CS11-737 Multilingual NLP

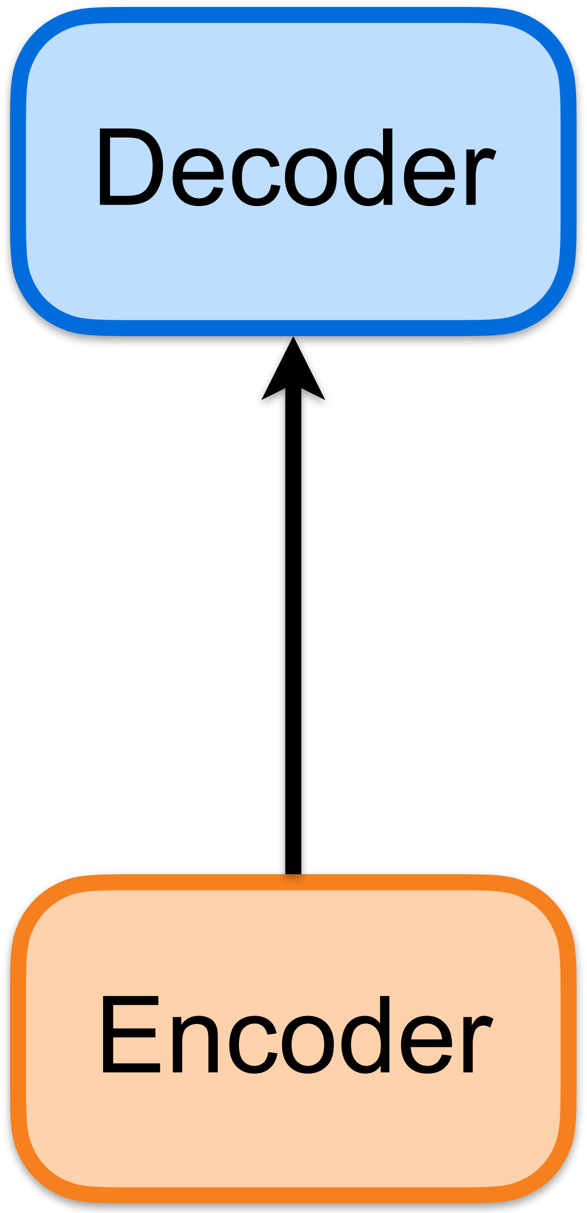
Sequence Decoding

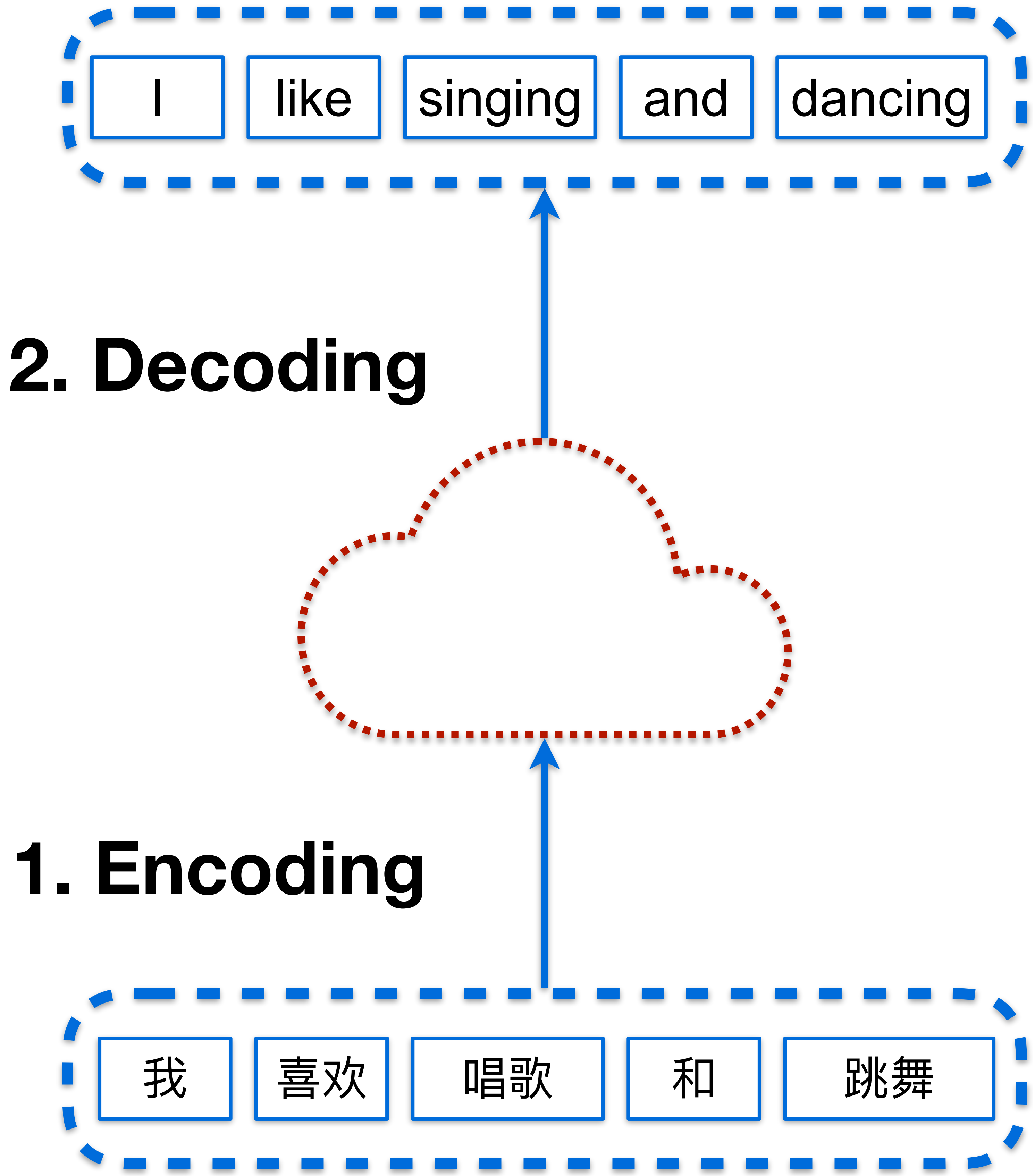
Lei Li

https://lileicc.github.io/course/11737mnlp23fa/

Carnegie Mellon University
Language Technologies Institute

Encoder-Decoder Paradigm





Transformer

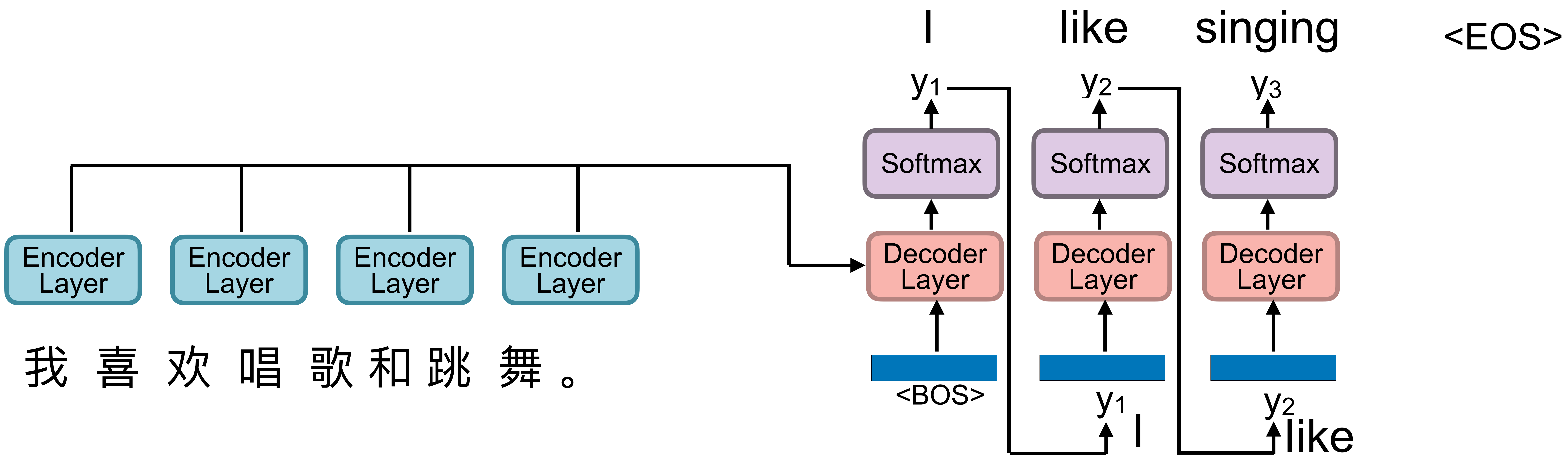
Inference

- Now already trained a model θ
- Decoding/Generation: Given an input sentence x, to generate the target sentence y that maximize the probability $P(y \mid x; \theta)$
- $\underset{y}{\operatorname{argmax}} P(y \mid x) = f_\theta(x, y)$
- Two types of error
  - the most probable translation is bad → fix the model
  - search does not find the most probably translation → fix the search
- Most probable translation is not necessary the highest BLEU one!

Autoregressive Generation

greedy decoding: output the token with max next token prob

$$\underset{y_t}{\operatorname{argmax}} P(y_t \mid x, y_{1:t-1})$$



But, this is not necessary the best

