

Statistical Machine Translation

CSCI 1460: Computational Linguistics

Lecture 14

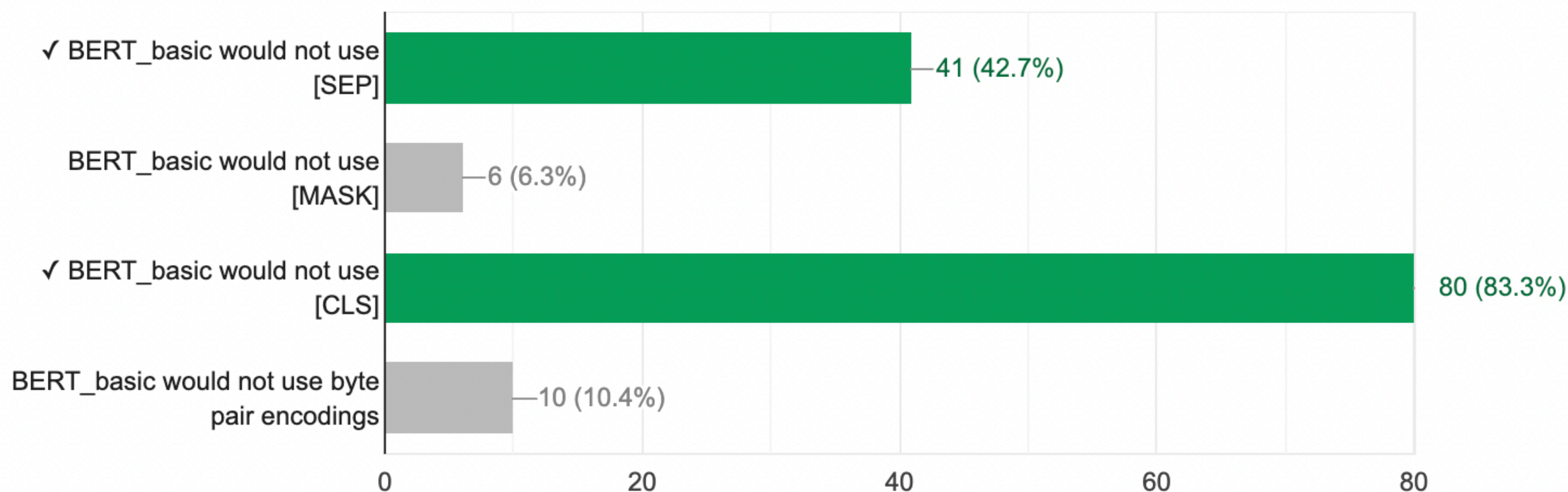
Ellie Pavlick

Fall 2023

BERT is trained with both a masked language modeling (MLM) objective and a next sentence prediction (NSP) objective. Since BERT was released, studies have suggested that the NSP objective doesn't do much to improve performance. Imagine you want to train a new BERT model, BERT_basic, which uses only MLM and does not use NSP. In which of the following ways would your BERT_basic likely differ from BERT?

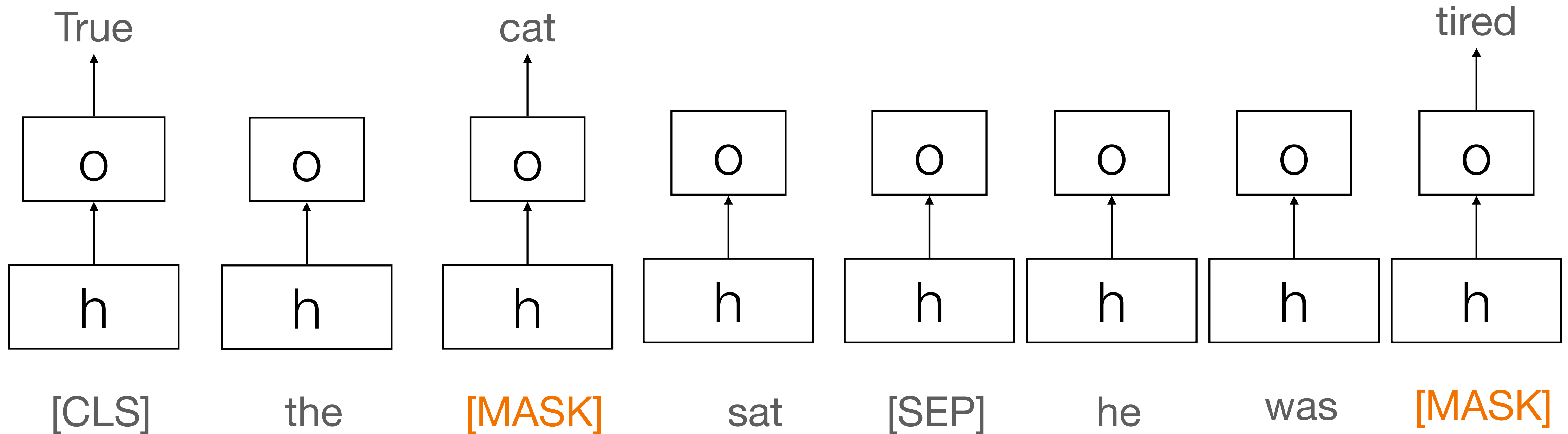


34 / 96 correct responses



BERT

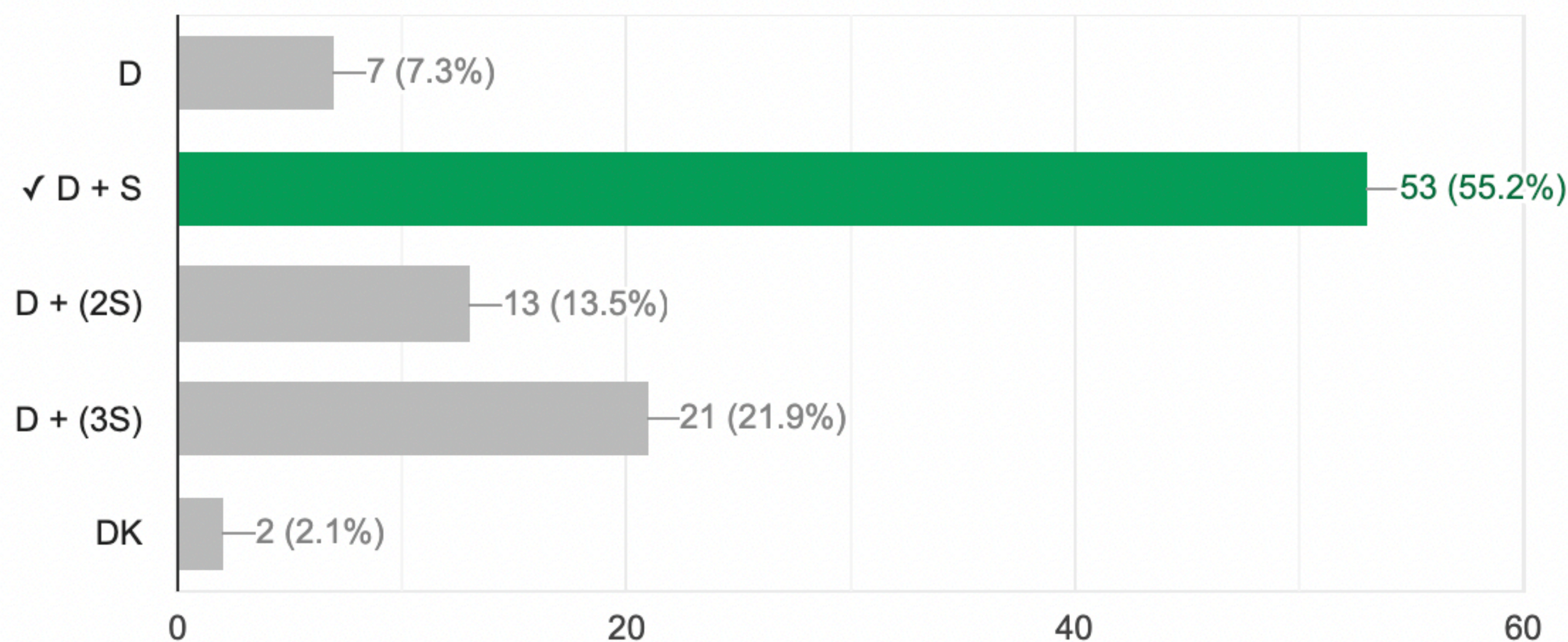
Masked Language Modeling



The below figure shows a baseline RNN model trained for sentiment analysis. It uses D dimensional static word embeddings for the input layer (blue). It then uses two hidden layers (orange and green), each which uses K dimensional hidden states. Assume ELMo uses S dimensions per token at each layer. If I augment the baseline RNN model with ELMo, using the method discussed in lecture, what would be the input dimension of the ELMo-augmented model?

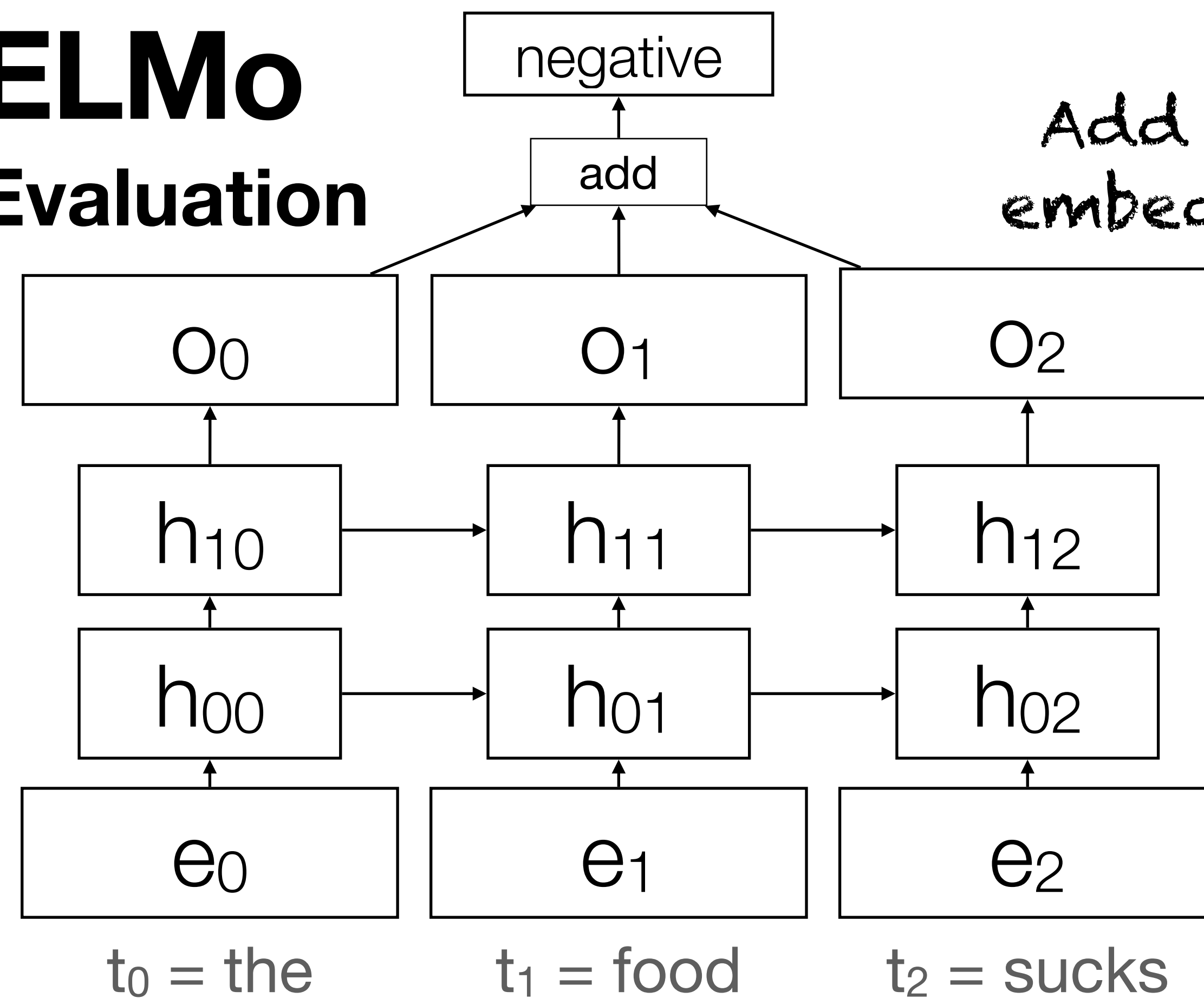


53 / 96 correct responses



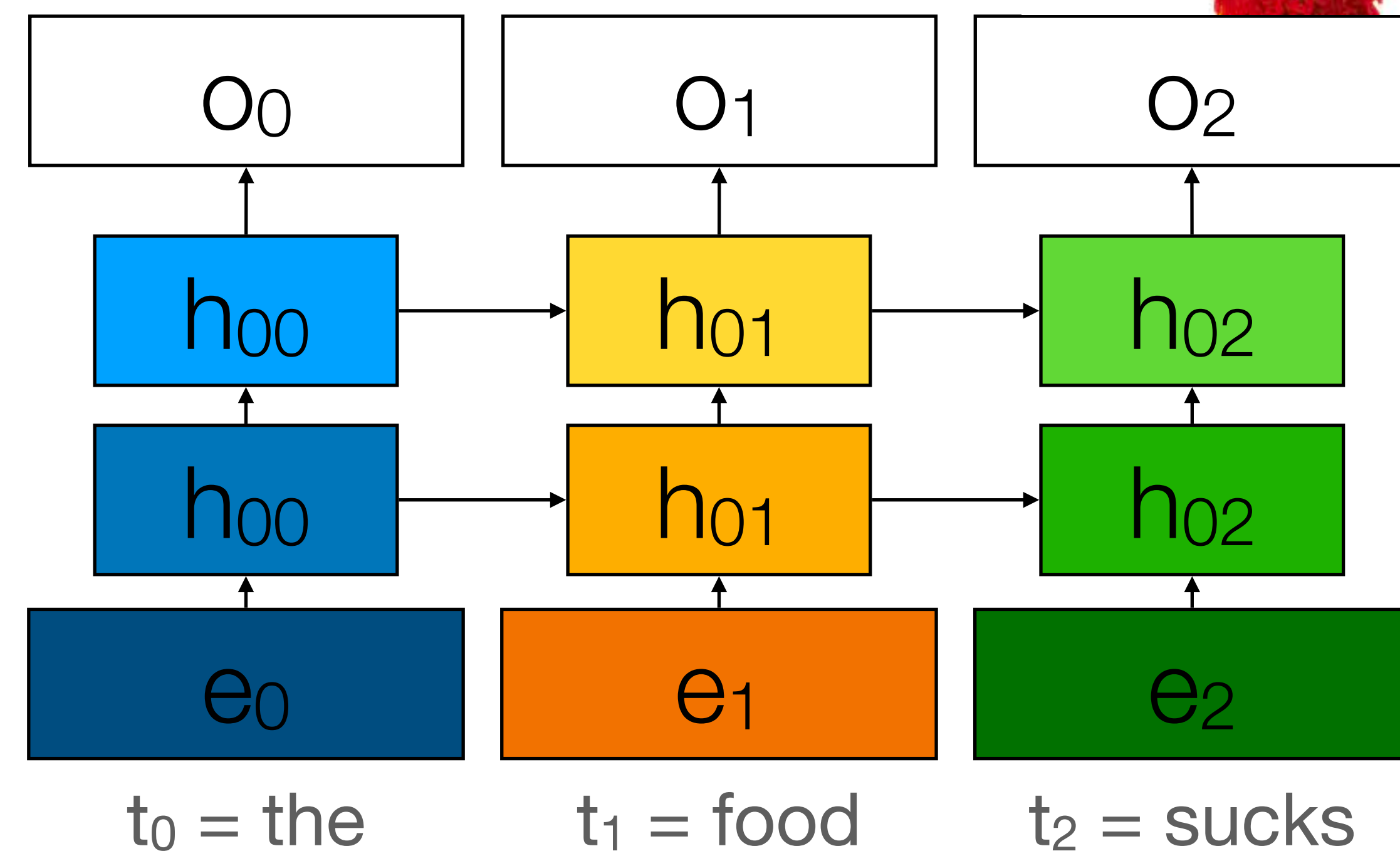
ELMo

Evaluation



"Baseline+ELMo"

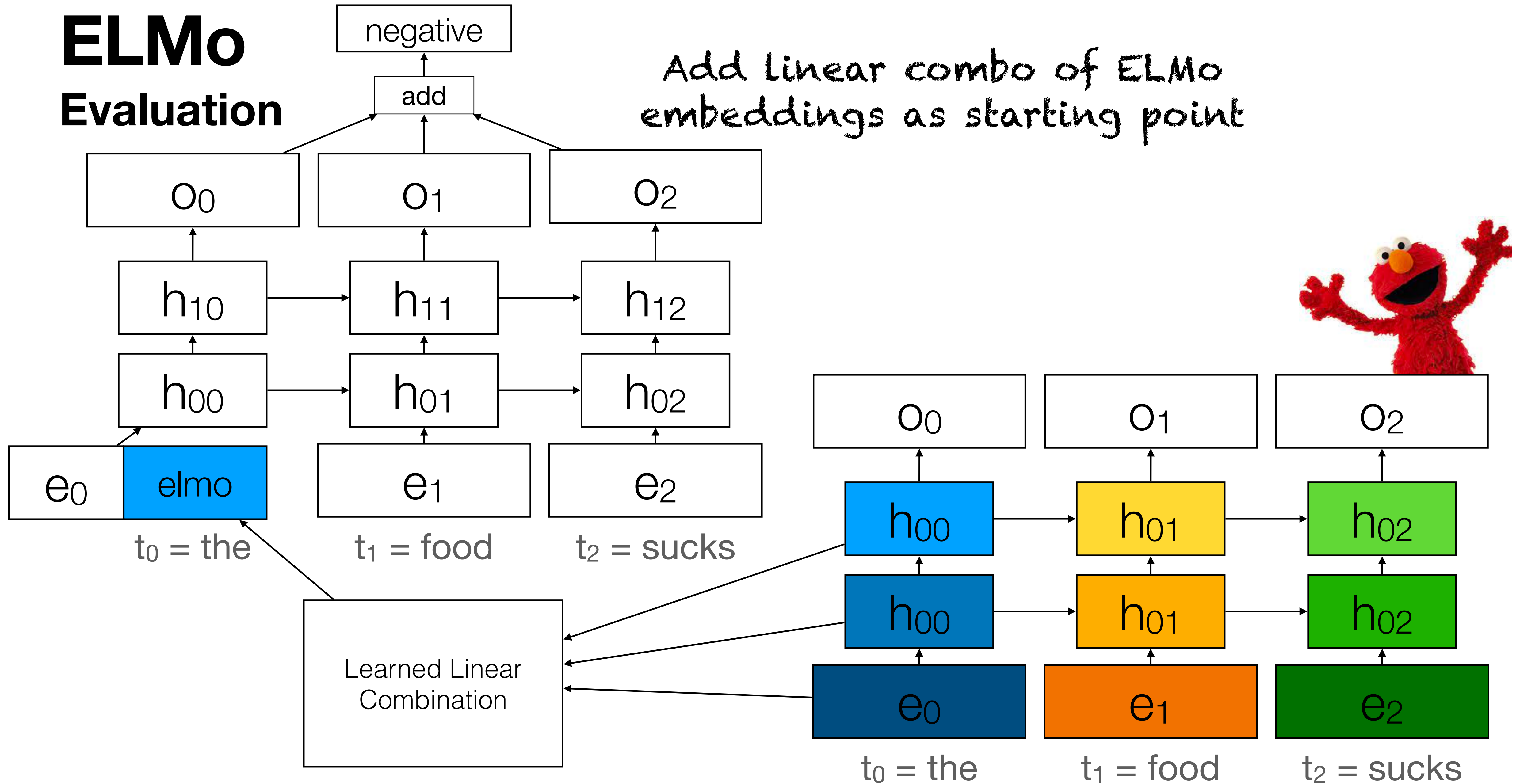
Add linear combo of ELMo embeddings as starting point



"Baseline+ELMo"

