自然語言處理與應用
Natural Language Processing and Applications

Decoding Strategies

Instructor: 林英嘉 (Ying-Jia Lin)
2025/03/31

Course GitHub

Slido # NLP_0331

Outline

- Recap: Language Generation
- Decoding Strategies
  - Greedy Decoding
  - Beam Search
  - Top-k / Top-p Sampling
- Evaluations

Natural Language Generation (NLG)

- Natural language generation (NLG) is a **process** that that **outputs** text.
- NLG includes a wide variety of NLP tasks.

Machine Translation

Abstractive Summarization

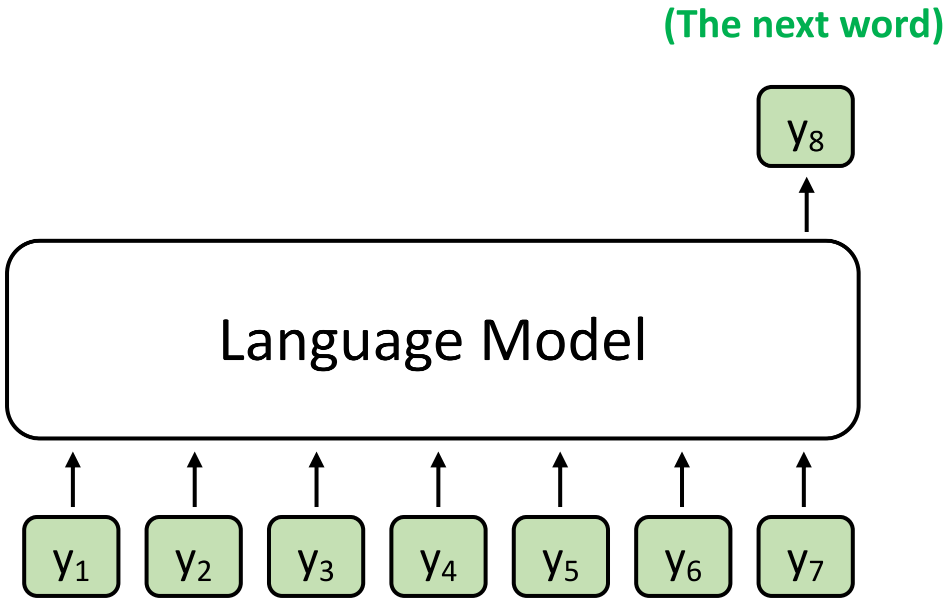
Dialogue Generation (e.g., ChatGPT)

Story Generation

Image Captioning

…

Recap: Language Model



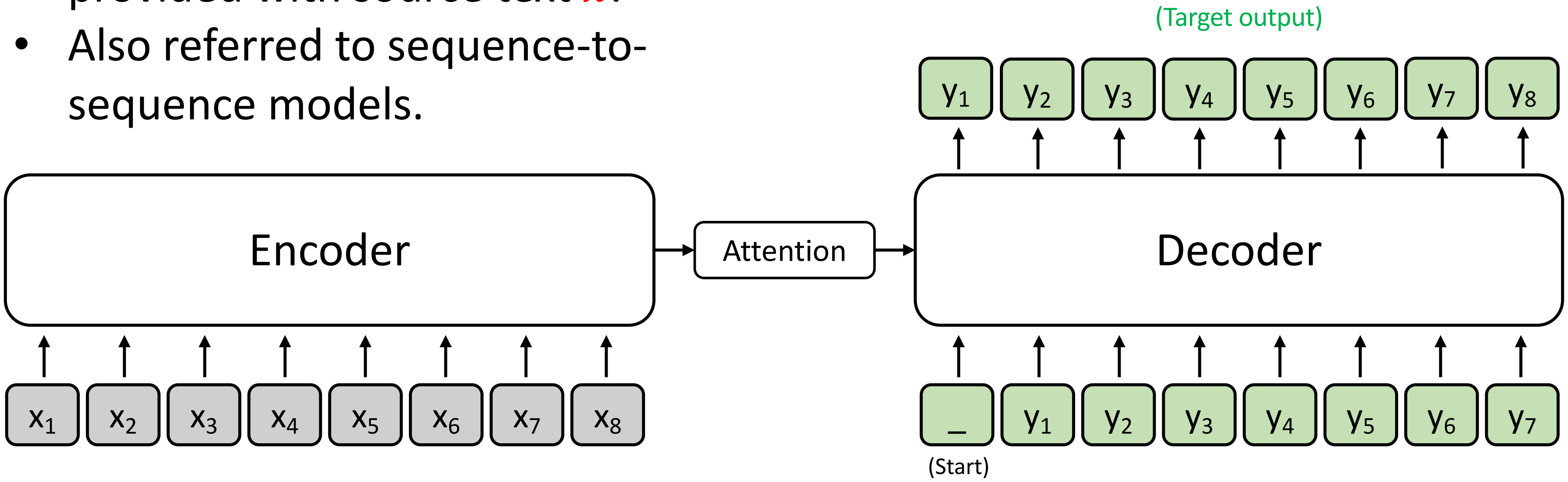
$$P(y_t|y_1, y_2, \dots, y_{t-1})$$

- A model that assigns probabilities to upcoming words is called **a language model**.
- The task involving predictions of upcoming words is **language modeling**.

Recap: Conditional Language Model

- In addition to previous words, a conditional language model is provided with source text x.
- Also referred to sequence-to-sequence models.

$$P(y_t | y_1, y_2, \dots, y_{t-1}, x)$$



Tasks of Conditional Language Model

- In addition to previous words (target), a conditional language model is provided with source text x.

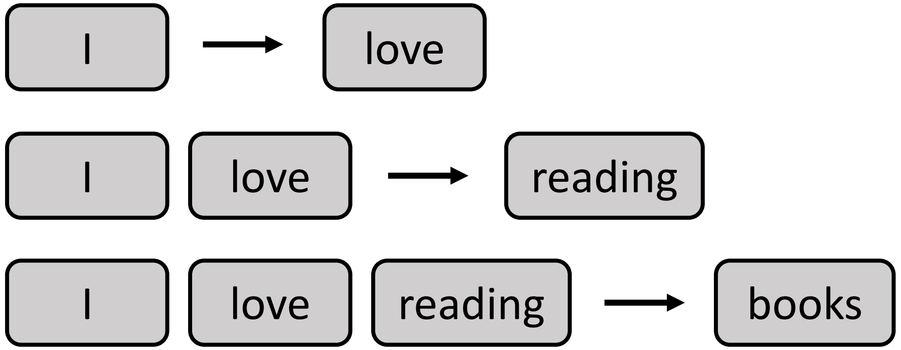
	Source	**Target**
Machine Translation	Language A	Language B
Summarization	Long Text	Concise Text
Dialogue Generation	User Input	Desired User Input
...		

How to train a (Conditional) Language Model?

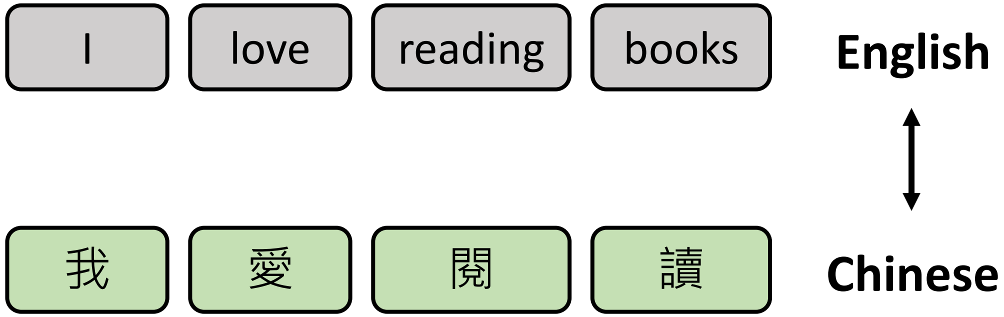
- First, you need a training corpus.

Example: I love reading books.

Language modeling (Unsupervised)

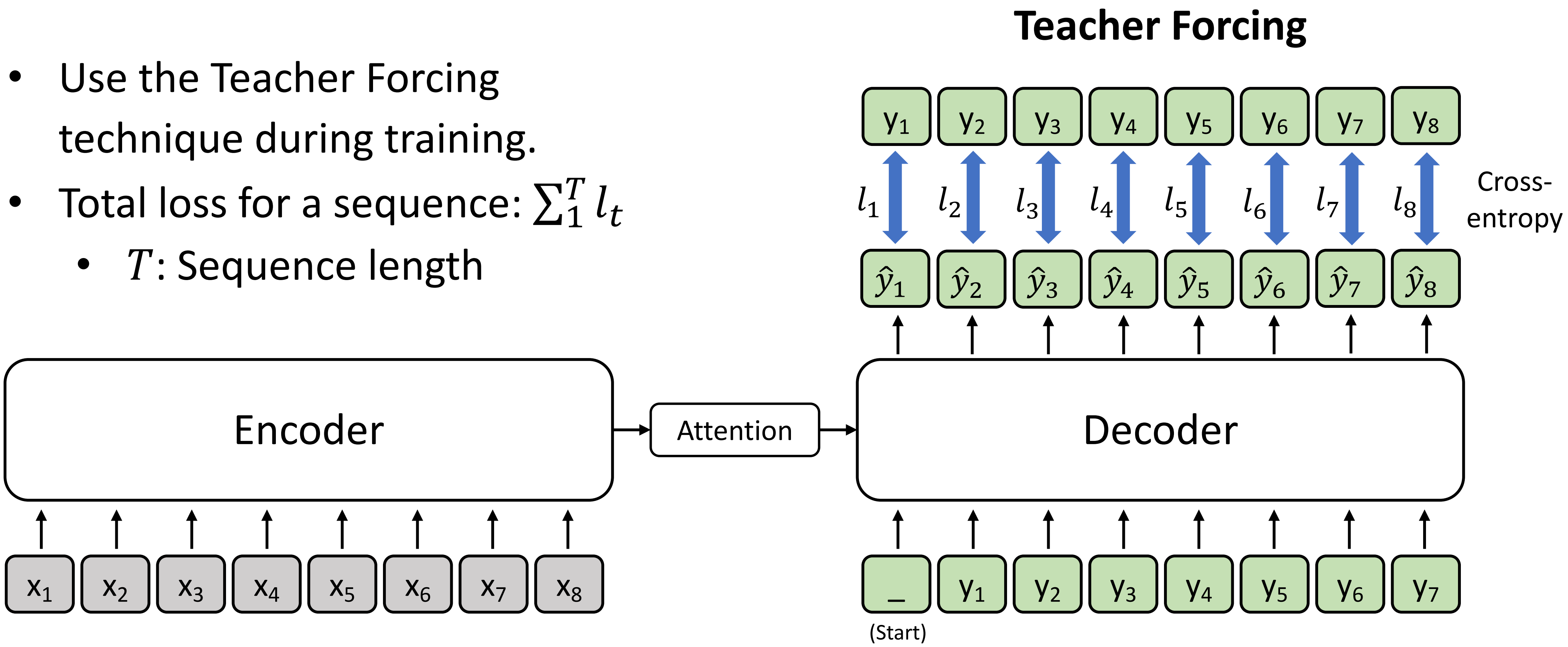


Machine Translation (Supervised)



How to train a (Conditional) Language Model?

- Use the Teacher Forcing technique during training.
- Total loss for a sequence: $\sum_1^T l_t$
  - T: Sequence length

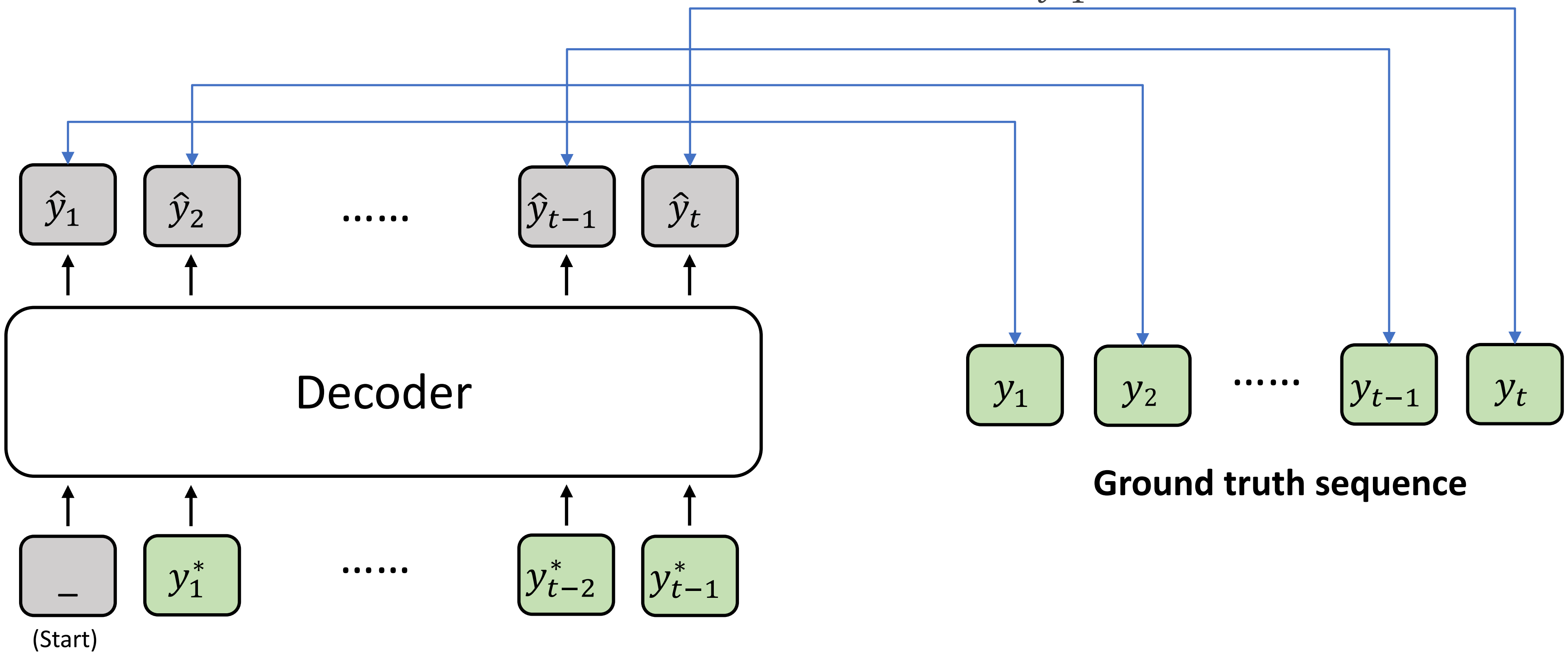


Teacher Forcing
y₁ y₂ y₃ y₄ y₅ y₆ y₇ y₈
l_1 l_2 l_3 l_4 l_5 l_6 l_7 l_8
Cross-entropy
$\hat{y}_1$ $\hat{y}_2$ $\hat{y}_3$ $\hat{y}_4$ $\hat{y}_5$ $\hat{y}_6$ $\hat{y}_7$ $\hat{y}_8$
Encoder
Attention
Decoder
x₁ x₂ x₃ x₄ x₅ x₆ x₇ x₈
_ y₁ y₂ y₃ y₄ y₅ y₆ y₇
(Start)

Teacher Forcing – Training Time

During training:

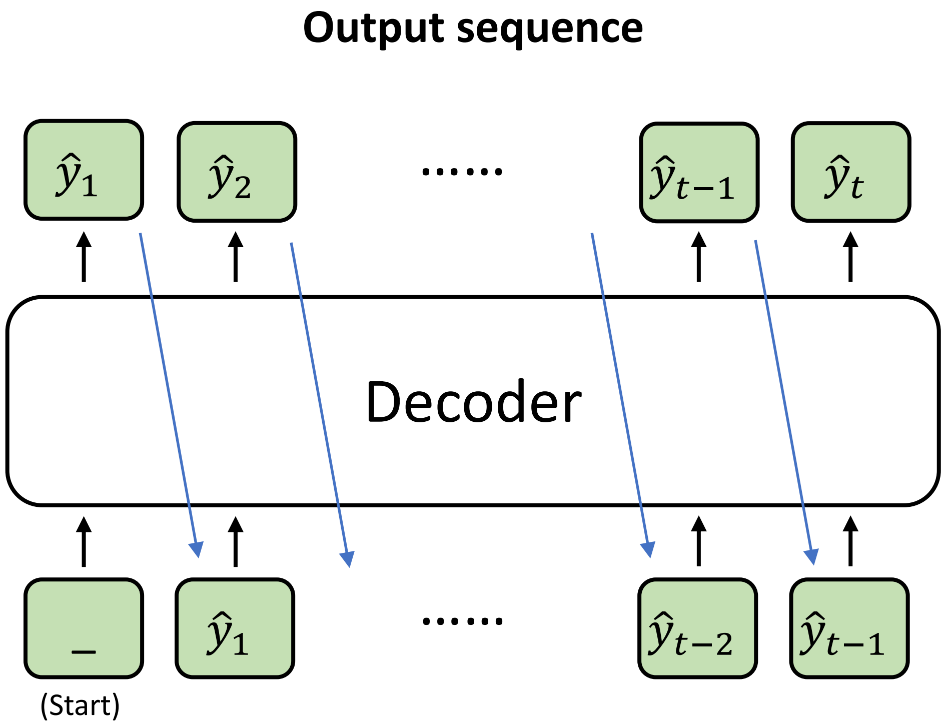
$$L_{ml} = -\sum_{t=1}^{n'} \log p(y_t | y_1, \dots, y_{t-1}, x)$$



Ground truth sequence

Teacher Forcing – Testing Time

During testing:



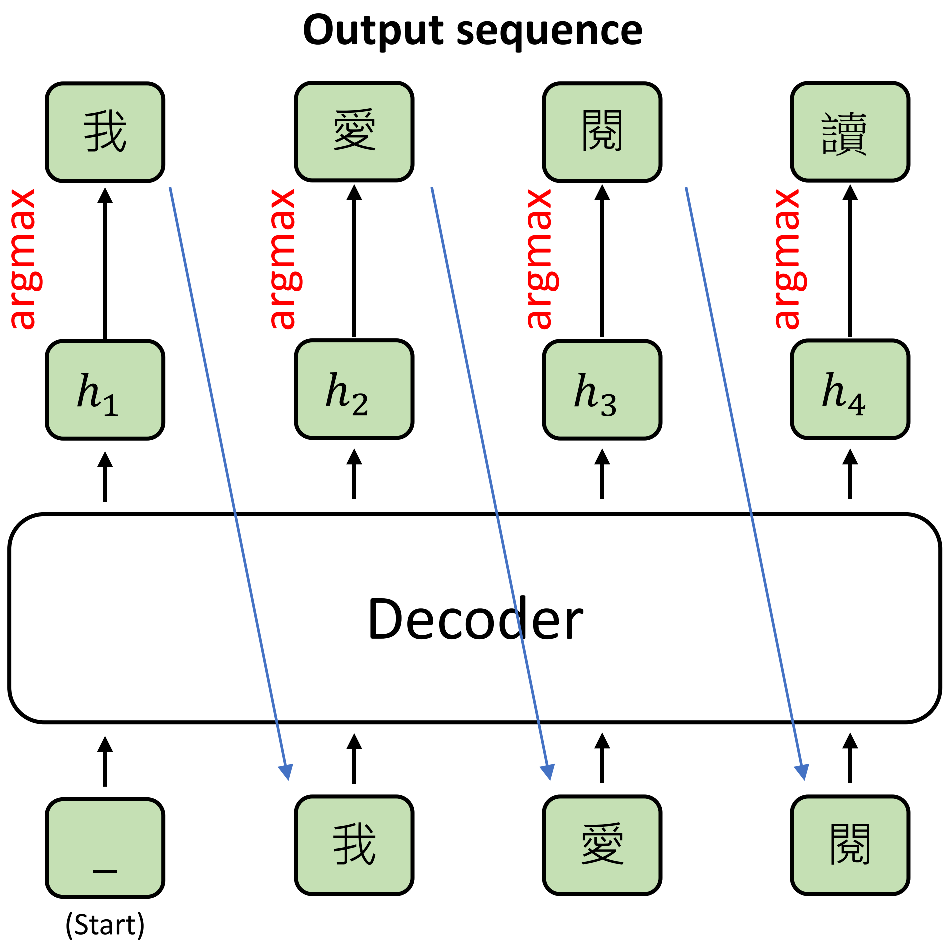
- Advantage: stabilize training and increase performance
- Question: How does the next word be determined?

