Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation

Yilin Yang[1] **Liang Huang**[1,2] **Mingbo Ma**[1,2]
[1]Oregon State University
Corvallis, OR, USA
[2]Baidu Research
Sunnyvale, CA, USA
{yilinyang721, liang.huang.sh, cosmmb}@gmail.com

Abstract

Beam search is widely used in neural machine translation, and usually improves translation quality compared to greedy search. It has been widely observed that, however, beam sizes larger than 5 hurt translation quality. We explain why this happens, and propose several methods to address this problem. Furthermore, we discuss the optimal stopping criteria for these methods. Results show that our hyperparameter-free methods outperform the widely-used hyperparameter-free heuristic of length normalization by +2.0 BLEU, and achieve the best results among all methods on Chinese-to-English translation.

1 Introduction

In recent years, neural machine translation (NMT) has surpassed traditional phrase-based or syntax-based machine translation, becoming the new state of the art in MT (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2014). While NMT training is typically done in a "local" fashion which does not employ any search (bar notable exceptions such as Ranzato et al. (2016), Shen et al. (2016), and Wiseman and Rush (2016)), the decoding phase of all NMT systems universally adopts beam search, a widely used heuristic, to improve translation quality.

Unlike phrase-based MT systems which enjoy the benefits of very large beam sizes (in the order of 100–500) (Koehn et al., 2007) , most NMT systems choose tiny beam sizes up to 5; for example, Google's GNMT (Wu et al., 2016) and Facebook's ConvS2S (Gehring et al., 2017) use beam sizes 3 and 5, respectively. Intuitively, the larger the beam size is, the more candidates it explores, and the better the translation quality should be. While this definitely holds for phrase-based MT systems, surprisingly, it is *not* the case for NMT: many researchers observe that translation quality degrades with beam sizes beyond 5 or 10 (Tu et al., 2017; Koehn and Knowles, 2017). We call this phenomenon the *"beam search curse"*, which is listed as one of the six biggest challenges for NMT (Koehn and Knowles, 2017).

However, there has not been enough attention on this problem. Huang et al. (2017) hint that length ratio is the problem, but do not explain why larger beam sizes cause shorter lengths and worse BLEU. Ott et al. (2018) attribute it to two kinds of "uncertainties" in the training data, namely the *copying* of source sentence and the *non-literal* translations. However, the first problem is only found in European language datasets and the second problem occurs in all datasets but does not seem to bother pre-neural MT systems. Therefore, their explanations are not satisfactory.

On the other hand, previous work adopts several heuristics to address this problem, but with various limitations. For example, RNNSearch (Bahdanau et al., 2014) and ConvS2S use *length normalization*, which (we will show in Sec. 6) seems to somewhat alleviate the problem, but far from being perfect. Meanwhile, He et al. (2016) and Huang et al. (2017) use word-reward, but their *reward* is a hyper-parameter to be tuned on dev set.

Our contributions are as follows:

- We explain why the beam search curse exists, supported by empirical evidence (Sec. 3).
- We review existing rescoring methods, and then propose ours to break the beam search curse (Sec. 4). We show that our hyperparameter-free methods outperfrom the previous hyperparameter-free method (length normalization) by +2.0 BLEU (Sec. 6).
- We also discuss the stopping criteria for our rescoring methods (Sec. 5). Experiments

show that with optimal stopping alone, the translation quality of the length normalization method improves by +0.9 BLEU.

2 Preliminaries: NMT and Beam Search

We briefly review the encoder-decoder architecture with attention mechanism (Bahdanau et al., 2014). An RNN encoder takes an input sequence $\mathbf{x} = (x_1, ..., x_m)$, and produces a sequence of hidden states. For each time step, the RNN decoder will predict the probability of next output word given the source sequence and the previously generated prefix. Therefore, when doing greedy search, at time step i, the decoder will choose the word with highest probability as y_i. The decoder will continue generating until it emits </eos>. In the end, the generated hypothesis is $\mathbf{y} = (y_1, ..., y_n)$ with $y_n =$ </eos>, with model score

$$S(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{y}|} \log p(y_i \mid \mathbf{x}, y_{1..\{i-1\}}) \quad (1)$$

As greedy search only explores a single path, we always use beam search to improve search quality. Let b denote the beam size, then at step i the beam B_i is an *ordered* list of size b:

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

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

In the most naive case, after reaching the maximum length (a hard limit), we get N possible candidate sequences $\{\mathbf{y}_1, ..., \mathbf{y}_N\}$. The *default* strategy chooses the one with highest model score. We will discuss more sophistcated ways of stopping and choosing candidates in later sections.

