Beam Decoding with Controlled Patience

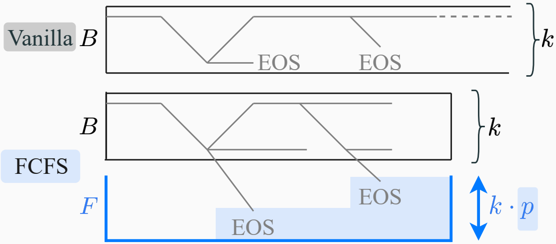
(Kasai et al., 2022)

Hiroyuki Deguchi
deguchi.hiroyuki.db0@is.naist.jp
https://arxiv.org/abs/2204.05424
https://github.com/jungokasai/beam_with_patience
2022/05/11 NAIST MT study group

Introduction

Beam search is widely used, but the implementation has variations.

- **Vanilla**: TensorFlow Addons library
- ***first come, first served; FCFS***: fairseq, Transformers
  - This paper proposes a ***patience factor*** on FCFS beam search.



Vanilla vs. FCFS with patience factor p:
k denotes the beam size. FCFS stores finished sentences in F, but they stay in (and later may fall off from) beam B during vanilla decoding.

Vanilla Beam Decoding

The top-k operation is applied over all sequences.

- Find the top-k sequences, including both finished and unfinished sequences: Line 5
- Decode until all top-k sequences are finished: Line 13
  - It tends to result in deeper search.

Vanilla Beam Decoding

k: beam size, M: maximum length,
$\mathcal{V}$: Vocabulary, $\mathrm{score}(\cdot)$: scoring function.

1: $B_0 \leftarrow \{\langle 0, \textsc{bos} \rangle\}$
2: **for** $t \in \{1, \dots, M-1\}$:
3:   **for** $\langle s, \mathbf{y} \rangle \in B_{t-1}$:
4:     **if** $\mathbf{y}.\mathrm{last}() = \textsc{eos}$:
5:       $H.\mathrm{add}(\langle s, \mathbf{y} \rangle)$
6:       **continue**
7:     **for** $y \in \mathcal{V}$:
8:       $s \leftarrow \mathrm{score}(\mathbf{y} \circ y)$, $H.\mathrm{add}(\langle s, \mathbf{y} \circ y \rangle)$
9:   $B_t \leftarrow \varnothing$
10:   **while** $|B_t| < k$:   # Find top k from H.
11:     $\langle s, \mathbf{y} \rangle \leftarrow H.\mathrm{max}()$, $B_t.\mathrm{add}(\langle s, \mathbf{y} \rangle)$
12:     $H.\mathrm{remove}(\langle s, \mathbf{y} \rangle)$
13:   **if** $\mathbf{y}.\mathrm{last}() = \textsc{eos}, \forall \mathbf{y} \in B_t$:   # All finished.
14:     **return** $B_t.\mathrm{max}()$
15: **return** $B_t.\mathrm{max}()$

First Come, First Served (FCFS)

The top-k operation is applied over unfinished sequences.

- Collect finished sequences in a *first come, first served* manner and removes them from the beam: Line 11
- FCFS has a wider breadth since it collects k unfinished sequences at every step regardless of how many sequences are finished with the EOS symbol.
  - Terminate when a total of k finished sequences is found.

FCFS Beam Decoding with Controlled Patience

k: beam size, M: maximum length, $\mathcal{V}$: Vocabulary
score$(\cdot)$: scoring function, p: patience factor.

1: $B_0 \leftarrow \{\langle 0, \textsc{bos}\rangle\}$, $F_0 \leftarrow \varnothing$
2: **for** $t \in \{1, \ldots, M-1\}$:
3: $\quad H \leftarrow \varnothing$, $F_t \leftarrow F_{t-1}$
4: $\quad$ **for** $\langle s, \mathbf{y}\rangle \in B_{t-1}$: # Expansion.
5: $\quad\quad$ **for** $y \in \mathcal{V}$:
6: $\quad\quad\quad s \leftarrow \text{score}(\mathbf{y} \circ y)$, $H.\text{add}(\langle s, \mathbf{y} \circ y\rangle)$
7: $\quad B_t \leftarrow \varnothing$
8: $\quad$ **while** $|B_t| < k$: # Find top k w/o EOS from H.
9: $\quad\quad \langle s, \mathbf{y}\rangle \leftarrow H.\text{max}()$
10: $\quad\quad$ **if** $\mathbf{y}.\text{last}() = \textsc{eos}$:
11: $\quad\quad\quad F_t.\text{add}(\langle s, \mathbf{y}\rangle)$ # Finished hypotheses.
12: $\quad\quad$ **else** $B_t.\text{add}(\langle s, \mathbf{y}\rangle)$
13: $\quad\quad$ **if** $|F_t| \geq k \cdot p$: # Originally, $p = 1$.
14: $\quad\quad\quad$ **return** $F_t.\text{max}()$
15: $\quad\quad H.\text{remove}(\langle s, \mathbf{y}\rangle)$
16: **return** $F_t.\text{max}()$

