Beam Decoding with Controlled Patience

Anonymous ACL submission

Abstract

Text generation with beam search has proven successful in a wide range of applications. The commonly-used implementation of beam decoding follows a *first come, first served* heuristic: it keeps a set of already completed sequences over time steps and stops when the size of this set reaches the beam size. We introduce a *patience factor*, a simple modification to this decoding algorithm, that generalizes the stopping criterion and provides flexibility to the depth of search. Extensive empirical results demonstrate that the patience factor improves decoding performance of strong pretrained models on news text summarization and machine translation over diverse language pairs, with a negligible inference slowdown. Our approach only modifies one line of code and can be thus readily incorporated in any implementation.[1]

FCFS Beam Decoding with Controlled Patience

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

$B_0 \leftarrow \{\langle 0, \textsc{bos} \rangle\}$, $F_0 \leftarrow \varnothing$
for $t \in \{1, \dots, M-1\}$:
  $H \leftarrow \varnothing$, $F_t \leftarrow F_{t-1}$
  for $\langle s, \mathbf{y} \rangle \in B_{t-1}$: # Expansion.
    for $y \in \mathcal{V}$:
      $s \leftarrow \text{score}(\mathbf{y} \circ y)$, $H.\text{add}(\langle s, \mathbf{y} \circ y \rangle)$
  $B_t \leftarrow \varnothing$
  while $|B_t| < k$: # Find top k w/o EOS from H.
    $\langle s, \mathbf{y} \rangle \leftarrow H.\text{max}()$
    if $\mathbf{y}.\text{last}() = \textsc{eos}$:
      $F_t.\text{add}(\langle s, \mathbf{y} \rangle)$ # Finished hypotheses.
    else $B_t.\text{add}(\langle s, \mathbf{y} \rangle)$
    if $|F_t| \geq k \cdot p$: # Originally, $p = 1$.
      return $F_t.\text{max}()$
    $H.\text{remove}(\langle s, \mathbf{y} \rangle)$
return $F_t.\text{max}()$

Figure 1: First come, first served (FCFS) beam decoding with patience factor p. The common implementation can be considered as a special case where $p = 1$. The highlighted line is the *only* modification that this work introduces for performance improvement. F_t: already completed sequences; B_t: beam of continuing sequences. H_t: expanded hypotheses before the top-k operation. The input sequence to score is omitted.

1 Introduction

Beam search has become a dominant inference algorithm for a wide range of language generation tasks, such as machine translation (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017), summarization (Nallapati et al., 2016; See et al., 2017), and image captioning (Anderson et al., 2018; Li et al., 2020). Beam decoding[2] is an approximate, pruned version of breadth-first search that seeks the highest-probability sequence under an autoregressive (left-to-right) language generation model. In this work, we examine a popular implementation of beam decoding and propose a simple modification (one line of code) that improves the decoding performance of strong, neural language generation models (Fig. 1).

A widely-used implementation of beam language decoding (e.g., `fairseq`, Ott et al., 2019; Hugging Face's `Transformers`, Wolf et al., 2020)[3] follows a *first come, first served* (FCFS) heuristic: when a total of k finished candidates is found (k is the beam size), it returns the best one from the k candidates and discards all of the current, unfinished k sequences in the beam. Beam size k thus determines both the breadth and depth of search. We propose a *patience factor* (Fig. 1) that decomposes these two roles and controls how many finished candidates have to be found before terminating the decoding. The patience factor generalizes the commonly-used implementation and provides flexibility in the depth of beam search by changing

[1] Our codebase is available at `anonymized`.

[2] In this paper, we use "beam decoding" to mean beam search applied to decoding for text generation.

[3] `https://github.com/pytorch/fairseq/blob/main/fairseq/sequence_generator.py`; `https://github.com/huggingface/transformers/blob/master/src/transformers/generation_utils.py`.

the stopping criterion.

