CS769 Advanced NLP

Conditioned Generation

Junjie Hu

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Slides adapted from Graham, Sergey, Chris
https://junjiehu.github.io/cs769-spring23/

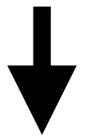
Goals for Today

- Formulation of Conditional Language Modeling
- **Decoding** Algorithms
- **Model Ensemble**
- Tasks and Evaluation

Language Models

- Language models are generative models of text

$$x \sim P(X)$$



"The Malfoys!" said Hermione.

Harry was watching him. He looked like Madame Maxime. When she strode up the wrong staircase to visit himself.

"I'm afraid I've definitely been suspended from power, no chance—indeed?" said Snape. He put his head back behind them and read groups as they crossed a corner and fluttered down onto their ink lamp, and picked up his spoon. The doorbell rang. It was a lot cleaner down in London.

Text Credit: Max Deutsch (https://medium.com/deep-writing/)

Conditioned Language Models

- Not just generate text, generate text according to some specification, i.e., $P(Y|X)$

Input *X*	Output *Y* (**Text**)	Task
Structured Data	NL Description	NL Generation
English	Japanese	Translation
Document	Short Description	Summarization
Utterance	Response	Response Generation
Image	Text	Image Captioning
Speech	Transcript	Speech Recognition

Formulation and Modeling

Calculating the Probability of a Sentence

$$P(X) = \prod_{i=1}^{I} P(x_i \mid x_1, \ldots, x_{i-1})$$

Next Word

Context

Conditional Text Generation Models

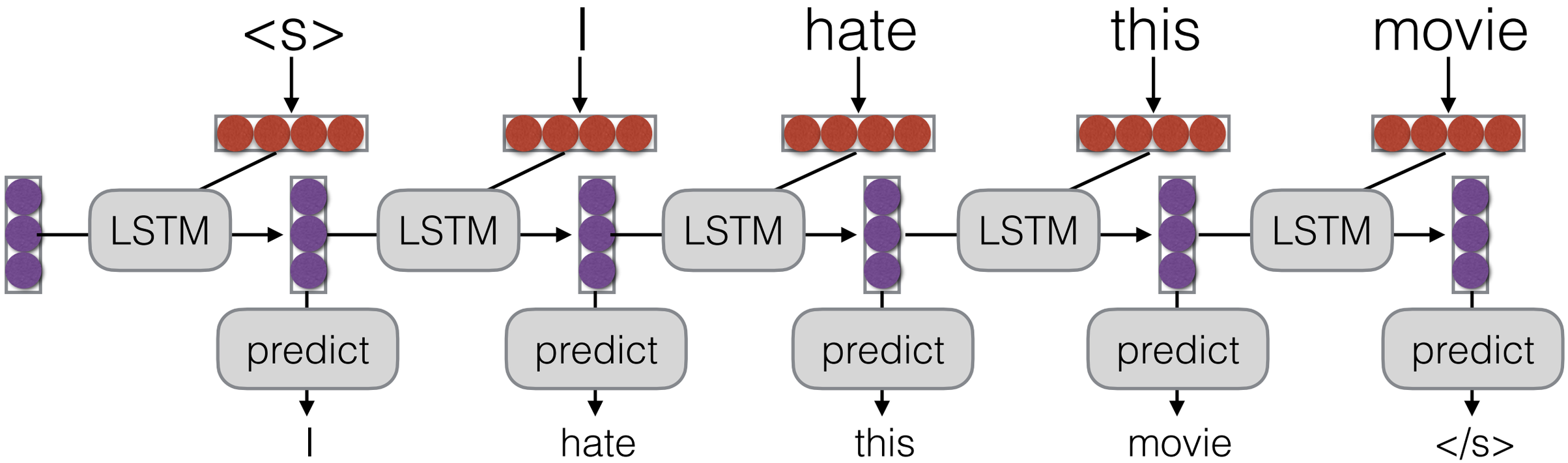
$$P(Y|X) = \prod_{j=0}^{J} P(y_j|X, y_0, \cdots, y_{j-1})$$

Added Context!

Sometimes we add some special tokens to *Y*:

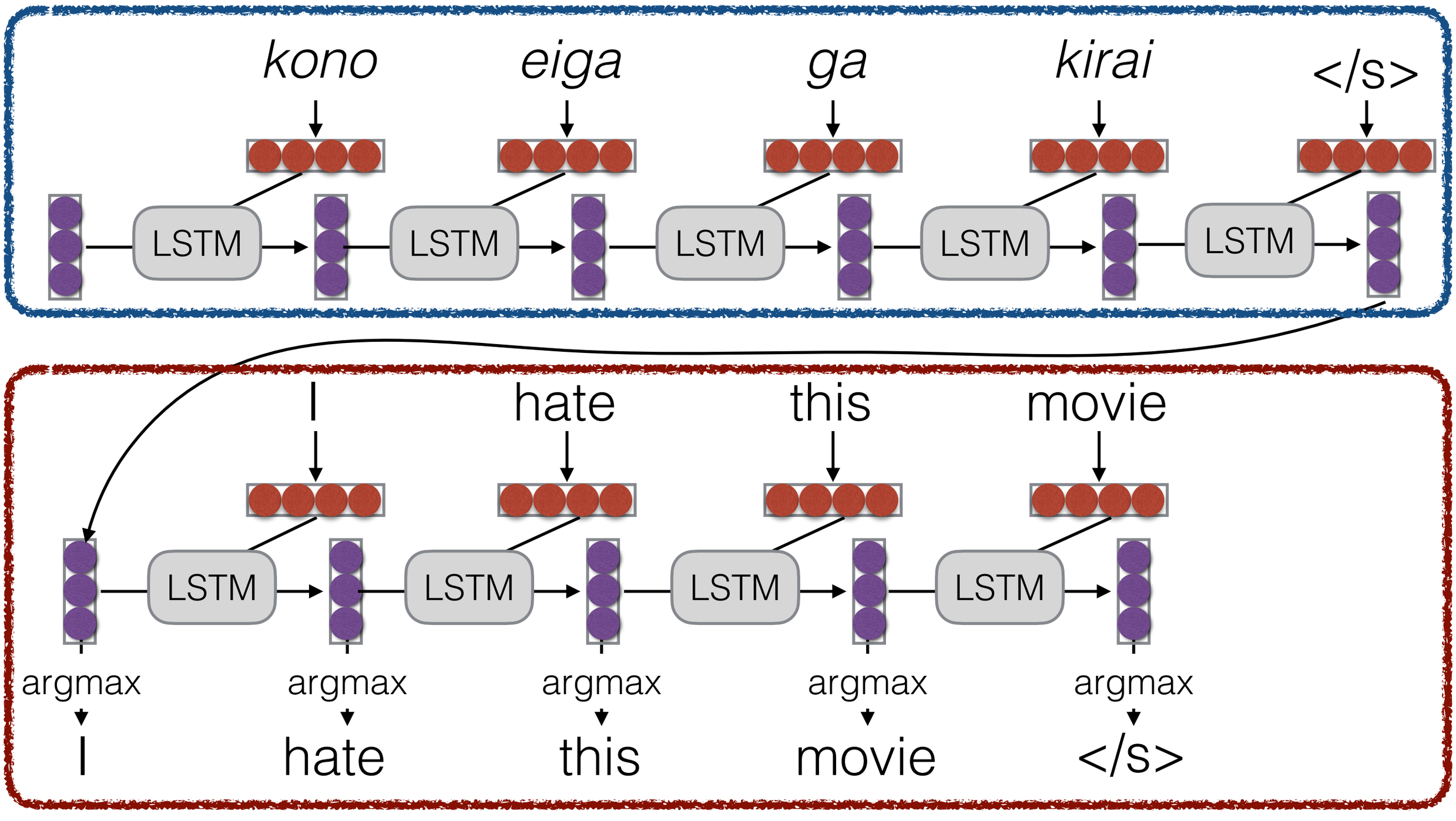
- y_0: start of sentence token, i.e., “[SOS]” or “<s>”
- y_J: end of sentence token, i.e., “[EOS]” or “</s>”