3 Beam Search Curse

The most popular translation quality metric, BLEU (Papineni et al., 2002), is defined as:

$$\text{BLEU} = bp \cdot \exp\left(\frac{1}{4}\sum_{n=1}^{4} \log p_n\right) \quad (2)$$

$$\text{where} \quad bp = \min\{e^{1-1/lr}, 1\} \quad (3)$$

$$\text{where} \quad lr = |\mathbf{y}|/|\mathbf{y}^*| \quad (4)$$

Here p_n are the n-gram precisions, and $|\mathbf{y}|$ and $|\mathbf{y}^*|$ denote the hypothesis and reference lengths, while bp is the *brevity penalty* (penalizing short translations) and lr is the *length ratio* (Shi et al., 2016; Koehn and Knowles, 2017), respectively.

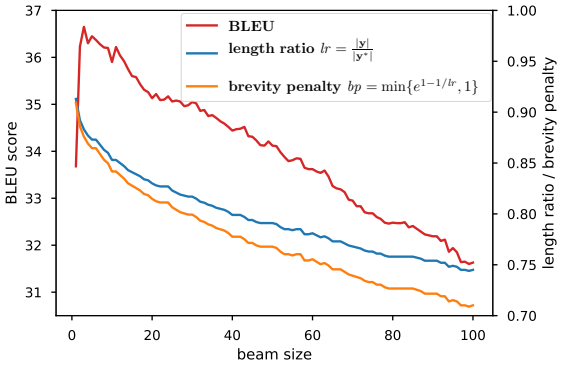

Figure 1: As beam size increases beyond 3, BLEU score on the dev set gradually drops. All terms are calculated by *multi-bleu.pl*.

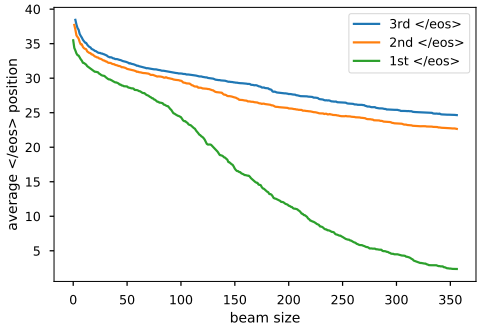

Figure 2: Searching algorithm with larger beams generates </eos> earlier. We use the average first, second and third </eos> positions on the dev set as an example.

With beam size increasing, $|\mathbf{y}|$ decrases, which causes the length ratio to drop, as shown in Fig. 1. Then the brevity penalty term, as a function of the length ratio, decreases even more severely. Since bp is a key factor in BLEU, this explains why the beam search curse happens.[1]

The reason why $|\mathbf{y}|$ decreases as beam size increases is actually twofold:

1. As beam size increases, the more candidates it could explore. Therefore, it becomes easier for the search algorithm to find the </eos> symbol. Fig. 2 shows that the </eos> indices decrease steadily with larger beams.[2]
2. Then, as shown in Fig. 3, shorter candidates have clear advantages *w.r.t.* model score.

[1] The length ratio is *not* just about BLEU: if the hypothesis length is only 75% of reference length, something that should have been translated must be missing; i.e., bad adequacy. We believe the same problem still exists if we use TER instead.

[2] Pre-neural SMT models, being probabilistic, also favor short translations (and derivations), which is addressed by word (and phrase) reward. The crucial difference between SMT and NMT is that the former stops when covering the whole input, while the latter stops on emitting </eos>.

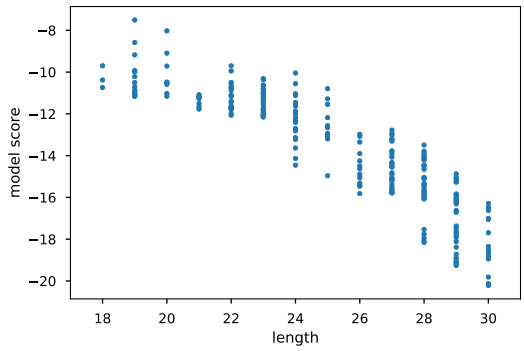

Figure 3: Candidate lengths vs. model score. This scatter plot is generated from 242 finished candidates when translated from one source sequence with beam size 80.

Hence, as beam size increases, the search algorithm will generate shorter candidates, and then prefer even shorter ones among them.

4 Rescoring Methods

We first review existing methods to counter the length problem and then propose new ones to address their limitations. In particular, we propose to predict the target length from the source sentence, in order to choose a hypothesis with proper length.

