Natural Language Processing
Lecture 08

Qun Liu, Valentin Malykh
Huawei Noah's Ark Lab

Spring 2022
A course delivered at KFU, Kazan

Content

1. Machine Translation (MT)
2. Machine translation evaluation
3. Statistical machine translation (SMT)
4. Neural machine translation (NMT) based on RNNs
5. Neural machine translation (NMT) with attentions

Content

1 Machine Translation (MT)

2 Machine translation evaluation

3 Statistical machine translation (SMT)

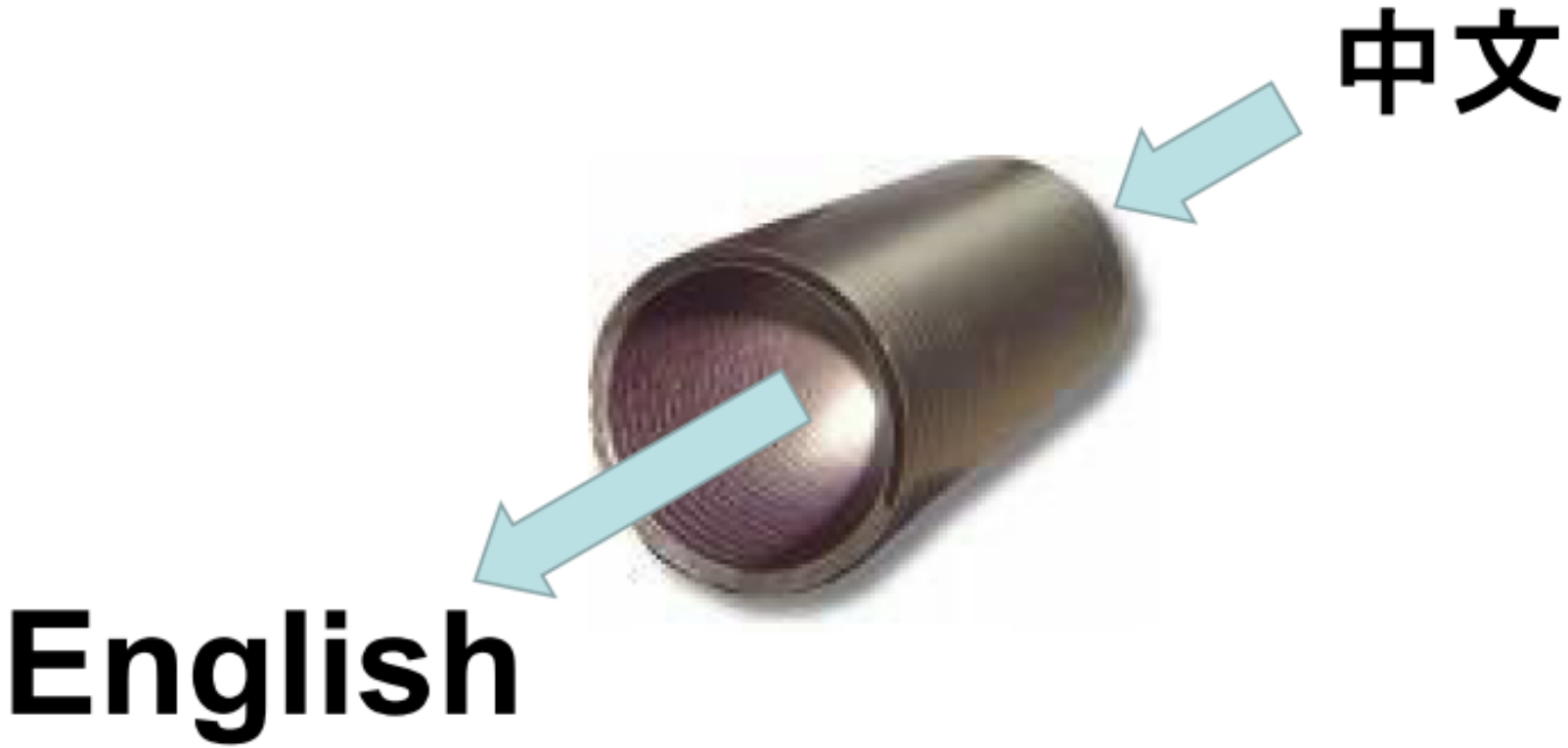
4 Neural machine translation (NMT) based on RNNs

5 Neural machine translation (NMT) with attentions

Holy grails of NLP (Recap)

- **Accurate machine translation between human languages**
- Free conversation between humans and computers

Accurate machine translation (Recap)



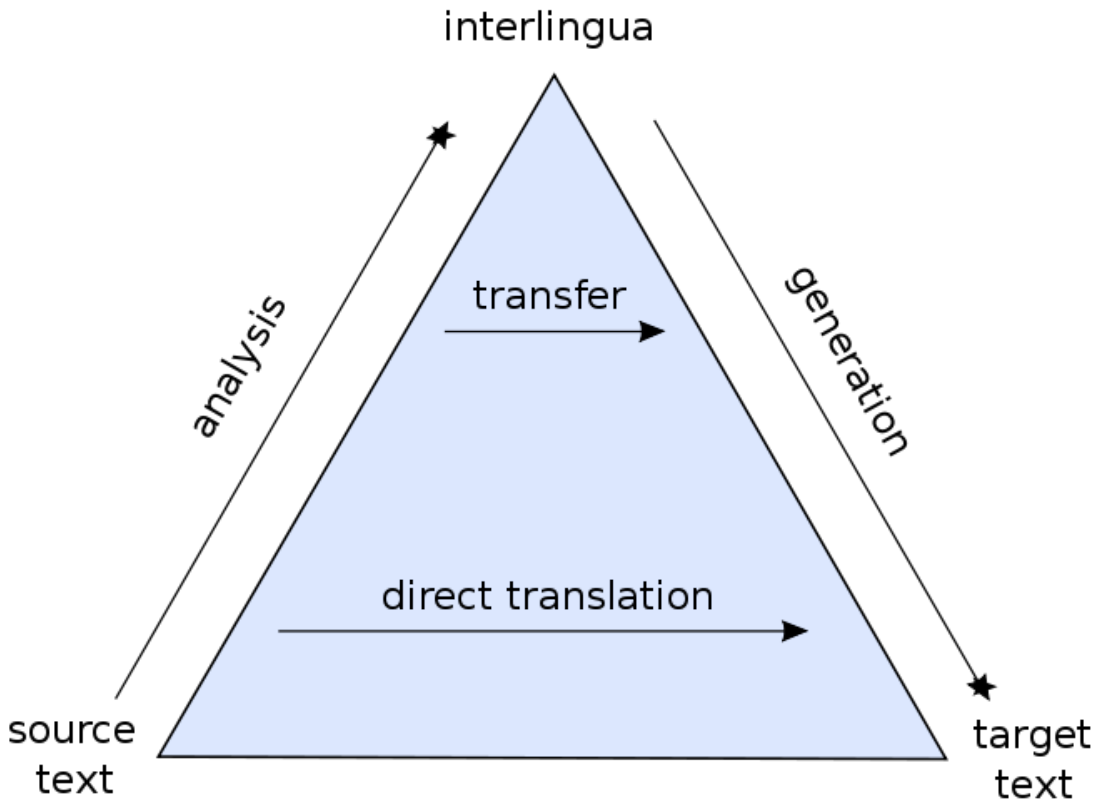
A brief history of MT

- 1940s-1950s: Early systems
- 1960s: ALPEC report (1966)
- 1970s: Operational systems: Systran, METEO
- 1980s: Eurotra project (EC-funded)
- Late 1980s–late1990s: the dawn of SMT (IBM), EBMT
- 2000s-early2010s: dominance of SMT
  - free on-line translation
- 2014-present: Neural MT

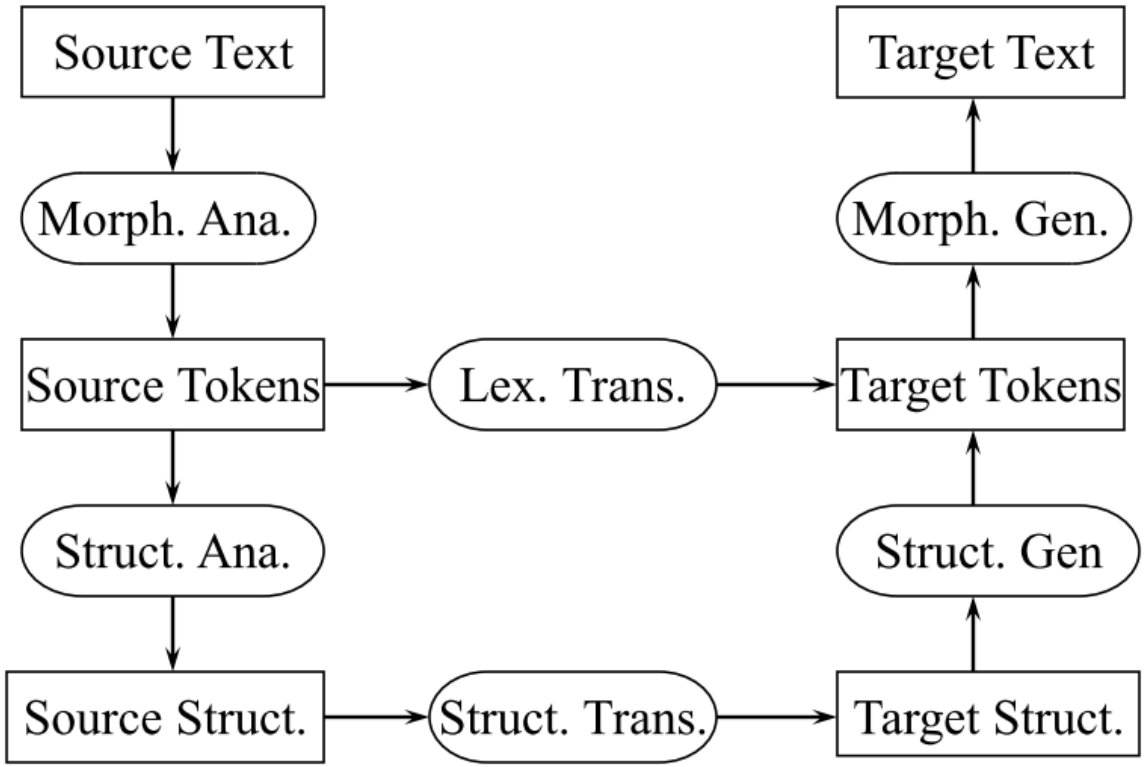
Different Approaches to MT

- Direct Translation
- Rule-based Machine Translation (RBMT)
- Memory-based Translation
- Example-based Machine Translation (EBMT)
- Statistical Machine Translation (SMT)
- Neural Machine Translation (NMT)

The Vauquois Pyramid

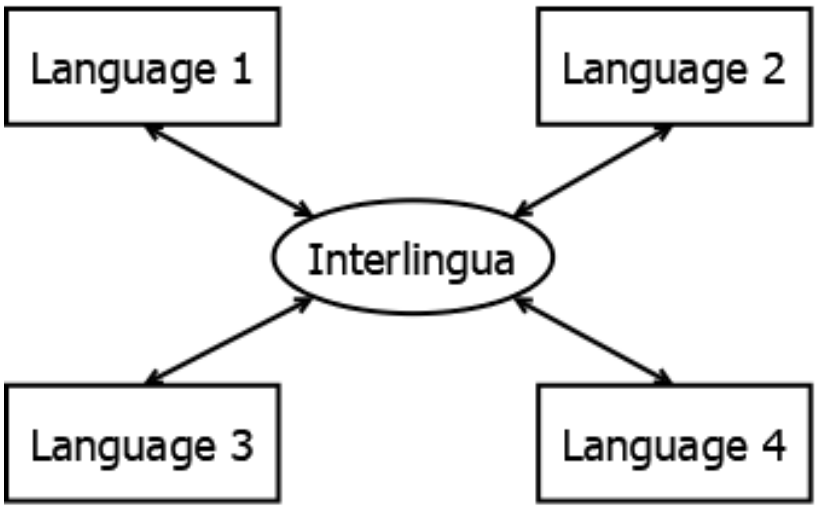


Transfer-based MT

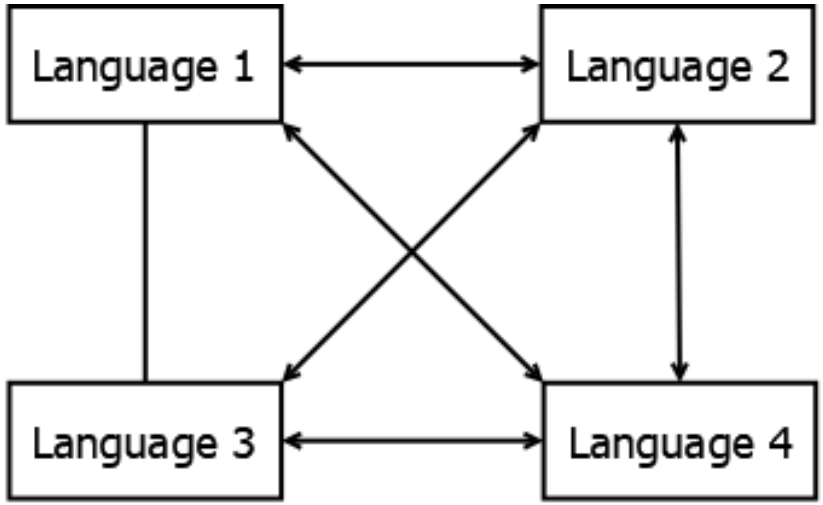


Transfer-based MT

Interlingua approach vs. transfer approach



Interlingua-based Approach



Transfer-based Approach

Multilingual MT

- Theoretically, interlingua-based approach is ideal because it requires much less components compared with transfer-based approach.
- However, using a human-defined interlingua (i.e. a specific knowledge expression) is practically too difficult.
- In business, a feasible way is to use a natural language (mostly English) as the interlingua, which is usually called a pivot language in this case.

Multilingual MT

- Theoretically, interlingua-based approach is ideal because it requires much less components compared with transfer-based approach.
- However, using a human-defined interlingua (i.e. a specific knowledge expression) is practically too difficult.
- In business, a feasible way is to use a natural language (mostly English) as the interlingua, which is usually called a pivot language in this case.

Multilingual MT

Makoto Nagao (Kyoto University):
When the pivot language (i.e. interlingua) is used, the results of the analytic stage must be in a form which can be utilized by all of the different languages into which translation is to take place. This level of subtlety is a practical impossibility. (Machine Translation, Oxford, 1989)

- In business, a feasible way is to use a natural language (mostly English) as the interlingua, which is usually called a pivot language in this case.

Multilingual MT

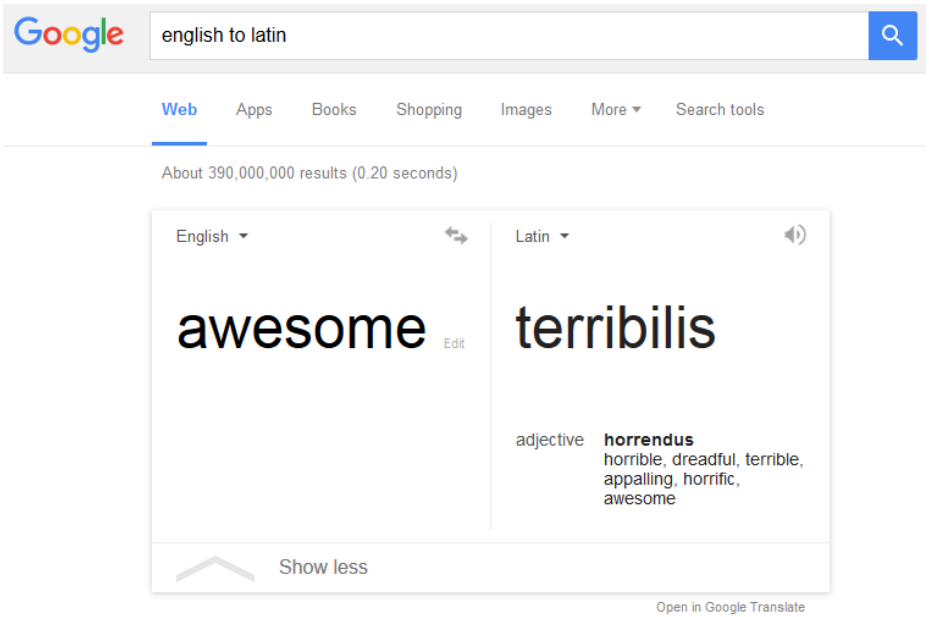
Patel-Schneider (METAL system):
METAL employs a modified transfer approach rather than an interlingua. If a meta-language (an interlingua) were to be used for translation purposes, it would need to incorporate all possible features of many languages. That would not only be an endless task but probably a fruitless one as well. Such a system would soon become unmanageable and perhaps collapse under its own weight. (A four-valued semantics for terminological reasoning, Artificial Intelligence, 38, 1989)

Multilingual MT

- Theoretically, interlingua-based approach is ideal because it requires much less components compared with transfer-based approach.
- However, using a human-defined interlingua (i.e. a specific knowledge expression) is practically too difficult.
- In business, a feasible way is to use a natural language (mostly English) as the interlingua, which is usually called a pivot language in this case.

Applications of MT

Since neural machine translation become popular, the quality of MT systems is greatly improved, and MT is used in more and more scenarios.



Web Translator

Applications of MT

Since neural machine translation become popular, the quality of MT systems is greatly improved, and MT is used in more and more scenarios.

Photo Translator

Applications of MT

Since neural machine translation become popular, the quality of MT systems is greatly improved, and MT is used in more and more scenarios.

Voice Translator

Applications of MT

Since neural machine translation become popular, the quality of MT systems is greatly improved, and MT is used in more and more scenarios.

Real-time Conference Translator

Content

1 Machine Translation (MT)

2 Machine translation evaluation

3 Statistical machine translation (SMT)

4 Neural machine translation (NMT) based on RNNs

5 Neural machine translation (NMT) with attentions

Machine translation evaluation is not easy

- Machine translation evaluation is not easy because corrent translations for the same text may differ a lot literally.
- Human evaluation:
  - Reliable
  - Inconsistent
  - Expensive
  - Slow
- Automatic evaluation:
  - Unreliable
  - Consistent
  - Cheap
  - Fast

Goals for Automatic MT Evaluation

- Meaningful: score should give intuitive interpretation of translation quality.
- Consistent: repeated use of metric should give same results.
- Correct: metric must rank better systems higher.
- Low cost: reduce time and money spent on carrying out evaluation.
- Tunable: automatically optimise system performance towards metric.

Automatic MT Evaluation Metrics

- Input:
  - output of machine translation systems.
  - reference translations give by human translators.
  - source text (optional).
- Output:
  - a score which represents the similarity between the MT output and the human reference given the source sentence.

Automatic MT Evaluation Metrics

- Plenty of automatic MT evaluation metrics are proposed by researchers, among which the most popular metrics include:
  - WER (Word Error Rate)
  - BLEU (BiLingual Evaluation Understudy)
  - TER (Translation Error Rate)
  - METEOR (Metric for Evaluation of Translation with Explicit ORdering)

BLEU: Motivation

- N-gram precision between the machine translation and the reference
- Compute n-gram precisions for 1 to n (typically $n = 4$)
- A harmony average of the n-gram precisions is calculated over different n's
- Compute on the entire test set, rather than single sentences, to avoid 0 value for higher n-gram precisions

BLEU: Definition

$$\text{BLEU} = \underbrace{\min(1, \frac{\textit{length\_of\_MT}}{\textit{length\_of\_reference}})}_{\text{Brevity Penalty}} \times \underbrace{(\prod_{i=1}^{n} \text{Precision}_i)^{\frac{1}{n}}}_{\text{N-gram Precision}}$$

$$\text{Precision}_i = \frac{\textit{clipped\_number\_of\_matched\_}\{i\}\textit{grams\_in\_MT}}{\textit{number\_of\_total\_}\{i\}\textit{grams\_in\_MT}}$$

- Brevity Penalty is used to avoid that a very short MT get a high BLEU score.
- The number of matched {*i*}grams is *clipped* to avoid that a single word in the reference is matched multiple times in MT.

Multiple References

To avoid translation variability, multiple references can by used:

- n-grams may match in any of the references
- closest reference length is used for brevity penalty

BLEU: An example

- **Candidate 1: the book is on the desk**

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

trigram:

$Count_{clip}(the, book, is) = 1$

$Count_{clip}(book, is, on) = 1$

$Count_{clip}(is, on, the) = 1$

$Count_{clip}(on, the, desk) = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

trigram:

$Count_{clip}(the, book, is) = 1$

$Count_{clip}(book, is, on) = 1$

$Count_{clip}(is, on, the) = 1$

$Count_{clip}(on, the, desk) = 1$

$\sum_{trigram \in C} Count(trigram) = 4$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

trigram:

$Count_{clip}(the, book, is) = 1$

$Count_{clip}(book, is, on) = 1$

$Count_{clip}(is, on, the) = 1$

$Count_{clip}(on, the, desk) = 1$

$\sum_{trigram \in C} Count(trigram) = 4$

$p_3 = 1$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

trigram:

$Count_{clip}(the, book, is) = 1$

$Count_{clip}(book, is, on) = 1$

$Count_{clip}(is, on, the) = 1$

$Count_{clip}(on, the, desk) = 1$

$\sum_{trigram \in C} Count(trigram) = 4$

$p_3 = 1$

$$\left.\begin{array}{l} c = 6 \\ r = 6 \end{array}\right\} = e^{1-\frac{r}{c}} = e^0 = 1 = BP$$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

BLEU: An example

- **Candidate 1: the book is on the desk**

- **Reference 1: there is a book on the desk**
- **Reference 2: the book is on the table**

unigram:

$Count_{clip}(the) = 2$

$Count_{clip}(book) = 1$

$Count_{clip}(is) = 1$

$Count_{clip}(on) = 1$

$Count_{clip}(desk) = 1$

$\sum_{unigram \in C} Count(unigram) = 6$

$p_1 = 1$

bigram:

$Count_{clip}(the, book) = 1$

$Count_{clip}(book, is) = 1$

$Count_{clip}(is, on) = 1$

$Count_{clip}(on, the) = 1$

$Count_{clip}(the, desk) = 1$

$\sum_{bigram \in C} Count(bigram) = 5$

$p_2 = 1$

trigram:

$Count_{clip}(the, book, is) = 1$

$Count_{clip}(book, is, on) = 1$

$Count_{clip}(is, on, the) = 1$

$Count_{clip}(on, the, desk) = 1$

$\sum_{trigram \in C} Count(trigram) = 4$

$p_3 = 1$

$$\left.\begin{matrix} c = 6 \\ r = 6 \end{matrix}\right\} = e^{1-\frac{r}{c}} = e^0 = 1 = BP$$

$$\begin{aligned} \text{BLEU} &= BP \times (p_1 \times p_2 \times p_3)^{\frac{1}{3}} \\ &= 1 \times (1 \times 1 \times 1)^{\frac{1}{3}} = 1 \end{aligned}$$

This example is extracted from Dr. Ming Zhou's slides on MT Evaluation

Content

1 Machine Translation (MT)

2 Machine translation evaluation

3 Statistical machine translation (SMT)

4 Neural machine translation (NMT) based on RNNs

5 Neural machine translation (NMT) with attentions

Content

3 Statistical machine translation (SMT)
- SMT: basic ideas
- Word-based Translation Models
- Phrase-based Translation Models
- Decoding Algorithms

Motivation

- In the time of rule-based MT, the translation knowledge was explicitly encoded by linguists, which is labor intensive, time consuming and inefficient.
- The basic idea of statistical machine translation is to learn a machine translation system directly from parallel translation data, without involving linguistics to encode translation knowledge.

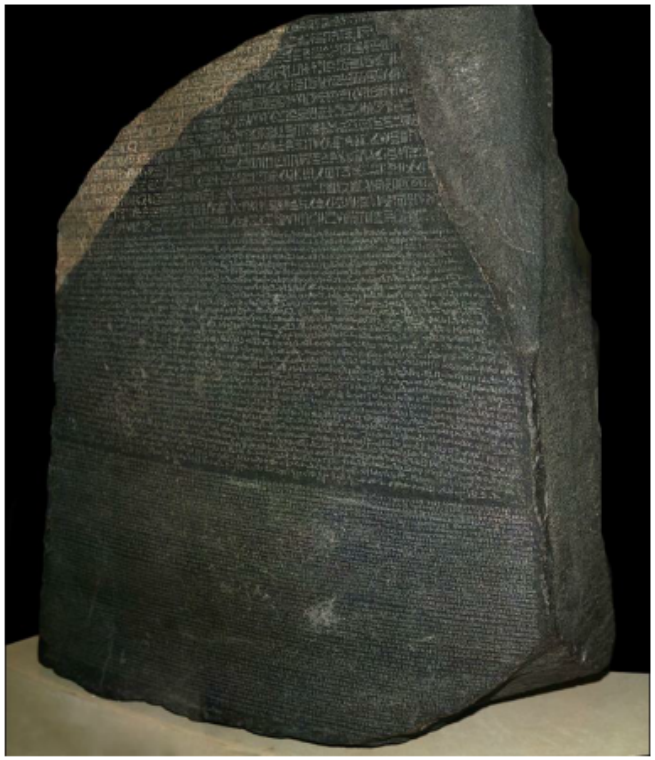
A brief history of SMT

- The first statistical machine translation was conducted by IBM researchers in late 1980s to early 1990s.
- The research of SMT was not very active in 1990s because the computing resources used by IBM researchers were higher than most researchers could have.
- In 1999, a group of researchers gathered in the JHU summer workshop and repeated IBM's early work successfully and release GIZA++, the implementation of the training algorithm of IBM models.
- After 2000 SMT became a dominate MT paradigm until it was replaced by NMT in recent years.