Sequence Decoding

$$\underset{y}{\operatorname{argmax}} P(y \mid x) = f_\theta(x, y)$$

- naive solution: exhaustive search
  - too expensive
- Beam search
  - (approximate) dynamic programming

Beam Search

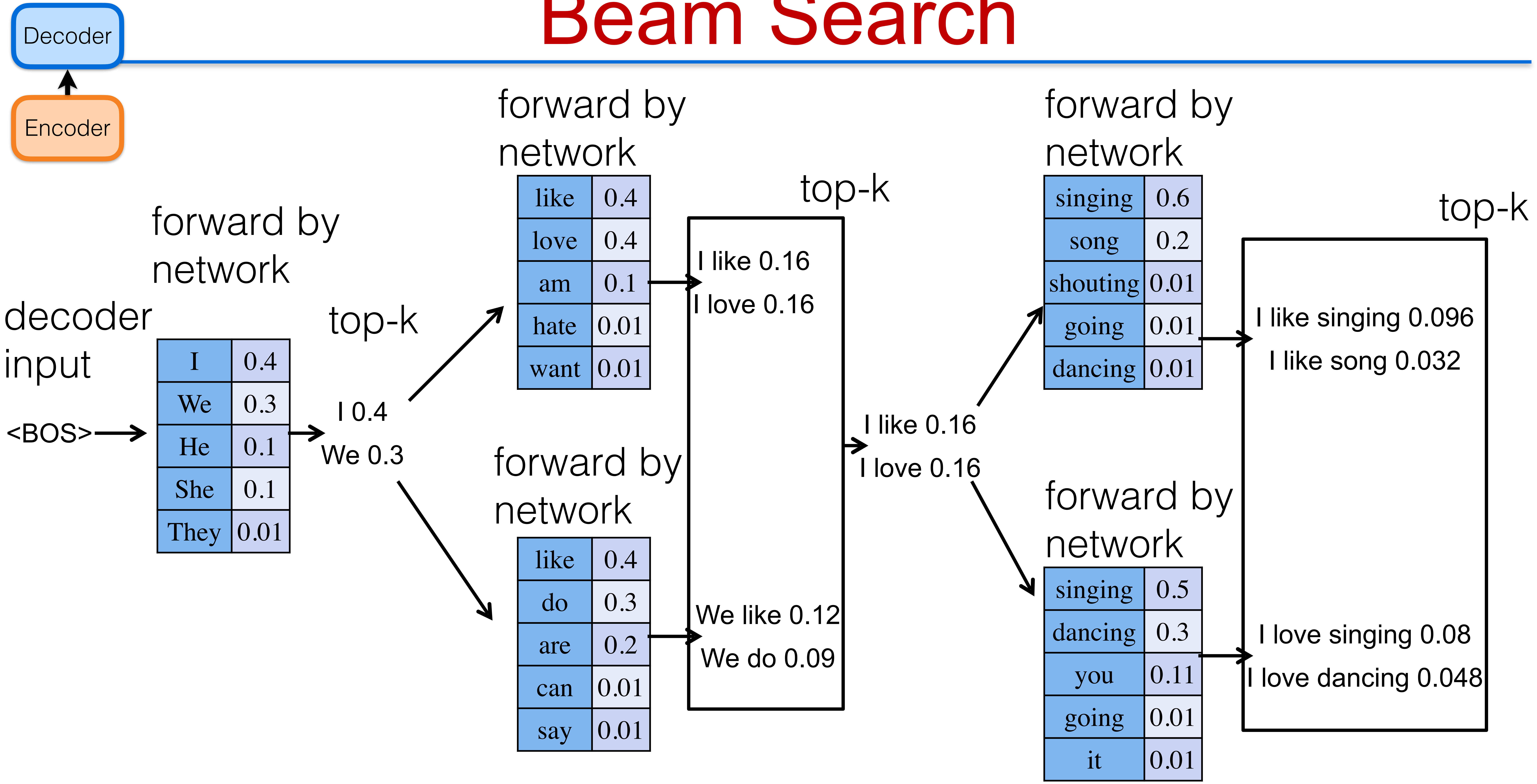
1. start with empty S

2. at each step, keep k best partial sequences

3. expand them with one more forward generation

4. collect new partial results and keep top-k

Beam Search


Decoder
Encoder
decoder input
<BOS>
forward by network
I 0.4
We 0.3
He 0.1
She 0.1
They 0.01
top-k
I 0.4
We 0.3
forward by network
like 0.4
love 0.4
am 0.1
hate 0.01
want 0.01
forward by network
like 0.4
do 0.3
are 0.2
can 0.01
say 0.01
top-k
I like 0.16
I love 0.16
We like 0.12
We do 0.09
I like 0.16
I love 0.16
forward by network
singing 0.6
song 0.2
shouting 0.01
going 0.01
dancing 0.01
forward by network
singing 0.5
dancing 0.3
you 0.11
going 0.01
it 0.01
top-k
I like singing 0.096
I like song 0.032
I love singing 0.08
I love dancing 0.048

