Speculative Beam Search for Simultaneous Translation

Renjie Zheng [2,*] **Mingbo Ma** [1,*] **Baigong Zheng** [1] **Liang Huang** [1,2]
[1]Baidu Research, Sunnyvale, CA, USA
[2]Oregon State University, Corvallis, OR, USA
zrenj11@gmail.com
{mingboma, baigongzheng, lianghuang}@baidu.com

Abstract

Beam search is universally used in full-sentence translation but its application to simultaneous translation remains non-trivial, where output words are committed on the fly. In particular, the recently proposed wait-k policy (Ma et al., 2019a) is a simple and effective method that (after an initial wait) commits one output word on receiving each input word, making beam search seemingly impossible. To address this challenge, we propose a speculative beam search algorithm that hallucinates several steps into the future in order to reach a more accurate decision, implicitly benefiting from a target language model. This makes beam search applicable for the first time to the generation of a single word in each step. Experiments over diverse language pairs show large improvements over previous work.

1 Introduction

Beam search has been widely used in neural text generation such as machine translation (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). It often leads to substantial improvement over greedy search and becomes an essential component in almost all text generation systems.

However, beam search is easy for the above tasks because they are all *full-sequence* problems, where the whole input sequence is available at the beginning and the output sequence only needs to be revealed in full at the end. By contrast, in language and speech processing, there are many *incremental processing* tasks with *simultaneity requirements*, where the output needs to be revealed to the user incrementally without revision (word by word, or in chunks) and the input is also being received incrementally. Two most salient examples are streaming speech recognition (Chiu et al., 2018), widely used in speech input and dialog systems (such as Siri), and simultaneous translation (Bangalore et al., 2012; Oda et al., 2015; Grissom II et al., 2014; Jaitly et al., 2016), widely used in international conferences and negotiations. In these tasks, the use of full-sentence beam search becomes seemingly impossible as output words need to be committed on the fly.

How to adapt beam search for such incremental tasks in order to improve their generation quality? We propose a general technique of *speculative beam search* (SBS), and apply it to simultaneous translation. At a very high level, to generate a single word, instead of simply choosing the highest-scoring one (as in greedy search), we further speculate w steps into the future, and use the ranking at step $w+1$ to reach a more informed decision for step 1 (the current step); this method implicitly benefits from a target language model, alleviating the label bias problem in neural generation (Murray and Chiang, 2018; Ma et al., 2019b).

We apply this algorithm to two representative approaches to simultaneous translation: the fixed policy method (Ma et al., 2019a) and the adaptive policy method (Gu et al., 2017). In both cases, we show that SBS improves translation quality while maintaining latency (i.e., simultaneity).

2 Preliminaries

We first review standard full-sentence NMT and beam search to set up the notations, and then review different approaches to simultaneous MT.

2.1 Full Sentence NMT and Beam Search

The encoder processes the input sequence $\boldsymbol{x} = (x_1, ..., x_n)$, where $x_i \in \mathbb{R}^d$ represents an input token as a d dimensional vector, and produces a new list of hidden states $\boldsymbol{h} = f(\boldsymbol{x}) = (h_1, ..., h_n)$

[*]These authors contributed equally.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 1395–1402, Hong Kong, China, November 3–7, 2019.

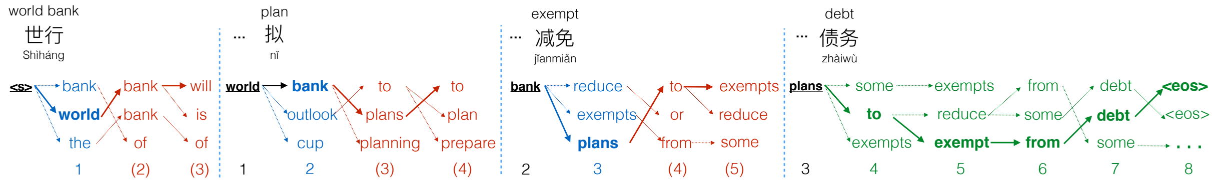


Figure 1: Wait-1 policy example to illustrate the procedure of SBS. The top Chinese words are the source side inputs which are incrementally revealed to the encoder. Gloss is annotated above Chinese word and Pinyin is underneath. There are two extra steps (speculative window) are taken (red part) beyond greedy. When source reaches the last word "债务" (debt), the decoder gets into tail and performs conventional beam search (in green).

to represent $\boldsymbol{x}$. The encoding function f can be RNN, CNN or Transformer.