We apply the one-line modification to strong off-the-shelf transformer models without any change to the trained models for machine translation (Tang et al., 2021) and text summarization (Lewis et al., 2020). Our experiments demonstrate that our method outperforms the original algorithm on the CNN/Dailymail (Hermann et al., 2015) and XSUM (Narayan et al., 2018) news summarization tasks and the WMT 2020/2021 machine translation tasks (Barrault et al., 2020; Akhbardeh et al., 2021) across diverse language pairs. Further, the introduction of the patience factor only results in a negligible inference slowdown, confirming its practical advantage in downstream applications.

Our analysis shows that, while the performance gain is sensitive to hyperparameters of beam decoding (beam size and length penalty; Johnson et al., 2017), the patience factor is consistently beneficial. Moreover, we extensively compare our results with the *vanilla* implementation of beam search that much prior work assumes (Meister et al., 2020b; Stahlberg and Byrne, 2019, *inter alia*). Empirically, we found that the vanilla algorithm performs competitively with FCFS on machine translation but substantially underperforms on summarization. The FCFS beam decoding with our patience factor is thus a simple yet effective algorithm for both language generation tasks.

2 Beam Decoding with Patience

Vanilla and FCFS Implementations Beam decoding has been applied to sequence-to-sequence models (Graves, 2012; Boulanger-Lewandowski et al., 2013a,b), and it is now used in many state-of-the-art systems for language generation tasks (Zhang et al., 2020, 2021; Tran et al., 2021; Raffel et al., 2020, *inter alia*). Figs. 1 and 2 describe its two major implementations. They differ primarily in the treatment of finished sequences with the EOS symbol at the end: FCFS collects finished sequences in a *first come, first served manner* and removes them from the beam (Line 11, Fig. 1), whereas the vanilla version finds the top k sequences, including both finished and unfinished sequences (Line 5 in Fig. 2). While often unspecified in the literature, our later experiments in §3.2 will show that this difference can affect the downstream performance substantially, especially on news text summarization.

Vanilla Beam Decoding

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

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

Figure 2: The vanilla version of beam decoding. The top-k operation is applied over H, the union of the finished and continuing sequences. This is implemented, for example, in the TensorFlow Addons library (Abadi et al., 2015).[4] See also Stahlberg and Byrne (2019); Meister et al. (2020b).

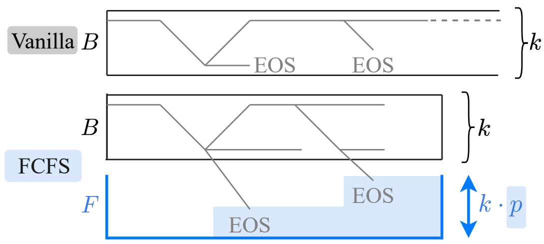

Figure 3: FCFS with patience factor p vs. vanilla beam decoding. k denotes the beam size. FCFS stores finished sentences in F, but they stay in (and later may fall off from) beam B during vanilla decoding. $k \cdot p$ determines the size of F. The illustration of beam decoding here is inspired by Huang et al. (2012).

Further comparing Figs. 1 and 2, we see their difference in terms of the breadth and depth of search. Given the same beam size k, 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.[5] The vanilla algorithm decodes until all top-k sequences are finished (Line 13, Fig. 2), and therefore it tends to result in deeper search. FCFS, in contrast, terminates when a total of k finished sequences is found.

Patience Factor for FCFS Beam size k in FCFS

[4] https://www.tensorflow.org/addons/api_docs/python/tfa/seq2seq/BeamSearchDecoder.

[5] In practice, this is implemented by taking the top $2k$ sequences at every step. We find at most k EOS symbols, so there are always at least k *unfinished* sequences. See https://github.com/huggingface/transformers/.

thus controls both the breadth and stopping criterion (i.e., depth) of search. We introduce the patience factor (Line 13, Fig. 1) that relaxes this assumption and separates the stopping criterion from the search breadth. Fig. 3 illustrates this patience factor as well as the difference between the FCFS and vanilla algorithms. The one-line change generalizes FCFS ($p=1$) and adds flexibility. We will show that this flexibility is beneficial on machine translation and summarization (§3.2).