Parallel corpus

- A parallel corpus, or a bitext, is a collection of texts in one language and its corrending translation in another language.
- Parallel corpara used for MT research are normally aligned at sentence level.
- Document-level MT needs parallel corpara aligned in both word level and document level.
- Many parallel corpara are aligned only in document level in their original form, so a technique called sentence alignement is developed to align the sentences in a document aligned corpus.

Rosette Stone



The Rosetta Stone	
Material	Granodiorite
Size	114.4 × 72.3 × 27.93 cm (45 x 28.5 x 11 in)
Writing	Ancient Egyptian hieroglyphs, Demotic script, and Greek script
Created	196 BC
Discovered	1799
Present	British Museum

Rosette stone is a good example that it is feasible to learn translation knowledge from a parallel corpus.

Statistical Translation Model

A statistical translation model is the probability that a target text e is the translation of a given source text f:

$$p(e|f), \text{where} \sum_{e} p(e|f) = 1$$

Statistical Machine Translation

Given a statistical translation $p(e|f)$, the machine translation problem can be transfered to a search problem: to search a target sentence e which has the highist translation probability given f:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e|f)$$

Noisy Channel Model

- Statistical machine translation with a single translation model does not work well.
- IBM researchers proposed a Noisy Channel Model for statistical machine translation, which includes an additional language model.

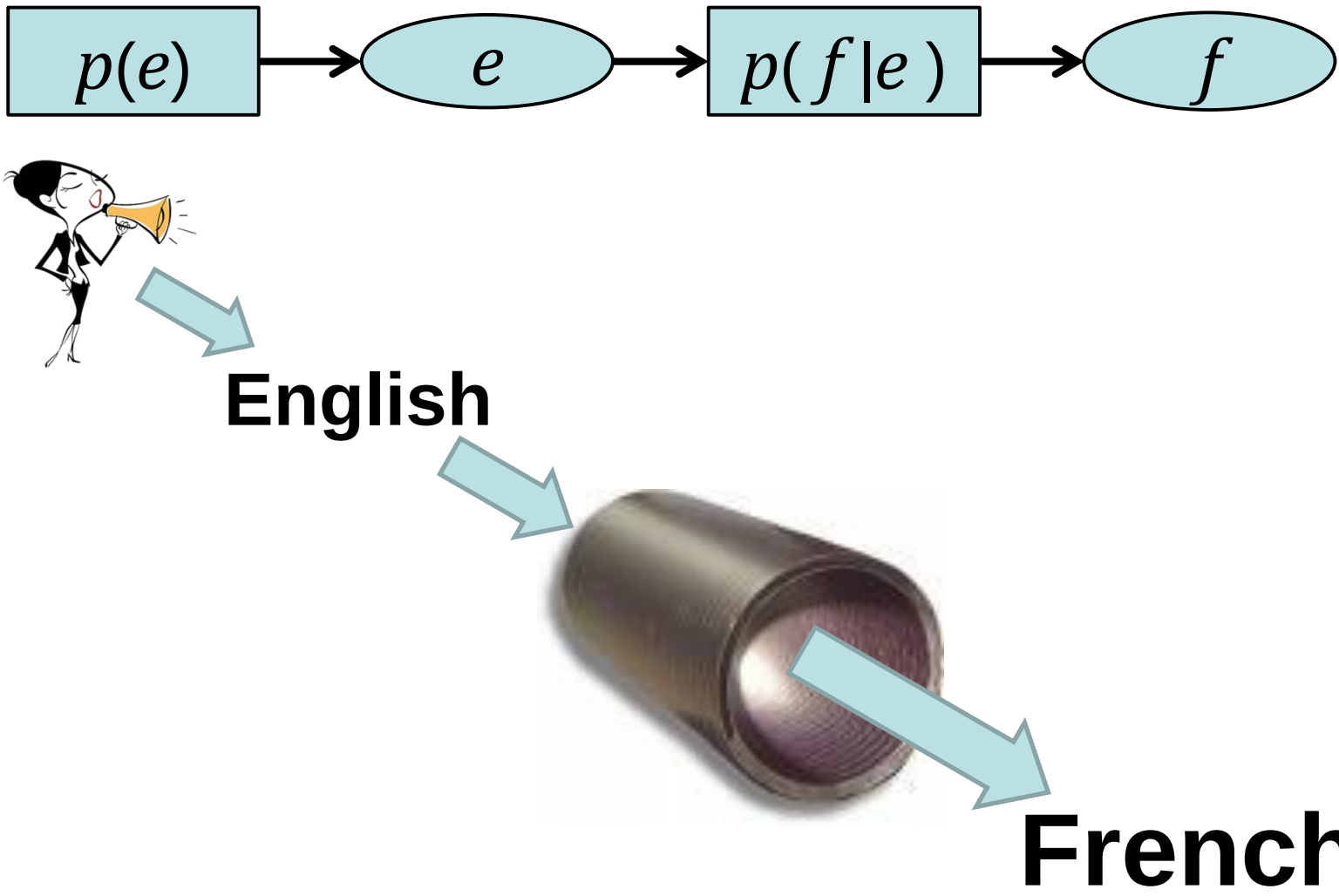
> Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, Paul S. Roossin, A Statistical Approach to Machine Translation, Computational Linguistics,1990

Noisy Channel Framework

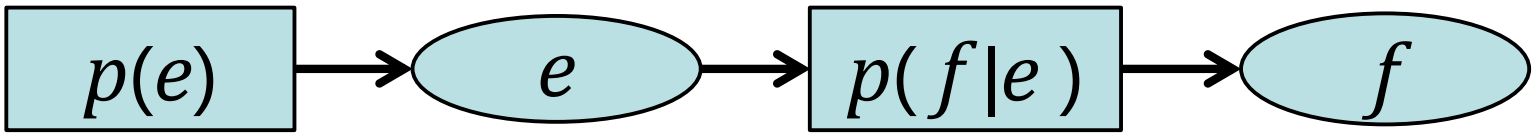
English

French

Noisy Channel Framework



Noisy Channel Framework



$$p(f) = p(e)p(f|e)$$

English

French

Noisy Channel Framework

Applying Bayes' Rule, we have:

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

Thus:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e|f) = \underset{e}{\operatorname{argmax}}\, p(e)p(f|e)$$

Noisy Channel Framework

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e)p(f|e)$$

Noisy Channel Framework

$$\hat{e} = \underbrace{\operatorname{argmax}}_{e} \underbrace{p(e)} \underbrace{p(f|e)}$$

Translation Model

Decoder

Language Model

Noisy Channel Framework

- The *translation model* models how likely it is that f is a translations of e – **adequacy**.
- The *language model* models how likely it is that e is an acceptable sentence – **fluency**.
- The *decoder* searches for the most likely e.

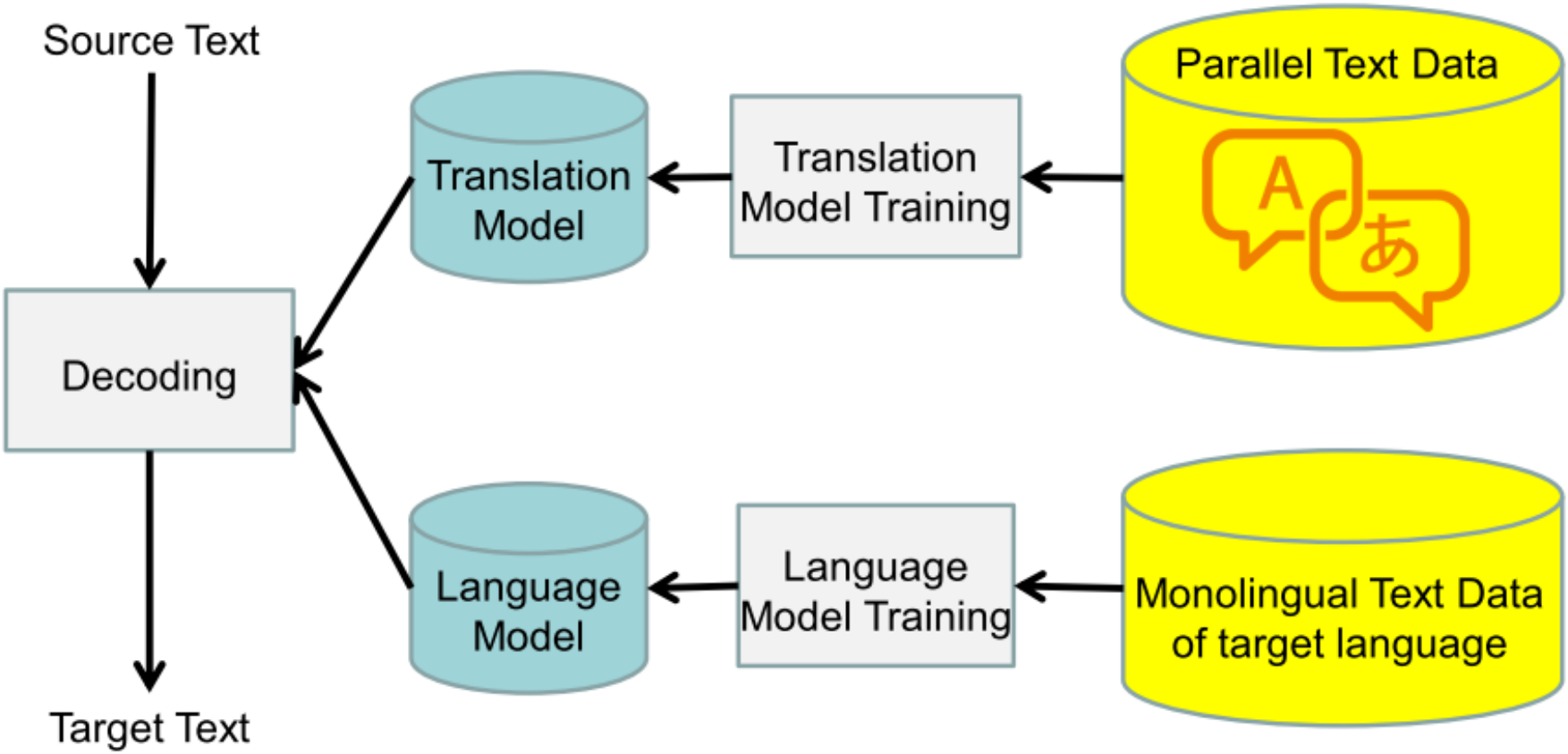
We have introduced language models in previouse lectures, here we will mainly focus on translation models and decoding algorithms

Noisy Channel Framework

- The *translation model* models how likely it is that f is a translations of e – **adequacy**.
- The *language model* models how likely it is that e is an acceptable sentence – **fluency**.
- The *decoder* searches for the most likely e.

We have introduced language models in previouse lectures, here we will mainly focus on translation models and decoding algorithms

SMT Workflow



Content

3 Statistical machine translation (SMT)
- SMT: basic ideas
- Word-based Translation Models
- Phrase-based Translation Models
- Decoding Algorithms

Categories of translation models

Various translation models have been proposed, which belong to different categories, according to the language units on which they are built up:

- Word-based models
  - IBM models 1-5
  - HMM models
- Phrase-based models
- Syntax-based models
  - Tree-to-string models
  - String-to-tree models
  - Tree-to-tree models
  - Dependency-based models

IBM Models

IBM researchers proposed 5 models with increased complexity:

- IBM Model 1: only consider lexical translation probabilities
- IBM Model 2: add a absolute reordering model
- IBM Model 3: add a fertility model
- IBM Model 4: add a relative reordering model
- IBM Model 5:

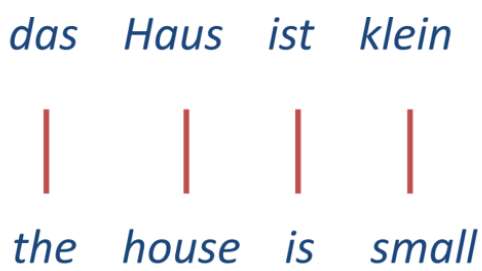
Lexical translation probabilities

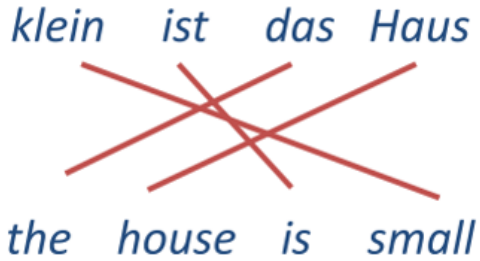
English	Chinese	Prob.
a	一	0.2
a	一个	0.4
a	个	0.2
a	一只	0.1
a	一本	0.05
a	...	...

English	Chinese	Prob.
book	书	0.7
book	预定	0.2
book	...	...
take	拿	0.4
take	带走	0.3
take	...	...

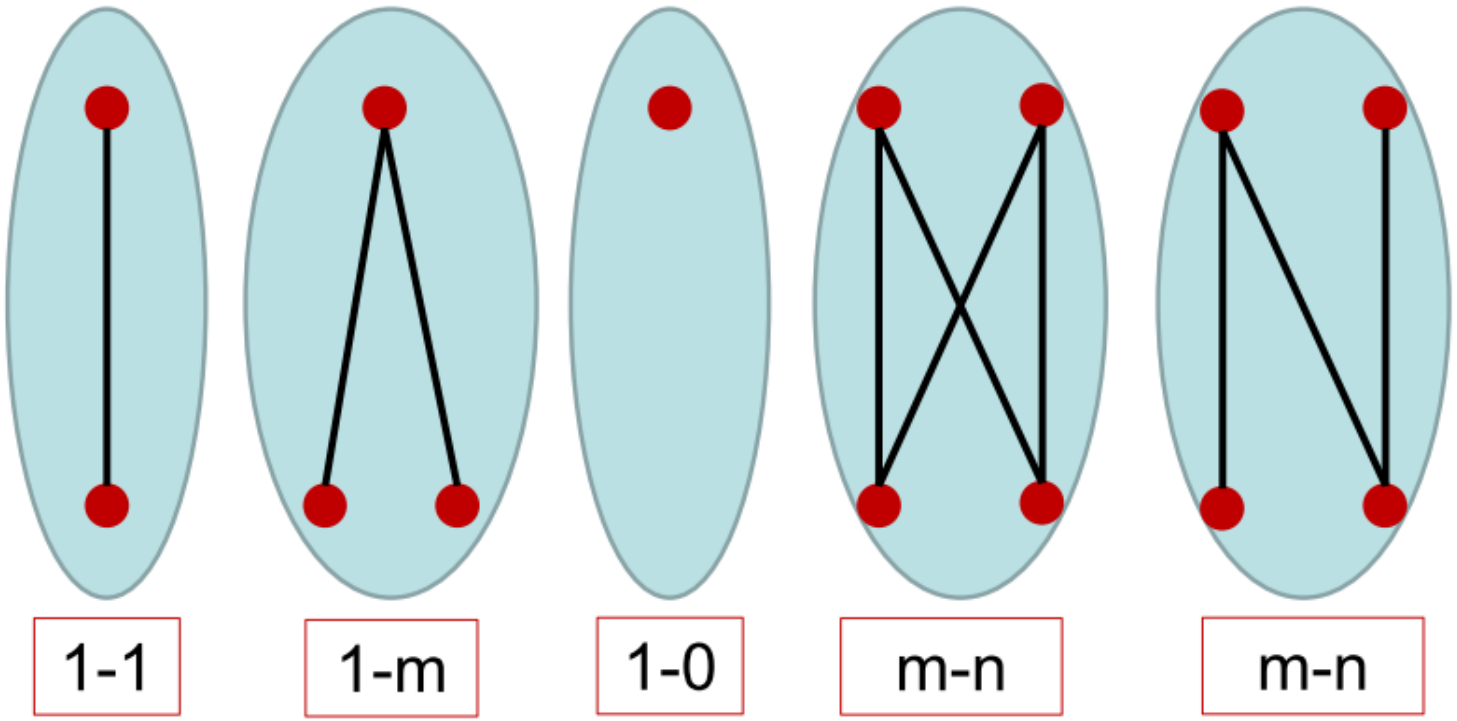
Word alignment

- To estimate the word translation probabilities, we need alignment between words in the parallel sentences

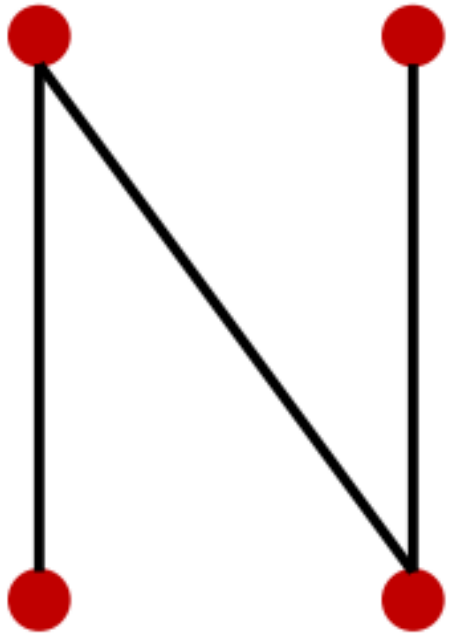




Word alignment patterns

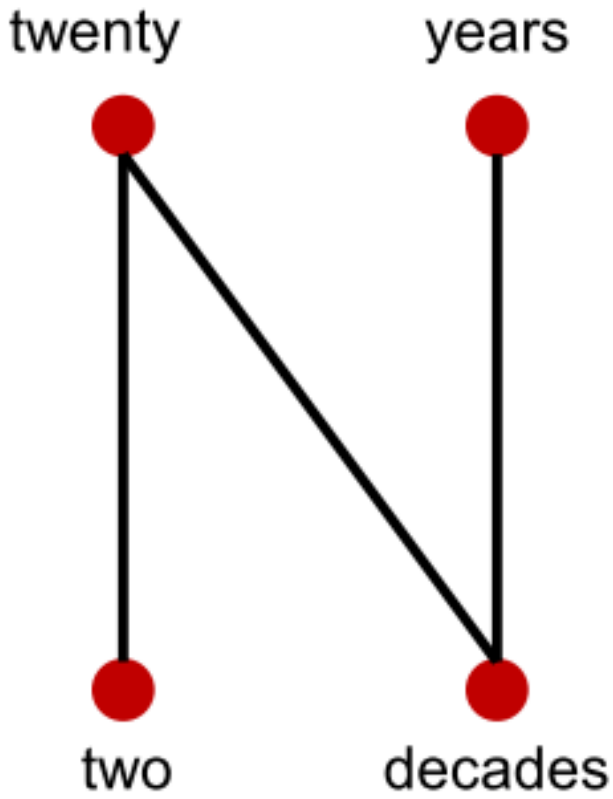


Word alignment patterns



Can you image a word alignment pattern like this?

Word alignment patterns



Can you image a word alignment pattern like this?

Learning lexical translation models

- We would like to estimate the lexical translation probabilities from a parallel corpus...
- but we do not have the alignments:
  - If we had the alignments, we could estimate the lexical translation probabilities.
  - If we had the probabilities, we could estimate the alignments.

Learning lexical translation models

- We would like to estimate the lexical translation probabilities from a parallel corpus...
- but we do not have the alignments:
  - If we had the alignments, we could estimate the lexical translation probabilities.
  - If we had the probabilities, we could estimate the alignments.

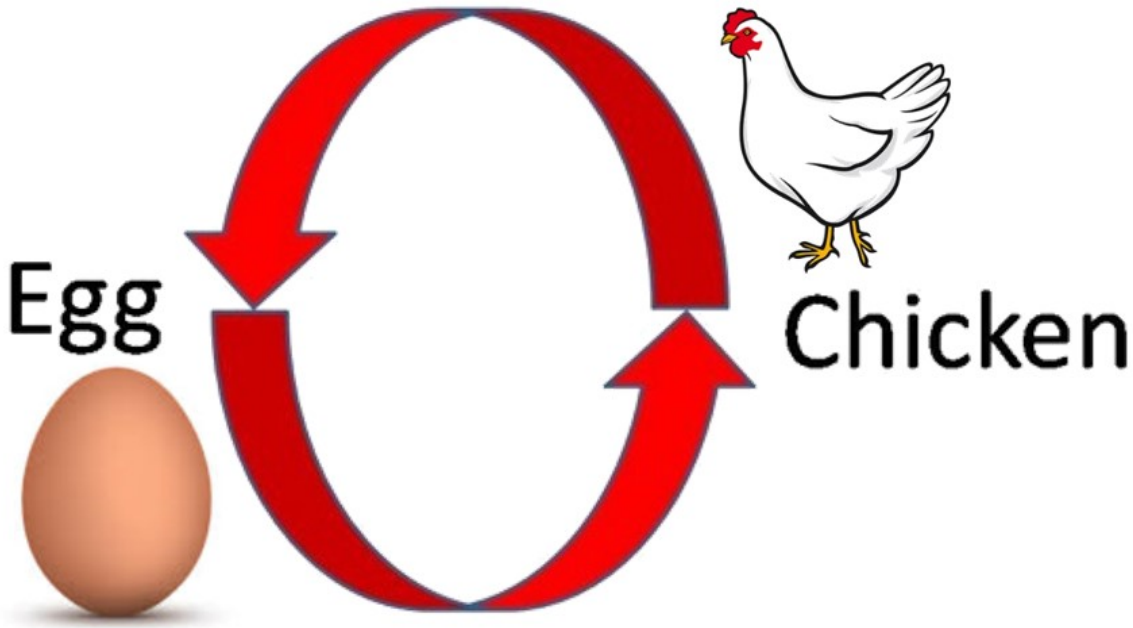
Learning lexical translation models

- We would like to estimate the lexical translation probabilities from a parallel corpus...
- but we do not have the alignments:
  - If we had the alignments, we could estimate the lexical translation probabilities.
  - If we had the probabilities, we could estimate the alignments.

Learning lexical translation models

- We would like to estimate the lexical translation probabilities from a parallel corpus...
- but we do not have the alignments:
  - If we had the alignments, we could estimate the lexical translation probabilities.
  - If we had the probabilities, we could estimate the alignments.

A Paradox



EM Algorithm

- Incomplete data
  - If we had complete data, we could estimate model.
  - If we had the model, we could fill in the gaps in the data.
- Solution: Expectation Maximization (EM) Algorithm
  - Initialize model parameters. (e.g. uniform)
  - Assign probabilities to the missing data. (E-step)
  - Astimate model parameters from completed data. (M-step)
  - Iterate E-step and M-step until the model converges.

How does EM algorithm work?

EM Algorithm consists of two steps:

1. Expectation-Step: Apply model to the data
   - parts of the data are hidden (here: alignments)
   - using the model, assign probabilities of the hidden data to possible values (alignments)
2. Maximization-Step: Estimate new model from data
   - take assigned values as fact
   - collect counts (weighted by probabilities)
   - estimate new model from counts

Iterate the E-step and the M-step until convergence

Example

Consider a parallel corpus containing just two pairs:

blue house	house
maison bleu	maison

How many possible alignments in the first pair?

How many in the second pair?

Example

Consider a parallel corpus containing just two pairs:

blue house	house
maison bleu	maison

How many possible alignments in the first pair?

How many in the second pair?

We will simplify the example by ruling out many-to-one or zero-to-one alignments.

Example

Consider a parallel corpus containing just two pairs:

blue house	house
maison bleu	maison

How many possible alignments in the first pair? 2

How many in the second pair? 1

We will simplify the example by ruling out many-to-one or zero-to-one alignments.

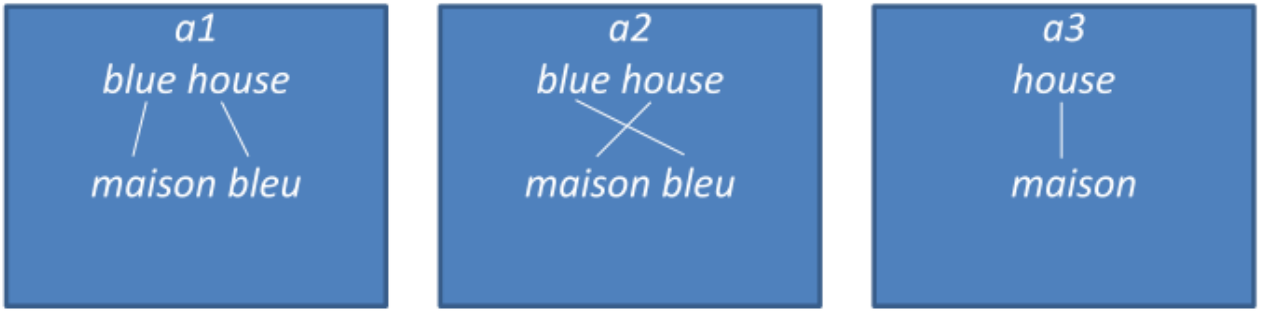
Step 1 (Initialisation)

Set parameter values uniformly.

