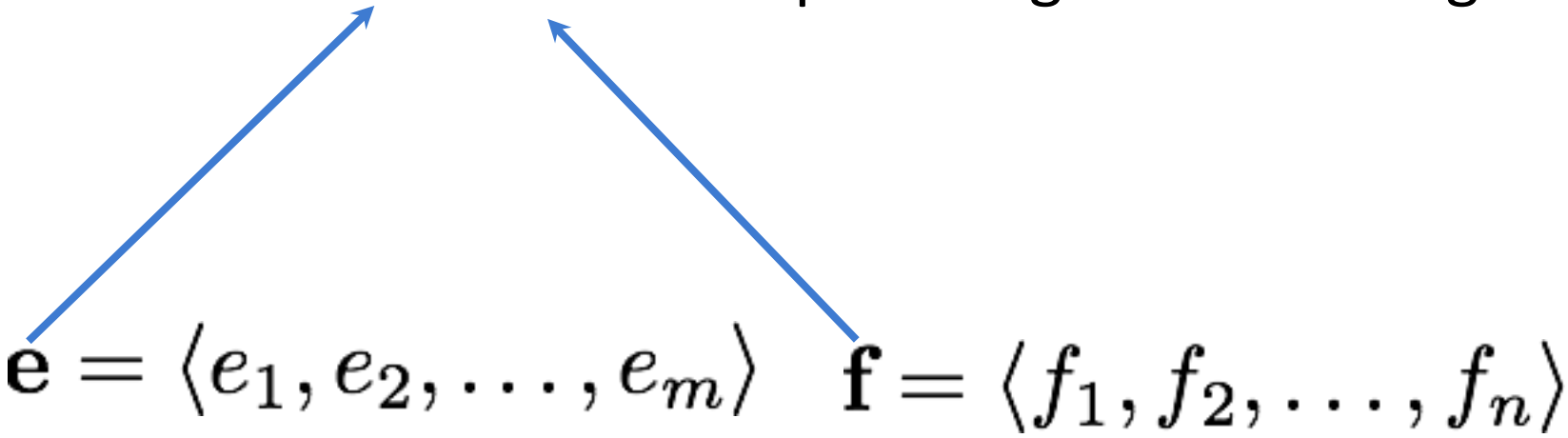
Translation	Count
house	5000
home	2000
shell	100
household	80

Maximum Likelihood Estimation (MLE)

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

Lexical Translation

- Goal: a model *p(**e**|**f**,m)*
- where **e** and **f** are complete English and Foreign sentences



$$\mathbf{e} = \langle e_1, e_2, \ldots, e_m \rangle \quad \mathbf{f} = \langle f_1, f_2, \ldots, f_n \rangle$$

Lexical Translation

- Goal: a model $p(\mathbf{e}|\mathbf{f},m)$
- where **e** and **f** are complete English and Foreign sentences
- Lexical translation makes the following ***assumptions***:
  - Each word $\mathbf{e}_i$ in **e** is generated from exactly one word in **f**
  - Thus, we have a latent *alignment* $\mathbf{a}_i$ that indicates which word $\mathbf{e}_i$ "came from." Specifically it came from $\mathbf{f}_{\mathbf{a}_i}$.
  - Given the alignments **a**, translation decisions are conditionally independent of each other and depend *only* on the aligned source word $\mathbf{f}_{\mathbf{a}_i}$.

Lexical Translation

- Putting our assumptions together, we have:

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

p(Alignment) p(Translation | Alignment)

where a is an m-dimensional latent vector with each element a_i in the range of [0,n].

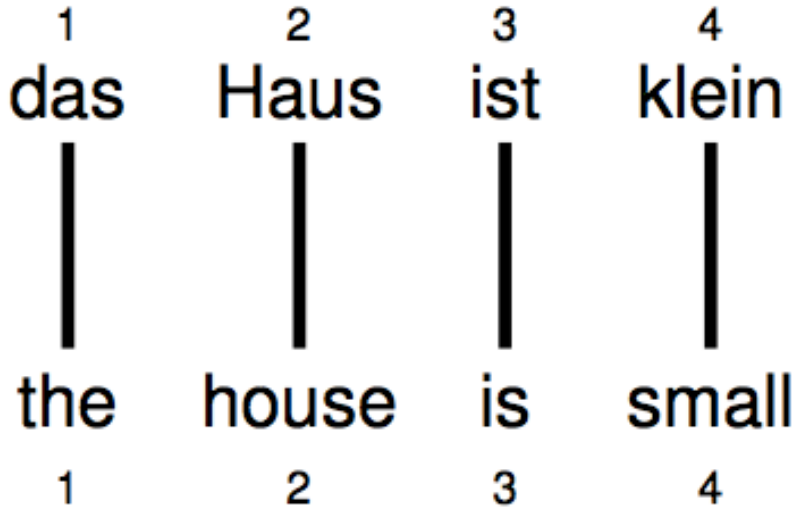
Word Alignment

$$p(\mathbf{a} \mid \mathbf{f}, m)$$

- Most of the research for the first 10 years of SMT was here. Word translations weren't the problem. Word *order* was hard.

Word Alignment

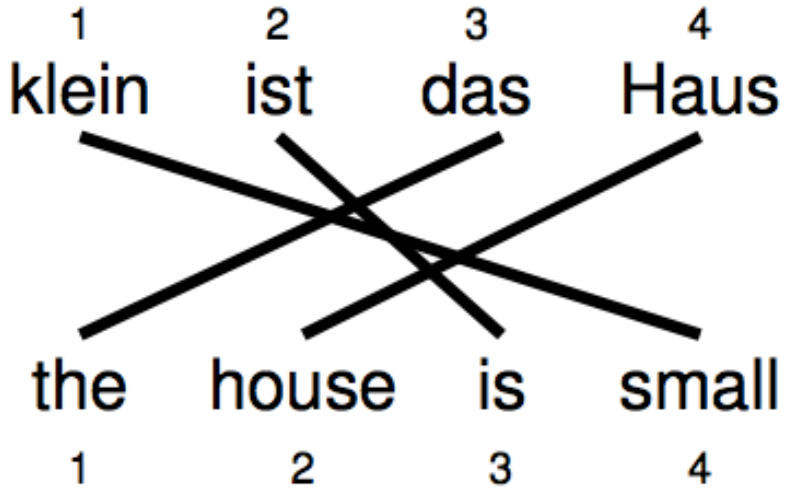
- Alignments can be visualized by drawing links between two sentences, and they are represented as vectors of positions:



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

Reordering

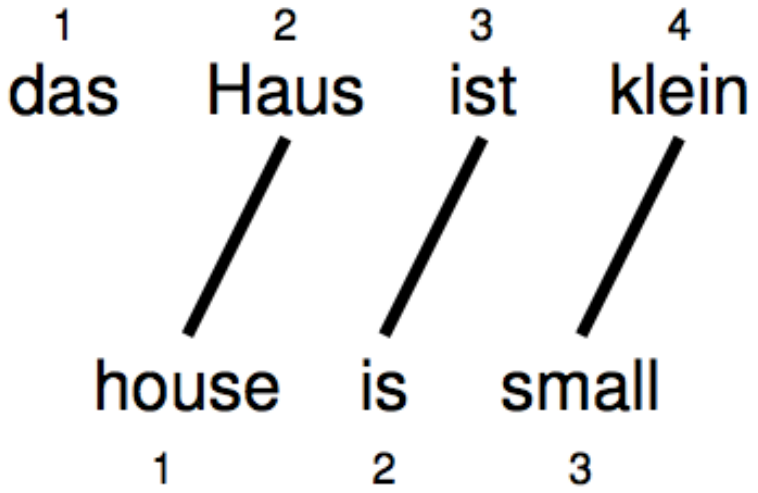
- Words may be reordered during translation



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

Word Dropping

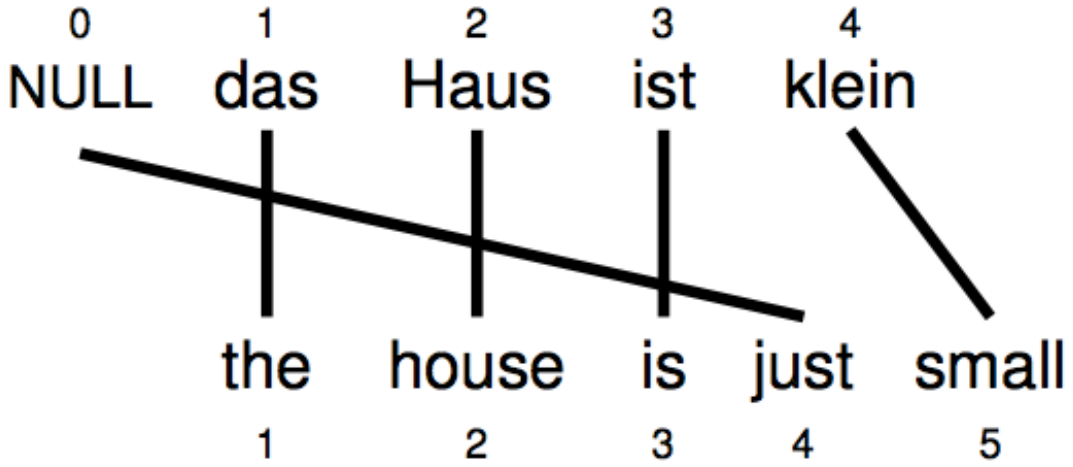
- A source word may not be translated at all



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

Word Insertion

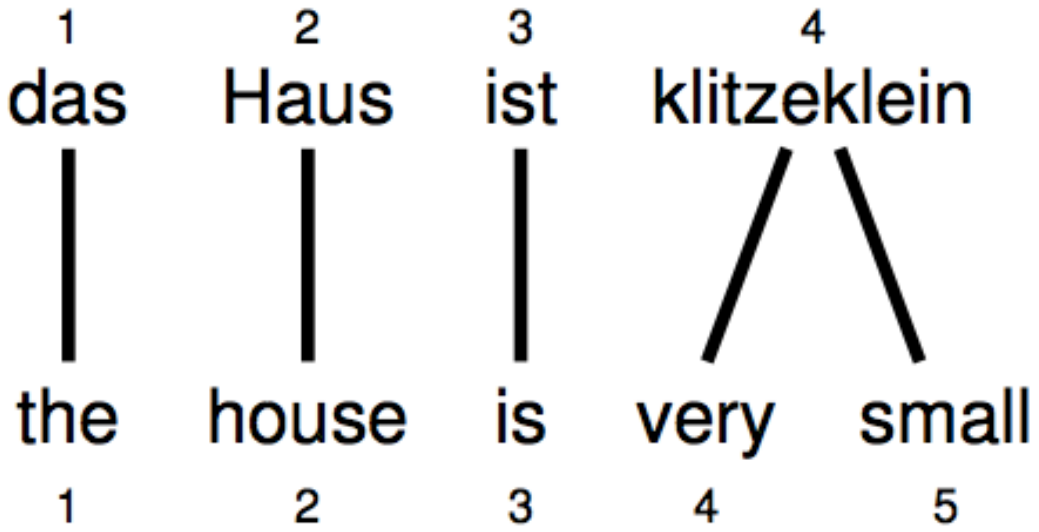
- Words may be inserted during translation
- E.g. English just does not have an equivalent
- But these words must be explained – we typically assume every source sentence contains a NULL token



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

One-to-many Translation

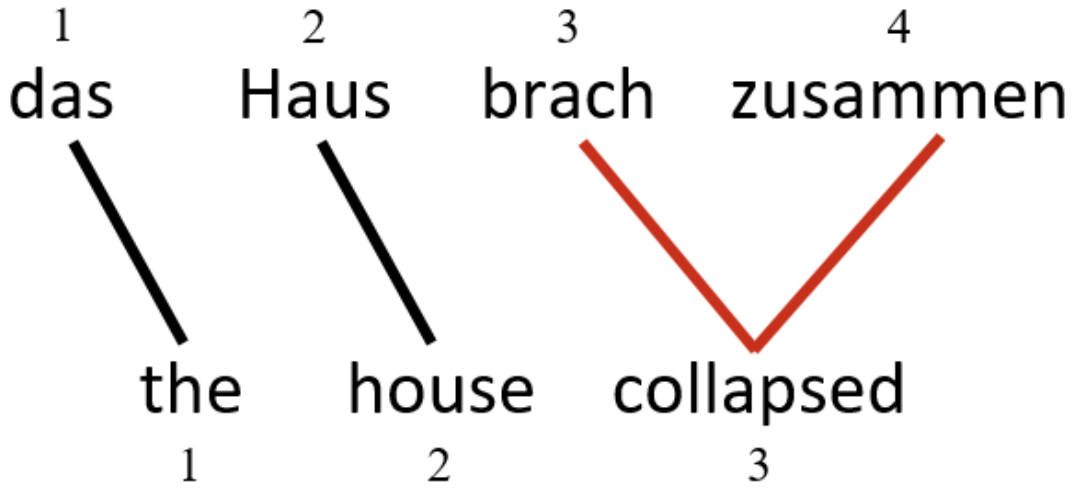
- A source word may translate into **more than one** target word



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

Many-to-one Translation

- More than one source word may **not** translate as a unit in lexical translation



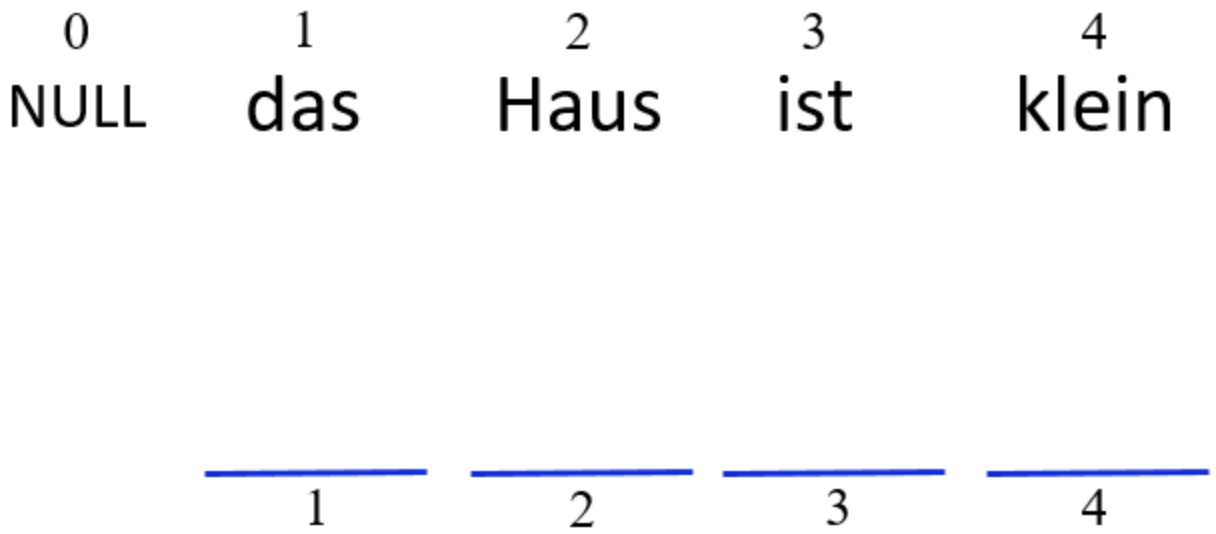
$\mathbf{a} = ???$ $\mathbf{a} = (1, 2, (3, 4)^\top)^\top$?