3 Experiments

We present extensive comparisons of beam decoding variants on text summarization and machine translation over a wide range of language pairs. Our simple addition of the patience factor improves performance across the board.

3.1 Experimental Setup

We evaluate four decoding algorithms on machine translation and summarization: **greedy**, **vanilla**, **FCFS**, and **FCFS with the patience factor**. For machine translation, we use multilingual BART (Tang et al., 2021), a strong, pretrained transformer model,[6] and WMT 2020/2021 news test data (Barrault et al., 2020; Akhbardeh et al., 2021) for four diverse language pairs (eight directions): WMT 2020 for EN↔PL (Polish) and 2021 for EN↔DE (German), EN↔JA (Japanese), and EN↔ZH (Chinese). We apply beam decoding with the same hyperparameters as Tang et al. (2021): beam size 5 and length penalty 1. We measure performance with the COMET score (Rei et al., 2020a,b), a state-of-the-art evaluation metric based on multilingual contextual representations. For summarization, we experiment with the CNN/Dailymail (CNNDM, Hermann et al., 2015) and XSUM (Narayan et al., 2018) datasets. We apply the off-the-shelf BART models (Lewis et al., 2020) that are finetuned on each dataset.[7] Performance is measured with ROUGE scores (Lin, 2004). We follow the original setting in Lewis et al. (2020): beam sizes 4 and 6 and length penalty 2 and 1 for CNNDM and XSUM, respectively. More experimental details are described in Appendix §A.

We experiment with the same patience factor on all datasets for each task, based on our preliminary development: $p = 2$ for machine translation and $p = 0.5$ for summarization. Here we avoid additional effort and demonstrate the practical value of our simple modification. We present detailed sensitivity analysis over p in §3.3.

3.2 Results

Seen in Table 1 are results from our experiments. FCFS with the patience factor outperforms the widely-used FCFS algorithm across the board; e.g., 53.0 vs. 52.1 on EN→PL. Particularly noteworthy are the performance gains on the two summarization datasets; e.g., 31.2 vs. 30.3 ROUGE-L on CNNDM. Comparing vanilla decoding and FCFS, we see that the former outperforms the latter (and is competitive with or slightly better than FCFS w/ p) on machine translation but underperforms substantially on summarization; e.g., 34.4 vs. 33.1 ROUGE-L on XSUM. Vanilla decoding even performs worse than greedy decoding in many cases. We suspect this performance degradation on summarization might be a reason why FCFS is used instead of vanilla decoding in popular libraries.

3.3 Analysis

Here we use the standard dev. split from the XSUM dataset and news test 2020 EN→DE and ZH→EN data. We fixed the value of p for each task so far, but Fig. 4 explores varying patience factors and their effects on the performance (A: EN→DE; B: XSUM) and the inference speed (C). The translation performance improves with larger patience factors with diminishing gains. On the other hand, summarization benefits more from patience factors smaller than the original value of 1, possibly due to issues in the scoring function (Wiseman and Rush, 2016) or ROUGE evaluations (Nenkova, 2006) and the nature of the summarization task that aims to generate concise text. Note, however, that we see consistent patterns with ROUGE from COMET (Rei et al., 2020b), which achieves the highest correlation to human judgment on CNNDM (Kasai et al., 2022a; see Table 3 in the appendix). Regardless, our patience factor provides useful flexibility for any generation task.

As expected, generation slows down as p increases (Fig. 4C). The inference slowdown from around $p=2$ is still negligible, again showing the practicality of our method. Fig. 5 explores the performance gains from the patience factor over varying beam sizes. The amount of improvement changes, but the patience factor is generally beneficial. We see similar trends for various values of

[6] https://github.com/pytorch/fairseq/tree/main/examples/multilingual#mbart50-models.

[7] https://github.com/pytorch/fairseq/tree/main/examples/bart.