On the other hand, the (greedy) decoder selects the highest-scoring word y_t given source representation $\boldsymbol{h}$ and previously generated target tokens, $\boldsymbol{y}_{<t} = (y_1, ..., y_{t-1})$. The greedy search continues until it emits <eos>, and the final hypothesis $\boldsymbol{y} = (y_1, ..., y_t)$ with $y_t =$ <eos>

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{t=1}^{|\boldsymbol{y}|} p(y_t \mid \boldsymbol{x},\, \boldsymbol{y}_{<t}) \quad (1)$$

As greedy search only explores one single path among exponential many alternatives, beam search is used to improve the search. At each step t, it maintains a beam B_t of size b, which is an ordered list of $\langle$hypothesis, probability$\rangle$ pairs; for example $B_0 = [\langle \texttt{<s>}, 1\rangle]$. We then define one-step transition from the previous beam to the next as

$$\text{next}_1^b(B) = \text{top}^b\{\langle \boldsymbol{y} \circ v,\ s \cdot p(v|\boldsymbol{x}, \boldsymbol{y})\rangle \mid \langle \boldsymbol{y}, s\rangle \in B\}$$

where $\text{top}^b(\cdot)$ returns the top-scoring b pairs, and $\circ$ is the string concatenation operator. Now $B_t = \text{next}_1^b(B_{t-1})$. As a shorthand, we also define the multi-step beam search function recursively:

$$\text{next}_i^b(B) = \text{next}_1^b(\text{next}_{i-1}^b(B)) \quad (2)$$

Full-sentence beam search (over a maximum of T steps) yields the best hypothesis $\boldsymbol{y}^*$ with score s^* (see Huang et al. (2017) for stopping criteria):

$$\langle \boldsymbol{y}^*, s^*\rangle = \text{top}^1(\text{next}_T^b([\langle \texttt{<s>}, 1\rangle])) \quad (3)$$

2.2 Simultaneous MT: Policies and Models

There are two main categories of policies in neural simultaneous translation decoding (Tab. 1):

1. The first method is to use a fixed-latency policy, such as the wait-k policy (Ma et al., 2019a). Such a method would, after an initial wait of k source words, commit one target word on receiving each new source word. When the source sentence ends, the decoder can do a *tail beam search* on the remaining target words, but beam search is seemingly impossible before the source sentence ends.

policy \ model	*sequence-to-sequence* (full-sentence model)	*prefix-to-prefix* (simultaneous model)
fixed-latency	test-time wait-k (Dalvi et al., 2018; Ma et al., 2019a)	wait-k (Ma et al., 2019a)
adaptive	RL (Gu et al., 2017) Supervised Learning (Zheng et al., 2019a)	MILk (Arivazhagan et al., 2019) Imitation Learning (Zheng et al., 2019b)

Table 1: Recent advances in simultaneous translation.

2. The second method learns an adaptive policy which uses either supervised (Zheng et al., 2019a) or reinforcement learning (Grissom II et al., 2014; Gu et al., 2017) to decide whether to READ (the next source word) or WRITE (the next target word) . Here the decoder can commit a chunk of *multiple words* for a series of consecutive WRITEs.

In terms of modeling (which is orthogonal to decoding policies), we can also divide most simultaneous translatoin efforts into two camps:

1. Use the standard full-sentence translation model trained by classical seq-to-seq (Dalvi et al., 2018; Gu et al., 2017; Zheng et al., 2019a). For example, the "test-time wait-k" scheme (Ma et al., 2019a) uses the full-sentence translation model and performs wait-k decoding at test time. However, the obvious training-testing mismatch in this scheme usually leads to inferior quality.

2. Use a genuinely simultaneous model trained by the recently proposed prefix-to-prefix framework (Ma et al., 2019a; Arivazhagan et al., 2019; Zheng et al., 2019b). There is no training-testing mismatch in this new scheme, with the cost of slower training.

3 Speculative Beam Search

We first present our speculative beam search on the fixed-latency wait-k policy (generating a single word per step), and then adapt it to the adaptive policies (generating multiple words per step).