Decoding Strategies

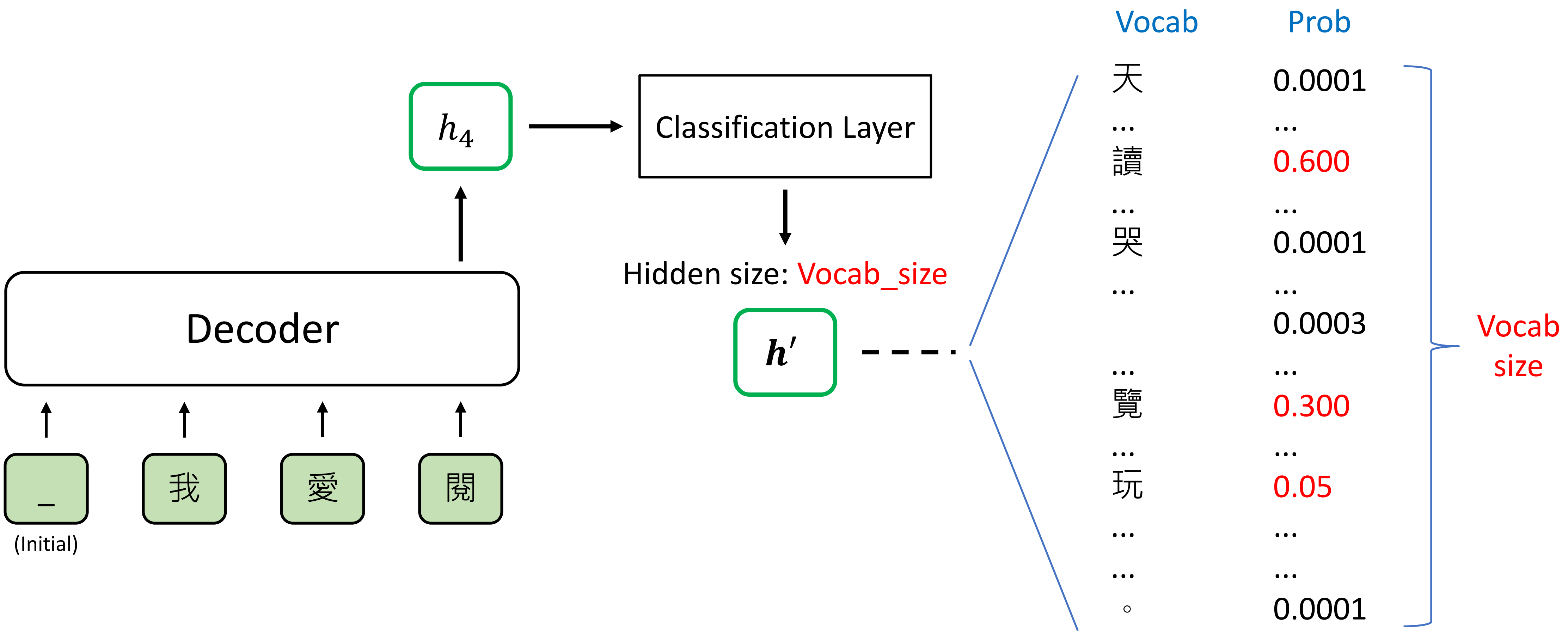
- Greedy Decoding
- Beam Search
- Top-k Sampling
- Top-p Sampling

Greedy Decoding

Example: I love reading books.



Greedy Decoding – Best Selection Process

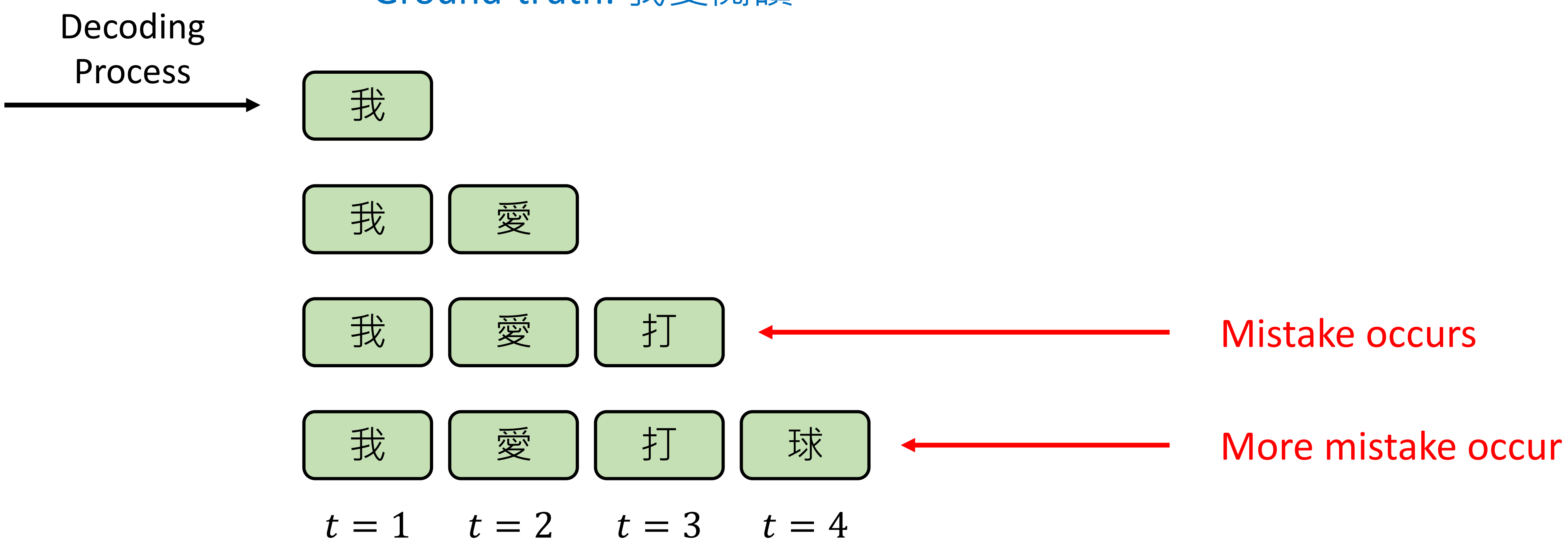


Vocab	Prob
天	0.0001
…	…
讀	0.600
…	…
哭	0.0001
…	…
	0.0003
…	…
覽	0.300
…	…
玩	0.05
…	…
…	…
。	0.0001

Problem of Greedy Decoding

- Greedy decoding cannot undo!

Ground-truth: 我愛閱讀



Decoding Process

我

我 愛

我 愛 打

Mistake occurs

我 愛 打 球

More mistake occur

$t = 1$ $t = 2$ $t = 3$ $t = 4$

Re-thinking Greedy Decoding

- Greedy decoding cannot undo!
- Greedy decoding only provides one best choice at each time step.
- How about providing more than one choices at each time step?

➡ **Beam Search**

Beam Search

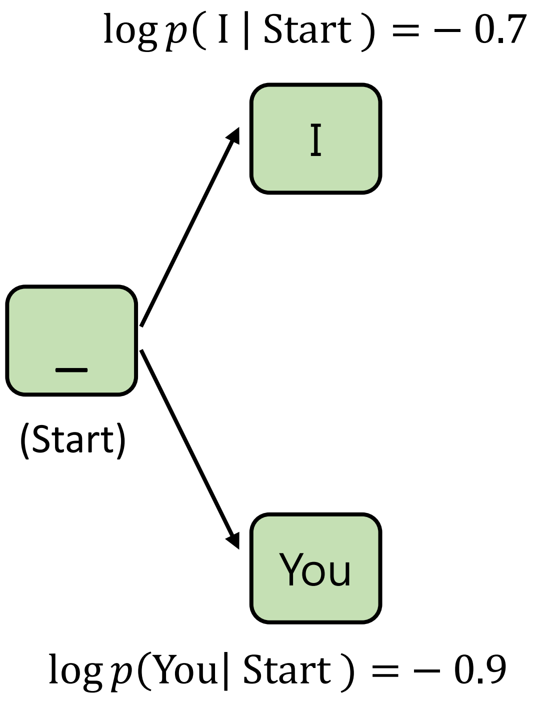
- Set the `Beam size` (or `Beam width`) = 2
  - This means that the number of candidates will be preserved at each decoding time.
  - Beam size is a hyperparameter for beam search decoding.
- At each decoding time step, a score is calculated via the following equation:

$$L_{ml} = -\sum_{t=1}^{n'} \log p(y_t^* | y_1^*, \ldots, y_{t-1}^*, x)$$

*代表 (強調) 當前時間點生成機率最大的選項

Beam Search ($t = 1$)

`Beam size` = 2

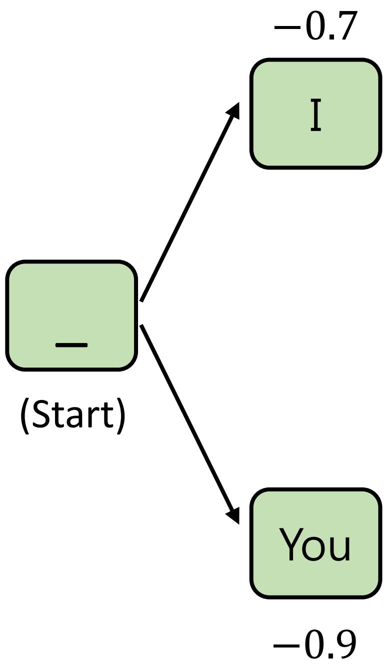


$\log p(\text{ I } | \text{ Start }) = -0.7$
I
_
(Start)
You
$\log p(\text{You}| \text{ Start }) = -0.9$

- At this decoding step, two choices are preserved.

Beam Search ($t = 1$)

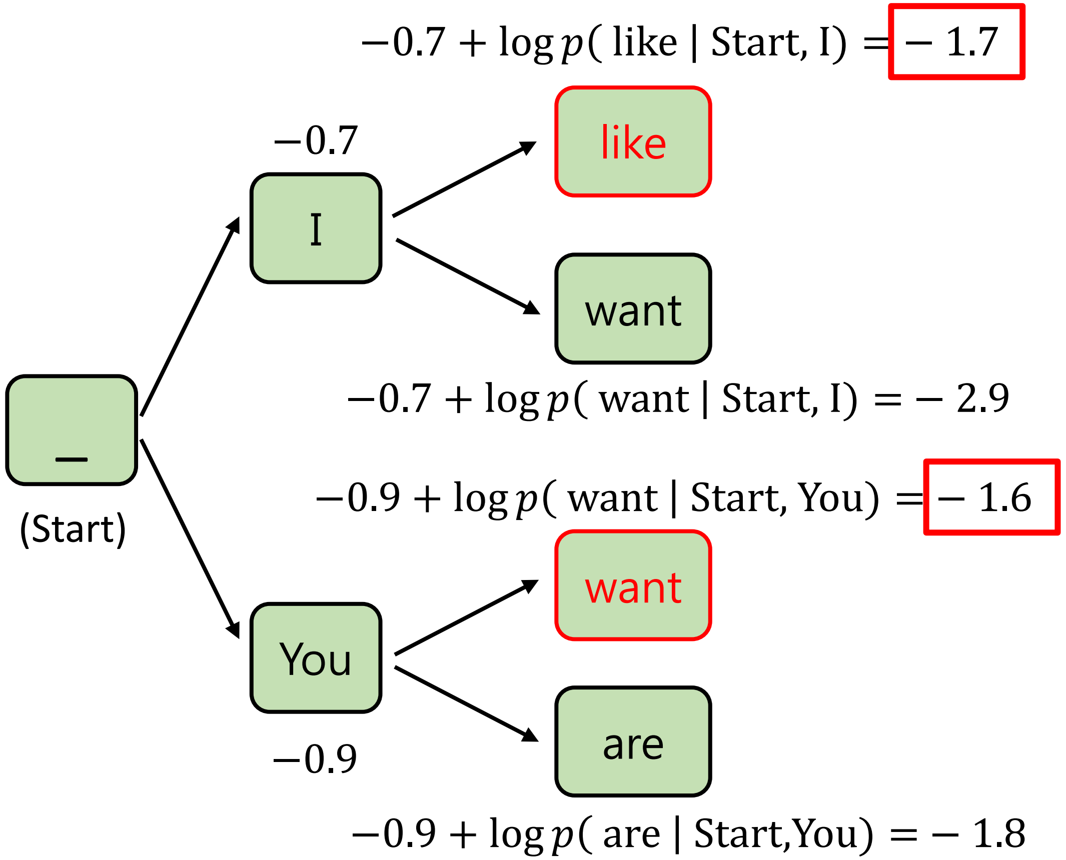
`Beam size` = 2



- At this decoding step, two choices are preserved.

Beam Search ($t = 2$)

`Beam size` = 2


$-0.7 + \log p(\text{ like} \mid \text{Start, I}) = \boxed{-1.7}$
-0.7
I
like
want
$-0.7 + \log p(\text{ want} \mid \text{Start, I}) = -2.9$
_
(Start)
$-0.9 + \log p(\text{ want} \mid \text{Start, You}) = \boxed{-1.6}$
want
You
are
-0.9
$-0.9 + \log p(\text{ are} \mid \text{Start,You}) = -1.8$

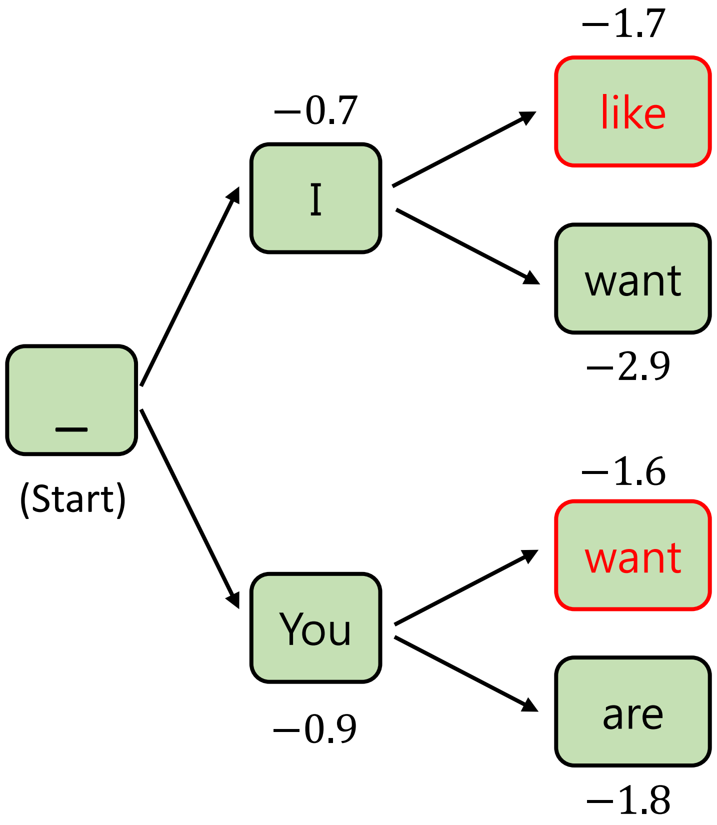
$$L_{ml} = -\sum_{t=1}^{T} \log p(y_t^* | y_1^*, \dots, y_{t-1}^*, x)$$

Note the loglikelihood! Being close to zero is better!

- At this decoding step, two choices are preserved, and the other two are discarded.