- t(bleu|house) = 1/2
- t(maison|house) = 1/2
- t(bleu|blue) = 1/2
- t(maison|blue) = 1/2

Step 2 (Expectation)

Compute the probability of all alignments.



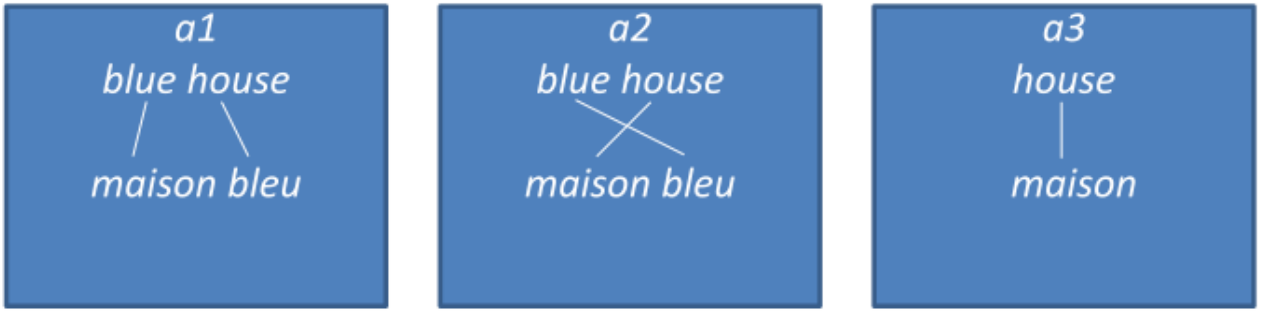
p(a1, maison bleu|blue house) = t(maison|blue) * t(bleu|house) = ½ * ½ = ¼

p(a2, maison bleu|blue house) = t(maison|house) * t(bleu|blue) = ½ * ½ = ¼

p(a3, maison| house) = t(maison|house) = ½

Step 3 (Expectation)

Normalise for all alignments.



p(a1|maison bleu,blue house) = 1/4 ÷ 2/4 = 1/2

p(a2|maison bleu,blue house) = 1/4 ÷2/4 = 1/2

p(a3|maison, house) = 1/2 ÷ 1/2 = 1

Step 4 (Maximisation)

Collect fractional counts

- tc(bleu|house) = 1/2
- tc(maison|house) = 1/2 + 1 = 3/2
- tc(bleu|blue) = 1/2
- tc(maison|blue) = 1/2

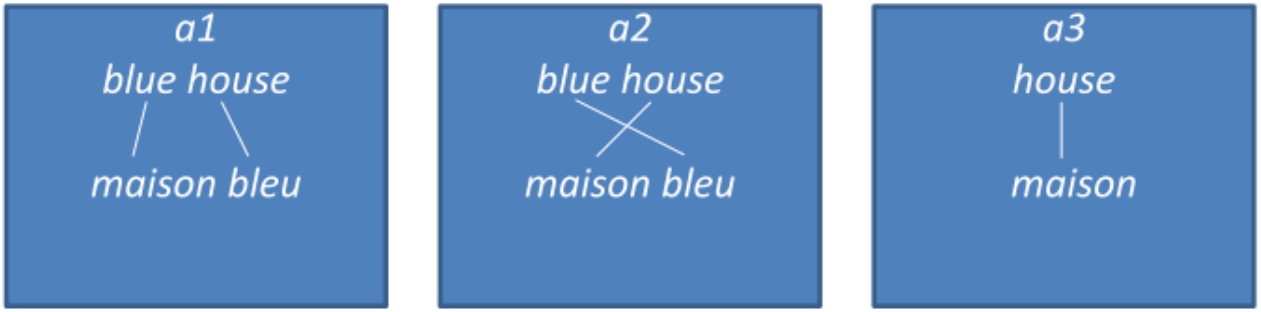
Step 5 (Maximisation)

Normalise fractional counts to yield revised parameter values

- t(bleu|house) = 1/2 ÷ 2 = 1/4
- t(maison|house) = 3/2 ÷ 2 = 3/4
- t(bleu|blue) = 1/2 ÷ 1 = 1/2
- t(maison|blue) = 1/2 ÷ 1 = 1/2

Repeat Step 2 (Expectation)

Compute the probability of all alignments.



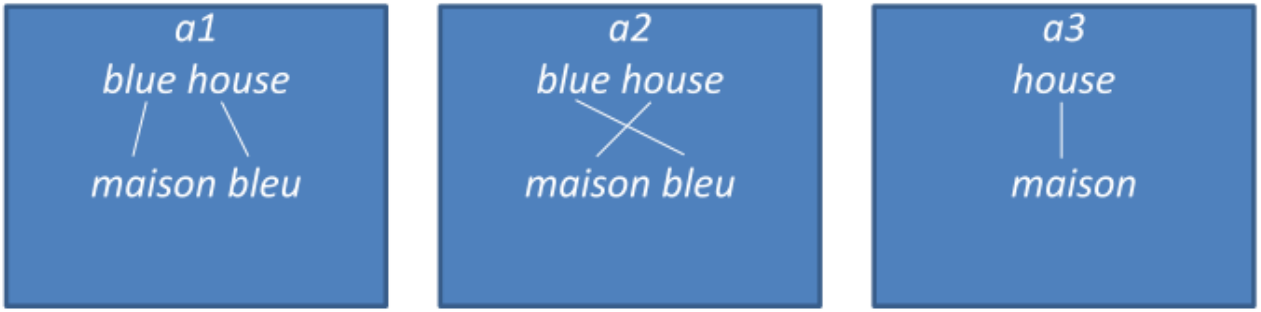
p(a1, maison bleu|blue house) = t(maison|blue) * t(bleu|house) = 1/2 * 1/4 = 1/8

p(a2, maison bleu|blue house) = t(maison|house) * t(bleu|blue) = 3/4 * 1/2 = 3/8

p(a3, maison| house) = t(maison|house) = 3/4

Repeat Step 3 (Expectation)

Normalise for all alignments.



p(a1|maison bleu,blue house) = 1/8 ÷ 4/8 = 1/4

p(a2|maison bleu,blue house) = 3/8 ÷ 4/8 = 3/4

p(a3|maison, house) = 3/4 ÷ 3/4 = 1

Repeat Step 4 (Maximisation)

Collect fractional counts

- tc(bleu|house) = 1/4
- tc(maison|house) = 3/4 + 1 = 7/4
- tc(bleu|blue) = 3/4
- tc(maison|blue) = 1/4

Repeat Step 5 (Maximisation)

Normalise fractional counts to yield revised parameter values

- t(bleu|house) = 1/4 ÷ 2 = 1/8
- t(maison|house) = 7/4 ÷ 2 = 7/8
- t(bleu|blue) = 3/4 ÷ 1 = 3/4
- t(maison|blue) = 1/4 ÷ 1 = 1/4

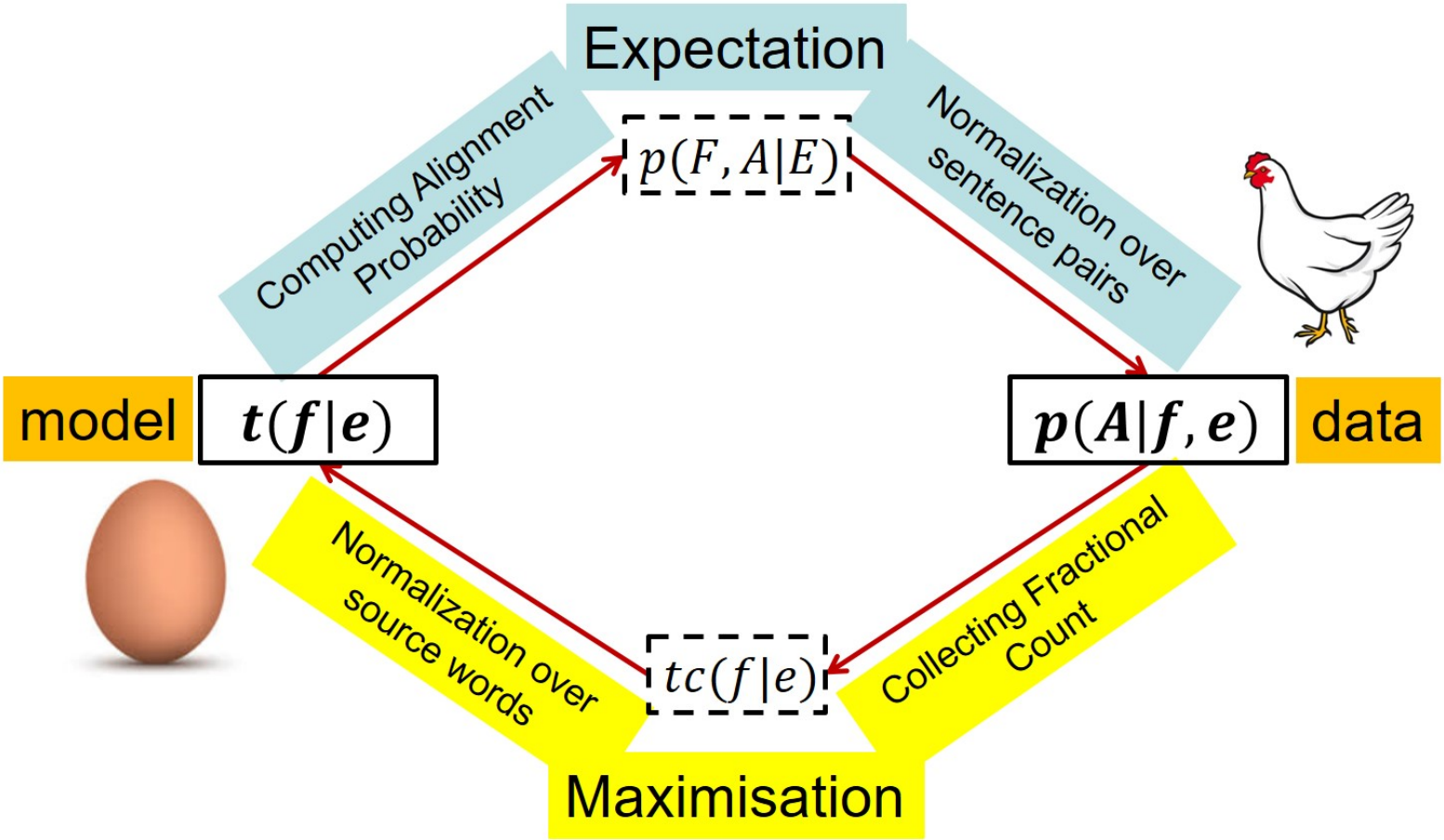
Convergence

Repeating steps 2, 3, 4 and 5 eventually yields:

- t(bleu|house) = 0.0001
- t(maison|house) = 0.9999
- t(bleu|blue) = 0.9999
- t(maison|blue) = 0.0001

It is proved that an EM algorithm is convergent.

EM Algorithm



Content

3 Statistical machine translation (SMT)
- SMT: basic ideas
- Word-based Translation Models
- Phrase-based Translation Models
- Decoding Algorithms

Shortcomings of word-based SMT

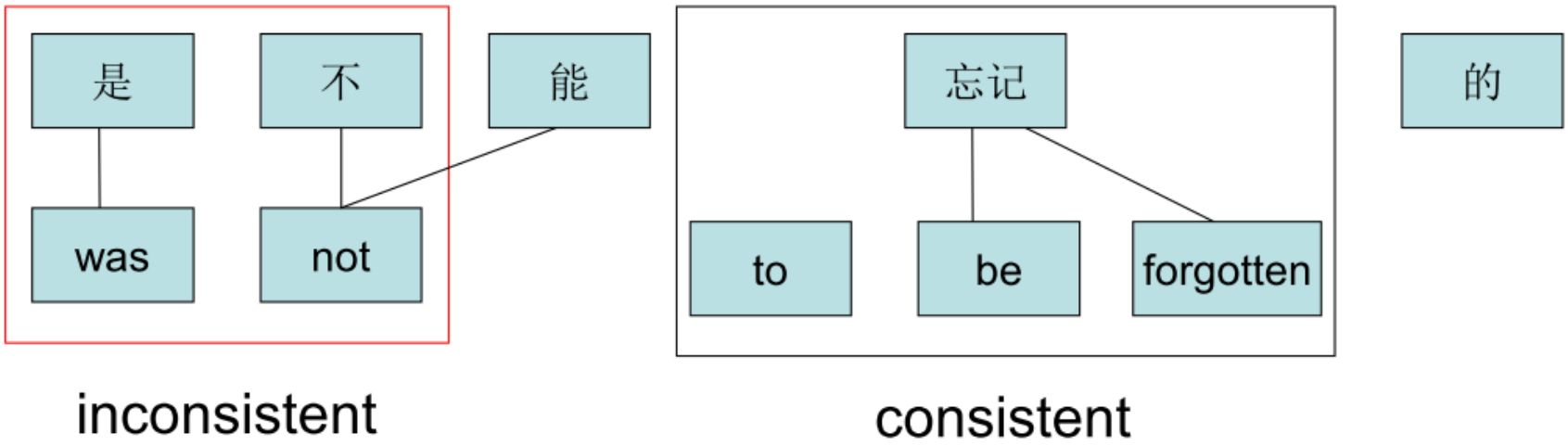
- Word-based translation models do not take into account contextual information for translation decisions
- They are not good at dealing with 1-to-many, many-to-1 and many-to-many translations.

Phrase-based Translation Models

- Phrase-based translate models are proposed to solve the problems for word-based models.
- Phrase-based models translate phrases as atomic units.
- A monolingual phrase can be any contiguous sequence of words in a sentence.
  - A phrase is not necessarily syntactically well-formed
  - A phrase is not necessarily semantically meaningful
- A bilingual phrase pair should be consistent with word alignment.

Bilingual Phrase Pairs

A bilingual phrase pair should be constistent with word alignment:



Bilingual Phrase Pairs

A real example taken from Europarl for the German phrase den Vorschlag:

English	Probability	English	Probability
the proposal	0.6277	the suggestions	0.0114
's proposal	0.1068	the proposed	0.0114
a proposal	0.0341	the motion	0.0091
the idea	0.025	the idea of	0.0091
this proposal	0.0227	the proposal ,	0.0068
proposal	0.0205	its proposal	0.0068
of the proposals	0.0159	it	0.0068
the proposals	0.0159	.....	

Learning a phrase translation table

- Task: learn the model from a parallel corpus
- Three stages:
  1. Word alignment: using IBM models or other method
  2. Extraction of phrase pairs
  3. Scoring phrase pairs

Bidirectional word alignment

- With IBM models, each target word can be aligned to at most one source word (patterns supported: 1-0,0-1,1-1,m-1).
- Therefore, it's not possible to end up with an alignment of one target word to many source words (patterns not supported: 1-m, m-m)
- To obtain a word alignment with all possible patterns, a symmetric word alignment algorithm should be adopted.

Bidirectional word alignment

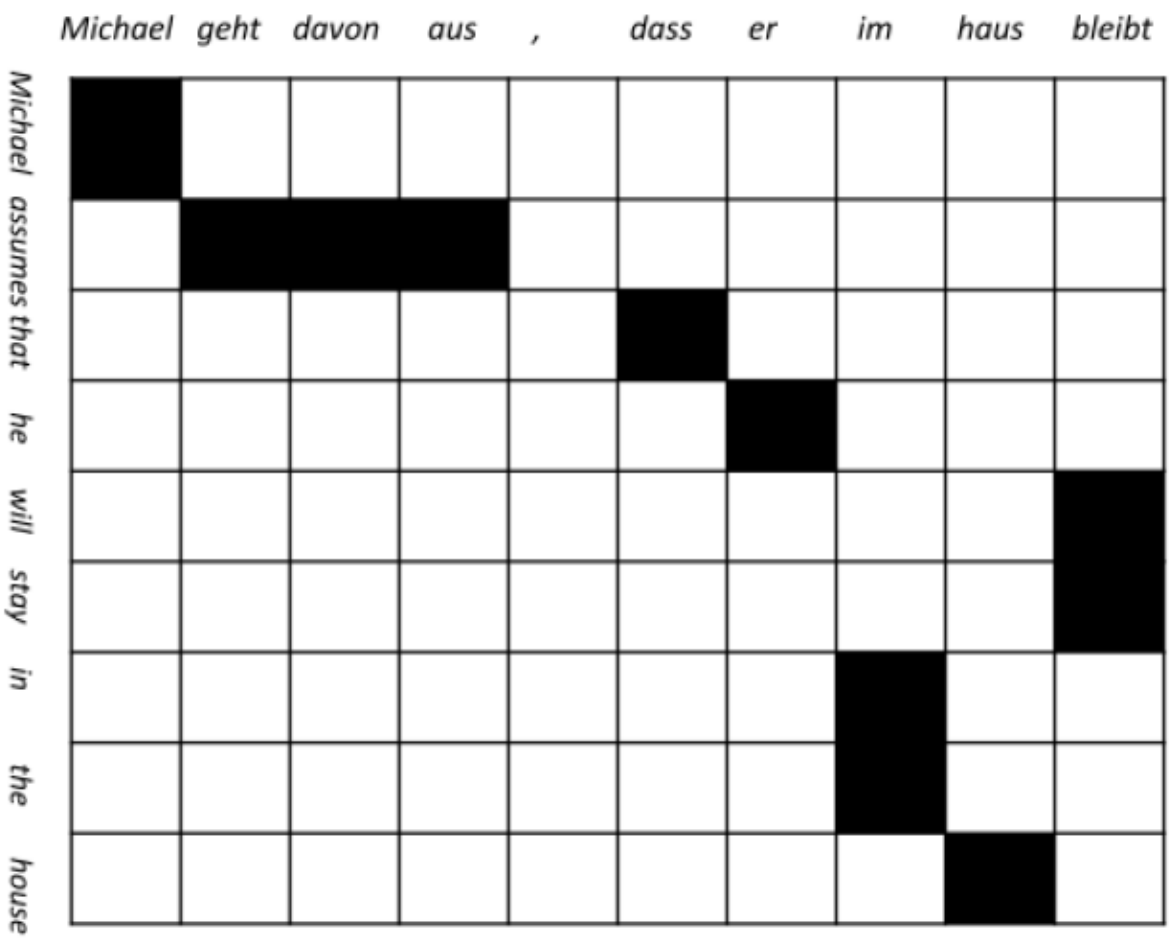
- A typical symmetric word alignment algorithm:
  - Word alignment using IBM Models in one direction.
  - Word alignment using IBM Models in the other direction.
  - Merge the above two alignment results with a certain criterion.

Consistent with word alignment

A phrase pair (e, f) is consistent with a bidirectional word alignment A if and only if:

- For all words $e_i \in e$, if exists an f_j: $(e_i, f_j) \in A$, then $f_j \in f$.
- For all word $f_j \in f$, if exists an e_i: $(e_i, f_j) \in A$, then $e_i \in e$.
- There exists an $e_i \in e$, and an $f_j \in f$: $(e_i, f_j) \in A$

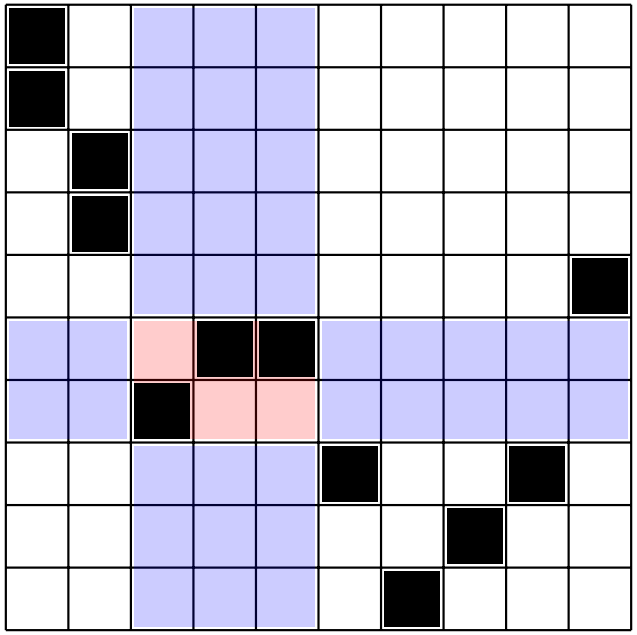
A matrix view of word alignment



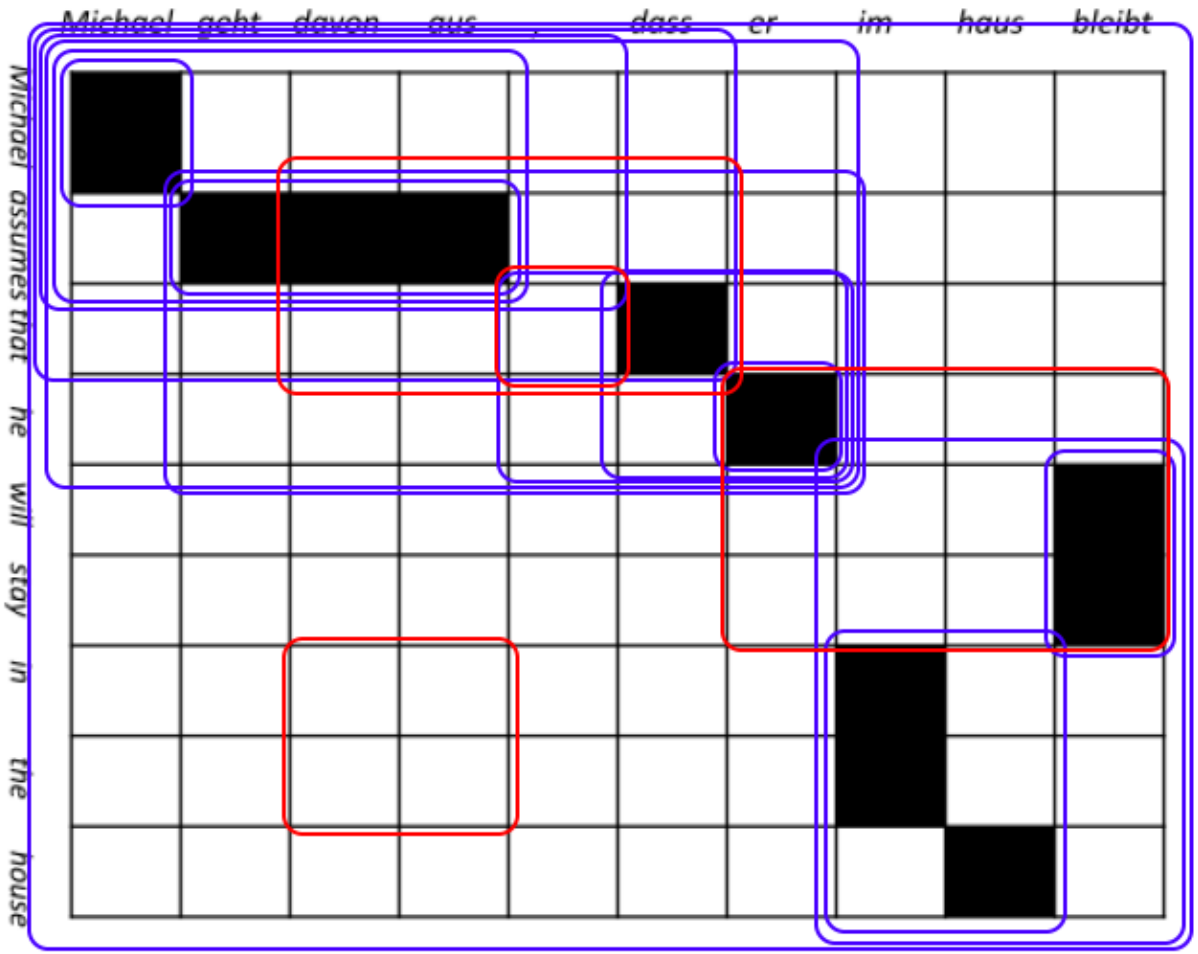
Consistent phrases in the matrix view

A consistent phrase pair defined by the red area should meet the following requirement:

- There should be one or more filled blocks in the red area.
- The blue areas should be all clear.