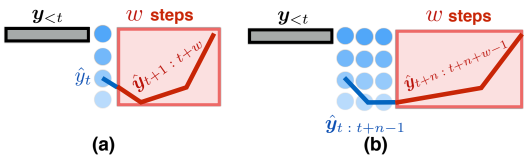


Figure 2: Illustration of SBS: (a) wait-k policy (Eqs. 4–5); (b) adaptive policy (Eqs. 6–7). Speculations in red.

3.1 Single-Step SBS

The wait-k policy conducts translation concurrently with the source input, committing output words one by one while the source sentence is still growing. In this case, conventional beam search is clearly inapplicable.

We propose to perform speculative beam search at each step by hallucinating w more steps into the future, and use the ranking after these $w+1$ steps to make a more informed decision for the current step. More formally, at step t, we generate y_t based on already committed prefix $\boldsymbol{y}_{<t}$:

$$\langle \hat{\boldsymbol{y}}, s_t \rangle = \text{top}^1\left(\text{next}^b_{1+w}([\langle \boldsymbol{y}_{<t}, 1\rangle])\right) \quad (4)$$

$$\boldsymbol{y}_{\leq t} = \boldsymbol{y}_{<t} \circ \hat{y}_t \quad (5)$$

where $\hat{\boldsymbol{y}} = \boldsymbol{y}_{<t} \circ \hat{y}_t \circ \hat{\boldsymbol{y}}_{t+1:t+w}$ has three parts, with the last one being a speculation of w steps (see Fig. 2). We use $\text{next}^b_{1+w}(\cdot)$ to speculate w steps. The candidate $\hat{y}_t$ is selected based on the accumulative model score w steps later. Then we commit $\hat{y}_t$ and move on to step $t+1$.

In the running example in Fig. 1, we have $w=2$ and $b=3$. In the greedy mode, after the wait-1 policy receives the first source word, "世行" (world bank), the basic wait-1 model commits "bank" which has the highest score. In SBS, we perform a beam search for $1+w=3$ steps with the two speculative steps marked in red. After 3 steps, the path "world bank will" becomes the top candidate, thus we choose to commit "world" instead of "bank" and restart a new speculative beam search with "world" when we receive a new source word, "拟"(plan to); the speculative part from the previous step (in red) is removed.

3.2 Chunk-based SBS

The RL-based adaptive policy system (Gu et al., 2017) can commit a chunk of multiple words whenever there is a series of consecutive WRITEs, and conventional beam search can be applied on each chunk to improve the search quality within that chunk, which is already used in that work.

However, on top of the obvious per-chunk beam search, we can still apply SBS to further speculate w steps after the chunk. For a chunk of length n starting at position t, we adapt SBS as:

$$\langle \hat{\boldsymbol{y}}, s_t \rangle = \text{top}^1\left(\text{next}^b_{n+w}([\langle \boldsymbol{y}_{<t}, 1\rangle])\right) \quad (6)$$

$$\boldsymbol{y}_{\leq t+n-1} = \boldsymbol{y}_{<t} \circ \hat{\boldsymbol{y}}_{t:t+n-1} \quad (7)$$

Here $\text{next}^b_{n+w}(\cdot)$ does a beam search of $n+w$ steps, with the last w steps speculated. Similarly,

$$\hat{\boldsymbol{y}} = \boldsymbol{y}_{<t} \circ \hat{\boldsymbol{y}}_{t:t+n-1} \circ \hat{\boldsymbol{y}}_{t+n:t+n+w-1}$$

has three parts, with the last being a speculation of w steps, and the middle one being the chunk of n steps returned and committed (see Fig. 2).

4 Experiments

4.1 Datasets and Latency Metrics

We evaluate our work on Chinese↔English simultaneous translation tasks. For the training data, we use the NIST corpus for Chinese↔English (2M sentence pairs). We first apply BPE (Sennrich et al., 2015) on all texts in order to reduce the vocabulary sizes. For Chinese↔English evaluation, we use NIST 2006 and NIST 2008 as our dev and test sets with 4 English references. For English→Chinese, we use the second among the four English references as the source text.