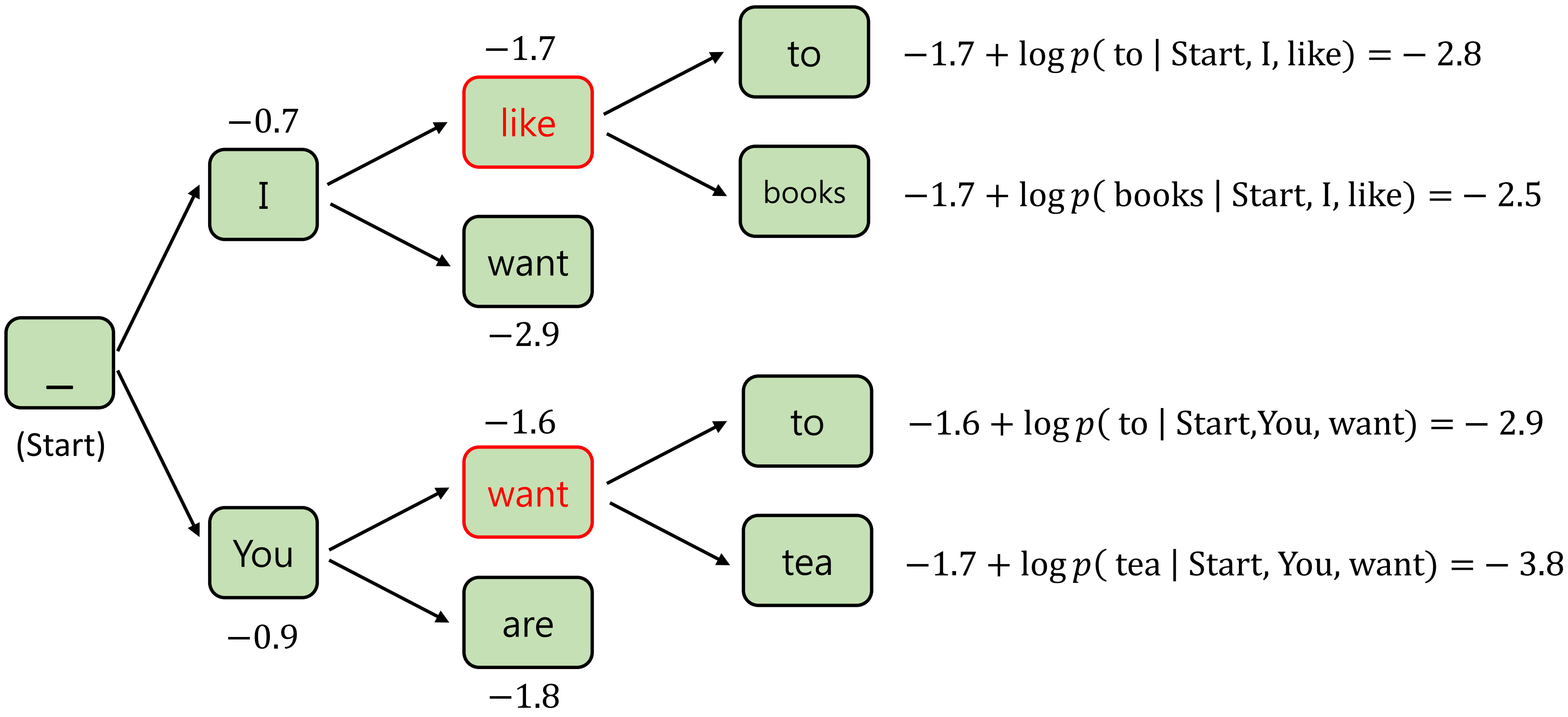
Beam Search ($t = 2$)

`Beam size` = 2



Beam Search ($t = 3$)

`Beam size` = 2



$-1.7 + \log p(\text{ to} \mid \text{Start, I, like}) = -2.8$

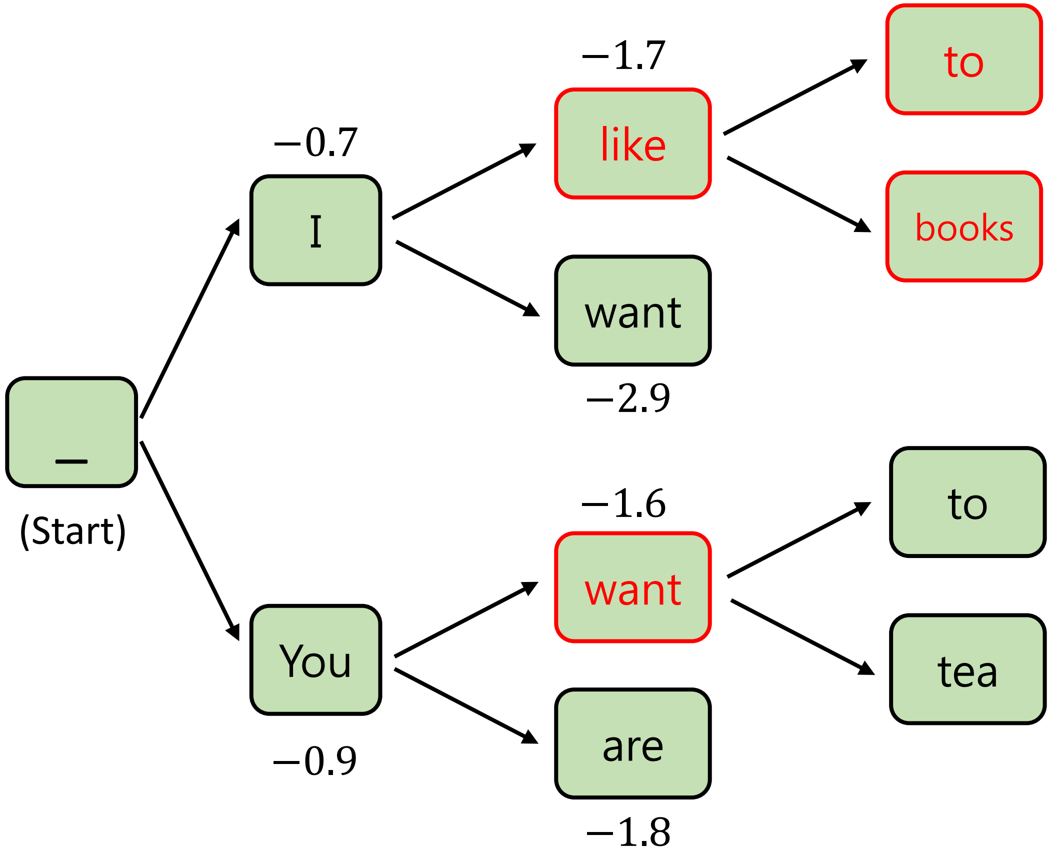
$-1.7 + \log p(\text{ books} \mid \text{Start, I, like}) = -2.5$

$-1.6 + \log p(\text{ to} \mid \text{Start,You, want}) = -2.9$

$-1.7 + \log p(\text{ tea} \mid \text{Start, You, want}) = -3.8$

Beam Search ($t = 3$)

`Beam size` = 2



_ (Start)

I -0.7

You -0.9

like -1.7

want -2.9

want -1.6

are -1.8

to

books

to

tea

$-1.7 + \log p(\text{ to} \mid \text{Start, I, like}) = \boxed{-2.8}$

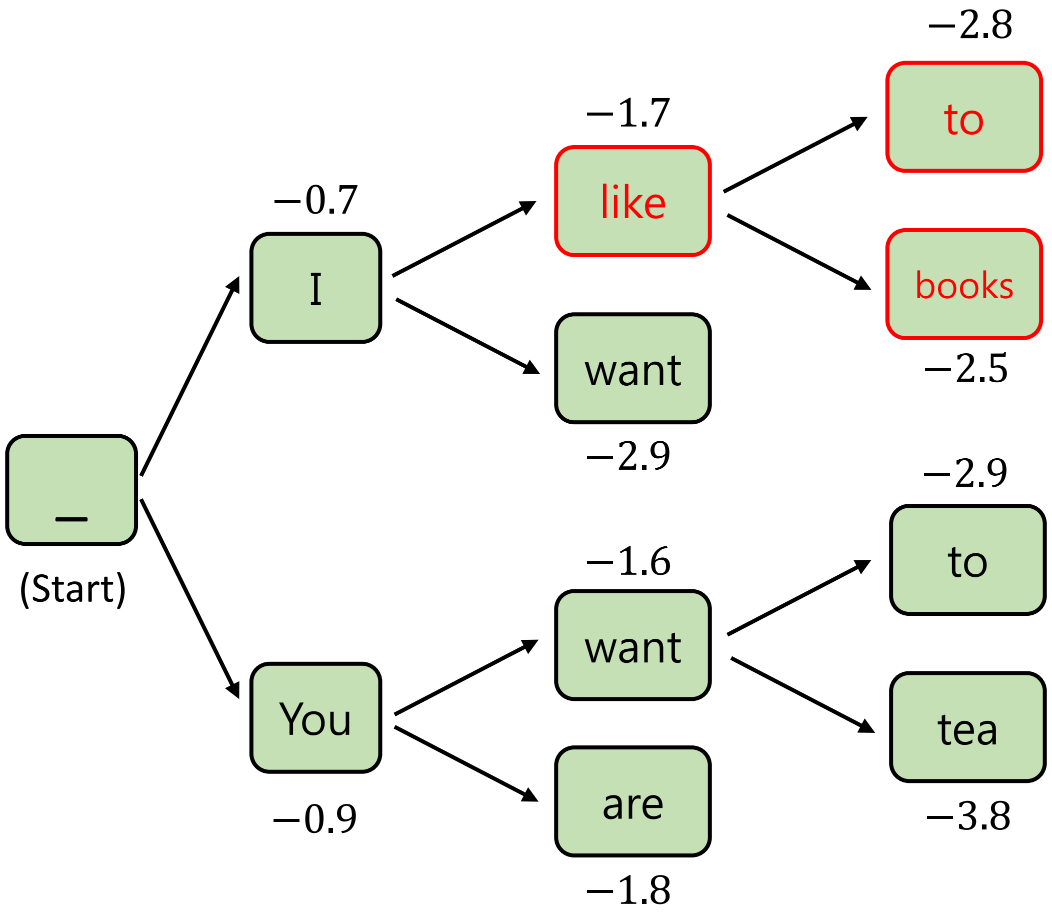
$-1.7 + \log p(\text{ books} \mid \text{Start, I, like}) = \boxed{-2.5}$

$-1.6 + \log p(\text{ to} \mid \text{Start,You, want}) = -2.9$

$-1.7 + \log p(\text{ tea} \mid \text{Start, You, want}) = -3.8$

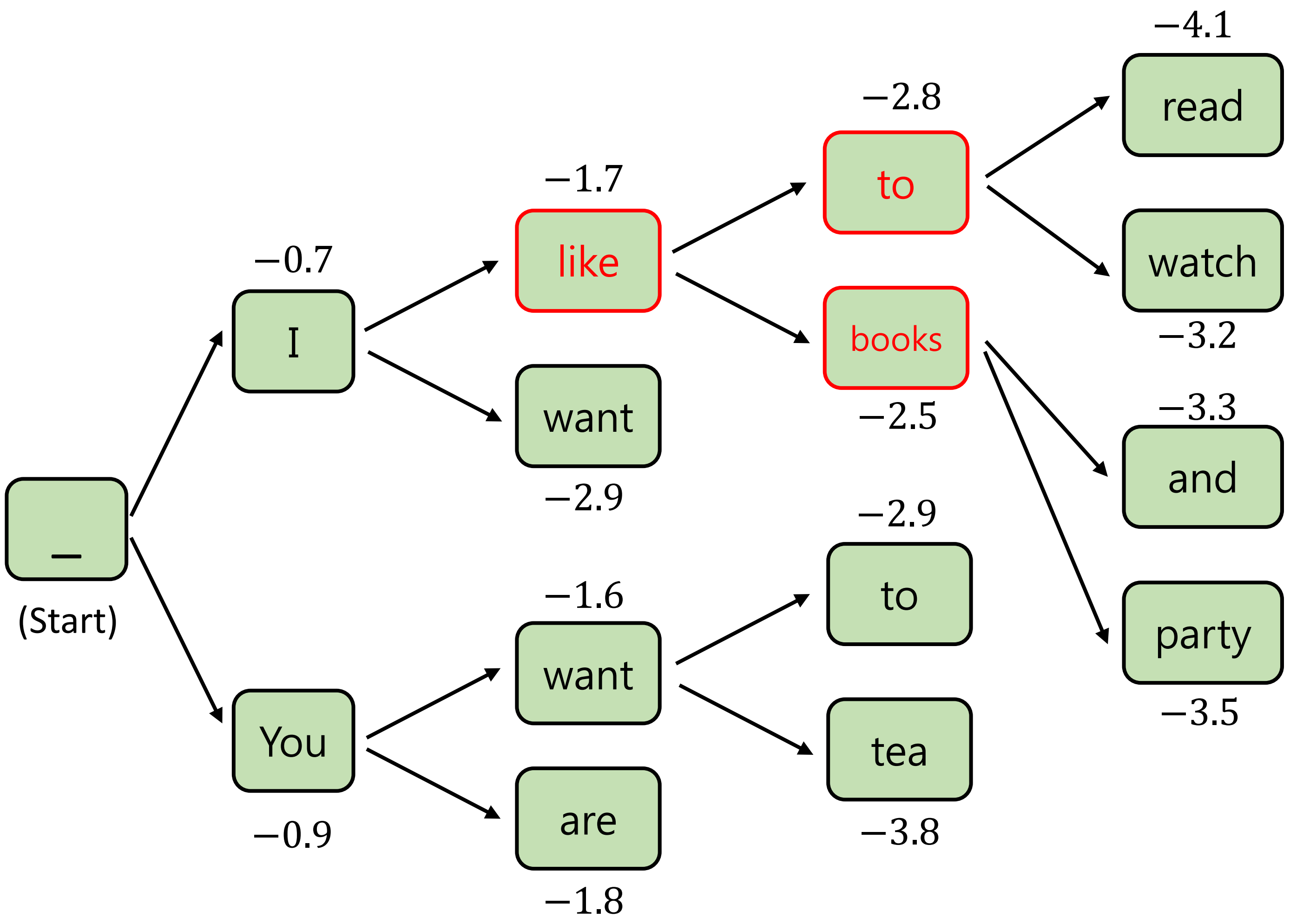
Beam Search ($t = 3$)

`Beam size` = 2



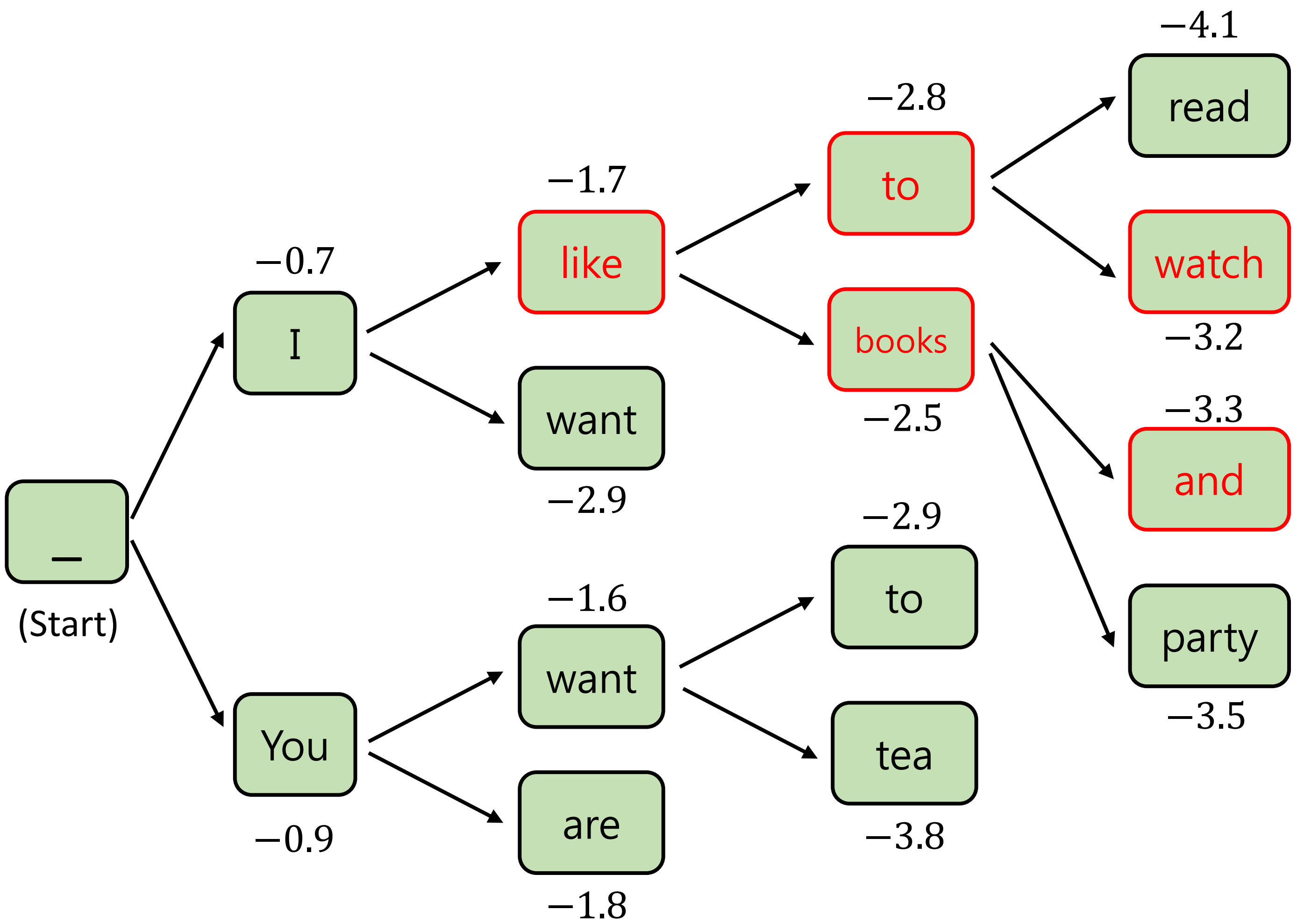
Beam Search ($t = 4$)

`Beam size` = 2



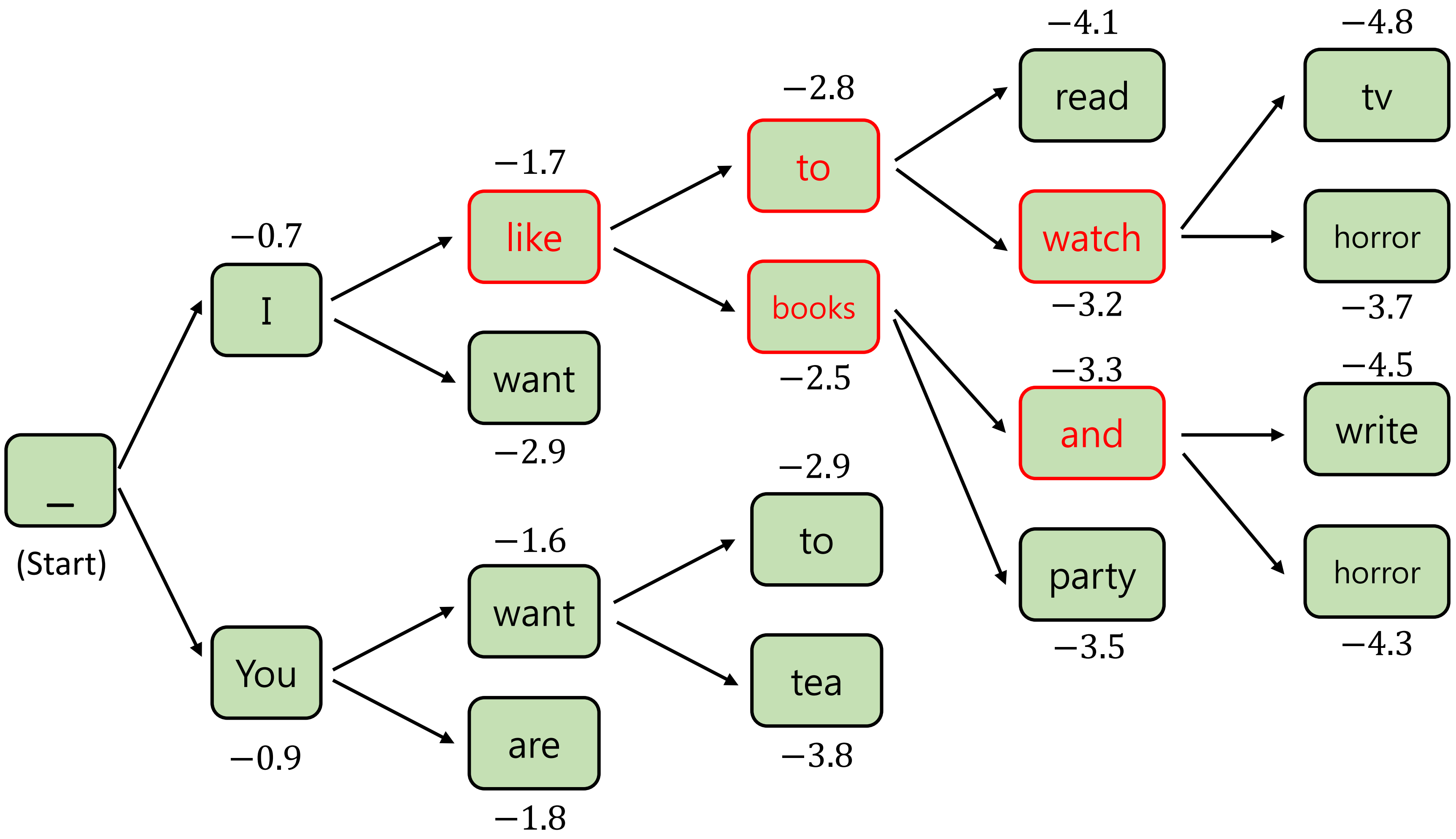
Beam Search ($t = 4$)