IBM Model 1

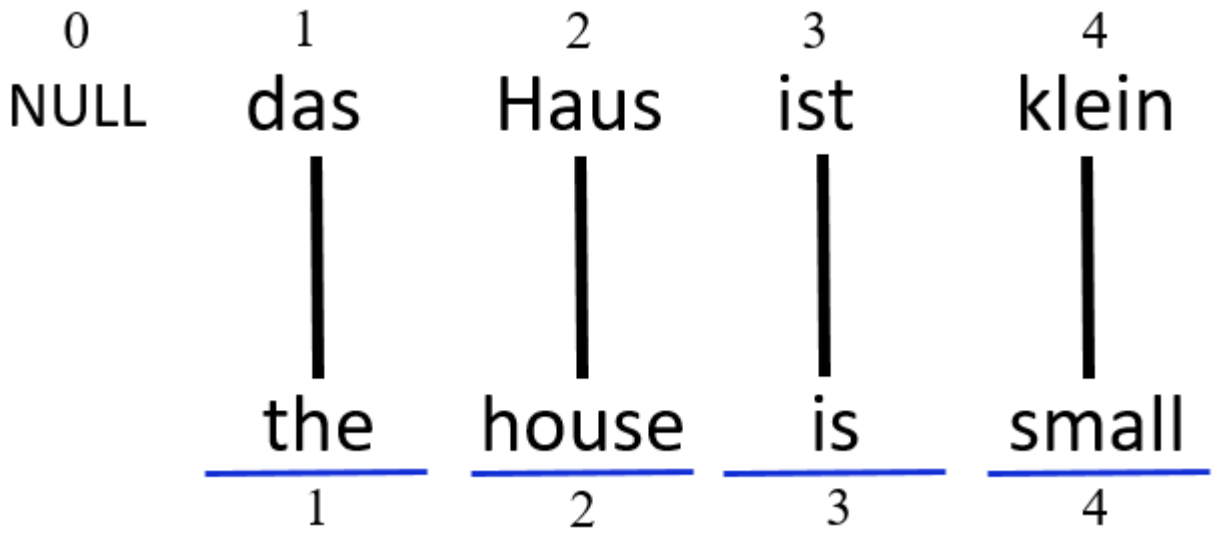
- Simplest possible lexical translation model
- Additional assumptions:
  - The *m* alignment decisions are independent
  - The alignment distribution for each $\mathbf{a}_i$ is uniform over all source words and NULL

$$\begin{aligned}
&\text{for each } i \in [1, 2, \ldots, m] \\
&\qquad a_i \sim \text{Uniform}(0, 1, 2, \ldots, n) \\
&\qquad e_i \sim \text{Categorical}(\boldsymbol{\theta}_{f_{a_i}})
\end{aligned}$$

Translating with Model 1

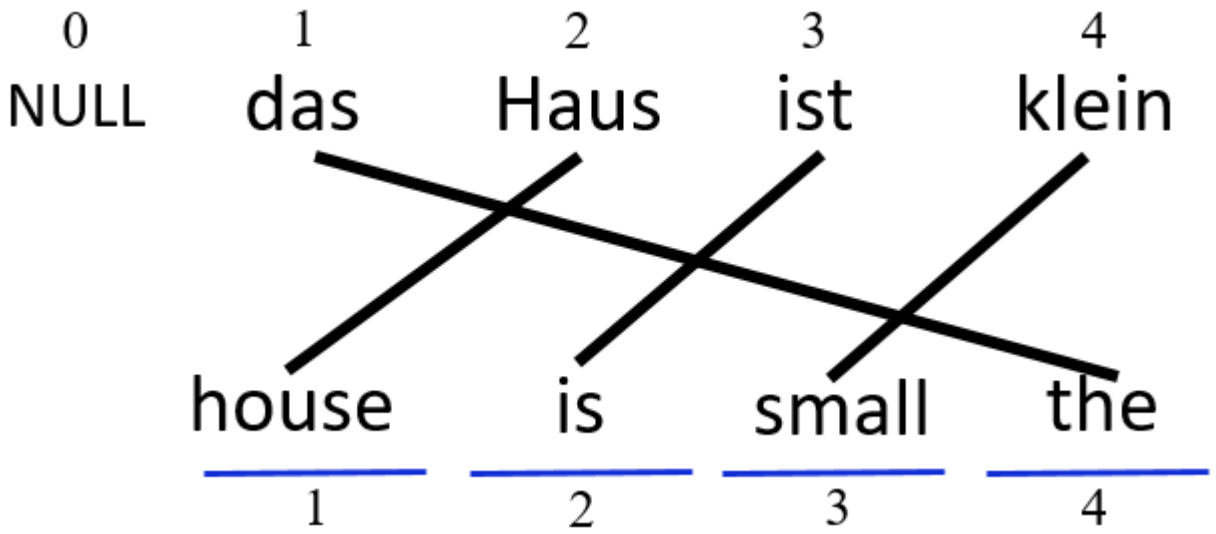


Translating with Model 1



Language model says: ☺

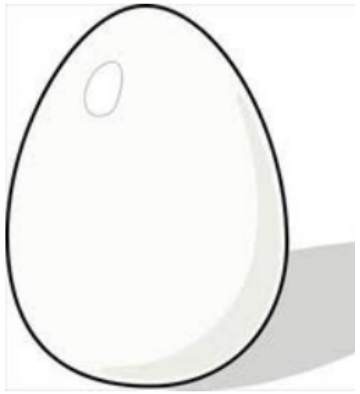
Translating with Model 1



Language model says: ☹

Learning Lexical Translation Models

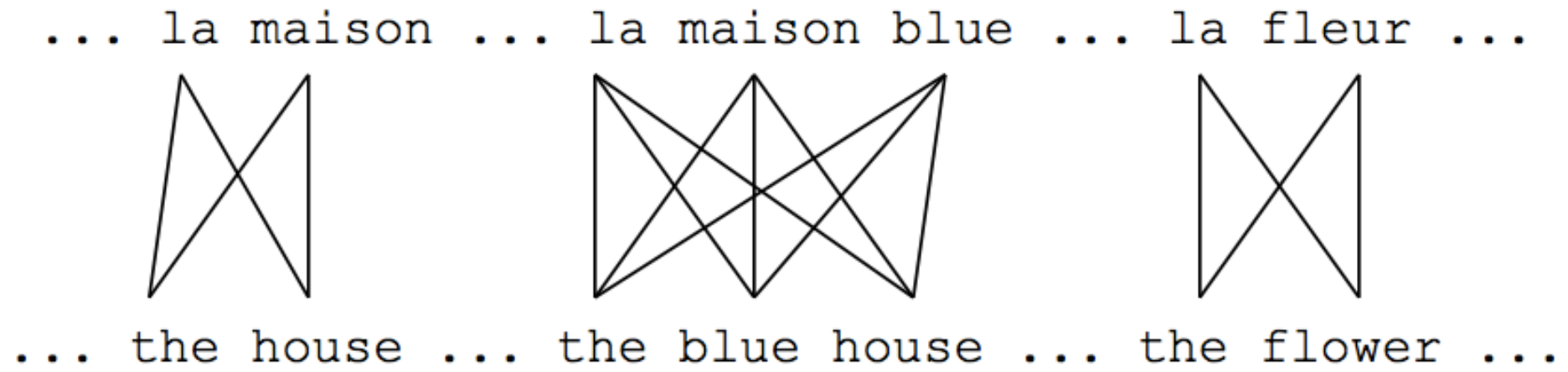
- How do we learn the parameters *p(e|f) on the training corpus of (f, e) sentence pairs*?
- "Chicken and egg" problem
  - If we had the alignments, we could estimate the translation probabilities (MLE estimation)
  - If we had the translation probabilities we could find the most likely alignments (greedy)



Expectation-Maximization (EM) Algorithm

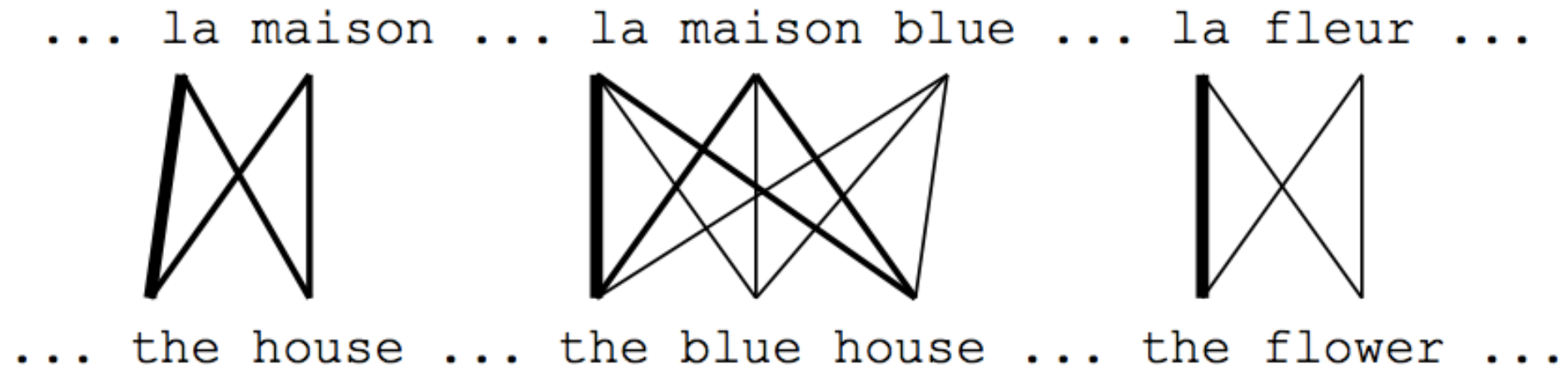
- Pick some random (or uniform) starting parameters
- Repeat until bored (~5 iterations for lexical translation models):
  - Using the current parameters, compute "expected" alignments $p(\mathbf{a}_i | \mathbf{e}, \mathbf{f})$ for every target word token in the training data
  - Keep track of the expected number of times *f* translates into *e* throughout the whole corpus
  - Keep track of the number of times *f* is used in the source of any translation
  - Use these frequency estimates in the standard MLE equation to get a better set of parameters

EM for IBM Model 1



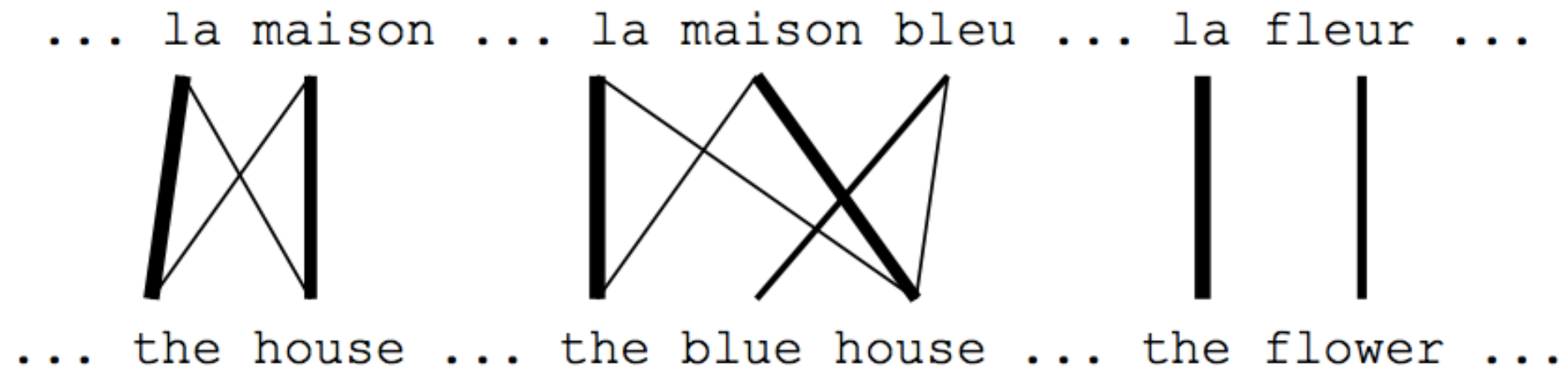
- Initial step: all alignments equally likely
- Model learns that, e.g., la is often aligned with the

EM for Model 1



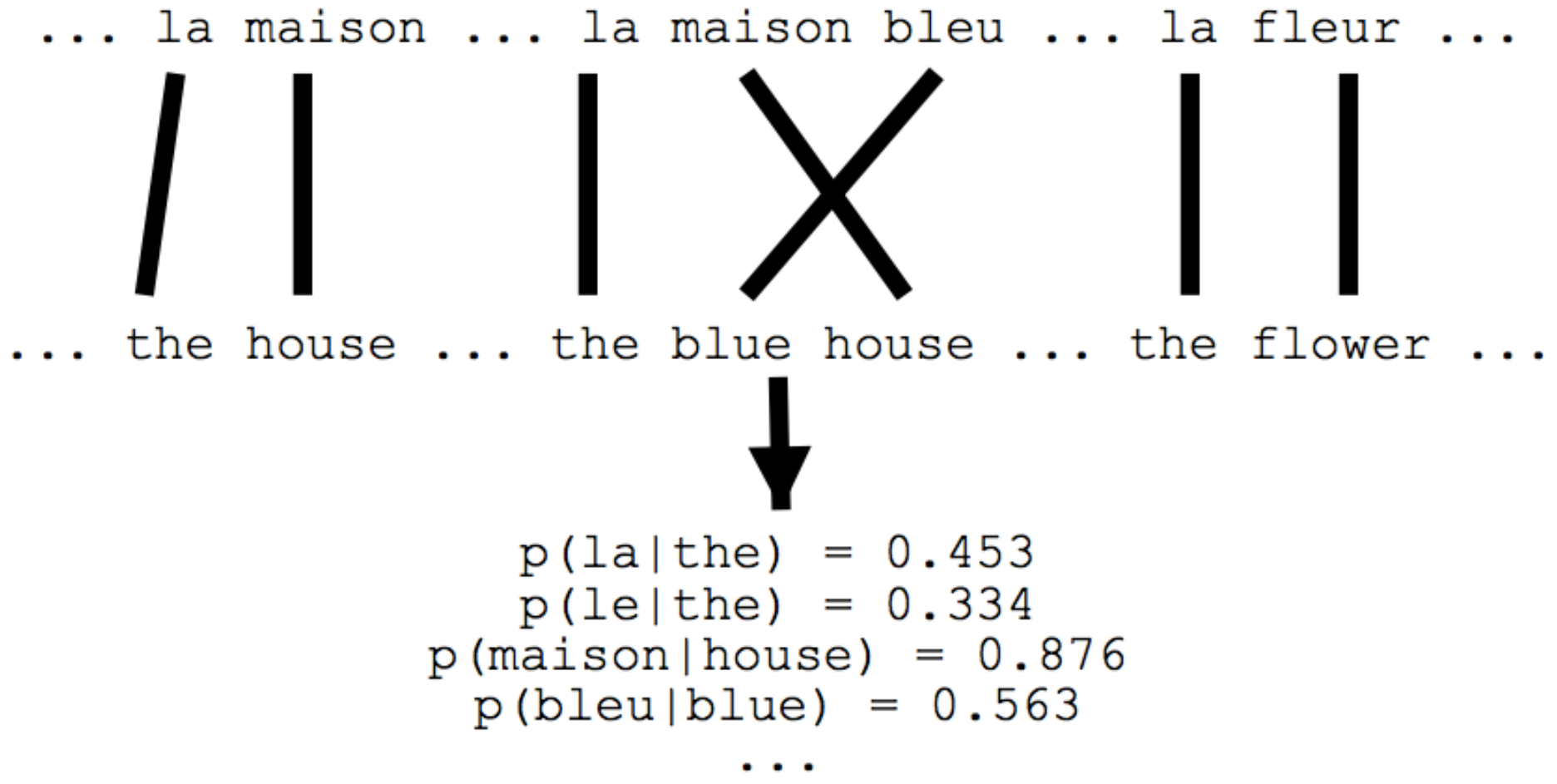
- After one iteration
- Alignments, e.g., between la and the are more likely

EM for Model 1



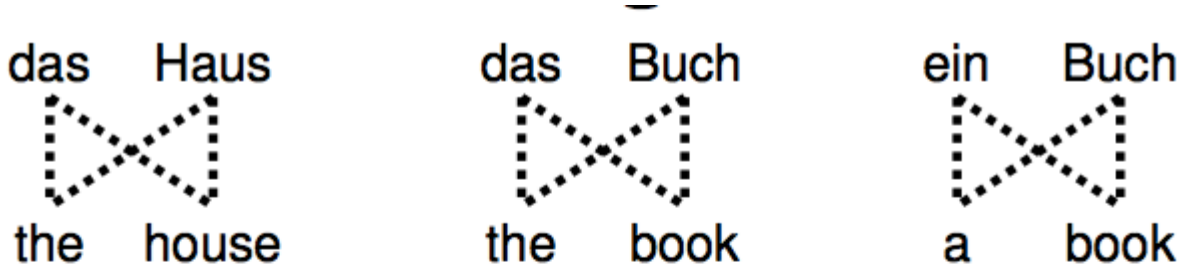
- After another iteration
- It becomes apparent that alignments, e.g., between fleur and flower are more likely (pigeon hole principle)

EM for Model 1



- Parameter estimation from the aligned corpus

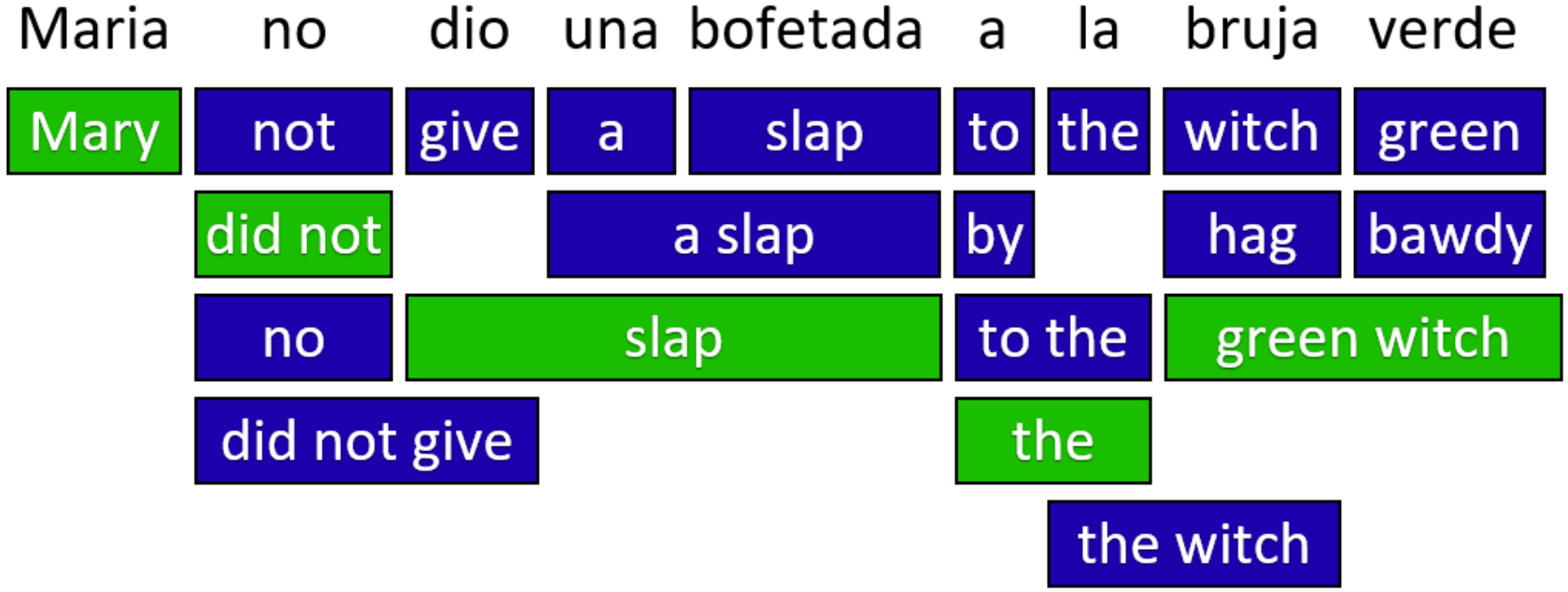
Convergence



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

Extensions: Lexical to Phrase Translation

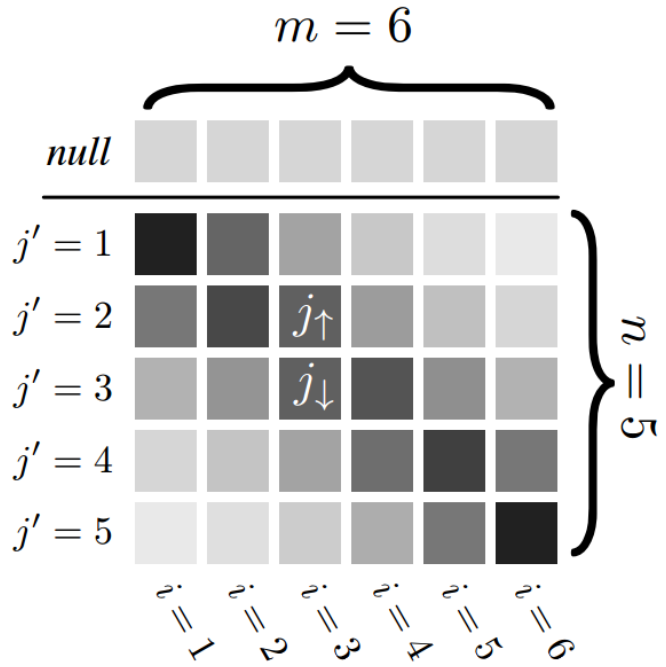
- Phrase-based MT:
  - Allow multiple words to translate as chunks (including many-to-one)
  - Introduce another latent variable, the source *segmentation*