Beam Search (pseudocode)

```
best_scores = []
add {[0], 0.0} to best_scores # 0 is for beginning of sentence token
for i in 1 to max_length:
  new_seqs = PriorityQueue()
  for (candidate, s) in best_scores:
    if candidate[-1] is EOS:
       prob = all -inf
       prob[EOS] = 0
     else:
      prob = using model to take candidate and compute next token probabilities (logp)
    pick top k scores from prob, and their index
    for each score, index in the top-k of prob:
      new_candidate = candidate.append(index)
      new_score = s + score
      if not new_seqs.full():
        add (new_candidate, new_score) to new_seqs
      else:
        if new_seqs.queue[0][1] < new_score:
```

Pruning for Beam Search

- Relative threshold pruning
  - prune candidates with too low score from the top one
  - Given a pruning threshold rp and an active candidate list C, a candidate cand ∈ C is discarded if: $\text{score}(cand) \leq rp * \max\{\text{score}(c)\}$

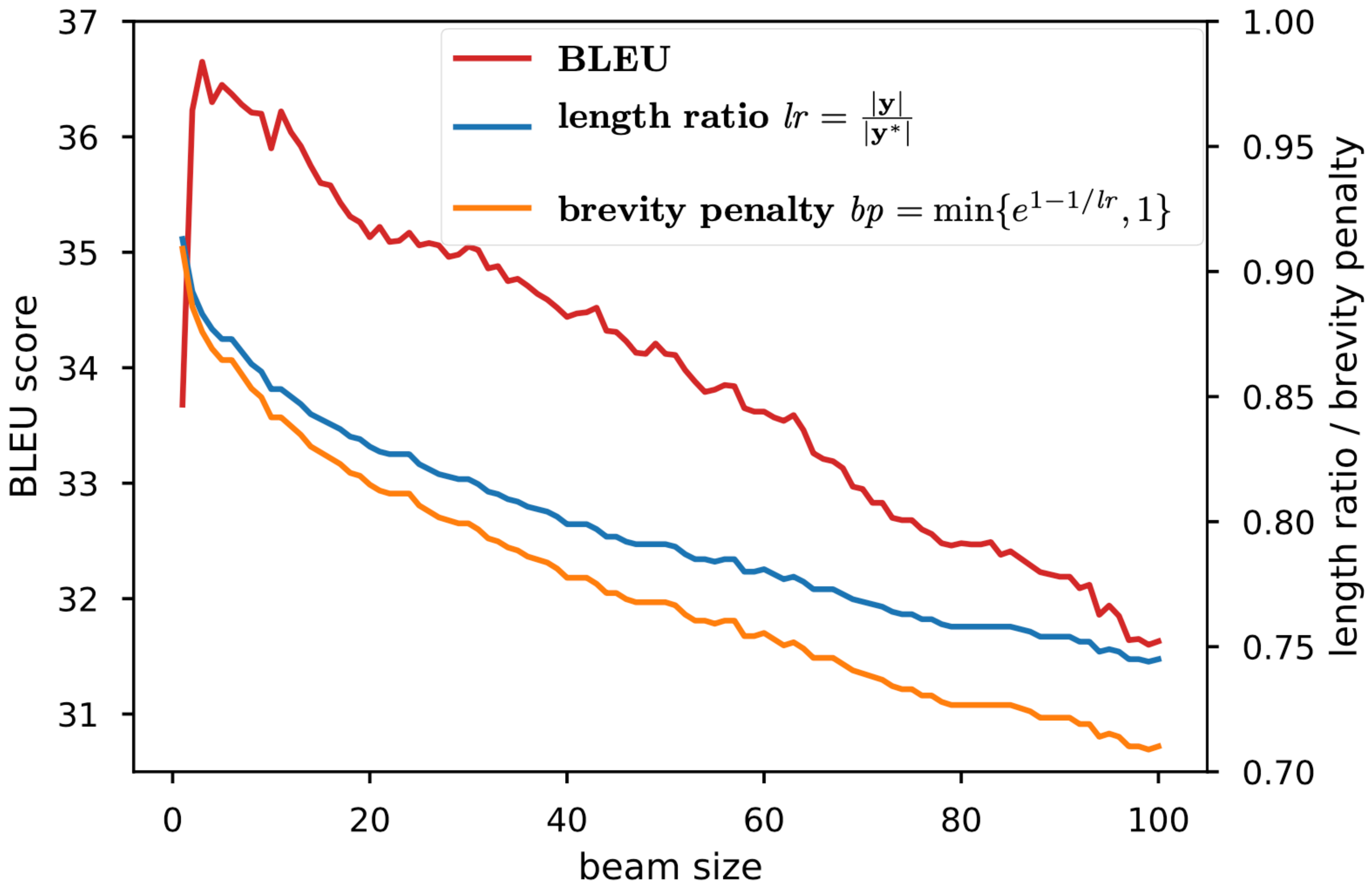
- Absolute threshold pruning:
  - $\text{score}(cand) \leq \max\{\text{score}(c)\} - ap$

- Relative local threshold pruning

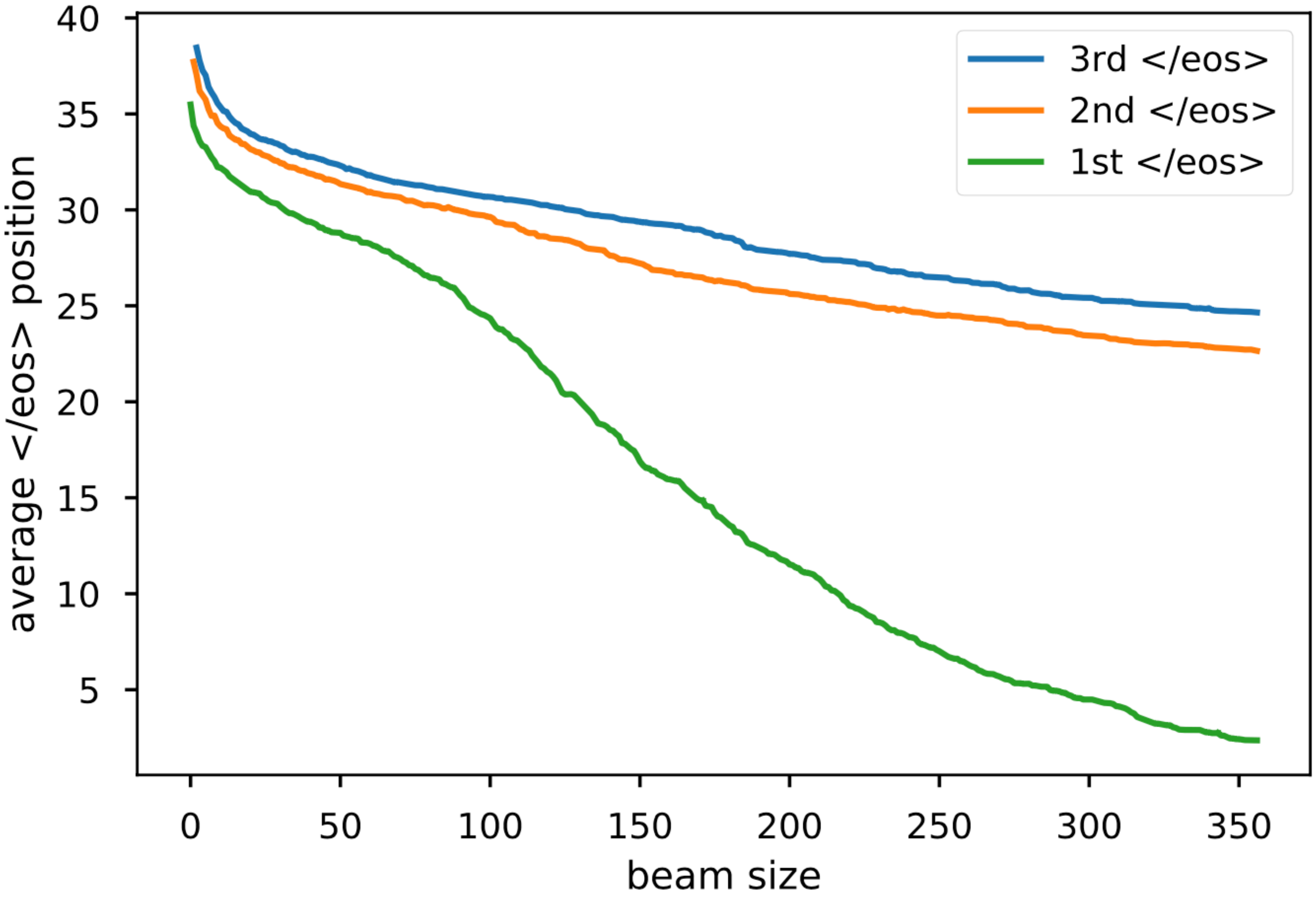
Freitag & Al-Onaizan. Beam Search Strategies for Neural Machine Translation. 2017.