`Beam size` = 2



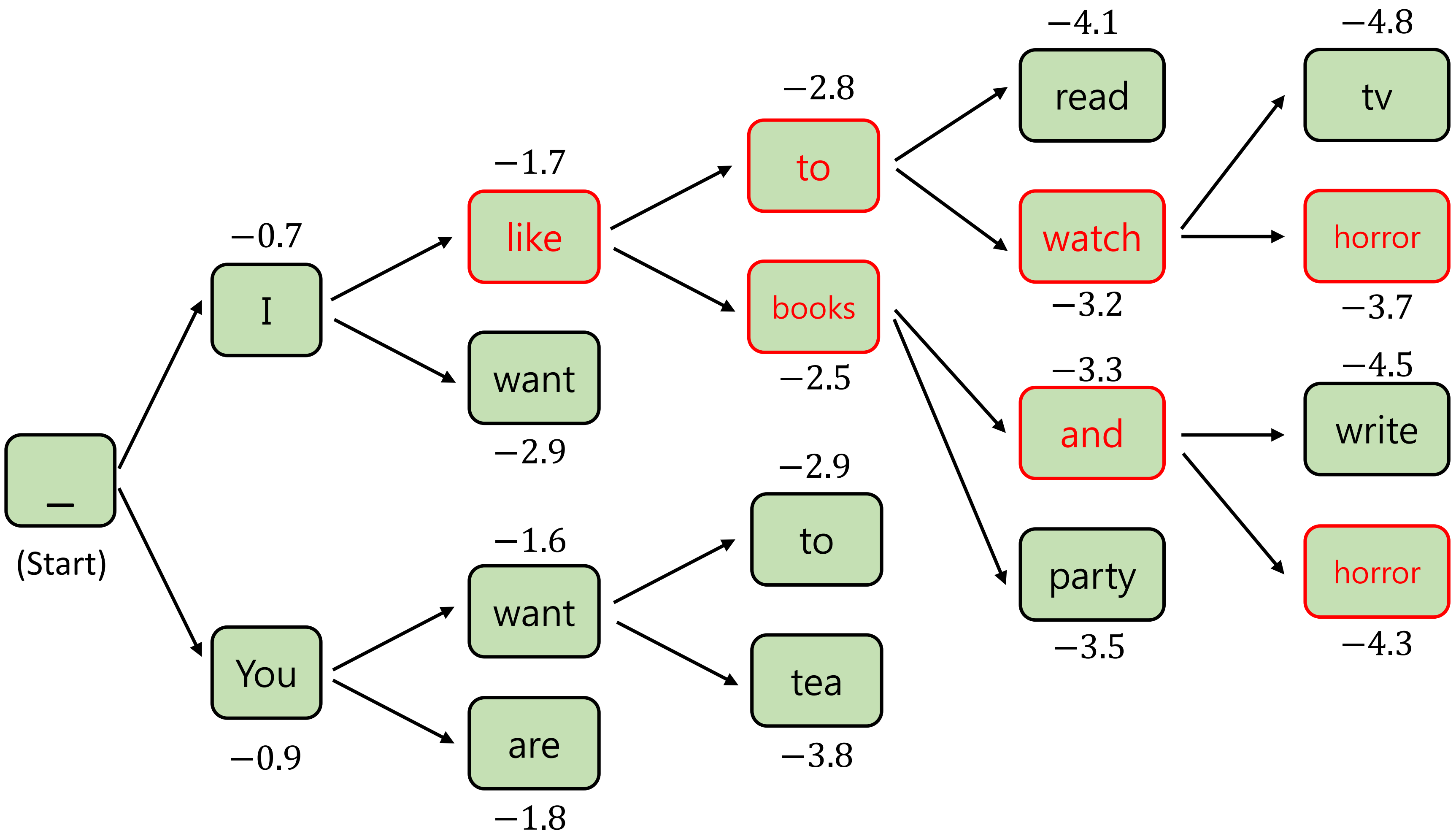
Beam Search ($t = 5$)

`Beam size` = 2



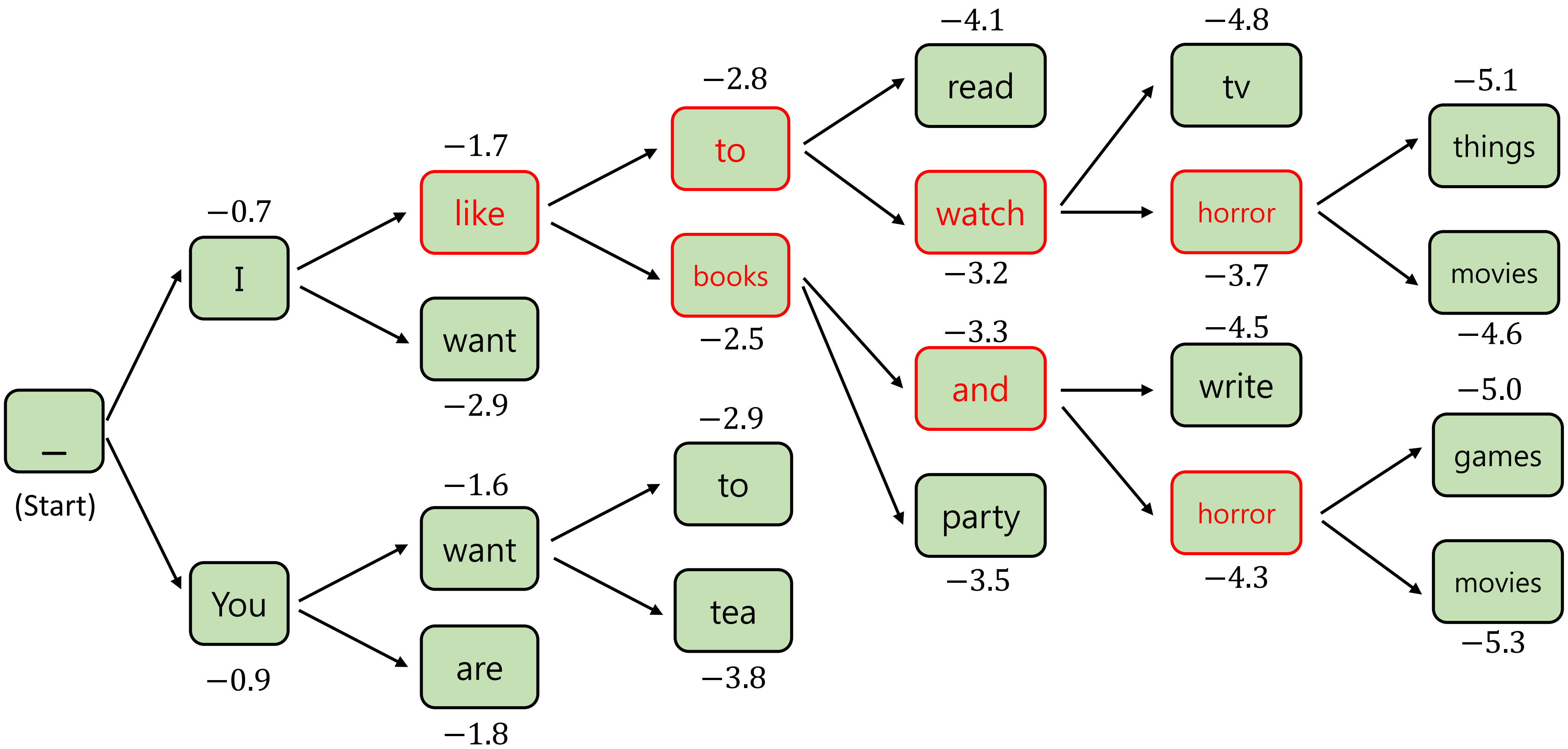
Beam Search ($t = 5$)

`Beam size` = 2



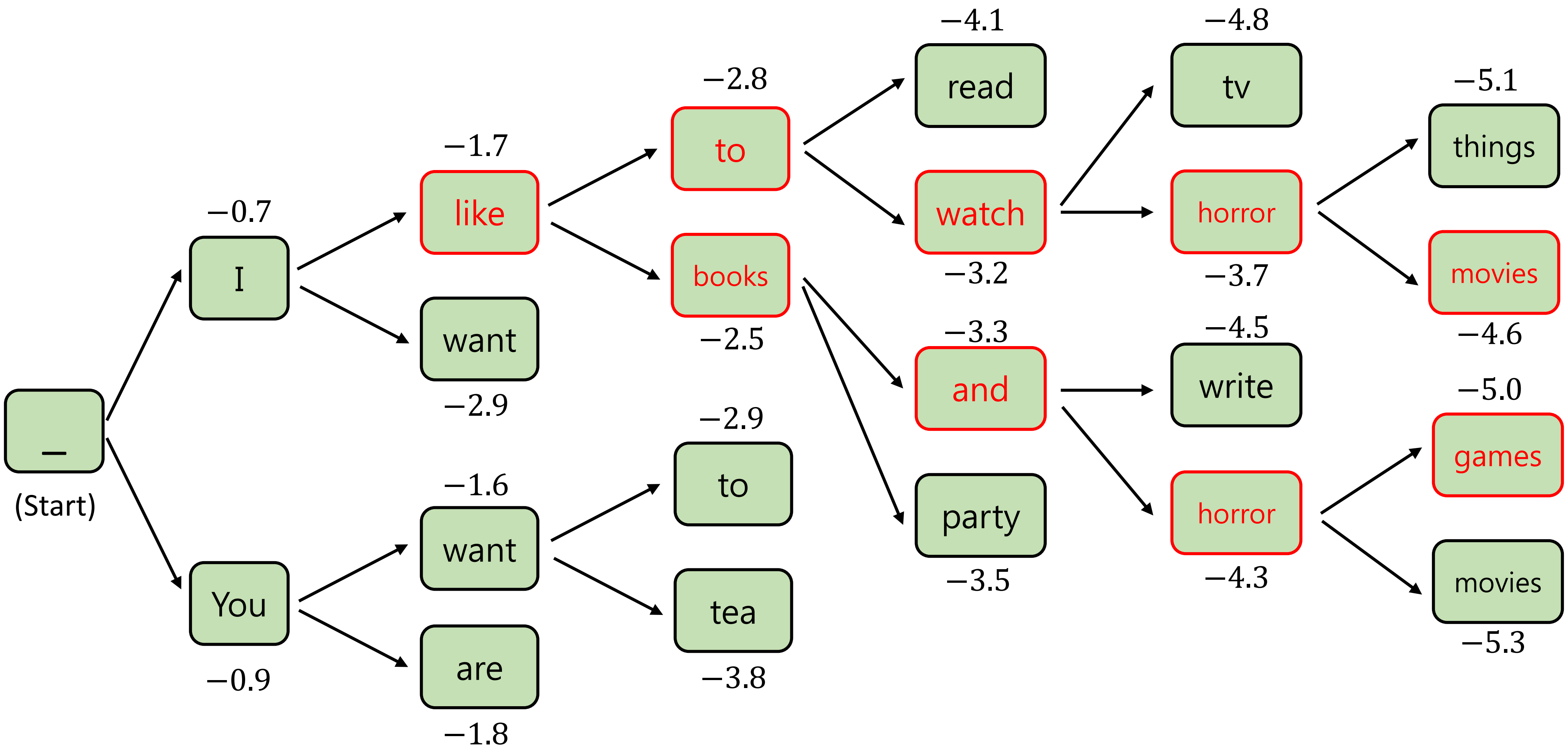
Beam Search ($t = 6$)

`Beam size` = 2



Beam Search ($t = 6$)

`Beam size` = 2



Stop Criterion (停止生成的情況)

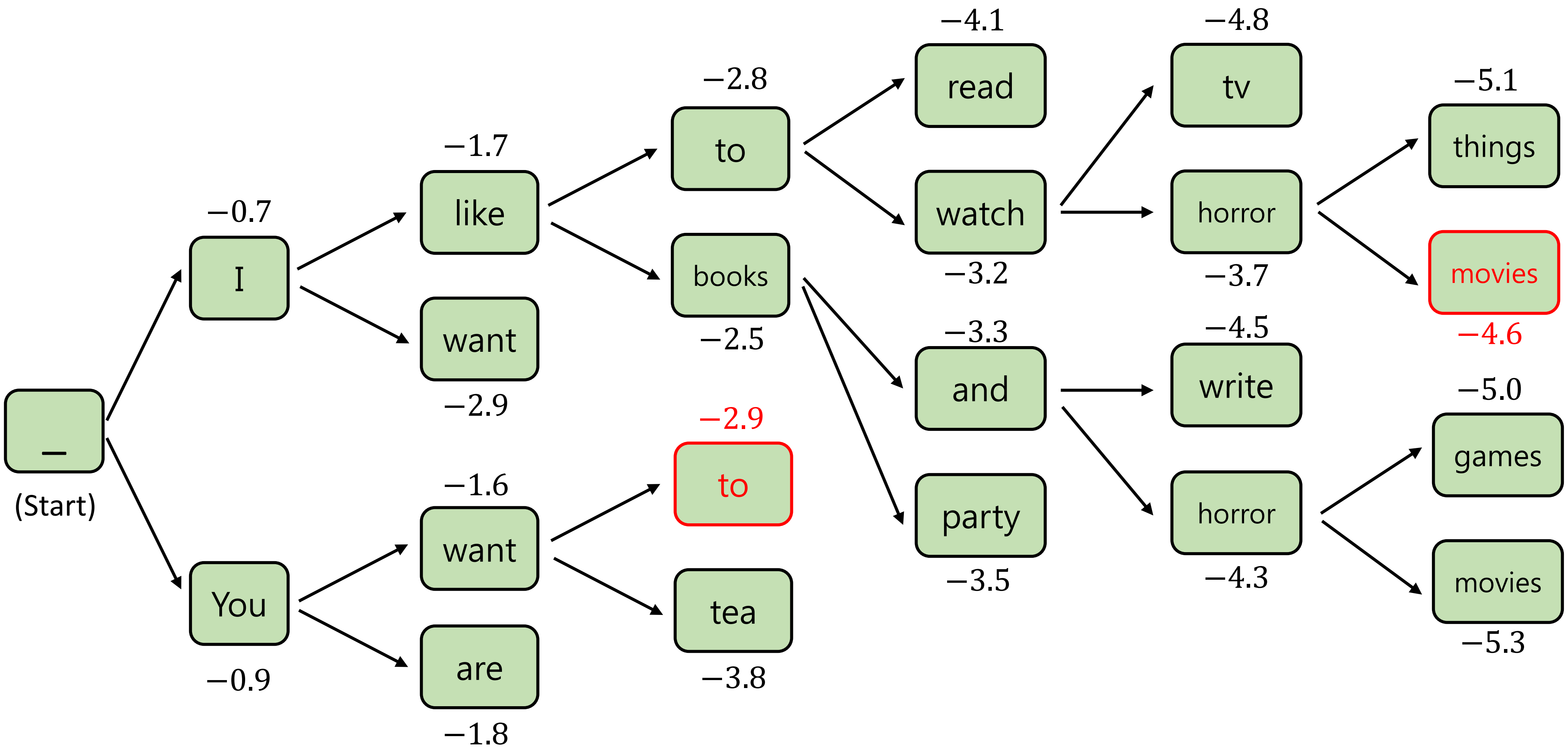
- There are two common stop criterions, either for greedy decoding or beam search decoding (or Top-p / Top-k sampling):
  - We consider a sequence of generation complete when the <EOS> token is produced by a model. *<EOS>: End of sequence
    - E.g., <Start> I like to watch horror movies <EOS>
  - A generated sequence reaches a pre-defined maximal length.