	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**

Table 1: We evaluate the four inference algorithms on the machine translation and news summarization test data with the COMET score (Rei et al., 2020b) and ROUGE scores (ROUGE-2/3/L), respectively. FCFS w/ p indicates our FCSF algorithm with the patience factor ($p=2$ for machine translation and $p=0.5$ for summarization). COMET uses crosslingual contextual representations from XLM-RoBERTa (Conneau et al., 2020) and has shown to have significantly higher correlation with expert human judgment than alternatives (Mathur et al., 2020b; Kasai et al., 2022a) like BLEU (Papineni et al., 2002). Nonetheless, we see consistent patterns from BLEU (Appendix §B). For CNNDM, we used 100 test articles with 10 human-written references each from Kryscinski et al. (2019).

the length penalty (see Fig. 6 in the appendix).

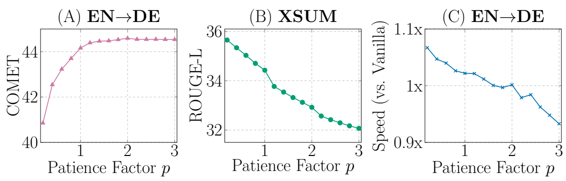

Figure 4: Effects of varying patience factors p on the dev. score (A and B) and inference speed (C). The inference speed is measured with batch size 20, relative to the vanilla decoding algorithm on the same single Nvidia A100-SXM GPU. Other languages pairs were similar to EN→DE (A). CNNDM also had similar trends to XSUM (B).

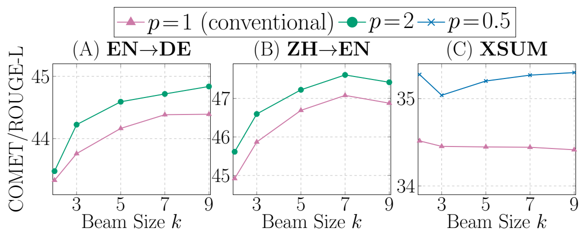

Figure 5: Effects of controlled patience on the dev. data over varying beam sizes. The length penalty value is 1. We evaluate with COMET for machine translation and ROUGE-L for XSUM summarization.

4 Further Related Work

Stopping Criteria for Beam Decoding The patience factor changes the stopping criterion and adds flexibility in the search depth of the common beam search algorithm. Similarly, several prior works studied stopping criteria to improve machine translation (Huang et al., 2017; Yang et al., 2018; Ma et al., 2019). Our machine translation experiments are consistent with their findings: stopping criteria that yield accurate search improve performance. In the case of summarization, however, we observed that less patient and thus *less accurate* search can improve ROUGE scores.

Breadth of Beam Decoding Much prior work explored downstream effects of the search breadth (Koehn and Knowles, 2017; Murray and Chiang, 2018; Ott et al., 2018; Cohen and Beck, 2019; Stahlberg and Byrne, 2019, *inter alia*). Beam decoding with larger beam sizes can find sequences with higher scores but lead to performance degradation (often called the *beam search curse*; Yang et al., 2018). Recent work (Meister et al., 2020a) argued that beam decoding with small beams introduces bias that is related to the uniform information density of human-produced text (Levy, 2005). Freitag and Al-Onaizan (2017) proposed a method to adaptively shrink the beam width based on the partial scores to speed up inference. This work focused on the stopping criteria (i.e., depth) and separated them from the breadth of the commonly-used beam decoding.

5 Conclusion

We introduced the patience factor that generalizes the widespread implementation of beam text decoding. Our extensive experiments showed that the patience factor improves the generation performance of strong, off-the-shelf models on machine translation and summarization with an insignificant slowdown in generation. As it only requires a minimal change in code, we hope that many researchers and practitioners of language generation will benefit from our simple yet effective modification.

Limitations and Ethical Considerations

We evaluated our decoding method both on machine translation and news summarization. Our machine translation experiments span diverse languages, including morphologically rich languages (e.g., Japanese and Polish) and languages with non-Latin scripts (e.g., Japanese and Chinese). Nonetheless, our summarization experiments are limited to English and the news domain mainly due to our budget constraints. There are also many other language generation tasks for which our method can be useful. Since our improvement only requires one line of code, we hope that practitioners will implement it for the domain and the task of their interest and further assess how our decoding algorithm performs over a wider range of applications.

