When to Finish? Optimal Beam Search for Neural Text Generation (modulo beam size)

Liang Huang and **Kai Zhao**[†] and **Mingbo Ma**
School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, Oregon, USA
{liang.huang.sh, kzhao.hf, cosmmb}@gmail.com

Abstract

In neural text generation such as neural machine translation, summarization, and image captioning, beam search is widely used to improve the output text quality. However, in the neural generation setting, hypotheses can finish in different steps, which makes it difficult to decide when to end beam search to ensure optimality. We propose a provably optimal beam search algorithm that will always return the optimal-score complete hypothesis (modulo beam size), and finish as soon as the optimality is established (finishing no later than the baseline). To counter neural generation's tendency for shorter hypotheses, we also introduce a bounded length reward mechanism which allows a modified version of our beam search algorithm to remain optimal. Experiments on neural machine translation demonstrate that our principled beam search algorithm leads to improvement in BLEU score over previously proposed alternatives.

1 Introduction

In recent years, neural text generation using recurrent networks have witnessed rapid progress, quickly becoming the state-of-the-art paradigms in machine translation (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2014), summarization (Rush et al., 2015; Ranzato et al., 2016), and image captioning (Vinyals et al., 2015; Xu et al., 2015). In the decoder of neural generation, beam search is widely employed to boost the output text quality, often leading to substantial improvement over greedy search (equivalent to beam size 1) in metrics such as BLEU or ROUGE; for example, Ranzato et al. (2016) reported +2.2 BLEU (on single reference) in translation and +3.5 ROUGE-2 in summarization, both using a beam of 10. Our own experiments on machine translation (see Sec. 5) show +4.2 BLEU (on four references) using a beam of 5.

However, unlike traditional beam search in phrase-based MT or shift-reduce parsing where all hypotheses finish in the same number of steps, here in neural generation, hypotheses can finish in vastly different numbers of steps. Once you find a completed hypothesis (by generating the </s> symbol), there are still other active hypotheses in the beam that can continue to grow, which might lead to better scores. Therefore when can you end the beam search? How (and when) can you guarantee that the returned hypothesis has the optimal score modulo beam size?

There have not been satisfying answers to these questions, and existing beam search strategies are heuristic methods that do not guarantee optimality. For example, the widely influential RNNsearch (Bahdanau et al., 2014) employs a "shrinking beam" method: once a completed hypothesis is found, beam size shrinks by 1, and beam search would finish if beam size shrinks to 0 or if the number of steps hits a hard limit. The best scoring completed hypothesis among all completed ones encountered so far is returned. On the other hand, OpenNMT (Klein et al., 2017), whose PyTorch version will be the baseline in our experiments, uses a very different strategy: beam search terminates whenever the highest-ranking hypothesis in the current step is completed (which is also the one returned), without considering any other completed hypotheses. Neither of these two methods guarantee optimality of the returned hypothesis.

We therefore propose a novel and simple beam search variant that will always return the optimal-score complete hypothesis (modulo beam size), and finish as soon as the optimality is established.

[†] Current address: Google Inc., New York, NY, USA.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2124–2129
Copenhagen, Denmark, September 7–11, 2017.

However, another well-known problem remains, that the generated sentences are often too short, compared to previous paradigms such as SMT (Shen et al., 2016). To alleviate this problem, previous efforts introduce length normalization (as a switch in RNNsearch) or length reward (He et al., 2016) borrowed from SMT (Koehn et al., 2007). Unfortunately these changes will invalidate the optimal property of our proposed algorithm. So we introduce a *bounded* length reward mechanism which allows a modified version of our beam search algorithm to remain optimal. Experiments on neural machine translation demonstrate that our principled beam search algorithm leads to improvement in BLEU score over previously proposed alternatives.

2 Neural Generation and Beam Search

Here we briefly review neural text generation and then review existing beam search algorithms.

Assume the input sentence, document, or image is embedded into a vector $\mathbf{x}$, from which we generate the output sentence $\mathbf{y}$ which is a *completed* hypothesis:[1]

$$\begin{aligned}\mathbf{y}^* &= \underset{\mathbf{y}:comp(\mathbf{y})}{\mathbf{argmax}}\, p(\mathbf{y} \mid \mathbf{x}) \\ &= \underset{\mathbf{y}:comp(\mathbf{y})}{\mathbf{argmax}} \prod_{i \le |\mathbf{y}|} p(y_i \mid \mathbf{x}, \mathbf{y}_{<i})\end{aligned}$$

where $\mathbf{y}_{<i}$ is a popular shorthand notation for the prefix $y_0 y_1 ... y_{i-1}$. We say that a hypothesis $\mathbf{y}$ is **completed**, notated $comp(\mathbf{y})$, if its last word is </s>, i.e.,

$$comp(\mathbf{y}) \triangleq (\mathbf{y}_{|\mathbf{y}|} = \text{</s>})$$

in which case it will not be further expanded.