Patience Factor for FCFS

Separate the stopping criterion from the search breadth.

- Beam size k in FCFS controls both the breadth and stopping criterion (i.e., depth) of search.
- Proposed **patience factor** relaxes this assumption: Line 13
  - It can be implemented by the one-line modification.

FCFS Beam Decoding with Controlled Patience

k: beam size, M: maximum length, $\mathcal{V}$: Vocabulary
score$(\cdot)$: scoring function, p: patience factor.

1: $B_0 \leftarrow \{\langle 0, \textsc{bos}\rangle\}$, $F_0 \leftarrow \varnothing$
2: **for** $t \in \{1, \dots, M-1\}$:
3: $\quad H \leftarrow \varnothing$, $F_t \leftarrow F_{t-1}$
4: $\quad$ **for** $\langle s, \mathbf{y}\rangle \in B_{t-1}$: # Expansion.
5: $\quad\quad$ **for** $y \in \mathcal{V}$:
6: $\quad\quad\quad s \leftarrow \text{score}(\mathbf{y} \circ y)$, $H.\text{add}(\langle s, \mathbf{y} \circ y\rangle)$
7: $\quad B_t \leftarrow \varnothing$
8: $\quad$ **while** $|B_t| < k$: # Find top k w/o EOS from H.
9: $\quad\quad \langle s, \mathbf{y}\rangle \leftarrow H.\text{max}()$
10: $\quad\quad$ **if** $\mathbf{y}.\text{last}() = \textsc{eos}$:
11: $\quad\quad\quad F_t.\text{add}(\langle s, \mathbf{y}\rangle)$ # Finished hypotheses.
12: $\quad\quad$ **else** $B_t.\text{add}(\langle s, \mathbf{y}\rangle)$
13: $\quad\quad$ **if** $|F_t| \geq k \cdot p$: # Originally, $p = 1$.
14: $\quad\quad\quad$ **return** $F_t.\text{max}()$
15: $\quad\quad H.\text{remove}(\langle s, \mathbf{y}\rangle)$
16: **return** $F_t.\text{max}()$

Main Results

- patience factor: $p = 2$ for MT, $p = 0.5$ for summarization
- metric: COMET (XLM-RoBERTa) for MT, ROUGE for summarization

	WMT 2020/2021 Machine Translation ($p\!=\!2$)								Summarization ($p\!=\!0.5$)					
	EN↔DE		**EN↔JA**		**EN↔PL**		**EN↔ZH**		**CNNDM**			**XSUM**		
Algorithm	→	←	→	←	→	←	→	←	R-2	R-3	R-L	R-2	R-3	R-L
Greedy	43.7	66.2	33.6	9.5	46.0	53.5	32.5	23.5	21.1	11.9	30.7	19.8	10.7	34.3
Vanilla	48.2	66.3	**38.7**	**15.7**	52.7	58.2	**33.9**	29.9	19.2	11.0	28.0	19.5	10.7	33.1
FCFS	47.9	66.2	38.0	15.0	52.1	58.1	33.7	29.6	20.4	11.6	30.3	20.4	11.4	34.4
FCFS w/ p	**48.3**	**66.4**	38.4	15.6	**53.0**	**58.4**	33.8	**30.2**	**21.4**	**12.4**	**31.2**	**21.0**	**11.8**	**35.4**

- In summarization, *vanilla* decoding performs worse than *greedy*.
  - It might be a reason why *FCFS* is used instead of *vanilla* in popular libraries.