What is Beam size?

- 3 to 5
- Why not larger?
  - larger does not necessarily produce higher BLEU



Larger Beam -> Shorter Translation



When to stop? Normalization of Score

- Length normalization: $\hat{S}_{\text{length\_norm}}(\mathbf{x}, \mathbf{y}) = S(\mathbf{x}, \mathbf{y}) / |\mathbf{y}|$
- Word-reward: promoting longer sentences
  - $\hat{S}_{\text{WR}}(\mathbf{x}, \mathbf{y}) = S(\mathbf{x}, \mathbf{y}) + r \cdot |\mathbf{y}|$
- Bounded word reward with length prediction
  - $L_{pred}(\mathbf{x}) = gr^*(\mathbf{x}) \cdot |\mathbf{x}|$

$$L^*(\mathbf{x}, \mathbf{y}) = \min\{|\mathbf{y}|, L_{pred}(\mathbf{x})\}$$

  - $\hat{S}_{\text{BWR}^*}(\mathbf{x}, \mathbf{y}) = S(\mathbf{x}, \mathbf{y}) + r \cdot L^*(\mathbf{x}, \mathbf{y})$

Yang et al. Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation. 2018

Diverse Beam Search and Reranking

Diverse Beam Search

- Top k results from NMT decoding are very similar
- Same for other text generation tasks
- Need more diversity?
  - e.g. in image-captioning, diverse candidates are desired

How

- Two approaches
  - MMI: maximizing mutual information of MI(X, Y) instead of P(Y|X)
  - Maximize the penalized score: log P(Y|X) + distance(Y and existing candidates)

Maximize mutual information (MMI)

- Mutual Information

$$\mathrm{MI}(X, Y) = \frac{p(X, Y)}{p(X)p(Y)}$$

$$\Downarrow$$

$$\arg\max \log p(Y|X) - \lambda \log p(Y)$$

need a separate Language model p(Y) for target language

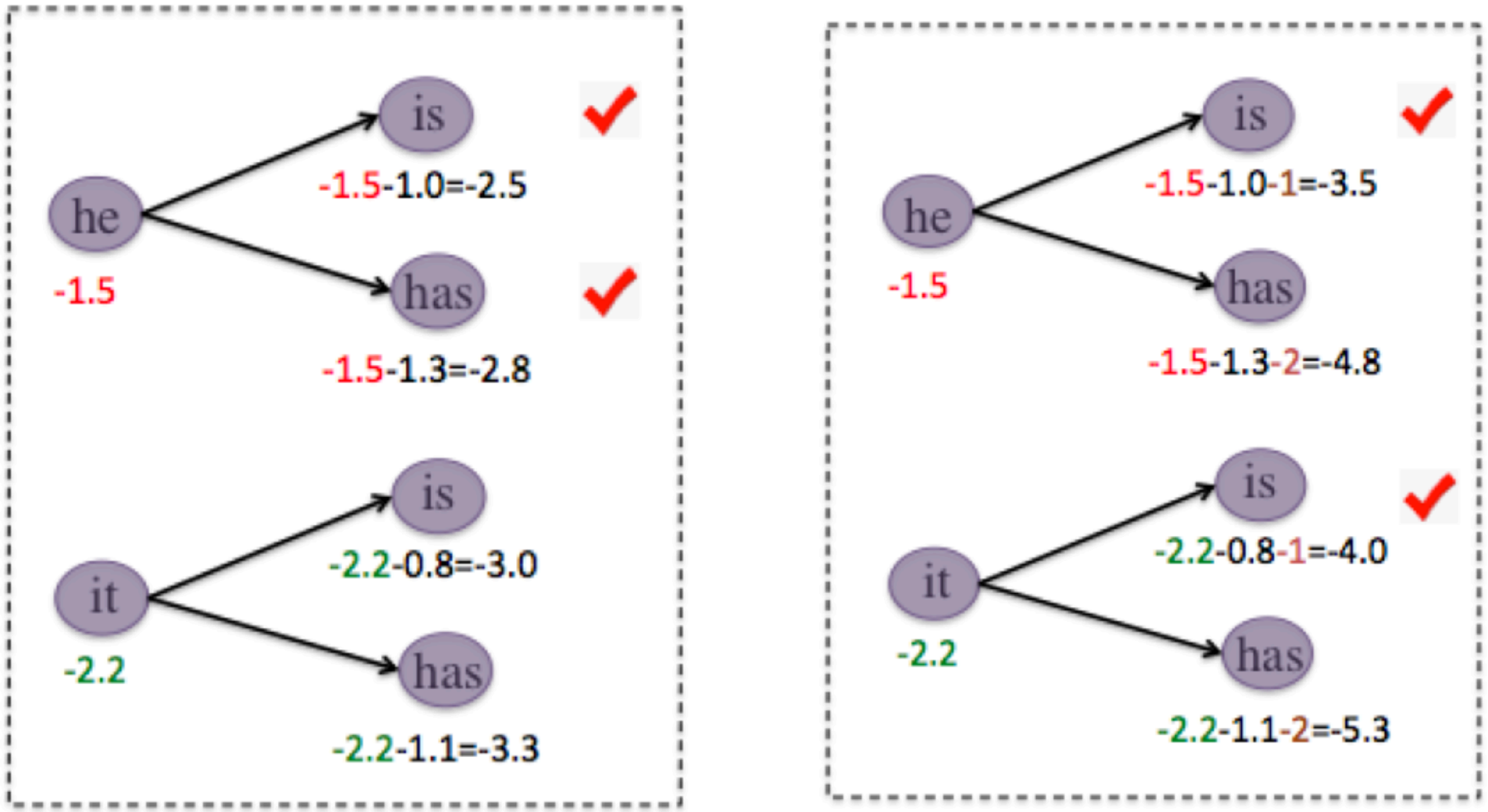
Li et al. A Diversity-Promoting Objective Function for Neural Conversation Models. 2016