Adapted from Koehn (2006)

Extensions: Alignment Heuristics

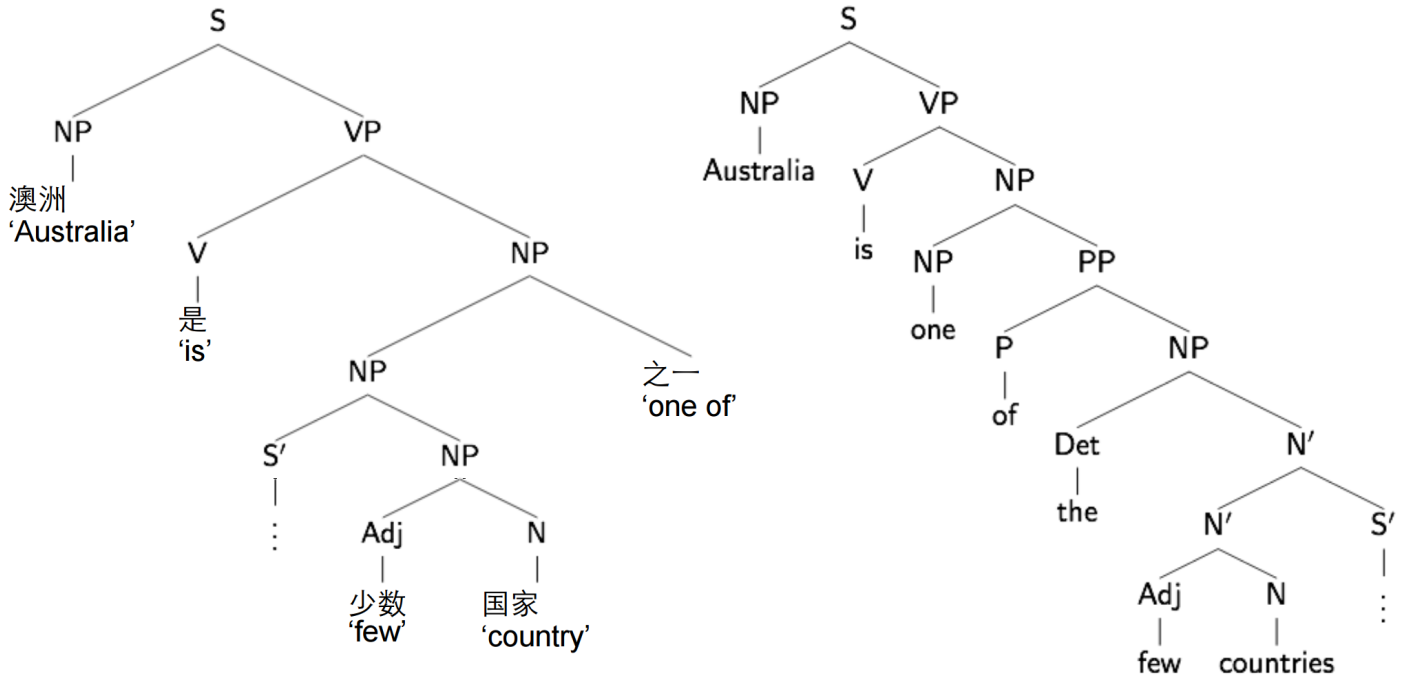
- Alignment Priors:
  - Instead of assuming the alignment decisions are uniform, impose (or learn) a prior over alignment grids:



Chahuneau et al. (2013)

Extensions: Hierarchical Phrase-based MT

- Syntactic structure
- Rules of the form:
- X之一 → one of the X



Chang (2005), Galley et al. (2006)

MT Evaluation

- How do we evaluate translation systems' output?
- Central idea: "The closer a machine translation is to a professional human translation, the better it is."
- Most commonly used metric is called BLEU, which is the geometric mean of the n-gram precision against the human translations plus a length penalty term.

BLEU: An Example

Candidate 1: *It is a guide to action which ensures that the military always obey the commands of the party.*

Reference 1: *It is a guide to action that ensures that the military will forever heed Party commands.*

Reference 2: *It is the guiding principle which guarantees the military forces always being under the command of the Party.*

Reference 3: *It is the practical guide for the army always to heed directions of the party.*

Unigram Precision : 17/18

Adapted from slides by Arthur Chan

Issue of N-gram Precision

- What if some words are over-generated?
  - e.g. “the”
- An extreme example

Candidate: *the the the the the the the.*

Reference 1: *The cat is on the mat.*

Reference 2: *There is a cat on the mat.*

- N-gram Precision: 7/7
- **Solution: reference word should be exhausted after it is matched.**

Adapted from slides by Arthur Chan

Issue of N-gram Precision

- What if some words are just dropped?
- Another extreme example

Candidate: *the.*

Reference 1: *My mom likes the blue flowers.*

Reference 2: *My mother prefers the blue flowers.*

- N-gram Precision: 1/1
- **Solution: add a penalty if the candidate is too short.**

Adapted from slides by Arthur Chan

BLEU

Geometric Average

$$\text{BLEU} = (p_1 \cdot p_2 \cdot p_3 \cdot p_4)^{\frac{1}{4}} \max(1,\ e^{1-\frac{r}{c}})$$

Clipped N-gram precisions for N=1, 2, 3, 4

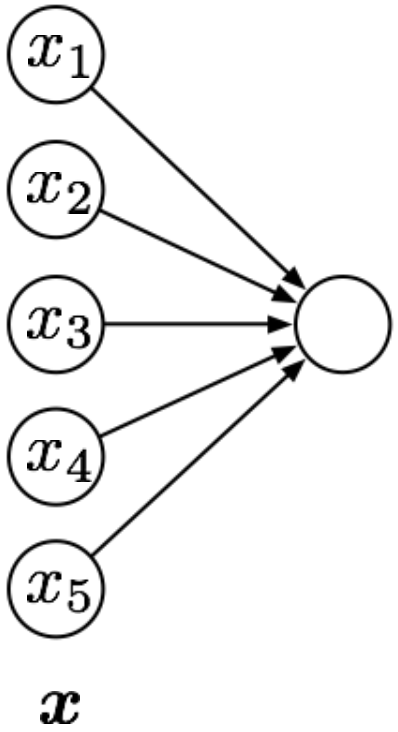
Brevity Penalty

- Ranges from 0.0 to 1.0, but usually shown multiplied by 100
- An increase of +1.0 BLEU is usually a conference paper
- MT systems usually score in the 10s to 30s
- Human translators usually score in the 70s and 80s

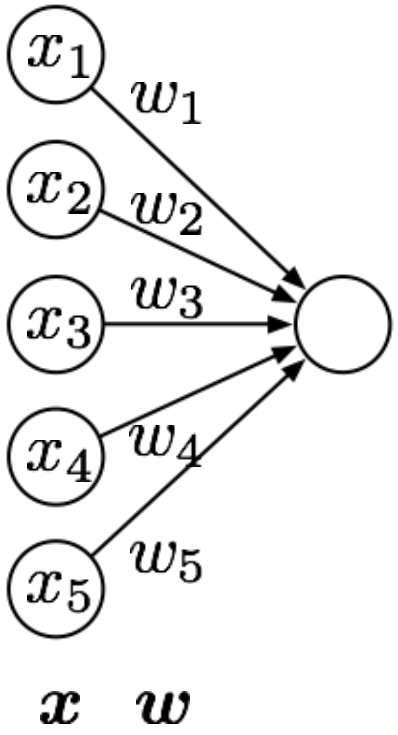
A Short Segue

- Word- and phrase-based ("symbolic") models were cutting edge for decades (up until ~2014)
  - Such models are still the most widely used in commercial applications
- Since 2014 most research on MT has focused on **neural** models

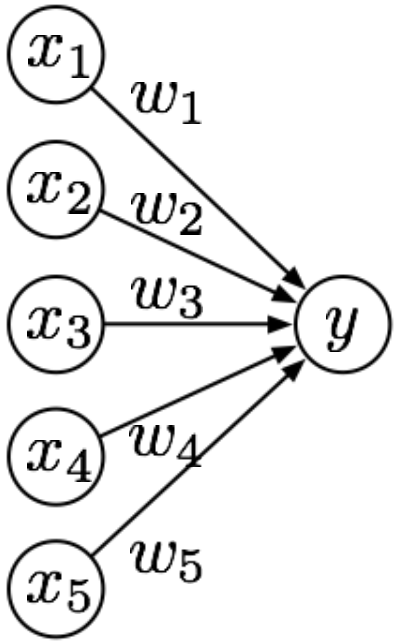
“Neurons”



“Neurons”

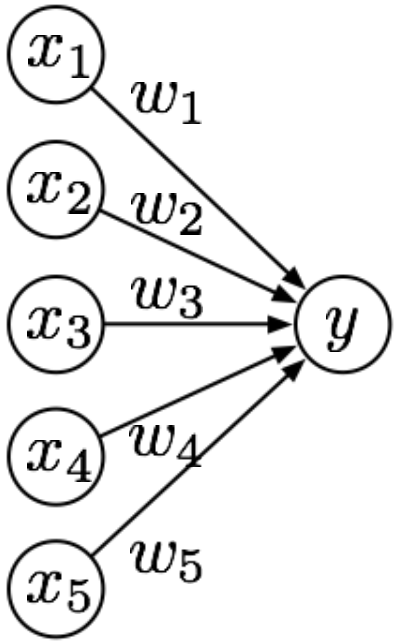


“Neurons”



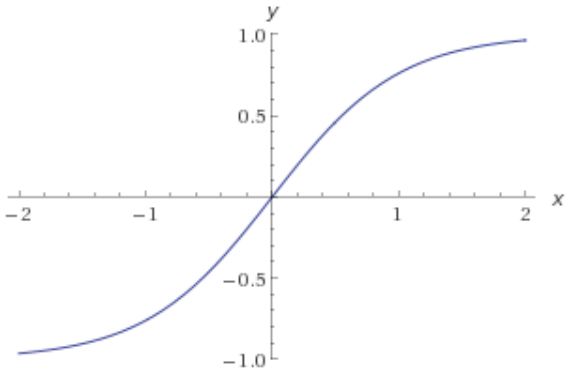
$$y = \boldsymbol{x} \cdot \boldsymbol{w}$$

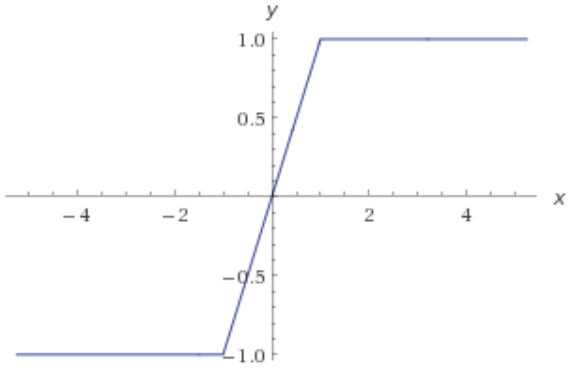
“Neurons”



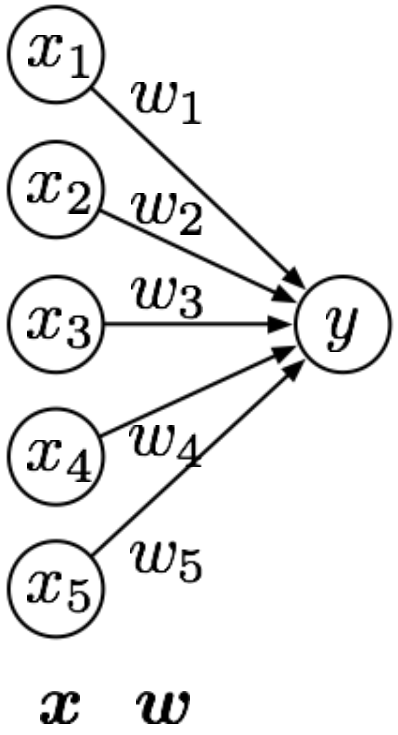
~~$y = \boldsymbol{x} \cdot \boldsymbol{w}$~~

$$y = g(\boldsymbol{x} \cdot \boldsymbol{w})$$

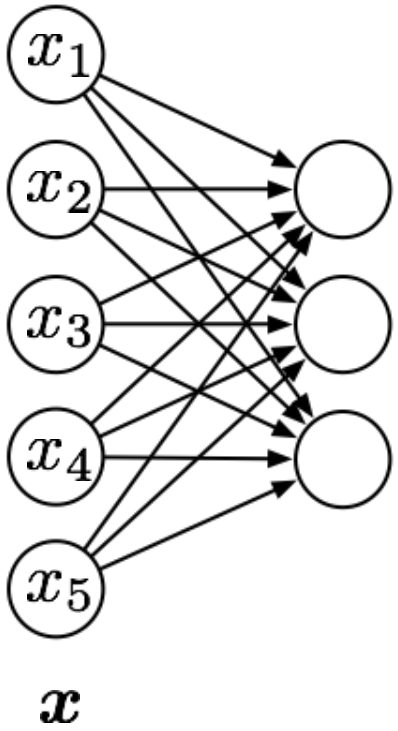




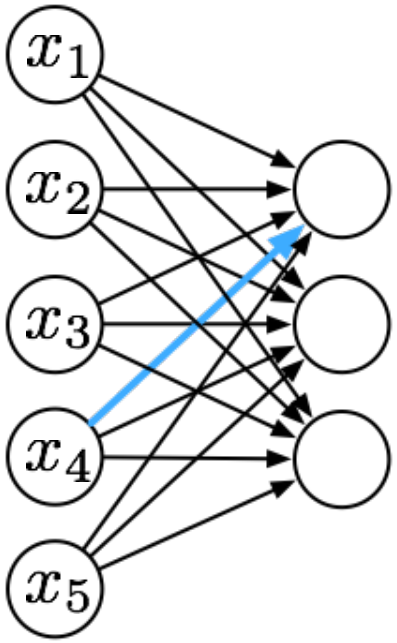
“Neurons”