We re-implement wait-k model (Ma et al., 2019a), test-time wait-k model (Dalvi et al., 2018) and adaptive policy (Gu et al., 2017) based on PyTorch-based OpenNMT (Klein et al., 2017). To reach state-of-the-art performance, we use Transformer based wait-k model and also use Transformer based pre-trained full sentence model for learning adaptive policy. The architecture of Transformer is the same as the base model from the original paper (Vaswani et al., 2017). We use Average Lagging (AL) (Ma et al., 2019a) as the latency metrics. AL measures the number of words delay for translating a given source sentence.

b \ w	0	1	2	3	4	5
1	34.57	-	-	-	-	-
3	-	+1.3	+1.8	+1.2	+2.0	+1.7
5	-	+1.6	+1.9	+1.3	+1.5	+1.3
7	-	+1.5	+2.0	+1.0	+1.6	+1.4
10	-	+1.4	+2.2	+1.4	+1.5	+1.7

Table 2: Zh→En wait-1 model BLEU improvement of SBS against greedy search ($b=1$, $w=0$) on dev-set. When $w \geq 5$ the performance of SBS becomes stable.

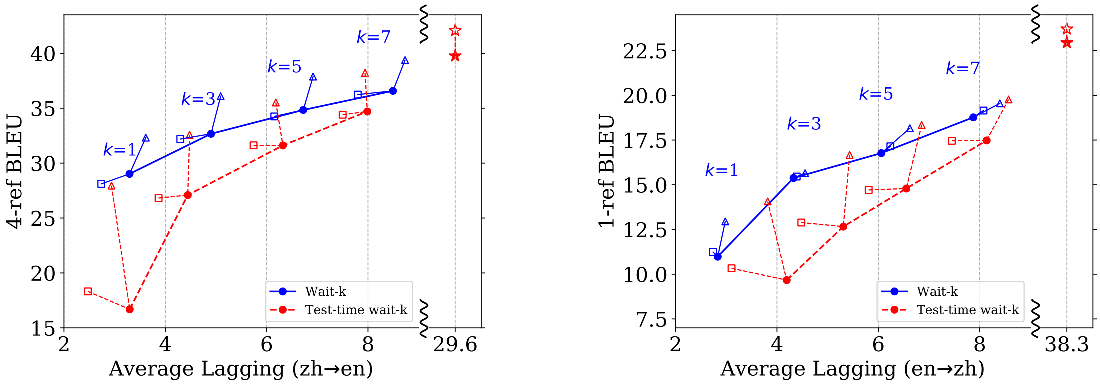


Figure 3: BLEU against AL using wait-k model. □ □: conventional beam search only in target tail (when source finishes). △ △: speculative beam search. ★☆:full-sentence baseline (greedy and beam-search).

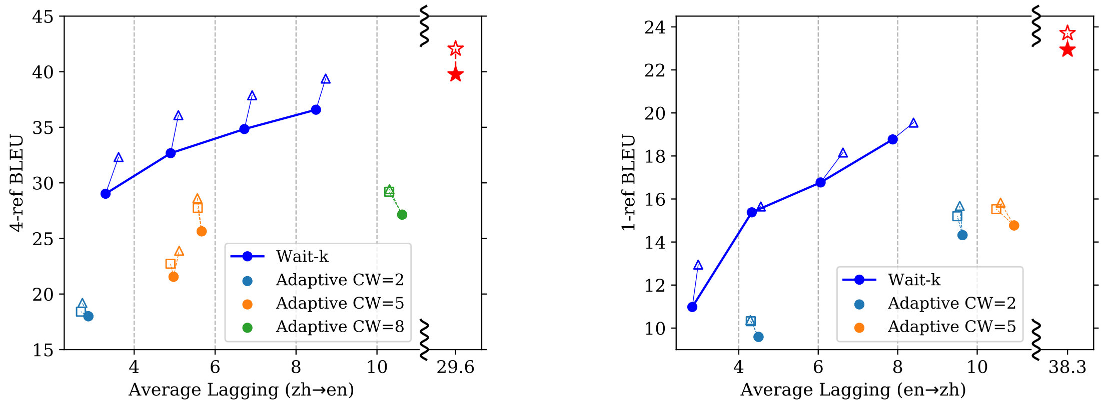


Figure 4: BLEU against AL using adaptive policy (compared with wait-k models) with different beam search methods. □ □ □: conventional beam search in chunk of consecutive write (Gu et al., 2017). △ △ △: speculative beam search. ★☆:full-sentence baseline (greedy and beam-search).