Add linear combo of ELMo embeddings as starting point

ELMo Evaluation

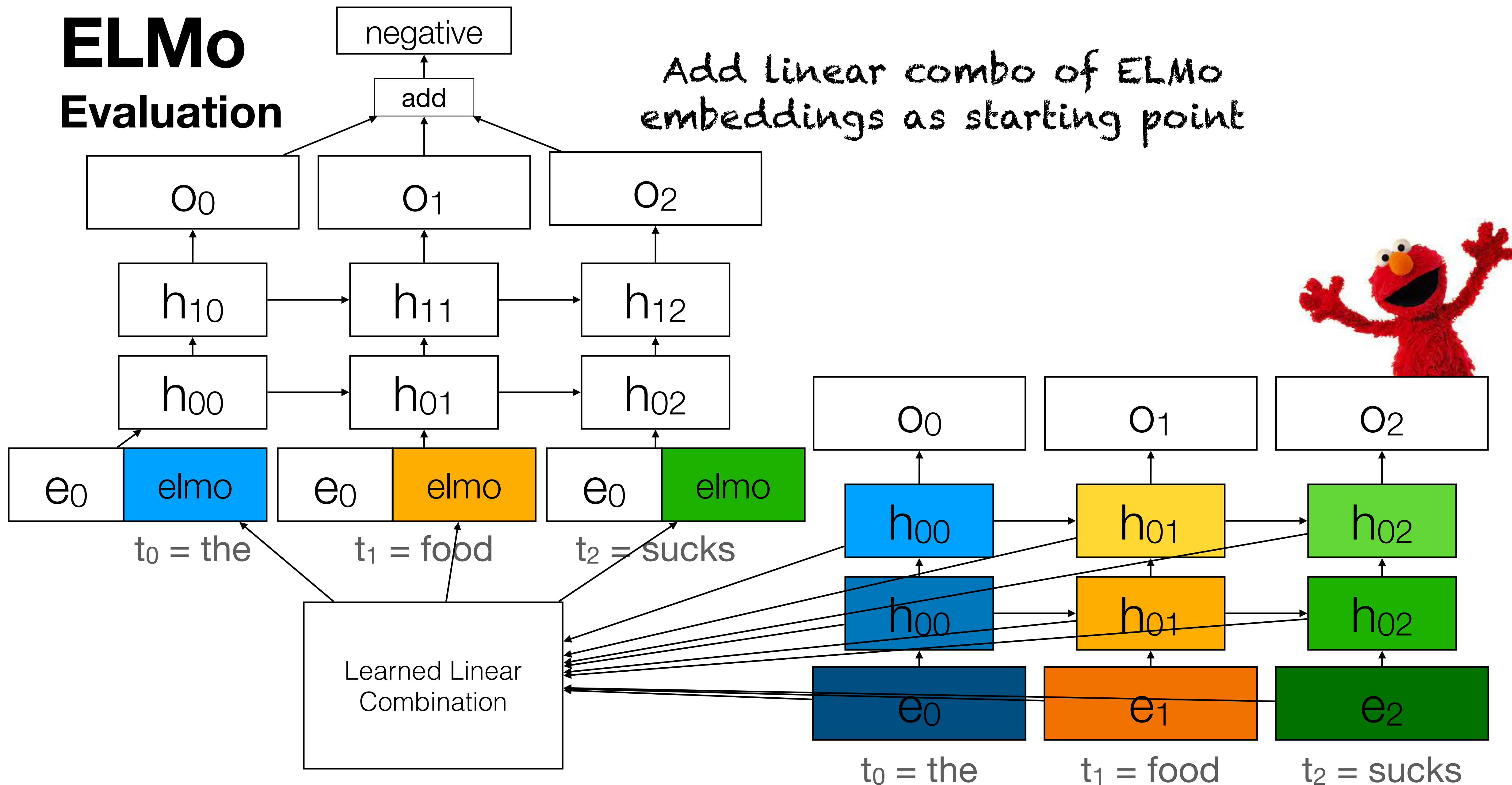


ELMo

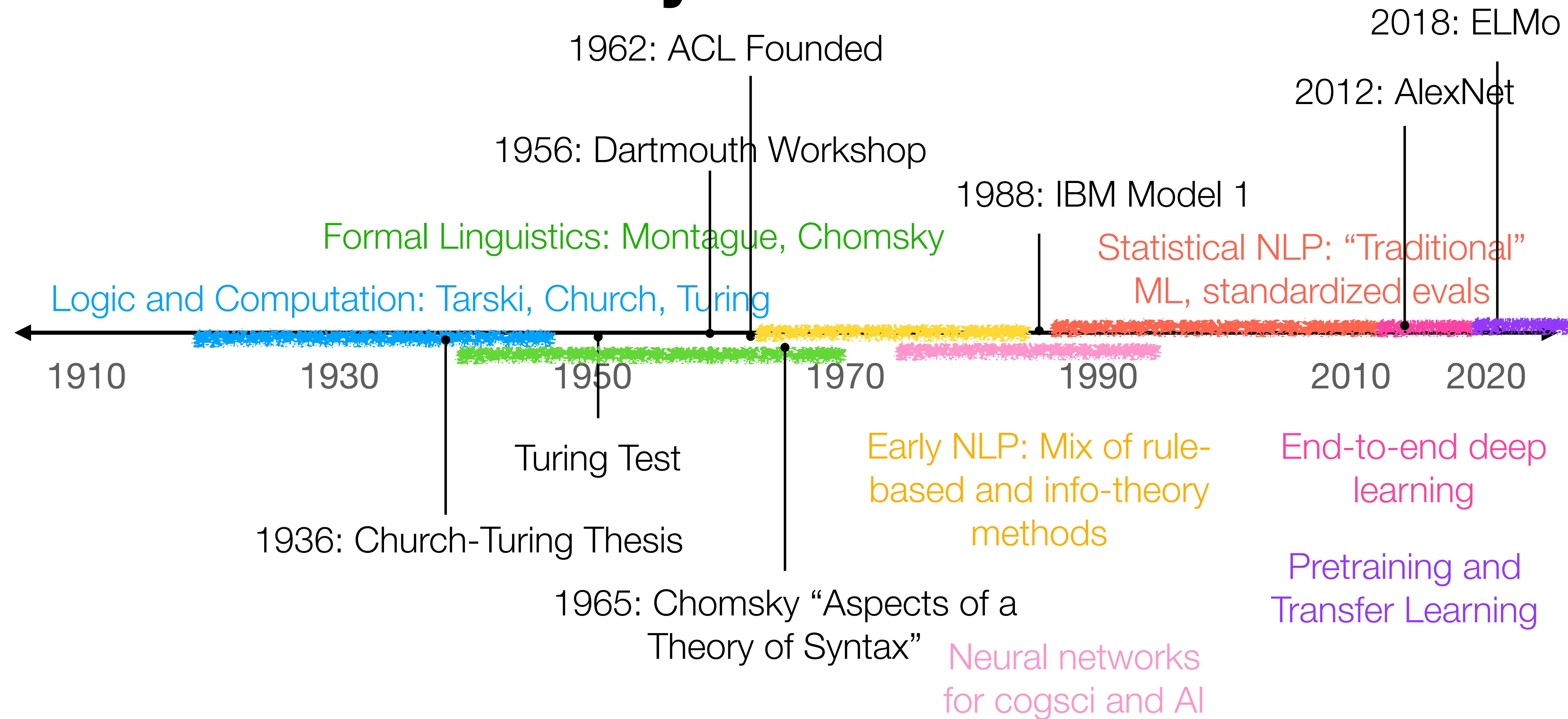
Evaluation

"Baseline+ELMo"

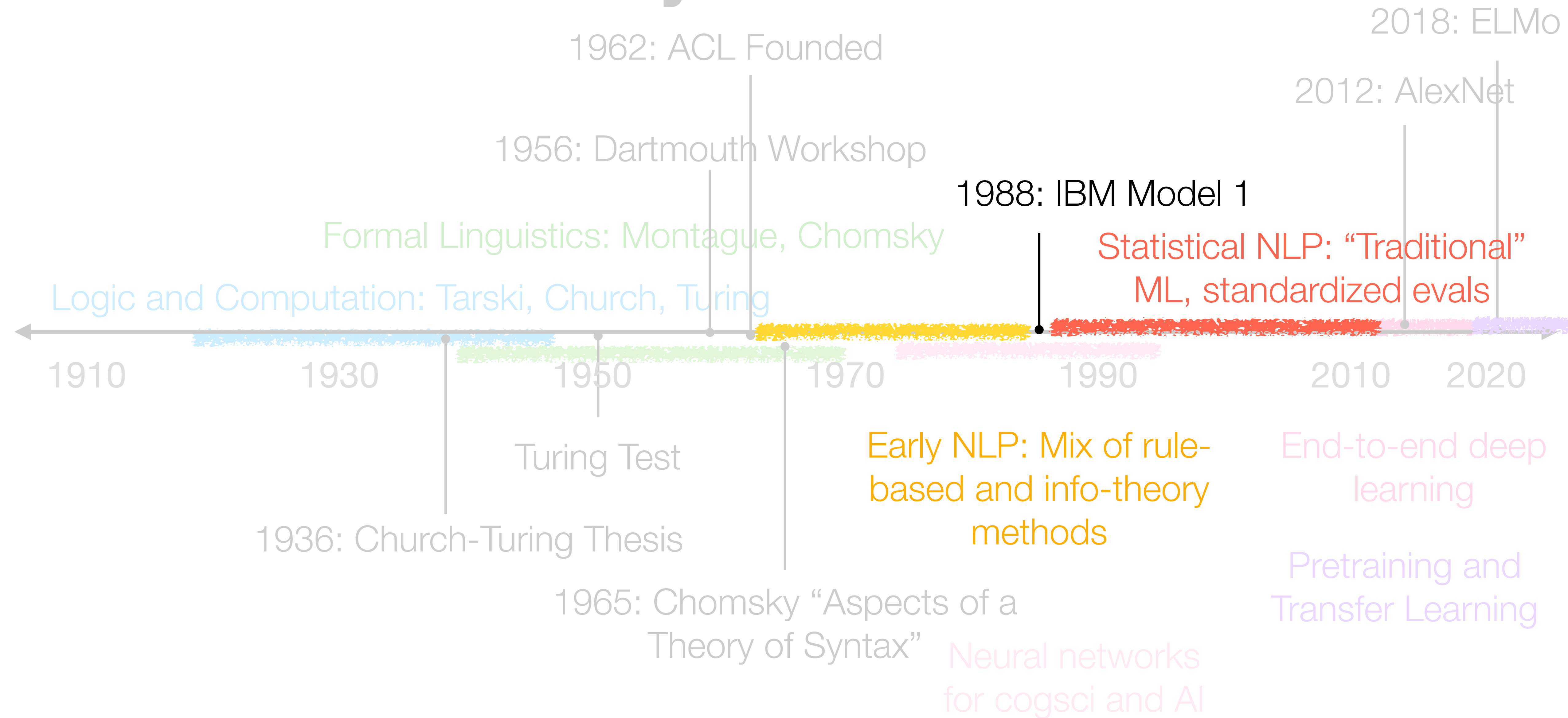
Add linear combo of ELMo embeddings as starting point



NLP: A brief history



NLP: A brief history



Topics

- Language Diversity and Challenges
- Noisy Channel Models for SMT
 - Translation Model (Word Alignment)
 - IBM Alignment Models
 - EM Algorithm
 - Language Model
 - Decoding

Topics

- **Language Diversity and Challenges**
- Noisy Channel Models for SMT
 - Translation Model (Word Alignment)
 - IBM Alignment Models
 - EM Algorithm
 - Language Model
 - Decoding

Why is MT so hard?

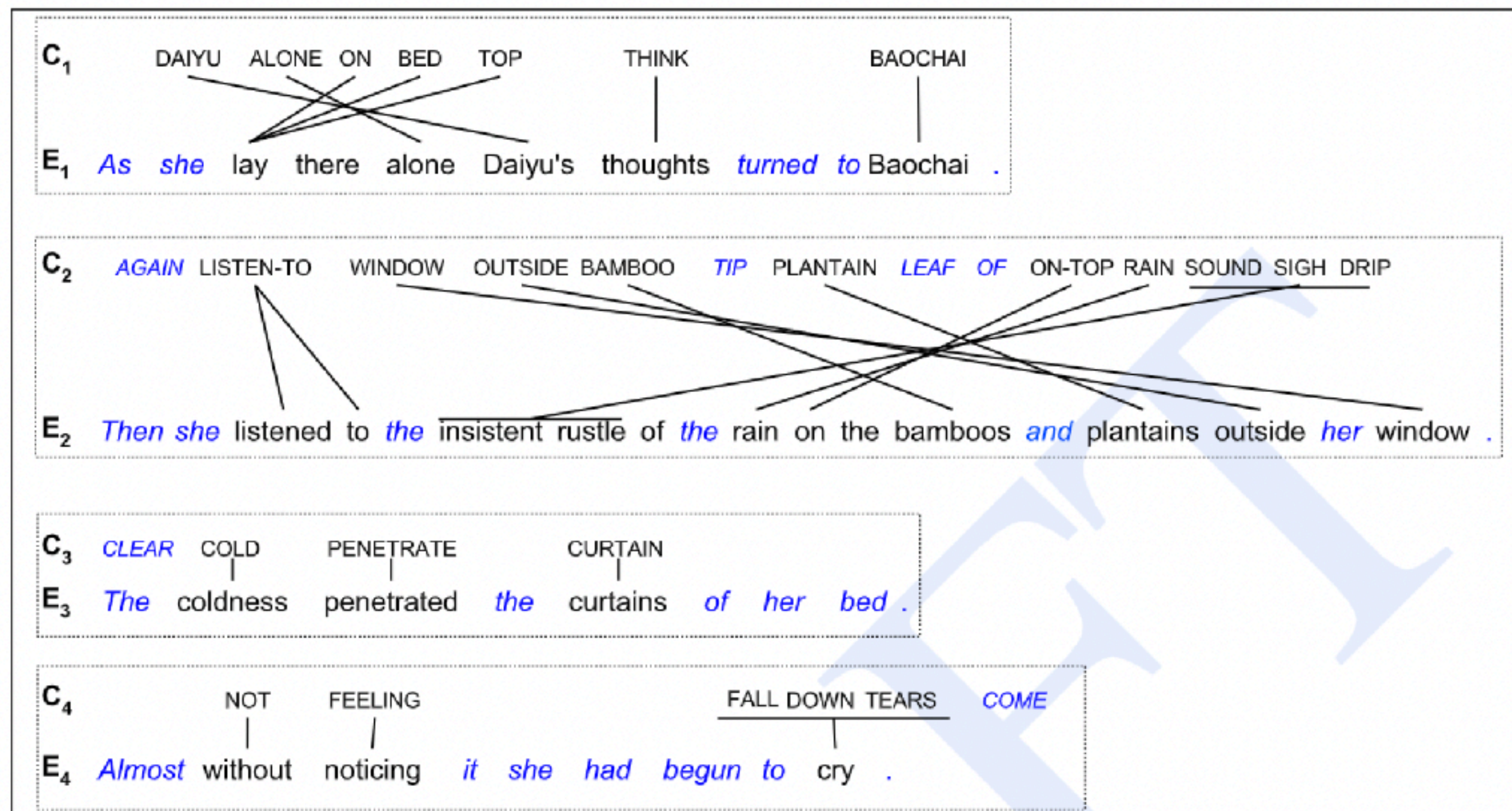


Figure 25.1 A Chinese passage from *Dream of the Red Chamber*, with the Chinese words represented by English glosses IN SMALL CAPS. Alignment lines are drawn between 'Chinese' words and their English translations. Words in italics are Chinese words not translated into English, or English words not in the original Chinese.

- perfect translation (faithful and fluent) is not possible!
- bamboo-tipped plantain leaf = bamboos and plantains
- curtains = curtains of her bed

Why is MT so hard?

- Morphology
 - Isolating (one morpheme per word) vs. polysynthetic
 - tuntussuqatarniksaitengqiggtuq (Yupik language, from Alaska)
 - tuntu -ssur -qatar -ni -ksaite -ngqiggte -uq
 - reindeer -hunt -FUTURE -say -NEG -again -3SG.IND
 - "He had not yet said again that he was going to hunt reindeer."
 - Agglutinative (morphemes are separable, e.g., German) vs. fusional languages (e.g., Russian)
 - comí (Spanish)
 - í = first person and past tense

Why is MT so hard?

- Syntax
 - SVO: German, English, French, Mandarin
 - SOV: Hindi, Japanese
 - VSO: Irish, Arabic, Biblical Hebrew
- Argument structure and marking
 - Possession:
 - The man's house
 - Az ember haza: the man house-his (Hungarian)
 - Motion, manner
 - The bottle floated out
 - La botella salió flotando: The bottle exited floating (Spanish)

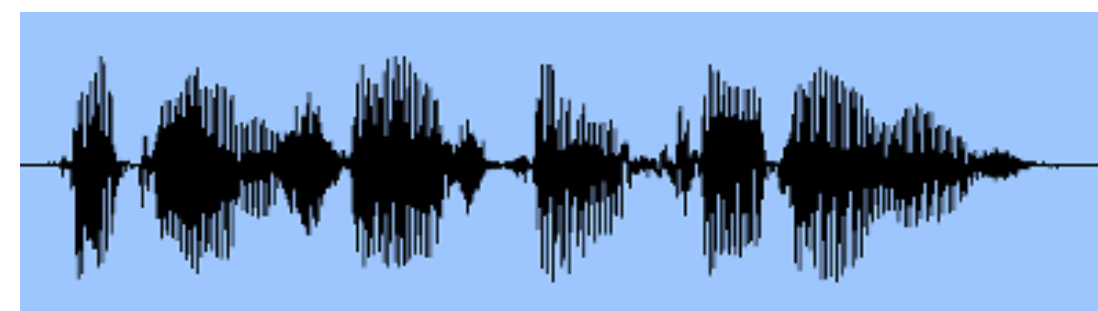
Why is MT so hard?

- Referential density:
 - “pro-drop” languages (e.g., Spanish, Mandarin) consider pronouns optional
 - Languages differ in how much “work” the listener has to do
- Lexicon: Every language has both polysemy and redundancy

Topics

- Language Diversity and Challenges
- **Noisy Channel Models for SMT**
 - Translation Model (Word Alignment)
 - IBM Alignment Models
 - EM Algorithm
 - Language Model
 - Decoding

Noisy Channel Model for Speech Recognition



audio wave

$$P(\text{Its easy to recognize speech} \mid \text{audio wave}) = 0.3$$

$$P(\text{Its easy to wreck a nice beach} \mid \text{audio wave}) = 0.5$$

$$P(\text{Its easy to recognize speech}) = 0.5$$

$$P(\text{Its easy to wreck a nice beach}) = 0.1$$

Noisy Channel Model

Its easy to recognize speech

Noisy Channel Model

- Warren Weaver: “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.’”
- Intuition: Consider a message sent across a noisy channel. To determine the message, you need to correct for the noise that was added.

Noisy Channel Model

$$P(tgt | src) = \frac{P(src | tgt)P(tgt)}{P(src)}$$

Noisy Channel Model

$$P(tgt | src) \propto P(src | tgt)P(tgt)$$

Noisy Channel Model

$$P(tgt | src) \propto P(src | tgt)P(tgt)$$

Translation Model

Noisy Channel Model

$$P(tgt | src) \propto P(src | tgt)P(tgt)$$

Translation Model

Intuition: source language is
a "corrupted" version of the
target language

Noisy Channel Model

$$P(tgt | src) \propto P(src | tgt)P(tgt)$$

Language Model

Noisy Channel Model

$$P(tgt | src) \propto P(src | tgt)P(tgt)$$

Language Model

(We have talked about this
extensively)

Topics

- Language Diversity and Challenges
- Noisy Channel Models for SMT
 - **Translation Model (Word Alignment)**
 - **IBM Alignment Models**
 - EM Algorithm
 - Language Model
 - Decoding

Word Alignment

Bitexts a.k.a. Bilingual Parallel Corpora

- Bitext: A corpus that contains translated pairs of texts
 - Can be aligned coarsely (e.g., document-level) or finely (sentence level)
 - Word-level alignment is rare

Word Alignment

Bitexts a.k.a. Bilingual Parallel Corpora

- Some exist naturally. E.g., EU translates all their documents (<https://www.statmt.org/europarl/>), authors hire translators to translate literature

Parallel Corpus (L1-L2)	Sentences	L1 Words	English Words
Bulgarian-English	406,934	-	9,886,291
Czech-English	646,605	12,999,455	15,625,264
Danish-English	1,968,800	44,654,417	48,574,988
German-English	1,920,209	44,548,491	47,818,827
Greek-English	1,235,976	-	31,929,703
Spanish-English	1,965,734	51,575,748	49,093,806
Estonian-English	651,746	11,214,221	15,685,733
Finnish-English	1,924,942	32,266,343	47,460,063
French-English	2,007,723	51,388,643	50,196,035
Hungarian-English	624,934	12,420,276	15,096,358
Italian-English	1,909,115	47,402,927	49,666,692
Lithuanian-English	635,146	11,294,690	15,341,983
Latvian-English	637,599	11,928,716	15,411,980
Dutch-English	1,997,775	50,602,994	49,469,373
Polish-English	632,565	12,815,544	15,268,824
Portuguese-English	1,960,407	49,147,826	49,216,896
Romanian-English	399,375	9,628,010	9,710,331

Word Bitexts

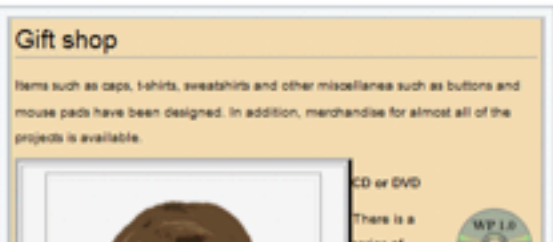
- Some all their documents www.statmt.org/eu translators to transl
- Can get creative if we are okay with “weakly aligned” —e.g., pairs of new articles or Wikipedia documents on the same topic

Natural language processing

From Wikipedia, the free encyclopedia

This article is about natural language processing done by computers. For the natural language processing done by the human brain, see [Language processing in the brain](#).

Natural language processing (NLP) is a subfield of [linguistics](#), [computer science](#), and [artificial intelligence](#) concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of [natural language](#) data. The goal is a computer capable of "understanding" the contents of documents, including the [contextual](#) nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.



Challenges in natural language processing frequen

Contents [\[hide\]](#)

1 History

1.1 Symbolic NLP (1950s – early 1990s)

1.2 Statistical NLP (1990s–2010s)

1.3 Neural NLP (present)

Procesamiento de lenguajes naturales

(Redirigido desde «[NLP](#)»)

El **procesamiento de lenguaje natural**,^{1 2} abreviado **PLN**^{3 4} —en inglés, *natural language processing*, NLP— es un campo de las [ciencias de la computación](#), de la [inteligencia artificial](#) y de la [lingüística](#) que estudia las interacciones entre las computadoras y el lenguaje humano. Se ocupa de la formulación e investigación de mecanismos eficaces computacionalmente para la comunicación entre personas y máquinas por medio del [lenguaje natural](#), es decir, de las [lenguas del mundo](#). No trata de la comunicación por medio de lenguas naturales de una forma abstracta, sino de diseñar mecanismos para comunicarse que sean eficaces computacionalmente —que se puedan realizar por medio de programas que ejecuten o simulen la comunicación—. Los modelos aplicados se enfocan no solo a la comprensión del lenguaje de por sí, sino a aspectos generales cognitivos humanos y a la organización de la memoria. El lenguaje natural sirve solo de medio para estudiar estos fenómenos. Hasta la década de 1980, la mayoría de los sistemas de PLN se basaban en un complejo conjunto de reglas diseñadas a mano. A partir de finales de 1980, sin embargo, hubo una revolución en PLN con la introducción de [algoritmos](#) de [aprendizaje automático](#) para el procesamiento del lenguaje.^{5 6}

Índice [\[ocultar\]](#)

1 Historia

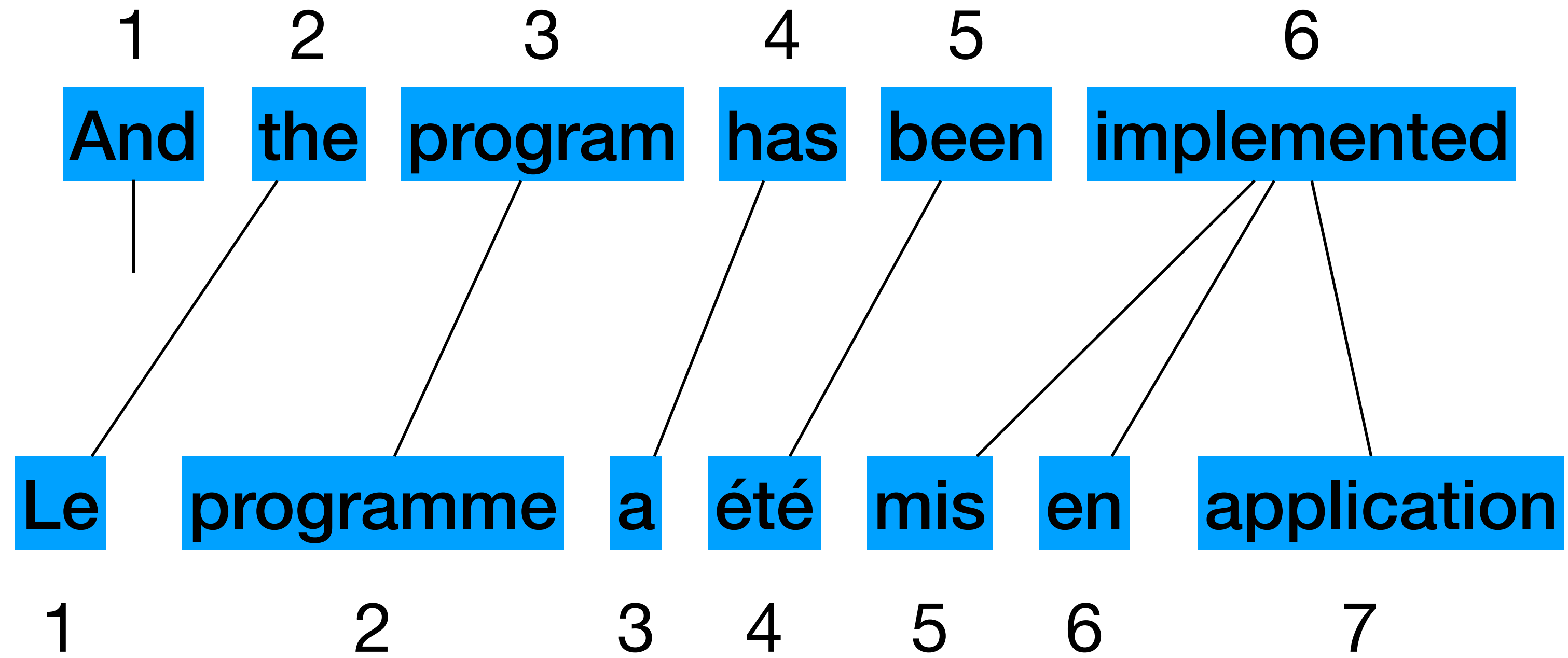
2 Dificultades en el procesamiento de lenguaje natural