A crucial difference in RNN-based neural generation compared to previous paradigms such as phrase-based MT is that we no longer decompose $p(y_i \mid \mathbf{x}, \mathbf{y}_{<i})$ into the translation model, $p(y_i \mid \mathbf{x})$, and the language model, $p(y_i \mid \mathbf{y}_{<i})$, and more importantly, we no longer approximate the latter by n-gram models. This ability to model arbitrarily-lengthed history using RNNs is an important reason for NMT's substantially improved fluency compared to SMT.

To (approximately) search for the best output $\mathbf{y}^*$, we use beam search, where the beam B_i at step i is an *ordered* list of size (at most) b, and expands to the next beam B_{i+1} of the same size:

$$B_0 = [\langle \text{<s>},\ p(\text{<s>} \mid \mathbf{x}) \rangle]$$

$$B_i = \overset{b}{\mathbf{top}} \{ \langle \mathbf{y}' \circ y_i,\ s \cdot p(y_i | \mathbf{x}, \mathbf{y}) \rangle \mid \langle \mathbf{y}', s \rangle \in B_{i-1} \}$$

where the notation $\mathbf{top}^b\, S$ selects the top b scoring items from the set S, and each item is a pair $\langle \mathbf{y}, s \rangle$ where $\mathbf{y}$ is the current prefix and s is its accumulated score (i.e., product of probabilities).

3 Optimal Beam Search (modulo beam size)

We propose a very simple method to optimally finish beam search, which guarantees the returned hypothesis is the highest-scoring completed hypothesis modulo beam size; in other words, we will finish as soon as an "optimality certificate" can be established that future hypotheses will never score better than the current best one.

Let $best_{\le i}$ be the *best completed hypothesis so far* up to step i, i.e.,

$$best_{\le i} \triangleq \max \{ \mathbf{y} \in \cup_{j \le i} B_j \mid comp(\mathbf{y}) \} \quad (1)$$

We update it every time we find a completed hypothesis (if there is none yet, then it remains undefined). Now at any step i, if $best_{\le i}$ is defined, and the highest scoring item $B_{i,1}$ in the current beam B_i scores worse than or equal to $best_{\le i}$, i.e., when

$$B_{i,1} \le best_{\le i} \quad (2)$$

we claim the optimality certificate is established, and terminate beam search, returning $best_{\le i}$ (here smaller means worse, since we aim for the highest-probability completed hypothesis).

Theorem 1 (optimality)**.** *When our beam search algorithm terminates, the current best completed hypothesis (i.e., $best_{\le i}$) is the highest-probability completed hypothesis (modulo beam size).*

Proof. If $B_{i,1} \le best_{\le i}$ then $B_{i,j} \le B_{i,1} \le best_{\le i}$ for all items $B_{i,j}$ in beam B_i. Future descendants grown from these items will only be no better, since probability ≤ 1, so all items in current and future steps are no better than $best_{\le i}$. □

Theorem 2 (early stopping)**.** *Our beam search algorithm terminates no later than OpenNMT's termination criteria (when $B_{i,1}$ is completed).*

Proof. When $B_{i,1}$ is itself completed, $best_{\le i} = \max\{B_{i,1}, \cdots\} \ge B_{i,1}$, so our stopping criteria is also met. □

[1] For simplicity reasons we do not discuss bidirectional LSTMs and attentional mechanisms here but our algorithms still work with those encoders (we have tested them).

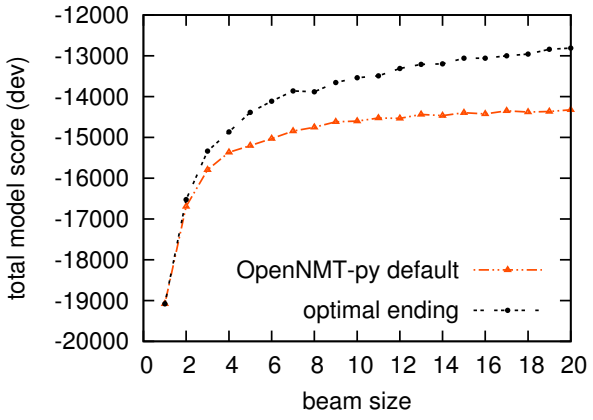


Figure 1: Comparison between optimal beam search and OpenNMT-py's default search, in terms of search quality (model score, ↑ is better).

