Introduction to Beam Search

Ki Hyun Kim

nlp.with.deep.learning@gmail.com

Generation == Search Problem

- 가장 확률이 높은 문장을 만들어내는 것은 최단 경로 찾기와 같은 문제

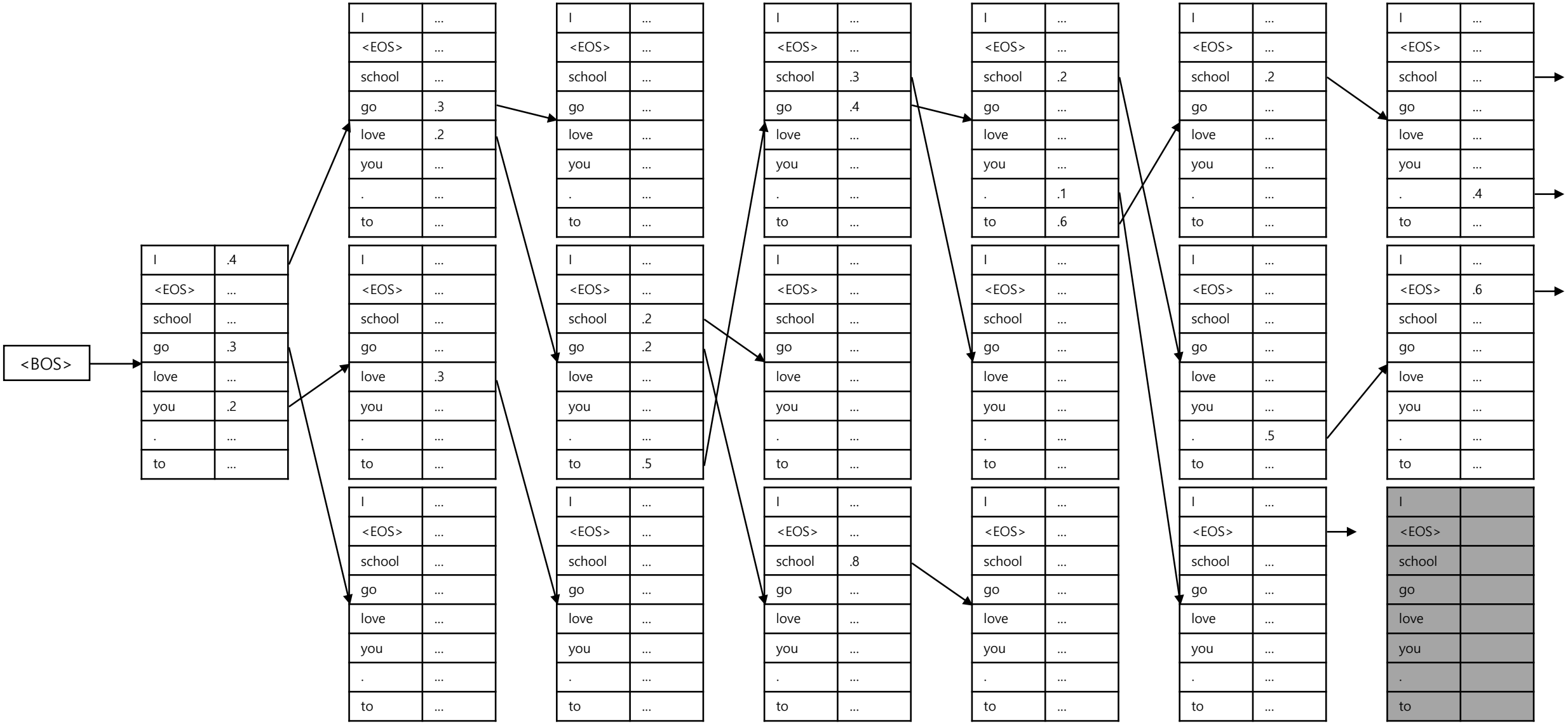
$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \log P(y|x;\theta),$$

$$\text{where } \log P(y|x;\theta) = \sum_{t=1}^{n} \log P(y_t|x, y_{<t};\theta) \text{ and } y = \{y_1, \cdots, y_n\}.$$

- Greedy Search
  - 지금의 최선이 나중에는 나쁜 선택이 될 수 있음

- Beam Search
  - Top-k를 tracking하여 greedy search를 조금 더 안전하게 수행

Beam Search (for 1 sample)