(One Type of) Language Model (Mikolov et al. 2011)



(One Type of) Conditional Language Model (Sutskever et al. 2014)

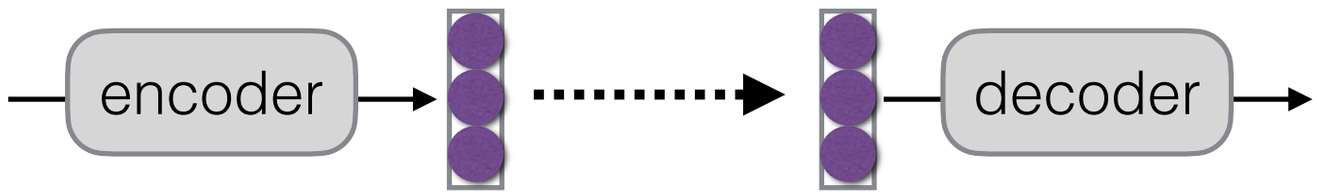
Encoder



Decoder

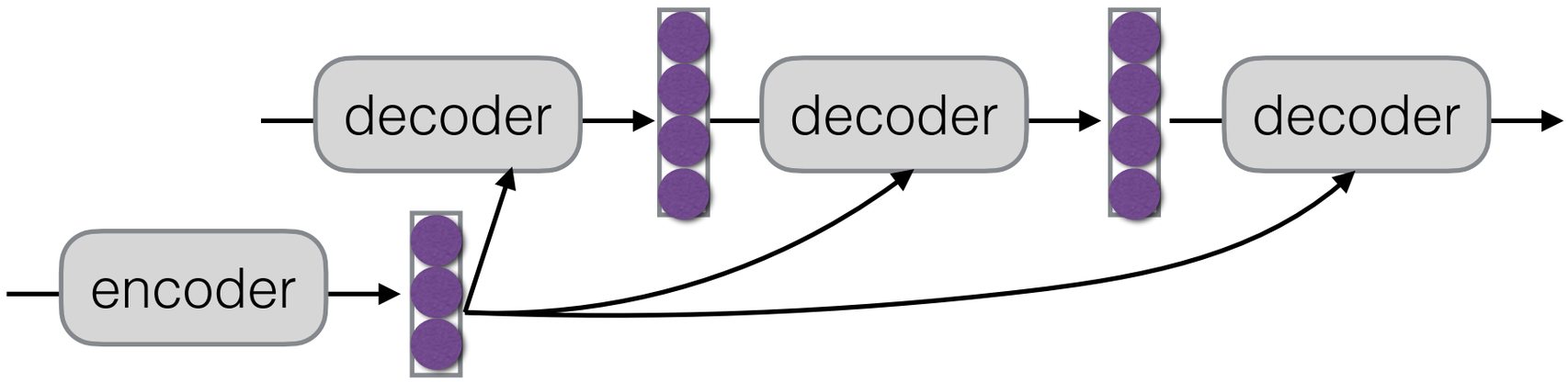
How to Pass Hidden State?

- **Initialize** decoder w/ **encoder hidden state** (Sutskever et al. 2014)



- **Transform** the encoder hidden state (can be different dimensions)

- Add encoder hidden state to each decoder time step (Kalchbrenner & Blunsom 2013)



- Use **attention** to selectively pick related inputs to each step

Decoding Methods for Text Generation

The Generation Problem

- We have a model of $P(Y|X)$, how do we use it to generate a sentence?

- Three methods:
  - **Sampling:** Try to generate a *random* sentence according to the probability distribution.
  - **Argmax:** Try to generate the sentence by taking one word with the *highest* score at each time step.
  - **Beam search:** Try to *approximately* generate the sentence with the *"highest"* overall score.