2.1 Ambigüedad

Estonian-English	651,740	11,214,221	15,085,733
Finnish-English	1,924,942	32,266,343	47,460,063
French-English	2,007,723	51,388,643	50,196,035
Hungarian-English	624,934	12,420,276	15,096,358
Italian-English	1,909,115	47,402,927	49,666,692
Lithuanian-English	635,146	11,294,690	15,341,983
Latvian-English	637,599	11,928,716	15,411,980
Dutch-English	1,997,775	50,602,994	49,469,373
Polish-English	632,565	12,815,544	15,268,824
Portuguese-English	1,960,407	49,147,826	49,216,896
Romanian-English	399,375	9,628,010	9,710,331

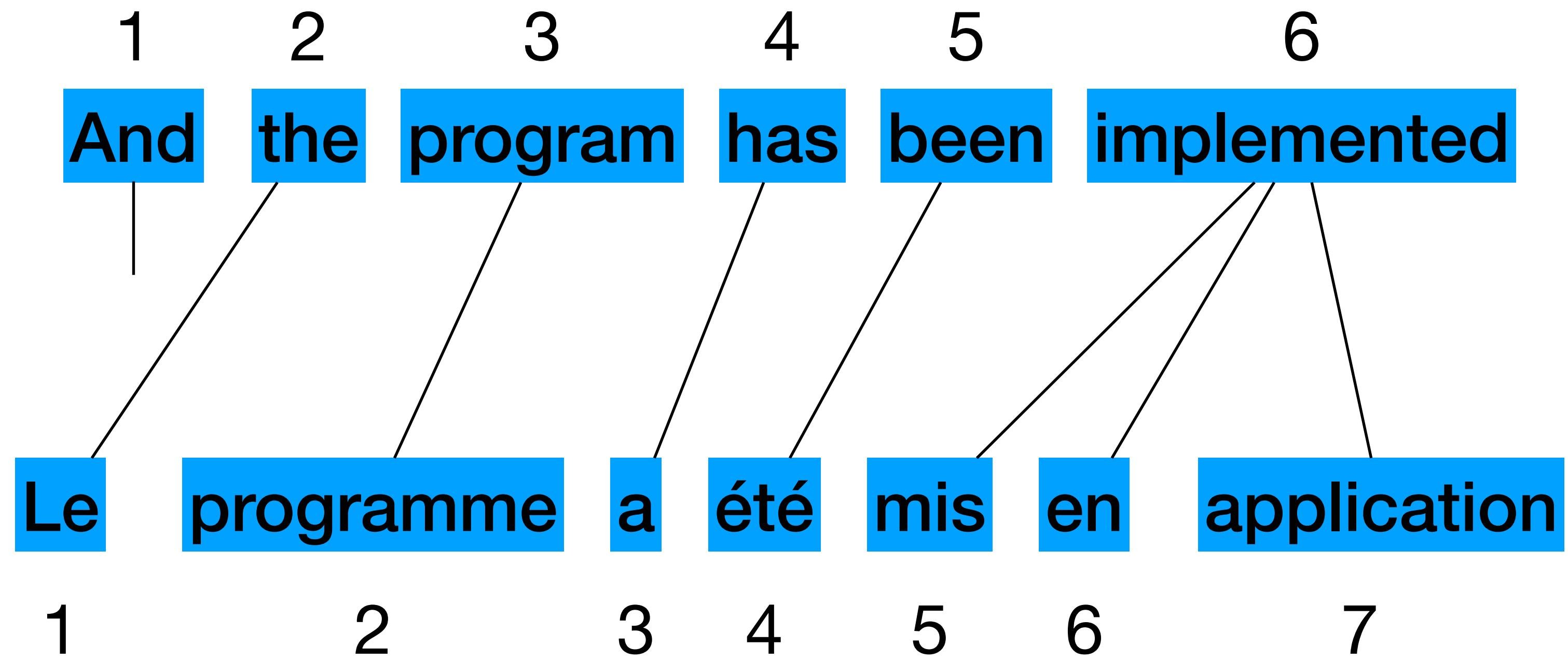
Word Alignment

IBM Model 1



Word Alignment

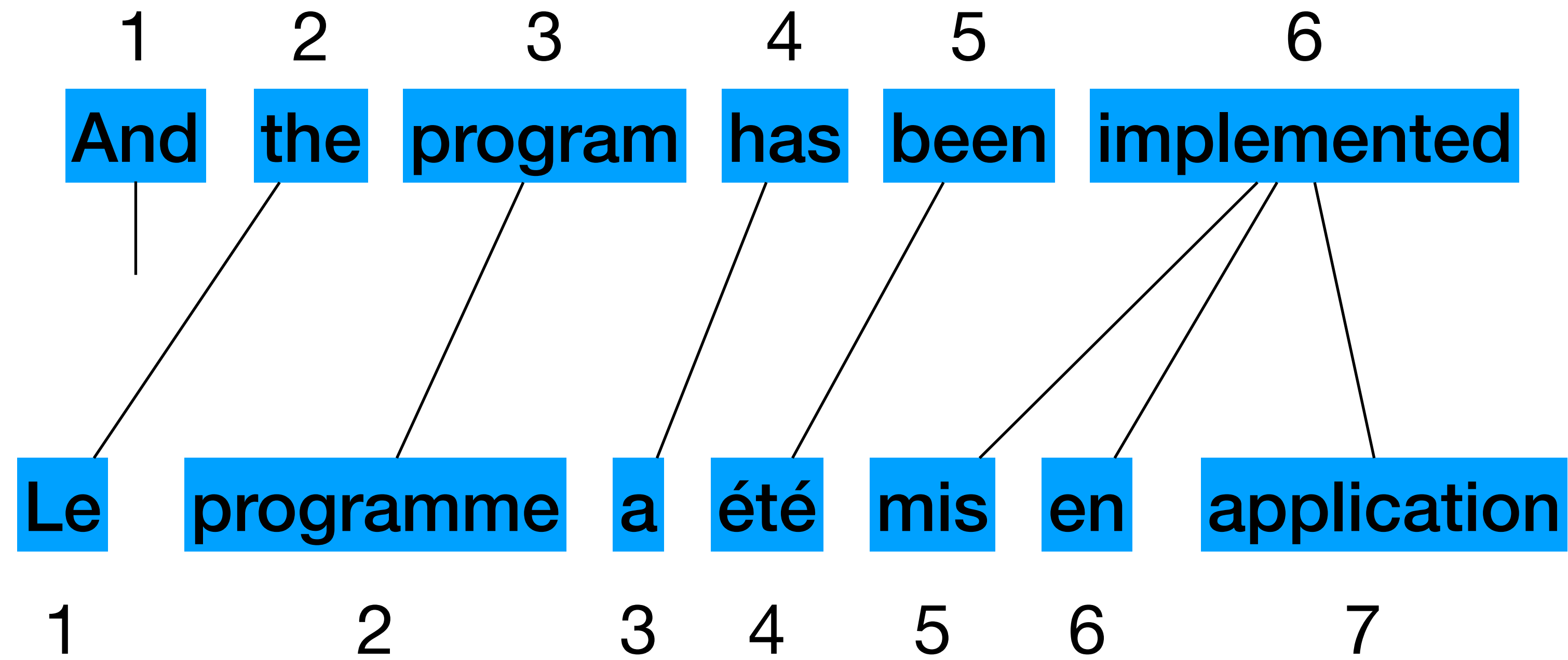
IBM Model 1



a	2	3	4	5	6	6	6
---	---	---	---	---	---	---	---

Word Alignment

IBM Model 1

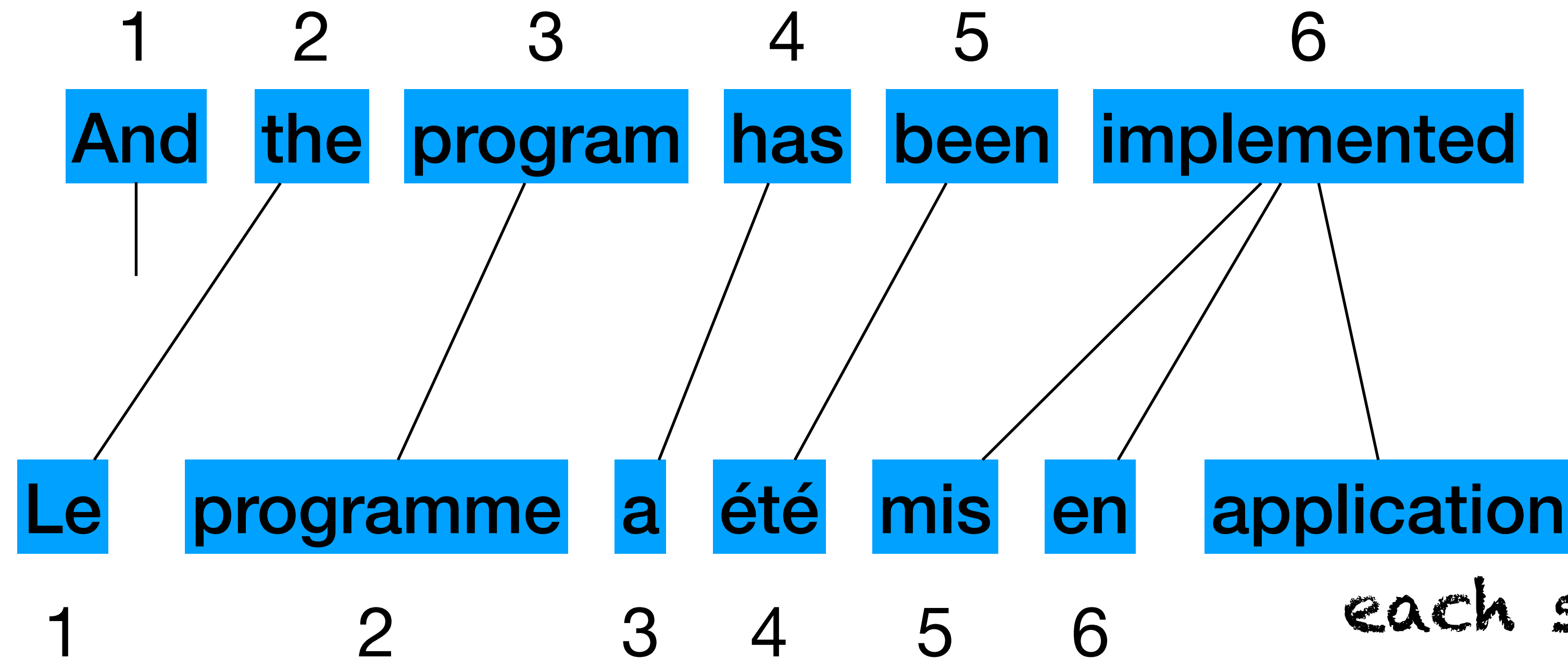


a	2	3	4	5	6	6	6
---	---	---	---	---	---	---	---

each source word
aligns to exactly
one target word

Word Alignment

IBM Model 1

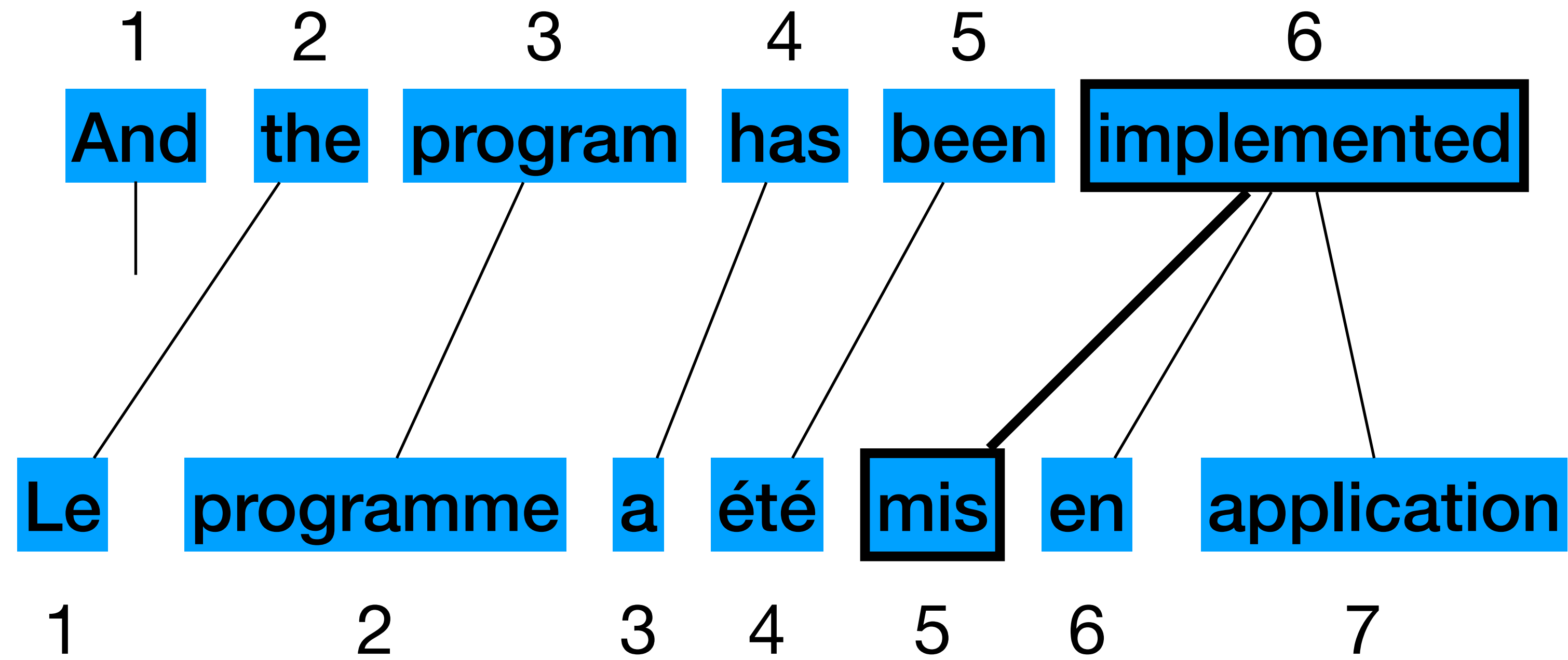


a	2	3	4	5	6	6	6
---	---	---	---	---	---	---	---

each source word
aligns to exactly
one target word
(later models
relax this)

Word Alignment

IBM Model 1



a	2	3	4	5	6	6	6
---	---	---	---	---	---	---	---

$$a[5] = 6$$

Word Alignment

IBM Model 1

- First (simplest) automatic unsupervised alignment model from IBM
- Generative Story:
 - Choose length L for src sentence
 - Choose an alignment $A = a_1 \dots a_L$
 - Then, generate target position t_i by translating whatever source phrase is aligned to position i in the target
- $\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$
 - find the alignment that makes each observed source word most likely given its aligned target word

Word Alignment

IBM Model 1

- Dilemma:
 - We want alignments so that we can figure out the probabilities of phrase translations, e.g., $P(s_i|t_j)$ for all s, t, i, j
 - To estimate those alignments, we need phrase translation probabilities
 - 😞
 - No fear! EM is here!

Topics

- Language Diversity and Challenges
- Noisy Channel Models for SMT
 - **Translation Model (Word Alignment)**
 - IBM Alignment Models
 - **EM Algorithm**
 - Language Model
 - Decoding

Expectation Maximization (EM) Algorithm

- Optimization algorithm for generative models
- Works for problems that have this format:
 - *If I had $P(a)$, it would be so easy to compute $P(b)$*
 - *If I had $P(b)$, it would be so easy to compute $P(a)$*
 - But, alas, I have neither. All I have is some observed data.
- Basic idea:
 - Randomly guess what $P(a)$ is
 - Use it to compute $P(b)$
 - Then, with your newly computed $P(b)$, recompute $P(a)$
 - Iterate until convergence

Expectation Maximization (EM) Algorithm

- E step: Estimate the likelihood of the observed data given the current parameters
- M step: Recompute the parameters so as to maximize the likelihood of the observed data

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

green house

the house

casa verde

la casa

t		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

Corpus

green house	casa verde
the house	la casa

t		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

green house

the house

casa verde

la casa

Parameters

		t		
		target		
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

All possible alignments

green house

the house

casa verde

la casa

a

green house
| |
casa verde

green house
 X
casa verde

the house
| |
la casa

the house
 X
la casa

t		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(A, S \mid T) = \prod_{j=1}^L t(s_j \mid t_{a_j})$$

green house

the house

casa verde

la casa

a

green house

casa verde

green house

casa verde

the house

la casa

the house

la casa

t		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(A, S \mid T) = \prod_{j=1}^L t(s_j \mid t_{a_j})$$

green house
the house

casa verde
la casa

a

green house
casa verde

green house
casa verde

the house
la casa

the house
la casa

$$\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(A, S \mid T) = \prod_{j=1}^L t(s_j \mid t_{a_j})$$

green house
the house

casa verde
la casa

a

green house
| |
casa verde

$$\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

green house
X
casa verde

$$\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

the house
| |
la casa

$$\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

the house
X
la casa

$$\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(a \mid s, t) = \frac{P(a, s \mid t)}{\sum_a P(a, s \mid t)}$$

green house
the house

casa verde
la casa

a

green house

casa verde

=1/2

green house

casa verde

=1/2

the house

la casa

=1/2

the house

la casa

=1/2

		target		
		green	house	the
source	casa	1/3	1/3	1/3
	la	1/3	1/3	1/3
	verde	1/3	1/3	1/3

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

E Step: Compute Likelihood of corpus given current parameters

Compute expected counts counts by adding fractional counts equal to $P(a|f,e)$.

green house
the house

casa verde
la casa

a

green house

casa verde

=1/2

green house

casa verde

=1/2

the house

la casa

=1/2

the house

la casa

=1/2

		target		
		green	house	the
source	casa	1/2	1	1/2
	la	0	1/2	1/2
	verde	1/2	1/2	0

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

E Step: Compute Likelihood of corpus given current parameters

Compute expected counts counts by adding fractional counts equal to $P(a|f,e)$.

green house
the house

casa verde
la casa

a

green house

blue

green

casa verde

=1/2

green house

orange

green

casa verde

=1/2

the house

green

house

la casa

=1/2

the house

orange

house

la casa

=1/2

		target		
		green	house	the
source	casa	1/2	1	1/2
	la	0	1/2	1/2
	verde	1/2	1/2	0

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{aj})$$

M Step: Compute MLE parameters

(here, that just means normalizing!)

green house

the house

casa verde

la casa

a

green house

casa verde

green house

casa verde

the house

la casa

the house

la casa

		target		
		green	house	the
source	casa	1/2	1/2	1/2
	la	0	1/4	1/2
	verde	1/2	1/4	0

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(A, S \mid T) = \prod_{j=1}^L t(s_j \mid t_{a_j})$$

green house
the house

casa verde
la casa

a

<div><div>green house</div><div><div></div><div></div></div><div>casa verde</div></div> <div>$\frac{1}{2} \times \frac{1}{4}$ $= \frac{1}{8}$</div>	<div><div>green house</div><div><div></div><div></div></div><div>casa verde</div></div> <div>$\frac{1}{2} \times \frac{1}{2}$ $= \frac{1}{4}$</div>	<div><div>the house</div><div><div></div><div></div></div><div>la casa</div></div> <div>$\frac{1}{2} \times \frac{1}{2}$ $= \frac{1}{4}$</div>	<div><div>the house</div><div><div></div><div></div></div><div>la casa</div></div> <div>$\frac{1}{2} \times \frac{1}{4}$ $= \frac{1}{8}$</div>
---	---	--	--

		target		
		green	house	the
source	casa	1/2	1/2	1/2
	la	0	1/4	1/2
	verde	1/2	1/4	0

Expectation Maximization (EM) Algorithm

Goal:

Estimate a and t using

$$\operatorname{argmax}_{a_j} (s_j \mid t_{a_j})$$

E Step: Compute Likelihood of corpus given current parameters

$$P(A, S \mid T) = \prod_{j=1}^L t(s_j \mid t_{a_j})$$

green house

the house

casa verde

la casa

a

green house

casa verde

1/2 x 1/4 = 1/8

green house

1/2 x 1/2 = 1/4

the house

la casa

1/2 x 1/2 = 1/4

the house

1/2 x 1/4 = 1/8

		target		
		green	house	the
source	casa	1/2	1/2	1/2
	la	0	1/4	1/2
	verde	1/2	1/4	0