Evaluating language generation remains a challenging research problem. We carefully set up our experiments to mitigate potential evaluation issues. The WMT 2020/2021 test data consist only of news text written in the original language, in contrast to the test data from WMT 2018 (Bojar et al., 2018) or earlier. For example, the WMT 2021 EN→DE and DE→EN test data come from completely different documents. This avoids the *translationese effect* that would overestimate the translation performance due to the simplicity of translated text (Graham et al., 2020). Moreover, some language pairs in the WMT 2020 and 2021 test data have multiple references per instance, which increases the correlation of automatic evaluations with human judgment (Kasai et al., 2022a). We presented results using automatic metrics from recent work (Rei et al., 2020b) as well as conventional, n-gram overlap metrics (Papineni et al., 2002; Lin, 2004). Recent automatic metrics have shown to have higher correlation with human judgements, but human judgments are sometimes inconsistent, especially when crowdsourced (Clark et al., 2021; Kasai et al., 2022b). Since our decoding method is a generalization of the widely-used beam search algorithm, we hope that it will be tested and used in real-world systems of language generation.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proc. of WMT*.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proc. of CVPR*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proc. of WMT*.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proc. of WMT*.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2013a. Audio chord recognition with recurrent neural networks. In *Proc. of ISMIR*.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2013b. High-dimensional sequence transduction. In *Proc. of ICASSP*.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proc. of ACL*.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proc. of ICML*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proc. of ACL*.

Sergey Edunov, Myle Ott, Marc'Aurelio Ranzato, and Michael Auli. 2020. On the evaluation of machine translation systems trained with back-translation. In *Proc. of ACL*.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proc. of NGT*.

Yvette Graham, Barry Haddow, and Philipp Koehn. 2020. Translationese in machine translation evaluation.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. In *Representation Learning Workshop*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proc. of NeurIPS*.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proc. of NAACL*.

Liang Huang, Kai Zhao, and Mingbo Ma. 2017. When to finish? optimal beam search for neural text generation (modulo beam size). In *Proc. of EMNLP*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL*.

Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Lavinia Dunagan, Jacob Morrison, Alexander R. Fabbri, Yejin Choi, and Noah A. Smith. 2022a. Bidimensional leaderboards: Generate and evaluate language hand in hand. In *Proc. of NAACL*.

Jungo Kasai, Keisuke Sakaguchi, Lavinia Dunagan, Jacob Morrison, Ronan Le Bras, Yejin Choi, and Noah A. Smith. 2022b. Transparent human evaluation for image captioning. In *Proc. of NAACL*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proc. of NGT*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proc. of EMNLP*.

Roger Levy. 2005. *Probabilistic Models of Word Order and Syntactic Discontinuity*. Ph.D. thesis, Stanford University.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Proc. of ECCV*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proc. of Text Summarization Branches Out*.

Mingbo Ma, Renjie Zheng, and Liang Huang. 2019. Learning to stop in structured prediction for neural machine translation. In *Proc. of NAACL*.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020a. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proc. of ACL*.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020b. Results of the WMT20 metrics shared task. In *Proc. of WMT*.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020a. If beam search is the answer, what was the question? In *Proc. of EMNLP*.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020b. Best-first beam search. *TACL*.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proc. of WMT*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proc. of CoNLL*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proc. of EMNLP*.

Ani Nenkova. 2006. Summarization evaluation for text and speech: issues and approaches. In *Proc. of INTERSPEECH*.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *Proc. of ICML*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. of NAACL: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proc. of WMT*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020a. COMET: A neural framework for MT evaluation. In *Proc. of EMNLP*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020b. Unbabel's participation in the WMT20 metrics shared task. In *Proc. of WMT*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proc. of ACL*.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proc. of EMNLP*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proc. of NeurIPS*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of ACL*.