Sampling

- **Generate** words one-by-one by the conditioned probability at each time step

while y_{j-1} != “[EOS]”:

$$y_j \sim P(y_j|X, y_0, \cdots, y_{j-1})$$

- Namely, sampling a sentence $[y_0, \cdots, y_J] \sim P(Y|X)$

Greedy Search

- One by one, pick the single highest-probability word

while y_{j-1} != “[EOS]”:

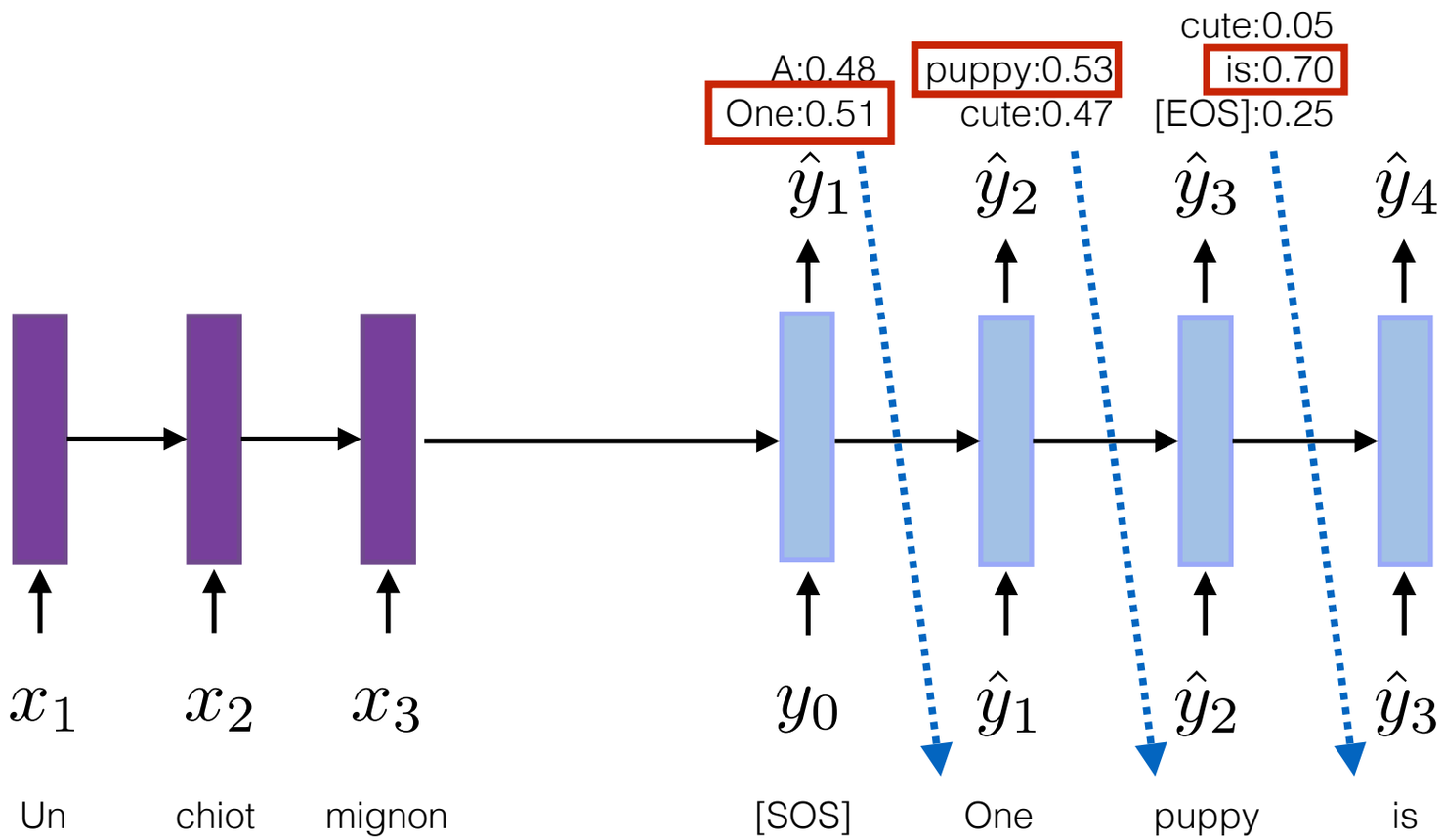
$$y_j = \arg\max P(y_j | X, y_0, \cdots, y_{j-1})$$

Problems:

- Will often generate the “easy” words first
- Will prefer multiple common words to one rare word
- May return a poor sentence that has low probabilities of words at the end.