		shìháng 世行	*nǐ* 拟	*jiǎnmiǎn* 减免	*zùi* 最	*qióng* 穷	*gúojiā* 国家	*zhàiwù* 债务	
Gloss		world bank	plan to	remit & reduce	most	poor	country	debt	
k=1[†]	Greedy		world bank		to	reduce	poverty -		stricken countries
	SBS		world bank		to	exemp- t		po-	or- est countries from debt
k=1[‡]	Greedy		world bank		to	reduce	or	exemp- t debt of po- or- est countries	
	SBS		world bank		inten- ds		to	reduce	or exemp- t debt of po- or- est countries
k=∞[*]									world bank plans to remit and reduce debts of po- or- est countries

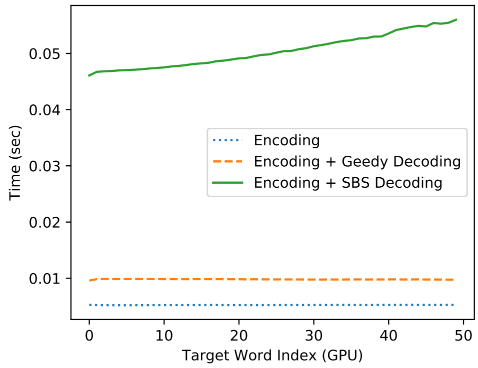
Figure 5: Chinese-to-English example on dev set. [†]: test-time wait-k; [‡]: wait-k. [*]: full-sentence beam search.

4.2 Performance on Wait-k Policy

We perform experiments on validation set using speculative beam search (SBS) with beam sizes $b \in \{3, 5, 7, 10\}$ and speculative window sizes $w \in \{1, 2, 3, 4, 5\}$. Table 2 shows the BLEU score of different b and w over wait-1 model. Compared with greedy decoding, SBS improves at least 1.0 BLEU score in all cases and achieves best performance by $b = 10, w = 2$. We search the best b and w for each model on dev-set and apply them on test-set in the following experiments.

Fig. 3 shows the performance of conventional greedy decoding, trivial tail beam search (only after source sentence is finished) and SBS on test set on Chinese↔English tasks. SBS largely boost test-time wait-k models with slightly worse latency (especially in English→Chinese because they tend to generate longer sentences). Wait-k models also benefit from speculation (especially in Chinese→English).

Fig. 5 shows a running example of greedy and SBS output of both wait-k and test-time wait-k models. SBS on test-time wait-k generates much better outputs than the greedy search, which misses some essential information. Wait-k models with speculation correctly translates "拟" into "intends to" instead of "to" in greedy output.



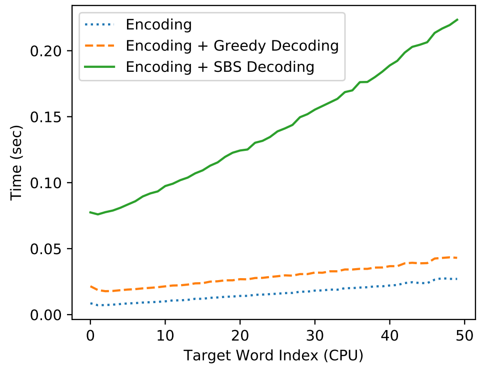


Figure 6: Average time of words with different indices (≤ 50) on zh→en wait-3 model. Results in left figure are measured on GPU while results in right figure results are measured on CPU. The SBS results adopt $w = 2, b = 5$.

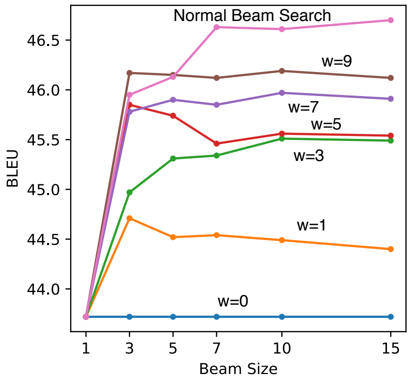


Figure 7: BLEU of SBS over full sentence translation with different window sizes w and beam sizes b. Window size $w = 0$ stands for greedy decoding. The top line stands for results using normal beam search.

4.3 Performance on Adaptive Policy

Fig. 4 shows the performance of proposed SBS on adaptive policies. We train adaptive policies using the combination of Consecutive Wait (CW $\in \{2, 5, 8\}$ (Gu et al., 2017)) and partial-BLEU as reward in reinforcement learning. We vary beam size $b \in \{5, 10\}$ in both chunk-based beam search (Gu et al., 2017) and our SBS with speculative window size $w \in \{2, 4\}$. Our proposed beam search achieves better results in most cases.