Maximizing Mutual Information

$$\arg\max (1-\lambda)\log p(Y|X) + \lambda \log p(X|Y)$$

- penalized forward decoding
  - p(Y|X) - \gamma rank_y

$$\hat{S}(Y_{t-1}^{k}, y_t^{k,k'}|x) = S(Y_{t-1}^{k}, y_t^{k,k'}|x) - \gamma k'$$



Standard Beam Search   Diversity Promoting Beam Search (γ set to 1)

Reranking

- Obtain N-best from beam search
- Rerank based on:
  - Score(y) = $\log p(y|x) + \lambda \log p(x|y) + \gamma \log p(y) + \eta LT$
- Alternative: learned reranking
  - Lee et al. Discriminative Reranking for Neural Machine Translation. 2021

Sampling

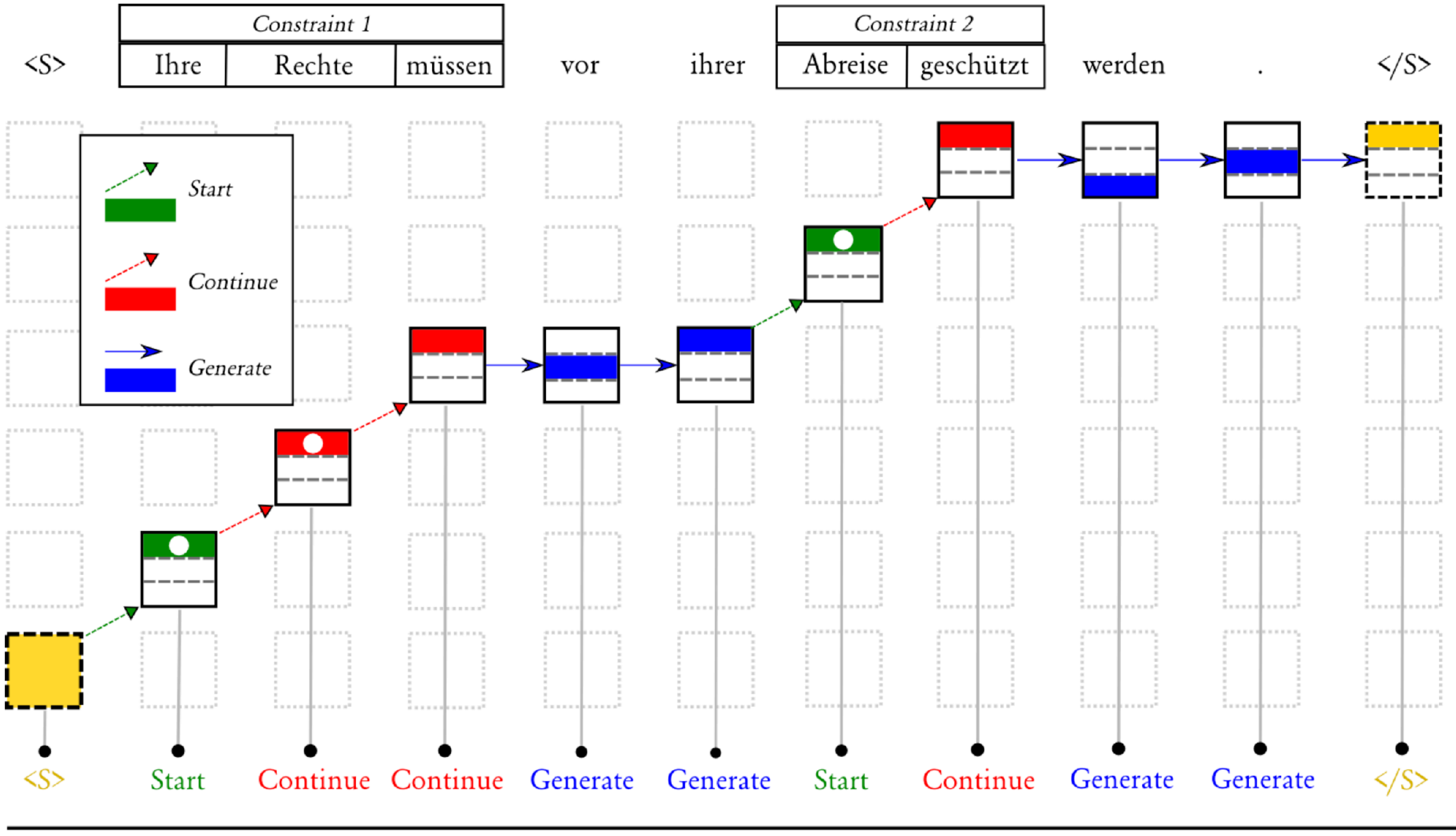
- Instead of $\underset{y}{\operatorname{argmax}} P(y|x) = f_\theta(x, y)$
- Generate samples of translation Y from the distribution P(Y|X)
- Q: how to generate samples from a discrete distribution?

Combine Sample and Beam Search

- Sample the first tokens
- continue beam search for the later
- why?

Lexical Constrained Decoding

- The generated sentence must contain given keywords
- To generate from
  - Vocabulary
  - Keywords