Example: Greedy Search

- Decode the most likely sequence

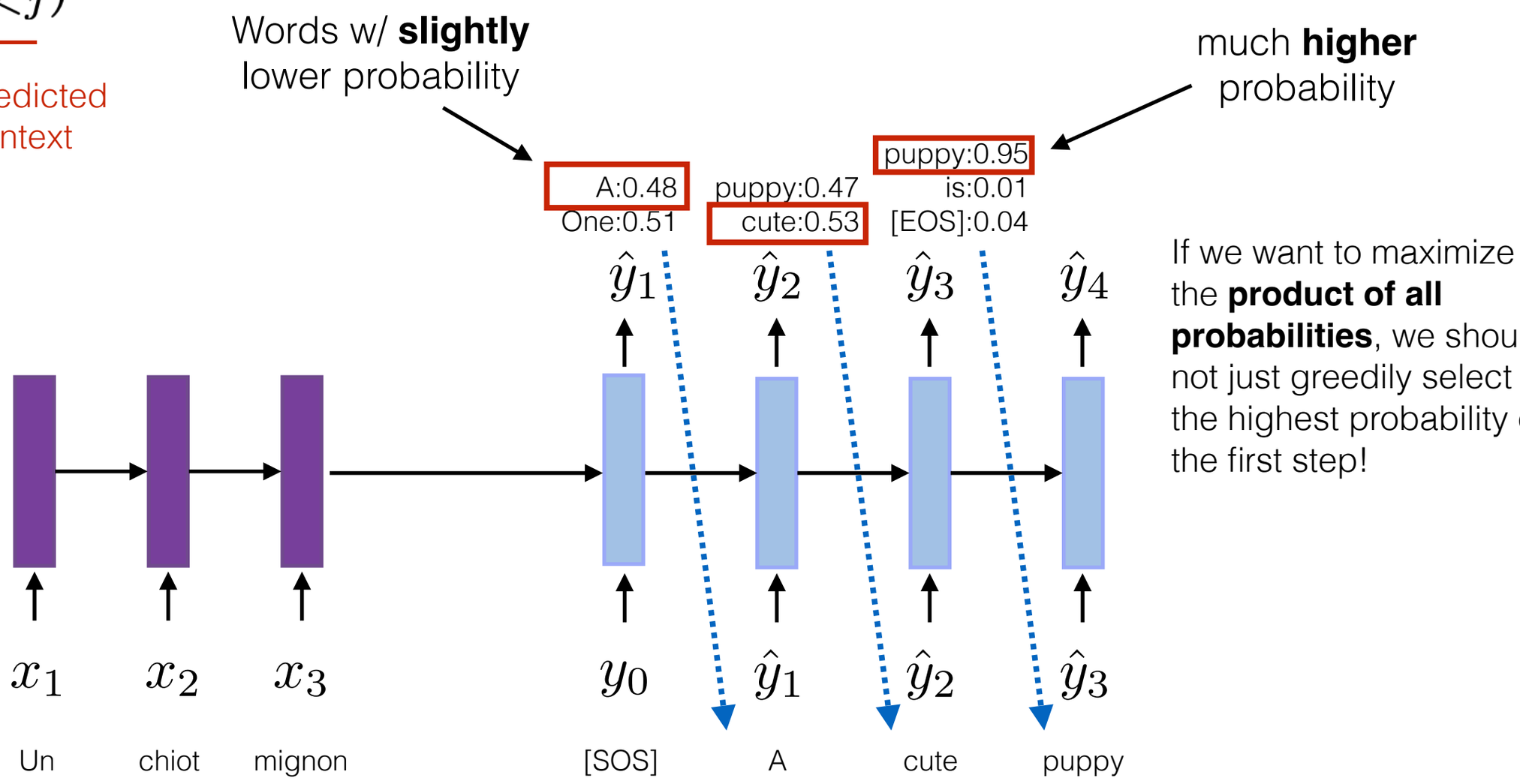


What if we choose words w/ lower probability?

Conditioned probability changes w/ **different predicted context**

$$P(y_j|X, y_{<j})$$

source input — predicted context

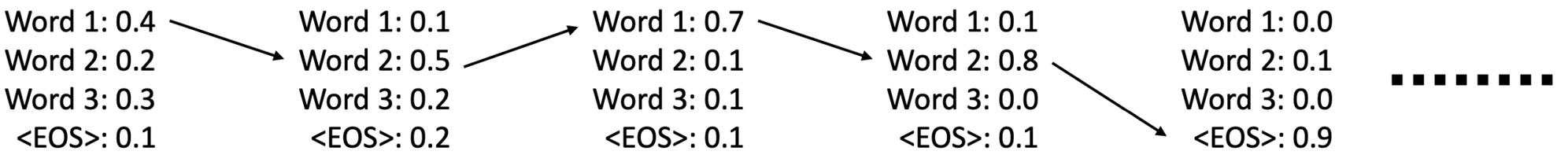


If we want to maximize the **product of all probabilities**, we should not just greedily select the highest probability on the first step!

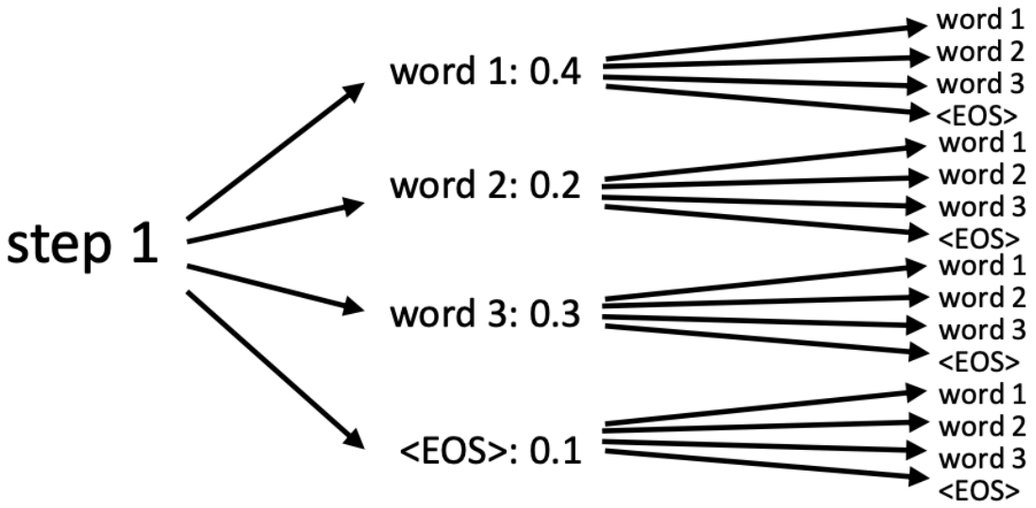
$$P(Y|X) = \prod_{j=0}^{J} P(y_j|X, y_0, \cdots, y_{j-1})$$

How many possible decodings are there?

- For a vocabulary of V words, there are V^J possible sequences of length J



Decoding is a tree search problem:



We could use any tree search algorithm

But exact search in this case is **very expensive**

Fortunately, the **structure** of this problem makes some simple **approximate search** methods work **very well**.

Decoding with approximate search

- **Basic intuition**: while choosing the **highest-probability** word on the first step may not be optimal, choosing a **very low-probability** word is very unlikely to lead to a good result.

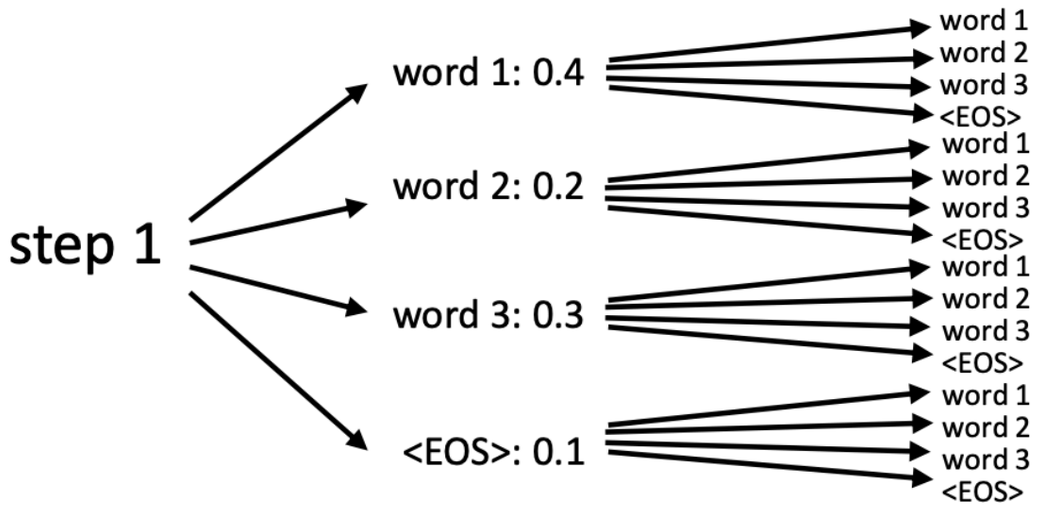
- **Equivalently**: we cannot be greedy at only one solution, but we can be somewhat greedy at multiple solutions.