Phrase pair extraction



Blue box: consistent phrase pairs, Red box: inconsistent phrase pairs

Phrase pair extraction

Phrase pairs extracted from the above example:

- michael assumes | michael geht davon aus ,
- michael assumes | michael geht davon aus
- assumes that | geht davon aus , dass
- assumes that he | geht davon aus , dass er
- that he | , dass er
- that he | dass er
- in the house | im haus
- michael assumes that | michael geht davon aus , dass
- michael assumes that he | michael geht davon aus , dass er
- michael assumes that he will stay in the house | michael geht davon aus , dass er im haus bleibt
- assumes that he will stay in the house | geht davon aus , dass er im haus bleibt
- that he will stay in the house | dass er im haus bleibt ,
- that he will stay in the house | dass er im haus bleibt
- he will stay in the house | er im haus bleibt
- will stay in the house | im haus bleibt

Scoring Phrase Pairs

- Phrase pair scoring: assign probabilities to all the phrase pairs extracted from the aligned corpus
- Scoring by relative probability:

$$\phi(f|e) = \frac{count(e,f)}{count(e)} = \frac{count(e,f)}{\sum_{f_i} count(e,f_i)}$$

Content

3 Statistical machine translation (SMT)
- SMT: basic ideas
- Word-based Translation Models
- Phrase-based Translation Models
- Decoding Algorithms

Decoding

Decoding is to search for the most likely target sentence e for a given source sentence f:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e)p(f|e)$$

Decoding processing

Maria no dio una bofetada a la bruja verde

Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing

Maria no dio una bofetada a la bruja verde

Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

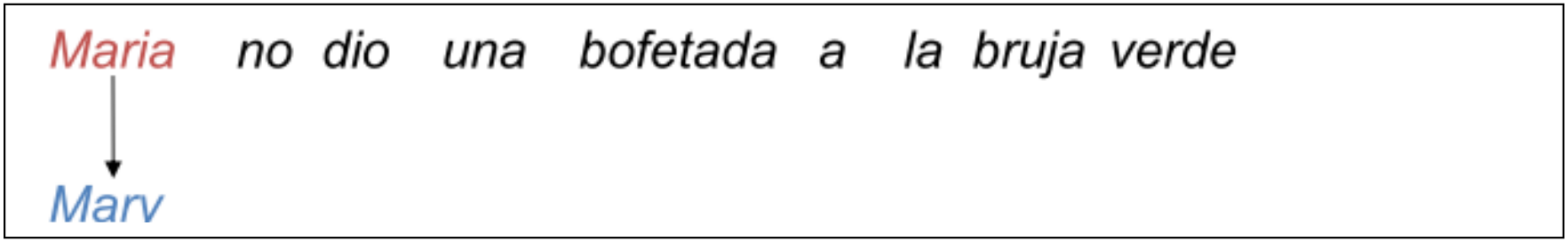
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing

Maria no dio una bofetada a la bruja verde

Mary

Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing

Maria no dio una bofetada a la bruja verde

Mary

Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

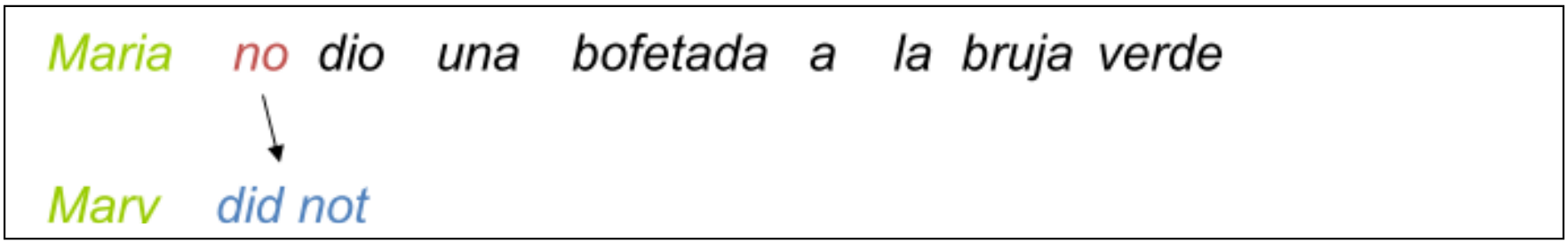
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

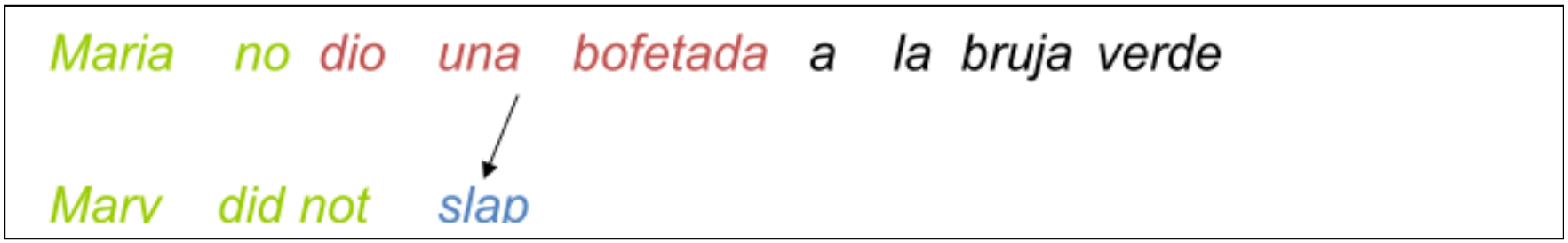
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

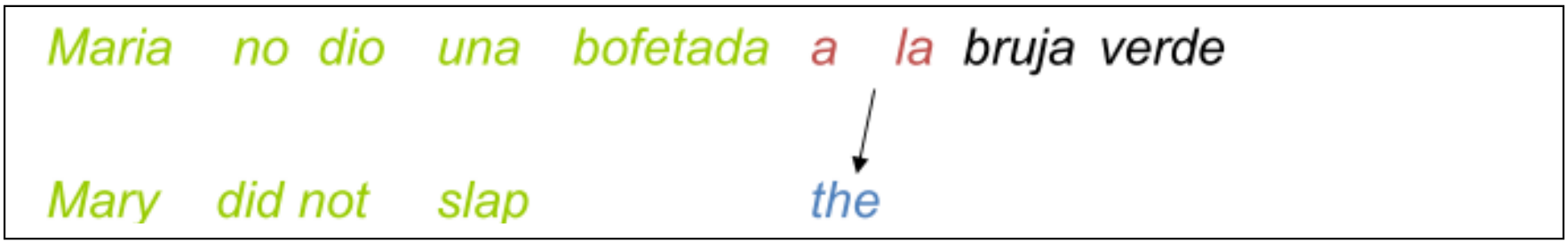
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

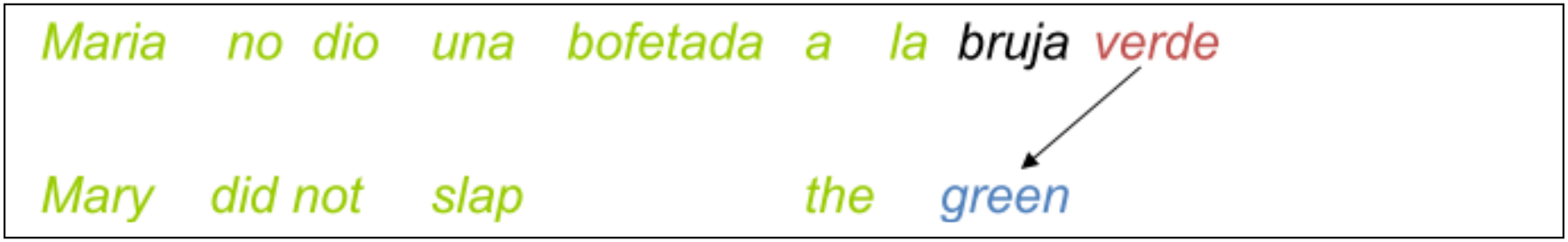
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

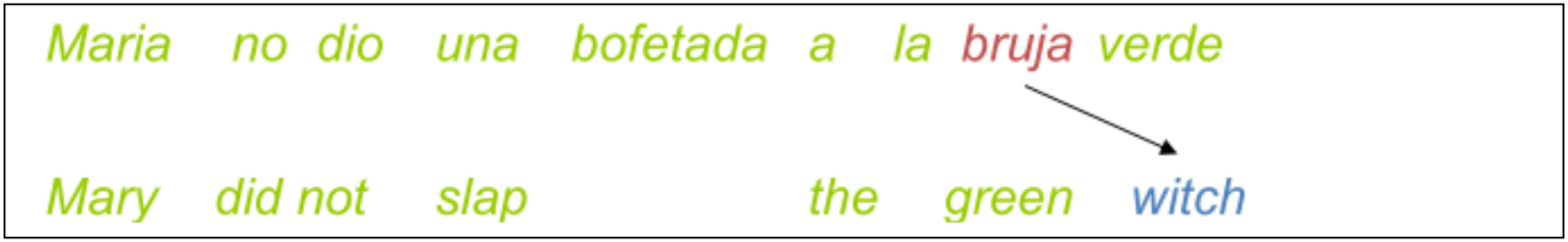
One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Decoding processing



Build the translation from left to right:

1. Match the untranslated source words against the phrase table, and locate a matched source phrase
2. Append the target phrase to the end of the partial translation
3. Mark the source words translated
4. Iterate (1) to (3) until all source words are marked translated.

One to many translation

Many to one translation

Reordering: skip forward

Reordering: skip backward

Scoring hypotheses

- According to the noisy channel model, the score of a hypothesis includes two components:
  - Language model score
  - Phrase-based translation model score
- Shortcomings of the above scoring mechanism:
  - The importance of language model and translation model may be different
  - More factors should be considered in the decisions, for example, word reordering (to select a next source phrase to translate, or skip forwards or backwards), etc.

Log-linear Framework

- A log-linear framework was proposed to replace the noisy channel framework to overcome the above shortcomings:

$$p(e|f) = \frac{\exp(\sum_{i=1}^{n} \lambda_i h_i(e, f))}{\sum_{e'} \exp(\sum_{i=1}^{n} \lambda_i h_i(e', f))}$$

$$\hat{e} = \underset{e}{\operatorname{argmax}} \sum_{i=1}^{n} \lambda_i h_i(e, f)$$

Log-linear Framework

- Advantages of the log-linear framework:
  - Abitrary number of user-defined features (h_i) can be added in the model.
  - Weights (λ_i) can be tuned to balance the importance among the features.

Fine-tuning the log-linear model

- A held-out data set (usually called development set) is used to train the parameters (λ_i) for the log-linear model;
- The training process of the log-linear model is called fune-tuning in SMT;
- Unlike in neural network training, the gradient descend algorithm is not applicable here, because the input data is discrete symbols and the function is not differetiable.
- Fune-tuning algorithms, e.g. MERT, or MIRA, were developed to fine-tune the log-linear model for SMT.

Features for log-linear model in SMT

- Under the noisy channel framework, only two features are considered:
  - Target language model.
  - Backward translation model.
- Under the log-linear framework, more features can be considered, typically including:
  - Forward translation model.
  - Forward and backward lexicalized translation models.
  - Additional language models. (e.g. with different ngram orders)
  - Word reordering models.
  - Length of the target sentence.
  - User-defined dictionaries.

Content

1 Machine Translation (MT)

2 Machine translation evaluation

3 Statistical machine translation (SMT)

4 Neural machine translation (NMT) based on RNNs

5 Neural machine translation (NMT) with attentions

What is Neural Machine Translation?

- Neural Machine Translation (NMT) is a way to do Machine Translation with a *single neural network*

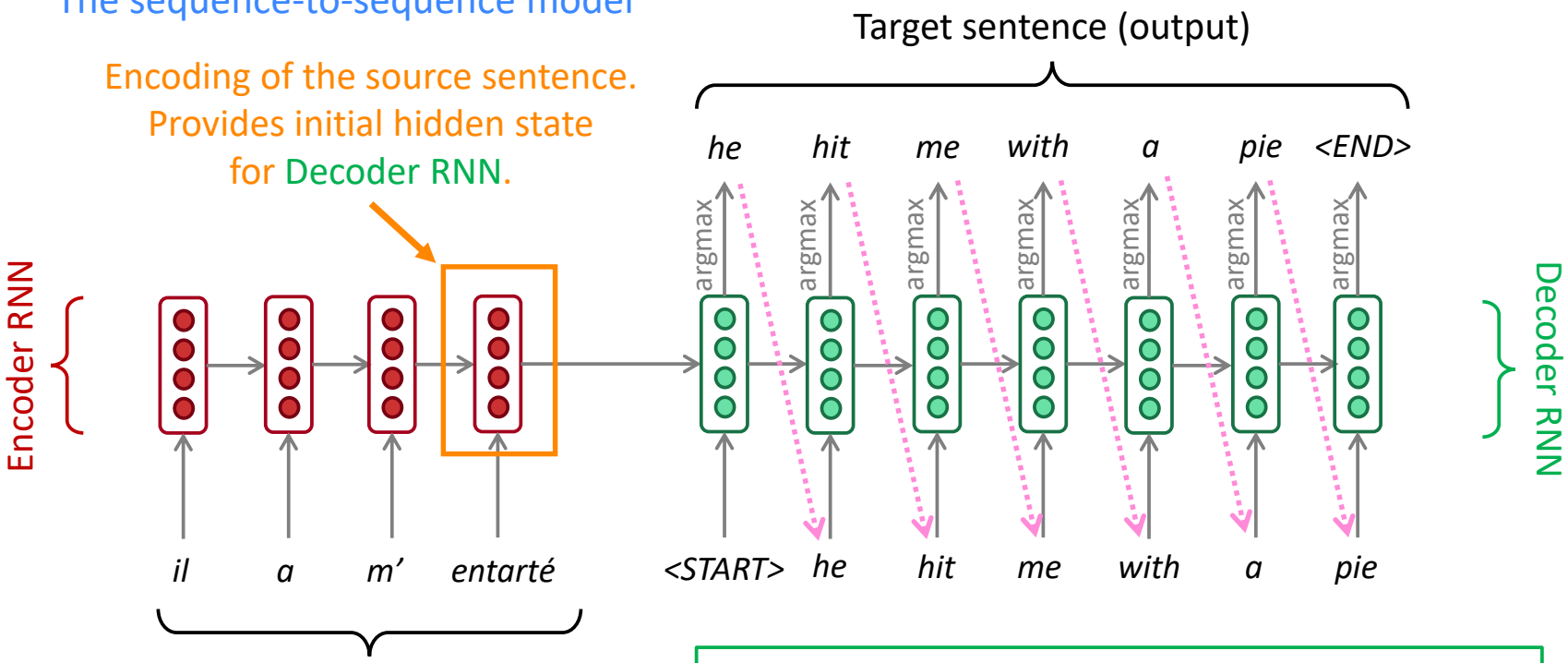
- The neural network architecture is called sequence-to-sequence (aka seq2seq) and it involves *two* RNNs.

23

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Neural Machine Translation (NMT)

The sequence-to-sequence model



Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

Encoder RNN produces an encoding of the source sentence.

Note: This diagram shows **test time** behavior: decoder output is fed in ⋯⋯→ as next step's input

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Sequence-to-sequence is versatile!

- Sequence-to-sequence is useful for *more than just MT*

- Many NLP tasks can be phrased as sequence-to-sequence:
  - Summarization (long text → short text)
  - Dialogue (previous utterances → next utterance)
  - Parsing (input text → output parse as sequence)
  - Code generation (natural language → Python code)

25

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Neural Machine Translation (NMT)

- The sequence-to-sequence model is an example of a **Conditional Language Model**.
  - **Language Model** because the decoder is predicting the next word of the target sentence *y*
  - **Conditional** because its predictions are *also* conditioned on the source sentence *x*

- NMT directly calculates $P(y|x)$:

$$P(y|x) = P(y_1|x)\,P(y_2|y_1,x)\,P(y_3|y_1,y_2,x)\ldots\underbrace{P(y_T|y_1,\ldots,y_{T-1},x)}$$

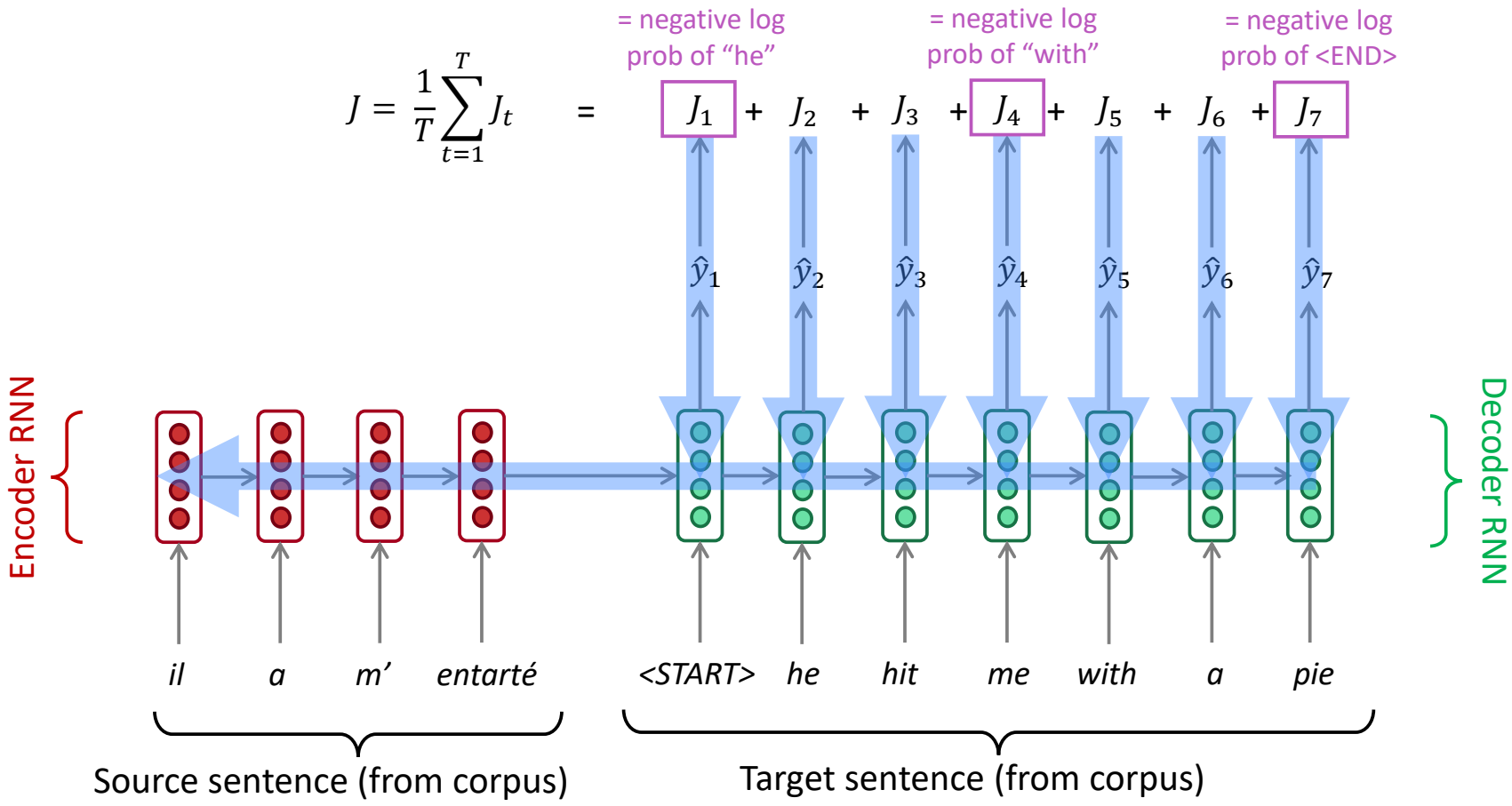
Probability of next target word, given target words so far and source sentence *x*

- **Question**: How to train a NMT system?
- **Answer**: Get a big parallel corpus…

26

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Training a Neural Machine Translation system



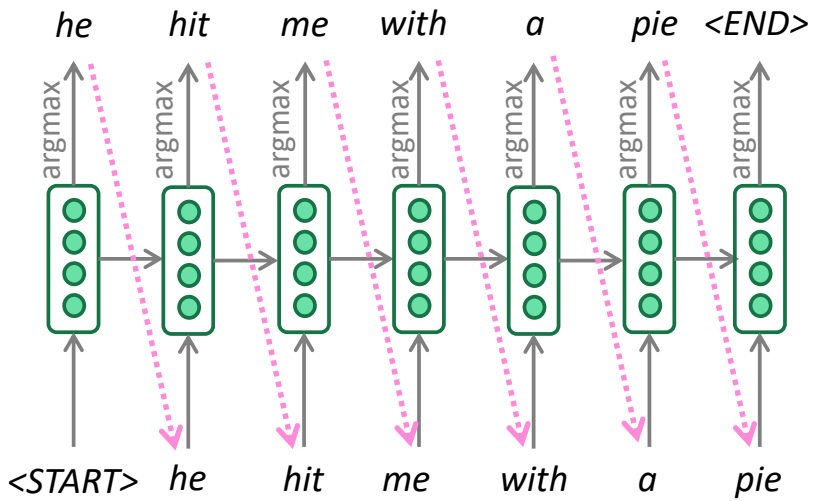
Seq2seq is optimized as a **single system.**
Backpropagation operates "*end-to-end*".

27

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Greedy decoding

- We saw how to generate (or “decode”) the target sentence by taking argmax on each step of the decoder



- This is greedy decoding (take most probable word on each step)
- **Problems with this method?**

28

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Problems with greedy decoding

- Greedy decoding has no way to undo decisions!
  - Input: *il a m'entarté* *(he hit me with a pie)*
  - → *he* ____
  - → *he hit* ____
  - → *he hit a* ____ (whoops! no going back now…)
- How to fix this?

29

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Exhaustive search decoding

- Ideally we want to find a (length *T*) translation *y* that maximizes

$$P(y|x) = P(y_1|x)\,P(y_2|y_1,x)\,P(y_3|y_1,y_2,x)\ldots,P(y_T|y_1,\ldots,y_{T-1},x)$$
$$= \prod_{t=1}^{T} P(y_t|y_1,\ldots,y_{t-1},x)$$

- We could try computing all possible sequences *y*
  - This means that on each step *t* of the decoder, we're tracking V^t possible partial translations, where *V* is vocab size
  - This $O(V^T)$ complexity is far too expensive!

30

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding

- Core idea: On each step of decoder, keep track of the *k* most probable partial translations (which we call *hypotheses*)
  - *k* is the beam size (in practice around 5 to 10)

- A hypothesis $y_1, \ldots, y_t$ has a score which is its log probability:

$$\text{score}(y_1, \ldots, y_t) = \log P_{\text{LM}}(y_1, \ldots, y_t | x) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

  - Scores are all negative, and higher score is better
  - We search for high-scoring hypotheses, tracking top *k* on each step

- Beam search is not guaranteed to find optimal solution
- But much more efficient than exhaustive search!