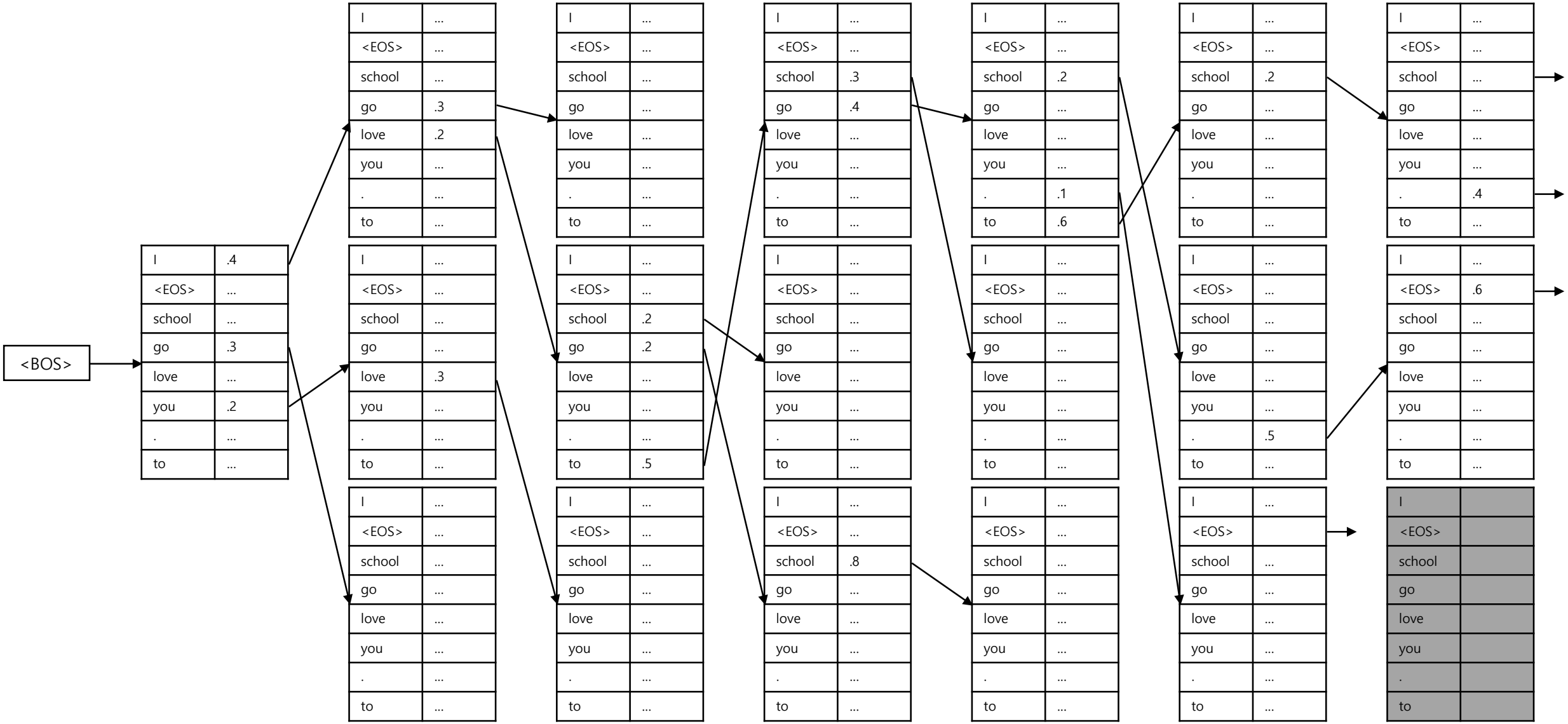
Evaluations

- Beam search가 가장 뛰어난 성능 개선을 보임

Strategy	# Chains	Valid Set		Test Set	
		NLL	BLEU	NLL	BLEU
Ancestral Sampling	50	22.98	15.64	26.25	16.76
Greedy Decoding	-	27.88	15.50	26.49	16.66
Beamsearch	5	20.18	17.03	22.81	18.56
Beamsearch	10	19.92	17.13	22.44	18.59

En-Cz: 12m training sentence pairs [Cho, arXiv 2016]

Parallelized Beam Search (for 1 sample)



Summary

- Auto-regressive 특성으로 인해, greedy search의 한계 발생
  - Beam search를 통해 한계를 완화

- k(beam size)번의 반복 inference를 통해, 좀 더 나은 성능의 추론 수행 가능
  - 하지만 sequential한 연산은 속도 저하 야기하므로 병렬적으로 처리해야 함

- Mini-batch를 처리하는 것처럼 k번을 동시에 처리 가능