- **This is not true in general!** This is a guess based on what we know about sequence decoding.

- **Beam search** intuition: store the **k** best sequences **so far**, and update each of them.

  - Special case of **k**=1 is greedy decoding

  - Often use **k** around 5-10

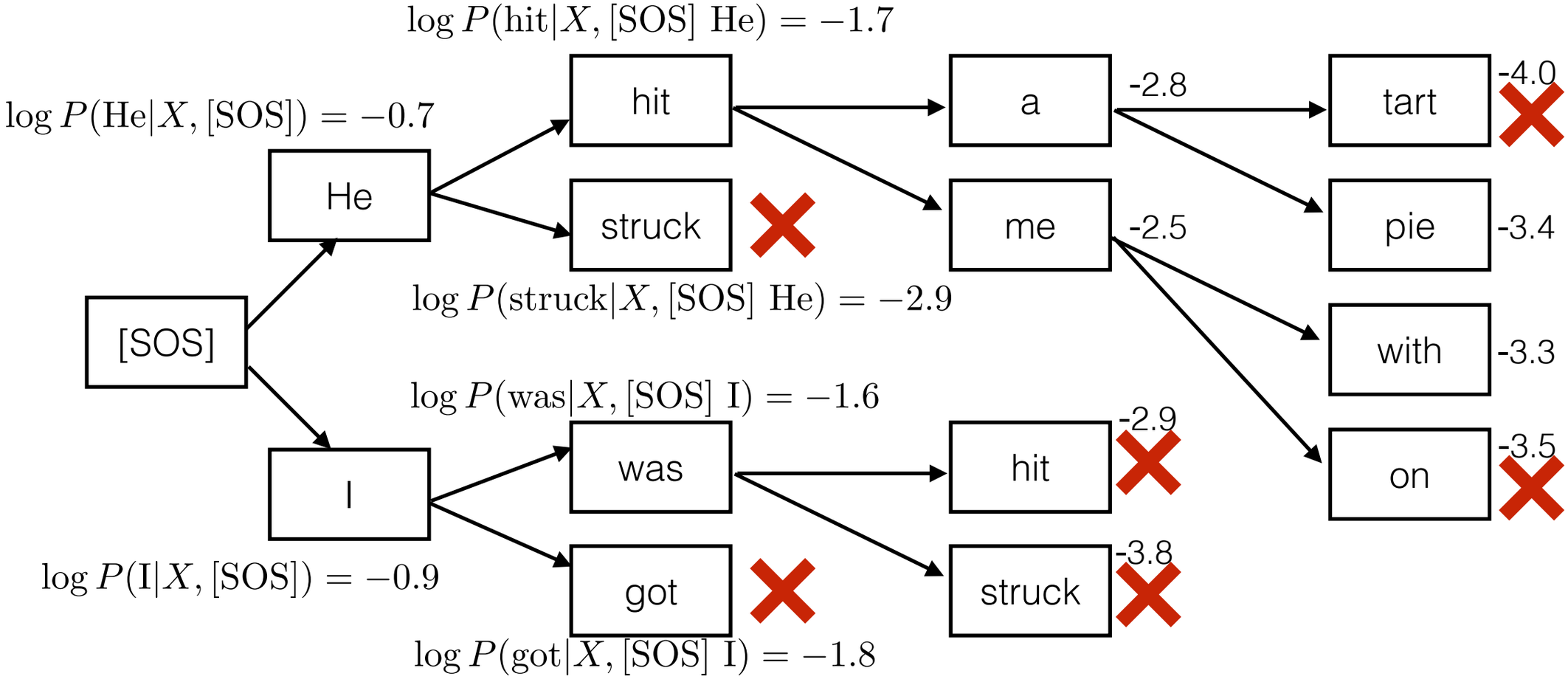


Beam search example

In practice, we **sum up** the log probabilities (to avoid underflow of probability multiplication)

$$P(Y|X) = \prod_{j=0}^{J} P(y_j|X, y_0, \cdots, y_{j-1}) \quad \Rightarrow \quad \log P(Y|X) = \sum_{j=0}^{J} \log P(y_j|X, y_0, \cdots, y_{j-1})$$

Example: **k**=2 (track the 2 most likely hypotheses)
Fr-En Machine translation: il m'a frappé avec une tarte → he hit me with a pie



Beam search summary

$$\log P(Y|X) = \sum_{j=0}^{J} \log P(y_j|X, y_0, \cdots, y_{j-1})$$

At each time step j :

There are k hypotheses in a beam

1. For **each incomplete hypothesis** $(y_0, y_1, \cdots, y_{j-1})$ that we are tracking, find the top k tokens $y_j^{(1)}, y_j^{(2)}, \cdots, y_j^{(k)}$ with the k **highest log-probability scores**