“Neural” Networks

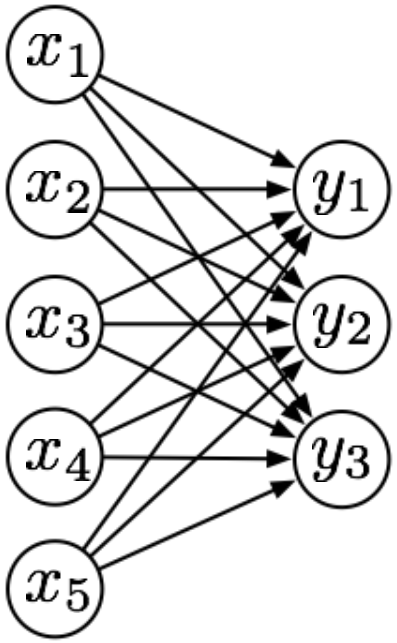


“Neural” Networks



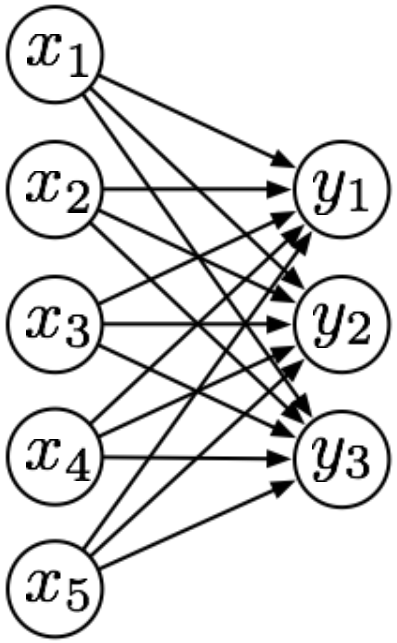
$\boldsymbol{x}$ $w_{4,1}$

“Neural” Networks



$$\boldsymbol{y} = \boldsymbol{x}^\top \mathbf{W}$$

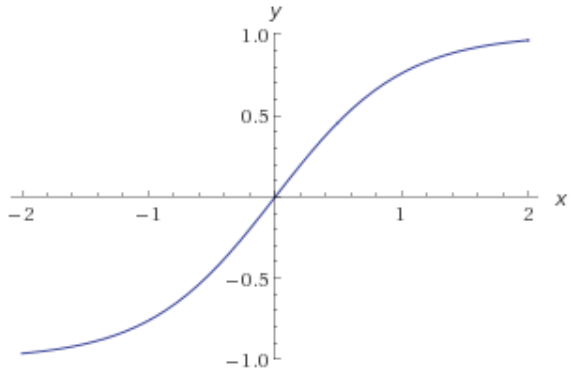
“Neural” Networks

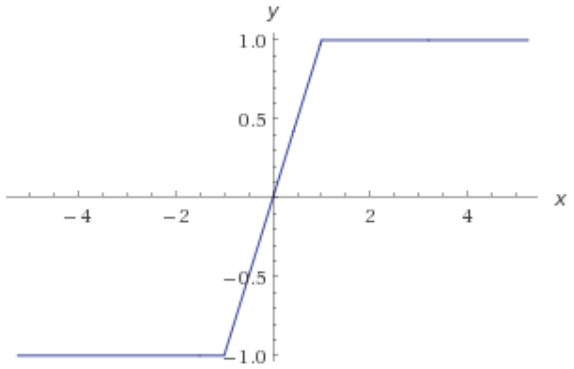


$\boldsymbol{x}$ $\mathbf{W}$ $\boldsymbol{y}$

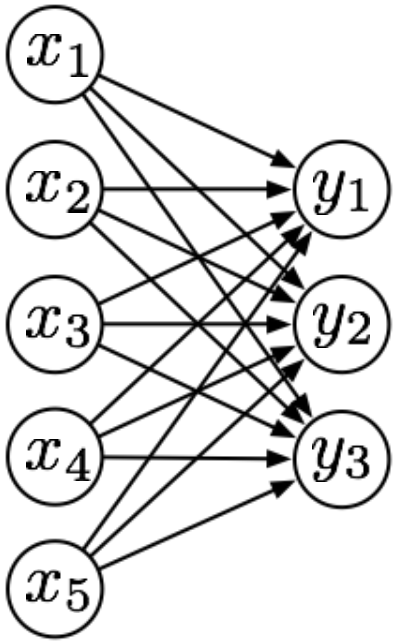
$$\boldsymbol{y} = \boldsymbol{x}^\top \mathbf{W}$$

$$\boldsymbol{y} = g(\boldsymbol{x}^\top \mathbf{W})$$





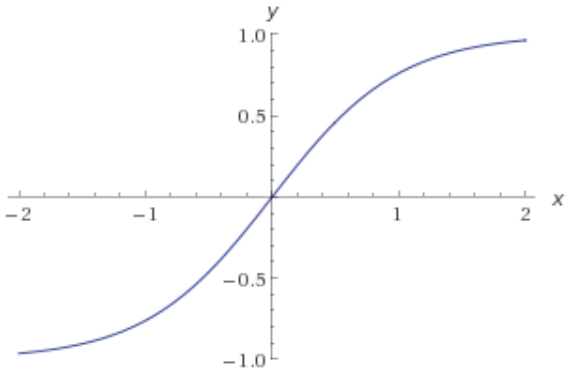
“Neural” Networks

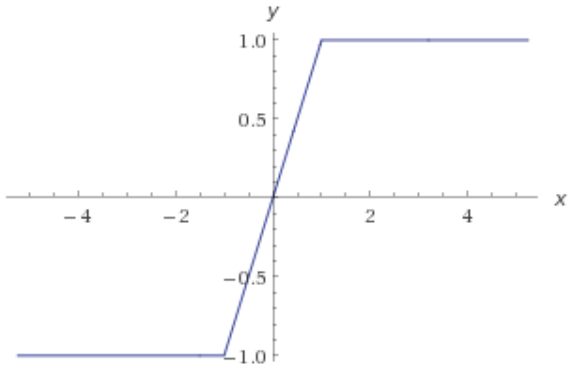


$\boldsymbol{x} \quad \mathbf{W} \quad \boldsymbol{y}$

~~$\boldsymbol{y} = \boldsymbol{x}^\top \mathbf{W}$~~

$$\boldsymbol{y} = g(\boldsymbol{x}^\top \mathbf{W})$$

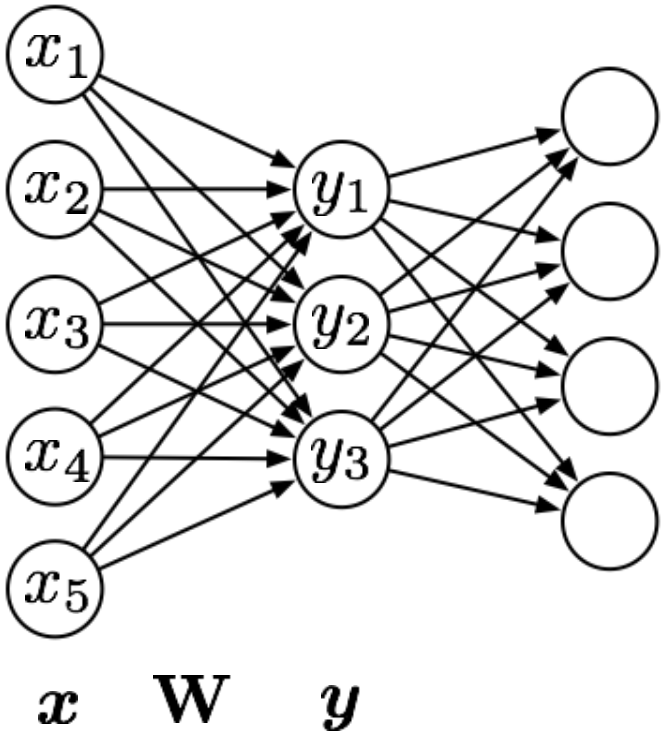




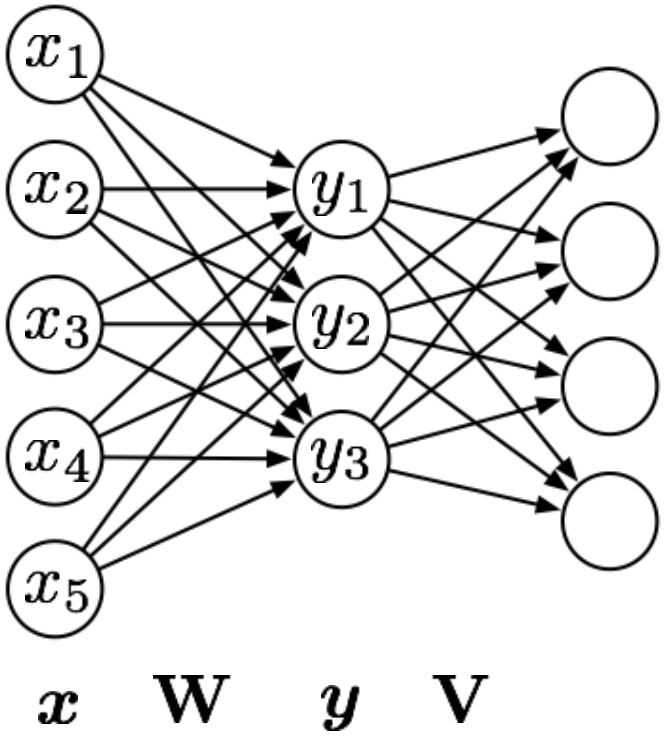
“Soft max”

$$g(\boldsymbol{u})_i = \frac{\exp u_i}{\sum_{i'} \exp u_{i'}}$$

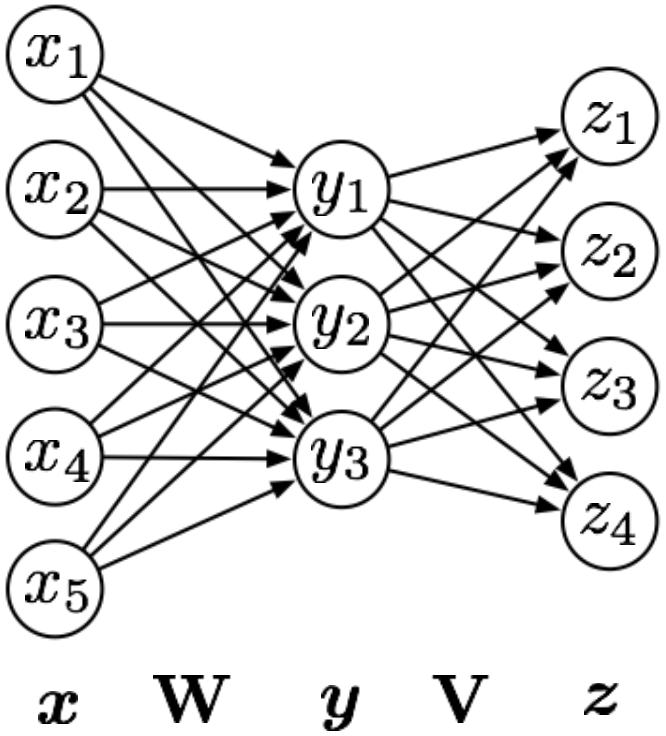
“Deep”



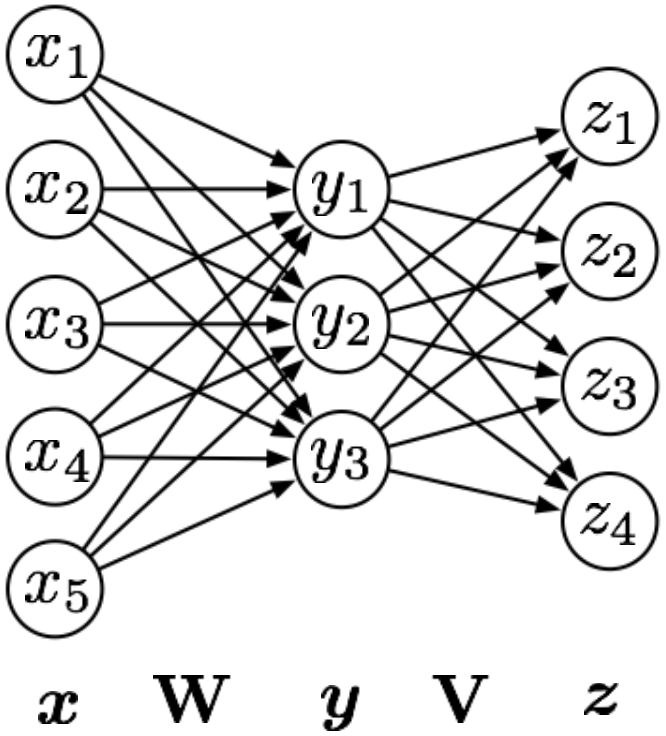
“Deep”


x_1
x_2
x_3
x_4
x_5
y_1
y_2
y_3
$\boldsymbol{x}$
$\mathbf{W}$
$\boldsymbol{y}$
$\mathbf{V}$

“Deep”

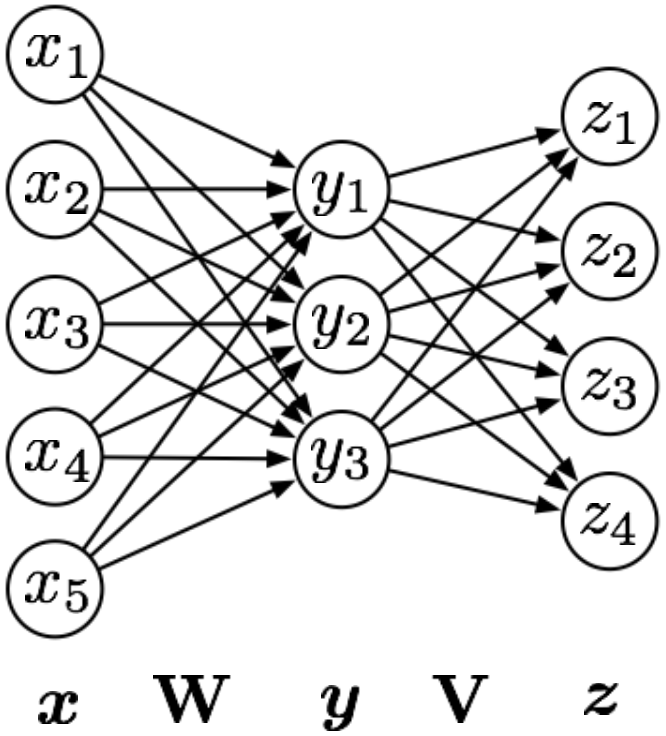

x_1
x_2
x_3
x_4
x_5
y_1
y_2
y_3
z_1
z_2
z_3
z_4
$\boldsymbol{x}$
$\mathbf{W}$
$\boldsymbol{y}$
$\mathbf{V}$
$\boldsymbol{z}$

“Deep”



$$\boldsymbol{z} = g(\boldsymbol{y}^\top \mathbf{V})$$

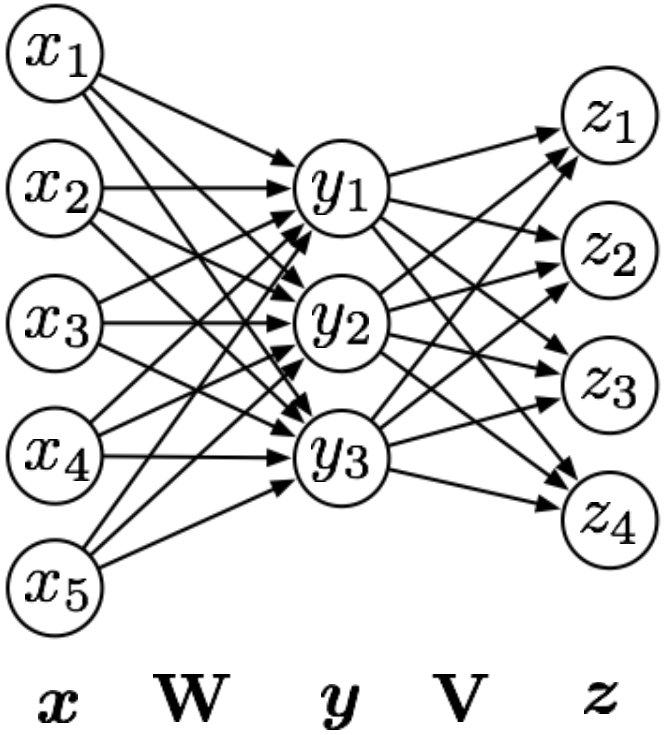
“Deep”



$$\boldsymbol{z} = g(\boldsymbol{y}^\top \mathbf{V})$$

$$\boldsymbol{z} = g(h(\boldsymbol{x}^\top \mathbf{W})^\top \mathbf{V})$$

“Deep”

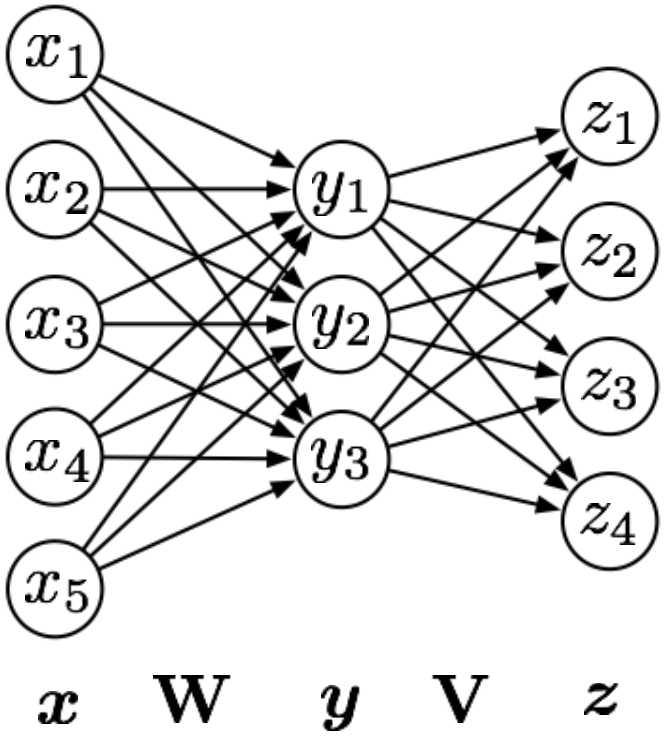


$$\boldsymbol{z} = g(\boldsymbol{y}^\top \mathbf{V})$$

$$\boldsymbol{z} = g(h(\boldsymbol{x}^\top \mathbf{W})^\top \mathbf{V})$$

$$\boldsymbol{z} = g(\mathbf{V} h(\mathbf{W}\boldsymbol{x}))$$

“Deep”

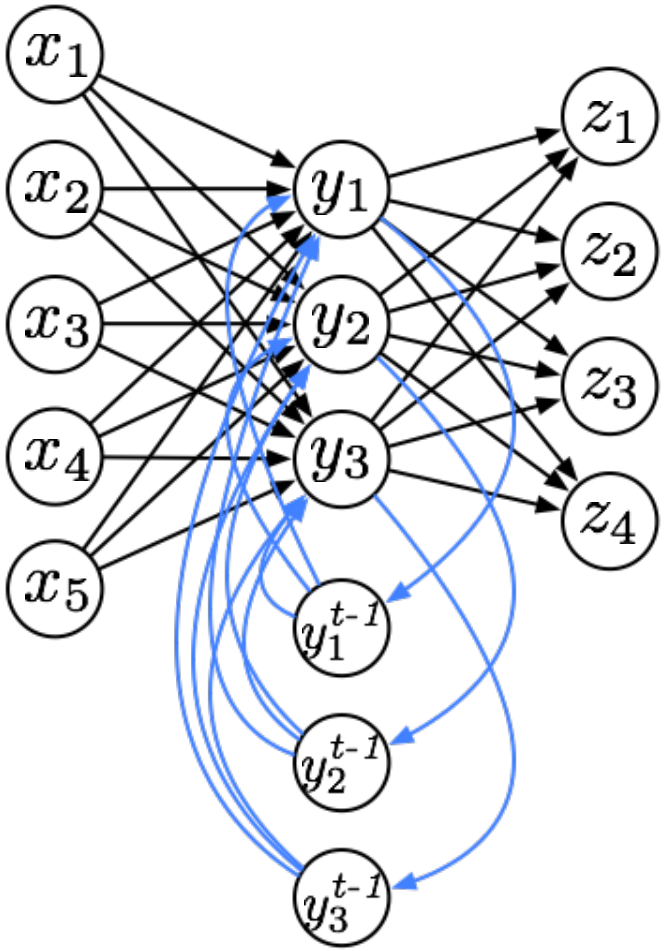


$$\boldsymbol{z} = g(\boldsymbol{y}^\top \mathbf{V})$$
$$\boldsymbol{z} = g(h(\boldsymbol{x}^\top \mathbf{W})^\top \mathbf{V})$$
$$\boldsymbol{z} = g(\mathbf{V} h(\mathbf{W}\boldsymbol{x}))$$

Note:

$$\text{if } g(\boldsymbol{x}) = h(\boldsymbol{x}) = \boldsymbol{x}$$
$$\boldsymbol{z} = \underbrace{(\mathbf{V}\mathbf{W})}_{\mathbf{U}}\boldsymbol{x}$$

“Recurrent”



Design Decisions

- How to represent inputs and outputs?
- Neural architecture?
  - How many layers? (Requires non-linearities to improve capacity!)
  - How many neurons?
  - Recurrent or not?
  - What kind of non-linearities?