Chau Tran, Shruti Bhosale, James Cross, Philipp Koehn, Sergey Edunov, and Angela Fan. 2021. Facebook AI's WMT21 news translation task submission. In *Proc. of WMT*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proc. of EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP: System Demonstrations*.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-)scoring methods and stopping criteria for neural machine translation. In *Proc. of EMNLP*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proc. of ICML*.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. VinVL: Making visual representations matter in vision-language models. In *Proc. of CVPR*.

Appendices

Hyperparameter	Value
WMT Machine Translation (All Pairs)	
beam size	5
length penalty	1
CNNDM Summarization	
beam size	4
length penalty	2
max-len-b	140
min-len	55
no-repeat-ngram-size	3
XSUM Summarization	
beam size	6
length penalty	1
max-len-b	60
min-len	10
no-repeat-ngram-size	3

Table 2: Beam decoding hyperparameters. We generally followed prior work: Tang et al. (2021) for machine translation and Lewis et al. (2020) for CNNDM and XSUM summarization.

A Beam Decoding Hyperparameters

Table 2 shows the beam decoding hyperparameters in our experiments. We generally follow the original settings of the pretrained, off-the-shelf models (Tang et al., 2021; Lewis et al., 2020).

B Additional Results

Table 3 reports BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020b) scores for the machine translation and summarization experiments, respectively. We use the `sacreBLEU` implementation for BLEU (Post, 2018). Note that much recent work (Mathur et al., 2020a; Kasai et al., 2022a,b; Edunov et al., 2020, *inter alia*) found poor correlation between BLEU scores and human judgment for evaluating strong language generation models. COMET is an automatic metric for machine translation that uses crosslingual contextual representations from XLM-RoBERTa (Conneau et al., 2020), but it can be used *monolingually* for evaluating summarization as well (Kasai et al., 2022a).

Fig. 6 explores the performance gains from the patience factor over varying length penalty values. Consistent with the trends from various beam sizes (Fig. 5), the amount of improvement changes, but the patience factor is generally beneficial.

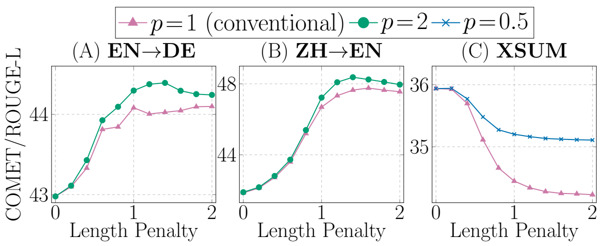

Figure 6: Effects of controlled patience on the dev. data over varying length penalty values. The beam sizes are all 5. We evaluate with COMET for machine translation and ROUGE-L for XSUM summarization.

	WMT 2020 and 2021 Machine Translation (BLEU)								Summarization	
	EN↔DE		**EN↔JA**		**EN↔PL**		**EN↔ZH**		**CNNDM**	**XSUM**
Algorithm	→	←	→	←	→	←	→	←	COMET	COMET
Greedy	42.9	46.6	20.2	17.4	19.8	30.7	31.2	21.7	**1.6**	0.1
Vanilla	**45.1**	48.4	21.6	19.7	**21.1**	**32.5**	**32.5**	23.6	-5.5	-1.6
FCFS	45.0	48.4	21.3	19.5	21.0	32.4	**32.6**	23.4	-4.2	2.2
FCFS w/ p	45.0	**48.5**	**21.7**	**19.8**	**21.1**	**32.5**	32.3	**23.7**	-1.1	**2.5**

Table 3: We evaluate the four decoding algorithms on machine translation and summarization and report BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020b) scores here. FCFS w/ p indicates our FCSF algorithm with the patience factor ($p=2$ for machine translation and $p=0.5$ for summarization). COMET is an automatic metric for machine translation that uses crosslingual contextual representations from XLM-RoBERTa (Conneau et al., 2020), but it can be used for evaluating summarization as well (Kasai et al., 2022a).