   $\log P(y_j|X, y_0, \cdots, y_{j-1})$

2. Sort the resulting k^2 length-j sequences by their **total log-probability**

   $\sum_{t=0}^{j} \log P(y_t|X, y_0, \cdots, y_{t-1})$

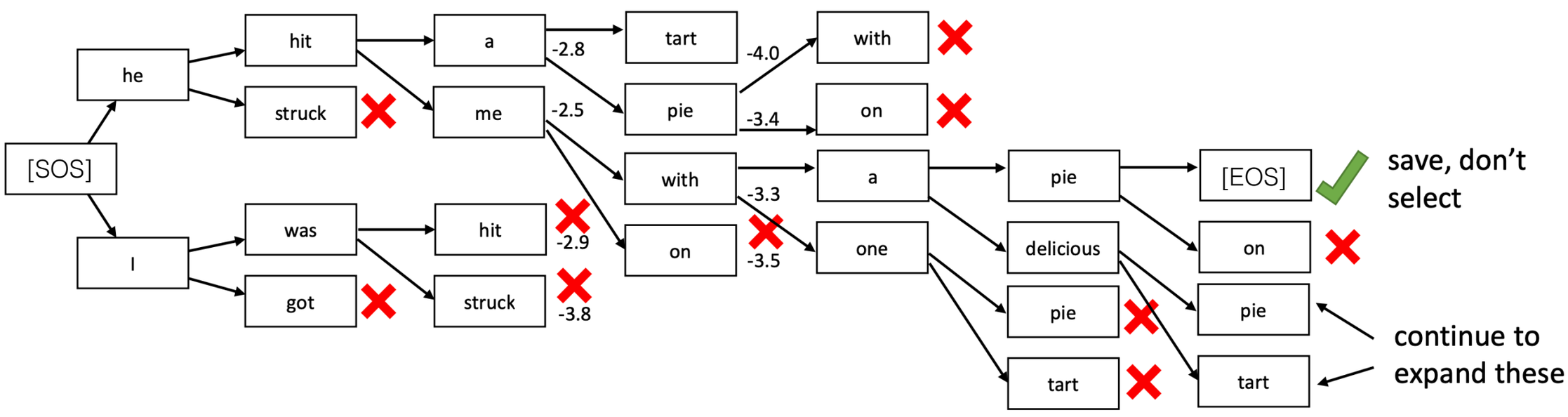
3. Keep the top k hypothesis (out of k^2) in the beam

4. Advance each hypothesis to time $j+1$

When do we stop decoding?

If one of the highest-scoring hypotheses ends in [EOS]

- Save it along with its score, but do not pick it to expand further
- Keep expanding the k remaining best hypotheses



Continue until either some cutoff length ***J*** or until we have k hypotheses that end in [EOS]

Which sequence do we pick?

- At the end of beam search, we might have 3 hypotheses
  1. [SOS] He hit me with a pie [EOS] $\log P(Y|X)$ = -4.5
  2. [SOS] He threw a pie [EOS] $\log P(Y|X)$ = -3.2
  3. [SOS] I was hit with a pie that he threw [EOS] $\log P(Y|X)$ = -7.2

Is this the best?

$$\log P(Y|X) = \sum_{j=0}^{J} \log P(y_j|X, y_0, \cdots, y_{j-1})$$

Problem: $P<1$ **always**, hence $\log P < 0$ **always**

The **longer** the sequence the **lower** its total score
(more negative numbers added together)

Simple "fix": normalize total score by sequence length

$$\text{score}(Y|X) = \frac{1}{J} \sum_{j=0}^{J} \log P(y_j|X, y_0, \cdots, y_{j-1})$$

Beam search summary

$$\text{score}(Y|X) = \frac{1}{J}\sum_{j=0}^{J} \log P(y_j|X, y_0, \cdots, y_{j-1})$$

At each time step j :

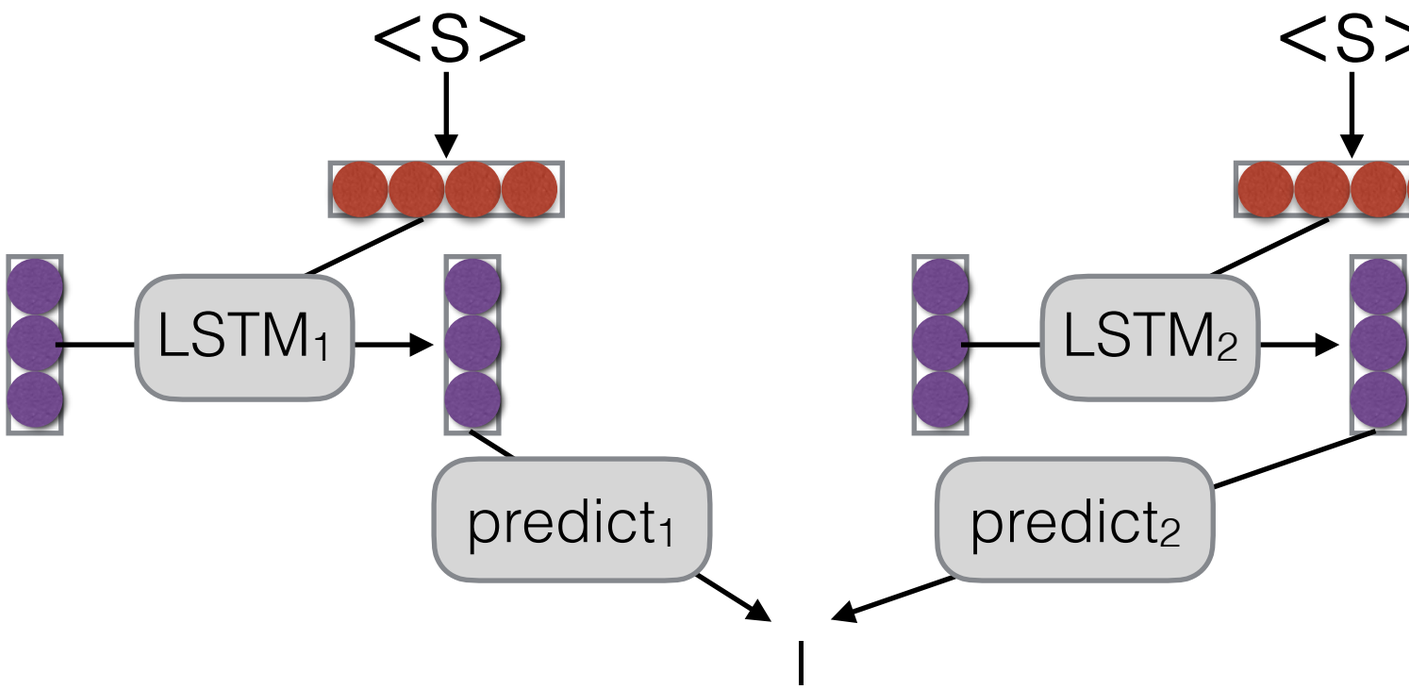
1. For each incomplete hypothesis $(y_0, y_1, \cdots, y_{j-1})$ that we are tracking, find the top k tokens $y_j^{(1)}, y_j^{(2)}, \cdots, y_j^{(k)}$ with the k highest log-probability scores
2. Sort the resulting k^2 length-j sequences by their total log-probability
3. Keep the top k hypothesis (out of k^2) in the beam
4. Advance each hypothesis to time $j+1$