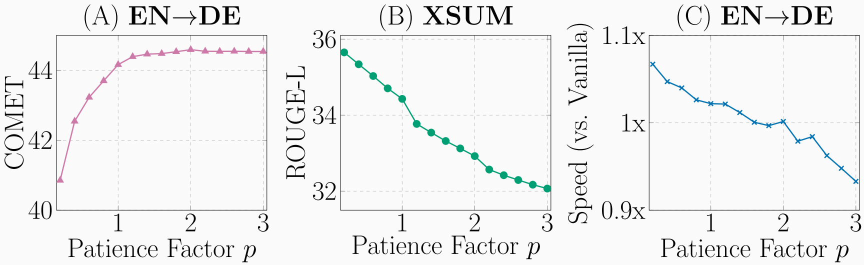
Evaluate Translation by BLEU

- patience factor: $p = 2$
- metric: BLEU

	WMT 2020 and 2021 Machine Translation (BLEU)							
	EN↔DE		**EN↔JA**		**EN↔PL**		**EN↔ZH**	
Algorithm	→	←	→	←	→	←	→	←
Greedy	42.9	46.6	20.2	17.4	19.8	30.7	31.2	21.7
Vanilla	**45.1**	48.4	21.6	19.7	**21.1**	**32.5**	**32.5**	23.6
FCFS	45.0	48.4	21.3	19.5	21.0	32.4	**32.6**	23.4
FCFS w/ p	45.0	**48.5**	**21.7**	**19.8**	**21.1**	**32.5**	32.3	**23.7**

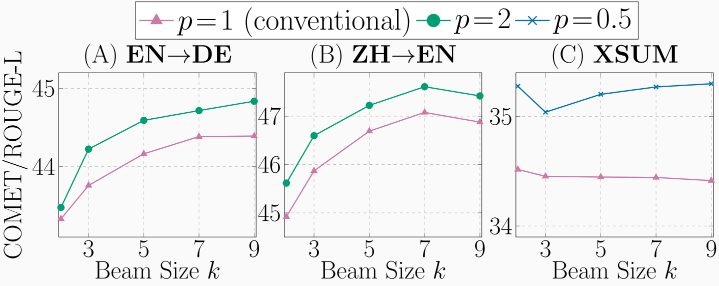
- In many cases, *FCFS w/*p slightly better than *FCFS*.

Analysis: Effects of varying patience factors p



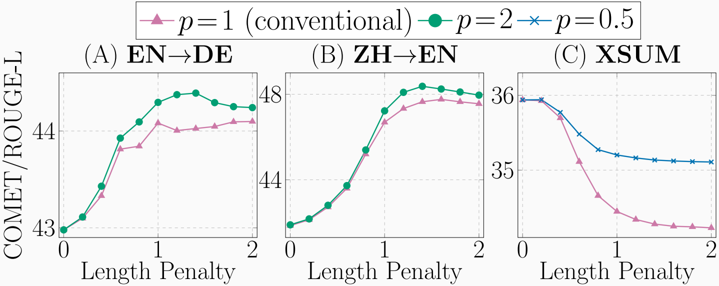
- The translation performance improves with larger patience factors with diminishing gains.
- Summarization benefits more from patience factors smaller than the original value of 1.
  - The nature of the summarization task that aims to generate concise text.

Analysis: Effects of controlled patience over varying beam sizes



- The amount of improvement changes, but the patience factor is generally beneficial.

Analysis: Effects of controlled patience over varying length penalty



- The amount of improvement changes, but the patience factor is generally beneficial.

[BTW] Length Penalty

- Length-normalized log probability:

$$\frac{\log(P(Y|X))}{\mathrm{lenpen}(Y)}$$

  - Google NMT: $\mathrm{lenpen}(Y) = \left(\frac{5+|Y|}{5+1}\right)^{\alpha}$, $\alpha = 0.6$
  - Fairseq: $\mathrm{lenpen}(Y) = |Y|^{\alpha}$, $\alpha = 1.0$
    - This paper uses the fairseq version's length penalty.

Related Work

Stopping Criterion for Beam Decoding

- Optimal Beam Search (Huang et al., EMNLP 2017)[1]
  - They propose a method to optimally finish beam search.
  - Modify the beam decoding procedure

Breadth of Beam Decoding

- Analyzing effects of the search breadth (Ott et al., ICML 2018)[2]

---

1. ``When to Finish? Optimal Beam Search for Neural Text Generation (modulo beam size)", Huang et al., 2017.
2. ``Analyzing Uncertainty in Neural Machine Translation", Ott et al., 2018.

Conclusion

- This paper named two major beam search implementation, vanilla and FCFS.
- This paper proposed **patience factor** for FCFS, controls the stopping criterion.
- Experiments shows that the proposed FCFS w/patience gains better translation/summarization performance than FCFS.