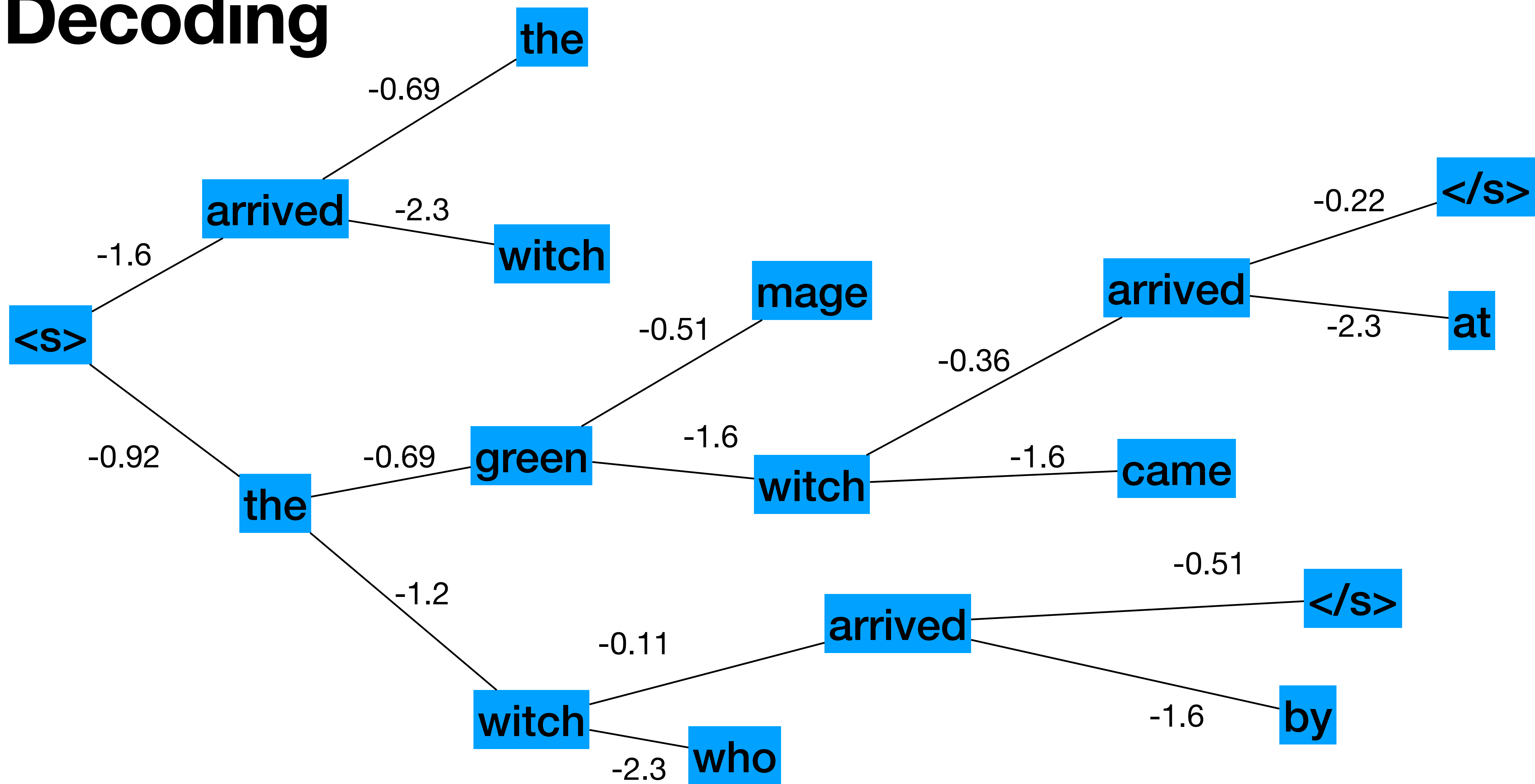
Topics

- Language Diversity and Challenges
- Noisy Channel Models for SMT
 - Translation Model (Word Alignment)
 - IBM Alignment Models
 - EM Algorithm
 - **Language Model**
 - **Decoding**

Decoding

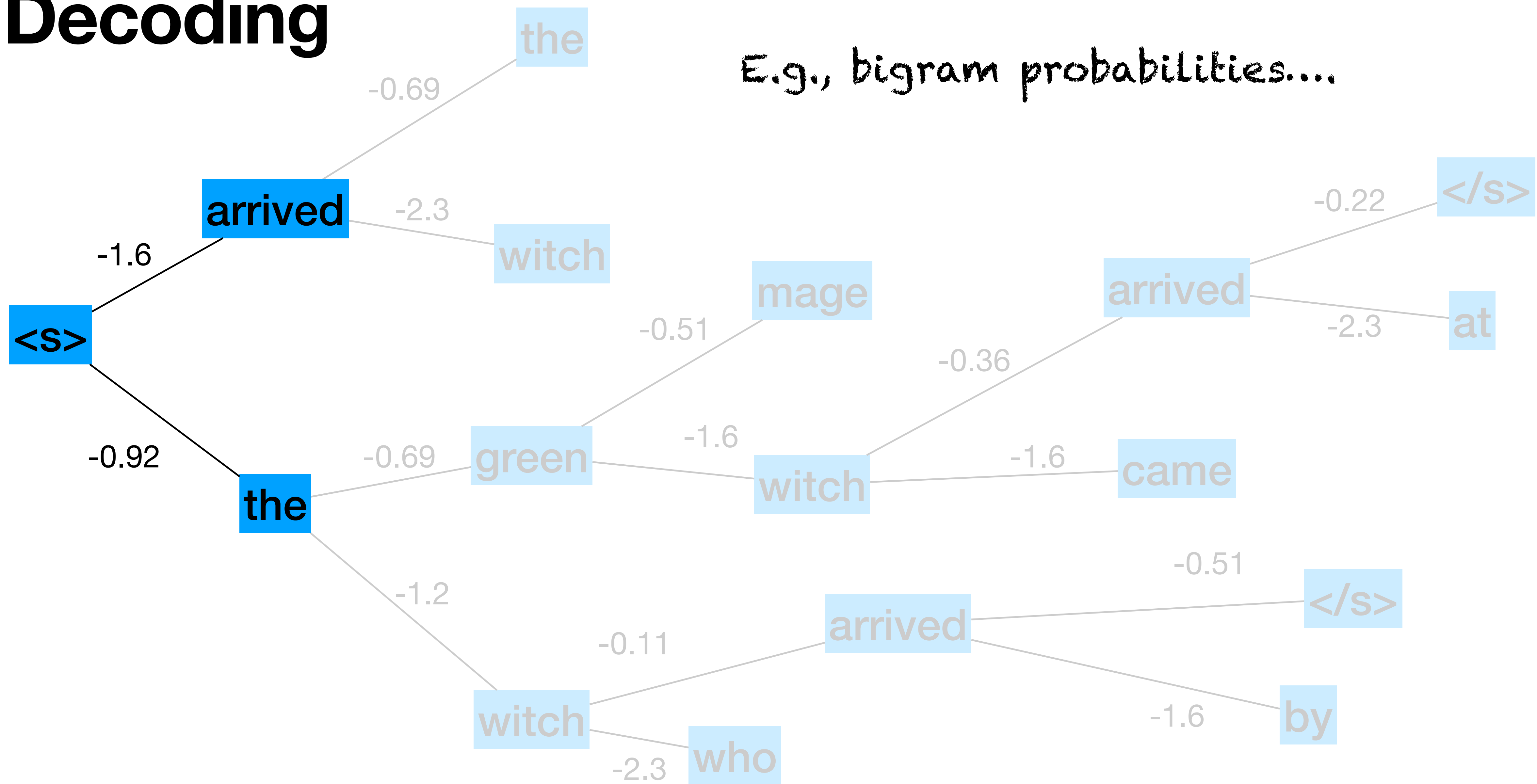
- Finding best translation (e.g., arg-maxing $P(s|t)P(t)$) is just a search problem!
- Typically, variants of A^* search
 - Often called “stack decoding” in NLP
- Two types of algorithms:
 - Greedy Decoding (for intuition, not used)
 - Beam Search (more common)

Decoding



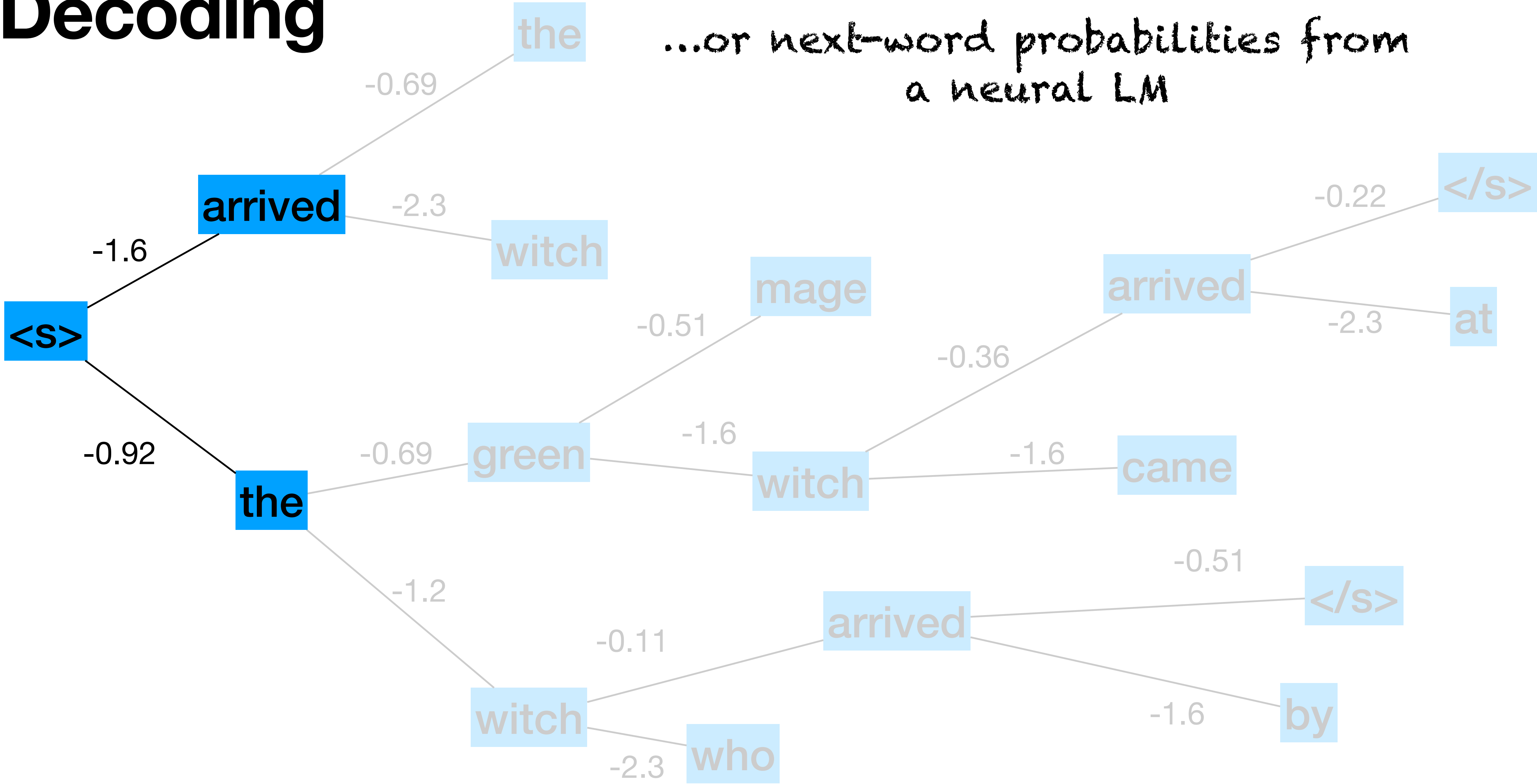
Decoding

E.g., bigram probabilities....



Decoding

...or next-word probabilities from
a neural LM



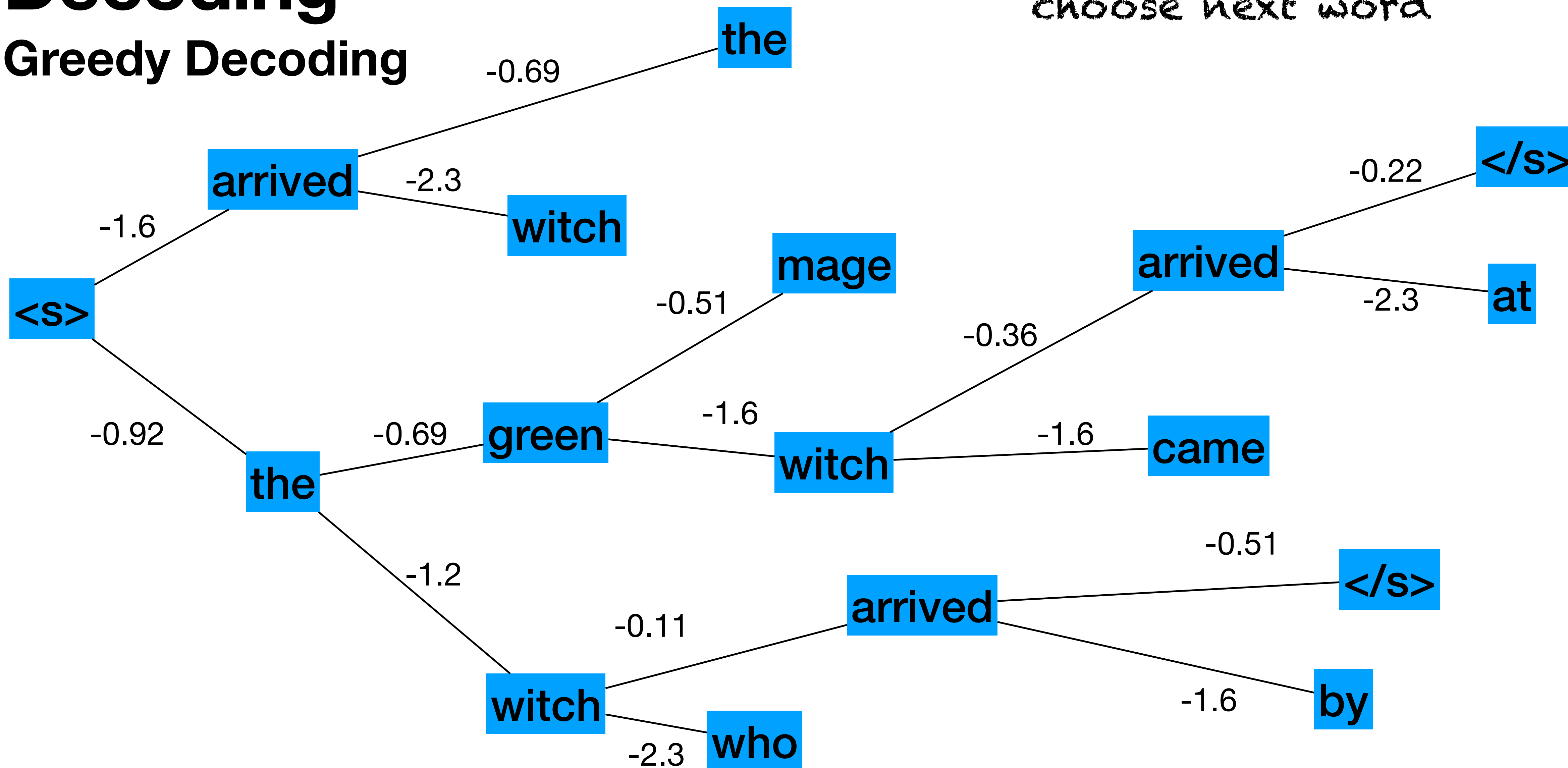
Decoding

- Finding best translation (e.g., arg-maxing $P(s|t)P(t)$) is just a search problem!
- Typically, variants of A^* search
 - Often called “stack decoding” in NLP
- Two types of algorithms:
 - **Greedy Decoding (for intuition, not used)**
 - Beam Search (more common)

Decoding

Greedy Decoding

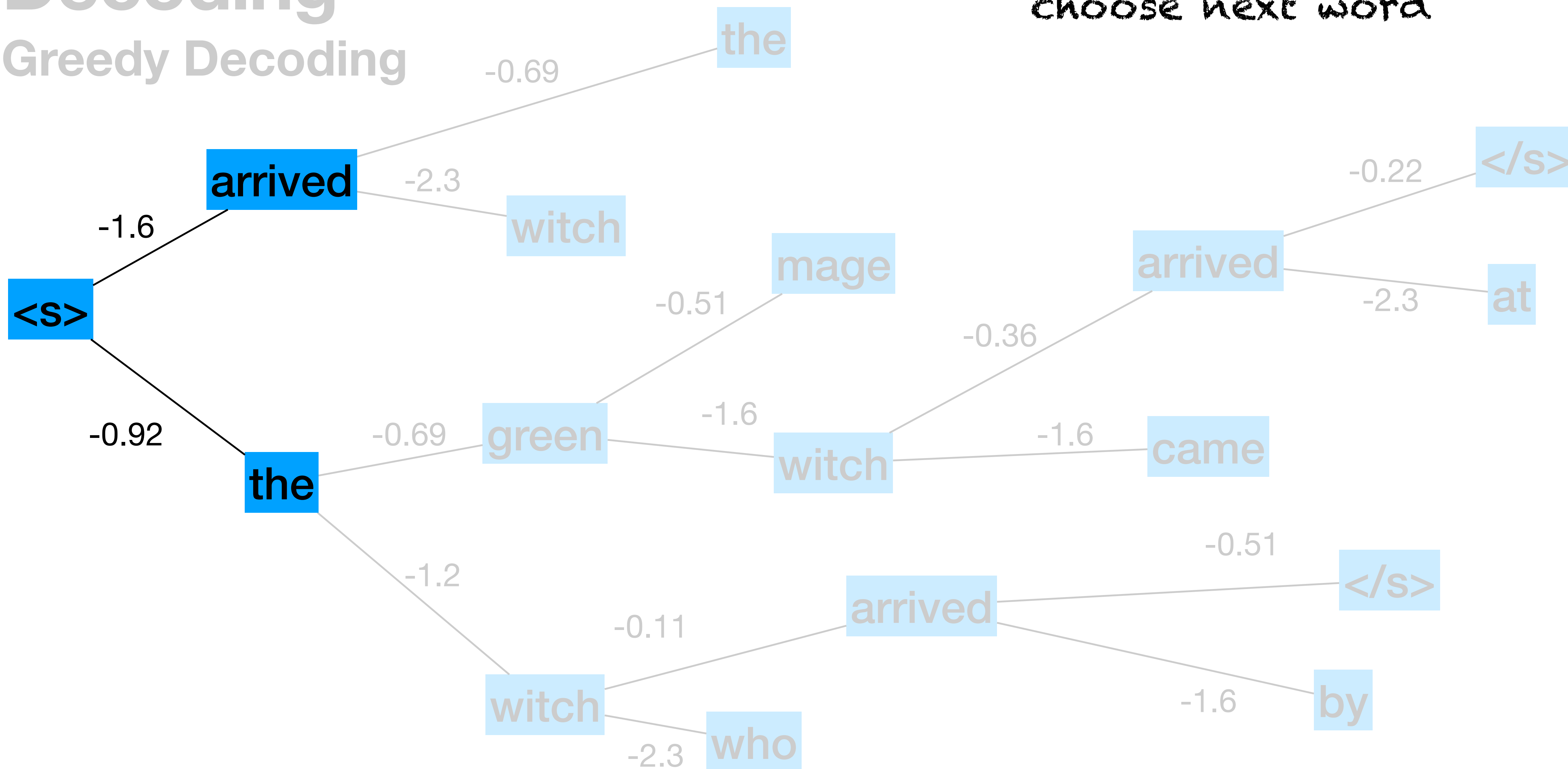
Sample proportionally to
choose next word



Decoding

Greedy Decoding

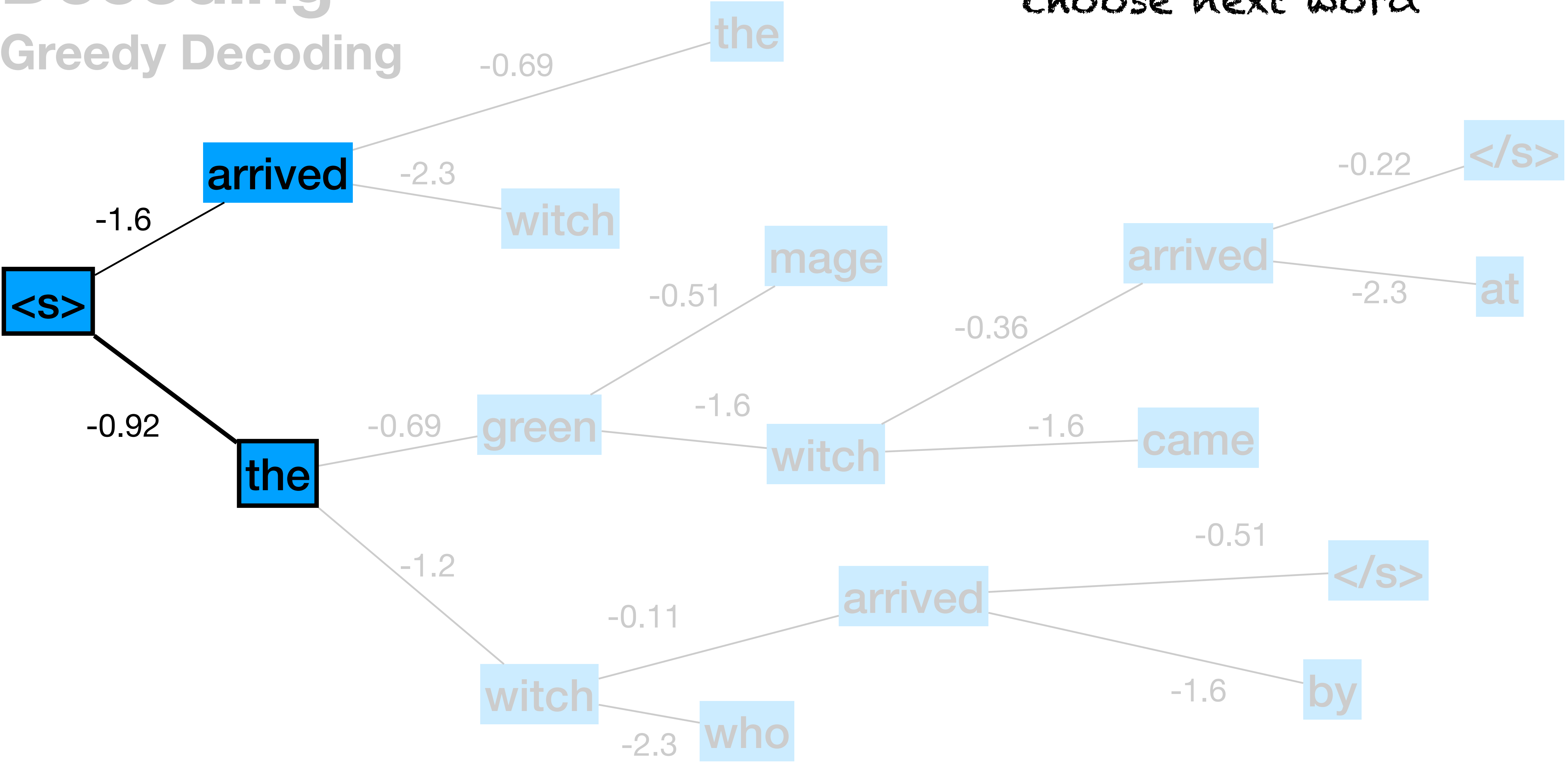
Sample proportionally to
choose next word



