LLM Sys

11868 LLM Systems
Decoding

Lei Li

Carnegie Mellon University
Language Technologies Institute

Recap about Tokenization

- Subword tokenization: Byte-Pair-Encoding
  - iteratively merging most frequent pairs of tokens
- Information-theoretic vocabulary (VOLT)
  - solving entropy constrained optimal transport problem
- Pre-tokenization through regex
- Number treatment
- Vocab sharing impact multilingual performance
  - how to solve languages in stagnant quad

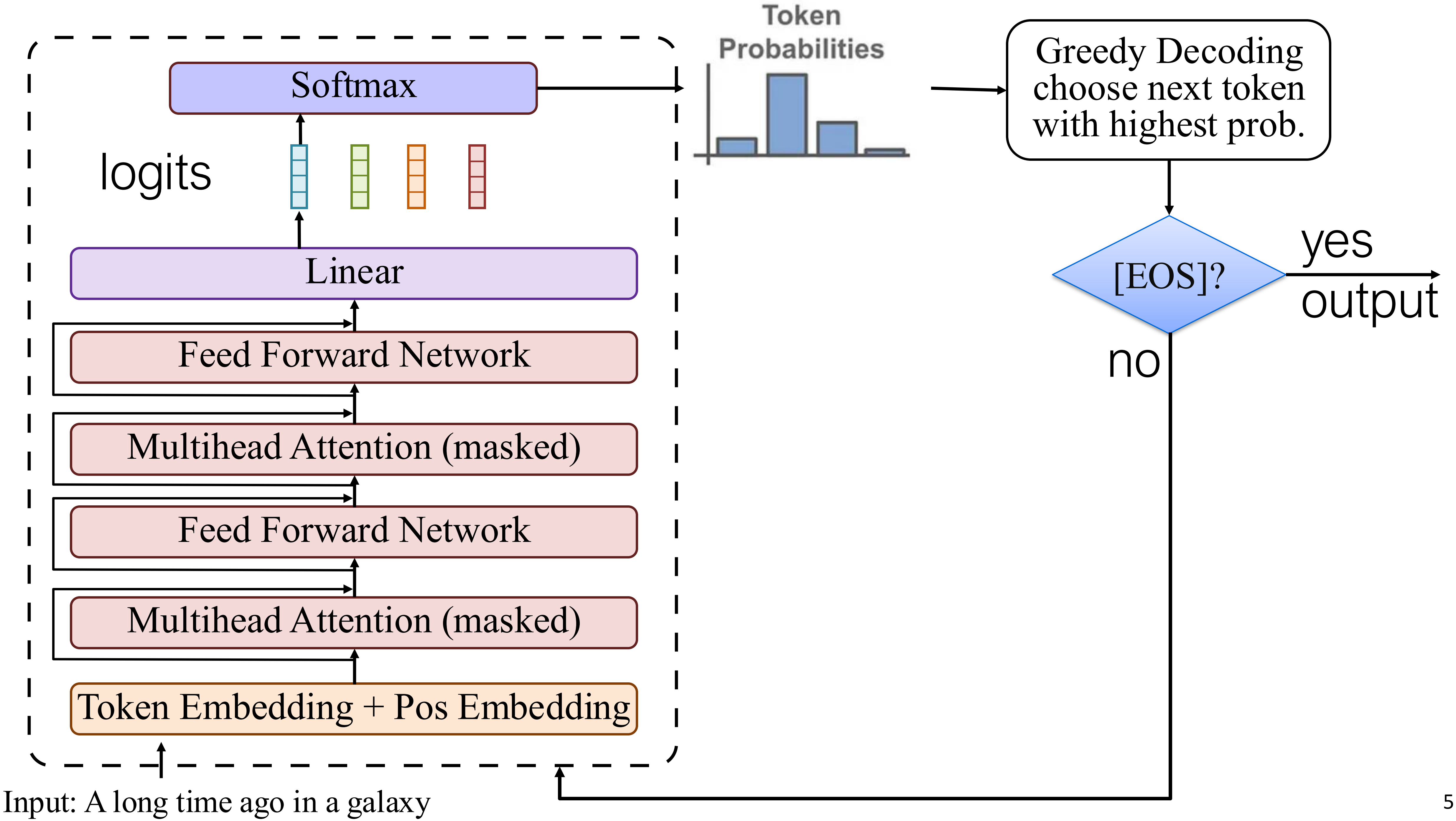
Outline

- Sequence Decoding overview
- Beam search algorithm

Sequence Decoding

$$\arg\max_{y} P(y|\mathrm{x}) = f_{\theta}(x, y)$$

- naive solution: exhaustive search
  - too expensive $\mathrm{O}(V^{N})$
- Greedy (max) decoding
- Sampling
- Beam search
  - (approximate) dynamic programming