Representing Language

- “One-hot” vectors
  - Each position in a vector corresponds to a word type

Aardvark Aabalone Abandon Abash … Dog …

dog = <0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0>

- Distributed representations
  - Vectors encode “features” of input words (character n-grams, morphological features, etc.)

dog = <0.79995, 0.67263, 0.73924, 0.77496, 0.09286, 0.802798, 0.35508, 0.44789>

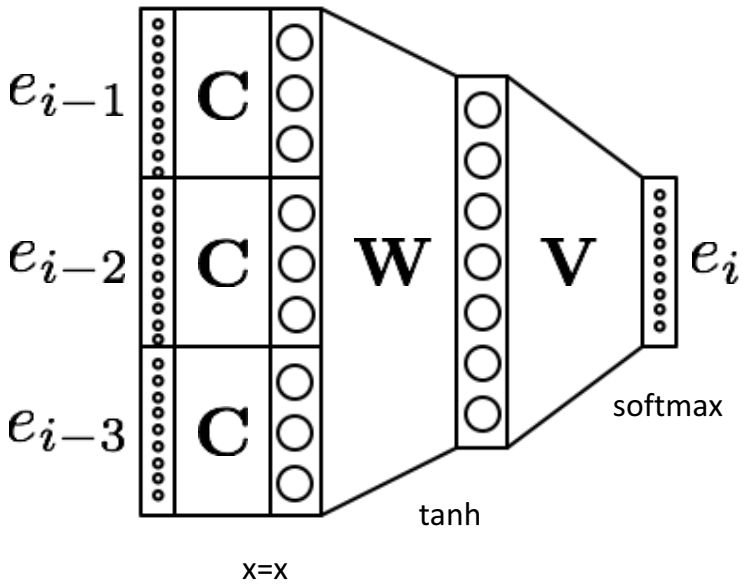
Training Neural Networks

- Neural networks are supervised models – you need a set of inputs paired with outputs
- Algorithm
  - Run until bored:
    - Give input to the network, see what it predicts
    - Compute loss(y, y*)
    - Use chain rule (aka “back propagation”) to compute gradient with respect to parameters
    - Update parameters (SGD, Adam, LBFGS, etc.)

Neural Language Models

$$p(\mathbf{e}) = \prod_{i=1}^{|\mathbf{e}|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1})$$

$$p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) =$$



Bengio et al. (2013)

Bengio et al. (2003)

	n	c	h	m	direct	mix	train.	valid.	test.
MLP1	5		50	60	yes	no	182	284	268
MLP2	5		50	60	yes	yes		275	257
MLP3	5		0	60	yes	no	201	327	310
MLP4	5		0	60	yes	yes		286	272
MLP5	5		50	30	yes	no	209	296	279
MLP6	5		50	30	yes	yes		273	259
MLP7	3		50	30	yes	no	210	309	293
MLP8	3		50	30	yes	yes		284	270
MLP9	5		100	30	no	no	175	280	276
MLP10	5		100	30	no	yes		265	**252**
Del. Int.	3						31	352	336
Kneser-Ney back-off	3							334	323
Kneser-Ney back-off	4							332	321
Kneser-Ney back-off	5							332	321

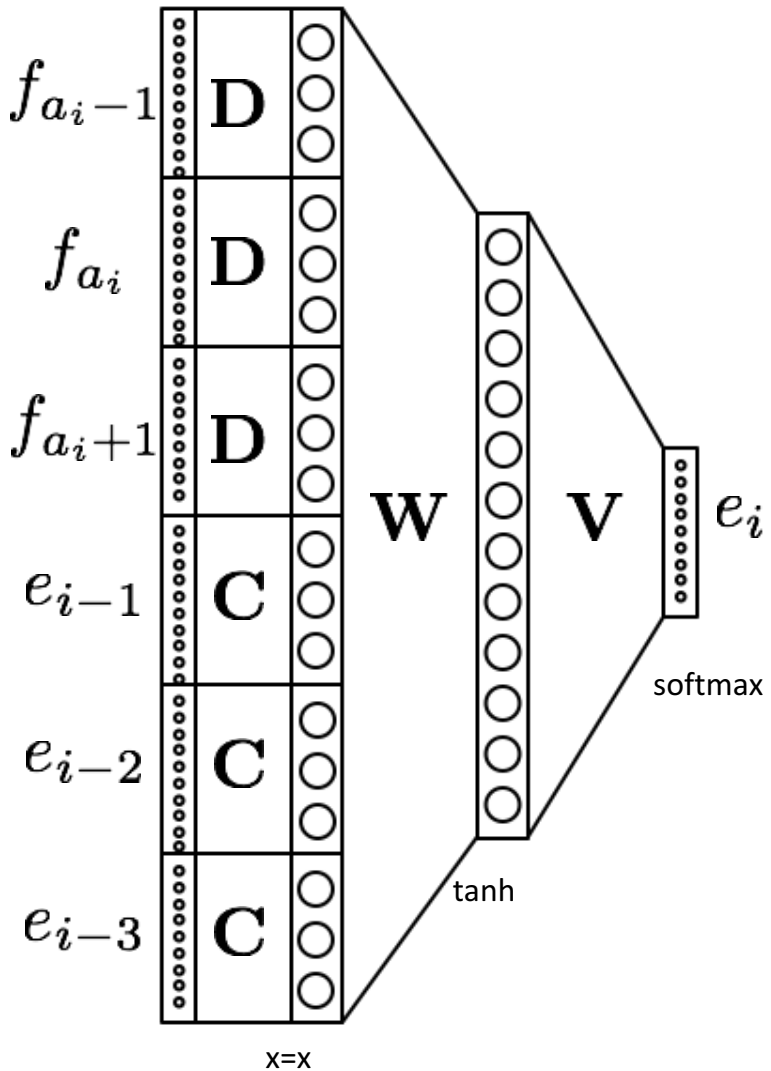
Neural Features for Translation

- Turn Bengio et al. (2003) into a translation model
- Condtional model, generate the next English word conditioned on
  - The previous *n* English words you generated
  - The aligned source word and its *m* neighbors

Devlin et al. (2014)

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

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



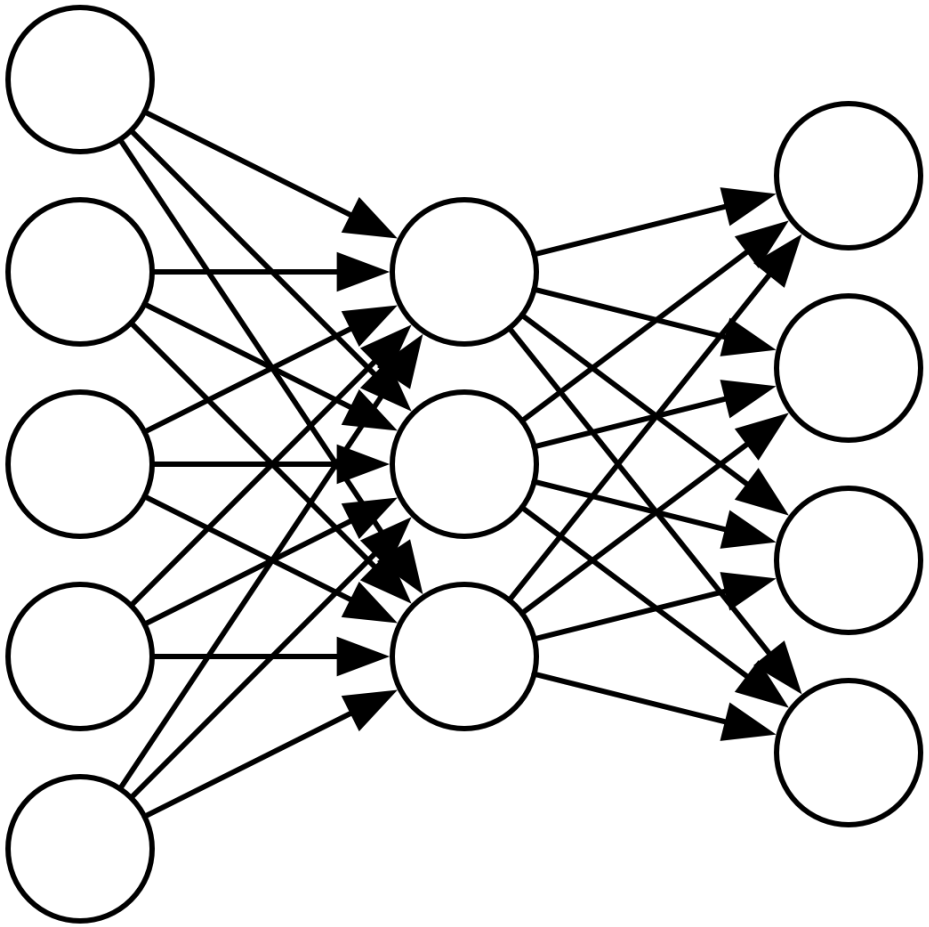
Devlin et al. (2014)

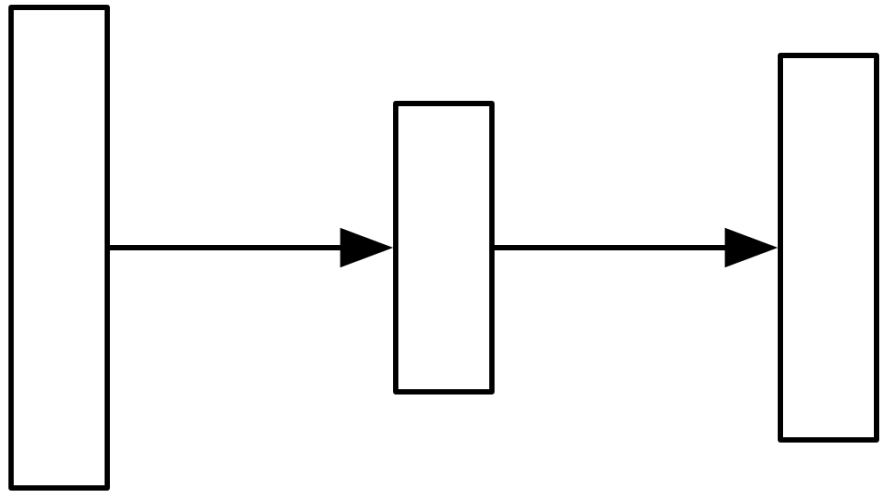
Neural Features for Translation

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%

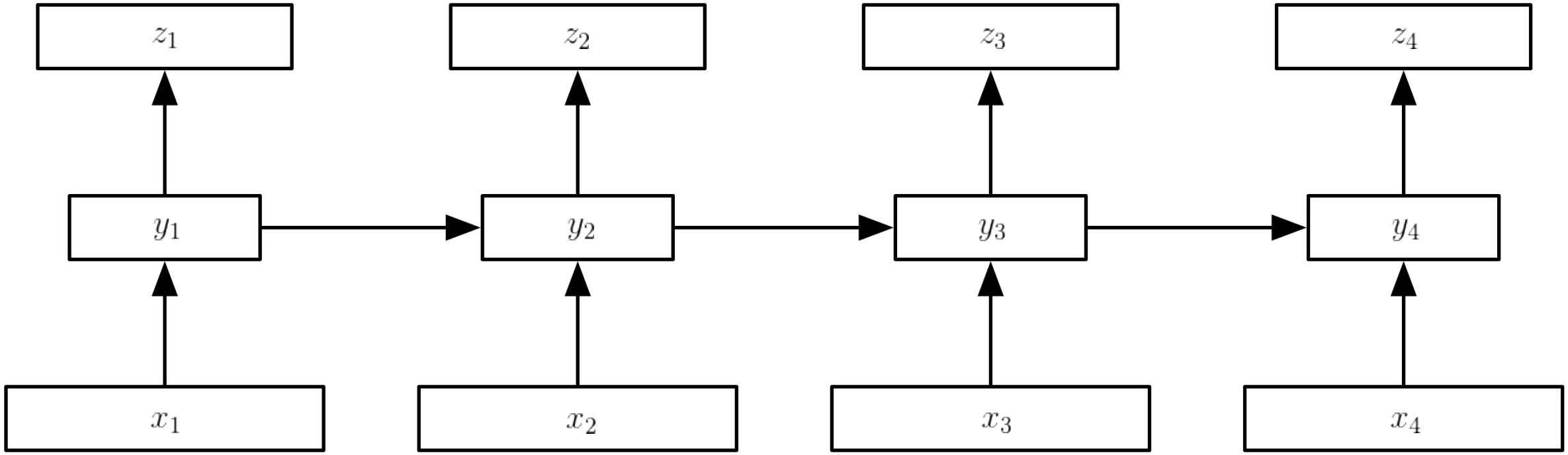
Devlin et al. (2014)

Notation Simplification



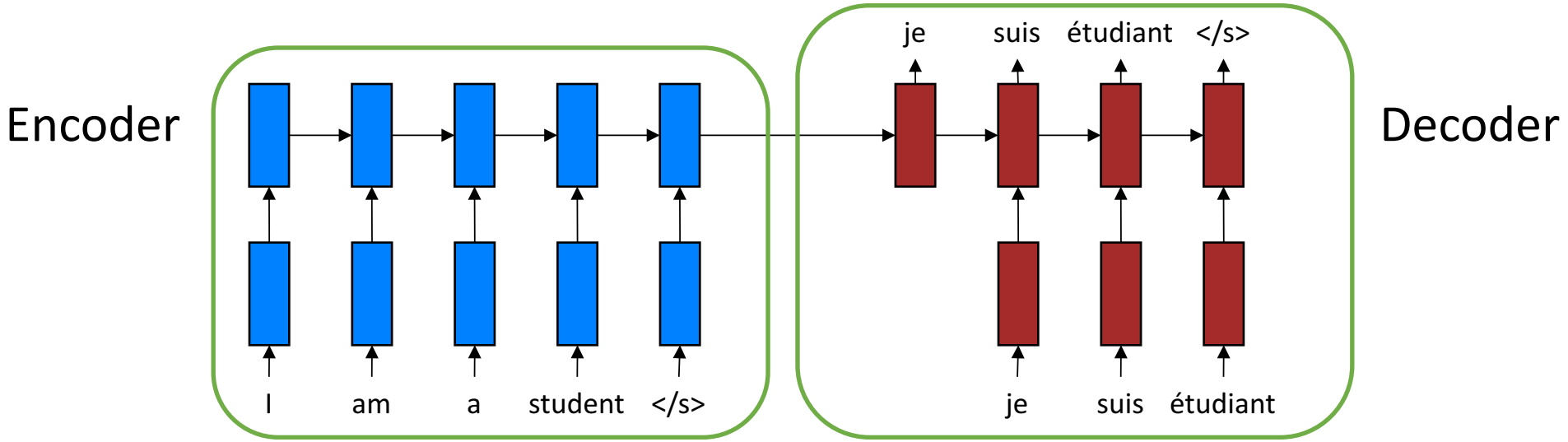


RNNs Revisited



Fully Neural Translation

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



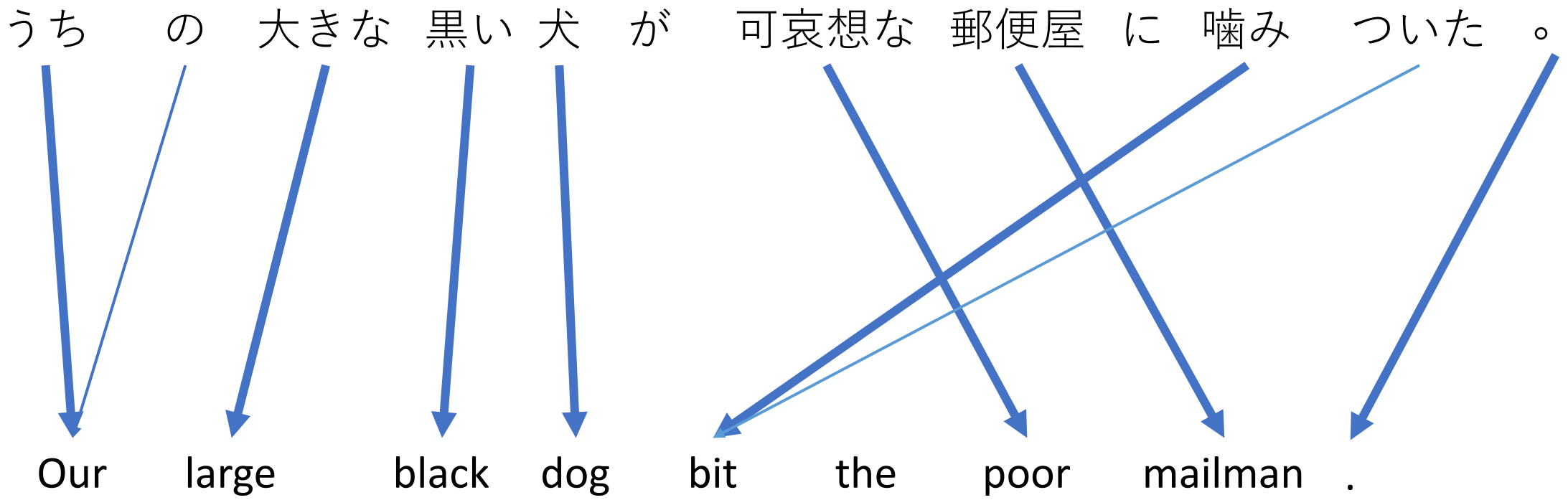
Sutskever et al. (2014)