Decoding

Greedy Decoding

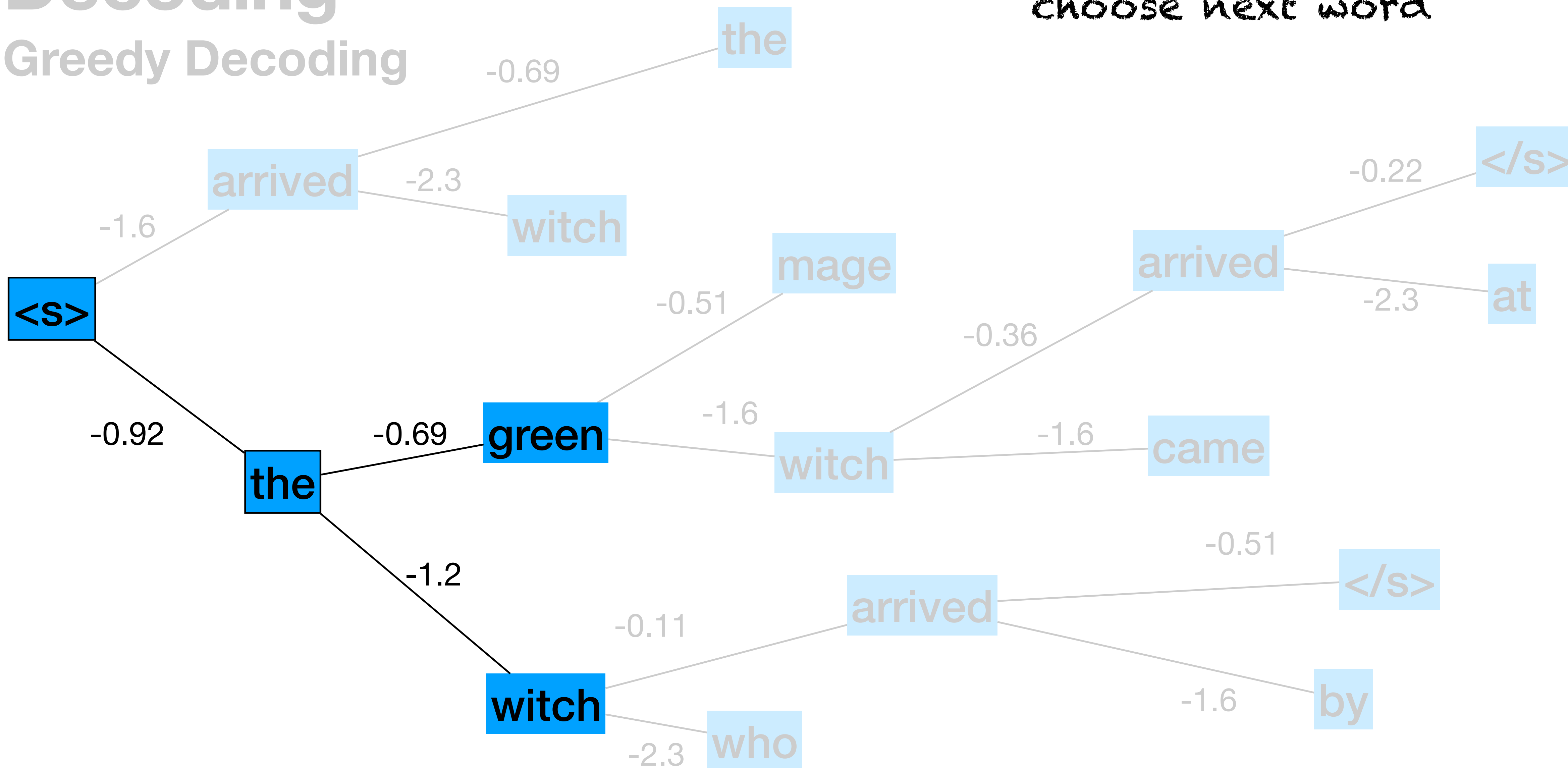
Sample proportionally to
choose next word



Decoding

Greedy Decoding

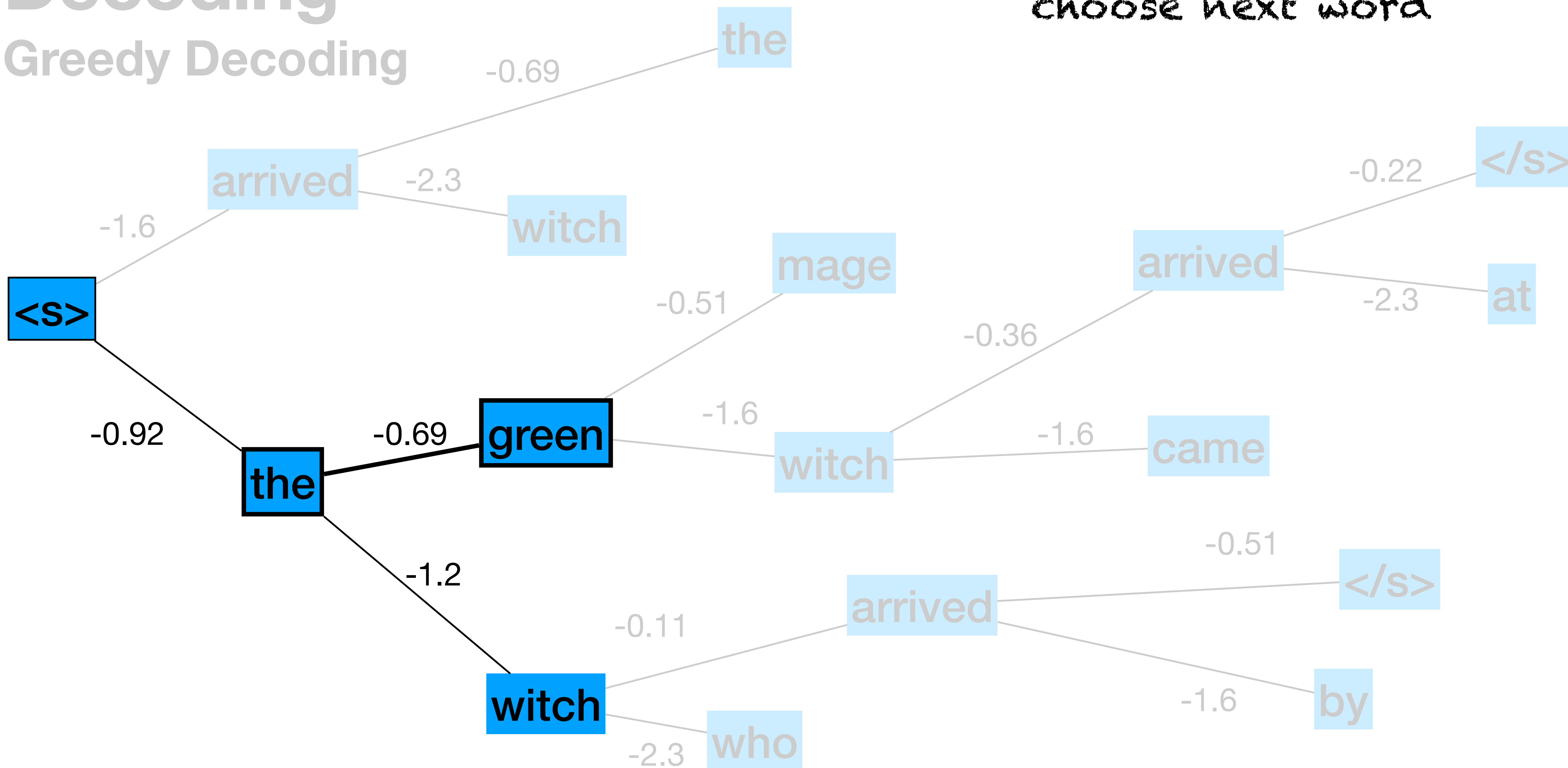
Sample proportionally to
choose next word



Decoding

Greedy Decoding

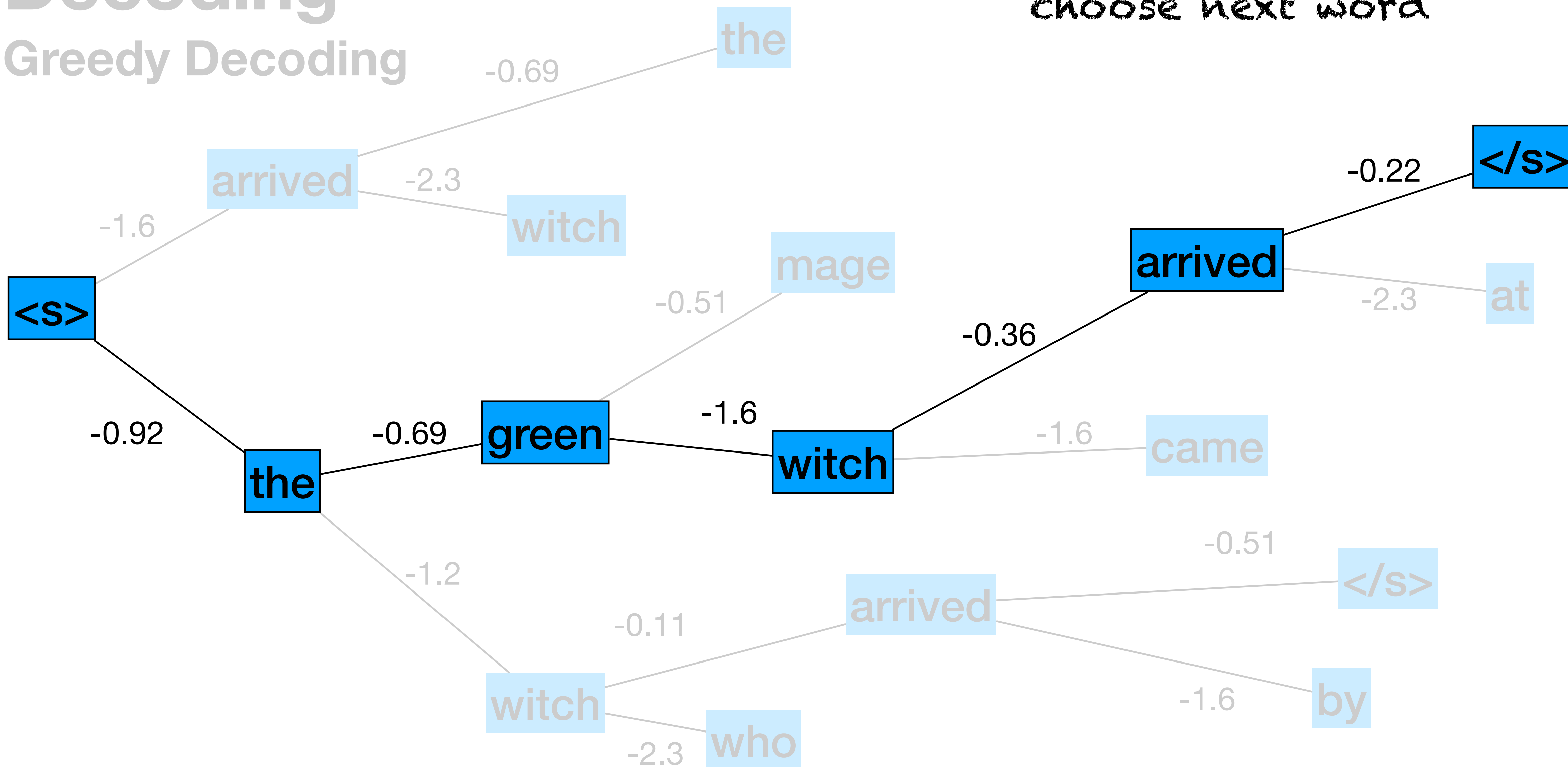
Sample proportionally to
choose next word



Decoding

Greedy Decoding

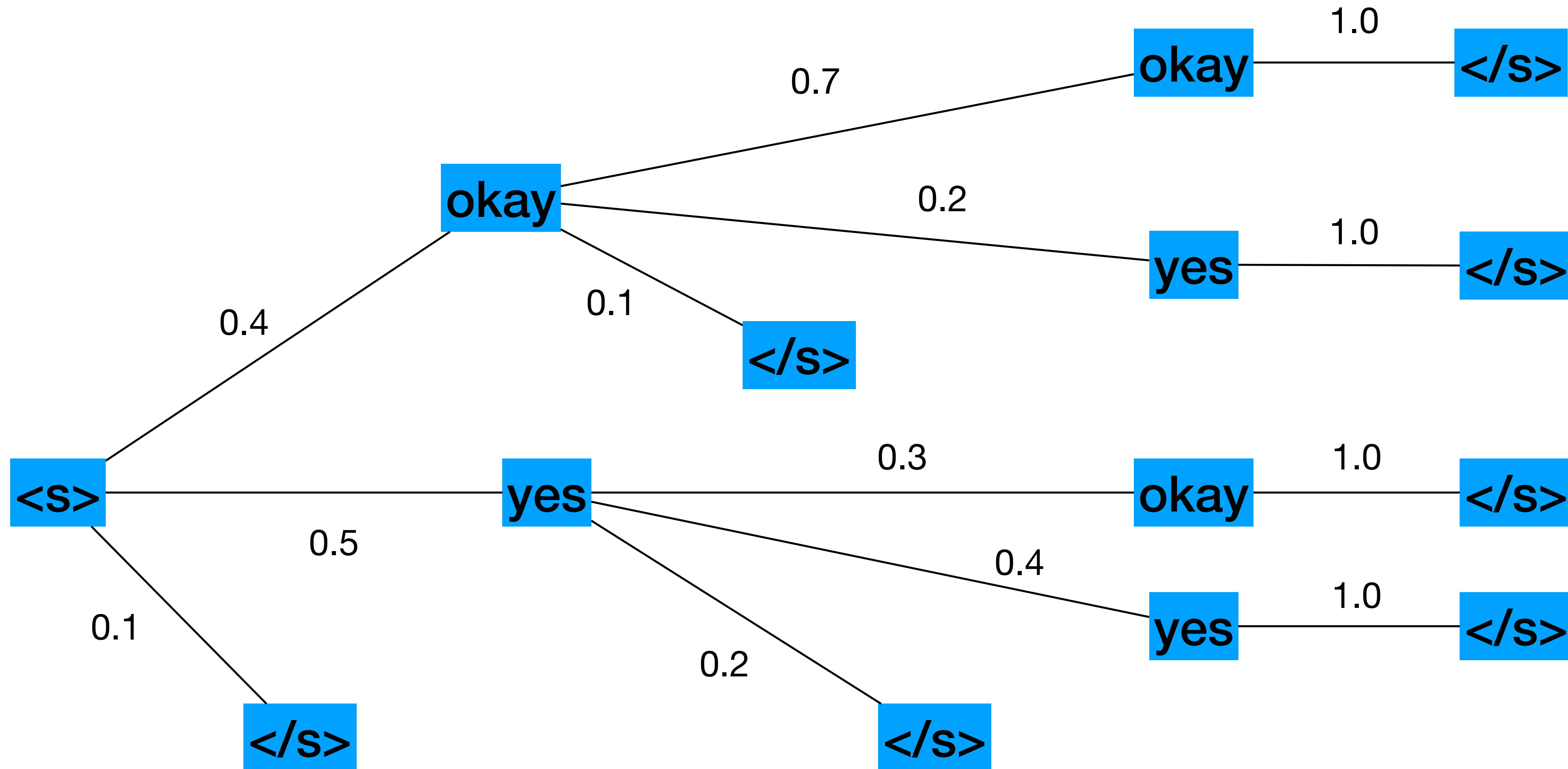
Sample proportionally to
choose next word



Decoding

Greedy Decoding

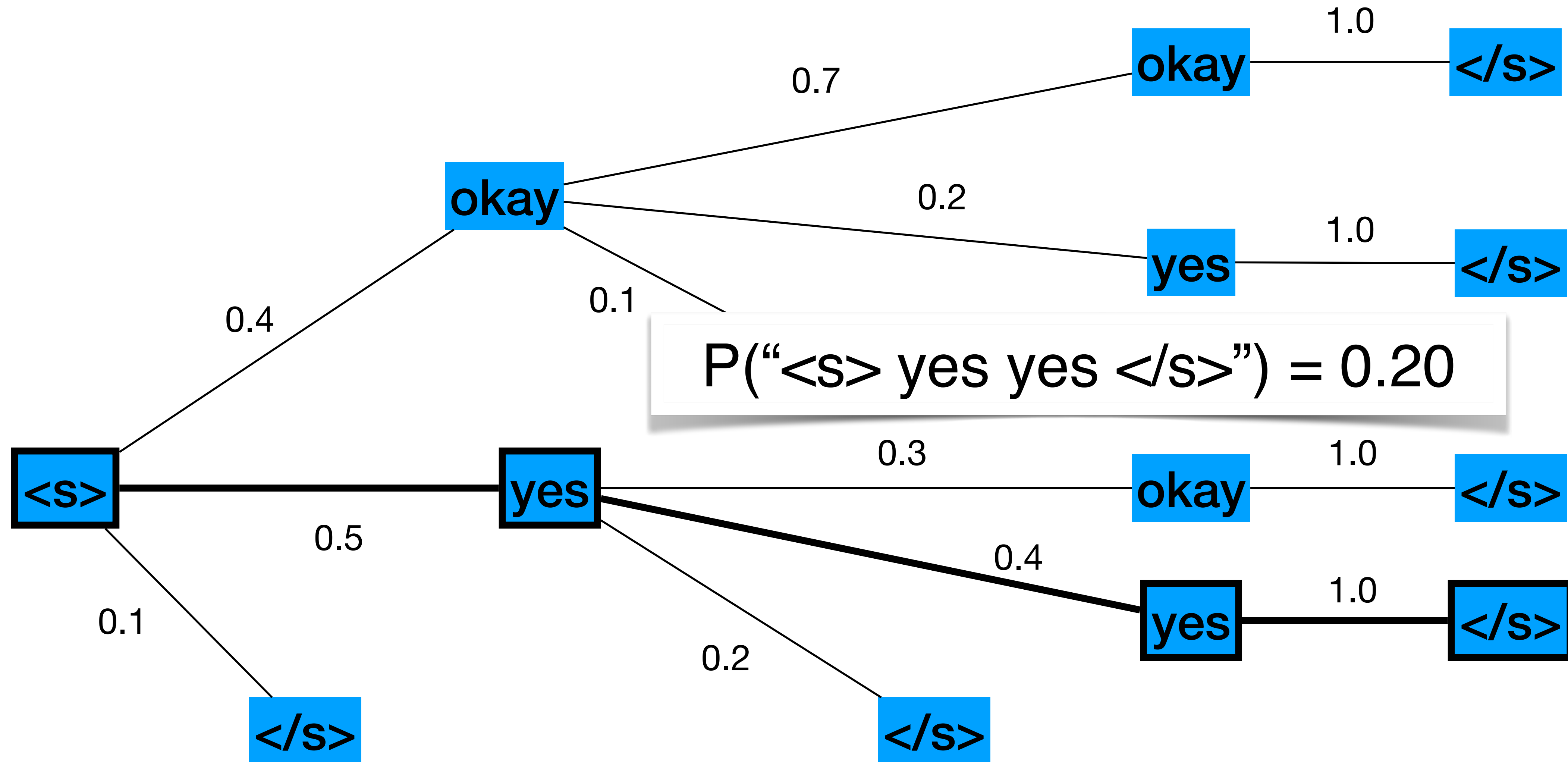
Often finds suboptimal path



Decoding

Greedy Decoding

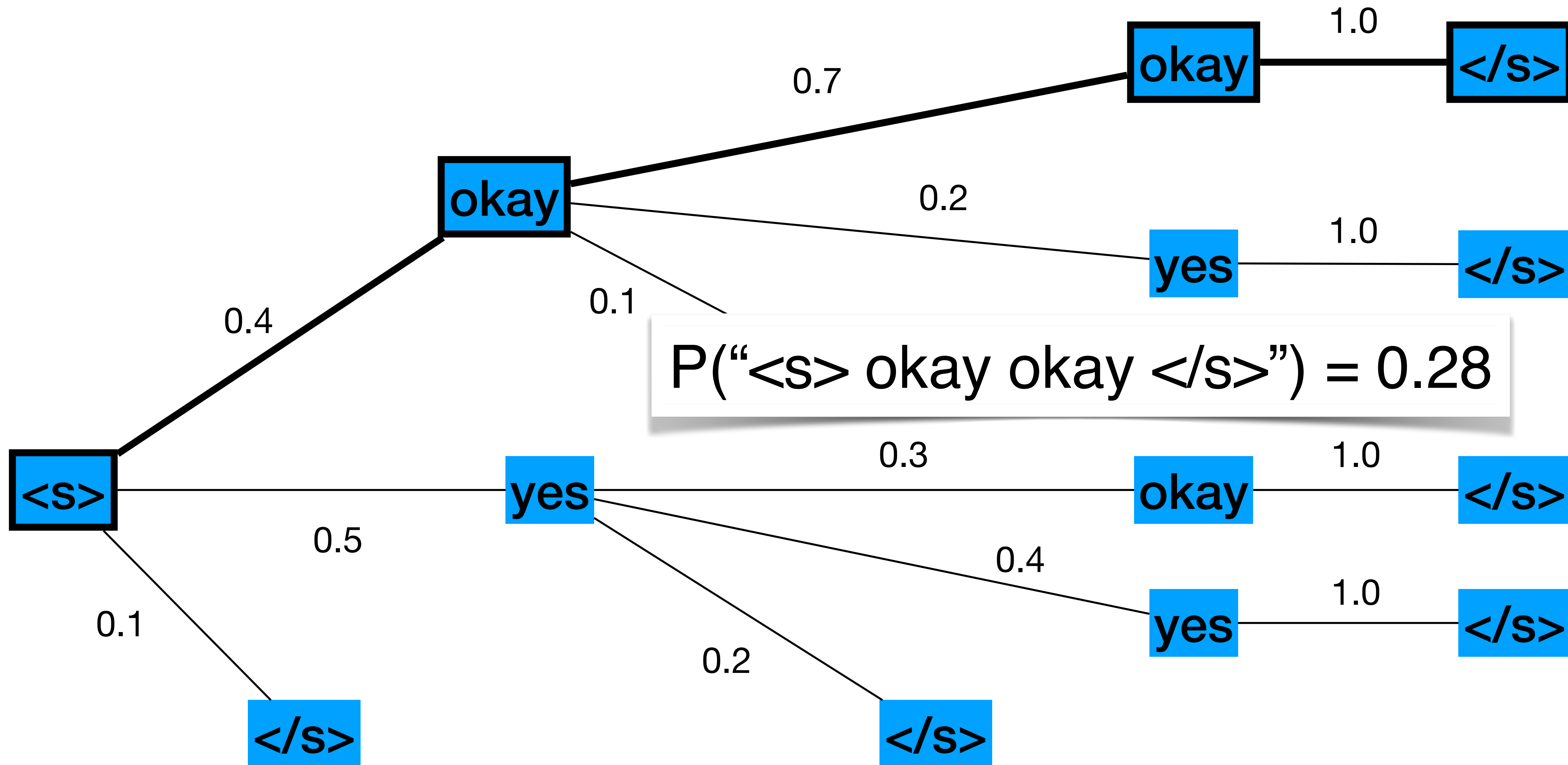
often finds suboptimal path



Decoding

Greedy Decoding

Often finds suboptimal path



Decoding

- Finding best translation (e.g., arg-maxing $P(s|t)P(t)$) is just a search problem!
- Typically, variants of A^* search
 - Often called “stack decoding” in NLP
- Two types of algorithms:
 - Greedy Decoding (for intuition, not used)
 - **Beam Search (more common)**

Decoding

Beam Search

- Problem: finding global most optimal sequence is too computationally hard, since there are V^n possible sequences
- Solution: consider top k at each time step (“beam size of k ”)