Token Probabilities
Greedy Decoding choose next token with highest prob.
Softmax
logits
Linear
Feed Forward Network
Multihead Attention (masked)
Feed Forward Network
Multihead Attention (masked)
Token Embedding + Pos Embedding
[EOS]?
yes
output
no
Input: A long time ago in a galaxy

Max Decoding

- For every next token, pick the one that maximizes the probability

$$\max p(x_t | x_{1\ldots t-1})$$

- equivalent to maximizing logits, no need to normalize

Sampling

- Instead of $\mathrm{argmax}_{\mathrm{y}} P(y|x) = f_\theta(x, y)$

- Generate samples of translation Y from the distribution P(Y|X)

- Q: how to generate samples from a discrete distribution?

Discrete Sampling

- sample n values x’s from k categories, with prob. p1, p2, … pk
- Direct sampling: O(nk)
- Binary Search: O(k + n logk)
- Alias sampling: O(k logk + n)

```
probs = torch.softmax(logits, dim=-1)
next_token = torch.multinomial(probs, num_samples=1)
```

Fast Sampling with Gumbel Max Trick

- sampling from $\text{Categorical}(\text{Softmax}(h))$ is equivalent to

$$\arg\max x$$
$$z \sim \text{Uniform}(0,1)$$
$$x = h - \log(-\log z)$$

- Theory: x follows Gumbel distribution, and argmax x follows Categorical($\frac{\exp h_i}{\sum_{j=1}^{k} h_j}$)

https: //timvieira.github.io/blog/post/2014/ 08/01/gumbel-max-trick-and-weightedreservoir-sampling/

Stochastic Beams and Where to Find Them: The Gumbel-Top-k Trick for Sampling Sequences Without Replacement. Kool et al. ICML 2019

```
class GumbelSampler:
  def __init__(self, batch_size, vocab_size, device):
    self.batch_size = batch_size
    self.vocab_size = vocab_size
    # Pre-compute noise
    self.noise = self._prepare_gumbel_noise(device)

  def _prepare_gumbel_noise(self, device):
    # Generate noise tensor once
    uniform_noise = torch.rand(self.batch_size,
self.vocab_size, device=device)
    return -torch.log(-torch.log(uniform_noise))

  def sample(self, logits):
    # Direct sampling without softmax
    return torch.argmax(logits + self.noise, dim=-1)
```