4.4 Running Time Analysis

Fig. 6 shows the average time for generating words with different target word indices on a GeForce GTX TITAN-X GPU and an Intel Core i7 2.8 GHz CPU. According to Ma et al. (2019a), wait-k models use bi-directional Transformer as the encoder, thus the time complexity of incrementally encoding one more source word is $O(m^2)$ where m is the source sentence length. This is the reason why it takes more time to encode words with larger index especially using CPU. It is generally accepted that Mandarin speech is about 120–150 syllables per minute, and in our corpus each token (after BPE) has on average 1.5 Chinese syllables (which is 1.5 characters since each Chinese character is monosyllabic), thus in the simultaneous Chinese-to-English speech-to-text translation scenario, the decoder receives a source token every 0.6–0.75 seconds which is much slower than our decoding speed (less than 0.25 seconds per token) even on a laptop CPU. Based on these statistics, our proposed speculative beam search algorithm can be used in real simultaneous translation.

4.5 Performance on Full Sentence MT

We analyze the performance of speculative beam search on full-sentence translation (see Fig. 7). By only performing beam search on a sliding speculative window, the proposed algorithm achieves much better BLEU scores compared with greedy decoding ($w = 0$) and even outperforms conventional beam search when $w = 9$, $b = 3$. Please note that the space complexity of this algorithm is $O((m + n + wb)d)$.[1] This is better than conventional beam search whose space complexity is $O((m + nb)d)$ when $w \ll n$.

5 Conclusions and Future Work

We have proposed speculative beam search for simultaneous translation. Experiments on three approaches to simultaneous translation demonstrate effectiveness of our method. This algorithm has the potential in other incremental tasks such as streaming ASR and incremental TTS.

Acknowledgments

We thank Kaibo Liu for his AL script[2] and help in training wait-k models, and the anonymous reviewers for suggestions.

[1] Here n is the length of target sentence and d is the representation dimension.

[2] https://github.com/SimulTrans-demo/STACL

References

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. *arXiv preprint arXiv:1906.05218*.

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

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proc. of NAACL-HLT*.

Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 493–499, New Orleans, Louisiana. Association for Computational Linguistics.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on empirical methods in natural language processing (EMNLP)*, pages 1342–1352.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O. K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 1053–1062.

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

Navdeep Jaitly, David Sussillo, Quoc V Le, Oriol Vinyals, Ilya Sutskever, and Samy Bengio. 2016. An online sequence-to-sequence model using partial conditioning. In *Advances in Neural Information Processing Systems*, pages 5067–5075.

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

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019a. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036.

Mingbo Ma, Renjie Zheng, and Liang Huang. 2019b. Learning to stop in structured prediction for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1884–1889.

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

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2015. Syntax-based simultaneous translation through prediction of unseen syntactic constituents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 198–207.

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*.

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.

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 *Advances in Neural Information Processing Systems 30*.

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.

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.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019a. Simpler and faster learning of adaptive policies for simultaneous translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019b. Simultaneous translation with flexible policy via restricted imitation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5816–5822.

A Supplemental Material

We also evaluate our work using Consecutive Wait (CW) as latency metric, which measures the average lengths of consecutive wait segments, and perform experiments on German↔English corpora available from WMT15[3]. We use newstest-2013 as dev-set and newstest-2015 as test-set.[4]

Fig. 8 show the translation quality on German↔English against AL of different decoding methods. Consistent to the results of Chinese↔English, our proposed speculative beam search gain large performance boost especially on test-time wait-k. Fig. 9 and Fig. 10 use CW as latency metrics. Since both the wait-k and test-time wait-k models use the same fixed policy, the CW latencies of the same k are identical.