Decoding

Beam Search

- Algorithm:
 - At first time step, choose K most likely words. These are the initial “hypotheses”
 - At each following step, the top K best hypotheses are carried forward, (i.e., softmax over the whole vocab given each hypothesis, so $k * V$ computations)
 - When $\langle /s \rangle$ is generated, that generation is output
 - Continue until beam is empty

Decoding

Beam Search

Never more than k
hypotheses at a
given time

- Algorithm:
 - At first time step, choose K most likely words. These are the initial “hypotheses”
 - At each following step, the top K best hypotheses are carried forward, (i.e., softmax over the whole vocab given each hypothesis, so $k * V$ computations)
 - When $\langle /s \rangle$ is generated, that generation is output
 - Continue until beam is empty

Decoding

Beam Search

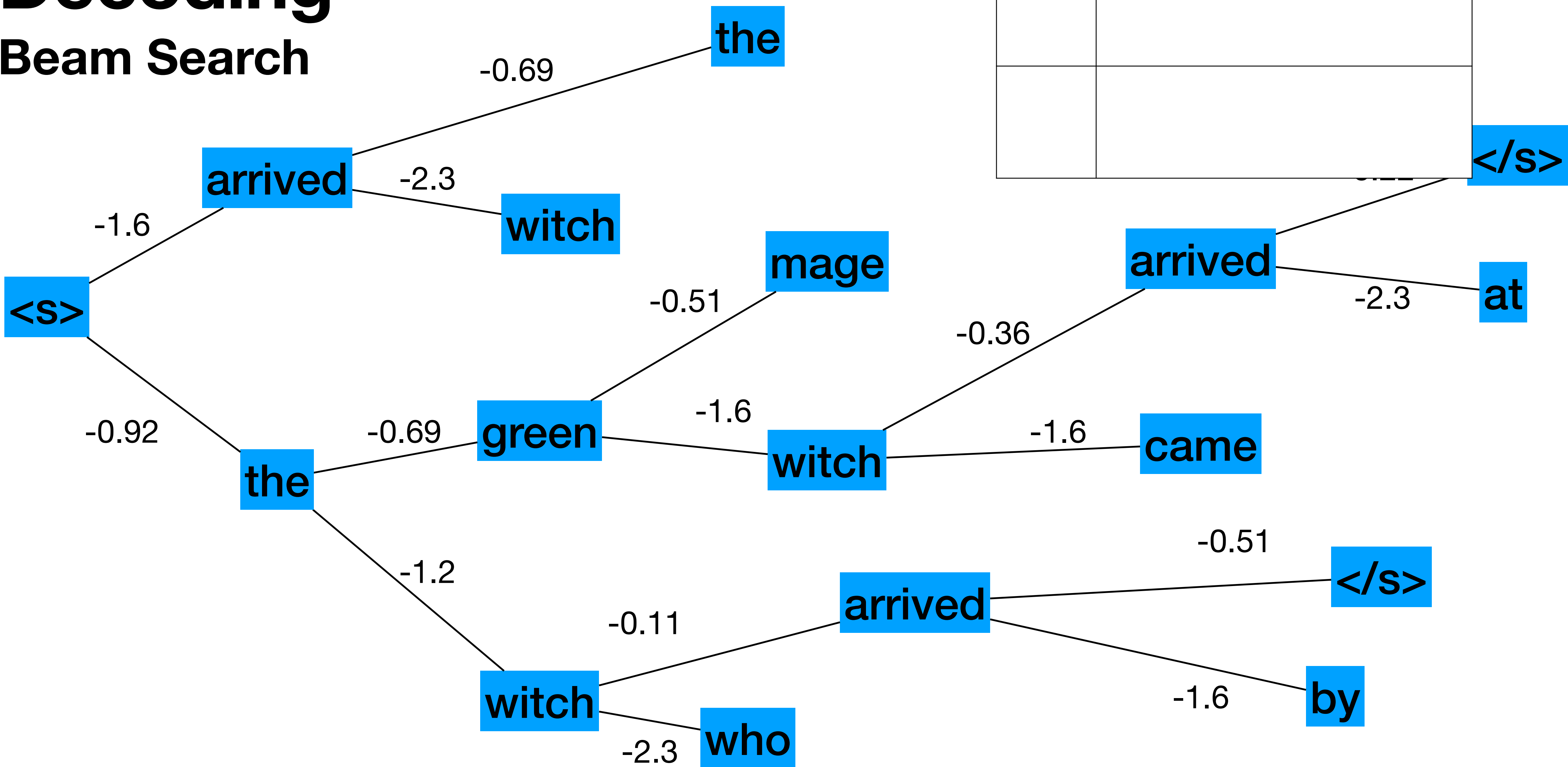
Final output is top K
candidates



- Algorithm:
 - At first time step, choose K most likely words. These are the initial “hypotheses”
 - At each following step, the top K best hypotheses are carried forward, (i.e., softmax over the whole vocab given each hypothesis, so $k * V$ computations)
 - When $\langle /s \rangle$ is generated, that generation is output
 - Continue until beam is empty