Outline

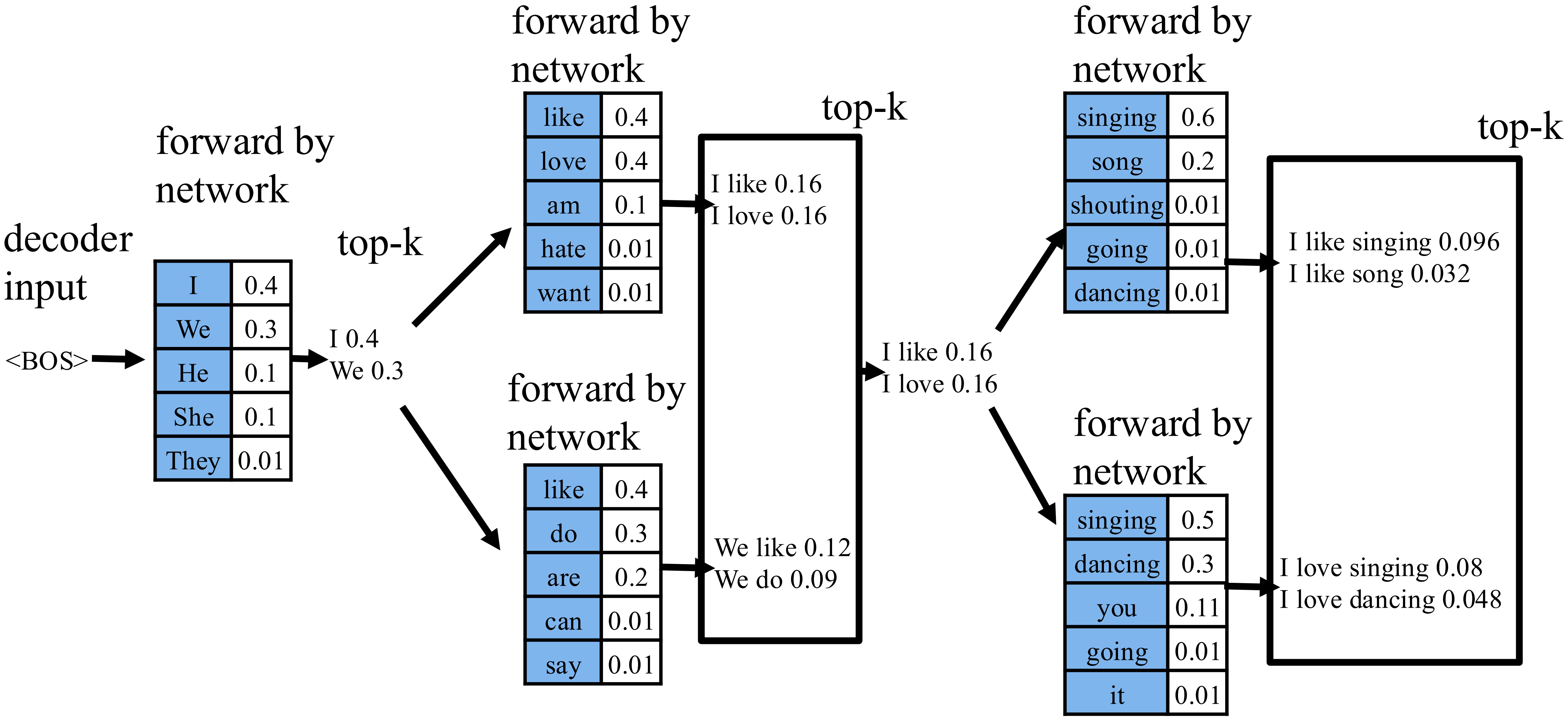
- Sequence Decoding overview
- Beam search algorithm

Beam Search

Find approximate solutions to $\arg\max_{y} P(y|\mathrm{x}) = f_\theta(x, y)$

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 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

```
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:
          new_seqs.get() # pop the one with lowest score
          add (new_candidate, new_score) to new_seqs
```

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 $\in$ C is discarded if: $\text{score}(\text{cand}) \leq rp * \max\{\text{score}(c)\}$

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

- Relative local threshold pruning

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

Combine Sample and Beam Search

- Sample the first tokens
- continue beam search for the later
- why?
    - to improve sequence diversity

Code example

- https://github.com/llmsystem/llmsys_code_examples/blob/main/decoding/decoding.ipynb

Project

- https://llmsystem.github.io/llmsystem2024spring/docs/Projects

- Proposal due: 2/26
  - You are highly encouraged to discuss your project with TAs

- Mid term Report: 4/1

- Poster Project Presentation: 4/24 or 4/25 (depending on room availability)

- Final Report: next day

Project Proposal

- What LLM System problem are you planning to address?
  - what are the **system challenges**?
- What are the existing state-of-art methods on this problem? Is the source code/model available?
- Possible directions for going forward.
- How do you evaluate the performance? what kind of workload?
- Who is your team and how are you planning to split the workload between team members?
- A rough timeline/milestones
- What CPU, GPU and storage infrastructure do need for this project? Please estimate the amount of computation time required.

Project Report Requirement

- Introduction/Motivation: This essentially lays out the problem definition, motivation, talks about why we need to work on it, the key contributions expected/presented in the work.
- Related Work/Background: This talks about key papers/works that provide context to your current work. Instead of listing down multiple past works, talk about the ones that minimally differ from your work, and how.
- Methodology: This section talks about your method, raises research questions and how you are going to address them.
- Experiments: This section can describe your experiments and the results you obtain.
- Analysis/Ablations: Typically, you would have multiple factors involved in your experimental setting. Analysis sections help you probe deeper into the results and help piece out contributions from individual modeling decisions made.
- Conclusion/Discussion: This would list the main takeaways from your work, discuss some future ideas (if any) and engage in discussion.
- Limitations: This section lays out some known limitations of your work.
- [final report only] Team Member Contributions List out each individual's contributions in this section.

Project Team Pairing