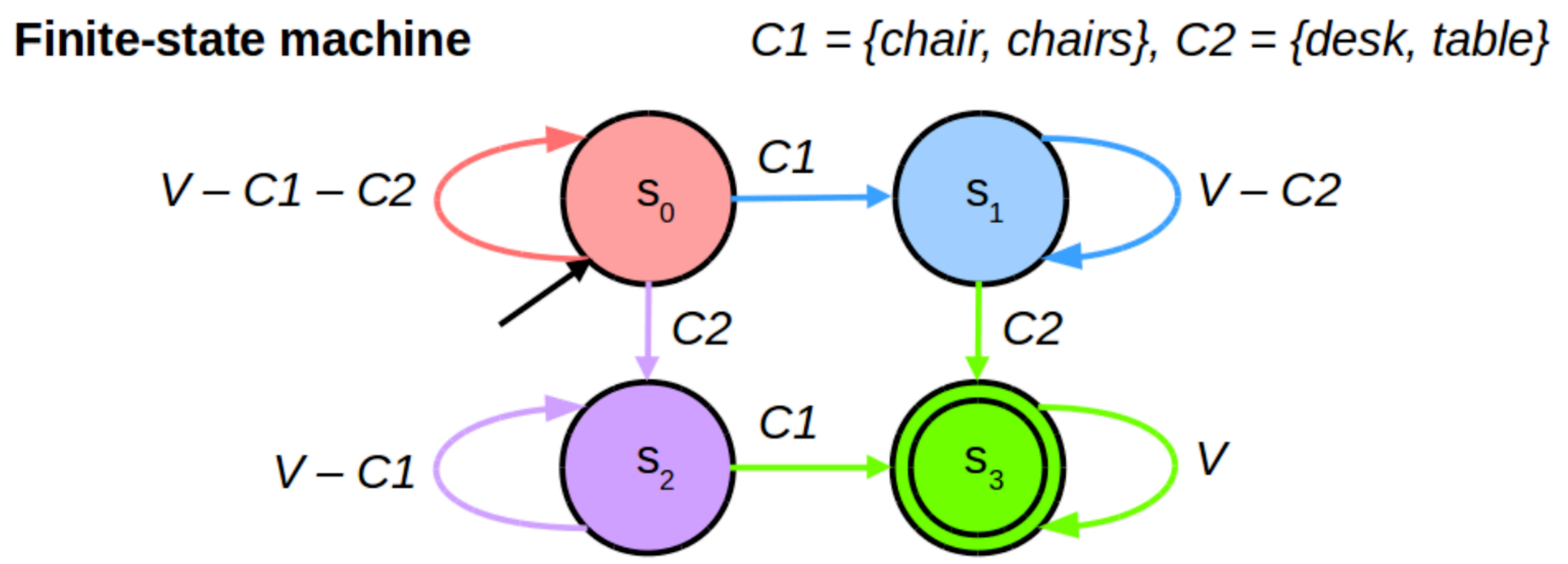


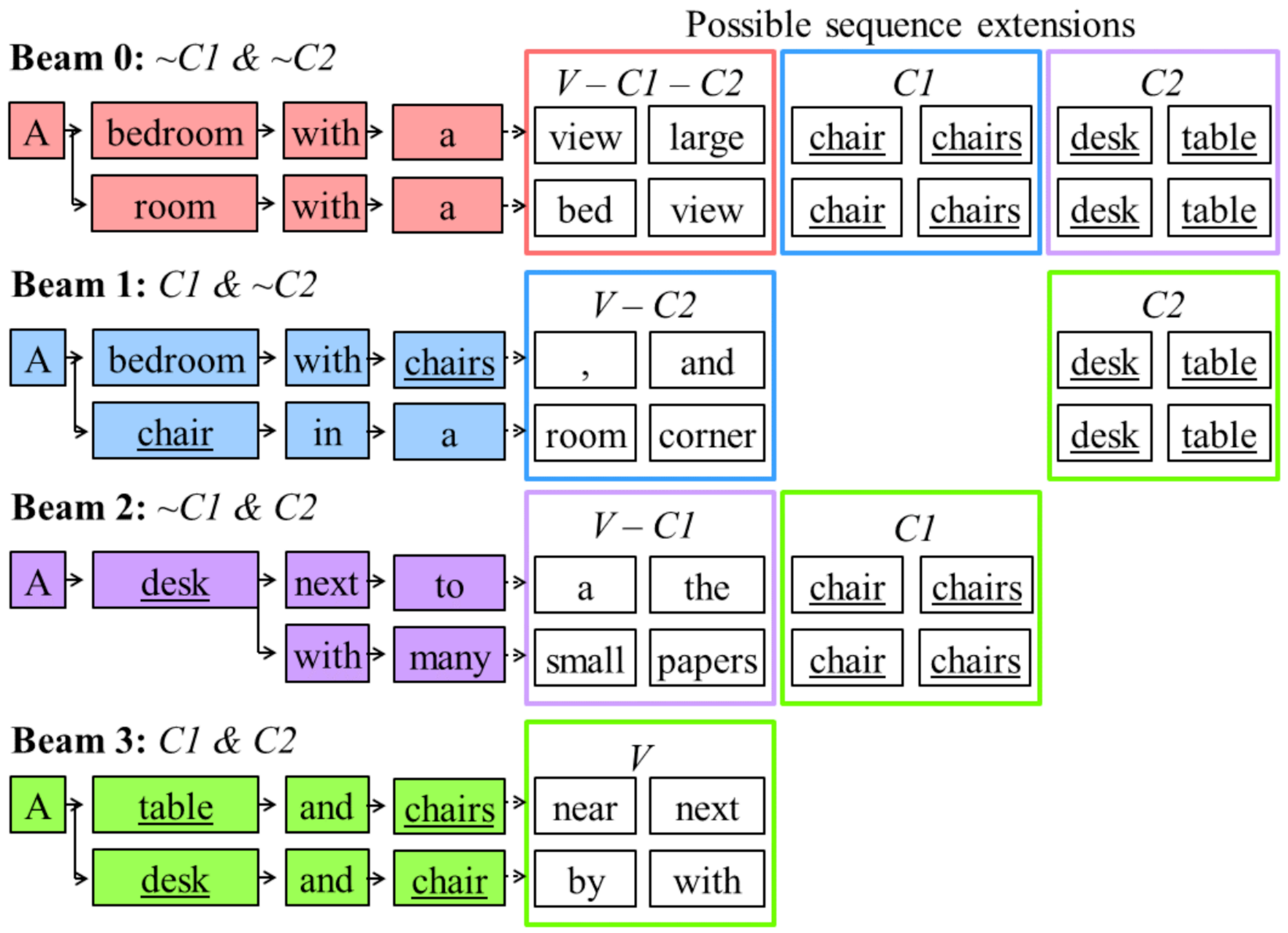
Input: Rights protection should begin before their departure .

Hokamp et al. Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search. 2017

Order-agnostic Constraints

- The generated sentence must contain given keywords
- Using finite state machine to represent constraint state.
- Expand with
  - Vocabulary
  - Constraint keywords





Andersen et al. Guided Open Vocabulary Image Captioning with Constrained Beam Search. 2017

Post-training Processing: Model Average

- Pick the model when converges
- Model average:
  - instead, using the last 5-10 epoch's models, and average the parameters to get one model
  - This turns out to generalize better than the last one.
  - Why? (over-fit)