Attentional Model

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

Bahdanau et al. (2014)

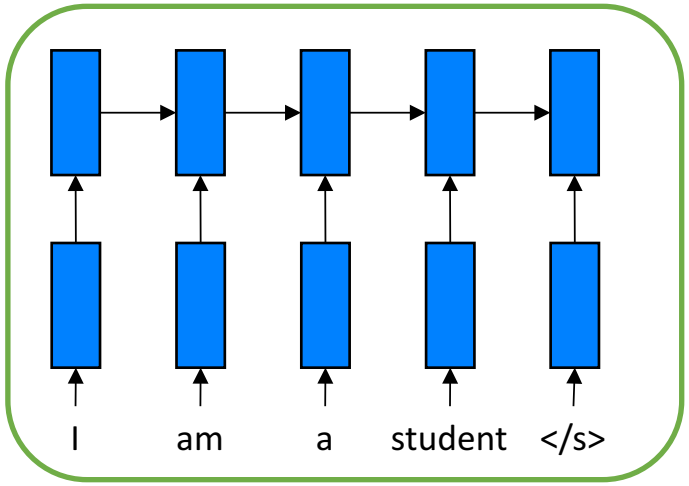
The Intuition

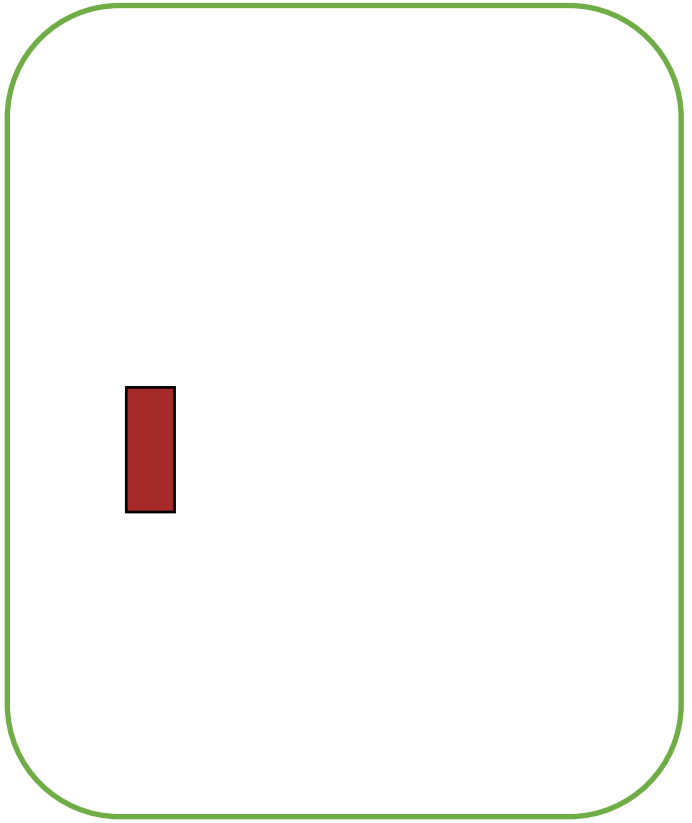


Bahdanau et al. (2014)

The Attention Model

Encoder

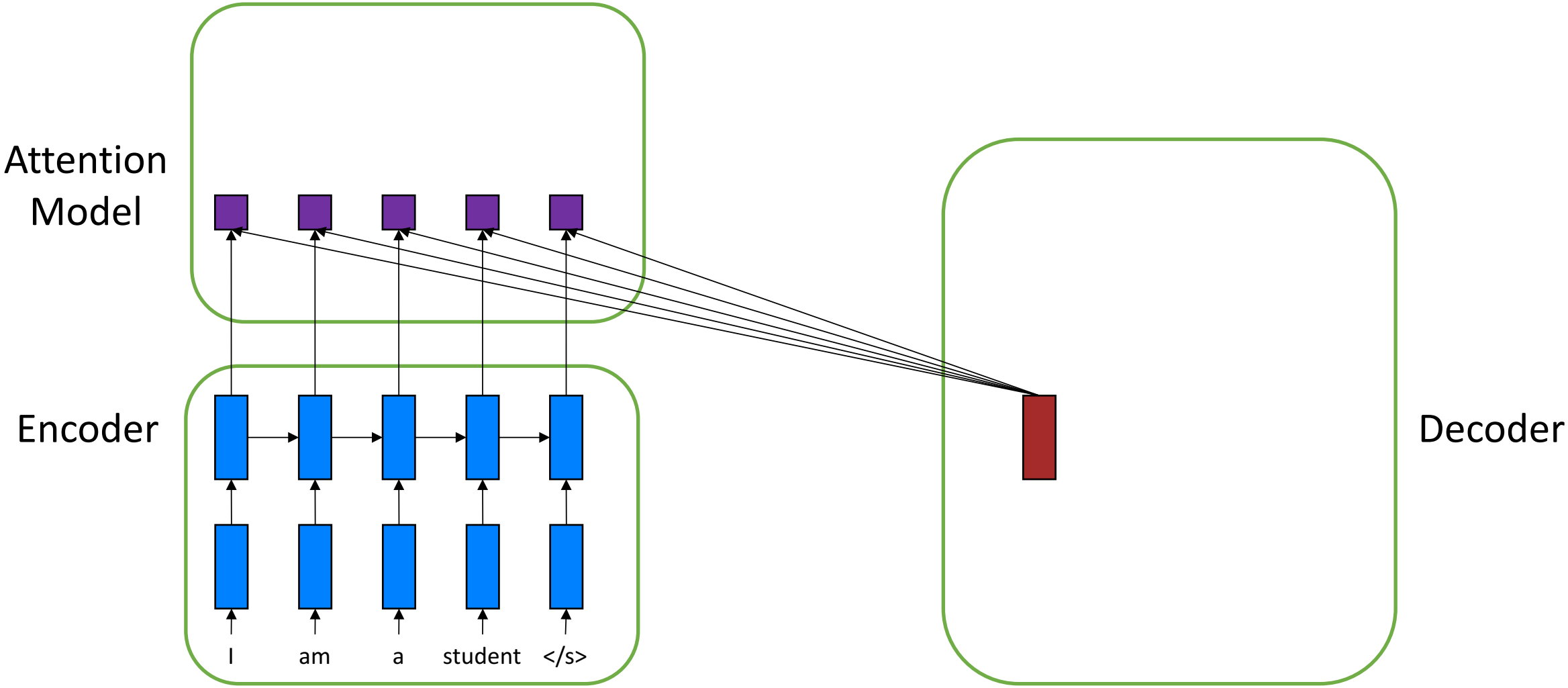




Decoder

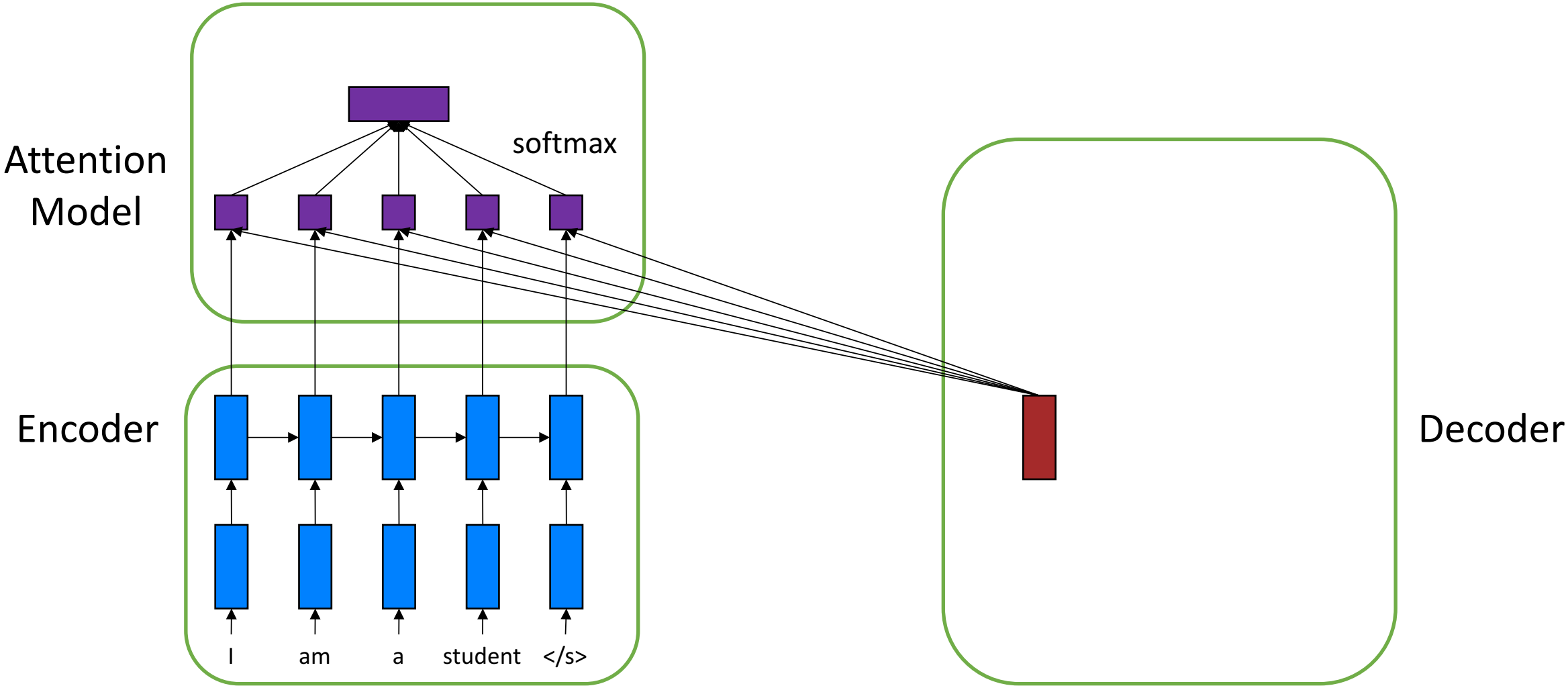
Bahdanau et al. (2014)

The Attention Model



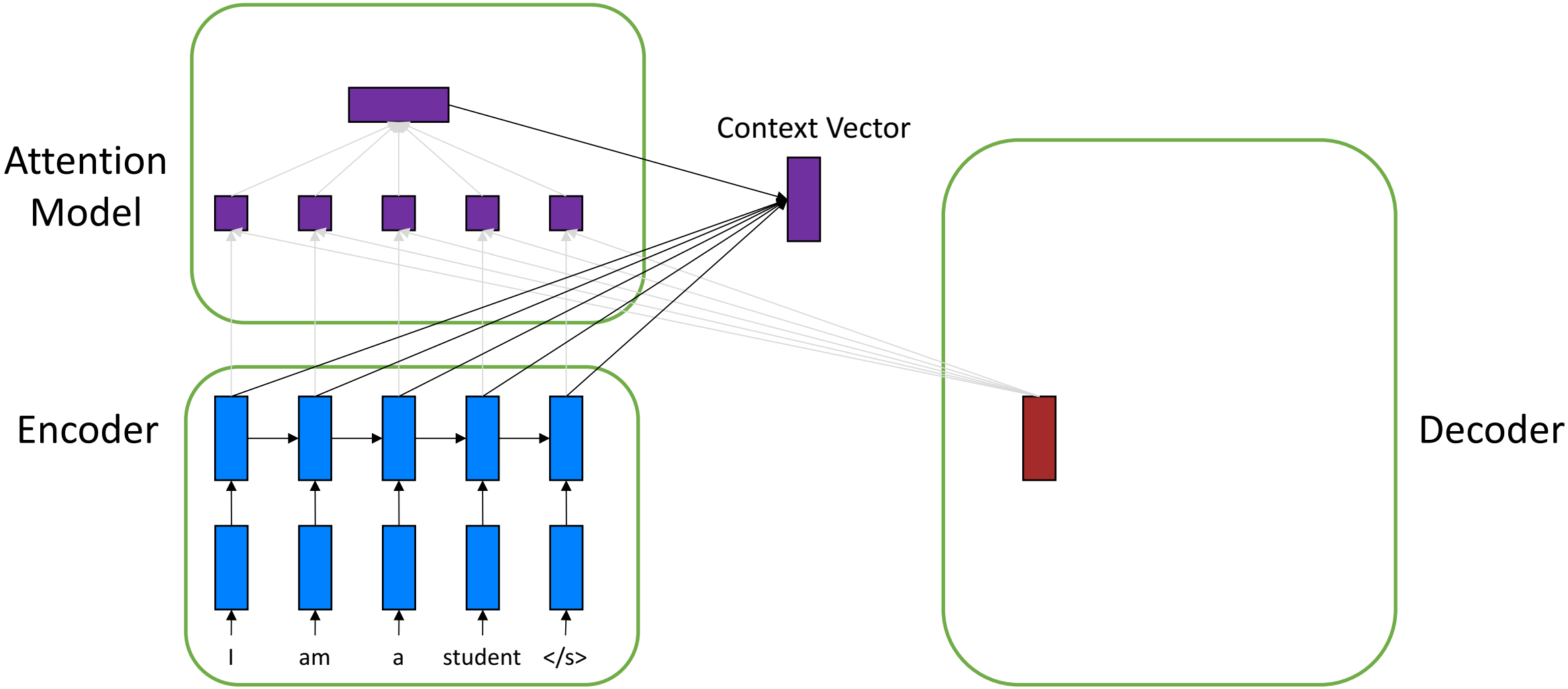
Bahdanau et al. (2014)

The Attention Model



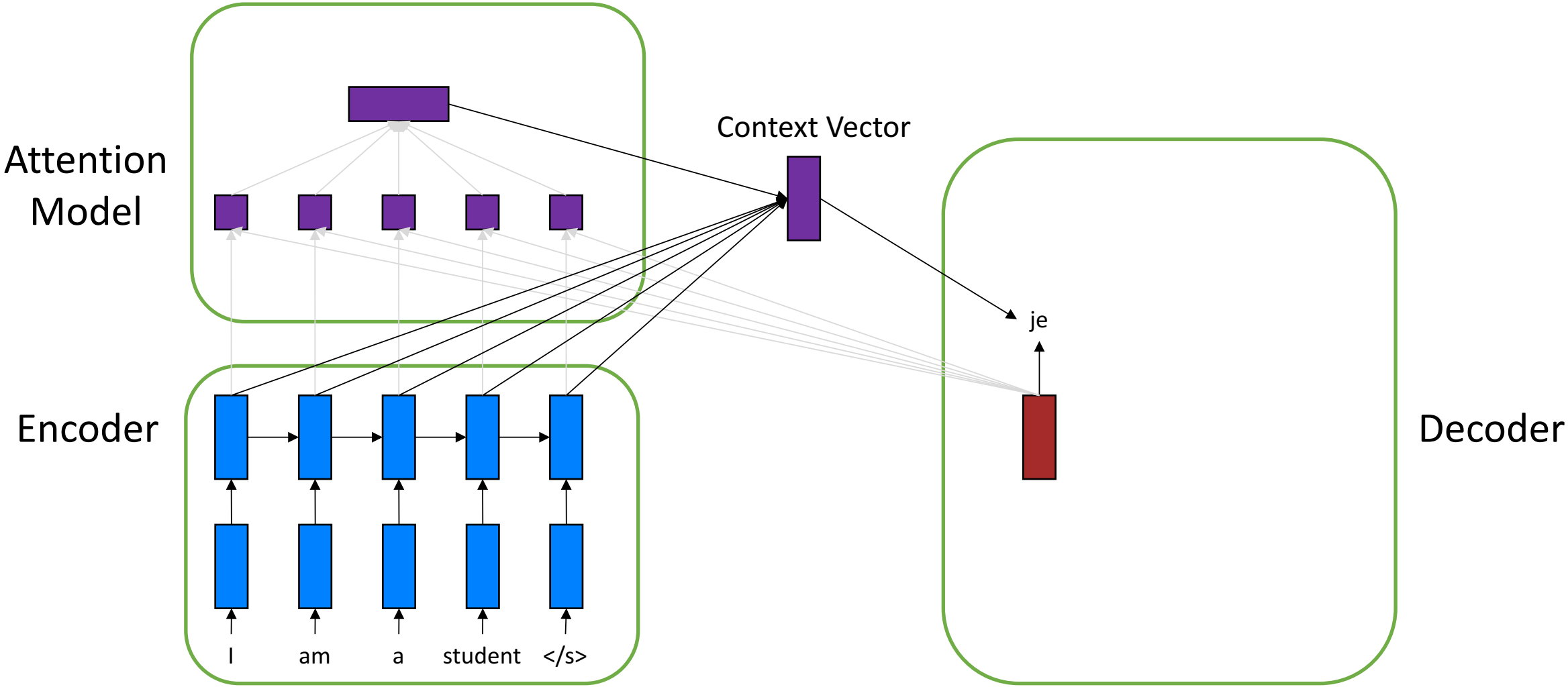
Bahdanau et al. (2014)