Return saved sequence with highest $\text{score}(Y|X)$

Model Ensemble

Ensemble

- Combine predictions from multiple models



- Why?
  - Multiple models make somewhat uncorrelated errors
  - Models tend to be more uncertain when they are about to make errors
  - Smooths over idiosyncrasies of the model

Linear Interpolation

- Take a weighted average of the M model probabilities

$$P(y_j \mid X, y_1, \ldots, y_{j-1}) = \sum_{m=1}^{M} \underbrace{P_m(y_j \mid X, y_1, \ldots, y_{j-1})}_{\text{Probability according to model } m} \underbrace{P(m \mid X, y_1, \ldots, y_{j-1})}_{\text{Probability of model } m}$$

- Second term often set to uniform distribution 1/M

Log-linear Interpolation

- Weighted combination of log probabilities, normalize

$$P(y_j \mid X, y_1, \ldots, y_{j-1}) =$$

$$\text{softmax}\left(\sum_{m=1}^{M} \lambda_m(X, y_1, \ldots, y_{j-1}) \log P_m(y_j \mid X, y_1, \ldots, y_{j-1})\right)$$

Normalize

Interpolation coefficient for model *m*

Log probability of model *m*

- Interpolation coefficient often set to uniform distribution 1/M

Linear or Log Linear?

- Think of it in logic!
- **Linear:** “Logical OR” — linear addition operator
  - the interpolated model likes any choice that **one model** gives a high probability
  - use models with models that capture different traits
  - necessary when any model can assign zero probability
- **Log Linear:** “Logical AND” — log-linear addition → production
  - interpolated model only likes choices where **all models** agree
  - use when you want to restrict possible answers

Parameter Averaging
(e.g. Bahar et al. 2017)

- **Problem:** Ensembling means we have to use *M* models at test time, increasing our time/memory complexity

- Parameter averaging is a cheap way to get some good effects of ensembling

- Basically, write out model checkpoints at several time steps, and take the average of checkpoint parameters at the end of training.

Ensemble Distillation (e.g. Kim et al. 2016)

- **Problem:** parameter averaging only works for models within the same run

- Knowledge distillation trains **a much smaller student model** to **copy the ensemble model (teacher model)**

  - Specifically, it tries to match the conditional probability over predicted words between the student and teacher models.

  - Why? We want the student model to make the same mistakes as an ensemble teacher model.

- Shown to increase accuracy notably

Case Studies in Conditional Language Modeling

Translation

- Most widely used example of generation?
- e.g. Translation from **WMT shared tasks** (http://statmt.org/wmt21/)
  - News translation
  - Similar language translation
  - Biomedical translation
  - Low-resource multilingual translation
  - Translation efficiency
  - Terminology-aware translation
  - Lifelong learning

Summarization

- Take a longer text, and output a shorter text
- Can use *extractive* models that copy input, or *abstractive* models that generate from scratch
- Multiple granularities:
  - Sentence compression (sentence -> shorter sentence)
  - Single document summarization (document -> shorter text)
  - Multi-document summarization (multiple documents -> text)

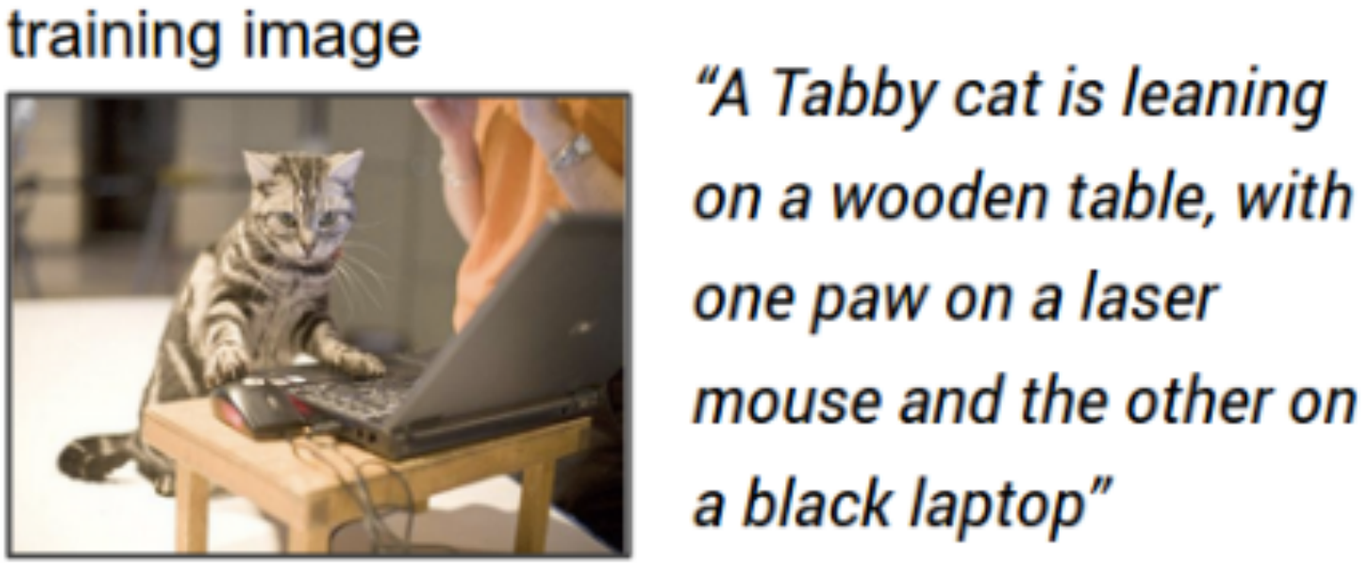
Dialog Response Generation

- Given an input utterance, generate a response
- e.g. Meena Chatbot (Adiwardana et al. 2020)

Human: Hi!
Meena: Hi there! How's it going?
Human: Pretty well.
Meena: I'm glad to hear it. :D
Human: What is your favorite book?
Meena: Oh, that's a toughie. Right now I'd have to say it's A Dance With Dragons, but Harry Potter is a close second.
Human: Oh I love Harry Potter
Meena: Me too! I've read the series at least five times over.
Human: Haha, same probably. Which one do you like the most
Meena: Probably Chamber of Secrets.
Human: Ah. I like Prisoner of azkaban best
Meena: Yeah, that's a good one too.
Human: Do you like the movies more or the books
Meena: I like the movies more. The books are good, don't get me wrong, but I like the pacing better in the movies.
Human: That is an unpopular opinion
Meena: I know. I'm the only one I know who prefers the movies.