31

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$

<START>

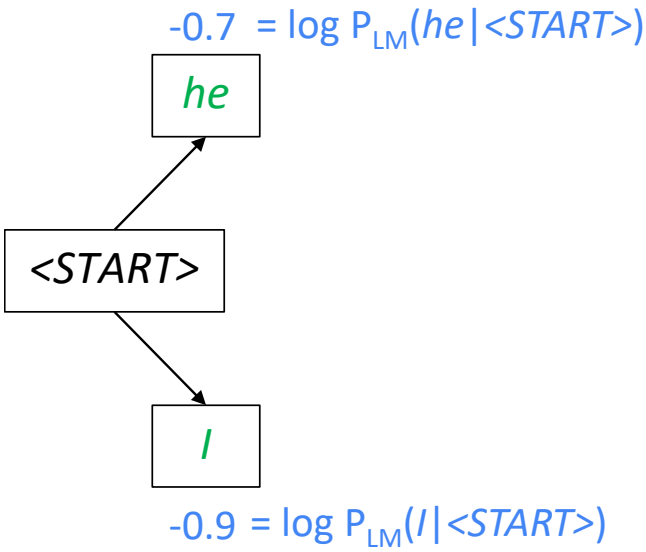
32

Calculate prob dist of next word

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$

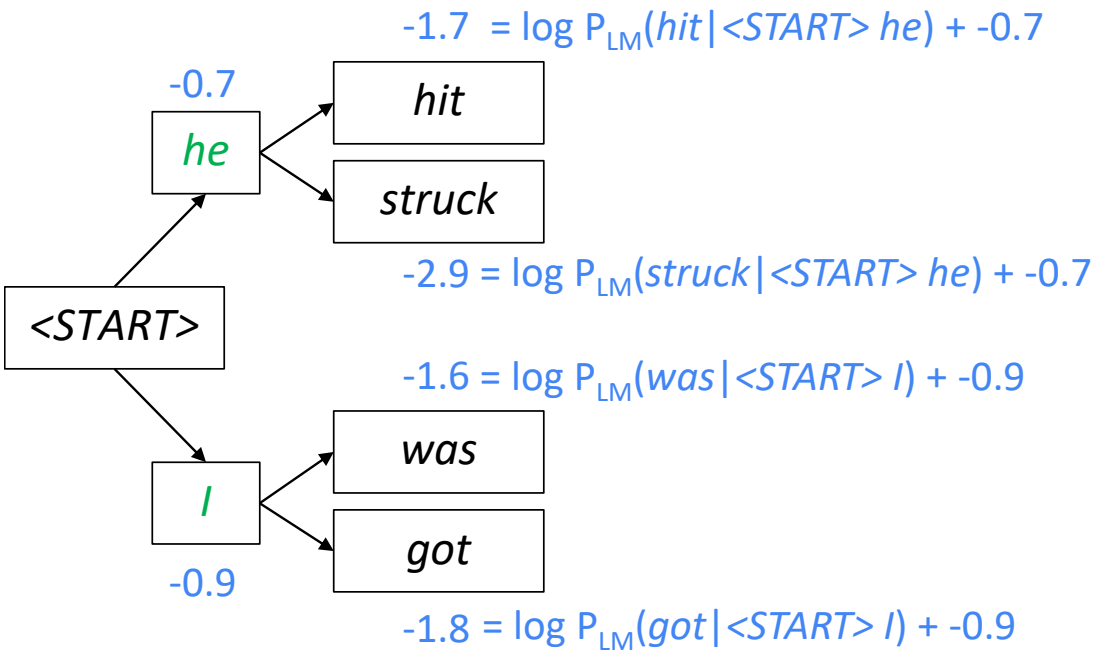


33

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



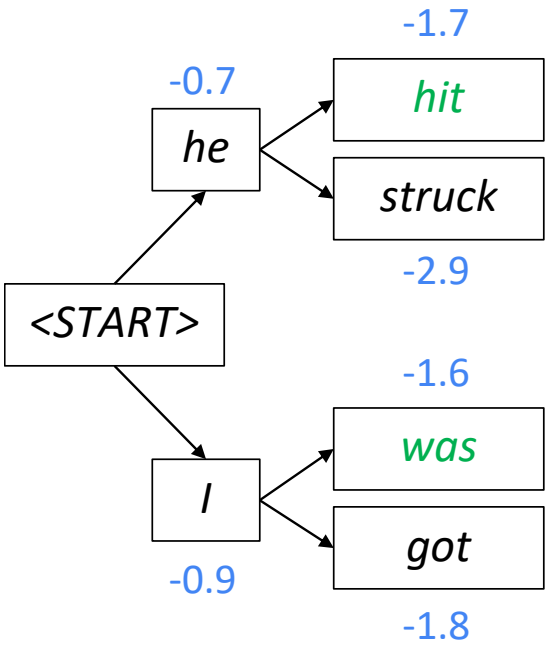
For each of the *k* hypotheses, find top *k* next words and calculate scores

34

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



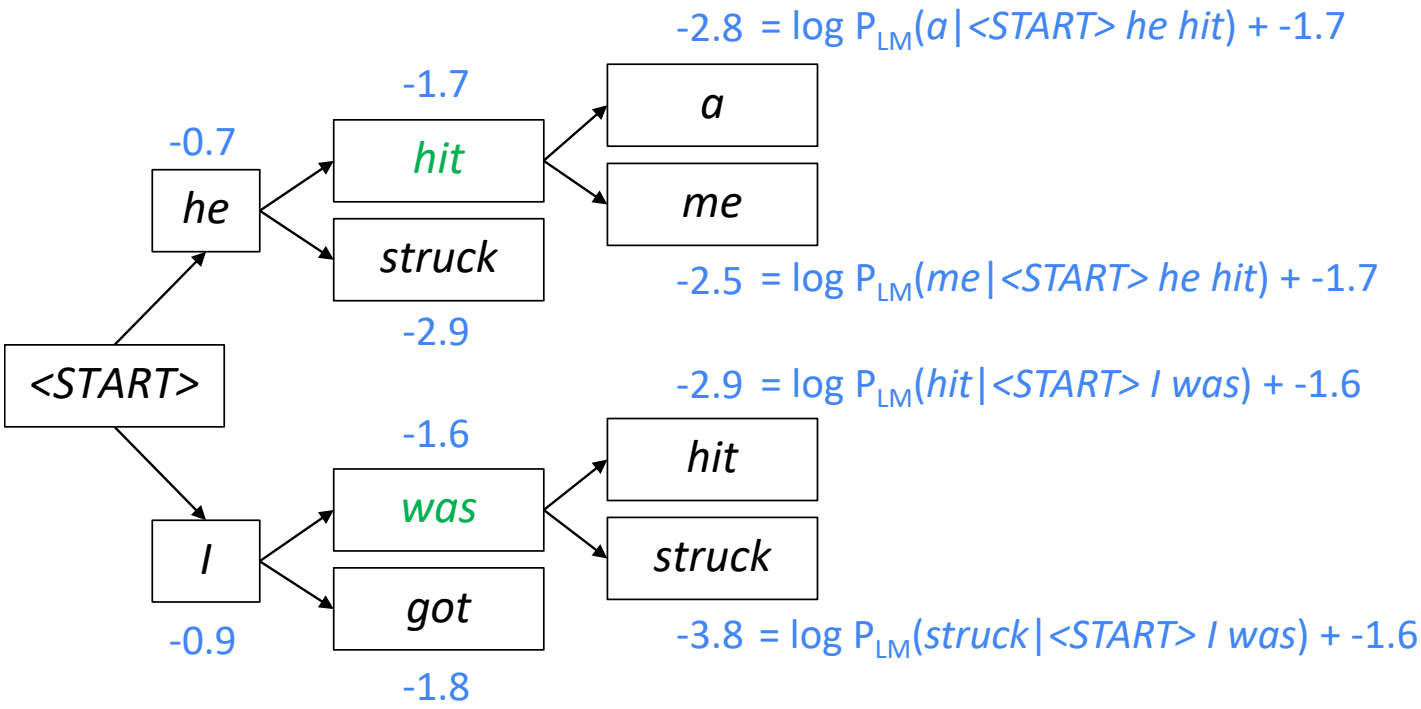
Of these k^2 hypotheses, just keep k with highest scores

35

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



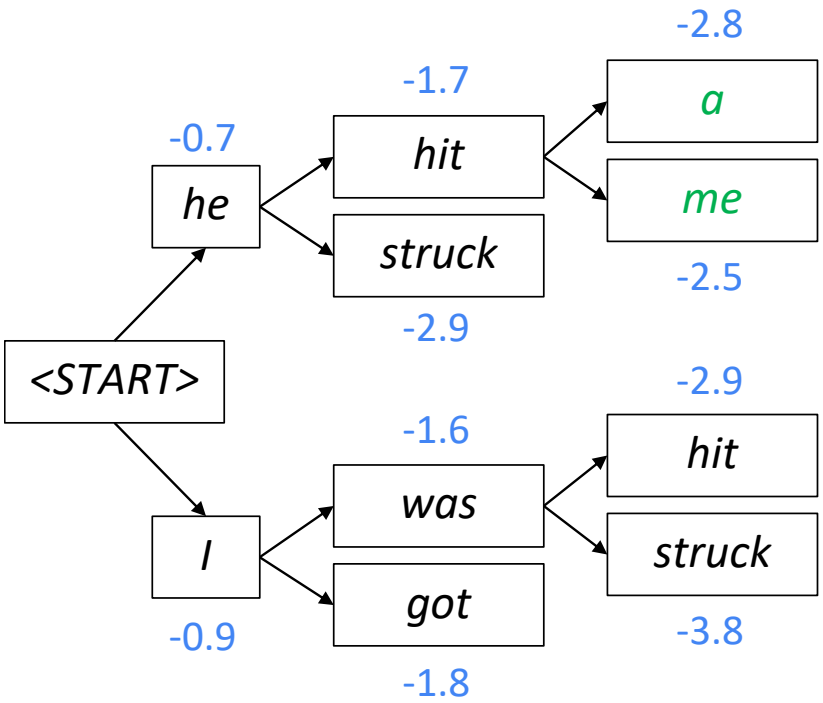
For each of the *k* hypotheses, find top *k* next words and calculate scores

36

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



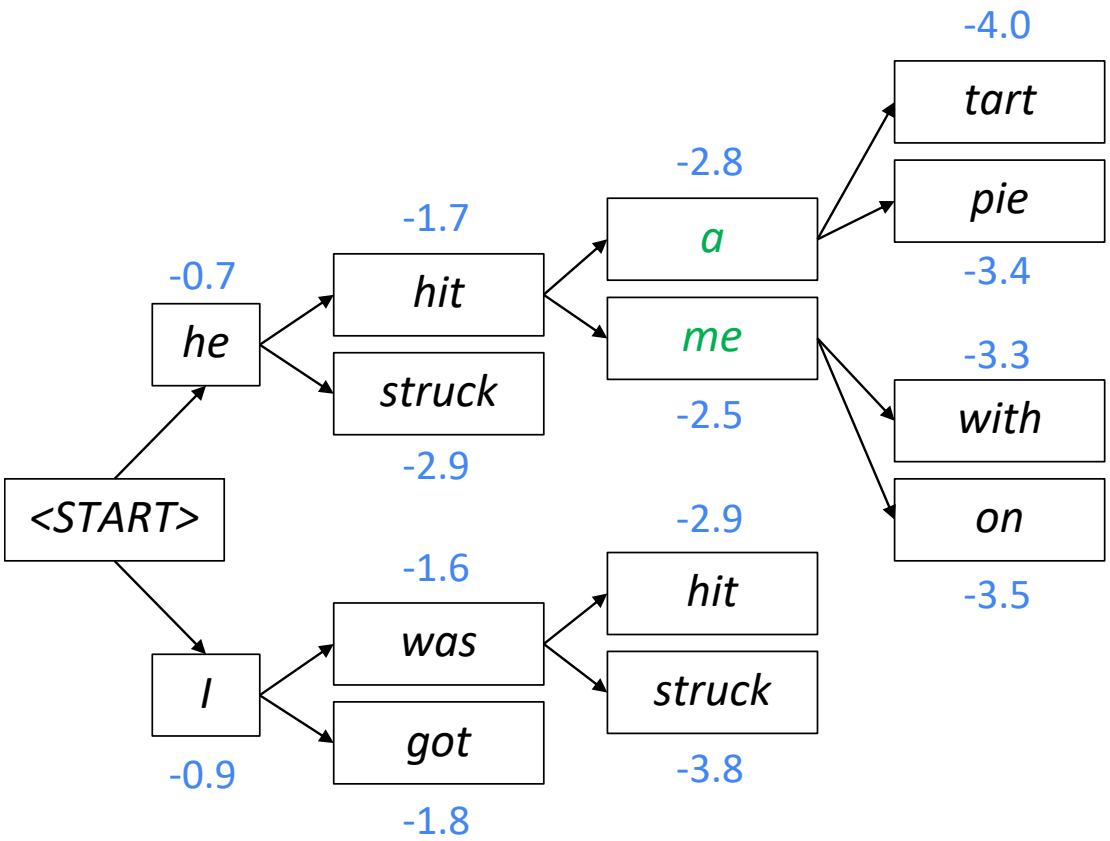
Of these k^2 hypotheses, just keep k with highest scores

37

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



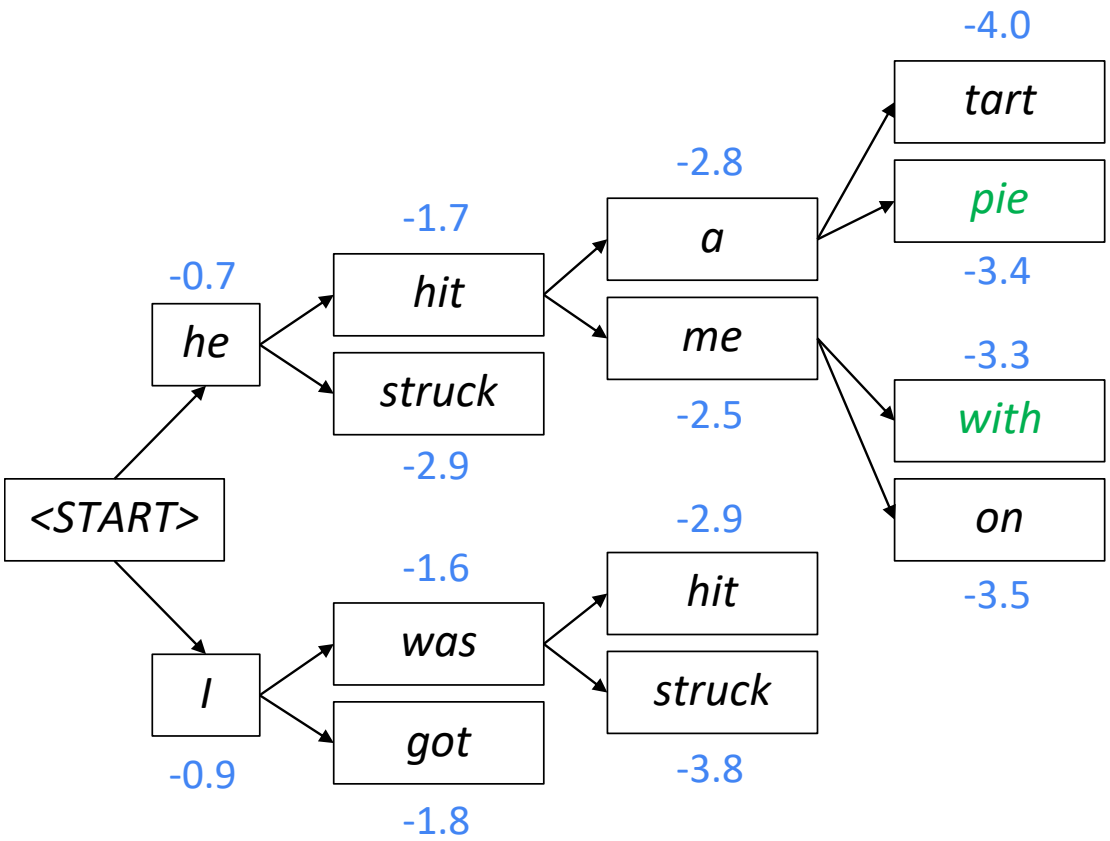
For each of the *k* hypotheses, find top *k* next words and calculate scores

38

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



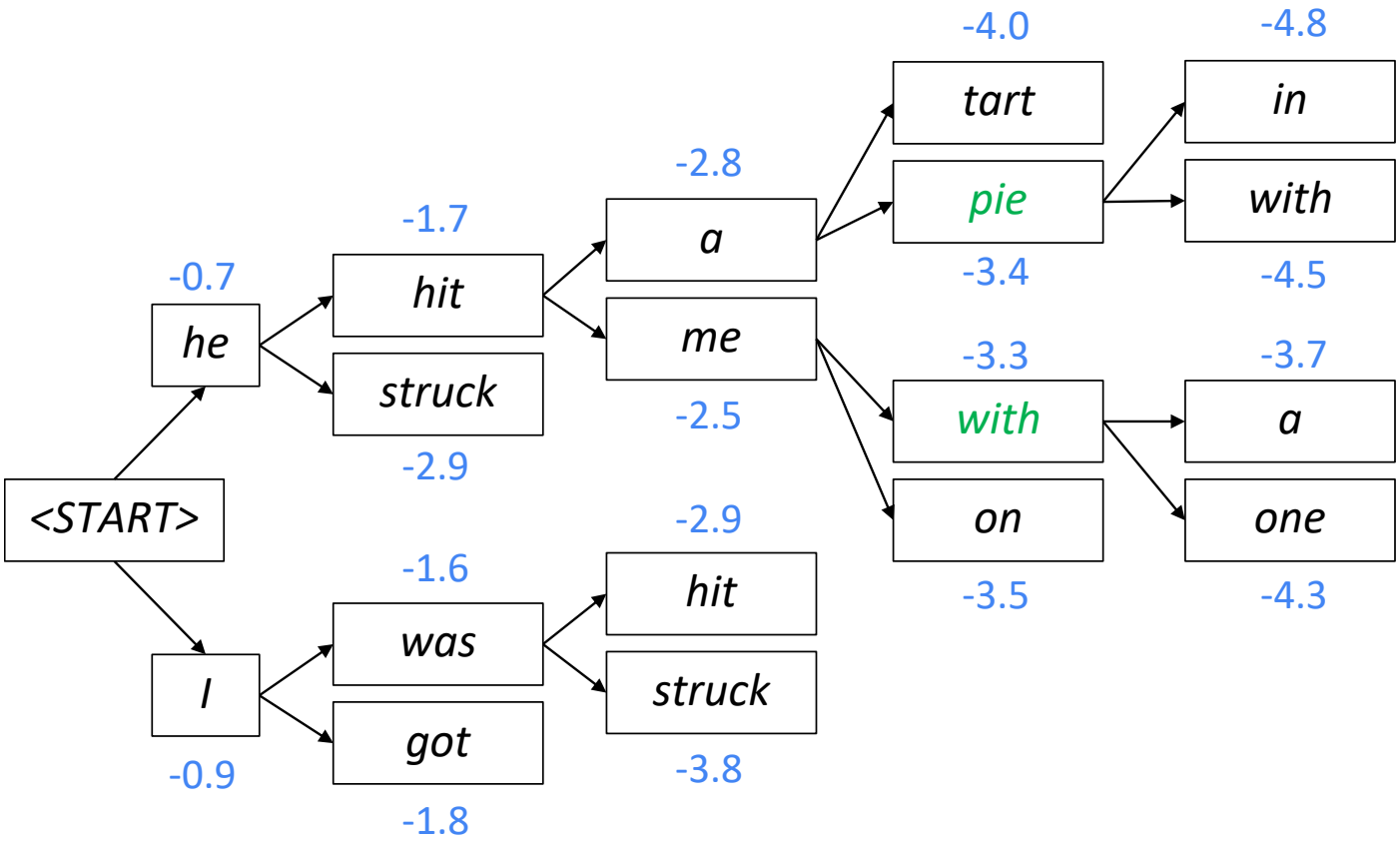
Of these k^2 hypotheses, just keep k with highest scores

39

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



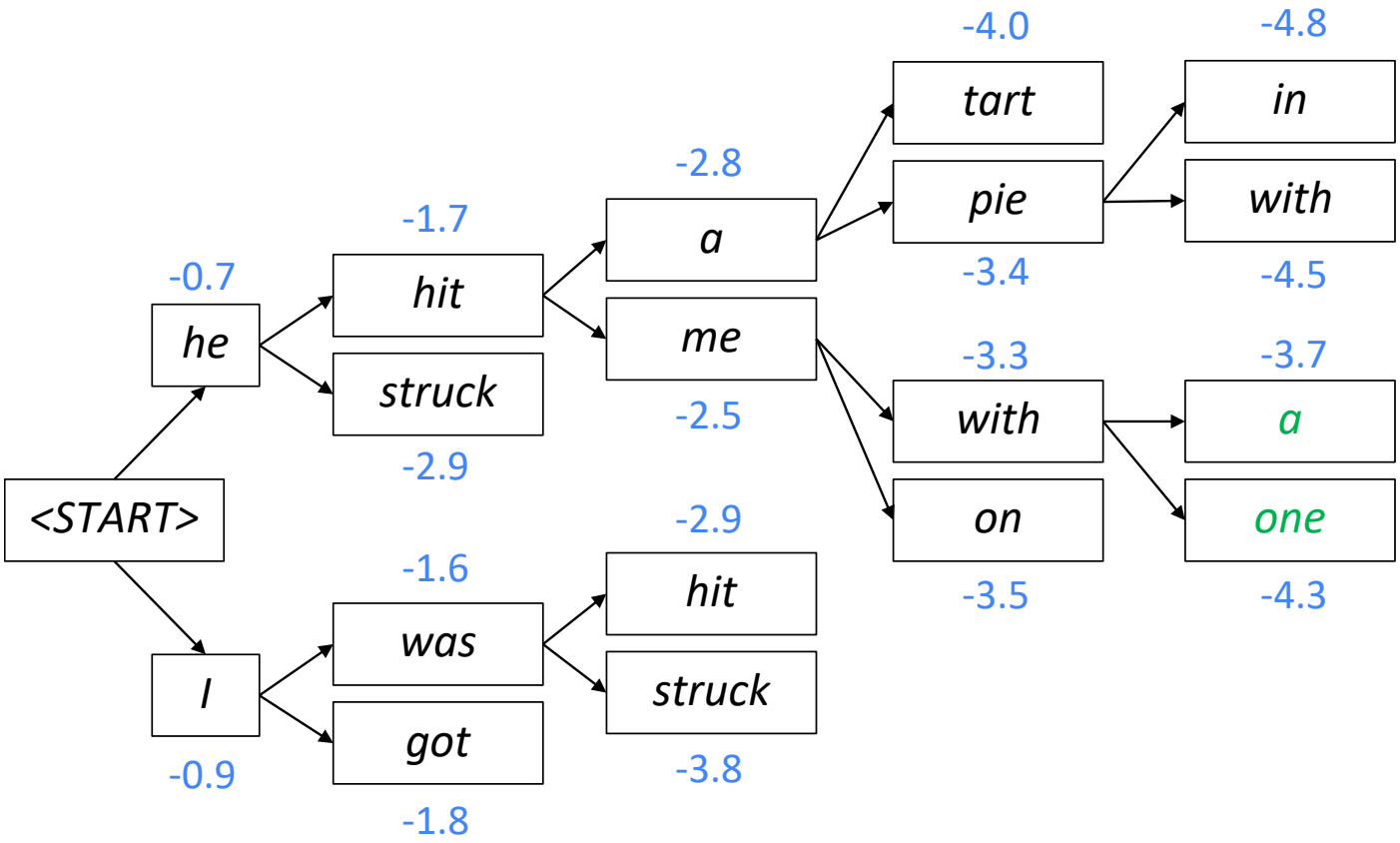
For each of the *k* hypotheses, find top *k* next words and calculate scores

40

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



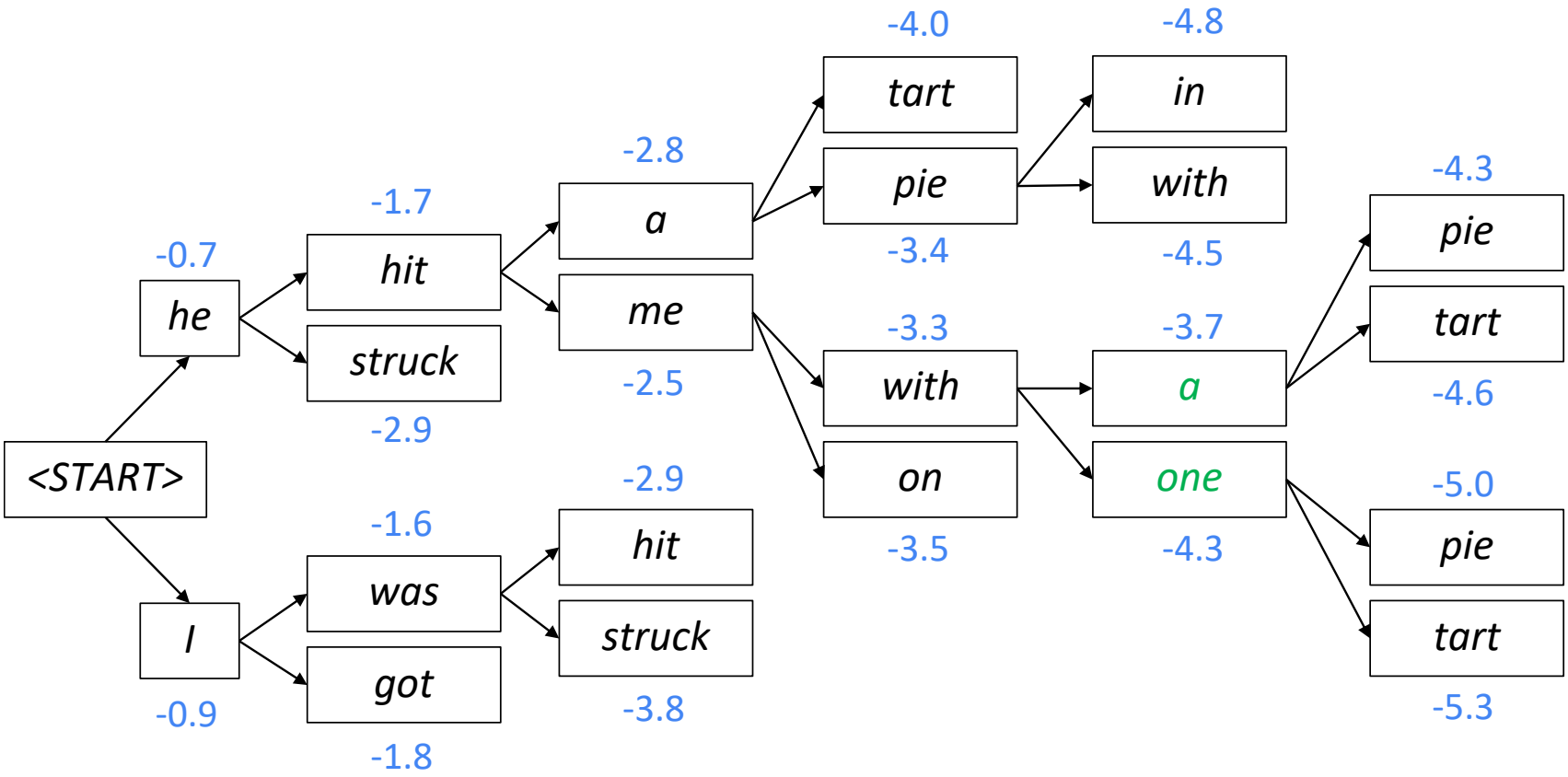
Of these k^2 hypotheses, just keep k with highest scores