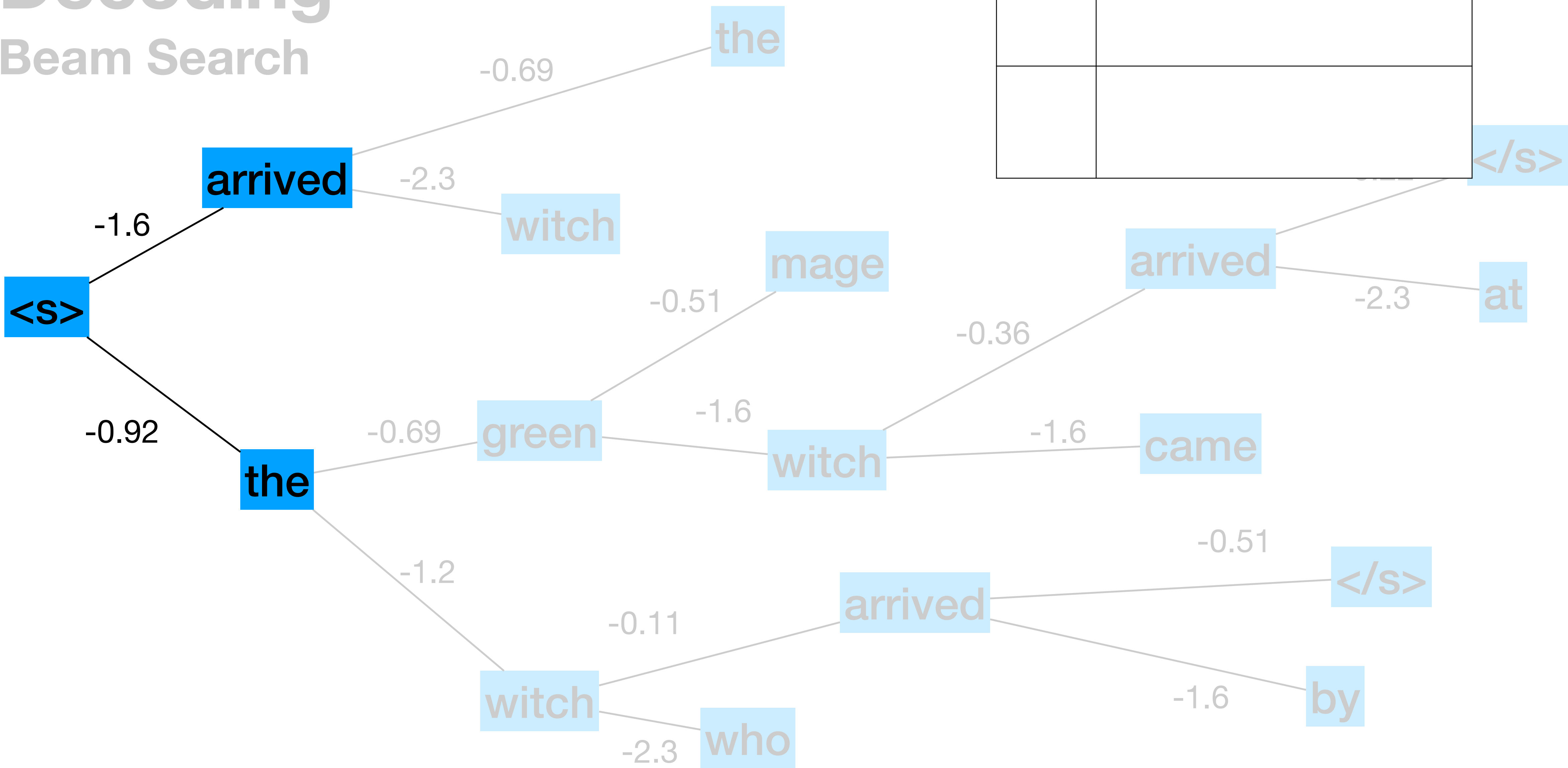
Decoding

Beam Search



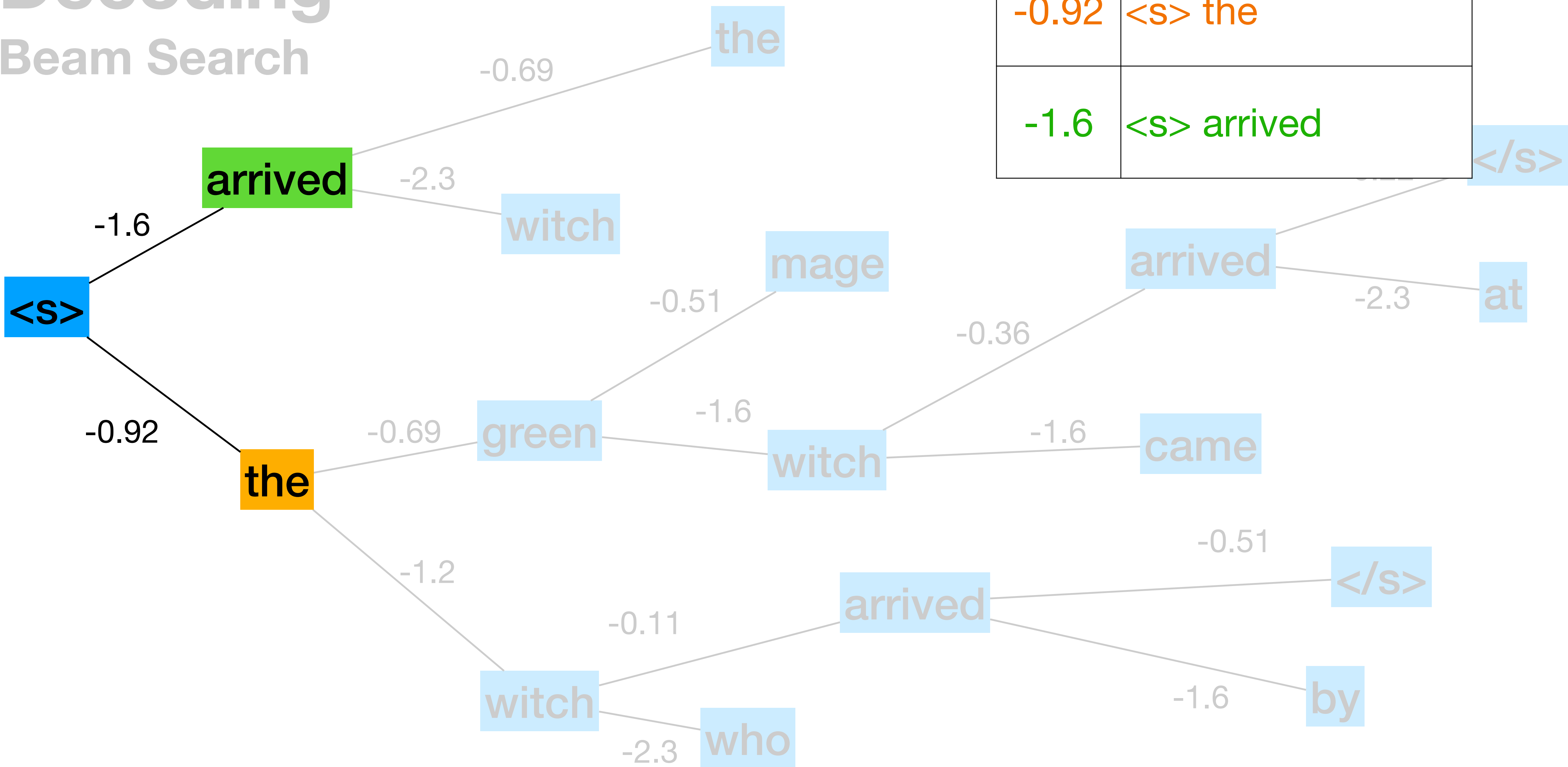
Decoding

Beam Search



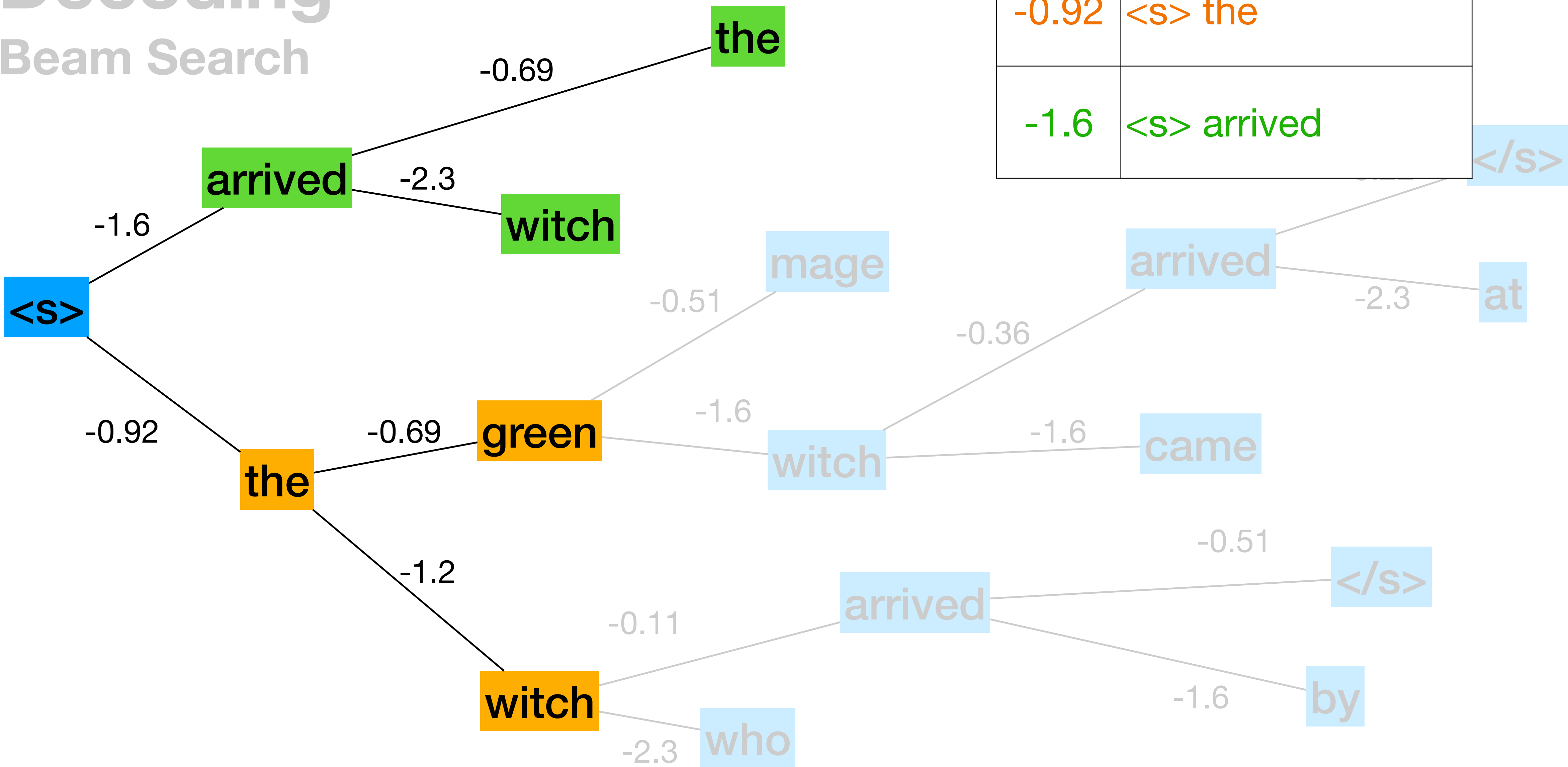
Decoding

Beam Search



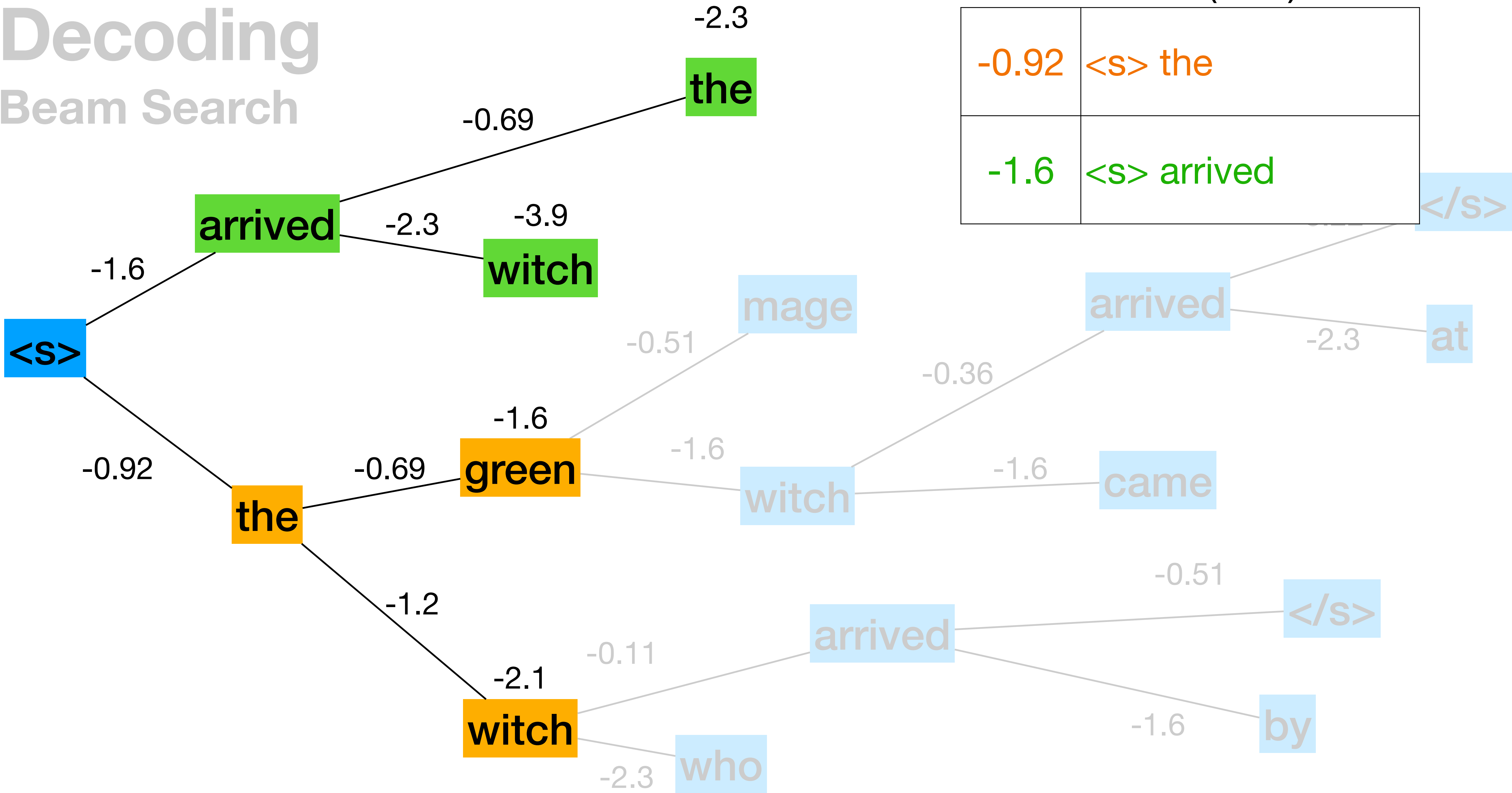
Decoding

Beam Search

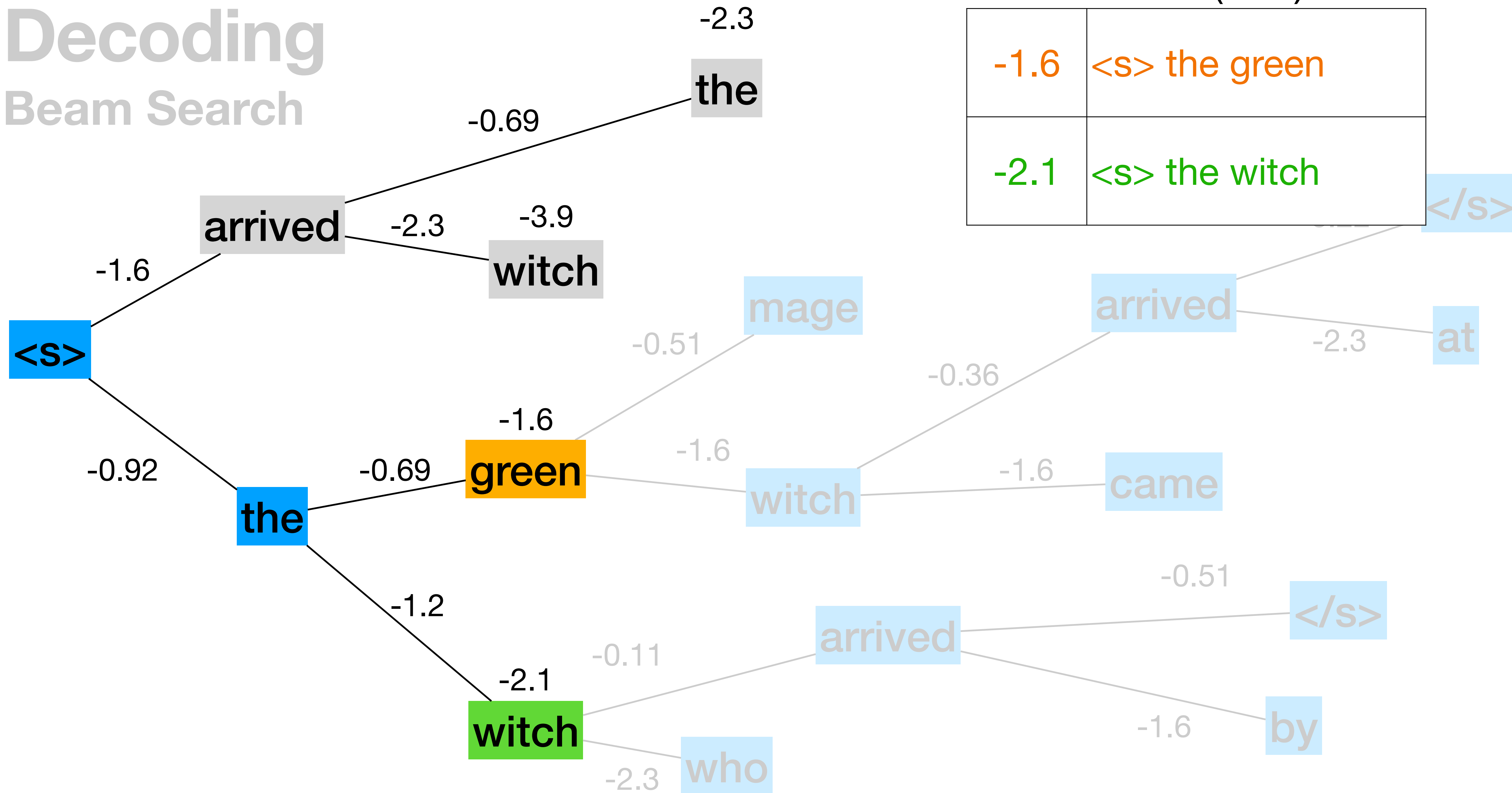


Decoding

Beam Search

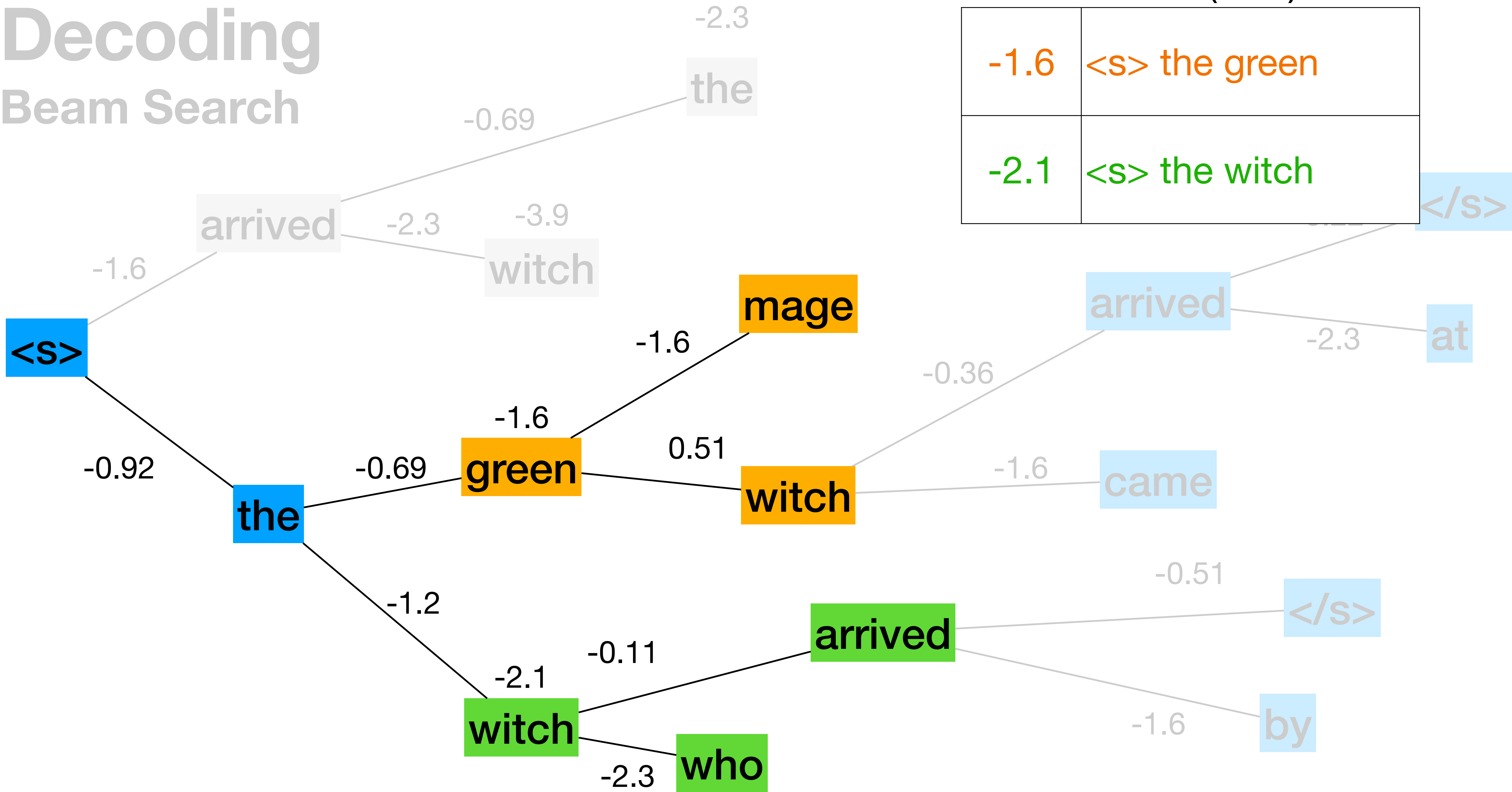


Decoding Beam Search



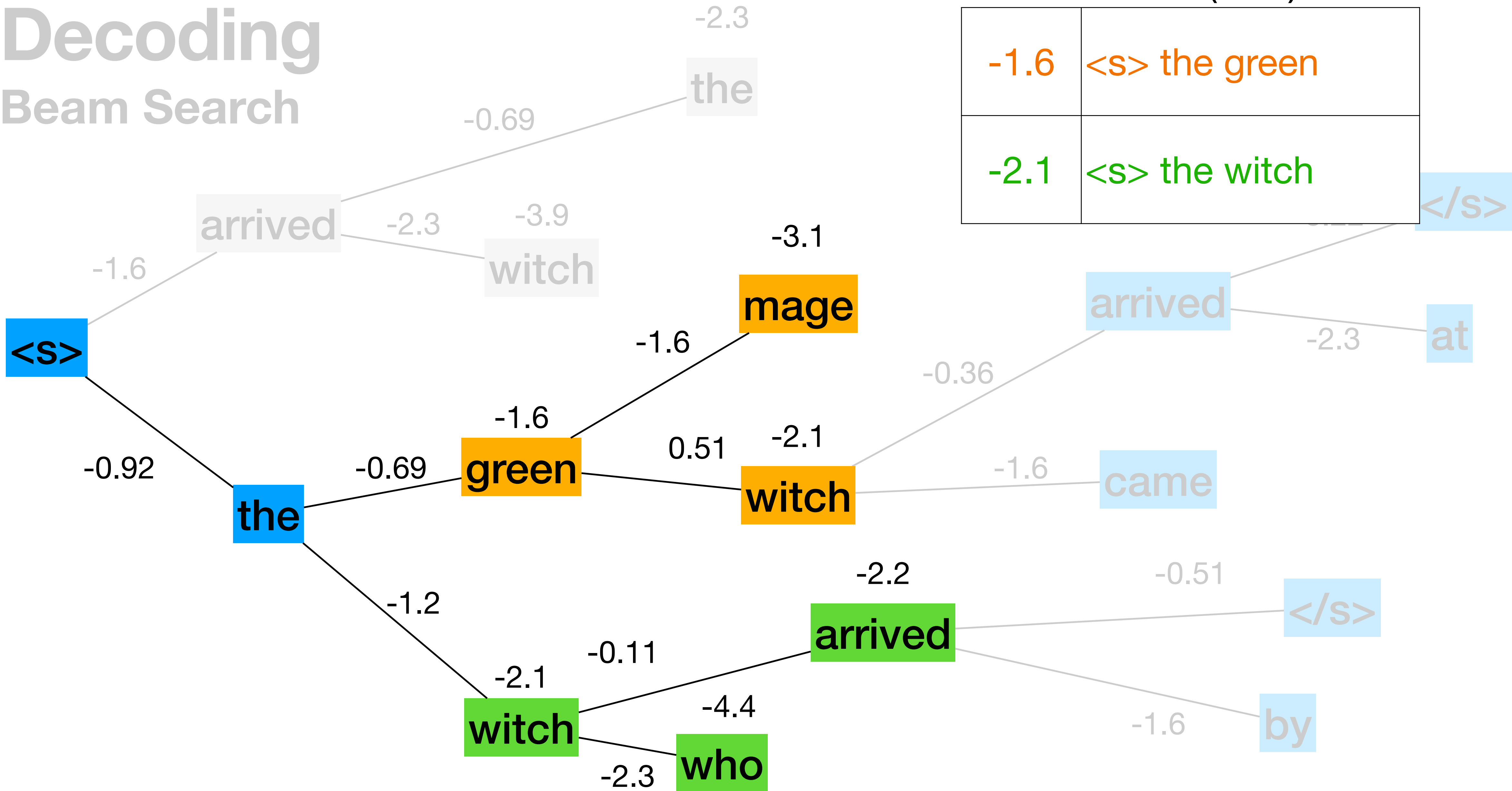
Decoding

Beam Search



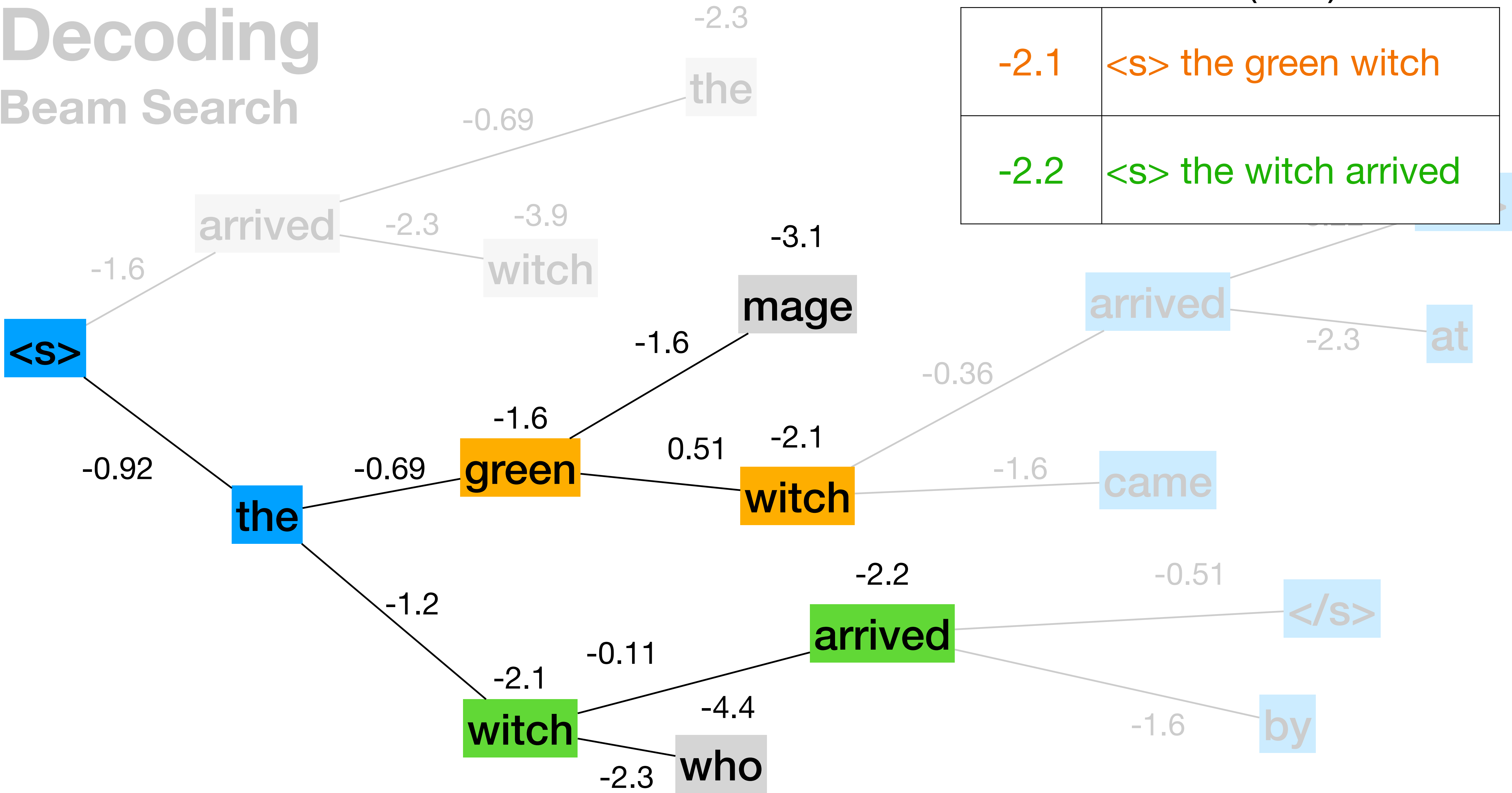
Decoding

Beam Search



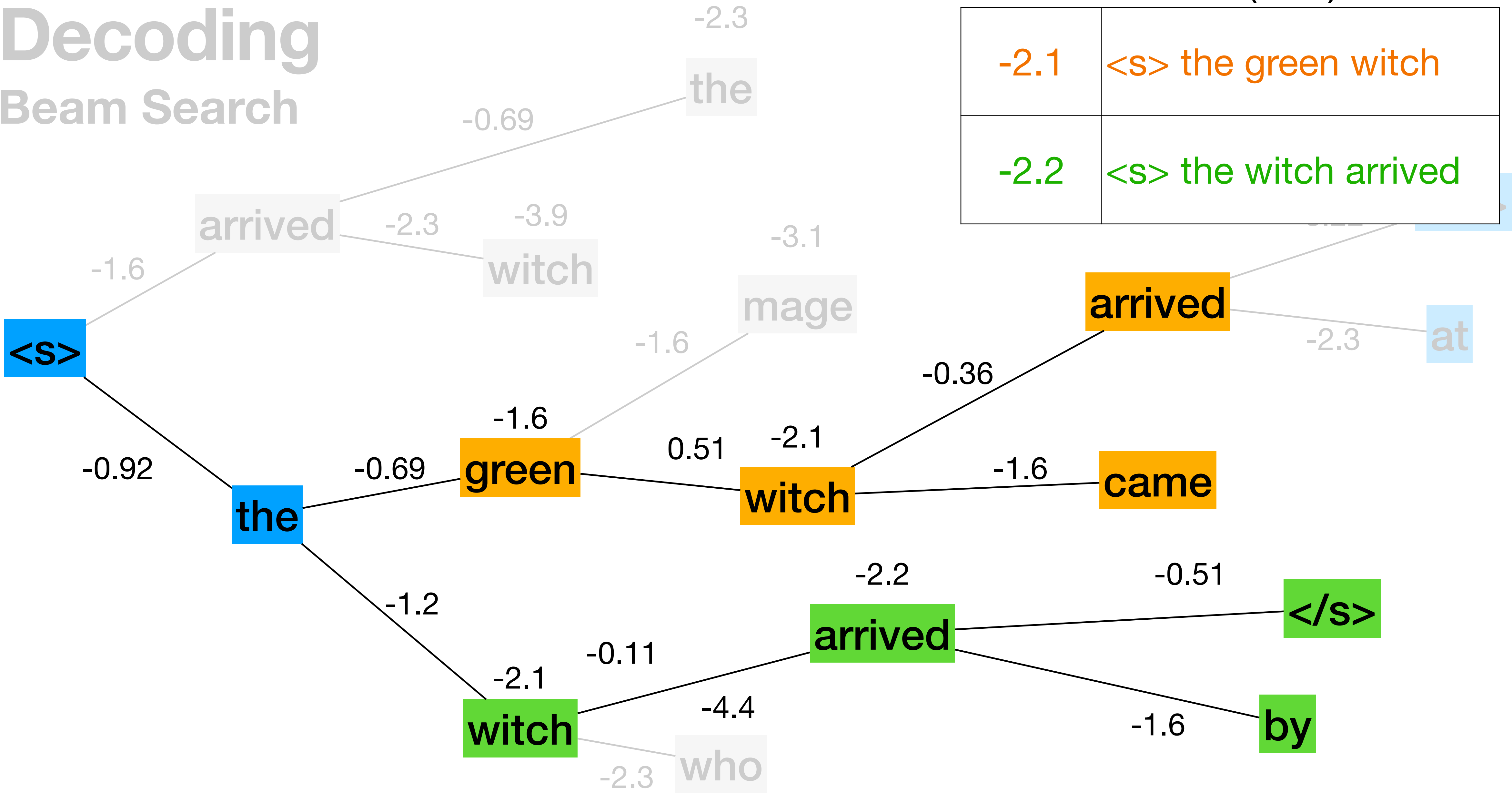
Decoding

Beam Search



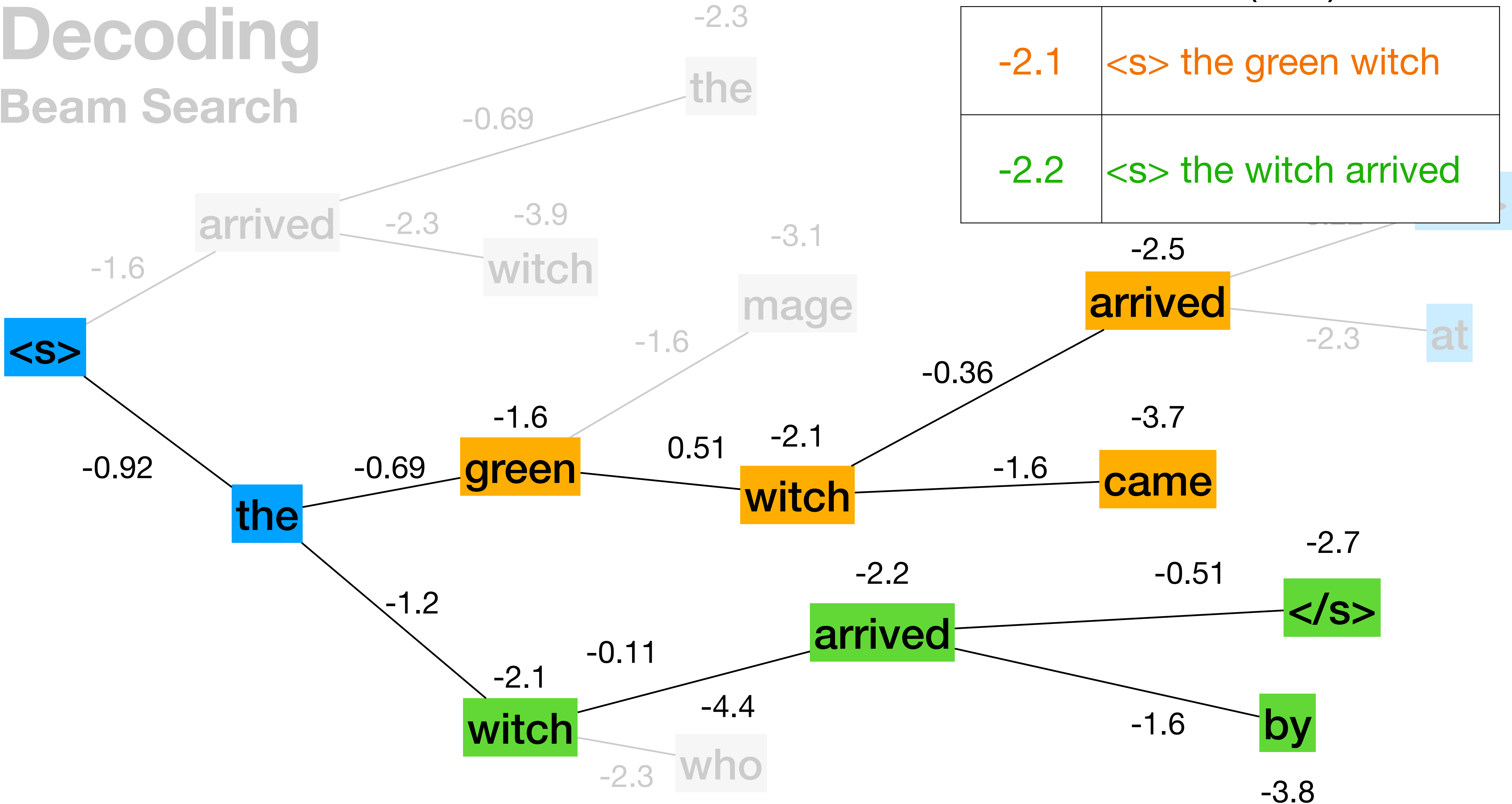
Decoding

Beam Search



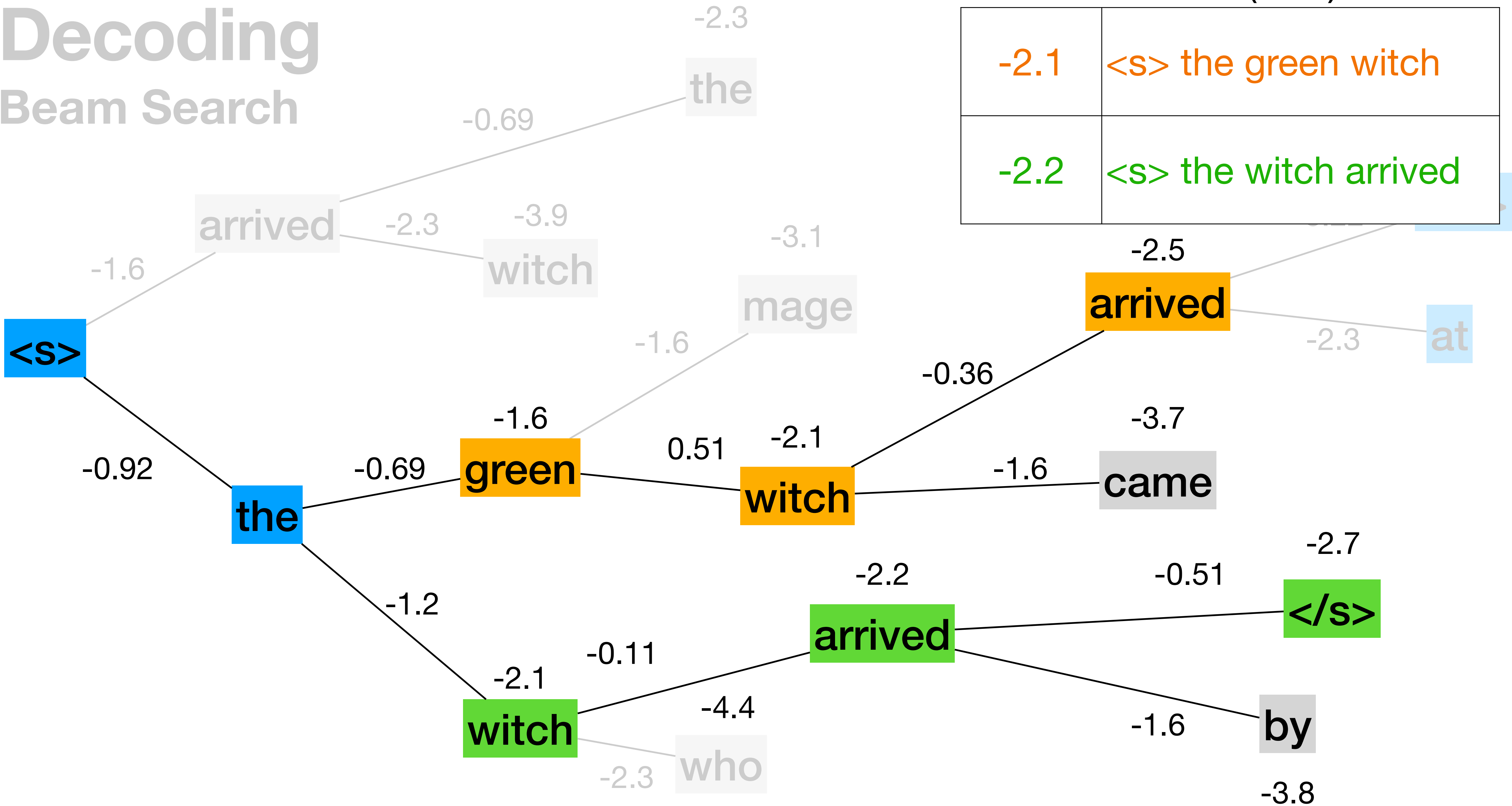
Decoding

Beam Search



Decoding

Beam Search

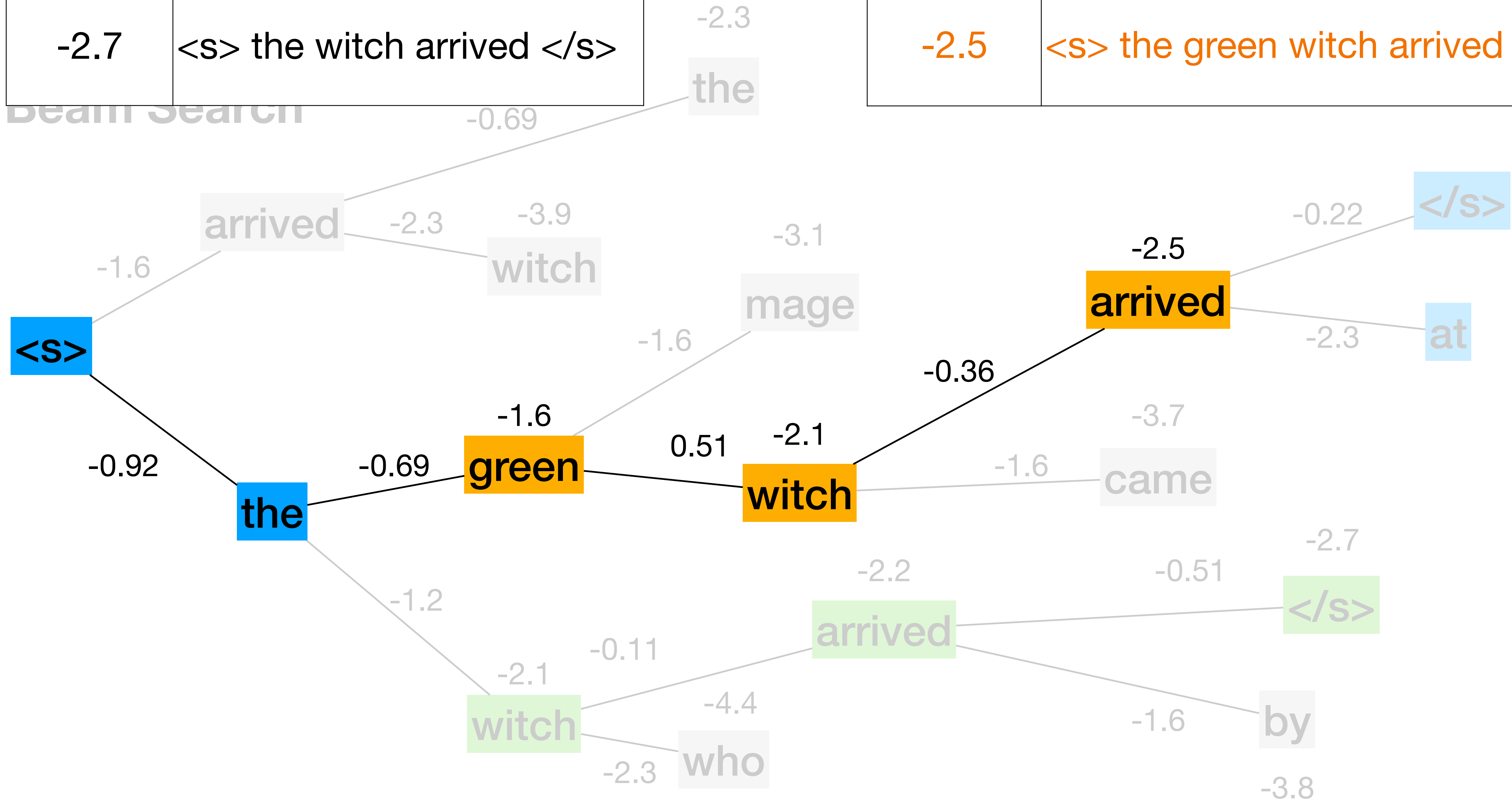


Output

Beam (k=1)

-2.7	<s> the witch arrived </s>
------	----------------------------

-2.5	<s> the green witch arrived
------	-----------------------------



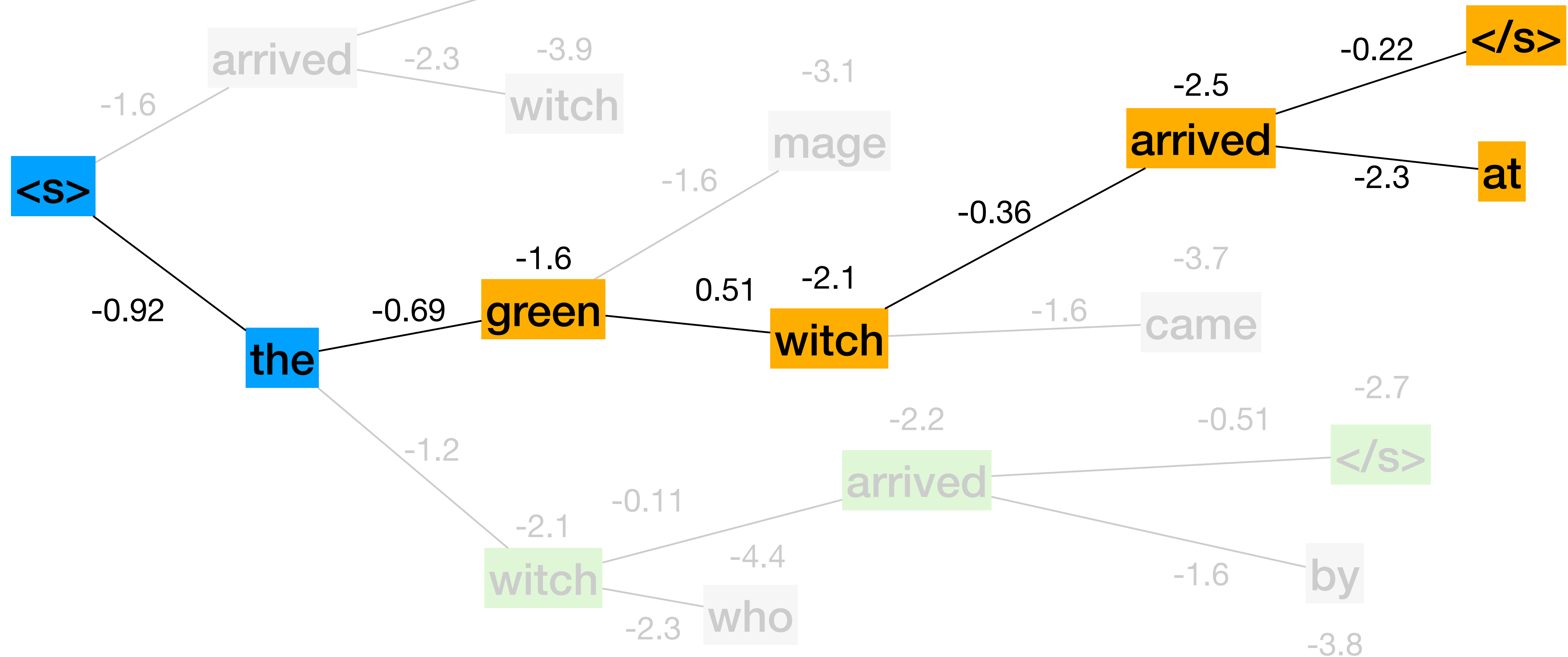
Output

Beam (k=1)

-2.7	<s> the witch arrived </s>
------	----------------------------

-2.5	<s> the green witch arrived
------	-----------------------------

Beam Search

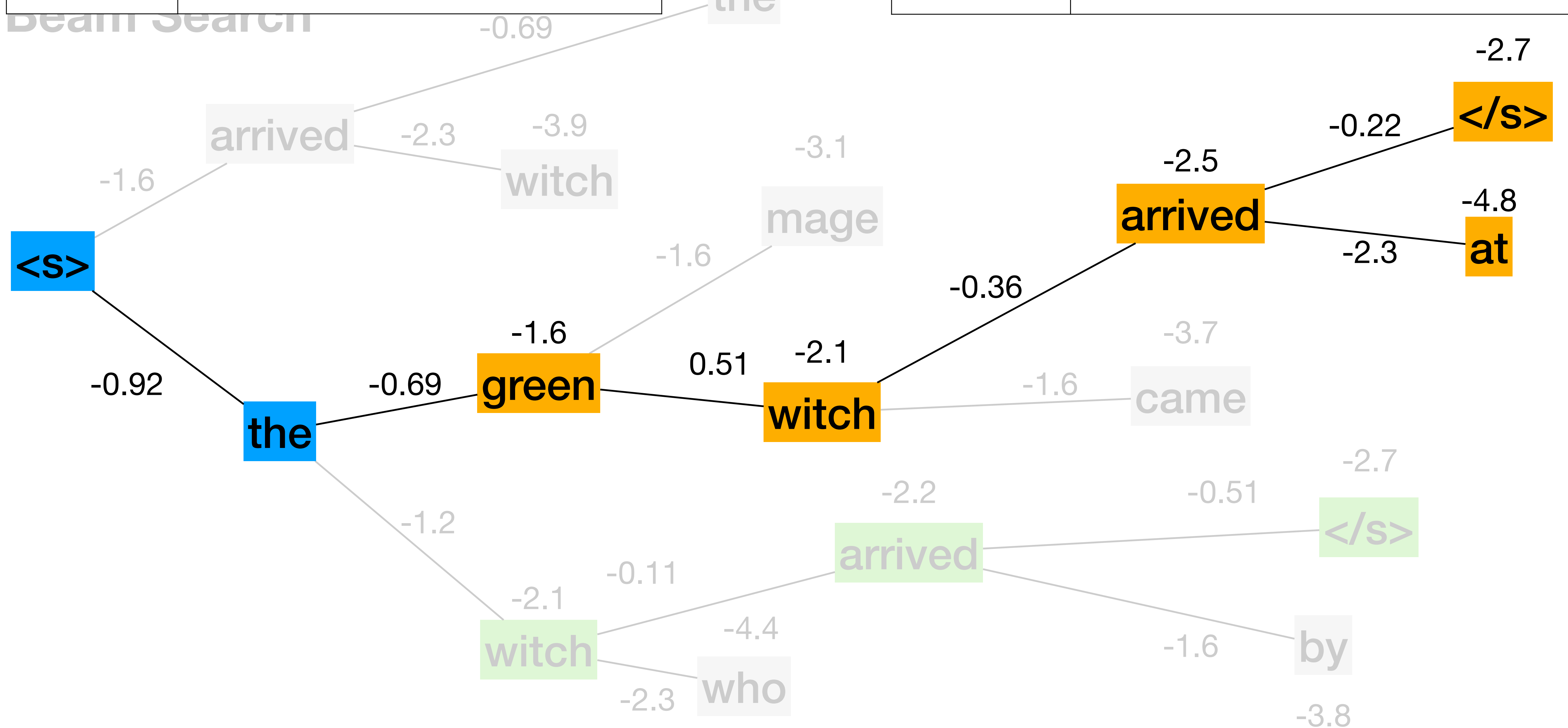