Problem of Beam Search

- Longer candidates will have lower scores.
- (Let’s see again the 6th time step)

Beam Search ($t = 6$)

`Beam size` = 2



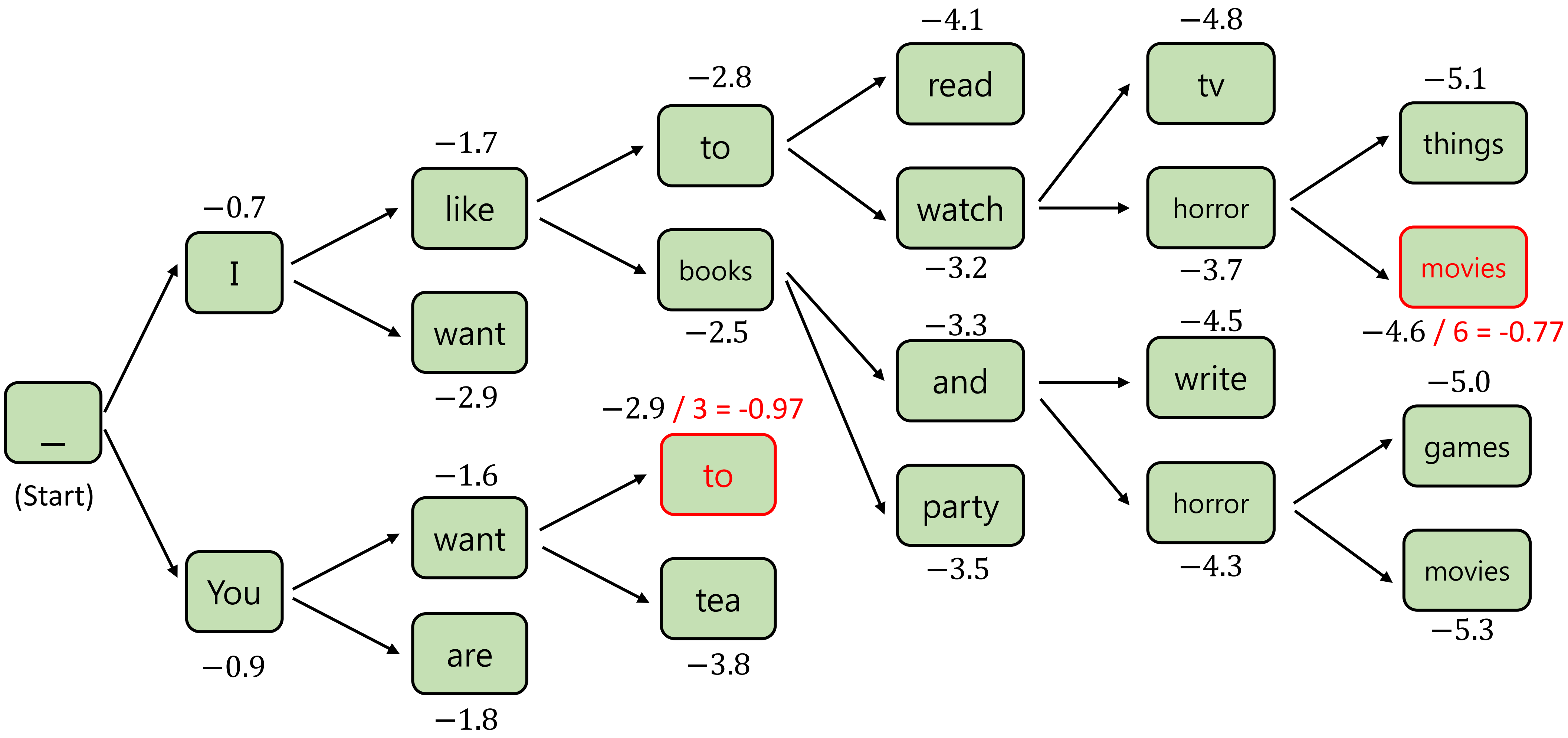
Problem of Beam Search (1)

- Longer candidates will have lower scores.
- Solution: Perform normalization to penalize on length

$$L_{ml} = -\frac{1}{T}\sum_{t=1}^{T} \log p(y_t^* | y_1^*, \ldots, y_{t-1}^*, x)$$

Beam Search ($t = 6$)

`Beam size` = 2



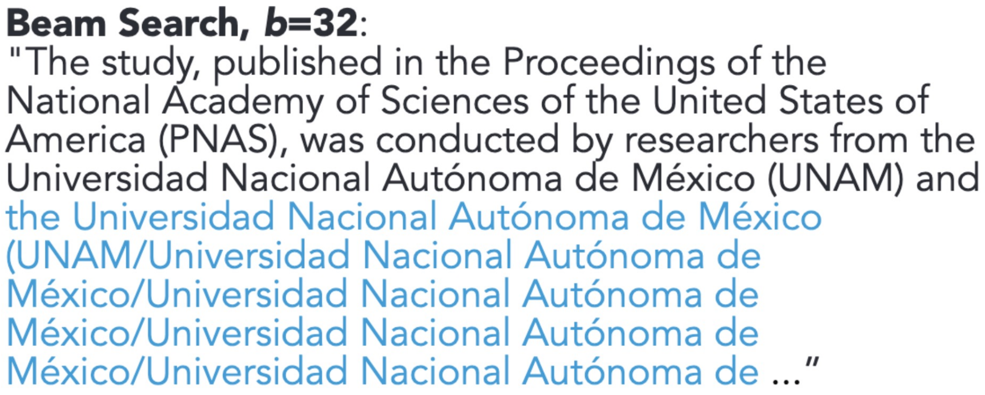
Problem of Beam Search (2)

- Decoding strategies that optimize for output with high probability, such as beam search, lead to text that is incredibly degenerate (e.g., repetitive words), even when using state-of-the-art models such as GPT-2 Large (in 2020).

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Problem of Beam Search (2) – Continued.

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.



Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Problem of Beam Search (2) – Continued.

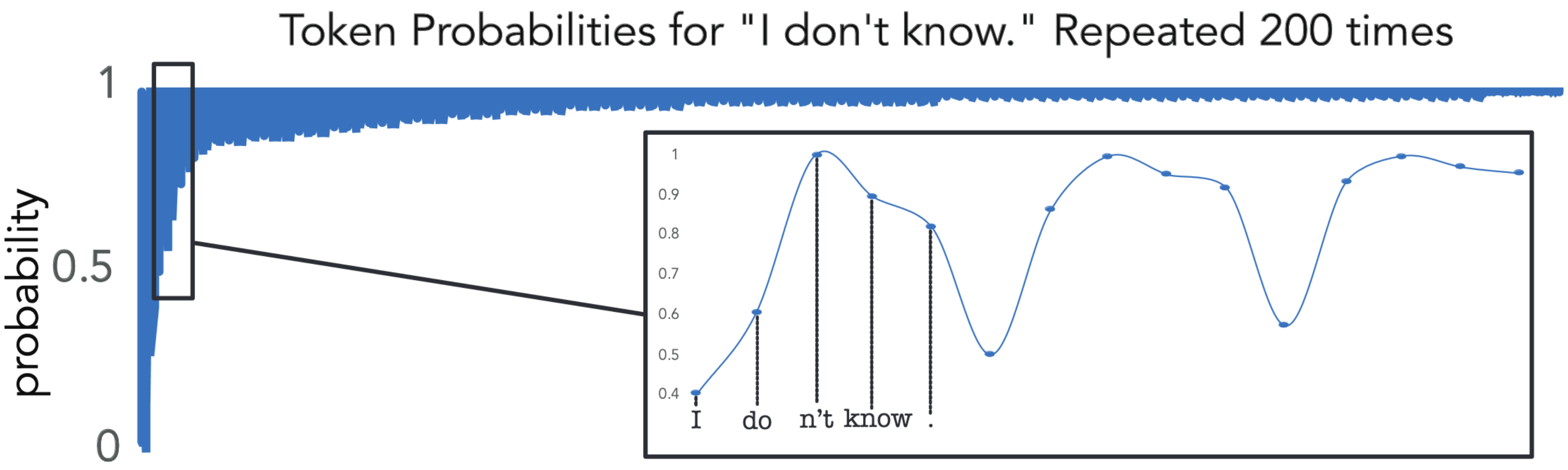


Figure 4: The probability of a repeated phrase increases with each repetition, creating a positive feedback loop. We found this effect to hold for the vast majority of phrases we tested, regardless of phrase length or if the phrases were sampled randomly rather than taken from human text.

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

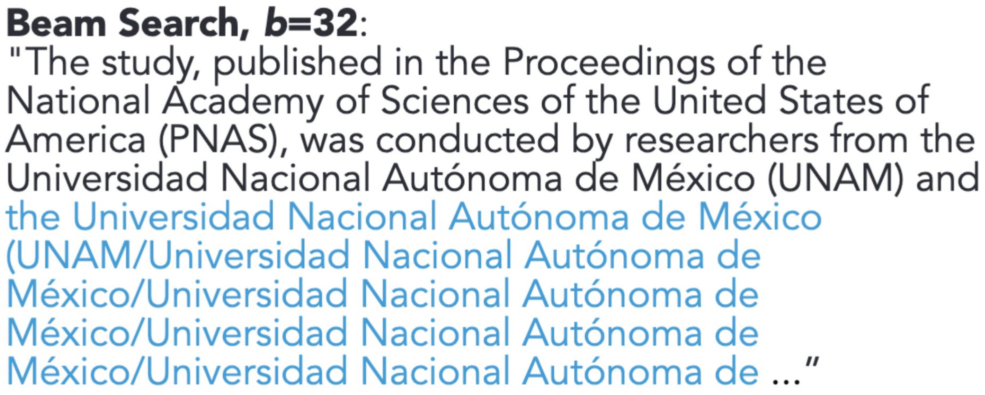
Why is Beam Search so weak?

- 現代語言模型通常使用 maximum likelihood 的方式 (language modeling) 進行訓練，這會導致模型過度偏向常見或高頻 tokens
- 當模型在 early steps 中對某些 tokens 給予極高機率時，這些 tokens 所在的路徑就會大幅壓制了其他 candidate tokens，導致生成缺少多樣性，甚至進入重複生成的loop (例如 I don’t know I don’t know I don’t know …)

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Problem of Beam Search (2) – Continued.

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.



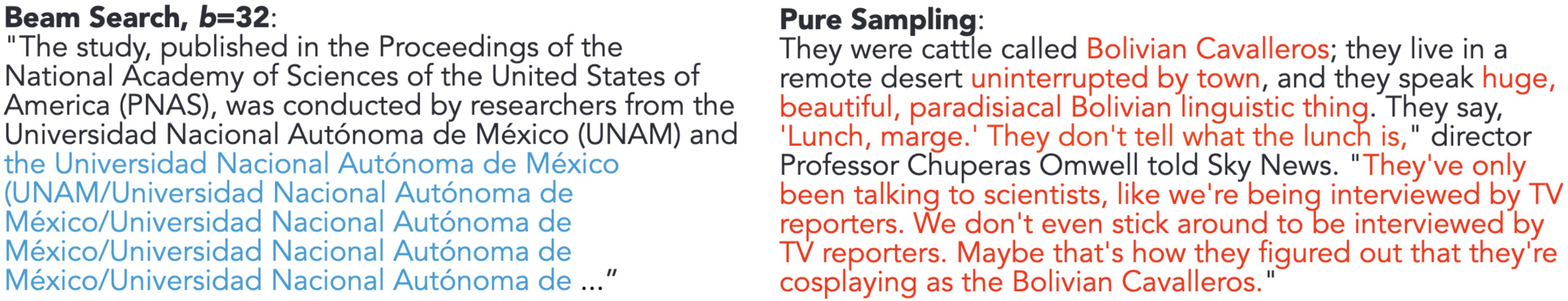
Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

How about random sampling according to the probabilities? (Pure Sampling)

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Problem of Beam Search (2) – Continued.

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.



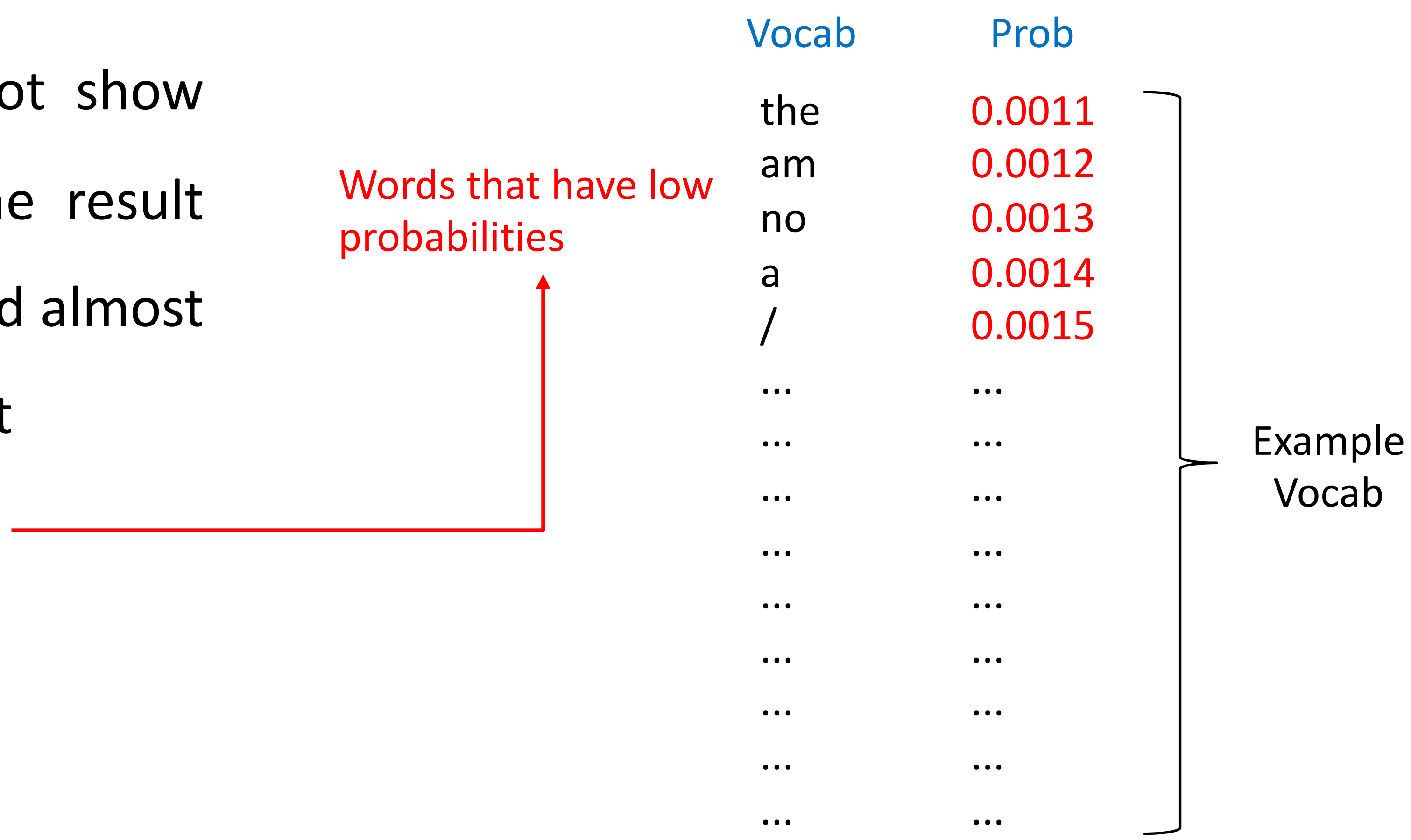
Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

Pure Sampling:
They were cattle called Bolivian Cavalleros; they live in a remote desert uninterrupted by town, and they speak huge, beautiful, paradisiacal Bolivian linguistic thing. They say, 'Lunch, marge.' They don't tell what the lunch is," director Professor Chuperas Omwell told Sky News. "They've only been talking to scientists, like we're being interviewed by TV reporters. We don't even stick around to be interviewed by TV reporters. Maybe that's how they figured out that they're cosplaying as the Bolivian Cavalleros."

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Observation between BS and Pure Sampling

- Pure Sampling does not show repetitive loop, but the result becomes incoherent and almost unrelated to the context
- Why? -> Unreliable tail



Words that have low probabilities

Vocab	Prob
the	0.0011
am	0.0012
no	0.0013
a	0.0014
/	0.0015
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…

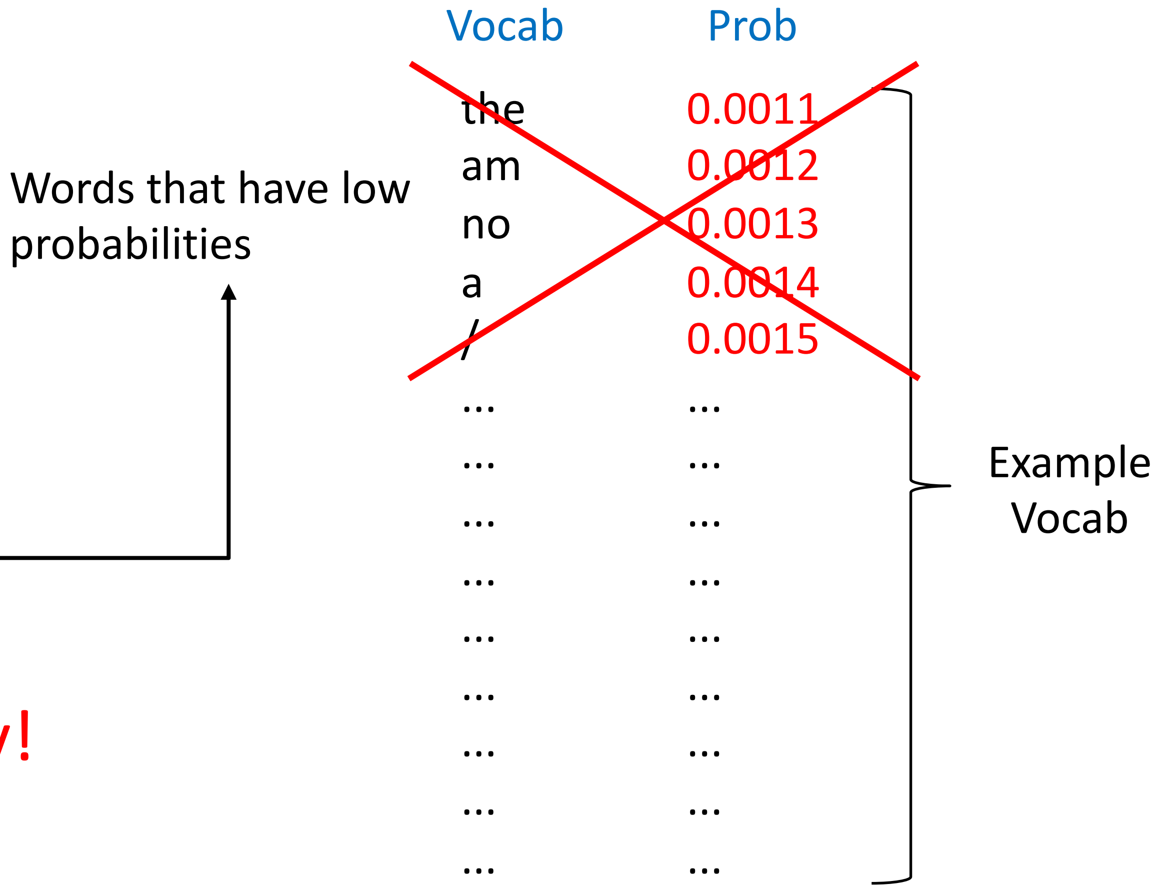
Example Vocab

Observation between BS and Pure Sampling

- Pure Sampling does not show repetitive loop, but the result becomes incoherent and almost unrelated to the context
- Why? -> Unreliable tail

(1) Let’s truncate the vocabulary!

(2) Let’s add more randomness!

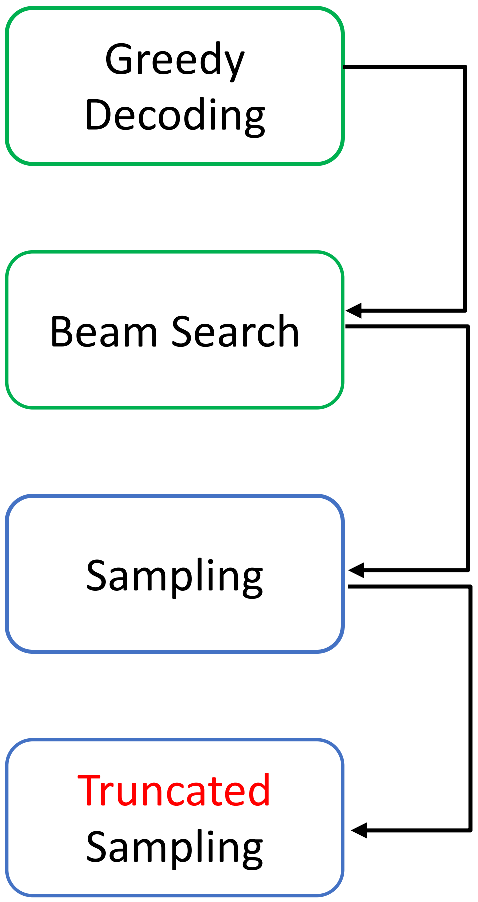


Words that have low probabilities

Vocab	Prob
~~the~~	~~0.0011~~
~~am~~	~~0.0012~~
~~no~~	~~0.0013~~
~~a~~	~~0.0014~~
~~/~~	~~0.0015~~
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…

Example Vocab

Summary and the Thinking Route



Greedy Decoding

Beam Search

Sampling

Truncated Sampling

Greedy decoding always selects the best choice at each time step and becomes problematic for early mistakes.

Models can fall into repetitive loops with beam search, especially for longer text.