4.1 Previous Rescoring Methods

RNNSearch (Bahdanau et al., 2014) first introduces the *length normalization* method, whose score is simply the average model score:

$$\hat{S}_{\text{length\_norm}}(\mathbf{x},\mathbf{y}) = \frac{S(\mathbf{x},\mathbf{y})}{|\mathbf{y}|} \tag{5}$$

This is the most widely used rescoring method since it is hyperparameter-free.

GNMT (Wu et al., 2016) incorporates length and coverage penalty into the length normalization method, while also adding two hyperparameters to adjust their influences. (please check out their paper for exact formulas).

Baidu NMT (He et al., 2016) borrows the *Word Reward* method from pre-neural MT, which gives a reward r to every word generated, where r is a hyperparameter tuned on the dev set:

$$\hat{S}_{\text{WR}}(\mathbf{x},\mathbf{y}) = S(\mathbf{x},\mathbf{y}) + r \cdot |\mathbf{y}| \tag{6}$$

Based on the above, Huang et al. (2017) propose a variant called *Bounded Word-Reward* which only rewards up to an "optimal" length. This length is calculated using a fixed "generation ratio" gr, which is the ratio between target and source sequence length, namely the average number of target words generated for each source word. It gives reward r to each word up to a bounded length $L(\mathbf{x},\mathbf{y}) = \min\{|\mathbf{y}|, gr \cdot |\mathbf{x}|\}$:

$$\hat{S}_{\text{BWR}}(\mathbf{x},\mathbf{y}) = S(\mathbf{x},\mathbf{y}) + r \cdot L(\mathbf{x},\mathbf{y}) \tag{7}$$

4.2 Rescoring with Length Prediction

To remove the fixed generation ratio gr from Bounded Word-Reward, we use a 2-layer MLP, which takes the mean of source hidden states as input, to predict the generation ratio $gr^*(\mathbf{x})$. Then we replace the fixed ratio gr with it, and get our predicted length $L_{pred}(\mathbf{x}) = gr^*(\mathbf{x}) \cdot |\mathbf{x}|$.

4.2.1 Bounded Word-Reward

With predicted length, the new predicted bound and final score would be:

$$L^*(\mathbf{x},\mathbf{y}) = \min\{|\mathbf{y}|, L_{pred}(\mathbf{x})\} \tag{8}$$

$$\hat{S}_{\text{BWR}^*}(\mathbf{x},\mathbf{y}) = S(\mathbf{x},\mathbf{y}) + r \cdot L^*(\mathbf{x},\mathbf{y}) \tag{9}$$

While the predicted length is more accurate, there is still a hyperparameter r (word reward), so we design two methods below to remove it.

4.2.2 Bounded Adaptive-Reward

We propose *Bounded Adaptive-Reward* to automatically calculate proper reward based on the current beam. With beam size b, the reward for time step t is the average negative log-probability of the words in the current beam.

$$r_t = -\frac{1}{b}\sum_{i=1}^{b} \log p(\text{word}_i) \tag{10}$$

Its score is very similar to (7):

$$\hat{S}_{\text{AdaR}}(\mathbf{x},\mathbf{y}) = S(\mathbf{x},\mathbf{y}) + \sum_{t=1}^{L^*} r_t \tag{11}$$

4.2.3 BP-Norm

Inspired by the BLEU score definition, we propose *BP-Norm* method as follows:

$$\hat{S}_{bp}(\mathbf{x},\mathbf{y}) = \log bp + \frac{S(\mathbf{x},\mathbf{y})}{|\mathbf{y}|} \tag{12}$$

bp is the same brevity penalty term as in (3). Here, we regard our predicted length as the reference length. The beauty of this method appears when

we drop the logarithmic symbol in (12):

$$\exp(\hat{S}_{bp}(\mathbf{x},\mathbf{y})) = bp \cdot \left(\prod_{i=1}^{|\mathbf{y}|} p(y_i|...)\right)^{\frac{1}{|\mathbf{y}|}}$$
$$= bp \cdot \exp\left(\frac{1}{|\mathbf{y}|}\sum_{i=1}^{|\mathbf{y}|}\log p(y_i|...)\right)$$

which is in the same form of BLEU score (3).

5 Stopping Criteria

Besides rescoring methods, the stopping criteria (when to stop beam search) is also important, for both efficiency and accuracy.

5.1 Conventional Stopping Criteria

By default, OpenNMT-py (Klein et al., 2017) stops when the topmost beam candidate stops, because there will not be any future candidates with higher model scores. However, this is not the case for other rescoring methods; e.g., the score of length normalization (5) could still increase.

Another popular stopping criteria, used by RNNSearch (Bahdanau et al., 2014), stops the beam search when exactly b finished candidates have been found. Neither method is optimal.