Output

Beam (k=1)

-2.7	<s> the witch arrived </s>
------	----------------------------

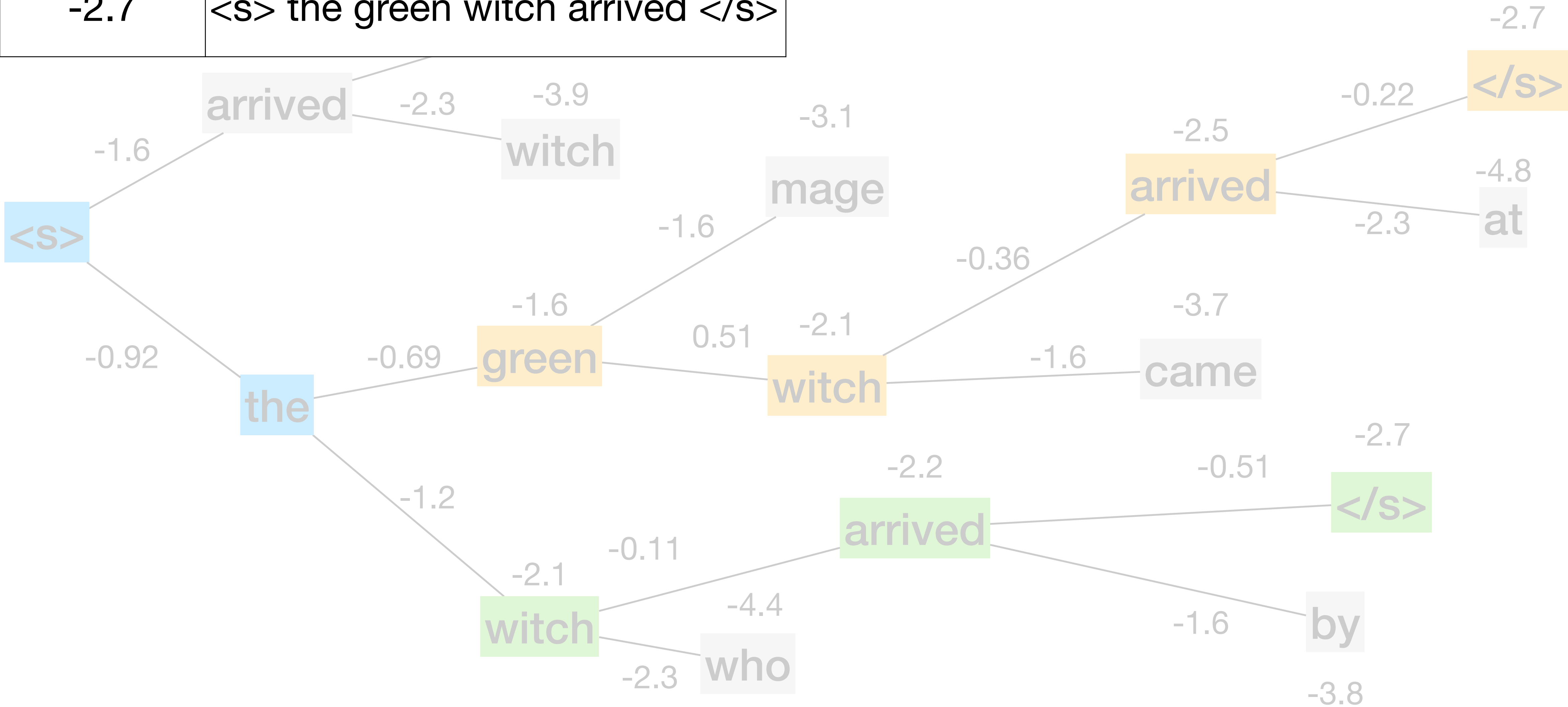
-2.5	<s> the green witch arrived
------	-----------------------------



Output

-2.7	<s> the witch arrived </s>
-2.7	<s> the green witch arrived </s>

Beam (k=0)





Decoding

Beam Search

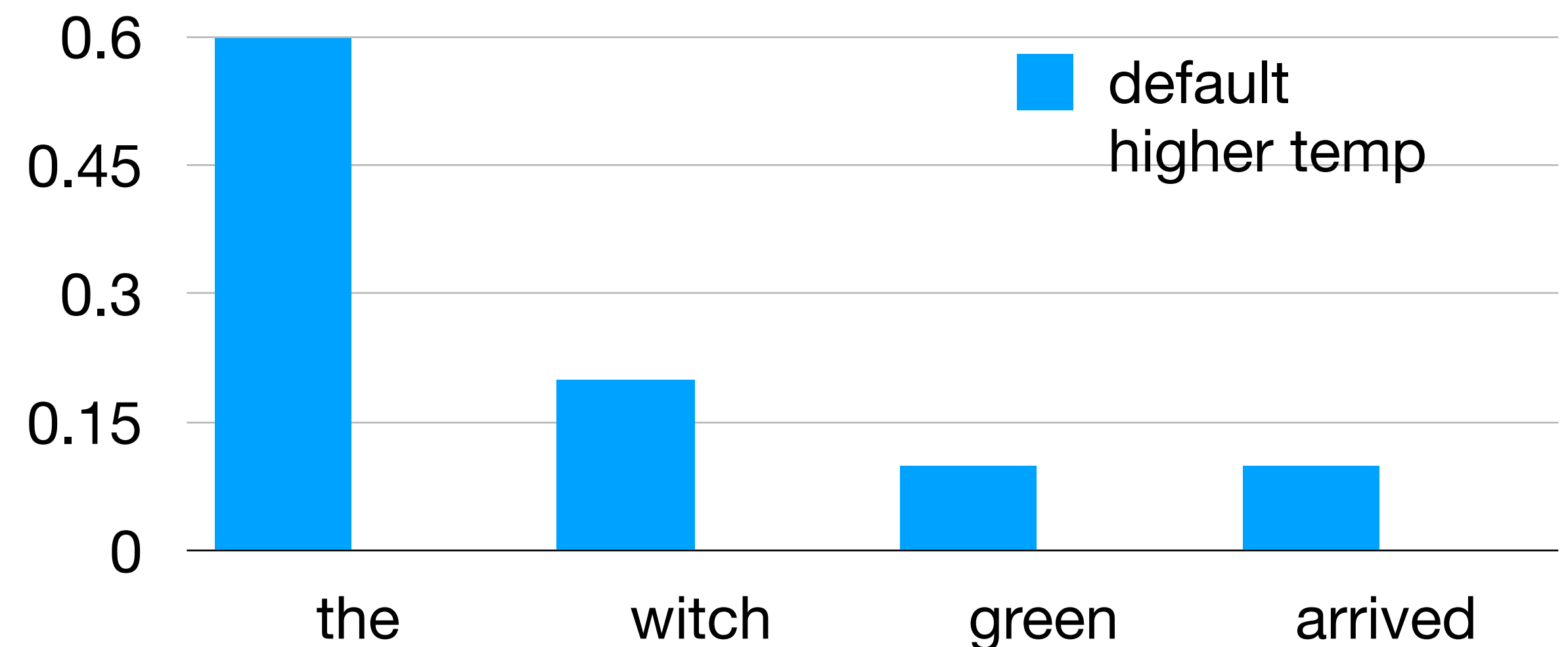
- Some Problems:
 - Heuristic \rightarrow still not guaranteed to find best solution
 - Struggles with long range dependencies
 - Slow and memory intensive
- In practice, people often apply sampling
 - Sample k continuations, rather than choosing the top k
 - Governed by a *temperature* parameter

Decoding

Beam Search

- Temperature
 - Higher temperature = more oversampling of lower probability continuations
 - Intuition: higher temperature = “flatter” distribution

$$\textit{softmax} = \frac{e^{w_i}}{\sum_u e^{w_j}}$$

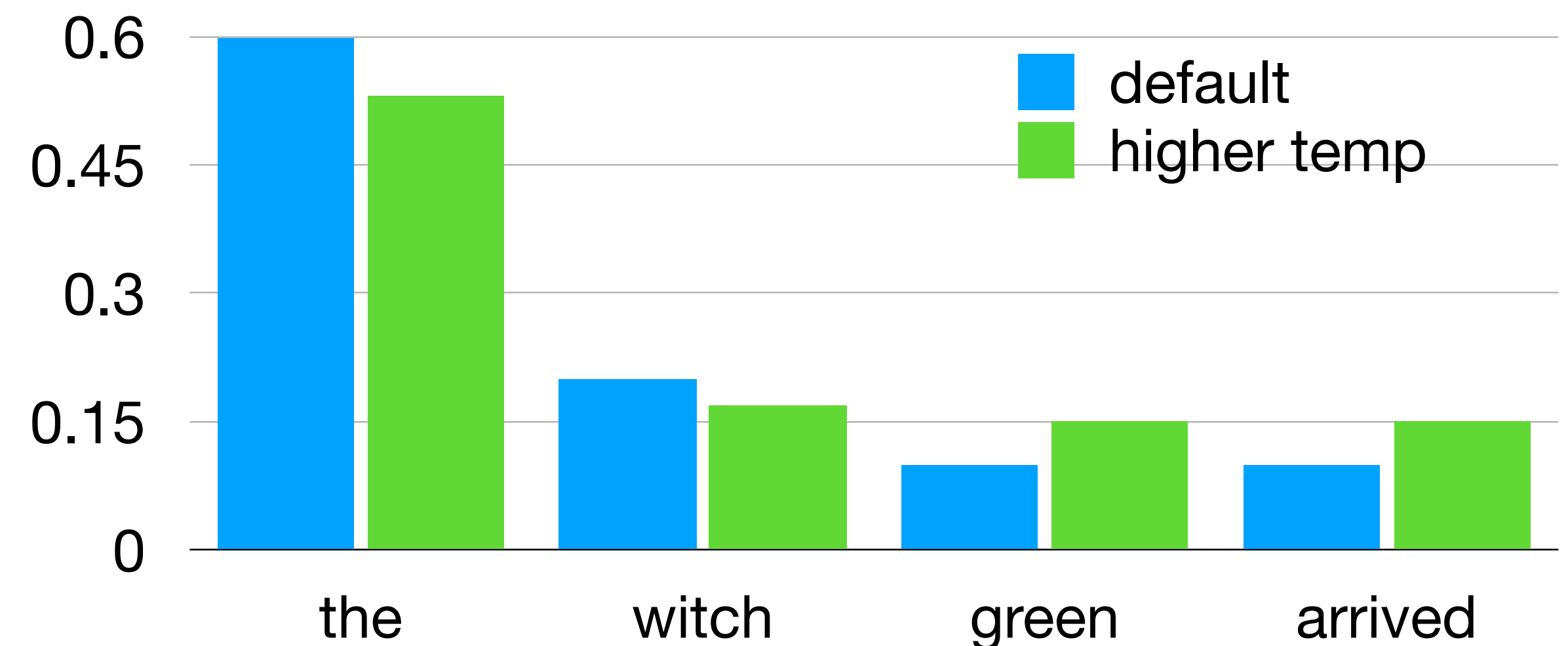


Decoding

Beam Search

- Temperature
 - Higher temperature = more oversampling of lower probability continuations
 - Intuition: higher temperature = “flatter” distribution

$$\textit{softmax} = \frac{e^{w_i/t}}{\sum_u e^{w_j/t}}$$



All done!
Questions?