Image Captioning
(e.g. Karpathy et al. 2015)

- Input is image features, output is text



- Use standard image encoders (e.g. CNN)

- Often pre-trained on large databases such as ImageNet

From Structured Data

(e.g. Wen et al 2015)

- When you say “Natural Language Generation” to an old-school NLPer, it means this

	SF Restaurant	SF Hotel
act type	inform, inform_only, reject, confirm, select, request, reqmore, goodbye	
shared	name, type, *pricerange, price, phone, address, postcode, *area, *near	
specific	*food *goodformeal ***kids-allowed**	***hasinternet** ***acceptscards** ***dogs-allowed**

bold=binary slots, *=slots can take “don’t care” value

Still a Difficult Problem!

- e.g. "Challenges in data-to-document generation" (Wiseman et al. 2017)

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AS ...
Heat	11	12	103	49	47	27
Hawks	7	15	95	43	33	20

PLAYER	AS	RB	PT	FG	FGA	CITY ...
Tyler Johnson	5	2	27	8	16	Miami
Dwight Howard	4	17	23	9	11	Atlanta
Paul Millsap	2	9	21	8	12	Atlanta
Goran Dragic	4	2	21	8	17	Miami
Wayne Ellington	2	3	19	7	15	Miami
Dennis Schroder	7	4	17	8	15	Atlanta
Rodney McGruder	5	5	11	3	8	Miami
Thabo Sefolosha	5	5	10	5	11	Atlanta
Kyle Korver	5	3	9	3	9	Atlanta
...						

The Utah Jazz (38 - 26) defeated the Houston Rockets (38 - 26) 117 - 91 on Wednesday at Energy Solutions Arena in Salt Lake City . The Jazz got out to a quick start in this one , out - scoring the Rockets 31 - 15 in the first quarter alone . Along with the quick start , the Rockets were the superior shooters in this game , going 54 percent from the field and 43 percent from the three - point line , while the Jazz went 38 percent from the floor and a meager 19 percent from deep . The Rockets were able to out - rebound the Rockets 49 - 49 , giving them just enough of an advantage to secure the victory in front of their home crowd . The Jazz were led by the duo of Derrick Favors and James Harden . Favors went 2 - for - 6 from the field and 0 - for - 1 from the three - point line to score a game - high of 15 points , while also adding four rebounds and four assists ....

Figure 2: Example document generated by the Conditional Copy system with a beam of size 5. Text that accurately reflects a record in the associated box- or line-score is highlighted in blue, and erroneous text is highlighted in red.

- Focused evaluation using, e.g. information extraction

Level of Constraint on Output

- Given the conditioning, the outputs can be more or less constrained, very rough approximation below

Similar Language Translation	Distant Language Translation	Image Captioning	Data-to-Text	Dialog Response

More constrained ⟷ Less constrained

- More freedom = more flexibility, but often more difficulty in modeling and evaluation

Controlled Generation

- Add a further constraint in addition to content-based ones
- **Politeness/Style Control:** Take an input *X* and a label indicating style, etc. (e.g. Sennrich et al. 2016)

source	Give me the telephone!
reference	Gib mir das Telefon! [T]
none	Gib mir das Telefon! [T]
polite	Geben Sie mir das Telefon! [V]
informal	Gib mir das Telefon! [T]

- **Personalization:** Take an input *X* and a side information about the speaker (e.g. Hoang et al. 2016)
- **Sentiment:** Control the sentiment of the generated sentence (e.g. Hu et al. 2018)
- etc. etc.

How do we Evaluate?

Basic Evaluation Paradigm

- Use parallel test set
- Use system to generate translations
- Compare target translations w/ reference

Human Evaluation

- Ask a human to do evaluation

太郎が花子を訪れた

	Taro visited Hanako	the Taro visited the Hanako	Hanako visited Taro
Adequate?	Yes	Yes	No
Fluent?	Yes	No	Yes
Better?	1	2	3

- Final goal, but slow, expensive, and sometimes inconsistent

Human Evaluation Shared Tasks

- **Machine Translation**
  - Conference on Machine Translation (WMT) shared tasks http://www.statmt.org/wmt20/
- **Composite Leaderboard**
  - GENIE leadeboard for QA, summarization, MT https://genie.apps.allenai.org/

BLEU

- Works by comparing n-gram overlap w/ reference

Reference: Taro visited Hanako

System: the Taro visited the Hanako

1-gram: 3/5

2-gram: 1/4

Brevity: min(1, |System|/|Reference|) = min(1, 5/3)

brevity penalty = 1.0

BLEU-2 = $(3/5*1/4)^{1/2} * 1.0$

= 0.387

- **Pros:** Easy to use, good for measuring system improvement
- **Cons:** Often doesn't match human eval, bad for comparing very different systems

Embedding-based Metrics

- Recently, many metrics based on neural models
  - **BertScore:** Find similarity between BERT embeddings (unsupervised) (Zhang et al. 2020)
  - **BLEURT:** Train BERT to predict human evaluation scores (Sellam et al. 2020)
  - **COMET:** Train model to predict human eval, also using source sentence (Rei et al. 2020)
  - **PRISM:** Model based on training paraphrasing model (Thompson and Post 2020)
  - **BARTScore:** Calculate the probability of source, reference, or system output (Yuan et al. 2021)

Perplexity

- Calculate the perplexity of the words in the held-out set *without* doing generation
- **Pros:** Naturally solves multiple-reference problem!
- **Cons:** Doesn't consider decoding or actually generating output.
- May be reasonable for problems with lots of ambiguity.

Which One to Use?

- **Meta-evaluation** runs human evaluation and automatic evaluation on the same outputs, calculates correlation

- Examples:
  - **WMT Metrics Task** for MT (Mathur et al. 2021)
  - **RealSumm** for summarization (Bhandari et al. 2020)

- Evaluation is hard, especially with good systems! Most metrics had no correlation w/ human eval over best systems at some WMT 2019 tasks

Questions?