5.2 Optimal Stopping Criteria

For Bounded Word-Reward, Huang et al. (2017) introduces a provably-optimal stopping criterion that could stop both early and optimally. We also introduce an optimal stopping criterion for BP-Norm. Each time we generate a finished candidate, we update our best score $\hat{S}^\star$. Then, for the topmost beam candidate of time step t, we have:

$$\hat{S}_{bp} = \frac{S_{t,0}}{t} + \min\{1 - \frac{L_{pred}}{t}, 0\} \leq \frac{S_{t,0}}{R} \quad (13)$$

where R is the maximum generation length. Since $S_{t,0}$ will drop after time step t, if $\frac{S_{t,0}}{R} \leq \hat{S}^\star$, we reach optimality. This stopping criterion could also be applied to length normalization (5).

Meawhile, for Bounded Adaptive-Reward, we can have a similar optimal stopping criterion: If the score of topmost beam candidate at time step $t > L_{pred}$ is lower than $\hat{S}^\star$, we reach optimality.

Proof. The first part of $\hat{S}_{\text{AdaR}}$ in (11) will decrease after time step t, while the second part stays the same when $t > L_{pred}$. So the score in the future will monotonically decrease. □

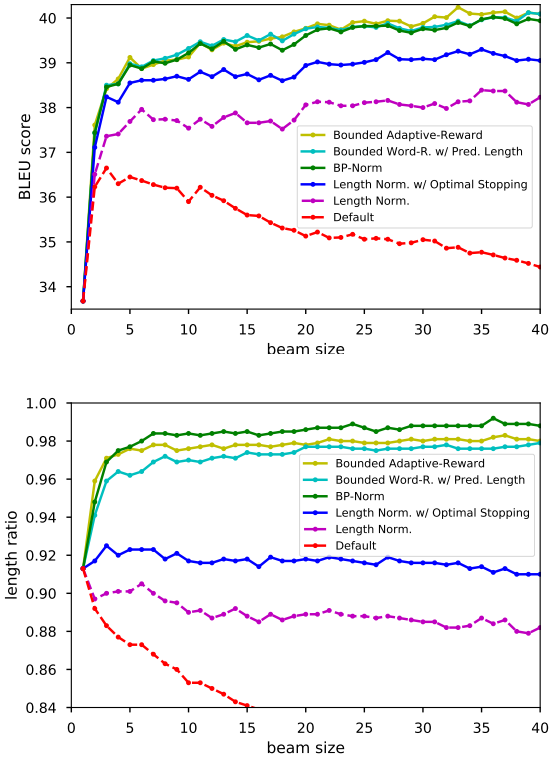

Figure 4: The BLEU scores and length ratios ($lr = |\mathbf{y}|/|\mathbf{y}^*|$) of various rescoring methods.

6 Experiments

Our experiments are on Chinese-to-English translation task, based on the OpenNMT-py codebase.[3] We train our model on 2M sentences, and apply BPE (Sennrich et al., 2015) on both sides, which reduces Chinese and English vocabulary sizes down to 18k and 10k respectively. We then exclude pairs with more than 50 source or target tokens. We validate on NIST 06 and test on NIST 08 (newswire portions only for both). We report case-insensitive, 4 reference BLEU scores.

We use 2-layers bidirectional LSTMs for the encoder. We train the model for 15 epochs, and choose the one with lowest perplexity on the dev set. Batch size is 64; both word embedding and hidden state sizes 500; and dropout 0.3. The total parameter size is 28.5M.

6.1 Parameter Tuning and Results

We compare all rescoring methods mentioned above. For the length normalization method, we also show its results with *optimal stopping*.

For Bounded Word-Reward method with and without our predicted length, we choose the best r on the dev set seperately. The length normal-

[3] https://github.com/OpenNMT/OpenNMT-py

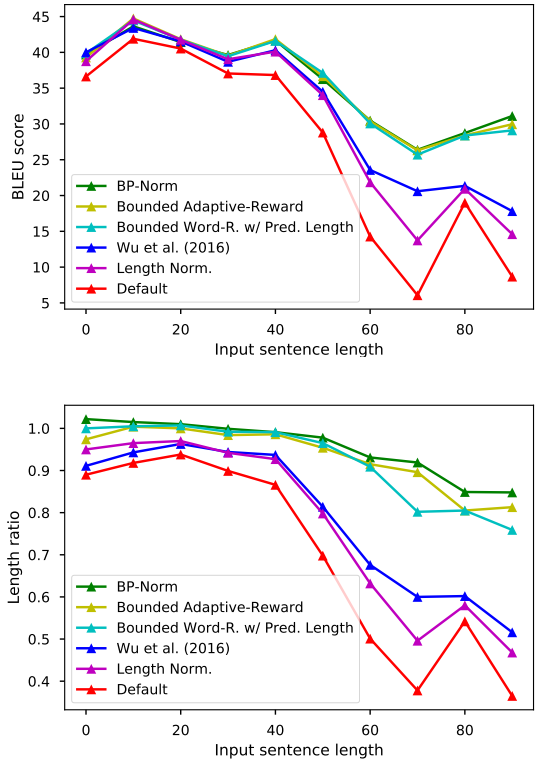

Figure 5: BLEU scores and length ratios on the dev set over various input sentence lengths.