Model Ensemble

- Train several separate MT model
- decode with

$$\arg\max_{y_t} \sum_k \log P(y_t \mid y_{<t}, x; M_k)$$

Distillation with Ensemble

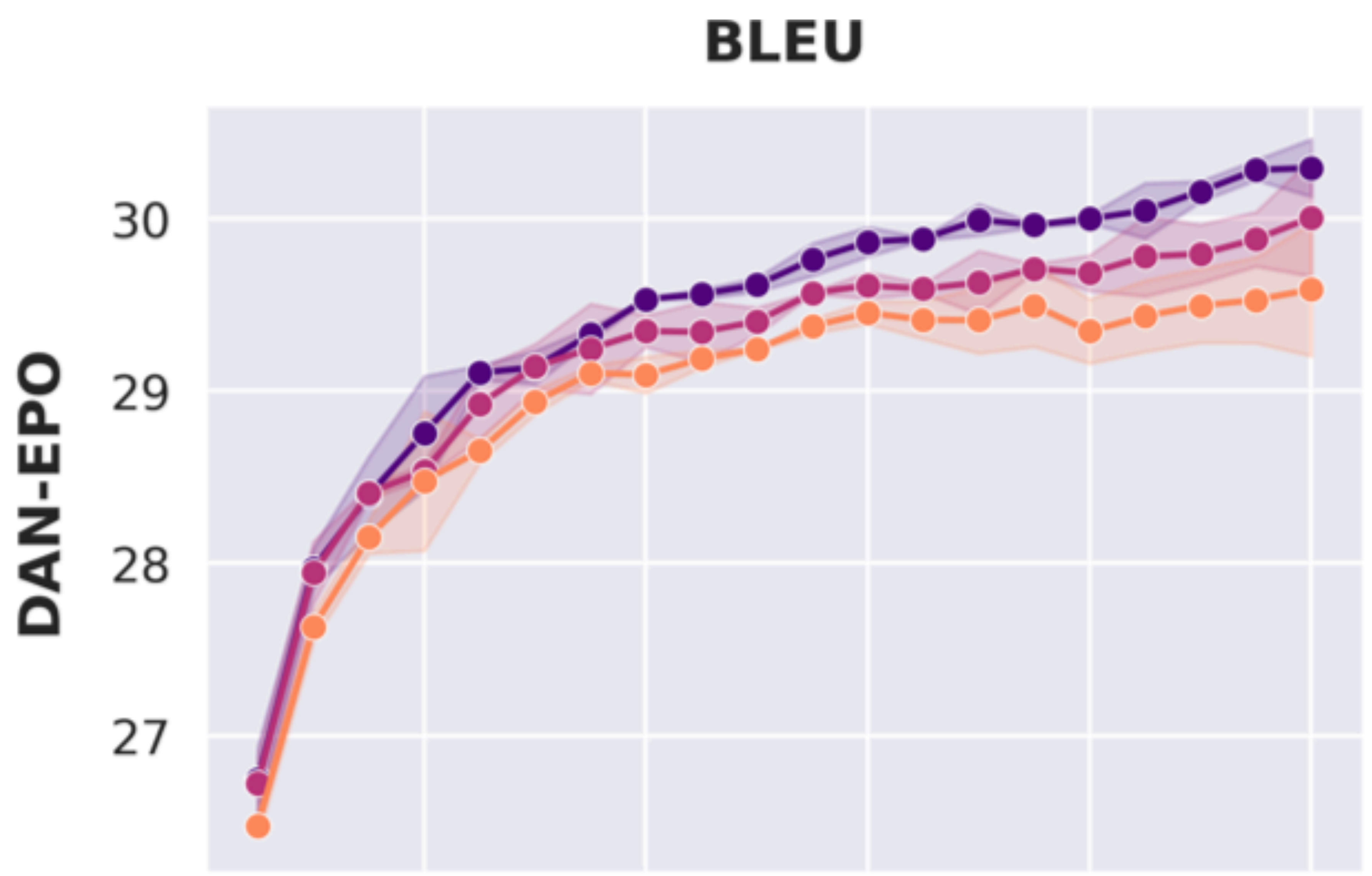
- In order to obtain a single model with good performance.
- Use ensemble model to create pseudo-parallel data
- Train a single MT model using both original training data and pseudo-parallel data.

Minimum Bayes Risk Decoding

- Bias in decoding:
  - length bias
  - word frequency
  - beam search curse
  - copy noise
  - low domain
- Decoding with Mode vs. with most "common" one

Minimum Bayes Risk Decoding

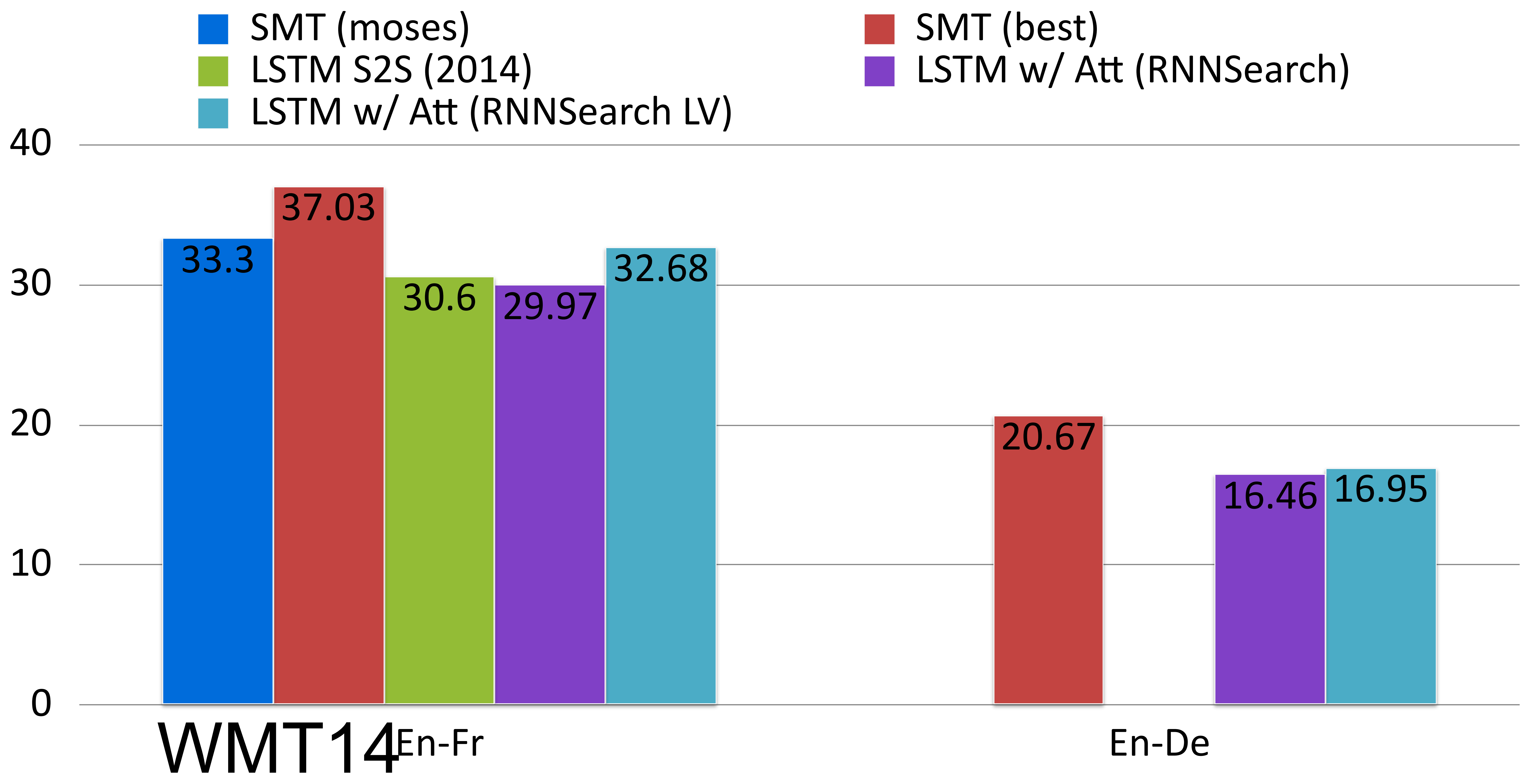
- Minimize risk = maximize average utility
- Utility: similarity among samples.
- $S_1, S_2, \ldots, S_n \sim P(y \mid x, \theta)$
- $\hat{y} \arg\max_{s_i} \frac{1}{n} \sum_j u(s_i, s_j)$



Muller and Sennrich. Understanding the Properties of Minimum Bayes Risk Decoding in Neural Machine Translation. 2021