41

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



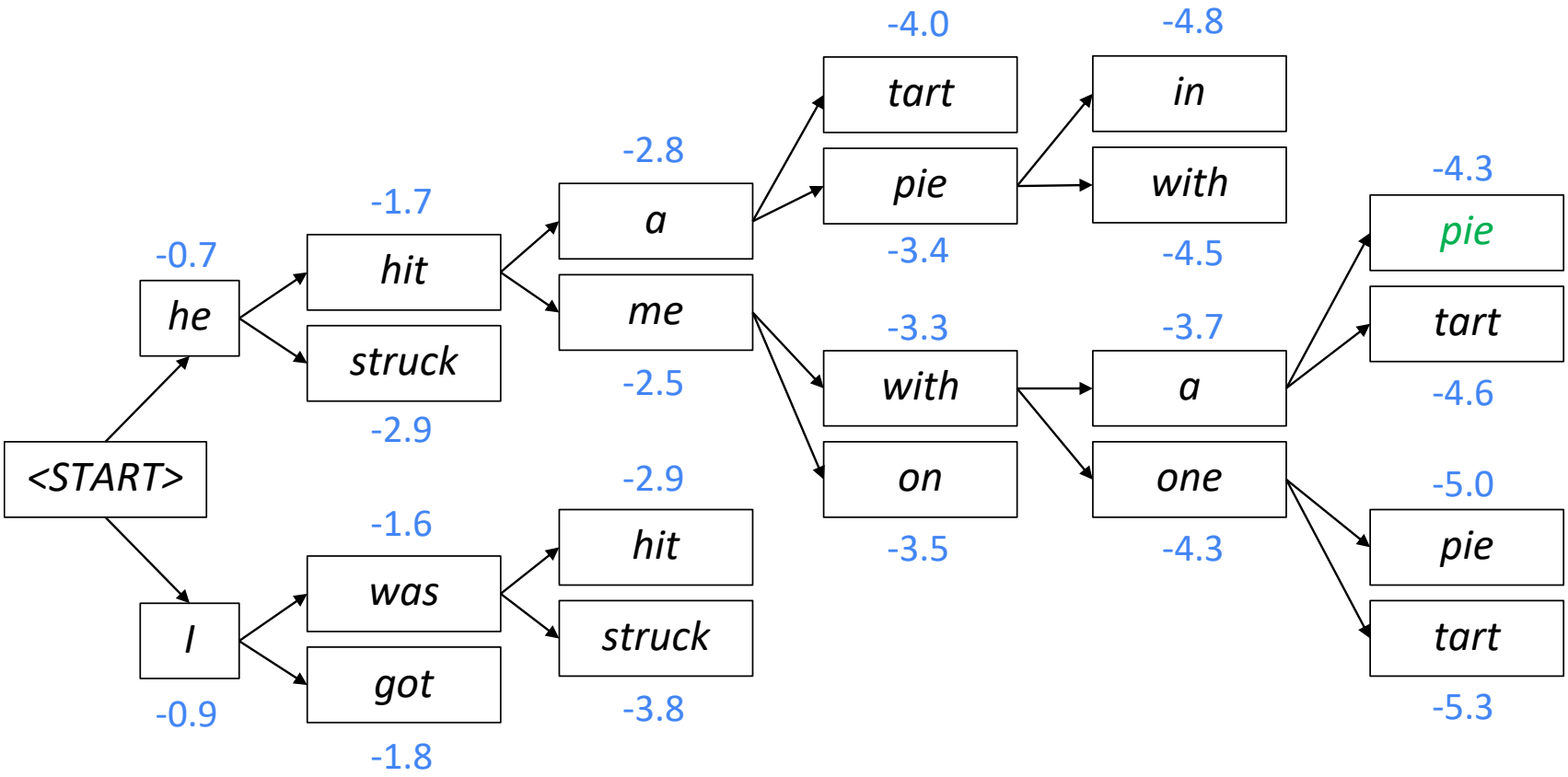
For each of the *k* hypotheses, find top *k* next words and calculate scores

42

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



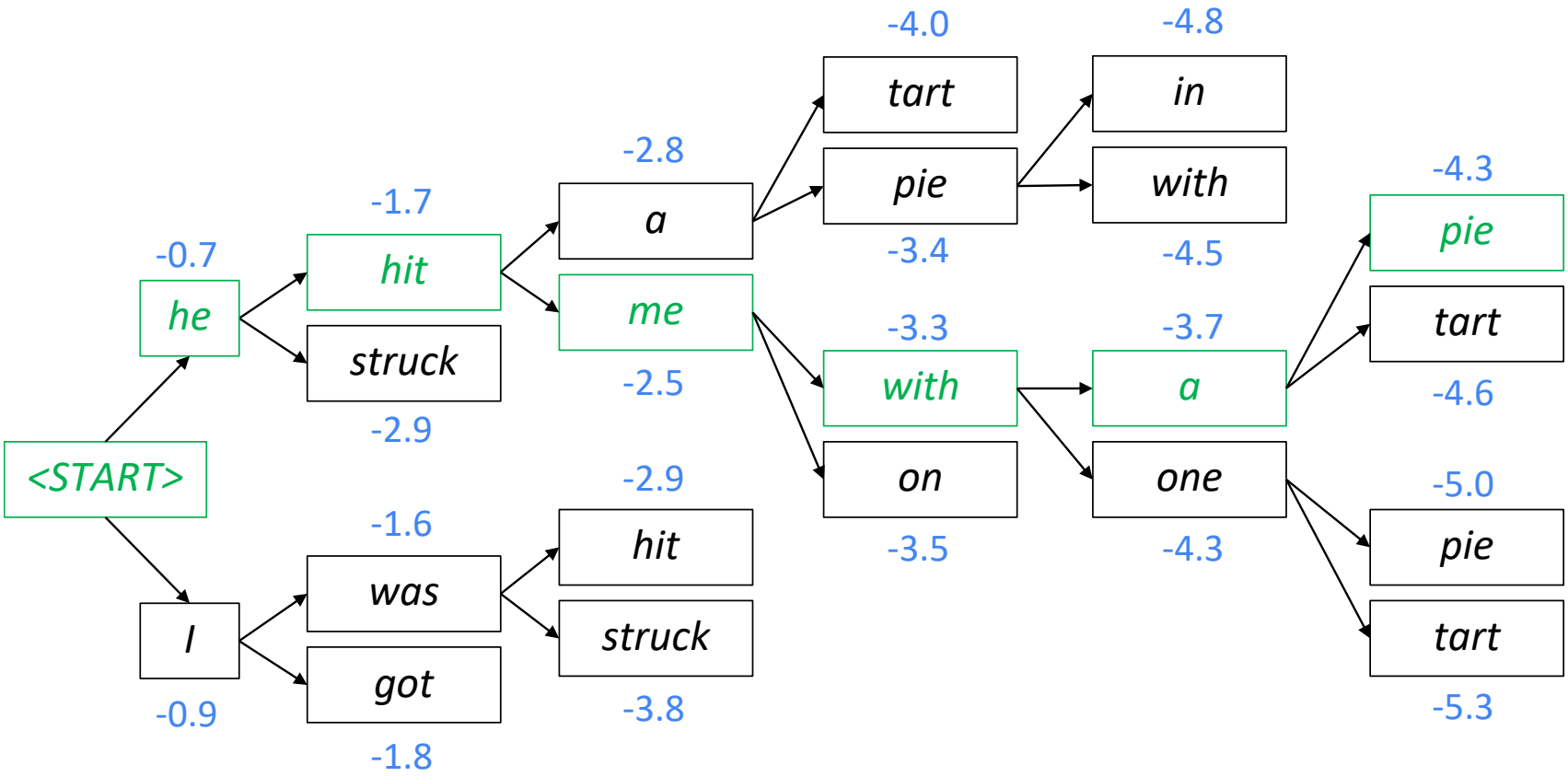
This is the top-scoring hypothesis!

43

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Backtrack to obtain the full hypothesis

44

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: stopping criterion

- In greedy decoding, usually we decode until the model produces a <END> token
  - For example: *<START> he hit me with a pie <END>*
- In beam search decoding, different hypotheses may produce <END> tokens on different timesteps
  - When a hypothesis produces <END>, that hypothesis is complete.
  - Place it aside and continue exploring other hypotheses via beam search.
- Usually we continue beam search until:
  - We reach timestep T (where T is some pre-defined cutoff), or
  - We have at least n completed hypotheses (where n is pre-defined cutoff)

45

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Beam search decoding: finishing up

- We have our list of completed hypotheses.
- How to select top one with highest score?

- Each hypothesis $y_1, \ldots, y_t$ on our list has a score

$$\text{score}(y_1, \ldots, y_t) = \log P_{\text{LM}}(y_1, \ldots, y_t | x) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

- Problem with this: longer hypotheses have lower scores

- Fix: Normalize by length. Use this to select top one instead:

$$\frac{1}{t} \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

46

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Advantages of NMT

Compared to SMT, NMT has many advantages:

- Better performance
  - More fluent
  - Better use of context
  - Better use of phrase similarities
- A single neural network to be optimized end-to-end
  - No subcomponents to be individually optimized
- Requires much less human engineering effort
  - No feature engineering
  - Same method for all language pairs

47

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Disadvantages of NMT?

Compared to SMT:

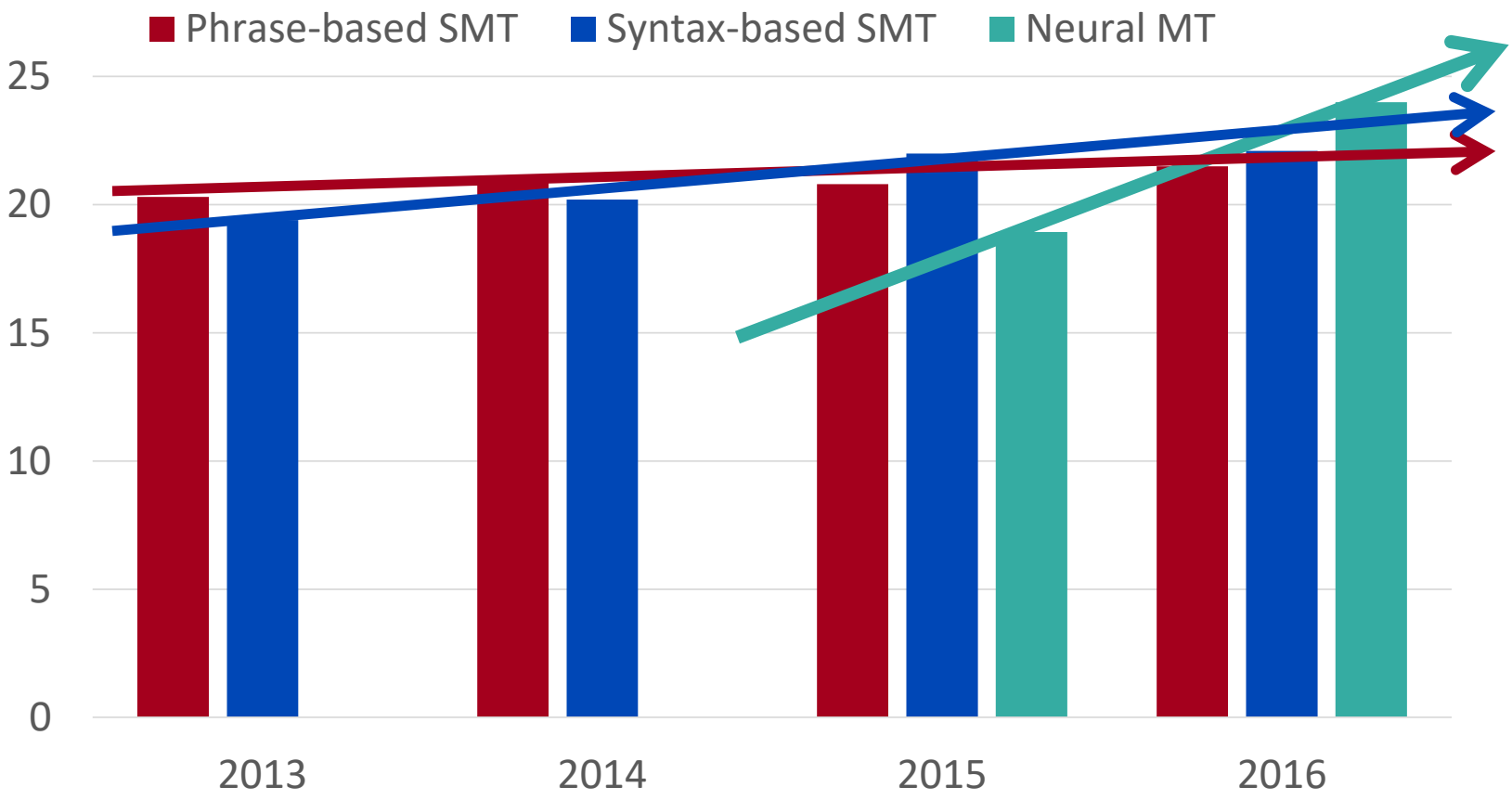
- NMT is less interpretable
  - Hard to debug
- NMT is difficult to control
  - For example, can't easily specify rules or guidelines for translation
  - Safety concerns!

48

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

MT progress over time

[Edinburgh En-De WMT newstest2013 Cased BLEU; NMT 2015 from U. Montréal]



50

Source: http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

NMT: the biggest success story of NLP Deep Learning

Neural Machine Translation went from a fringe research activity in **2014** to the leading standard method in **2016**

- **2014**: First seq2seq paper published
- **2016**: Google Translate switches from SMT to NMT
- This is amazing!
  - **SMT** systems, built by hundreds of engineers over many years, outperformed by NMT systems trained by a handful of engineers in a few months

51

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

So is Machine Translation solved?

- **Nope!**
- Many difficulties remain:
  - Out-of-vocabulary words
  - Domain mismatch between train and test data
  - Maintaining context over longer text
  - Low-resource language pairs

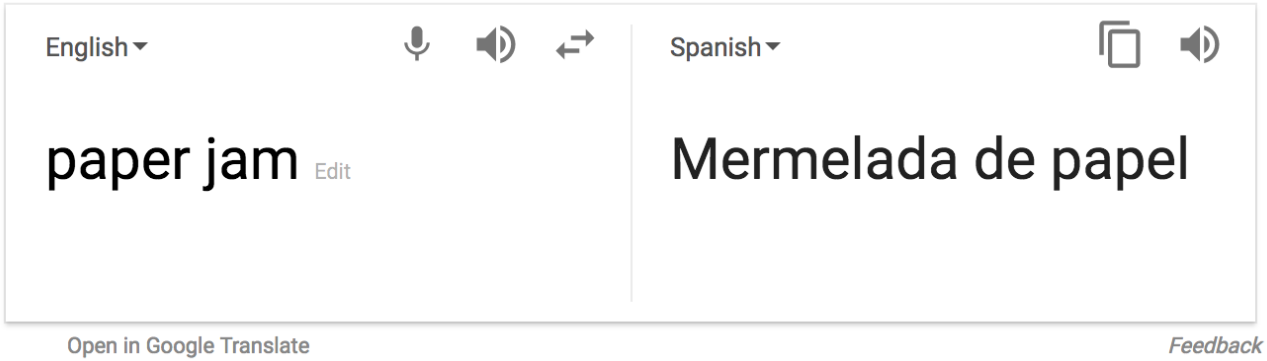
Further reading: *"Has AI surpassed humans at translation? Not even close!"*
https://www.skynettoday.com/editorials/state_of_nmt

52

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

So is Machine Translation solved?

- **Nope!**
- Using common sense is still hard



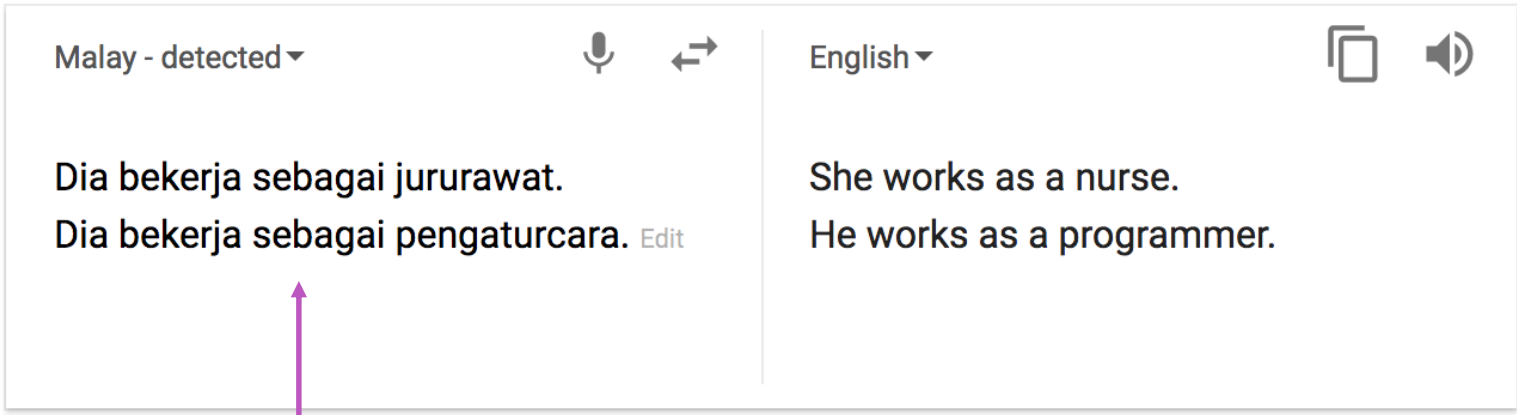
?

53

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

So is Machine Translation solved?

- **Nope!**
- NMT picks up biases in training data



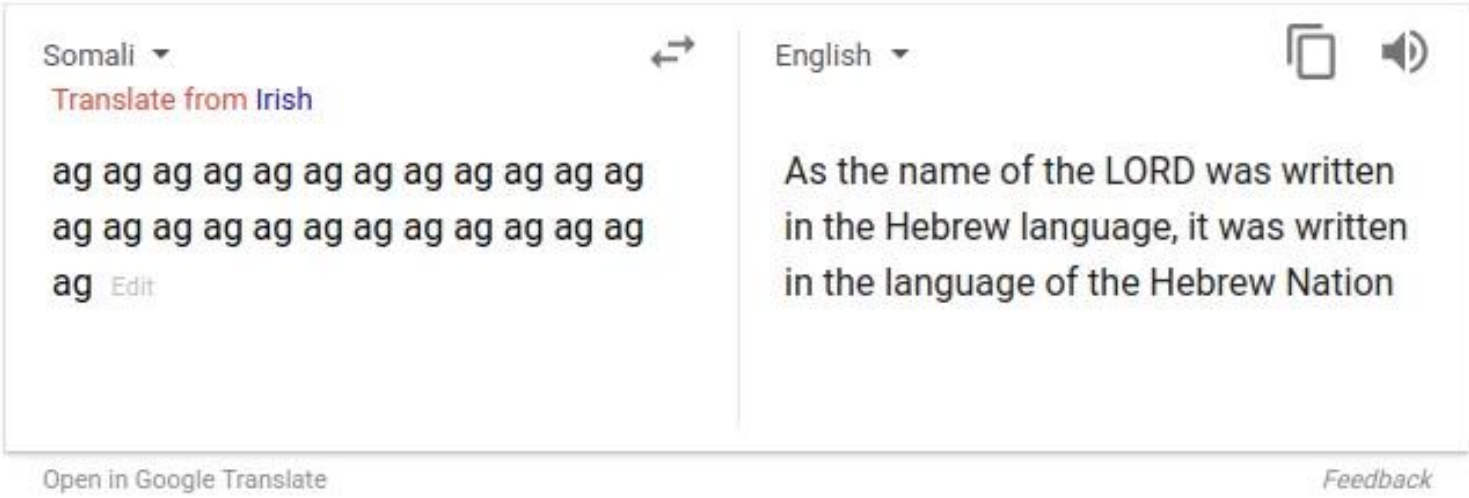
Didn't specify gender

Source: https://hackernoon.com/bias-sexist-or-this-is-the-way-it-should-be-ce1f7c8c683c

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

So is Machine Translation solved?

- Nope!
- Uninterpretable systems do strange things



Picture source: https://www.vice.com/en_uk/article/j5npeg/why-is-google-translate-spitting-out-sinister-religious-prophecies
Explanation: https://www.skynettoday.com/briefs/google-nmt-prophecies

55

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

NMT research continues

NMT is the **flagship task** for NLP Deep Learning

- NMT research has pioneered many of the recent innovations of NLP Deep Learning

- In **2019**: NMT research continues to thrive
  - Researchers have found *many, many* improvements to the "vanilla" seq2seq NMT system we've presented today
  - But one improvement is so integral that it is the new vanilla…

ATTENTION

56

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Content

1 Machine Translation (MT)

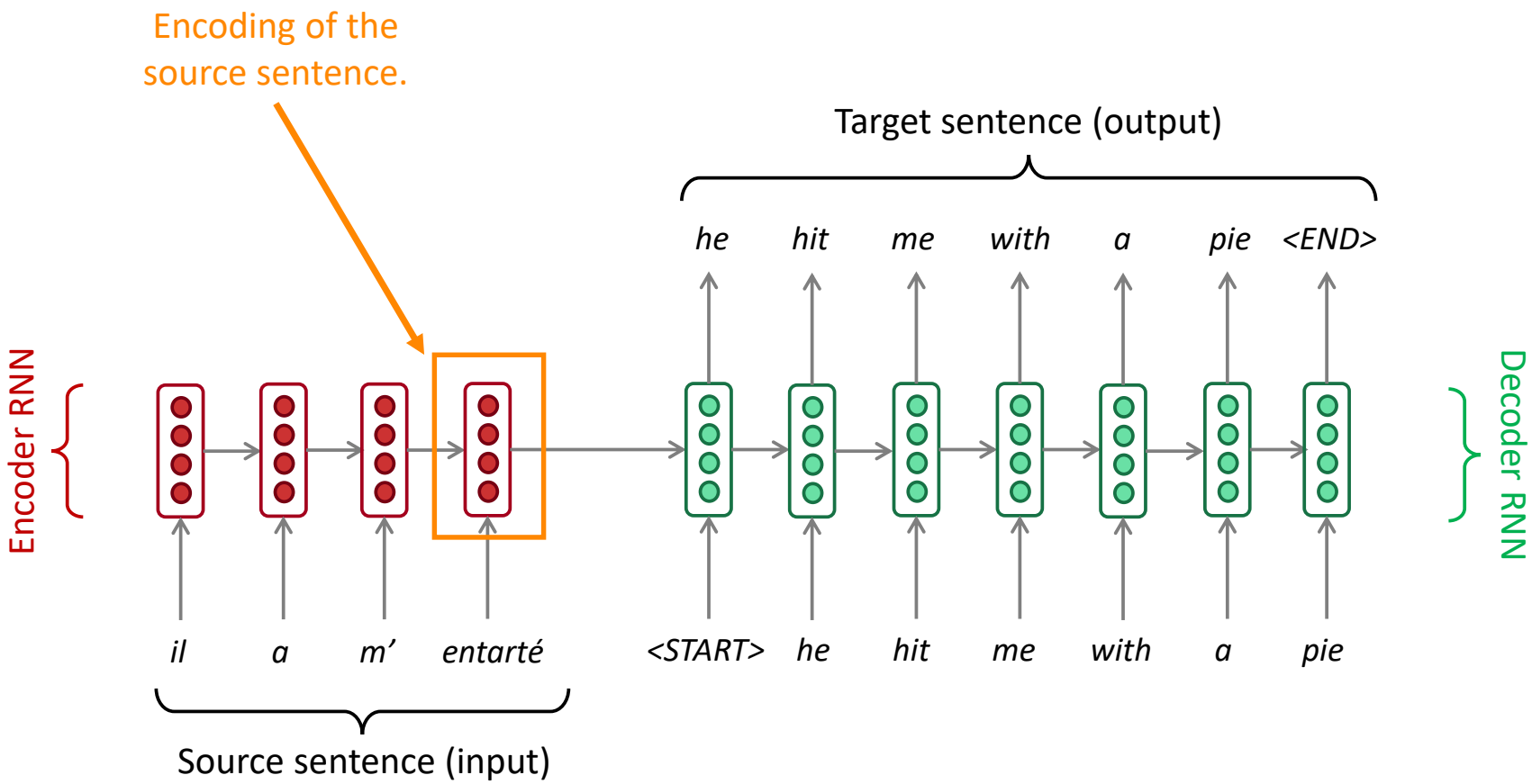
2 Machine translation evaluation

3 Statistical machine translation (SMT)

4 Neural machine translation (NMT) based on RNNs

5 Neural machine translation (NMT) with attentions

Sequence-to-sequence: the bottleneck problem

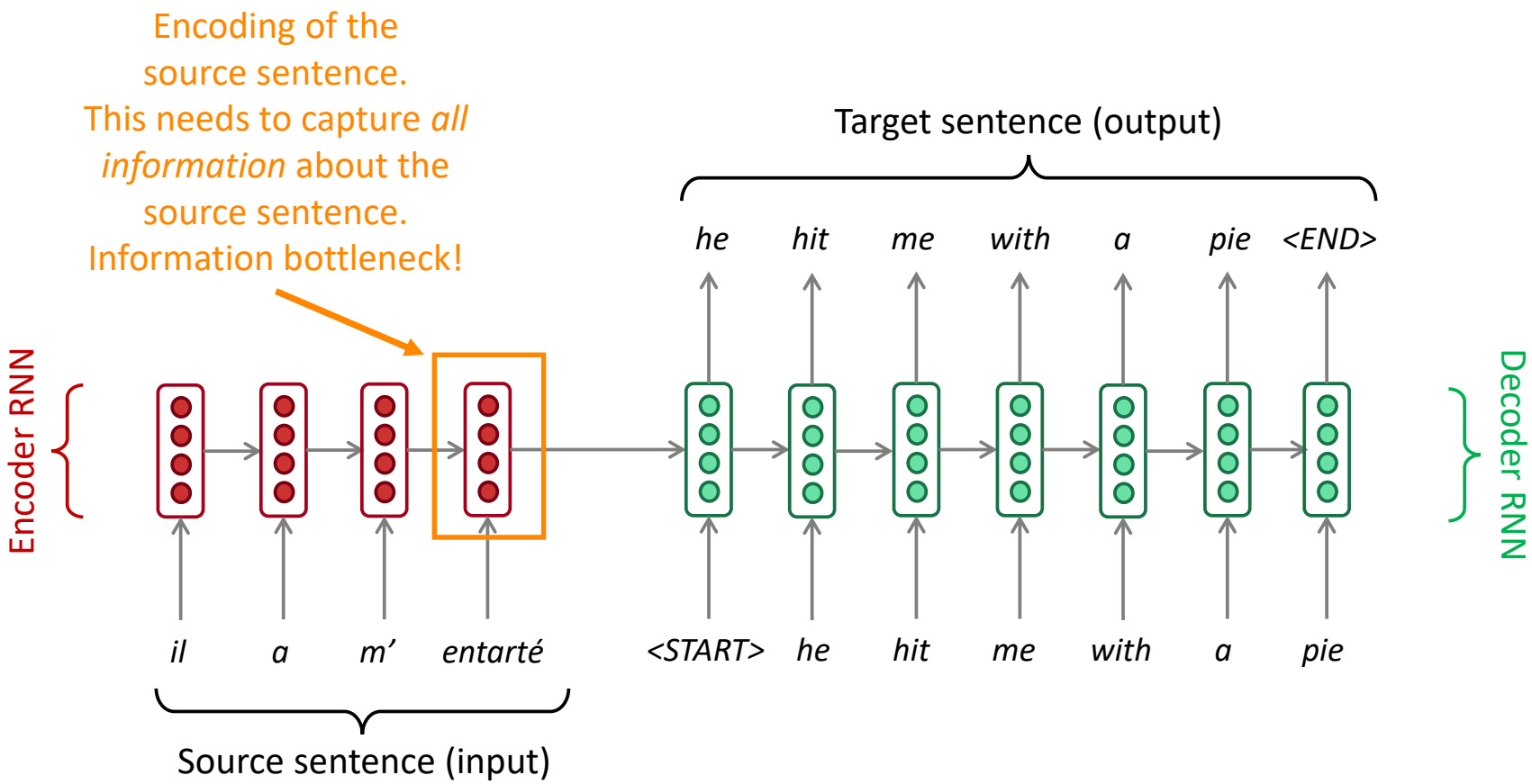


Problems with this architecture?