Small beam ($b = 14, 15, 16$)	dev		test	
	BLEU	ratio	BLEU	ratio
Moses (b=70)	30.14	-	29.41	-
Default (b=5)	36.45	0.87	32.88	0.87
Length Norm.	37.73	0.89	34.07	0.89
+ optimal stopping*	38.69	0.92	35.00	0.92
Wu et al. (2016) α=β=0.3	38.12	0.89	34.26	0.89
Bounded word-r. r=1.3	39.22	0.98	35.76	0.98
with predicted length				
Bounded word-r. r=1.4*	**39.53**	0.97	35.81	0.97
Bounded adaptive-reward*	39.44	0.98	35.75	0.98
BP-Norm*	39.35	0.98	**35.84**	0.99

Large beam ($b = 39, 40, 41$)	dev		test	
	BLEU	ratio	BLEU	ratio
Moses (b=70)	30.14	-	29.41	-
Default (b=5)	36.45	0.87	32.88	0.87
Length Norm.	38.15	0.88	34.26	0.88
+ optimal stopping*	39.07	0.91	35.14	0.91
Wu et al. (2016) α=β=0.3	38.40	0.89	34.41	0.88
Bounded word-r. r=1.3	39.60	0.98	35.98	0.98
with predicted length				
Bounded word-r. r=1.4*	40.11	0.98	36.13	0.97
Bounded adaptive-reward*	**40.14**	0.98	**36.23**	0.98
BP-Norm*	39.97	0.99	**36.22**	0.99

Table 1: Average BLEU scores and length ratios over small and large beams. $\star$ indicates our methods.

ization used by Wu et al. (2016) has two hyper-parameters, namely α for length penalty and β for coverage penalty. We jointly tune them on the dev set, and choose the best config. (α=0.3, β=0.3).

Figure 4 show our results on the dev set. We see that our proposed methods get the best performance on the dev set, and continue growing as beam size increases. We also observe that optimal stopping boosts the performance of length normalization method by around +0.9 BLEU. In our experiments, we regard our predicted length as the *maximum generation length* in (13). We further observe from Fig. 5 that our methods keep the length ratio close to 1, and greatly improve the quality on longer input sentences, which are notoriously hard for NMT (Shen et al., 2016).

Table 1 collects our results on both dev and test sets. Without loss of generality, we show results with both small and large beam sizes, which average over b=14,15,16 and b=39,40,41, respectively.

6.2 Discussion

From Table 1, we could observe that with our length prediction model, Bounded word-reward method gains consistent improvement. On the other hand, results from length normalization method show that optimal stopping technique gains significant improvement by around +0.9 BLEU. While with both, our proposed methods beat all previous methods, and gain improvement over hyperparameter-free baseline (i.e. length normalization) by +2.0 BLEU.

Among our proposed methods, Bounded word-reward has the reward r as an hyper-parameter, while the other two methods get rid of that. Among them, we recommend the BP-Norm method, because it is the simplest method, and yet works equally well with others.

7 Conclusions

We first explain why the beam search curse exists and then formalize all previous rescoring methods. Beyond that, we also propose several new methods to address this problem. Results from the Chinese-English task show that our hyperparameter-free methods beat the hyperparameter-free baseline (length normalization) by +2.0 BLEU.

Acknowledgements

Kenton Lee suggested the length prediction idea. This work was partially supported by DARPA N66001-17-2-4030, and NSF IIS-1817231 and IIS-1656051. We thank the anonymous reviewers for suggestions and Juneki Hong for proofreading.

References

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

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proc. of ICML*.

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*, pages 151–157. AAAI Press.

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

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*, pages 177–180. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *CoRR*, abs/1706.03872.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. *arXiv preprint arXiv:1803.00047*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, Philadephia, USA.

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

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*.

Xing Shi, Kevin Knight, and Deniz Yuret. 2016. Why neural translations are the right length. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2278–2282.

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.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction. In *AAAI*, pages 3097–3103.

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

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*.