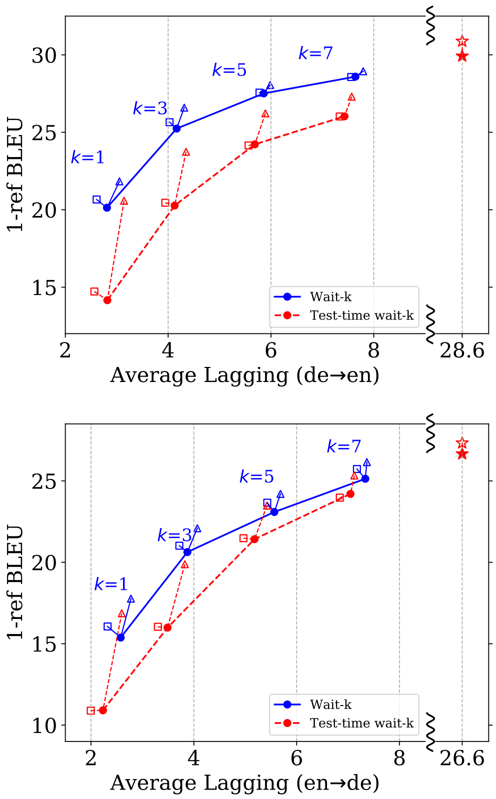


Figure 8: Translation quality against AL on English↔German simultaneous translation using wait-k model. □ □: conventional beam search only on target tail. △ △: speculative beam search. ★☆:full-sentence (greedy and beam-search).

[3] http://www.statmt.org/wmt15/translation-task.html

[4] The German↔English results are slightly different from those in Ma et al. (2019a) because of different decoding settings. We do not allow that the decoder stops earlier than the finish of source sentence while it is allowed in German↔English experiments of Ma et al. (2019a). This makes our generated sentences longer and further results in worse AL compared with the results in Ma et al. (2019a).

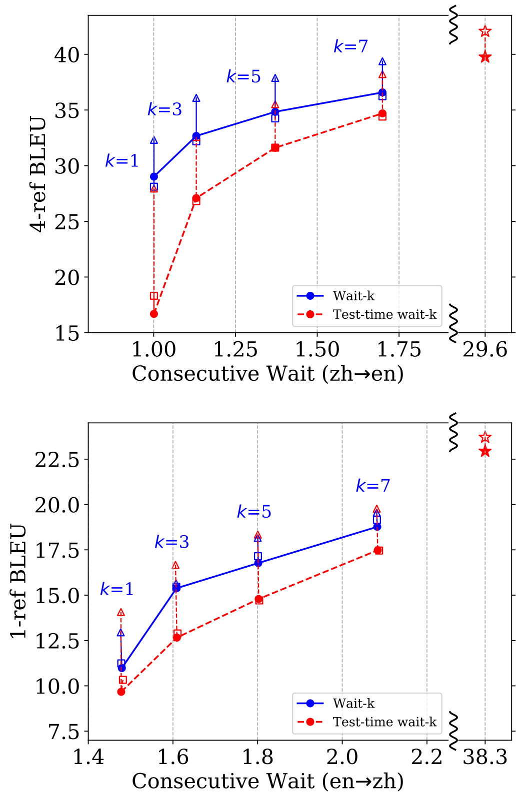


Figure 9: Translation quality against CW on Chinese↔English simultaneous translation using wait-k model.

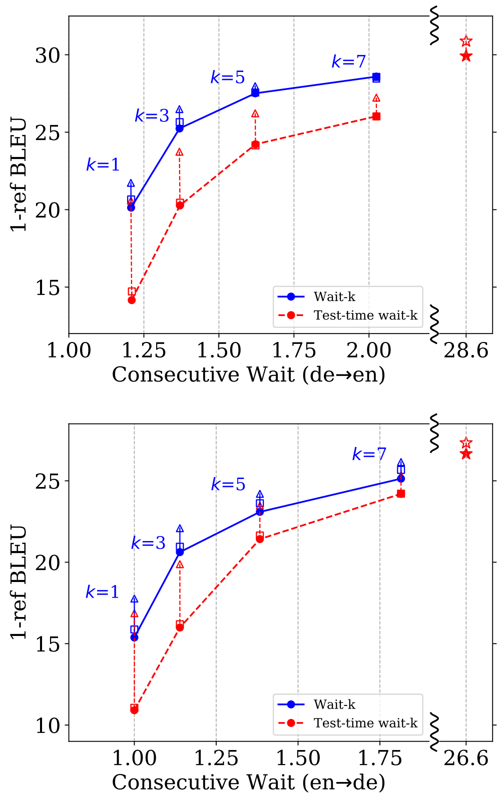


Figure 10: Translation quality against CW on English↔German simultaneous translation using wait-k model.