58

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Sequence-to-sequence: the bottleneck problem



59

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Attention

- **Attention** provides a solution to the bottleneck problem.

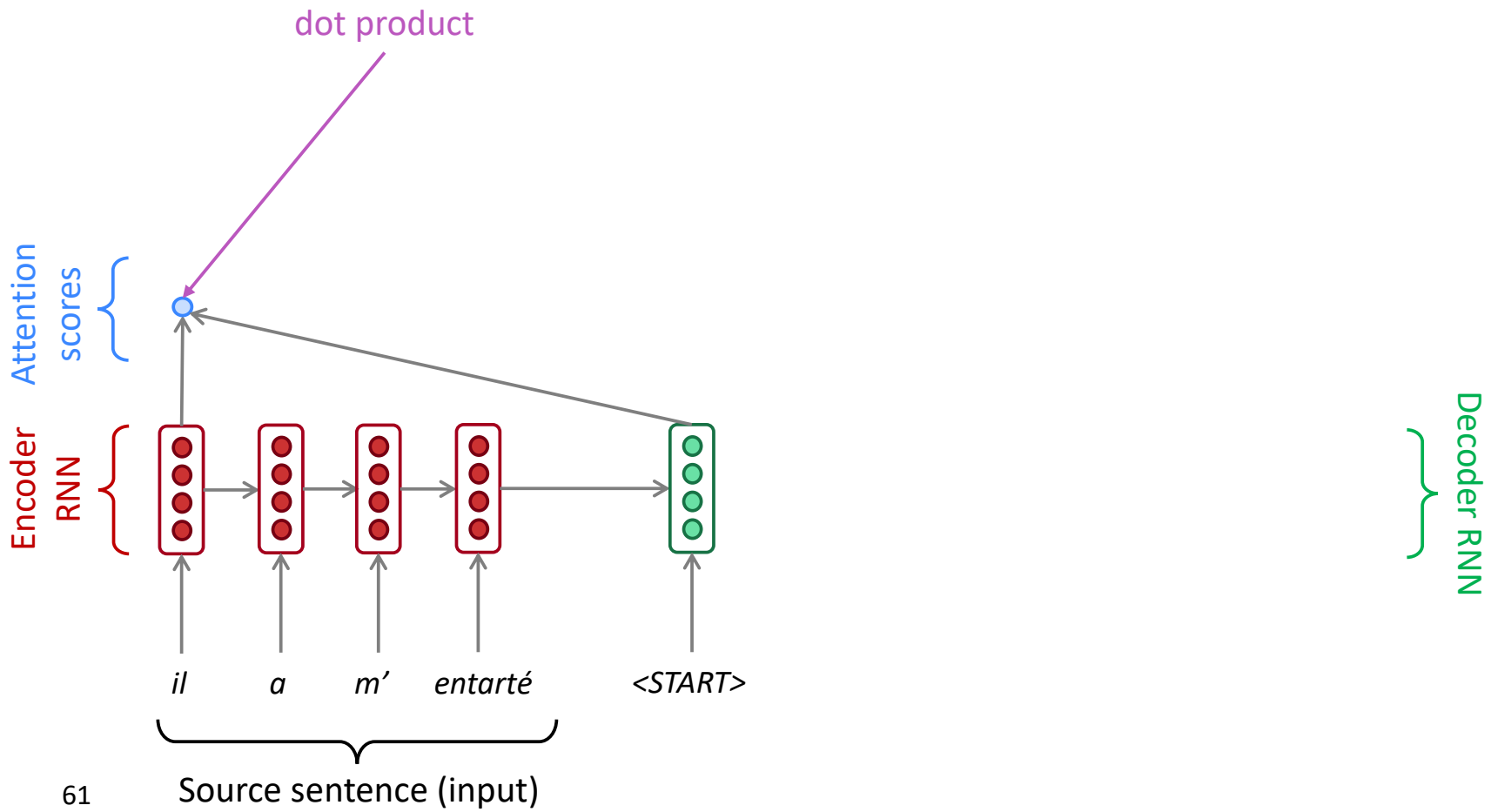
- Core idea: on each step of the decoder, use *direct connection to the encoder* to *focus on a particular part* of the source sequence

- First we will show via diagram (no equations), then we will show with equations

60

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

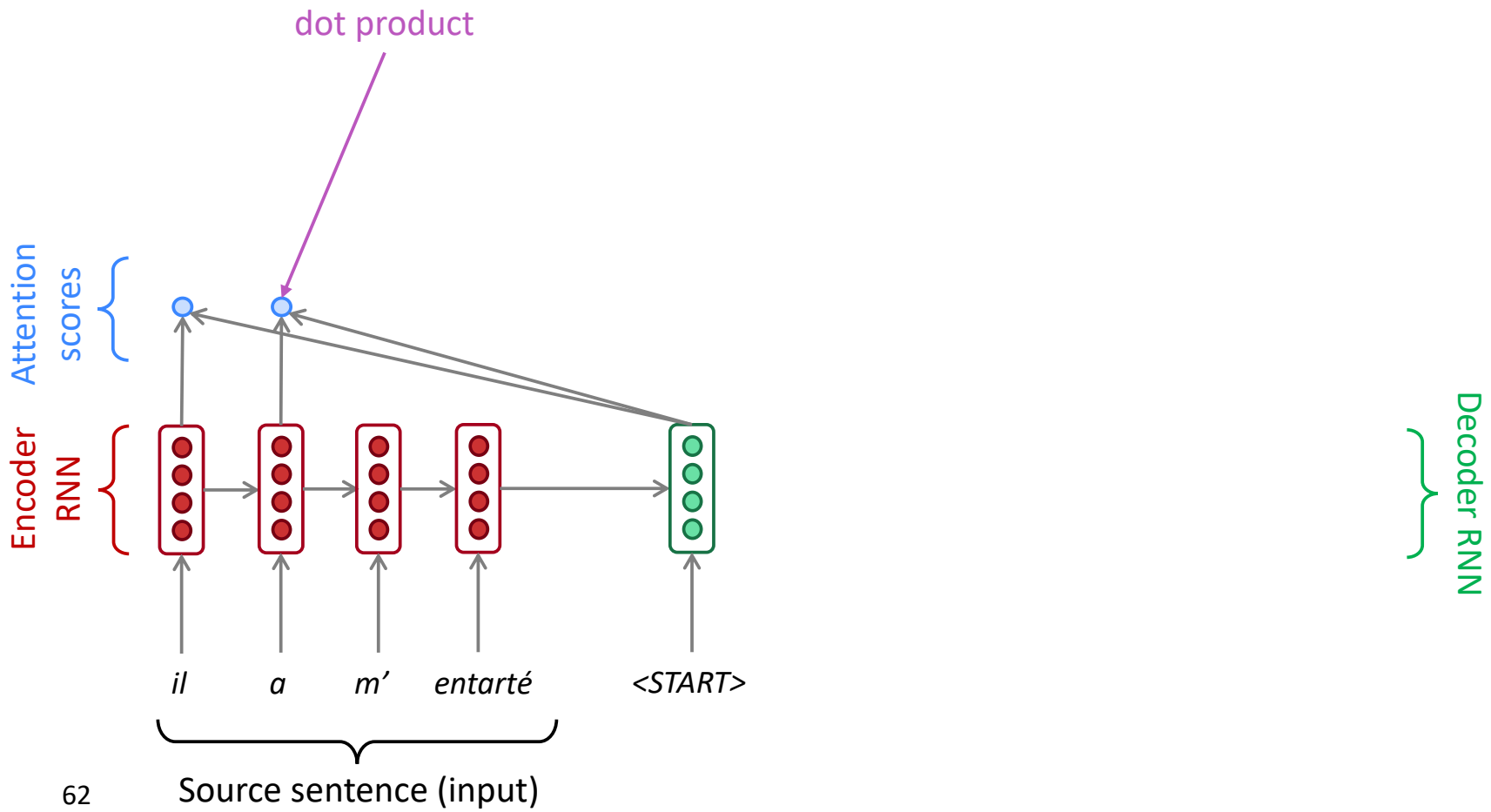
Sequence-to-sequence with attention



61

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

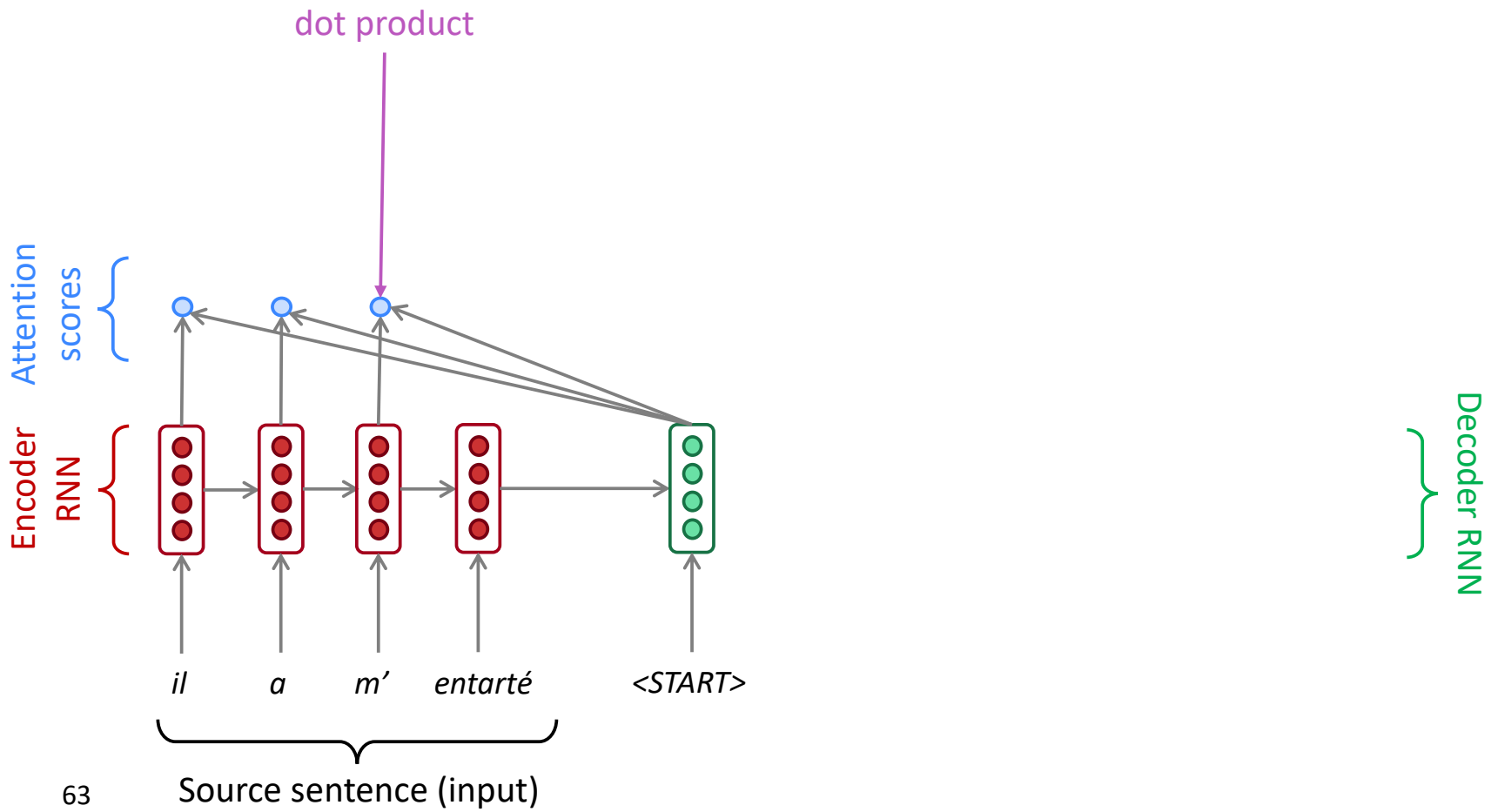
Sequence-to-sequence with attention



62

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

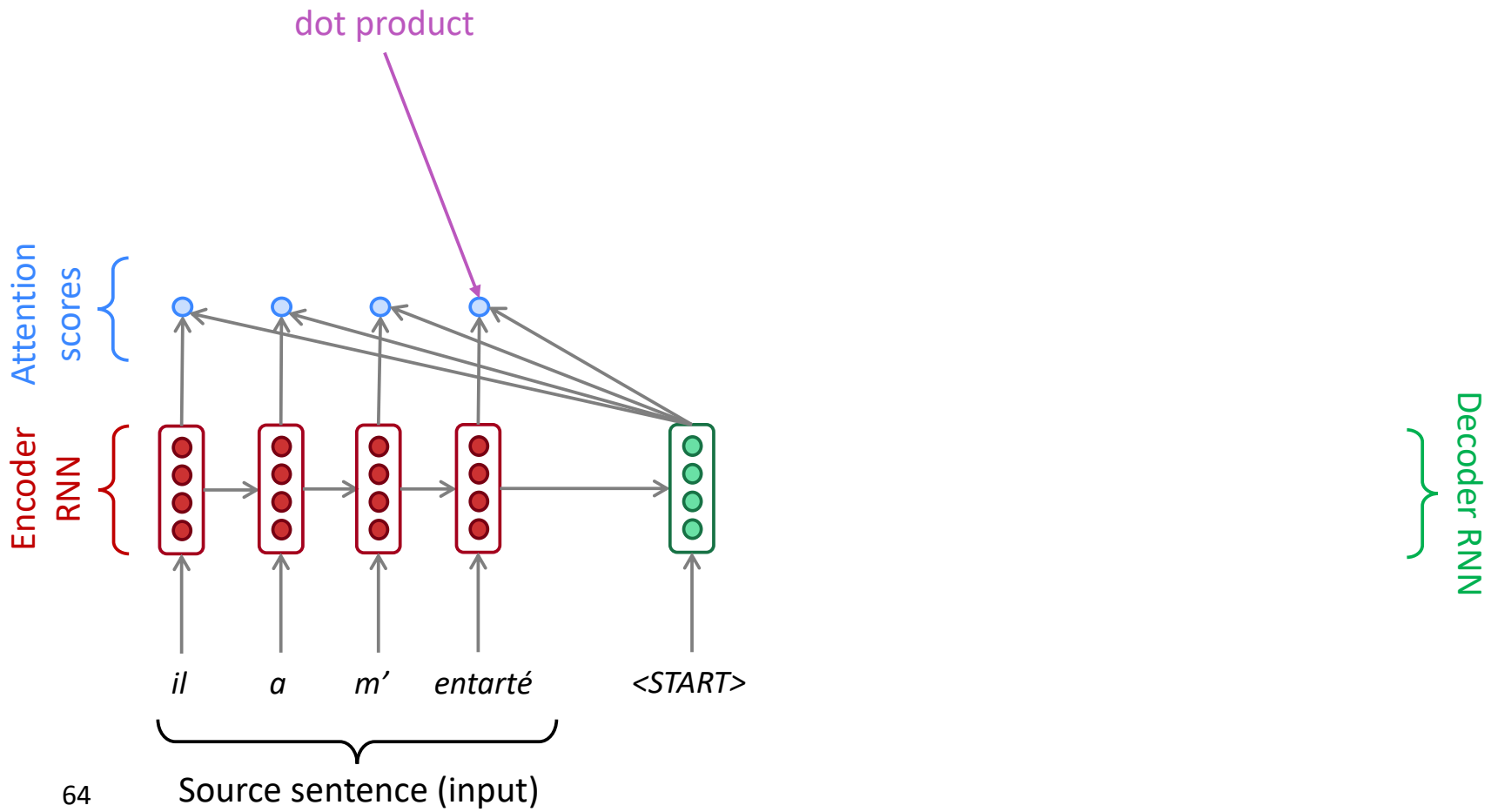
Sequence-to-sequence with attention



63

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

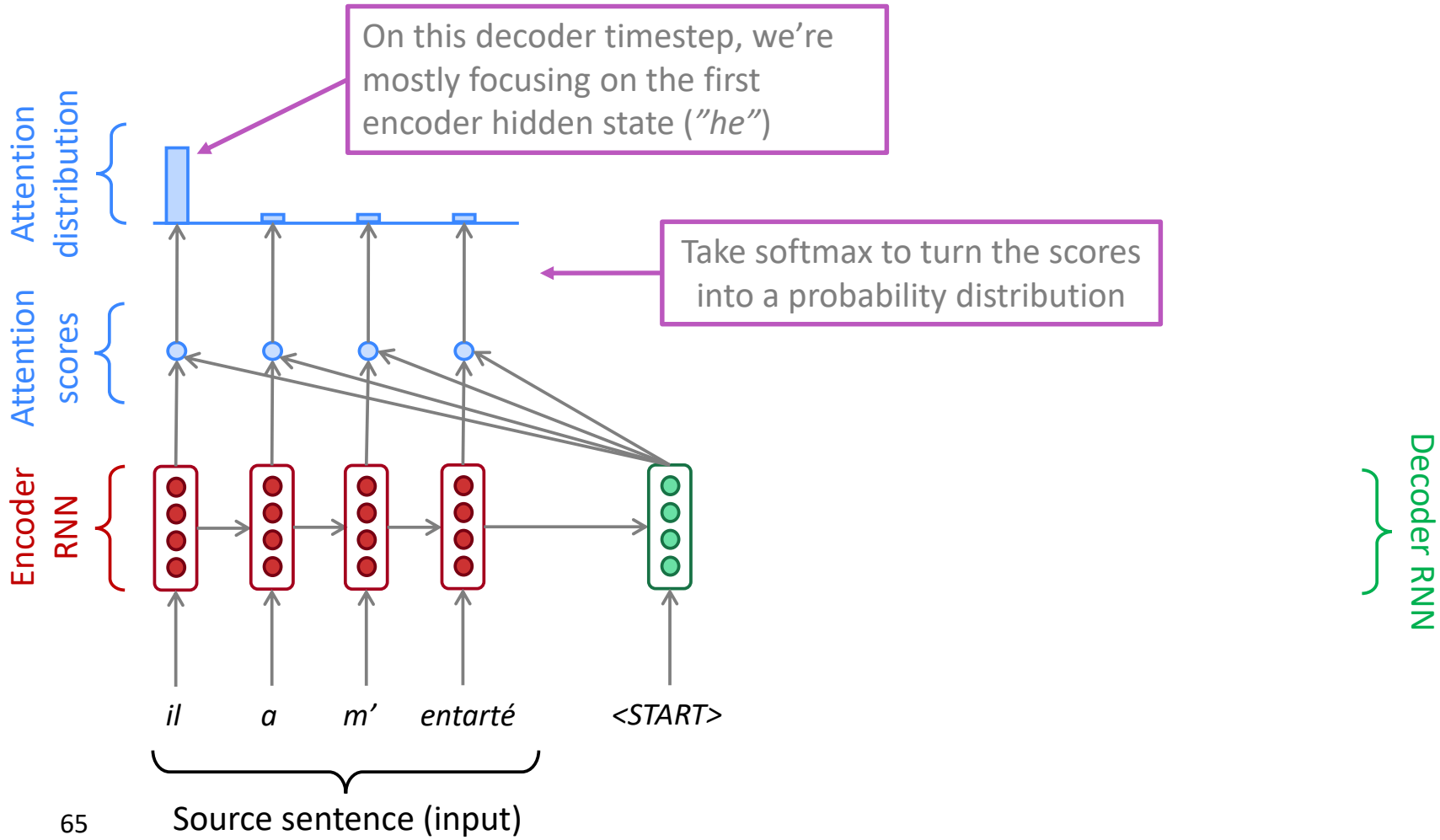
Sequence-to-sequence with attention



64

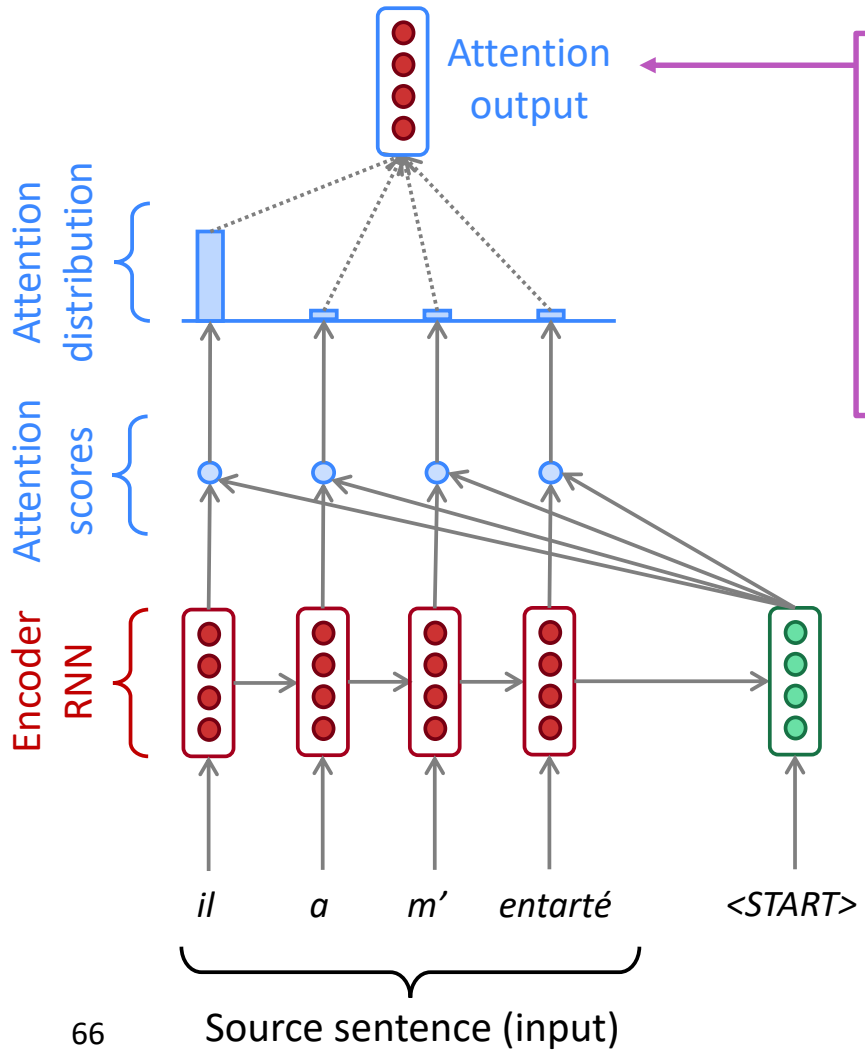
Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Sequence-to-sequence with attention



Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Sequence-to-sequence with attention



Use the attention distribution to take a **weighted sum** of the encoder hidden states.