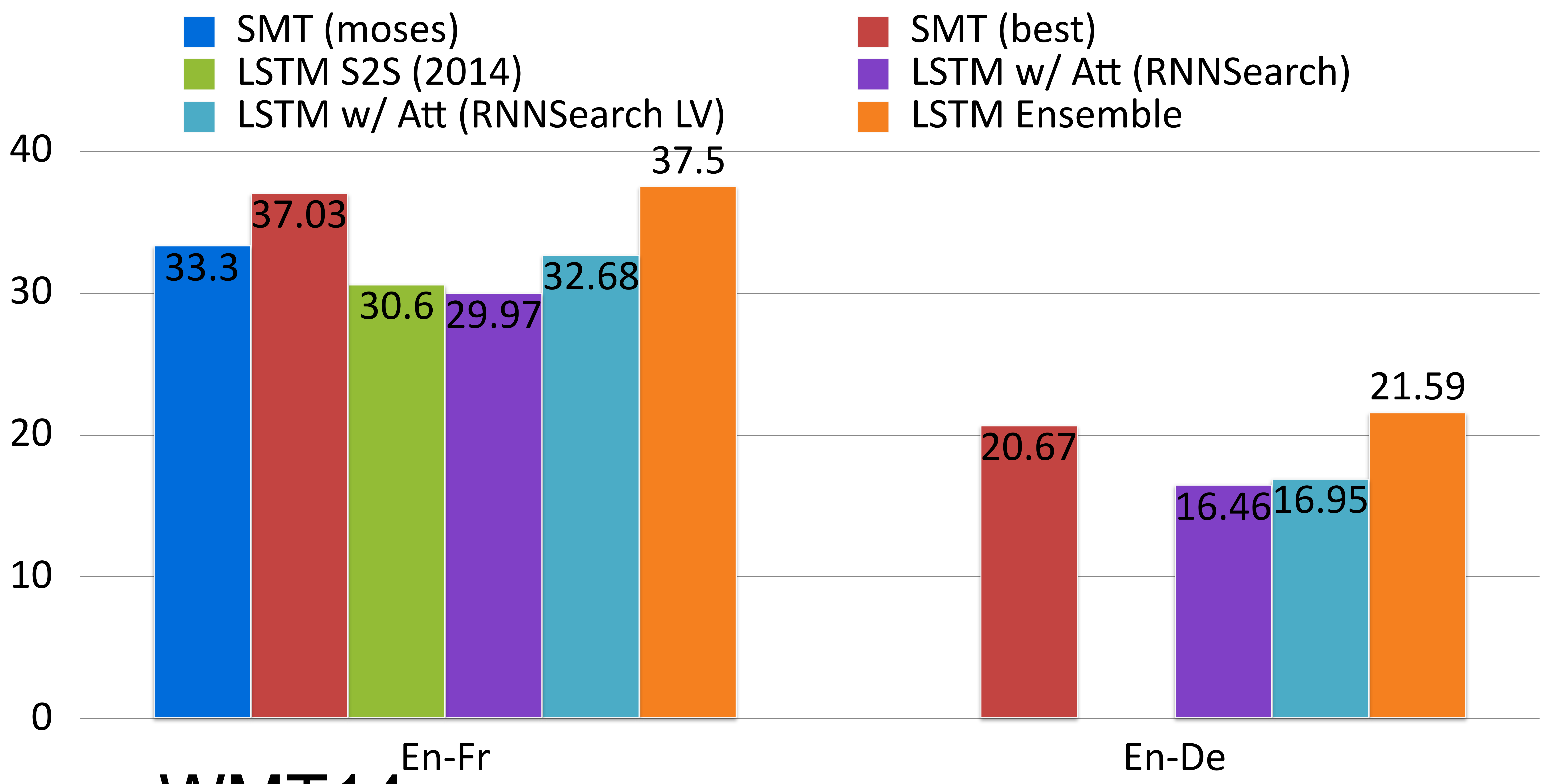
Machine Translation using Seq2seq and Transformer

LSTM Seq2Seq w/ Attention



Jean et al. On Using Very Large Target Vocabulary for Neural Machine Translation. 2015

Performance with Model Ensemble



WMT14

Luong et al. Effective Approaches to Attention-based Neural Machine Translation. 2015

Results on WMT14

- The most widely used benchmark (WMT14 En-De and En-Fr)

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Effectiveness of Choices

- num. heads
- dim of key
- num layers
- hid dim
- ffn dim
- dropout
- pos emb

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

Deep Transformer

- 30 ~ 60 encoder
- 12 decoder
- dynamic linear combination of layers (DLCL)
  - or. deeply supervised
  - combine output from all layers

Wang et al. Learning Deep Transformer Models for Machine Translation, 2019.

Model		Param.	Batch (×4096)	Updates (×100k)	†Times	BLEU	Δ
Vaswani et al. (2017) (Base)		65M	1	1	reference	27.3	-
Bapna et al. (2018)-deep (Base, 16L)		137M	-	-	-	28.0	-
Vaswani et al. (2017) (Big)		213M	1	3	3x	28.4	-
Chen et al. (2018a) (Big)		379M	16	†0.075	1.2x	28.5	-
He et al. (2018) (Big)		†210M	1	-	-	29.0	-
Shaw et al. (2018) (Big)		†210M	1	3	3x	29.2	-
Dou et al. (2018) (Big)		356M	1	-	-	29.2	-
Ott et al. (2018) (Big)		210M	14	0.25	3.5x	29.3	-
post-norm	Transformer (Base)	62M	1	1	1x	27.5	reference
	Transformer (Big)	211M	1	3	3x	28.8	+1.3
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	failed	failed
	DLCL (Base)	62M	1	1	1x	27.6	+0.1
	DLCL-deep (Base, 25L)	121M	2	0.5	1x	29.2	+1.7
pre-norm	Transformer (Base)	62M	1	1	1x	27.1	reference
	Transformer (Big)	211M	1	3	3x	28.7	+1.6
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	28.9	+1.8
	DLCL (Base)	62M	1	1	1x	27.3	+0.2
	DLCL-deep (Base, 30L)	137M	2	0.5	1x	**29.3**	**+2.2**

Wang et al. Learning Deep Transformer Models for Machine Translation, 2019.

Hot Topics in MT

- Parallel Decoding (e.g. NAT, GLAT, DAT,…)
- Low-resource MT
- Unsupervised MT
- Multilingual NMT, Zero-shot NMT
- Speech-to-text translation
  - (Offline) ST
  - Streaming ST

Summary

- Sequence-to-sequence encoder-decoder framework for conditional generation, including Machine Translation
- Key components in Transformer
  - Positional Embedding (to distinguish tokens at different pos)
  - Multihead attention
  - Residual connection
  - layer norm
- Sequence Decoding with Beam search

Class discussion

- Pick a 4-line excerpt from a short text (e.g. poem, text message) in English

- Use Google translate, VolcTrans(translate.volcengine.com), ChatGPT to back-translate the text via a pivot language, e.g.,
    - English → Spanish → English
    - English → L1 → L2 → English, where L1 and L2 are typologically different from English and from each other

- Compare the original text and its English back-translation, and share your observations. For example,
    - What information got lost in the process of translation?
    - Are there translation errors associated with linguistic properties of pivot languages and with linguistic divergences across languages?
    - Try different pivot languages: can you provide insights about the quality of MT for those language pairs?