This above Theorem shows that our search is stopping earlier once the optimality certificate is established, exploring fewer items than OpenNMT's default search. Also note that the latter, even though exploring more items than ours, still can return suboptimal solutions; e.g., when $B_{i,1}$ is worst than $best_{\leq i}$ (they never stored $best_{\leq i}$). In practice, we noticed our search finishes about 3–5 steps earlier than OpenNMT at a beam of 10, and this advantage widens as beam size increases, although the overall speedup is not too noticeable, given the target language sentence length is much longer. Also, our model scores (i.e., log-probabilities) are indeed better (see Fig. 1), where the advantage is also more pronounced with larger beams (note that OpenNMT baseline is almost flat after $b = 10$, while our optimal beam search still steadily improves). Combining these two Theorems, it is interesting to note that our method is not just optimal but also faster.

4 Optimal Beam Search for Bounded Length Reward

However, optimal-score hypothesis, though satisfying in theory, is not ideal in practice, since neural models are notoriously bad in producing very short sentences, as opposed to older paradigms such as SMT (Shen et al., 2016). To alleviate this problem, two methods have been proposed: (a) length normalization, used in RNNsearch as an option, where the revised score of a hypothesis is divided by its length, thus favoring longer sentences; and (b) explicit length reward (He et al., 2016) borrowed from SMT, rewarding each generated word by a constant tuned on the dev set.

Unfortunately, each of these methods breaks the optimality proof of our beam search algorithm in Section 3, since a future hypothesis, being longer, might end up with a higher (revised) score. We therefore devise a novel mechanism called "bounded length reward", that is, we reward each word until the length of the hypothesis is longer than the "estimated optimal length". In machine translation and summarization, this optimal length l can be $ratio \cdot |\mathbf{x}|$ where $|\mathbf{x}|$ is the source sentence length, and $ratio$ is the average ratio of reference translation length over source sentence length on the dev set (in our Chinese-to-English NMT experiments, it is 1.27 as the English side is a bit longer). Note that we use the same $ratio$ estimated from dev on test, assuming that the optimal length ratio for test (which we do not know) should be similar to those of dev ones. We denote $\tilde{sc}(\mathbf{y})$ to be the revised score of hypothesis $\mathbf{y}$ with the bounded length reward, i.e.,

$$\tilde{sc}(\mathbf{y}) \stackrel{\Delta}{=} sc(\mathbf{y}) + r \cdot \min\{l, |\mathbf{y}|\}.$$

We also define $\tilde{best}_{\leq i}$ to be the revised version of $best_{\leq i}$ that optimizes the revised instead of the original score, i.e.,

$$\tilde{best}_{\leq i} \stackrel{\Delta}{=} \mathop{\mathbf{argmax}}_{\mathbf{y} \in \cup_{j \leq i} B_{j,comp(\mathbf{y})}} \tilde{sc}(\mathbf{y})$$

Now with bounded length reward, we can modify our beam search algorithm a little bit and still guarantee optimality. First we include in the revised cost a reward r for each generated word, as long as the length is less than l, the estimated optimal length. If at step i, the highest scoring item $B_{i,1}$'s revised score (i.e., including bounded length reward) plus the heuristic "future" extra length reward of a descendant, $r \cdot \max\{l - i, 0\}$, is worse than (or equal to) the similarly revised version of $best_{\leq i}$, i.e.,

$$\tilde{sc}(B_{i,1}) + r \cdot \max\{l - i, 0\} \leq \tilde{sc}(\tilde{best}_{\leq i}) \quad (3)$$

at which time we claim the revised optimality certificate is established, and terminate the beam search and return $\tilde{best}_{\leq i}$.