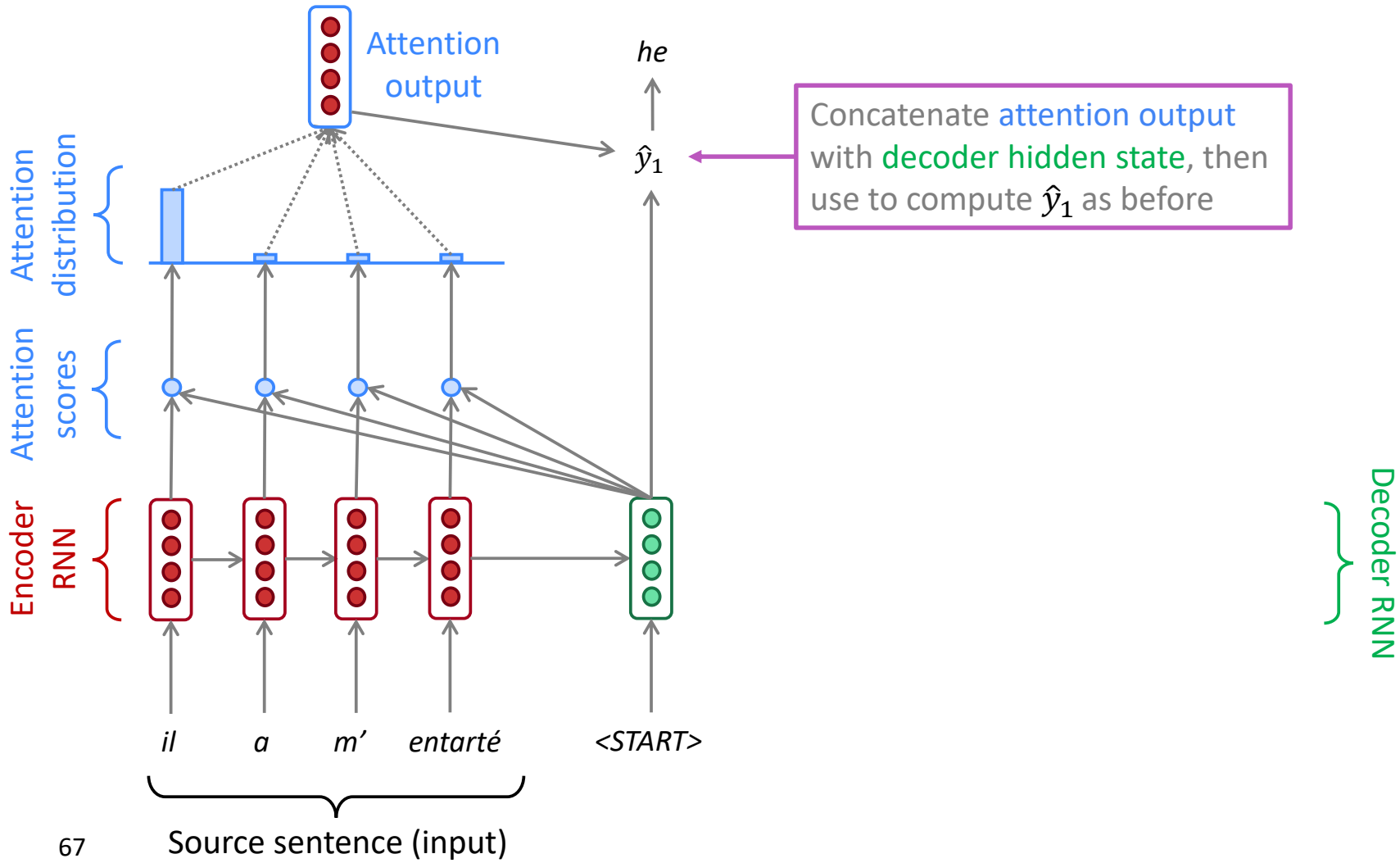
The attention output mostly contains information from the hidden states that received high attention.

Decoder RNN

66

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

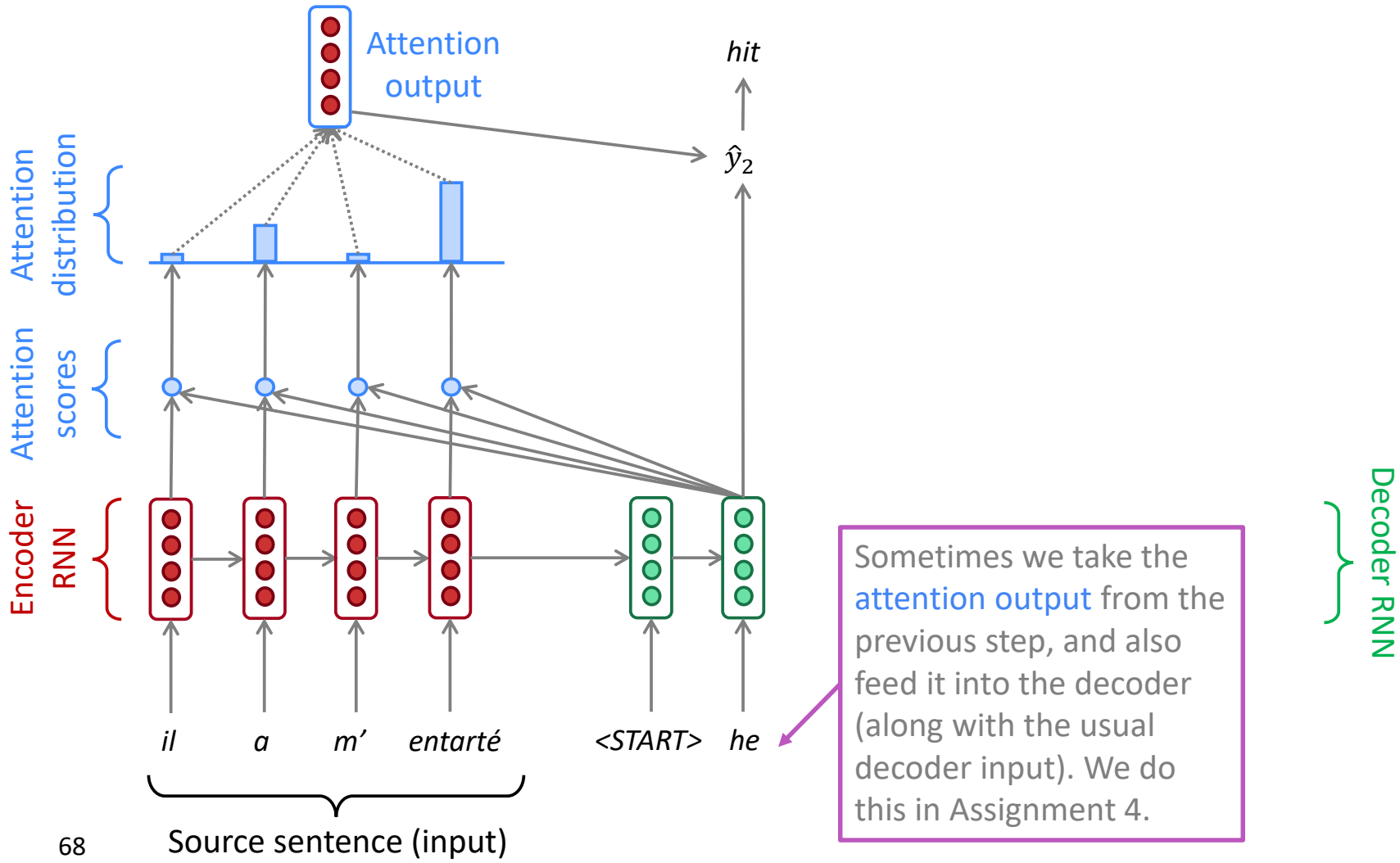
Sequence-to-sequence with attention



67

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

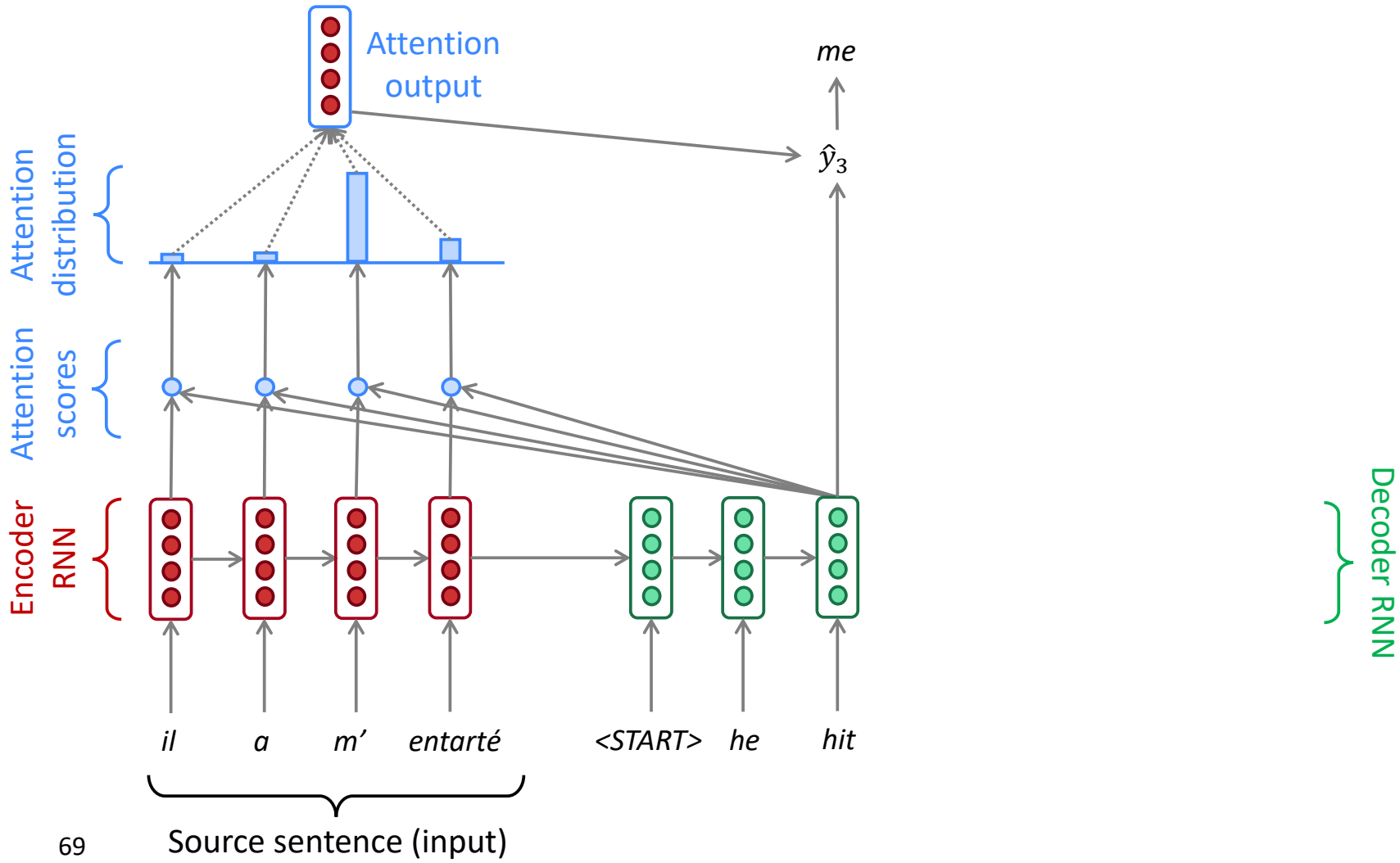
Sequence-to-sequence with attention



68

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

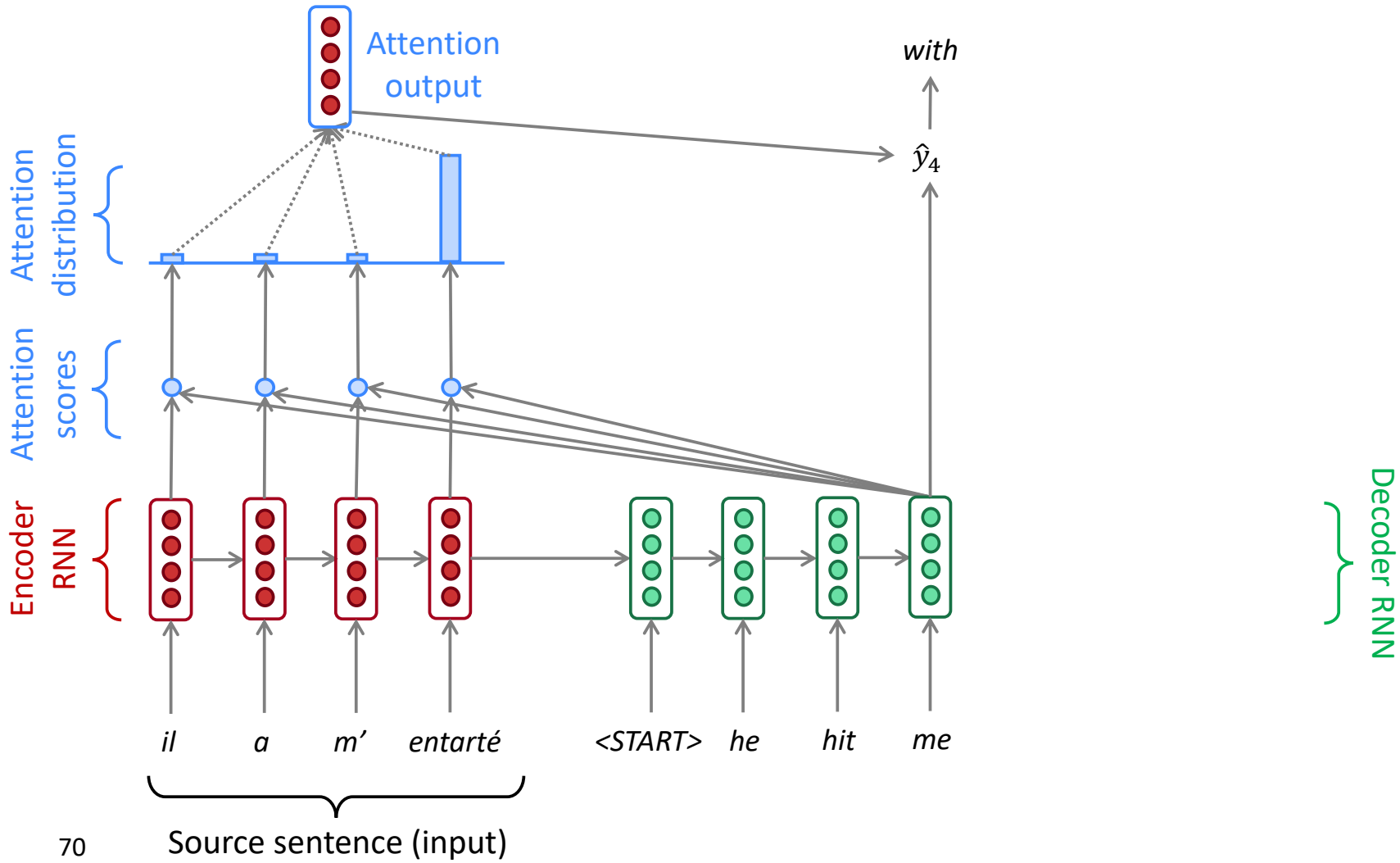
Sequence-to-sequence with attention



69

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

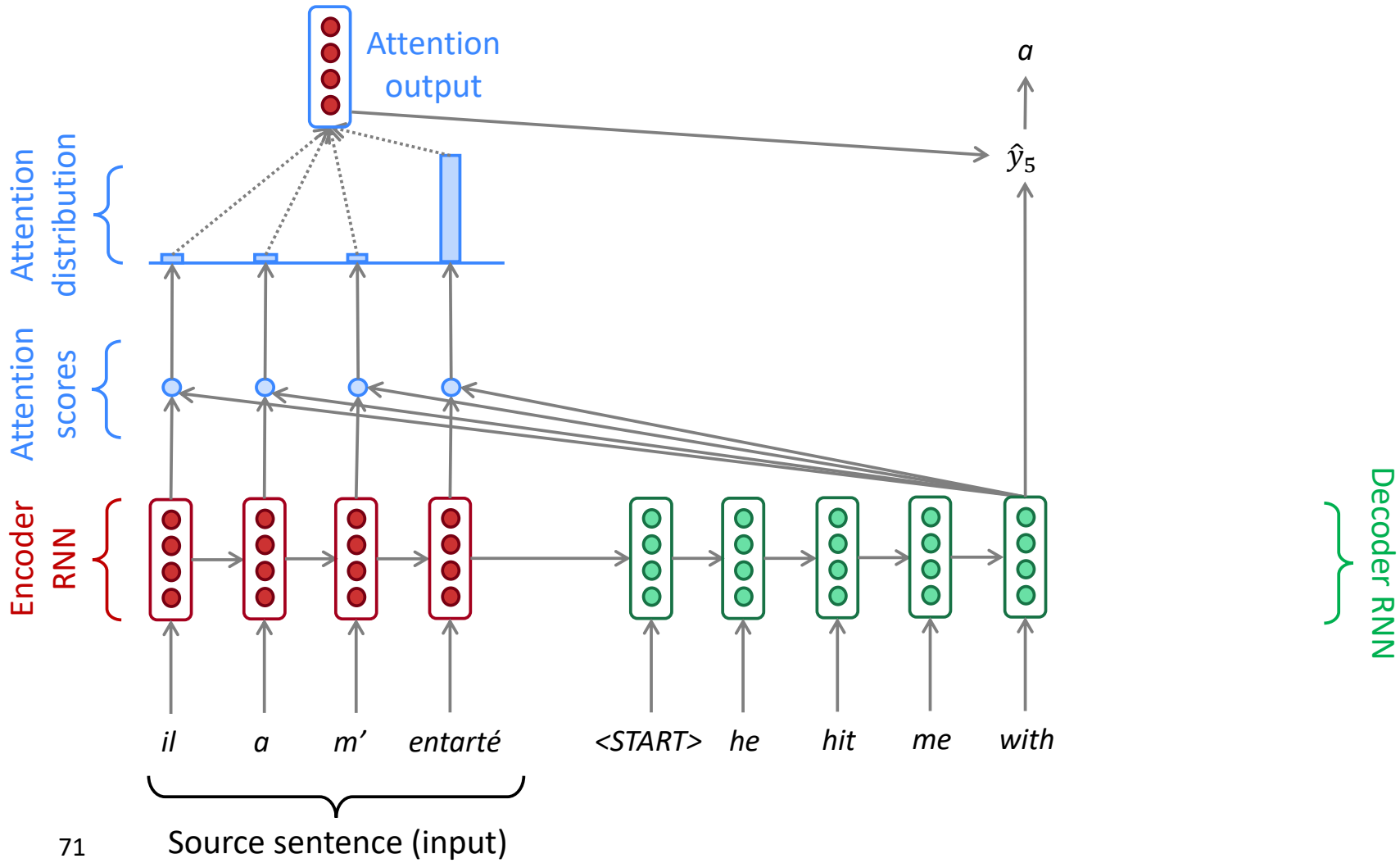
Sequence-to-sequence with attention



70

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

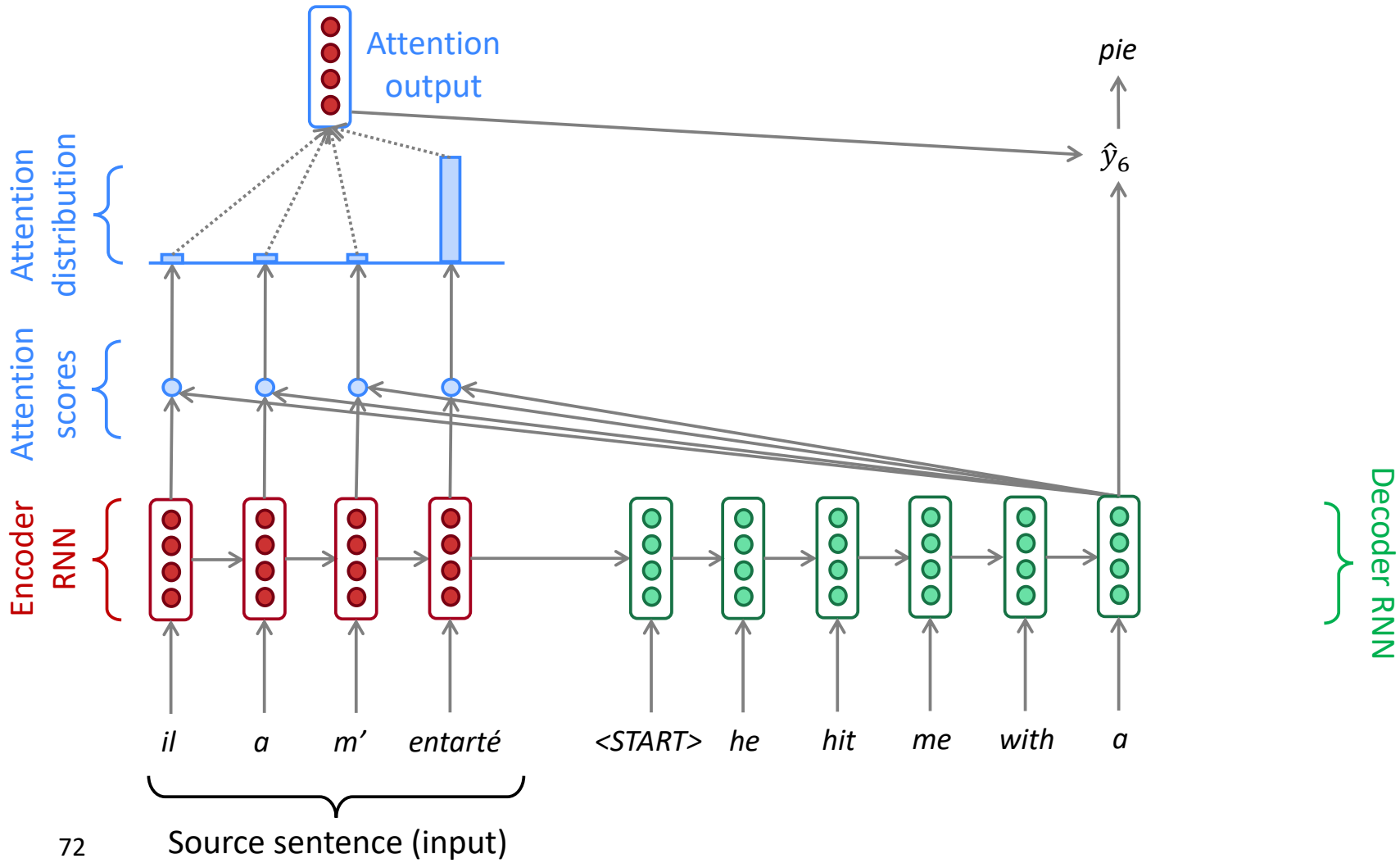
Sequence-to-sequence with attention



71

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Sequence-to-sequence with attention



72

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Attention: in equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

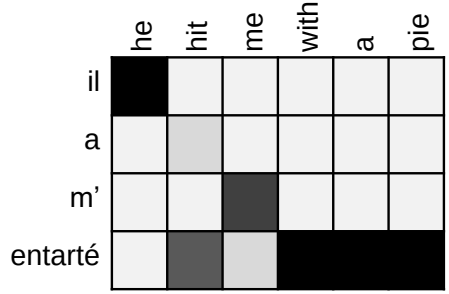
$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

73

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Attention is great

- Attention significantly improves NMT performance
  - It's very useful to allow decoder to focus on certain parts of the source
- Attention solves the bottleneck problem
  - Attention allows decoder to look directly at source; bypass bottleneck
- Attention helps with vanishing gradient problem
  - Provides shortcut to faraway states
- Attention provides some interpretability
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get (soft) alignment for free!
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



74

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Attention is a *general* Deep Learning technique

- We've seen that attention is a great way to improve the sequence-to-sequence model for Machine Translation.
- However: You can use attention in many architectures (not just seq2seq) and many tasks (not just MT)

- **More general definition of attention**:
  - Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.

- We sometimes say that the query *attends to* the values.
- For example, in the seq2seq + attention model, each decoder hidden state (query) *attends to* all the encoder hidden states (values).

75

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Attention is a *general* Deep Learning technique

More general definition of attention:

Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.

Intuition:

- The weighted sum is a *selective summary* of the information contained in the values, where the query determines which values to focus on.
- Attention is a way to obtain a *fixed-size representation of an arbitrary set of representations* (the values), dependent on some other representation (the query).

76

Christopher Manning, Natural Language Processing with Deep Learning, 2019 (slides)

Content

1. Machine Translation (MT)
2. Machine translation evaluation
3. Statistical machine translation (SMT)
4. Neural machine translation (NMT) based on RNNs
5. Neural machine translation (NMT) with attentions