Random sampling takes words with low probabilities into considerations.

Sampling

- Sampling 就是「讓模型隨機選下一個詞，而不是每次都選最有可能的那個詞」。
- Sampling 是根據模型對每個詞給出的 機率分布 來進行「隨機抽樣」

Token	Probability (p)
cat	0.5
dog	0.3
car	0.15
book	0.05

Sampling 不是亂抽，而是根據機率 (模型的信心) 來產生下一次生成

Top-k Sampling

*To prevent confusion, we call each unit in a vocabulary as “word” instead of “token”, even though it is a sub-word unit.

- Core idea: truncate the vocabulary with the most probable words
- Steps:

1. Define a value k as the size of truncated vocabulary.
2. Leave the k words with the highest probabilities. Now you get a new truncated vocabulary $V^{(k)}$. (假設k=40，那$V^{(k)}$就只剩40個 tokens)
3. Re-build the probability distribution based on the following normalization:

3-1. 把 $V^{(k)}$ 的機率值加總

3-2. $V^{(k)}$ 內的每個機率值/加總

$$p' = \sum_{x \in V^{(k)}} P(x|x_{1:x-1})$$

$$\mathrm{P}'(\mathrm{x}|x_{1:x-1}) = \begin{cases} P(x|x_{1:x-1})/p' & \text{if } x \in V^{(k)} \\ 0 & \text{otherwise} \end{cases}$$

Top-p Sampling

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations. 2020.

- Core idea: truncate the vocabulary based on probability mass
- Steps:
  1. Define a value p as the probability threshold.
  2. Truncate the vocabulary whose sum of probabilities is greater than *p* :

$$\sum_{x \in V^{(p)}} P(x|x_{1:x-1}) \geq p$$

where $V^{(p)} \subset V$ is the smallest set that fulfills the equation.
Now you get a new truncated vocabulary $V^{(p)}$.

Top-p Sampling

- Core idea: truncate the vocabulary based on probability mass
- Steps:

3. Re-build the probability distribution based on the following normalization:

3-1. 把 $V^{(p)}$ 的機率值加總

3-2. $V^{(p)}$ 內的每個機率值/加總

$$p' = \sum_{x \in V^{(p)}} P(x|x_{1:x-1})$$

$$\mathrm{P'(x|}x_{1:x-1}) = \begin{cases} P(x|x_{1:x-1}/p') & \text{if } x \in V^{(p)} \\ 0 & \text{otherwise} \end{cases}$$

Top-p Sampling example

Token	Probability (p)
cat	0.5
dog	0.3
car	0.15
book	0.05

- 設 p 為 0.8
  - 那 $V^{(p)}$ 中只會有 cat 和 dog 兩個詞

Softmax

- When generating the next word, \`softmax\` is performed to get the probabilities among the words in the vocabulary.
- Softmax formula:

$$\frac{\exp(u_l/\mathrm{t})}{\sum_{l'}^{|V|} \exp\,(u_{l'}/t)}$$

u_l: logits (model outputs before softmax)
$|V|$: size of the vocabulary
t: softmax temperature

Softmax

	Word	Logits		Probability
Example Vocab	the	0.0011	→	0.78
	am	0.0012	→	0.11
	no	0.0013	→	0.03
	a	0.0014	→	0.02
	/	0.0015	→	0.06

- Note that softmax is required for every decoding strategies since we need to find out the next word from a vocabulary.

Properties of Softmax Temperature

u_l: logits (model outputs before softmax)
$|V|$: size of the vocabulary
t: softmax temperature

- Softmax formula:

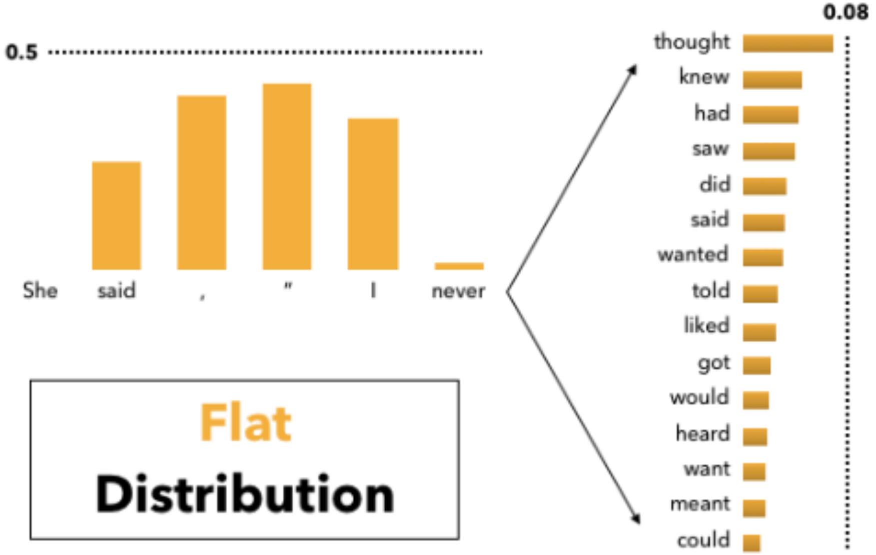
$$\frac{\exp(u_l/\mathrm{t})}{\sum_{l'}^{|V|}\exp\left(u_{l'}/t\right)}$$

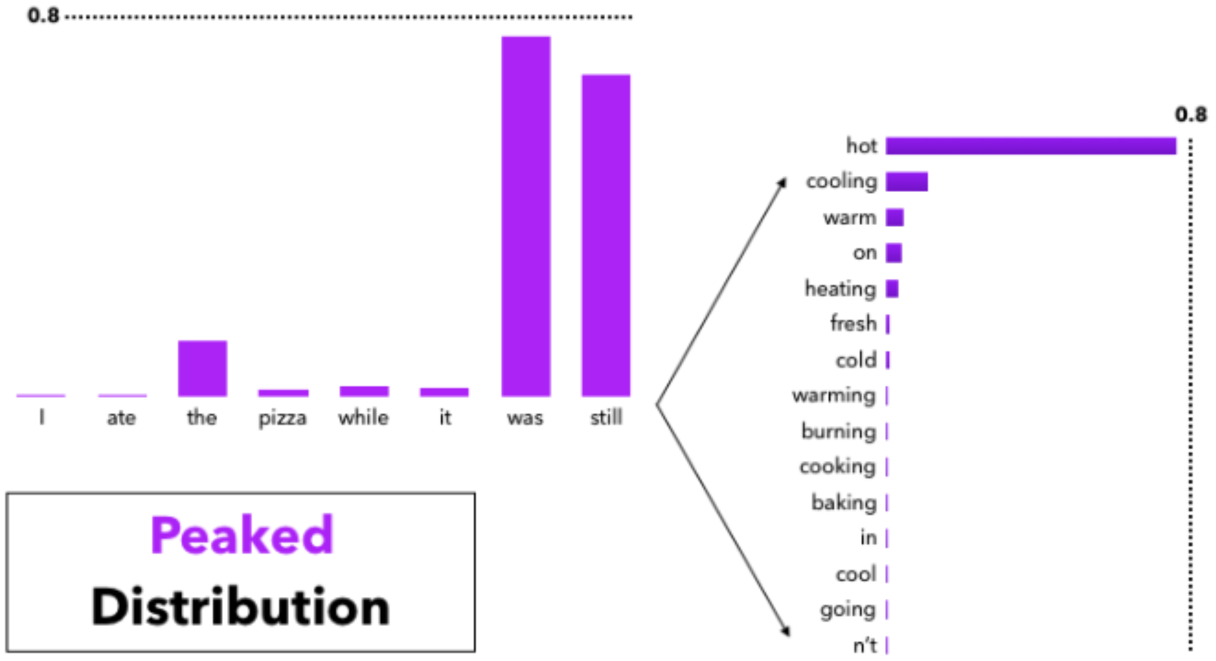
- Larger t -> Lower probability value -> Smaller ranges of probability distribution -> More diverse Outputs
- Smaller t -> Higher probability value -> Greater ranges of probability distribution -> Less diverse Outputs

Softmax Temperature

Temperature 高的時候 (more diverse)

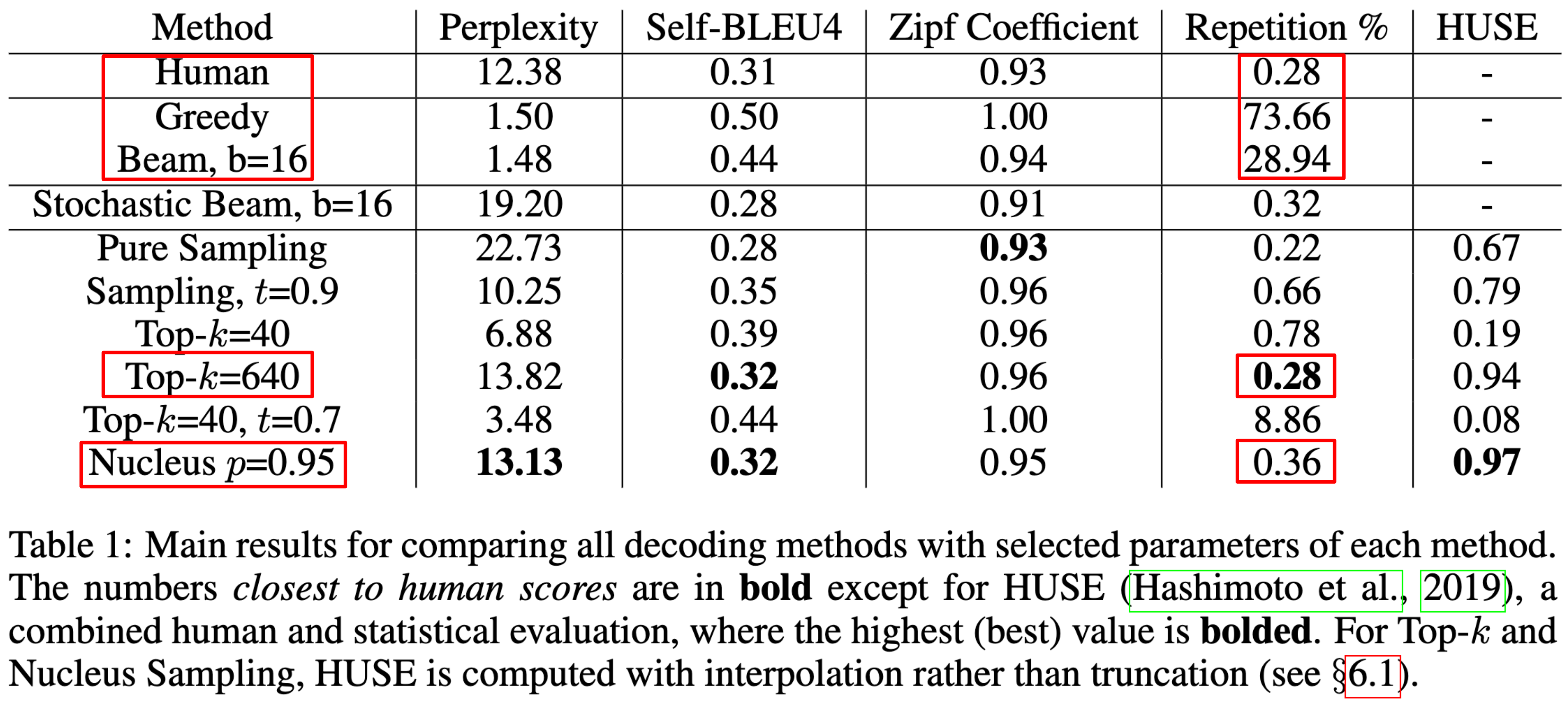
Temperature 低的時候 (less diverse)





Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations (ICLR). 2020.

Top-k and Top-p Sampling Improve Repetition

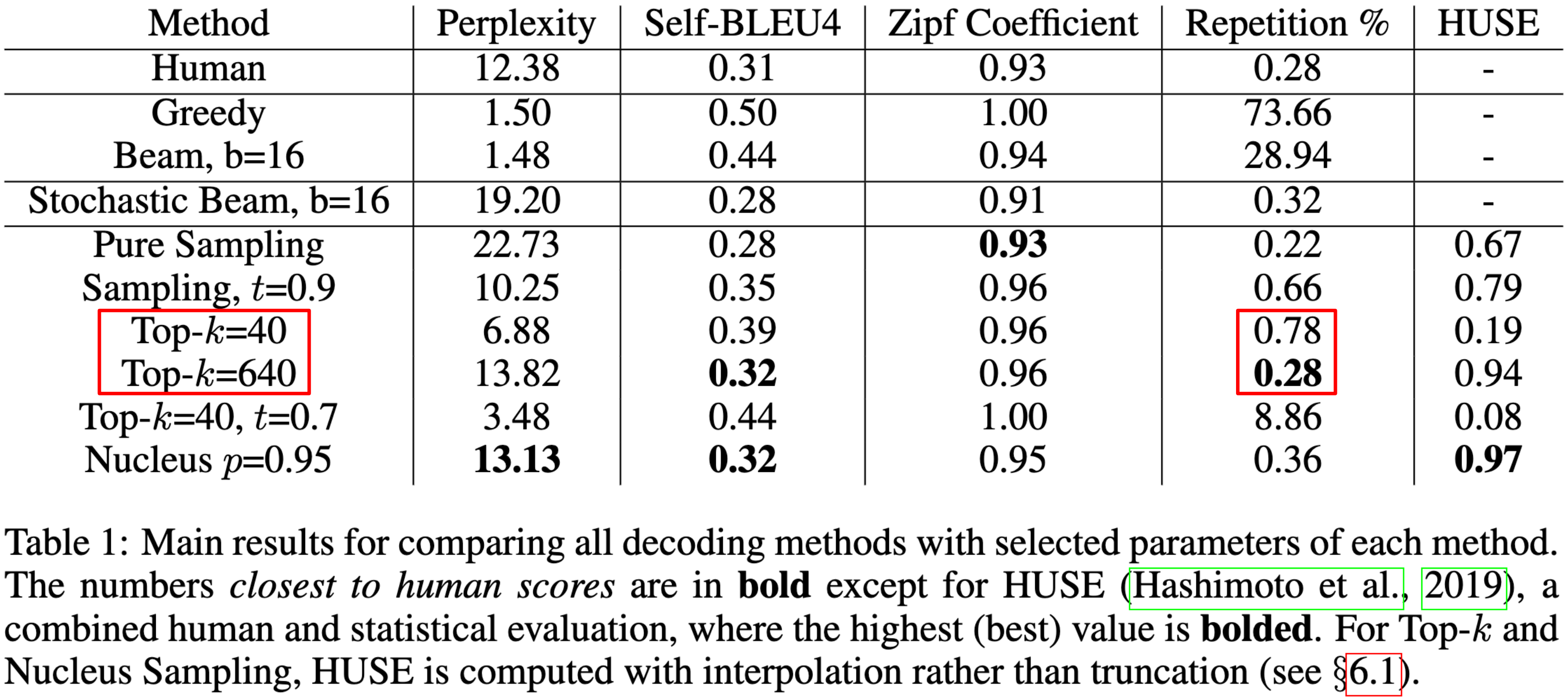


Method	Perplexity	Self-BLEU4	Zipf Coefficient	Repetition %	HUSE
Human	12.38	0.31	0.93	0.28	-
Greedy	1.50	0.50	1.00	73.66	-
Beam, b=16	1.48	0.44	0.94	28.94	-
Stochastic Beam, b=16	19.20	0.28	0.91	0.32	-
Pure Sampling	22.73	0.28	**0.93**	0.22	0.67
Sampling, t=0.9	10.25	0.35	0.96	0.66	0.79
Top-k=40	6.88	0.39	0.96	0.78	0.19
Top-k=640	13.82	**0.32**	0.96	**0.28**	0.94
Top-k=40, t=0.7	3.48	0.44	1.00	8.86	0.08
Nucleus p=0.95	**13.13**	**0.32**	0.95	0.36	**0.97**

Table 1: Main results for comparing all decoding methods with selected parameters of each method. The numbers *closest to human scores* are in **bold** except for HUSE (Hashimoto et al., 2019), a combined human and statistical evaluation, where the highest (best) value is **bolded**. For Top-k and Nucleus Sampling, HUSE is computed with interpolation rather than truncation (see §6.1).

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations. 2019.

The k value should be carefully picked



Method	Perplexity	Self-BLEU4	Zipf Coefficient	Repetition %	HUSE
Human	12.38	0.31	0.93	0.28	-
Greedy	1.50	0.50	1.00	73.66	-
Beam, b=16	1.48	0.44	0.94	28.94	-
Stochastic Beam, b=16	19.20	0.28	0.91	0.32	-
Pure Sampling	22.73	0.28	**0.93**	0.22	0.67
Sampling, t=0.9	10.25	0.35	0.96	0.66	0.79
Top-k=40	6.88	0.39	0.96	0.78	0.19
Top-k=640	13.82	**0.32**	0.96	**0.28**	0.94
Top-k=40, t=0.7	3.48	0.44	1.00	8.86	0.08
Nucleus p=0.95	**13.13**	**0.32**	0.95	0.36	**0.97**

Table 1: Main results for comparing all decoding methods with selected parameters of each method. The numbers *closest to human scores* are in **bold** except for HUSE (Hashimoto et al., 2019), a combined human and statistical evaluation, where the highest (best) value is **bolded**. For Top-k and Nucleus Sampling, HUSE is computed with interpolation rather than truncation (see §6.1).

Holtzman, Ari, et al. "The Curious Case of Neural Text Degeneration." International Conference on Learning Representations. 2019.

Comparison for Top-k and Top-p Sampling

	Top-k Sampling	Top-p Sampling
Hyperparameter	k: top-k words are preserved	p: sum of the minimum set of words exceeds the value of p
Performance (Who is better?)	By cases (these two are both widely used)	
Hyperparameter Tuning	Harder	Easier
Common Hyperparameter Value	k=40	p=0.95

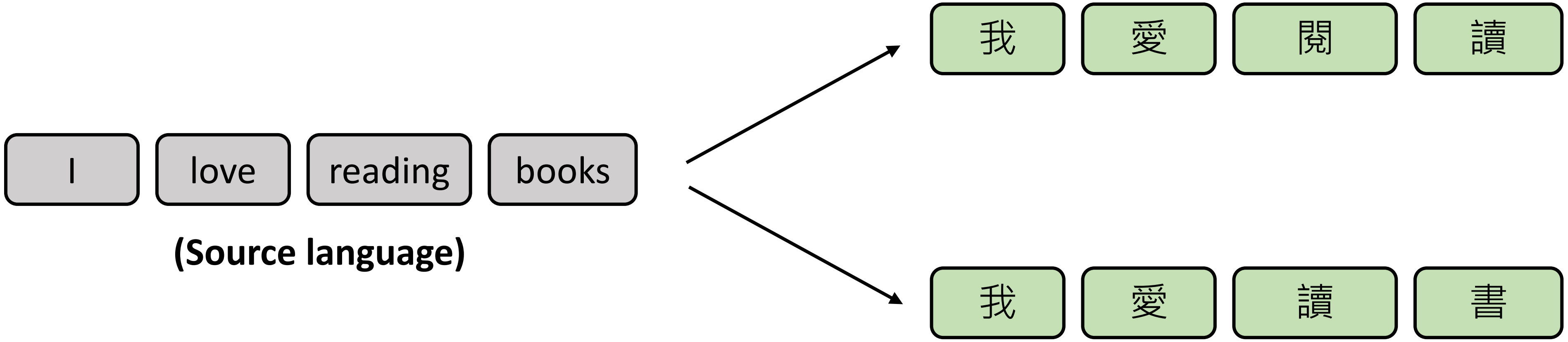
Evaluations

- Perplexity
- BLEU Score
- ROUGE Score
- BERTScore
- BLEURT

How to evaluate natural language generation?