The Attention Model



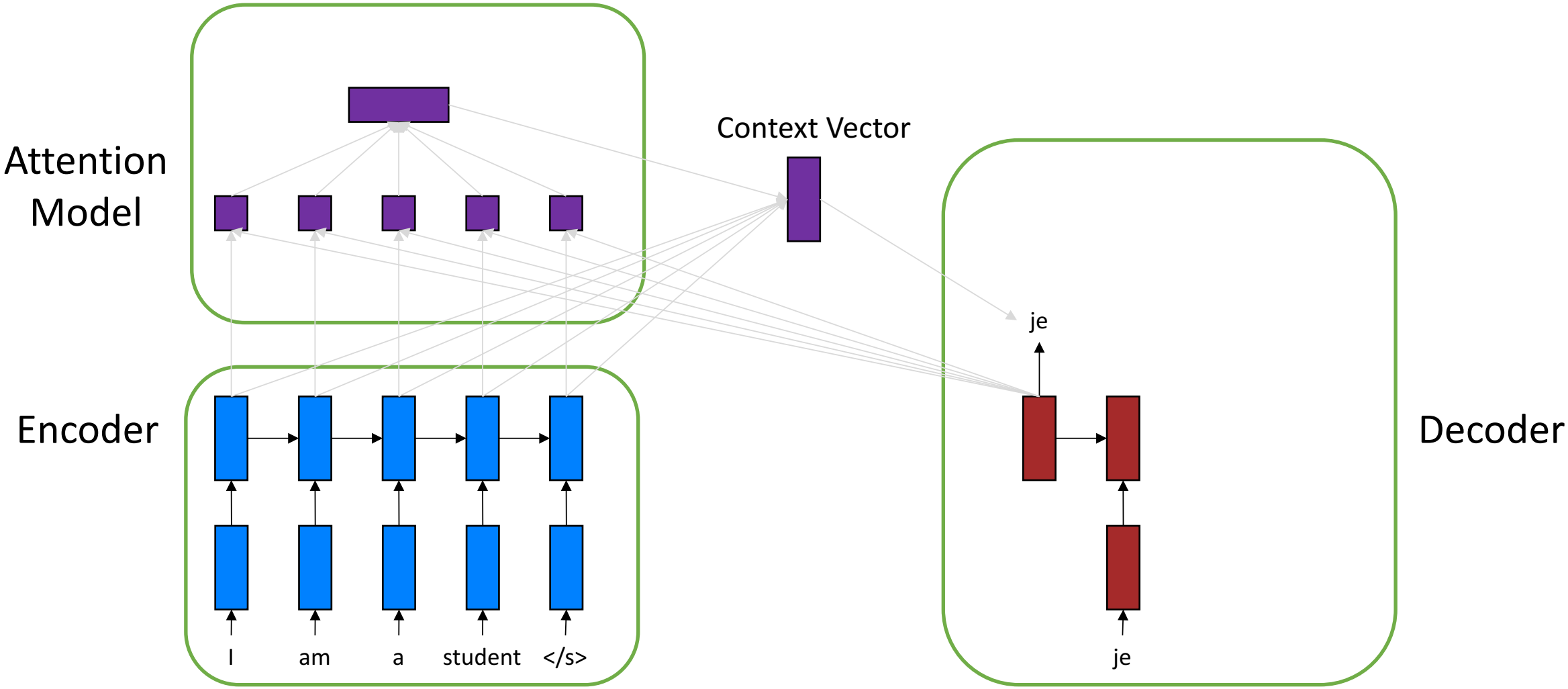
Bahdanau et al. (2014)

The Attention Model



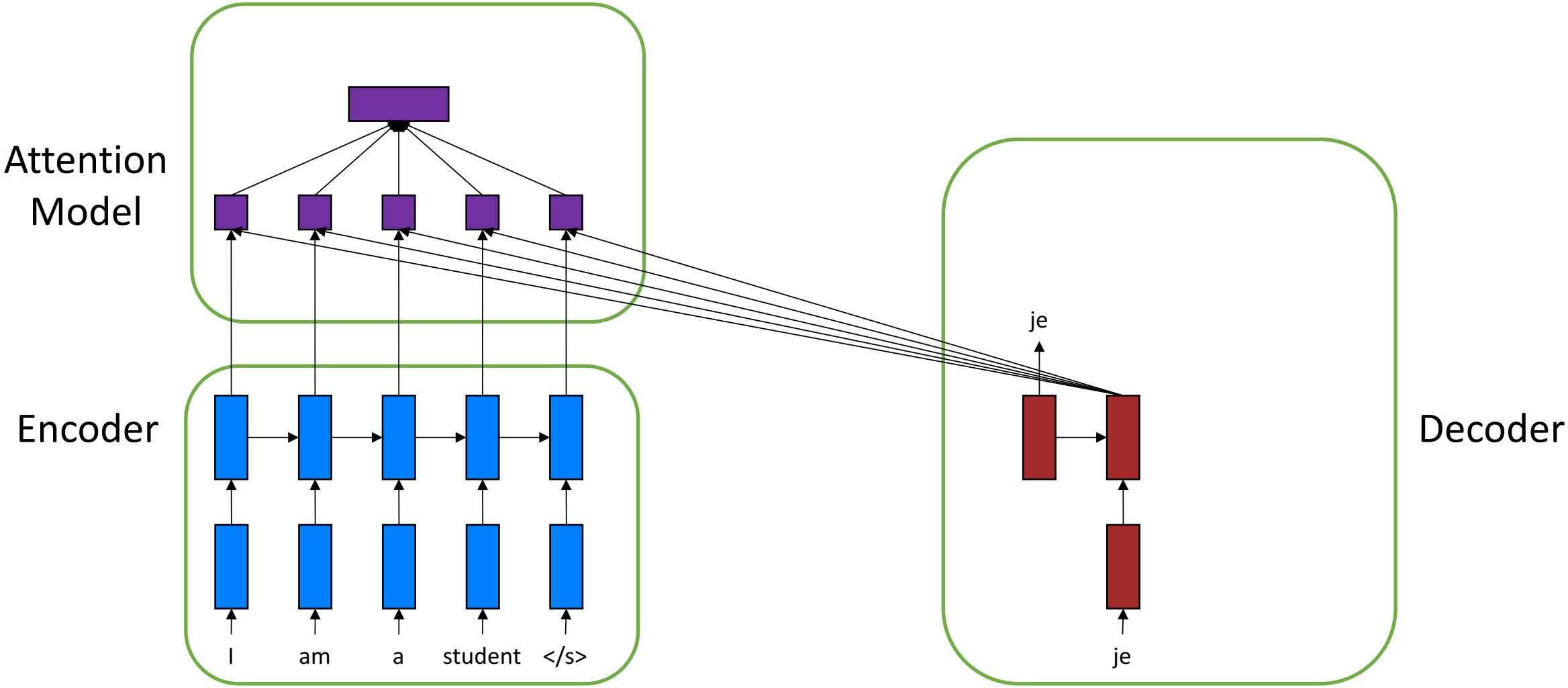
Bahdanau et al. (2014)

The Attention Model



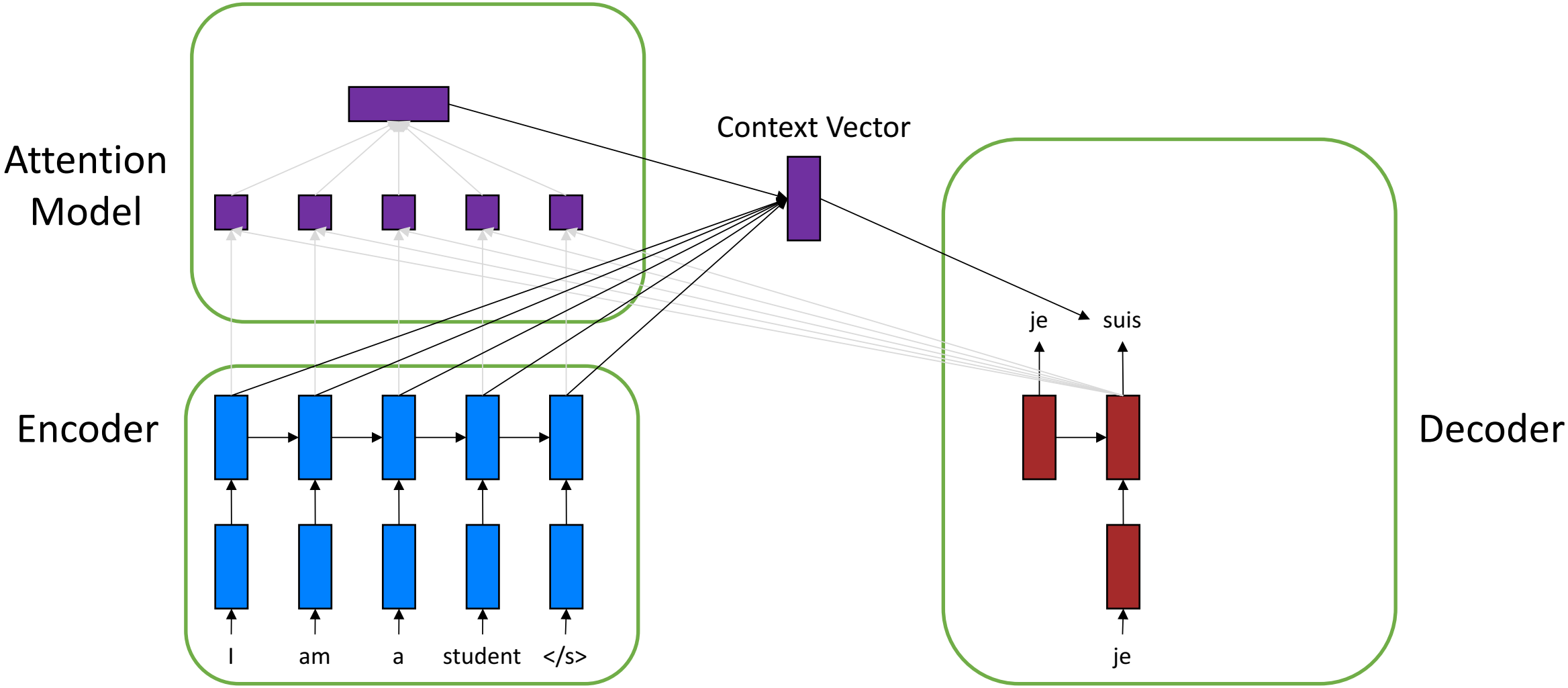
Bahdanau et al. (2014)

The Attention Model



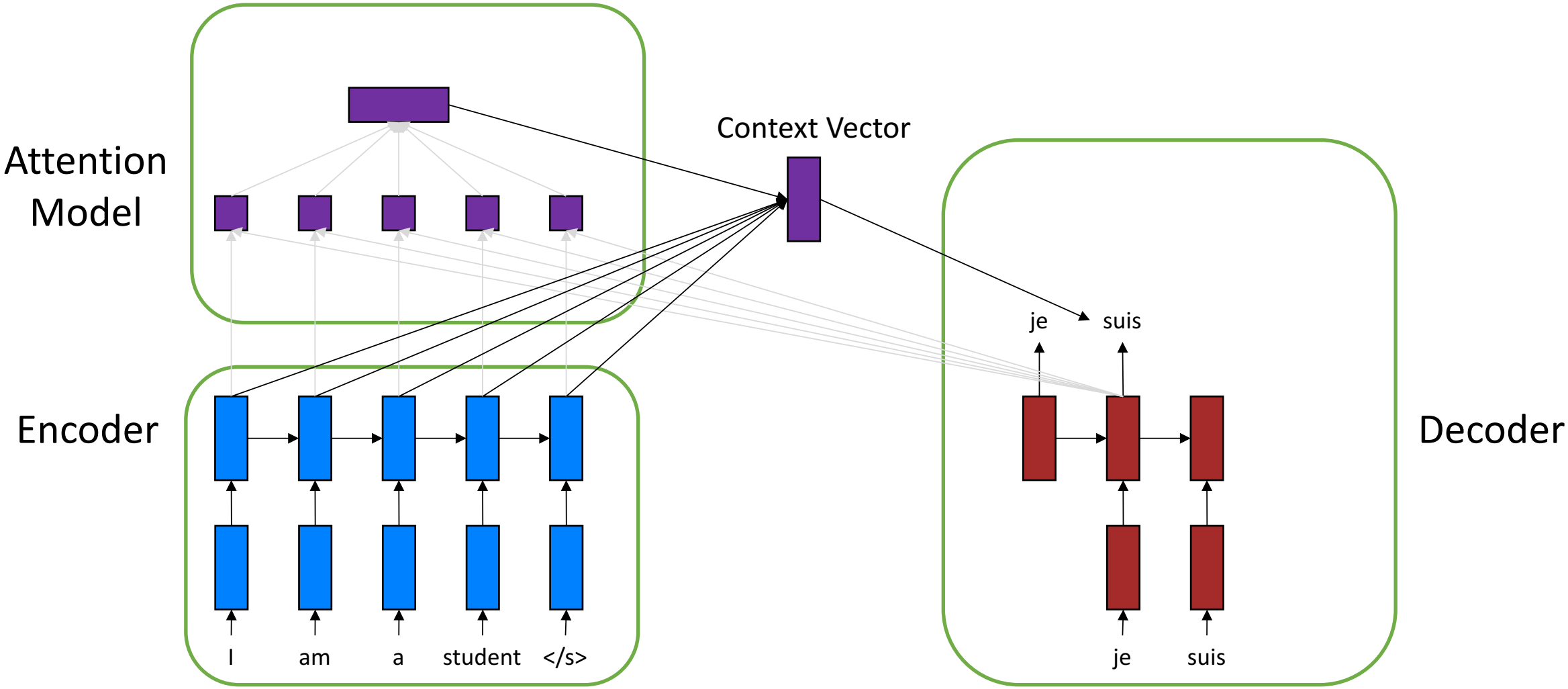
Bahdanau et al. (2014)

The Attention Model



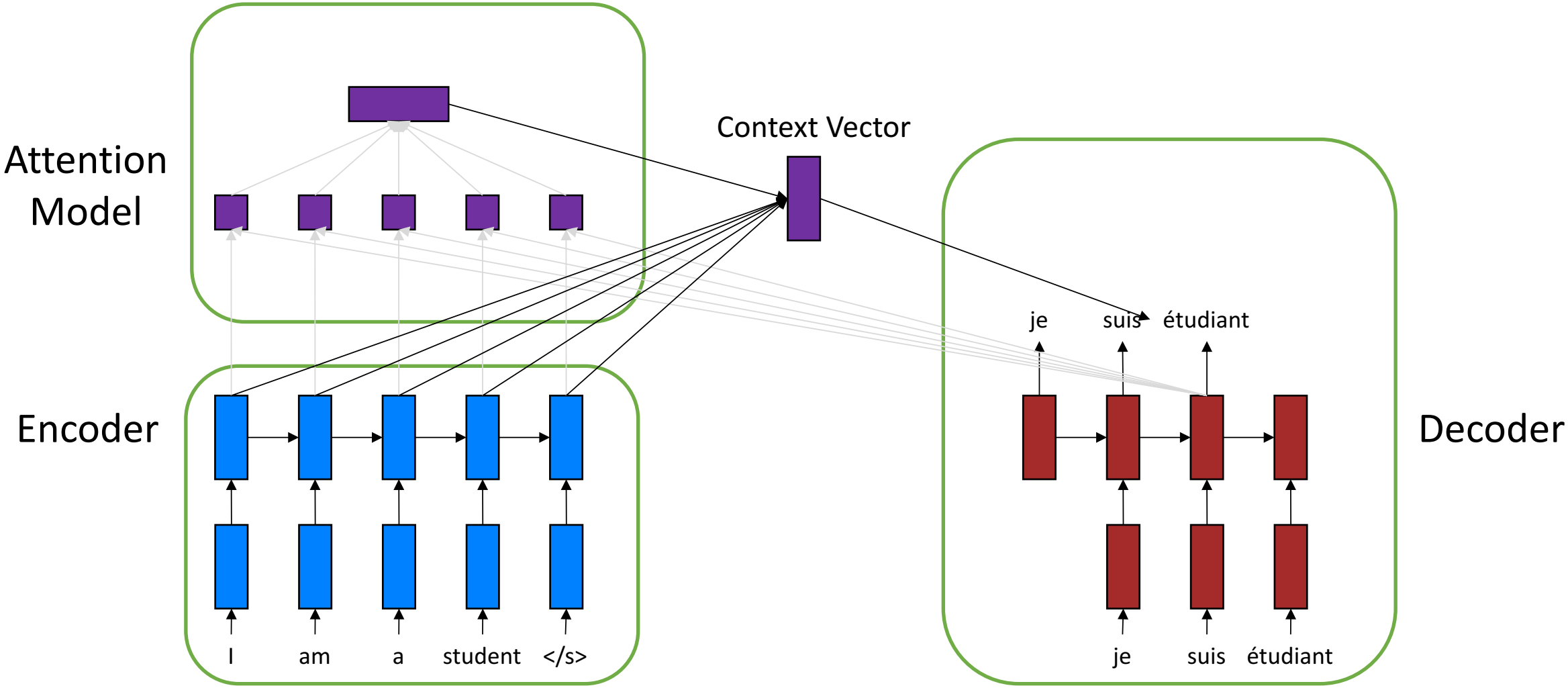
Bahdanau et al. (2014)