Actually with some trivial math we can simplify the stopping criteria to

$$sc(B_{i,1}) + r \cdot l \leq \tilde{sc}(\tilde{best}_{\leq i}). \quad (4)$$

This much simplified but still equivalent criteria can speed up decoding in practice, since this

	sents	tokens	vocab.	w/ BPE
Chinese	1M	28M	112k	18k
English	1M	23M	93k	10k

Table 1: Machine translation training set.

reward r	0	1	1.1	1.2	1.3	1.4	1.5
BLEU	32.2	34.6	34.6	**34.7**	34.6	34.6	34.6
len. ratio	0.88	.95	.96	.97	.98	.98	.99
best b	4	17	17	15	20	20	17

Table 2: Tuning length reward r (with beam size b=1..20) for optimal bounded-reward beam search.

means we actually do not need to compute the revised score for every hypothesis in the beam; we only need to add the bounded length reward when one is finished (i.e., when updating $\tilde{best}_{\leq i}$), and the simplified criteria only compares it with the original score of a hypothesis plus a constant reward $r \cdot l$.

Theorem 3 (modified optimality)**.** *Our modified beam search returns the highest-scoring completed hypothesis where the score of an item is its log-probability plus a bounded length reward.*

Proof. by admissibility of the heuristic. □

Theorem 4 (correctness of the simplified criteria)**.** *Eq. 4 is equivalent to Eq. 3.*

Proof. trivial. □

5 Experiments: Neural Translation

5.1 Data Preparation, Training, and Baselines

We conduct experiments on Chinese-to-English neural machine translation, using OpenNMT-py,[2] the PyTorch port of the Lua-based OpenNMT (Klein et al., 2017). We choose this library because PyTorch's combination of Python with Torch's dynamic computation graphs made it much easier to implement various search algorithms on it than on Theano-based implementations derived from RNNsearch (Bahdanau et al., 2014) (such as the widely used GroundHog[3] and Laulysta[4] codebases) as well as the original LuaTorch version of OpenNMT. We use 1M Chinese/English sentence pairs for training (see Table 1 for statistics); we also trained on 2M sentence pairs and only saw a minor improvement so below we report results from 1M training. To alleviate the vocabulary size issue we employ byte-pair encoding (BPE) (Sennrich et al., 2015) which reduces the source and target language vocabulary sizes to 18k and 10k, respectively; we found BPE to significantly improve BLEU scores (by at least +2 BLEU) and reduce training time. Following other papers on Chinese-English translation such as Shen et al. (2016), we use NIST 06 newswire portion (616 sentences) for development and NIST 08 newswire portion (691 sentences) for testing; we will report case-insensitive 4-reference BLEU-4 scores (using original segmentation).

Following OpenNMT-py's default settings, we train our NMT model for 20 epochs to minimize perplexity on the training set (excluding 15% sentences longer than 50 source tokens), with a batch size of 64, word embedding size of 500, and dropout rate of 0.3. The total number of parameters is 29M. Training takes about an hour per epoch on Geforce 980 Ti GPU, and the model at epoch 15 reaches the lowest perplexity on the dev set (9.10) which is chosen as the model for testing.

On dev set, the default decoder of OpenNMT-py reaches 29.2 BLEU with beam size 1 (greedy) and 33.2 BLEU with the default beam size of 5. To put this in perspective, the most commonly used SMT toolkit Moses (Koehn et al., 2007) reaches 30.1 BLEU (with beam size 70) using the same 1M sentence training set (trigram language model trained on the target side). With 2.56M training sentence pairs, Shen et al. (2016) reported 32.7 BLEU on the same dev set using Moses and 30.7 BLEU using the baseline RNNsearch (GroundHog) with beam size 10 (without BPE, without length normalization or length reward). So our OpenNMT-py baseline is extremely competitve.

5.2 Beam Search & Bounded Length Reward

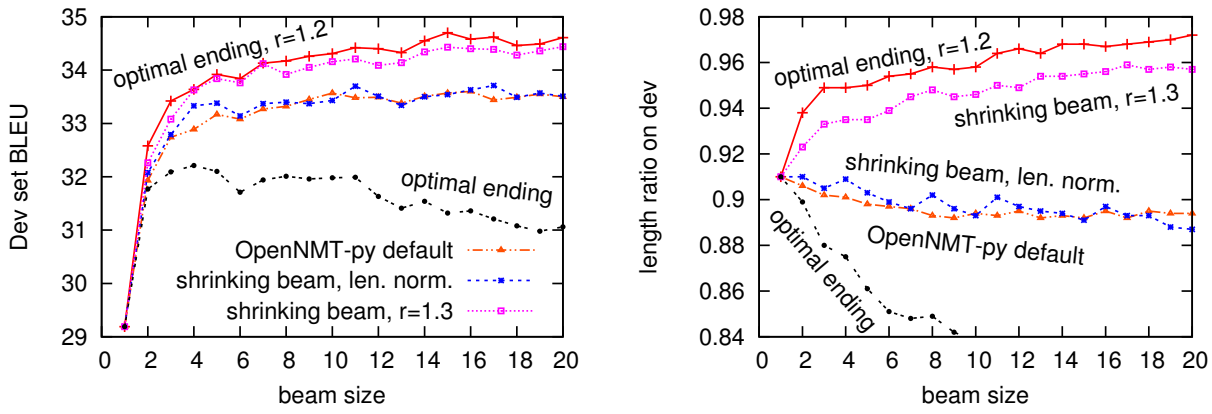
We compare the following beam search variants:

1. OpenNMT-py's default beam search, finishing only when the top hypothesis in a step is completed (see Section 2);

2. The "shrinking beam" method in RNNsearch with two variants to encourage longer translations:

   (a) length normalization; Google NMT (Wu et al., 2016) also adopted a similar

[2] https://github.com/opennmt/opennmt-py
[3] https://github.com/lisa-groundhog/
[4] https://github.com/laulysta/nmt/



(a) BLEU vs. beam size

(b) length ratio vs. beam size

Figure 2: BLEU score and length ratio against beam size (on dev) of various beam search algorithms for neural machine translation.

mechanism.

(b) unbounded length reward (tuned on dev set) in Baidu NMT (He et al., 2016).

3. Our optimal-ending beam search (Section 3);

4. Our modified optimal-ending beam search for bounded length reward (Section 4).

Notice that length reward has no effect on both methods 1 and 2(a) above. To tune the optimal length reward r we run our modified optimal-ending beam search algorithm with all combinations of $r = 0, 0.5, 1, 1.1, 1.2, 1.3, 1.4$ with beam sizes $b = 1 \ldots 20$ on the dev set, since different beam sizes might prefer different length rewards. We found $r = 1.2$ to be the best among all length rewards (see Table 2) which is used in Figure 2 and $b = 15$ is the best for $r = 1.2$.

We can observe from Figure 2 that (a) our optimal beam search with bounded length reward performs the best, and at b=15 it is +5 BLEU better than b=1; (b) pure optimal beam search degrades after b=4 due to extremely short translations; (c) both the shrinking beam method with length normalization and OpenNMT-py's default search alleviate the shortening problem, but still produce very short translations (length ratio ∼0.9). (d) the shrinking beam method with length reward works well, but still 0.3 BLEU below our best method. These are confirmed by the test set (Tab. 3).

decoder	b	dev	test
Moses	70	30.14	29.41
OpenNMT-py default	16	33.60	29.75
shrinking, len. norm.	17	33.71	30.11
shrinking, reward r=1.3	15	34.42	30.37
optimal beam search, r=1.2	15	**34.70**	**30.61**

Table 3: Final BLEU scores on the test set (nist 08) using best settings from the dev set (nist 06).

6 Conclusions

We have presented a beam search algorithm for neural sentence generation that always returns optimal-score completed hypotheses. To counter neural generation's natural tendancy for shorter hypotheses, we introduced a *bounded length reward* mechanism which allows a modified version of our beam search algorithm to remain optimal. Experiments on top of strong baselines have confirmed that our principled search algorithms (together with our bounded length reward mechanism) outperform existing beam search methods in terms of BLEU scores. We will release our implementations (which will hopefully be merged into OpenNMT-py) when this paper is published. [5]

Acknowledgments

We thank the anonymous reviewers from both EMNLP and WMT for helpful comments. This work is supported in part by NSF IIS-1656051, DARPA N66001-17-2-4030 (XAI), a Google Faculty Research Award, and HP.

[5] While implementing our search algorithms we also found and fixed an obscure but serious bug in OpenNMT-py's baseline beam search code (not related to discussions in this paper), which boosts BLEU scores by about +0.7 in all cases. We will release this fix as well.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Wei He, Zhongjun He, Hua Wu, and Haifeng Wang. 2016. Improved neural machine translation with smt features. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, pages 151–157.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*. volume 3, page 413.

G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. *ArXiv e-prints* .

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*. Association for Computational Linguistics, pages 177–180.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. *ICLR* .

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* .

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909* .

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of ACL*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 3156–3164.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*. volume 14, pages 77–81.