- Natural language is hard to evaluate due to subjectivity and language diversity.

For example: Machine Translation



(Source language)

(Target language)

- Human evaluations
- Automatic evaluations (We will focus on this topic.)

(Recap) Perplexity

Perplexity (PPL) is a quantitative criterion used to evaluate the capacities of language modeling models. -> How confident is a model to its output?

- Given the sequence of words $W = w_1 w_2 ... w_N$ and an N-gram model. The PPL of the model was computed by:

$$Perplexity(W) = P(w_1 w_2 ... w_N)^{-\frac{1}{N}} = \sqrt[N]{\prod_{k=1}^{n} \frac{1}{P(w_k | w_{k-N+1:k})}}$$

The lower the value of perplexity, the better the language modeling capability of the model.

BLEU (Bilingual Evaluation Understudy)

- A word-based metric.
  - It is very sensitive to word tokenization
- Core concept: Compute precision for n-grams:
  - Unigrams -> BLEU-1
  - Bigrams -> BLEU-2
  - Trigrams -> BLEU-3
  - 4-grams -> BLEU-4

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Precision and Recall

$$\text{Precision} = \frac{\text{Relevant and retrieved instances}}{\text{All retrieved instances}} \longleftarrow \text{Predicted by a model}$$

$$\text{Recall} = \frac{\text{Relevant and retrieved instances}}{\text{All relevant instances}} \longleftarrow \text{Ground-truths}$$

Relevant and retrieved instances: Intersection between predictions and ground-truths

Calculation of BLEU Score (Example)

Assume we now translate from Chinese to English.

Calculate BLEU-1 score

Chinese: 我想要讀那本書

Reference1: I want to read the book.

Reference2: I want to read that book.

Model output: the the the the the the.

Calculation of BLEU Score (Example)

Assume we now translate from Chinese to English.

Calculate BLEU-1 score

Chinese: 我想要讀那本書

Reference1: I want to read the book.

Reference2: I want to read that book.

Model output: the the the the the the.

Precision: $\frac{6}{6}$

100%! Can this be true?

Calculation of BLEU Score (Example)

Assume we now translate from Chinese to English.

Calculate BLEU-1 score

Chinese: 我想要讀那本書

Reference1: I want to read the book.

Reference2: I want to read that book.

Model output: the the the the the the.

~~Precision: $\frac{6}{6}$~~

Modified Precision: $\frac{1}{6}$

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Why should we use modified precision?

- The output sequences can be total mistakes.
  - E.g., the the the the the the
- Original precision is in favor of longer output sequences.
- Therefore, we should use modified precision to prevent bad evaluations.

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

← More than one references can be provided for machine translation!

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

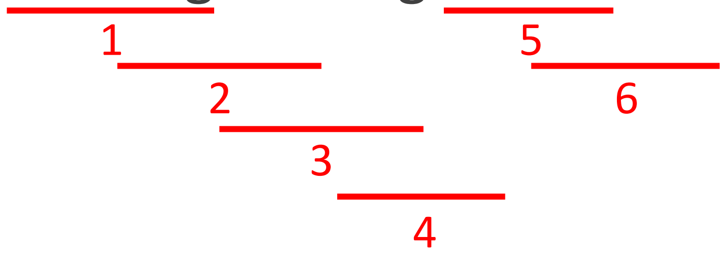
Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.



	Count	
the dog	2	(duplicated)
dog the	1	
dog on	1	
on the	1	
the bed	1	

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

Clips to the reference

	Count	Count_{clip}
the dog	2	1
dog the	1	
dog on	1	
on the	1	
the bed	1	

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

	Count	Count$_{clip}$
the dog	2	1
dog the	1	0
dog on	1	
on the	1	
the bed	1	

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

	Count	Count_{clip}
the dog	2	1
dog the	1	0
dog on	1	1
on the	1	
the bed	1	

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

Count only one time even mapped to both references.

	Count	Count_{clip}
the dog	2	1
dog the	1	0
dog on	1	1
on the	1	1
the bed	1	

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Calculation of BLEU Score (Example)

Calculate BLEU-2 score

Reference1: The dog is on the bed.

Reference2: There is a dog on the bed.

Model output: The dog the dog on the bed.

Count only one time even mapped to both references.

	Count	$\text{Count}_{\text{clip}}$
the dog	2	1
dog the	1	0
dog on	1	1
on the	1	1
the bed	1	1

Modified Precision: $\frac{4}{6}$

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Formula of BLEU Score (1)

Summation for unigram, bigram, tri-gram, and 4-gram

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

Summation for all candidates (model outputs) of each translation

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Formula of BLEU Score (2)

Summation for unigram, bigram, tri-gram, and 4-gram

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

Summation for all candidates (model outputs) of each translation

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

What we’ve learned BLEU so far

- The BLEU score is calculated from the summation of 1-gram to 4-gram.
  - You can also measure n-gram individually.
- We use modified precision to prevent bad evaluations.
- What will happen if a model tends to generate really short sentences?

➡ **More penalty for calculating BLEU score!**

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation."
Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

Brevity Penalty (BP)

- BP is used to penalize short candidates.

c: The length of a candidate sequence
r: The length of a reference sequence that is closest to c (shorter one)

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

Then,

$$\textsc{Bleu} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

N=4 to include 1-gram to 4-gram

Weight for each n-gram (was set 1/4 in the original paper)

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

ROUGE Score

- ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
- Mainly for text summarization
- Metric Input: Summary (prediction), Reference (gold summary)
- Common metrics: ROUGE-1, ROUGE-2, ROUGE-L
    - L: Longest common subsequence
- Please note that current papers calculate ROUGE-F as default!!!
    - In other words, ROUGE-1F, ROUGE-2F, ROUGE-LF

Lin, Chin-Yew. "Rouge: A package for automatic evaluation of summaries." Text summarization branches out. 2004.

ROUGE-1 Example

predictions = ["The", "cat", "sat", "on", "the", "mat"]

references = ["A", "cat", "was", "sitting", "on", "the", "mat"]

ROUGE-1 recall = Number of matching unigrams / Number of unigrams in the reference = 4/7

ROUGE-1 precision = Number of matching unigrams / Number of unigrams in the machine-generated summary = 4/6

ROUGE-1 F1-score = Harmonic mean of the precision and the recall = 2 * 4/7 * 4/6 / (4/7 + 4/6)

ROUGE-2 Example

predictions = ["The cat", "cat sat", "sat on", "on the", "the mat"]

references = ["A cat", "cat was", "was sitting", "sitting on", "on the", "the mat"]

ROUGE-2 recall = Number of matching bigrams / Number of bigrams in the reference = 2/6

ROUGE-2 precision = Number of matching bigrams / Number of bigrams in the machine-generated summary = 2/5

ROUGE-2 F1-score = Harmonic mean of the precision and the recall = 2 * 2/6 * 2/5 / (2/6 + 2/5)

ROUGE-L Example

predictions = ["The", "cat", "sat", "on", "the", "mat"]

references = ["A", "cat", "was", "sitting", "on", "the", "mat"]

The longest common subsequence is ["cat", "on", "the", "mat"]

ROUGE-L recall = Number of matching unigrams / Number of unigrams in the reference = 4/7

ROUGE-L precision = Number of matching unigrams / Number of unigrams in the machine-generated summary = 4/6

ROUGE-L F1-score = Harmonic mean of the precision and the recall = 2 * 4/7 * 4/6 / (4/7 + 4/6)

ROUGE-L Example

The order should be kept for the LCS problem

predictions = ["The", "cat", "sat", "on", "the", "mat"]

references = ["on", "the", "mat”, "sitting", ”a", "cat"]

The longest common subsequence is ["on", "the", "mat"]

ROUGE-L recall = Number of matching unigrams / Number of unigrams in the reference = 3/6

ROUGE-L precision = Number of matching unigrams / Number of unigrams in the machine-generated summary = 3/6

ROUGE-L F1-score = Harmonic mean of the precision and the recall = 2 * 0.5 * 0.5 / (0.5 + 0.5)

Why do we need BLEU and ROUGE?

- BLEU is mainly designed for machine translation.
  - For example, the Brevity Penalty.
- ROUGE measures the overlapping between predicted and gold summaries.
- Can we just use one of them?
  - Conventionally, no.
  - Different tasks are evaluated with different metrics.

Comparison for Human and Automatic Evaluations (e.g., BLEU and ROUGE)

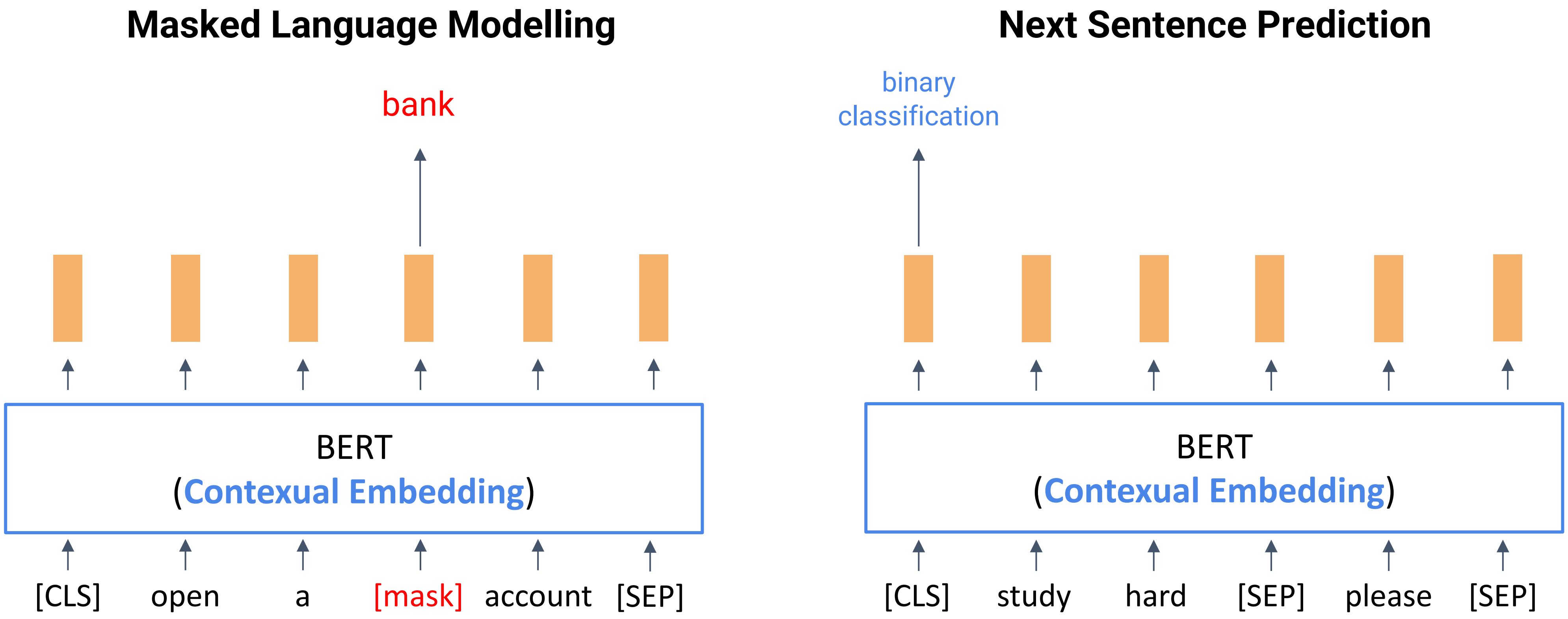
- Human evaluations
  - Pros: More accurate for subjectivity, flexibility for any desired comparison
  - Cons: Less objective, time-consuming, expensive
- Automatic evaluations
  - Pros: Objective enough to serve as common evaluation metrics, fast
  - Cons: Cannot meet language diversity
    - Take machine translation for instance, there are always other valid ways to translate the source sentence.

Issue of BLEU and ROUGE

- Cons: Cannot meet language diversity
  - This mainly comes from the way for measuring overlapping rates.
- **Question**: Can we create an automatic metric to fix the issue?

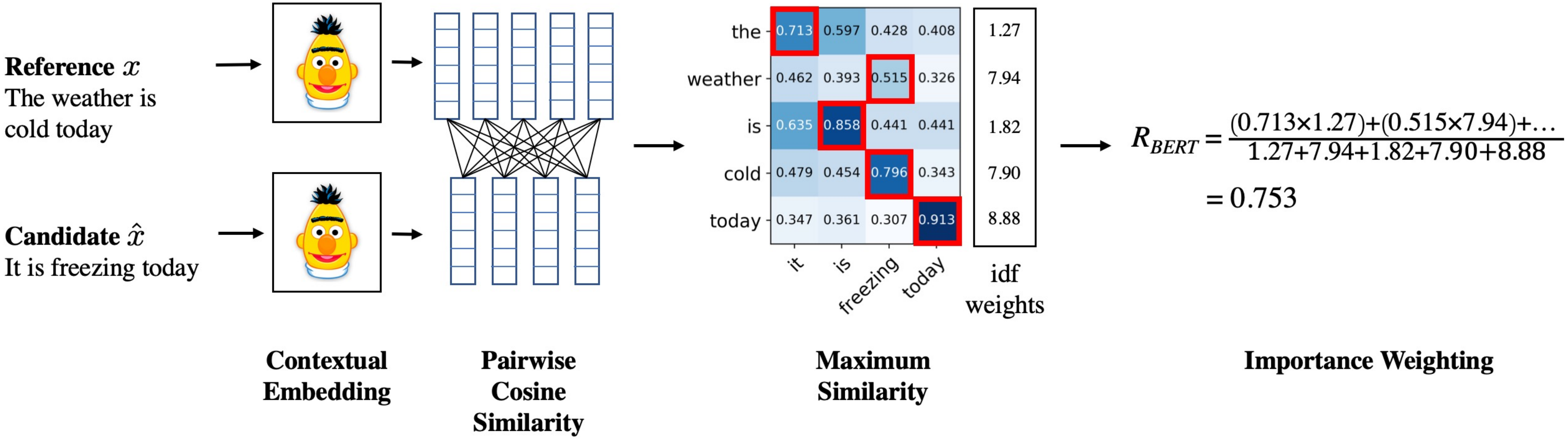
- Next, we are going to introduce two learned automatic evaluation metrics
  - BERTScore (ICLR 2020)
  - BLEURT (ACL 2020)

(Recap) BERT: Bidirectional Encoder Representations from Transformers



Devlin, Jacob, et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL 2019.

BERTScore – Overview



	it	is	freezing	today	idf weights
the	0.713	0.597	0.428	0.408	1.27
weather	0.462	0.393	0.515	0.326	7.94
is	0.635	0.858	0.441	0.441	1.82
cold	0.479	0.454	0.796	0.343	7.90
today	0.347	0.361	0.307	0.913	8.88

$$R_{BERT} = \frac{(0.713\times1.27)+(0.515\times7.94)+\ldots}{1.27+7.94+1.82+7.90+8.88}$$
$$= 0.753$$

Zhang, Tianyi, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations. 2020.

BERTScore – Steps

Step 0: Prepare Reference x, Candidate $\hat{x}$, and a pre-trained BERT model

Step 1: Infer x and $\hat{x}$ with BERT respectively, get a sequence of output vectors $\langle \mathrm{x}_1, \ldots, \mathrm{x}_k \rangle$ for x and a sequence of output vectors $\langle \hat{\mathrm{x}}_1, \ldots, \hat{\mathrm{x}}_k \rangle$ for $\hat{x}$

Zhang, Tianyi, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations. 2020.

BERTScore – Steps

Zhang, Tianyi, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations. 2020.