The Attention Model



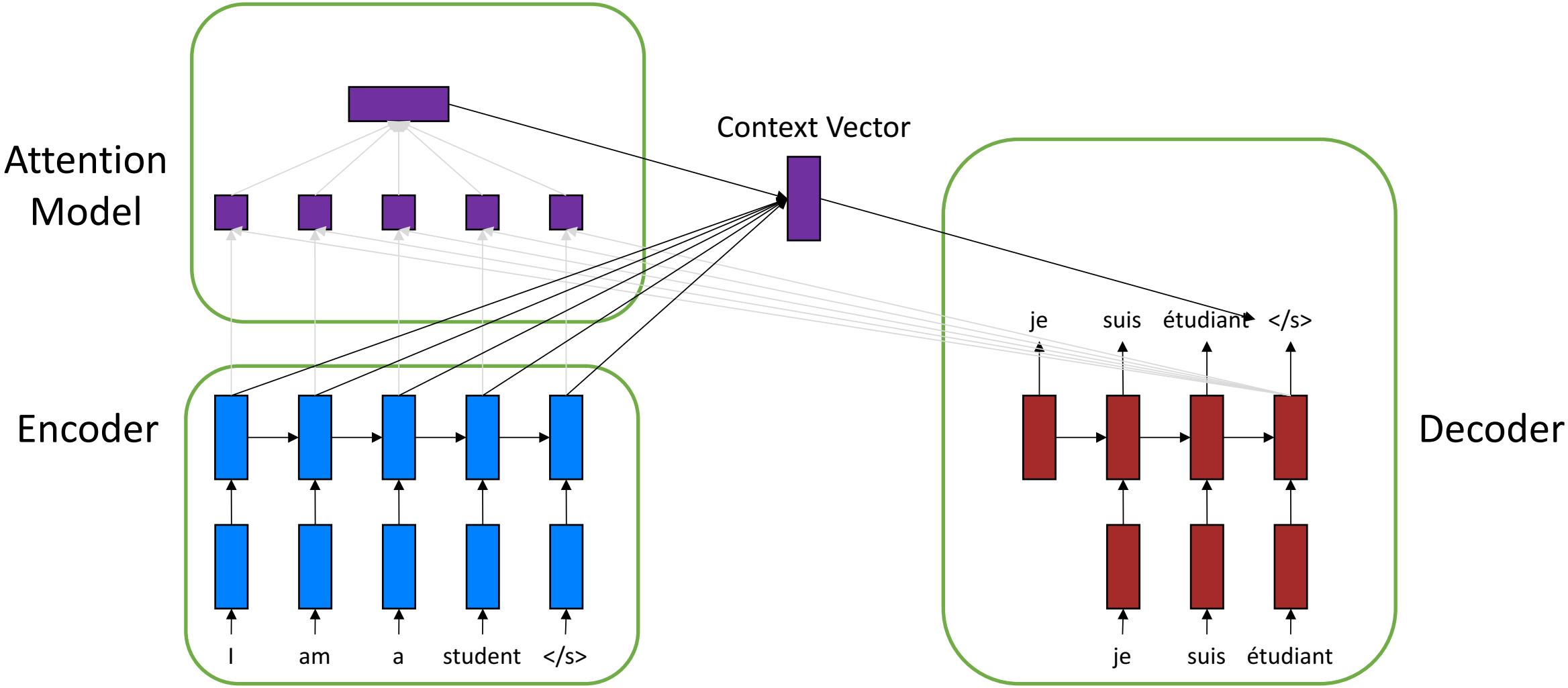
Bahdanau et al. (2014)

The Attention Model



Bahdanau et al. (2014)

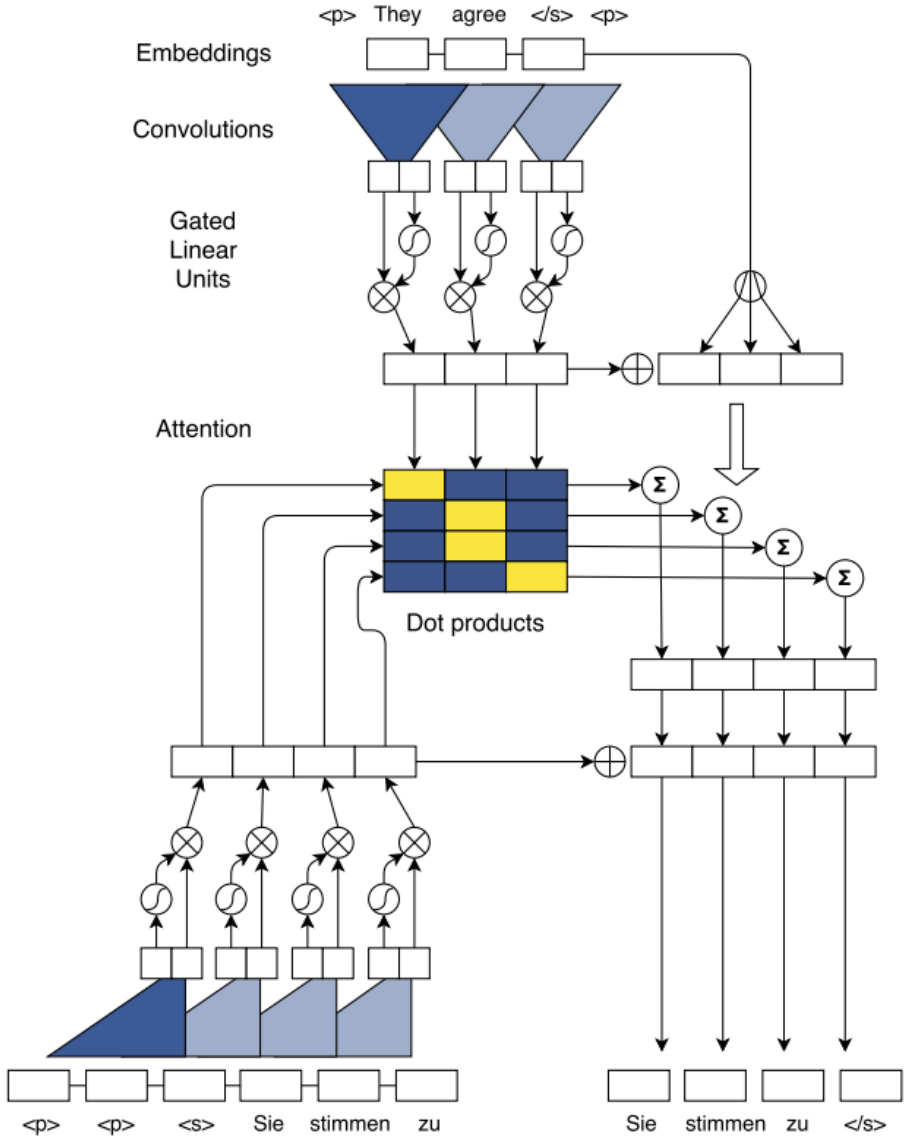
The Attention Model



Bahdanau et al. (2014)

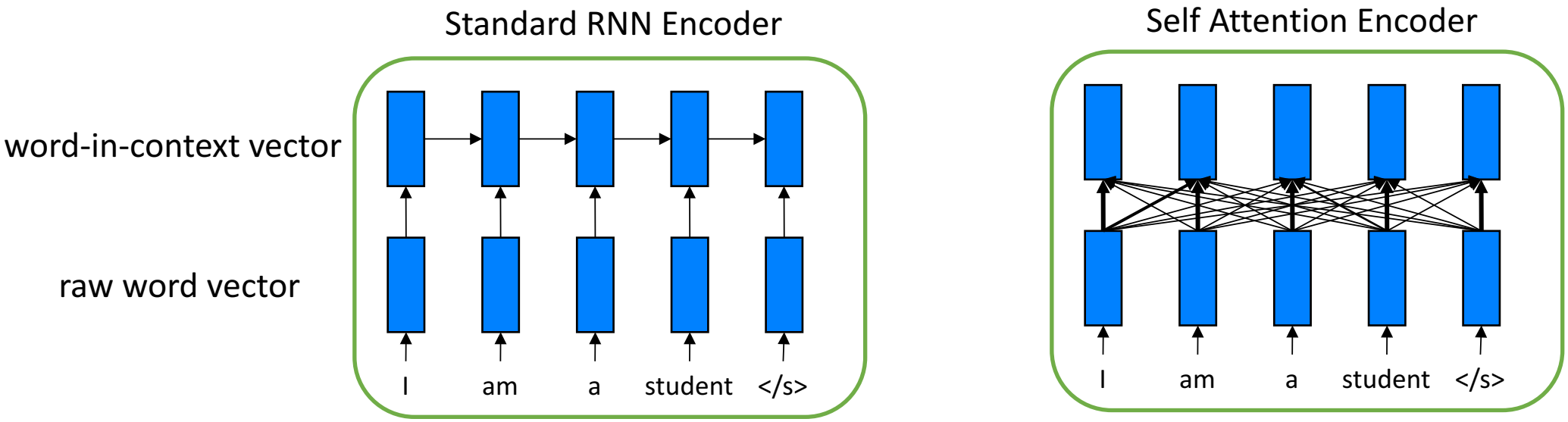
Convolutional Encoder-Decoder Gehring et. al 2017

- 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



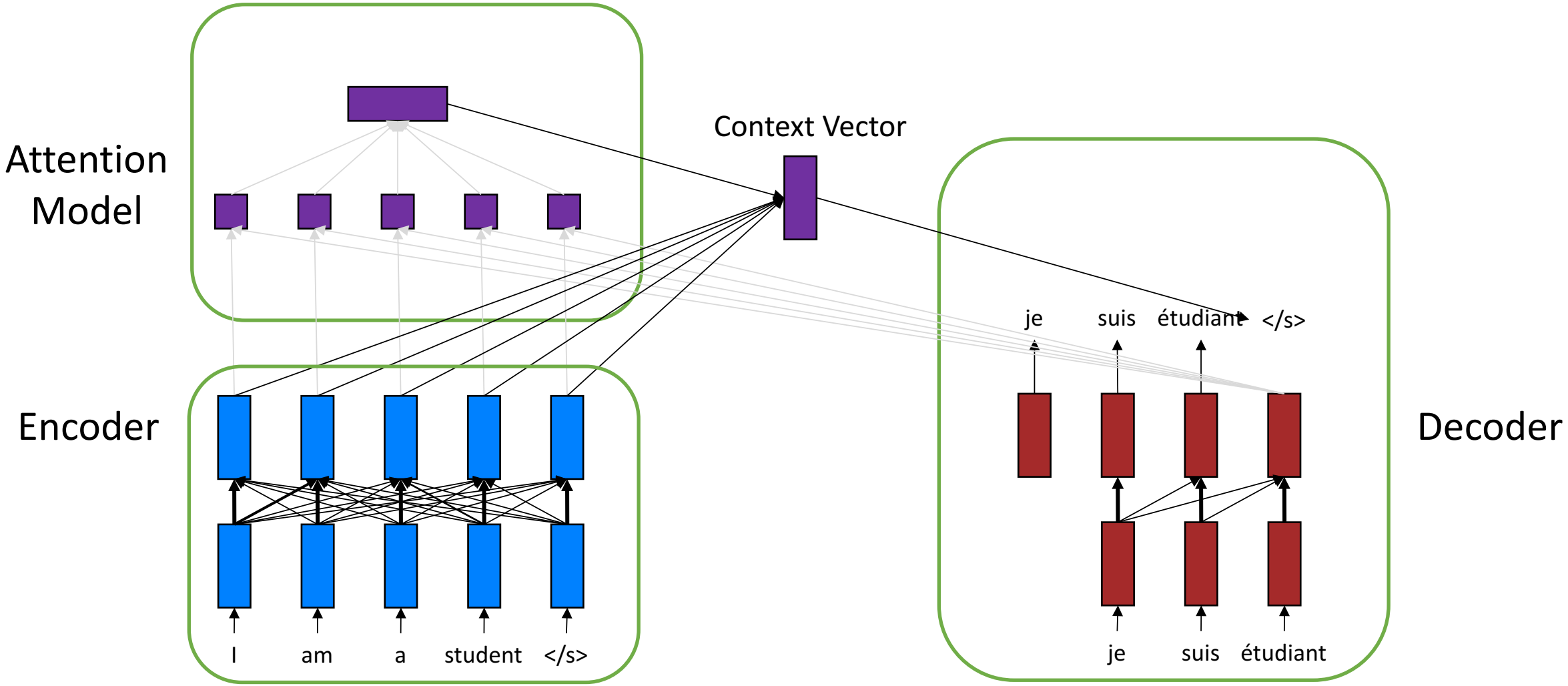
The Transformer

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



Vaswani et al. (2017)

The Transformer

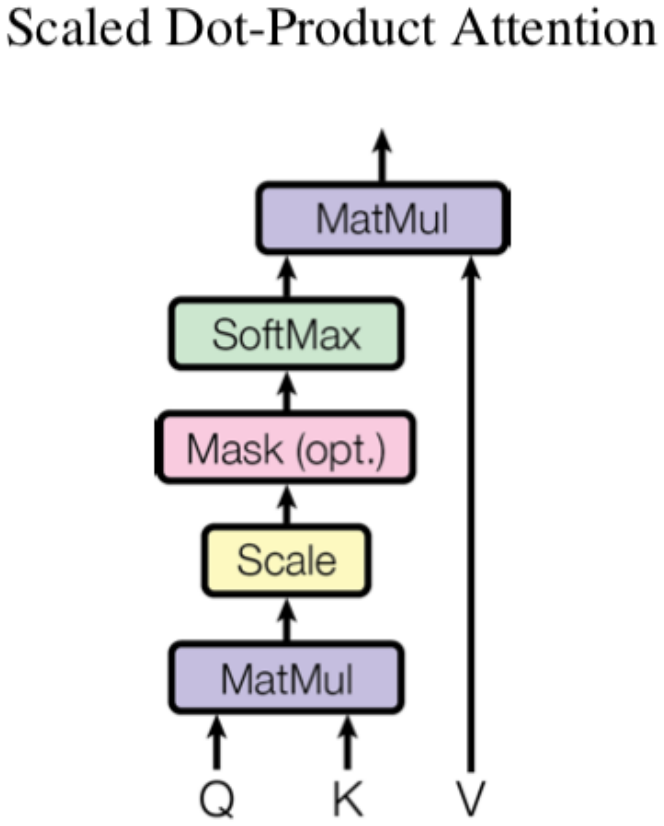


Vaswani et al. (2017)

Transformer

- Traditional attention:
  - Query: decoder hidden state
  - Key and Value: encoder hidden state
  - Attend to source words based on the current decoder state

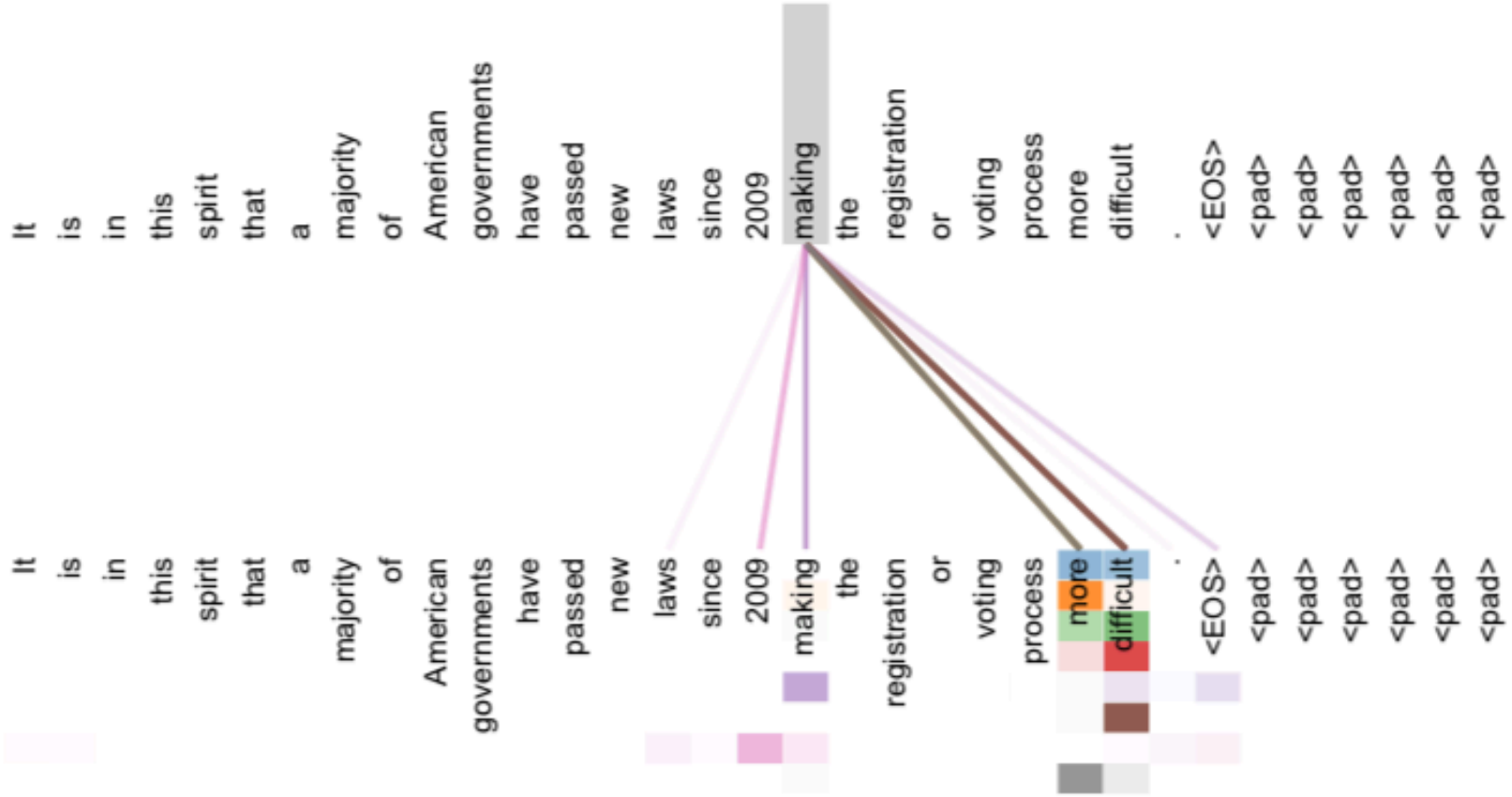
- Self-attention:
  - Query, Key, Value are the same
  - Attend to surrounding source words based on the current source word
  - Attend to preceeding target words based on the current target word



Vaswani et al. (2017)

Visualization of Attention Weight

- Self-attention weight can detect long-term dependency within a sentence, e.g., make … more difficult



The 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	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Current Research Directions on Neural MT

- Incorporation syntax into Neural MT
- Handling of morphologically rich languages
- Optimizing translation quality (instead of corpus probability)
- Multilingual models
- Document-level translation