Step 0: Prepare Reference x, Candidate $\hat{x}$, and a pre-trained BERT model

Step 1: Infer x and $\hat{x}$ with BERT respectively, get a sequence of output vectors $\langle \mathrm{x}_1, \dots, \mathrm{x}_k \rangle$ for x and a sequence of output vectors $\langle \hat{\mathrm{x}}_1, \dots, \hat{\mathrm{x}}_k \rangle$ for $\hat{x}$

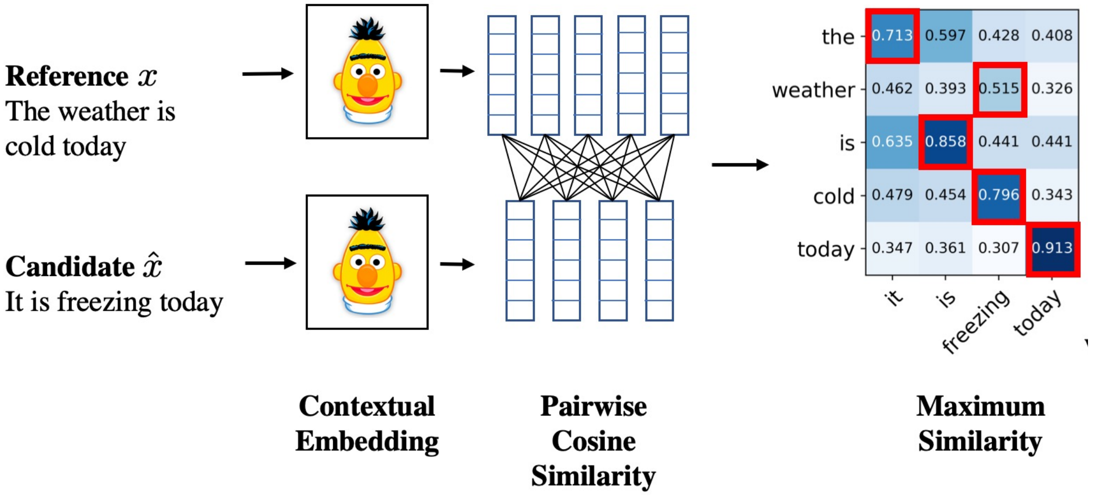
Step 2: Measure pairwise cosine similarity

Recall $$R_{\mathrm{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\boxed{\hat{x}_j \in \hat{x}}} \mathbf{x}_i^{\top} \hat{\mathbf{x}}_j$$

Based on reference

Precision $$P_{\mathrm{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{\boxed{x_i \in x}} \mathbf{x}_i^{\top} \hat{\mathbf{x}}_j$$

Based on candidate



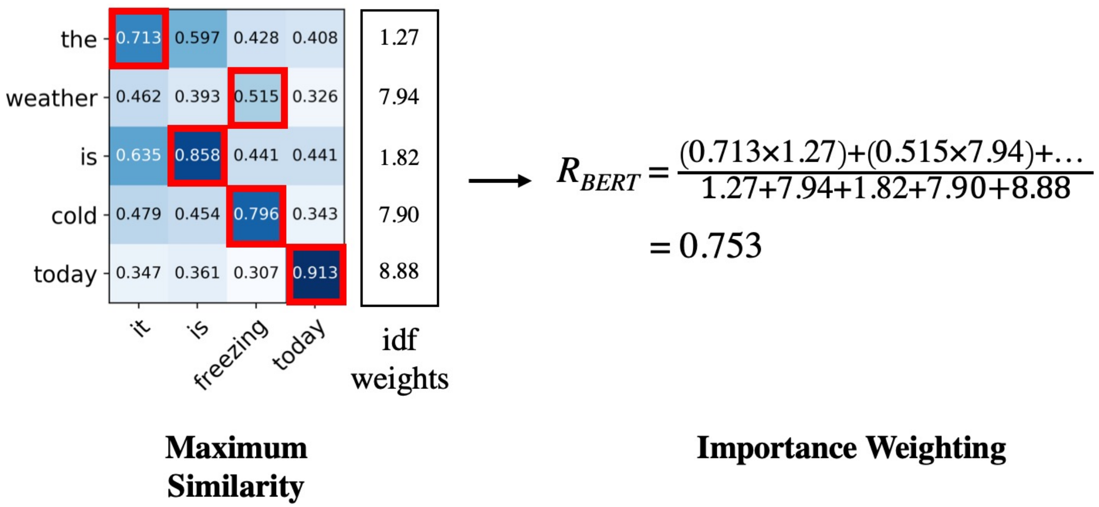
	it	is	freezing	today
the	0.713	0.597	0.428	0.408
weather	0.462	0.393	0.515	0.326
is	0.635	0.858	0.441	0.441
cold	0.479	0.454	0.796	0.343
today	0.347	0.361	0.307	0.913

BERTScore – Importance Weighting

Given M reference sentences $\{x^{(i)}\}_{i=1}^{M}$, the idf (inverse document frequency) score of a word-piece token w is:

$$\mathrm{Idf}(w) = -\log \frac{1}{M} \sum_{i=1}^{M} \mathbb{I}\left[w \in x^{(i)}\right]$$

$$R_{\mathrm{BERT}} = \frac{\sum_{x_i \in x} \mathrm{idf}(x_i) \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^{\top} \hat{\mathbf{x}}_j}{\sum_{x_i \in x} \mathrm{idf}(x_i)}$$



	it	is	freezing	today	idf weights
the	0.713	0.597	0.428	0.408	1.27
weather	0.462	0.393	0.515	0.326	7.94
is	0.635	0.858	0.441	0.441	1.82
cold	0.479	0.454	0.796	0.343	7.90
today	0.347	0.361	0.307	0.913	8.88

$$\longrightarrow R_{BERT} = \frac{(0.713\times1.27)+(0.515\times7.94)+\ldots}{1.27+7.94+1.82+7.90+8.88} = 0.753$$

Maximum Similarity

Importance Weighting

Zhang, Tianyi, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations. 2020.

Summary of BERTScore

- BERTScore leverages the contextual representation abilities of BERT to measure the semantic similarities between a reference and a candidate.
- In the paper, BERTScore correlates better with human judgments and provides stronger model selection performance than existing metrics.
- However, BERTScore does not involve training process.

Can we train BERT for a better evaluation metric?

Zhang, Tianyi, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations. 2020.

BLEURT – Quick Introduction

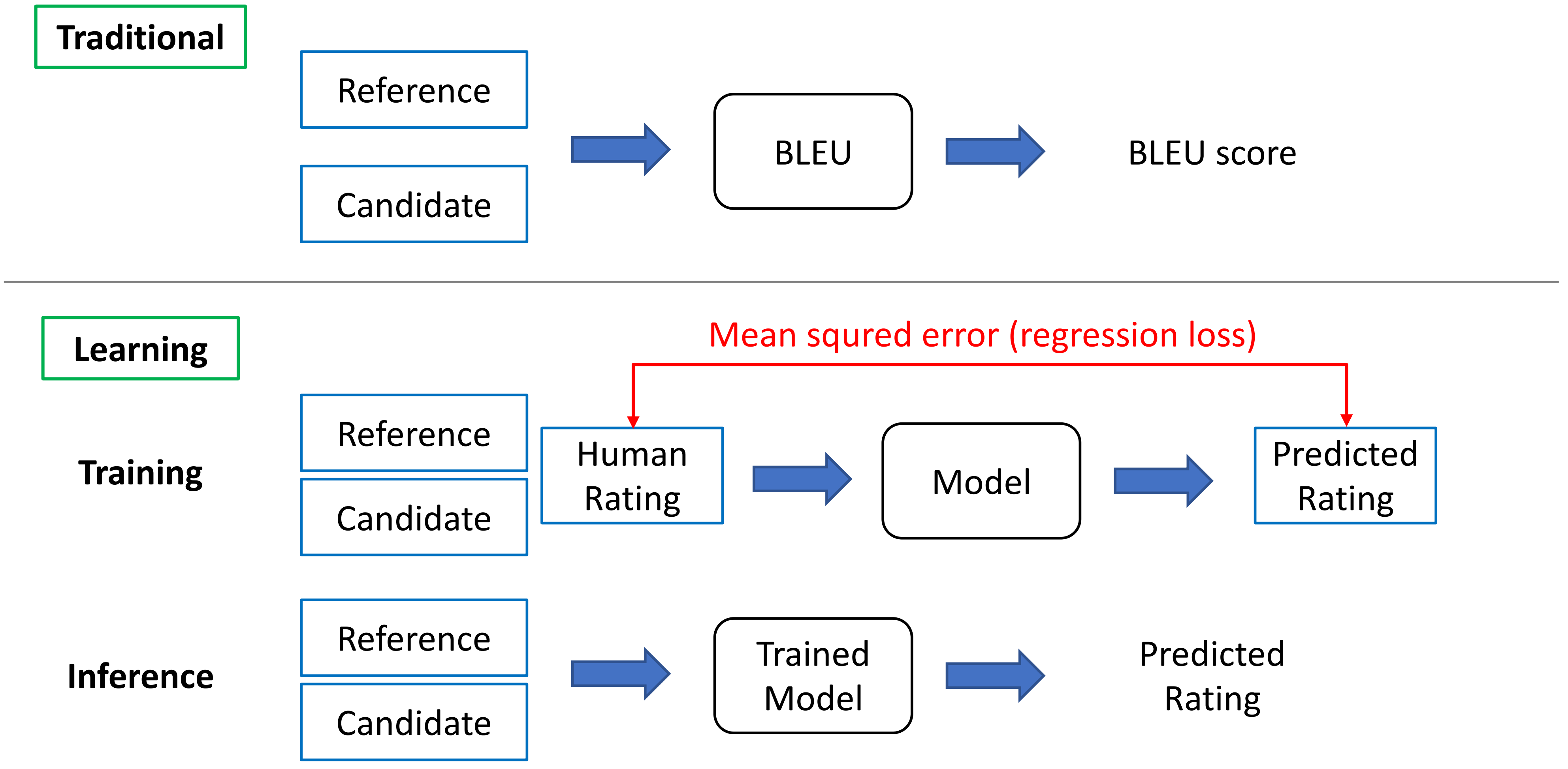
- BLEURT: Learning Robust Metrics for Text Generation, published by Google
- BLEURT trains BERT for a more robust evaluation metric.
  - Mainly for machine translation.
    - Also get hints from the name “BLEURT”
  - Trained checkpoint can be obtained. We don’t need to perform training.

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. "BLEURT: Learning Robust Metrics for Text Generation." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

BLEURT – Motivations

- Learned metrics can be tuned to measure task-specific properties, such as fluency, faithfulness, grammar, or style.
- NLG systems tend to get better over time, and therefore a model trained on ratings data from 2015 may fail to distinguish top performing systems in 2019, especially for newer research tasks.

Training on Human Ratings



BLEURT – Steps

Step 0: Reference-candidate pairs $(z, \tilde{z})$ and the pre-trained BERT model

Step 1: Data augmentation for $(z, \tilde{z})$ to to perform pre-training

Data augmentation strategies

- Random masking
- Back-translation
- Dropping words randomly

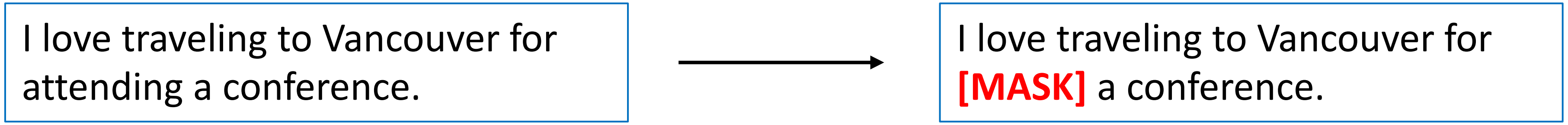
Total 6.5 million variants of $(z, \tilde{z})$ were created.

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. "BLEURT: Learning Robust Metrics for Text Generation." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

Random masking

Two kinds of masking strategies were adopted:

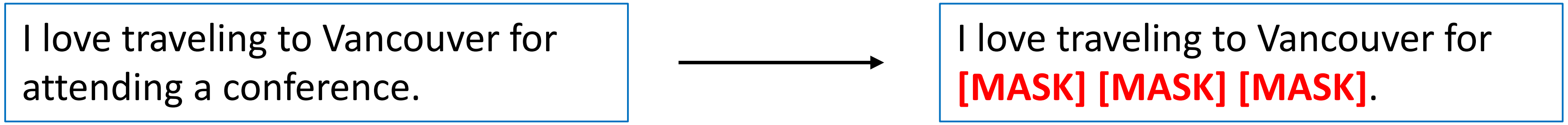
Token masking



I love traveling to Vancouver for attending a conference.

I love traveling to Vancouver for [MASK] a conference.

Span masking

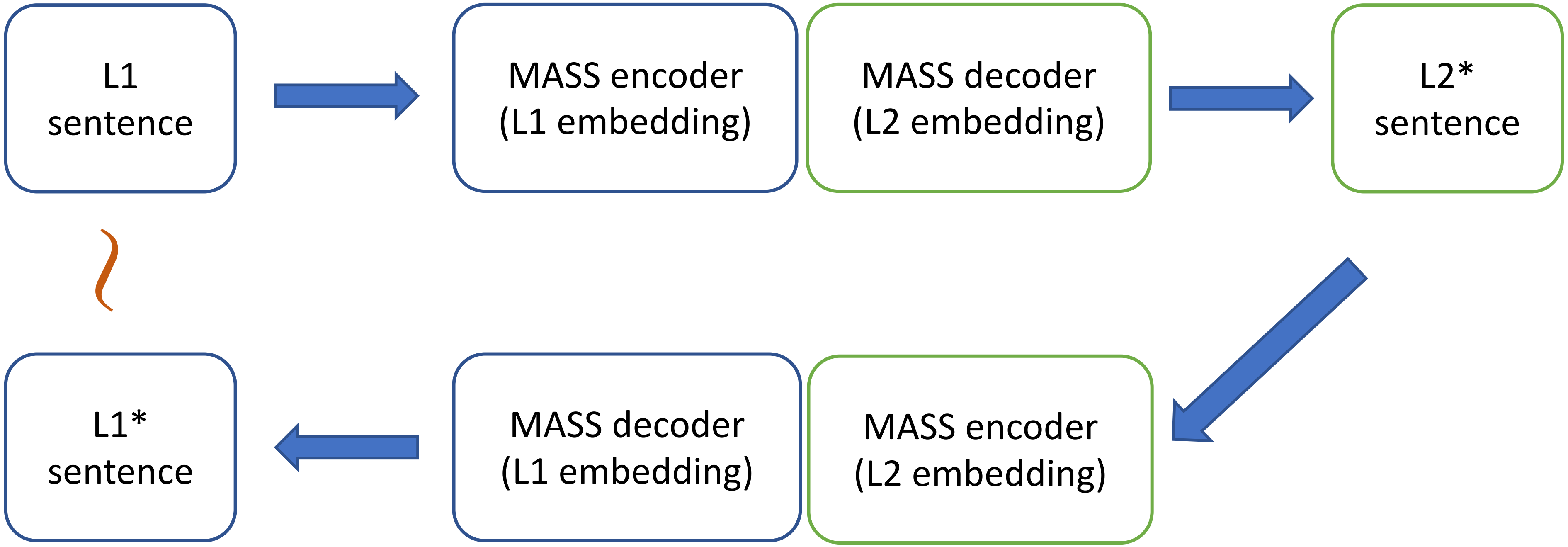


I love traveling to Vancouver for attending a conference.

I love traveling to Vancouver for [MASK] [MASK] [MASK].

Backtranslation

- L1: English; L2: French or German



L1 sentence

MASS encoder (L1 embedding)

MASS decoder (L2 embedding)

L2* sentence

MASS encoder (L2 embedding)

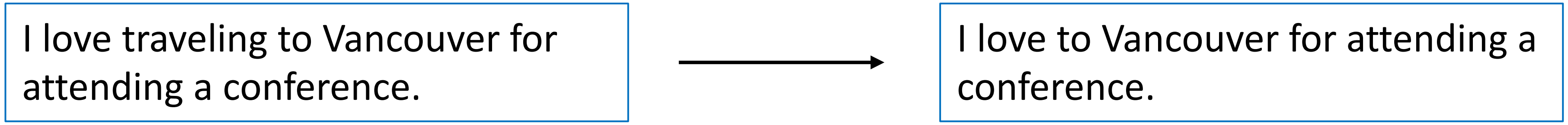
MASS decoder (L1 embedding)

L1* sentence

≀

Dropping words randomly

- The authors found it useful in their experiments to randomly drop words to create other examples.



I love traveling to Vancouver for attending a conference.

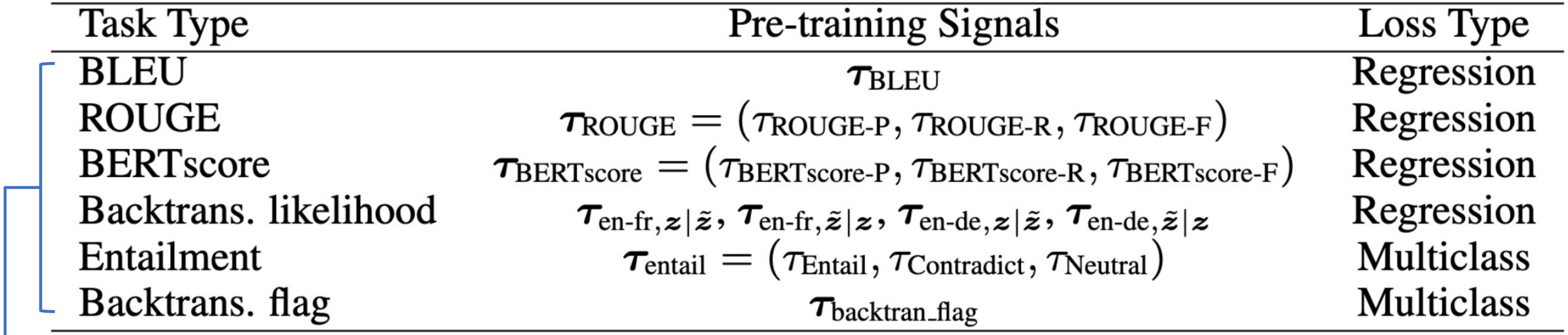
I love to Vancouver for attending a conference.

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. "BLEURT: Learning Robust Metrics for Text Generation." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

BLEURT – Step 3

Step 3: Pre-training each sentence pair $(z, \tilde{z})$ with the following tasks.

Note that this is not conventional BERT pre-training! It is multi-task pre-training!



Task Type	Pre-training Signals	Loss Type
BLEU	τ_{BLEU}	Regression
ROUGE	$\tau_{\text{ROUGE}} = (\tau_{\text{ROUGE-P}}, \tau_{\text{ROUGE-R}}, \tau_{\text{ROUGE-F}})$	Regression
BERTscore	$\tau_{\text{BERTscore}} = (\tau_{\text{BERTscore-P}}, \tau_{\text{BERTscore-R}}, \tau_{\text{BERTscore-F}})$	Regression
Backtrans. likelihood	$\tau_{\text{en-fr},\boldsymbol{z}\mid\tilde{\boldsymbol{z}}}, \tau_{\text{en-fr},\tilde{\boldsymbol{z}}\mid\boldsymbol{z}}, \tau_{\text{en-de},\boldsymbol{z}\mid\tilde{\boldsymbol{z}}}, \tau_{\text{en-de},\tilde{\boldsymbol{z}}\mid\boldsymbol{z}}$	Regression
Entailment	$\tau_{\text{entail}} = (\tau_{\text{Entail}}, \tau_{\text{Contradict}}, \tau_{\text{Neutral}})$	Multiclass
Backtrans. flag	$\tau_{\text{backtran\_flag}}$	Multiclass

- Ground-truth values can be computed for each $(z, \tilde{z})$ pair!
- Losses for the six tasks were sum up during pre-training.

- Regression: mean squared error
- Multiclass: Cross-entropy

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. "BLEURT: Learning Robust Metrics for Text Generation." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

Task 4: Backtranslation Likelihood

- Existing translation models (trained) are needed.
  - Transformers (Vaswani et al., 2017): EN-FR
  - Transformers (Vaswani et al., 2017): DE-EN
- Equations use EN-FR for an example

$P(x_t|x_1, \dots, x_{t-1}, z)$

$$\boldsymbol{z}^*_{\mathtt{fr}} = \arg\max P_{\mathtt{en}\rightarrow\mathtt{fr}}(\boldsymbol{z}_{\mathtt{fr}}|\boldsymbol{z})$$

Best translated French sentence (details absent in the paper)

$$P(\tilde{\boldsymbol{z}}|\boldsymbol{z}) \approx P_{\mathtt{fr}\rightarrow\mathtt{en}}(\tilde{\boldsymbol{z}}|\boldsymbol{z}^*_{\mathtt{fr}})$$

Backtranslation Likelihood

Task 5 and Task 6

- **Textual Entailment**
  - We report the probability of three labels: Entail, Contradict, and Neutral, using BERT fine-tuned on the MNLI dataset.
- **Backtranslation flag**
  - A Boolean that indicates whether the perturbation was generated with backtranslation or with mask-filling.

BLEURT – Final Step

Step 4: Fine-tune the model (trained from Step 3) on the <Reference, Candidate ,Rating> data using the regression loss

The <Reference, Candidate ,Rating> data include

- WMT (machine translation task)
- WebNLG (for general text generation)

Summary of BLEURT

- This approach uses (continual) pre-training and fine-tuning to create a learned evaluation metric for machine translation and general NLG.
- According to the paper, BLEURT is better aligned to human ratings then BERTScore.
- BLEURT should work for text summarization, but the authors did not test it.

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. "BLEURT: Learning Robust Metrics for Text Generation." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

Thank you!

Instructor: 林英嘉

✉ yjlin@cgu.edu.tw