ORTHROS: NON-AUTOREGRESSIVE END-TO-END SPEECH TRANSLATION WITH DUAL-DECODER

Hirofumi Inaguma[1]*, Yosuke Higuchi*[2]*, Kevin Duh*[3]*, Tatsuya Kawahara*[1]*, Shinji Watanabe*[3]

[1]Kyoto University, Japan [2]Waseda University, Japan [3]Johns Hopkins University, USA

ABSTRACT

Fast inference speed is an important goal towards real-world deployment of speech translation (ST) systems. End-to-end (E2E) models based on the encoder-decoder architecture are more suitable for this goal than traditional cascaded systems, but their effectiveness regarding decoding speed has not been explored so far. Inspired by recent progress in non-autoregressive (NAR) methods in text-based translation, which generates target tokens in parallel by eliminating conditional dependencies, we study the problem of NAR decoding for E2E-ST. We propose a novel NAR E2E-ST framework, *Orthros*, in which both NAR and autoregressive (AR) decoders are jointly trained on the shared speech encoder. The latter is used for selecting better translation among various length candidates generated from the former, which dramatically improves the effectiveness of a large length beam with negligible overhead. We further investigate effective length prediction methods from speech inputs and the impact of vocabulary sizes. Experiments on four benchmarks show the effectiveness of the proposed method in improving inference speed while maintaining competitive translation quality compared to state-of-the-art AR E2E-ST systems.

***Index Terms*—** End-to-end speech translation, non-autoregressive decoding, conditional masked language model

1. INTRODUCTION

There is a growing interest in speech translation [1] due to the increase in demand for international communications. The goal is to transform speech from one language to text in another language. Traditionally, the dominant approach is to cascade automatic speech recognition (ASR) and machine translation (MT) systems. Thanks to recent progress in neural approaches, researchers are shifting to the development of end-to-end speech translation (E2E-ST) systems [2], aiming to optimize a direct mapping from source speech to target text translation by bypassing ASR components. The potential advantages of E2E modeling are (a) mitigation of error propagation from incorrect transcriptions, (b) low-latency inference, and (c) applications in endangered language documentation [3]. However, most efforts have been devoted to investigating methods to improve translation *quality* by the use of additional data [4–12], better parameter initialization [13–15], and improved training methods [16, 17].

For applications of ST systems in lectures and dialogues, inference speed is also an essential factor from the user's perspective [18]. Although E2E models are more suitable for small latency inference than cascaded models since the ASR decoder and the MT encoder processing can be skipped, their effectiveness regarding inference speed has not been well studied. Moreover, incremental left-to-right token generation of *autoregressive* (AR) E2E models increases computational complexity and, therefore, still suffer from slow inference. To speed up inference while achieving comparable translation quality to AR models, *non-autoregressive* (NAR) decoding has been investigated in text-based translation [19–28]. The NAR inference enables to generate tokens in parallel by eliminating conditional token dependencies.

Motivated by this, we propose a novel NAR framework, *Orthros*, for the E2E-ST task. Orthros has dual decoders on the shared speech encoder: NAR and AR decoders. The NAR decoder is used for the fast token generation, and the AR decoder selects better translation among multiple length candidates during inference. As the AR decoder can rescore all tokens in parallel and reuse encoder outputs, its overhead is minimal. This architecture design is motivated by the difficulty of estimating a suitable target length given a speech in advance. We adopt the conditional masked language model (CMLM) [23] for the NAR decoder, in which a subset of tokens are repeatedly updated by partial masking through constant iterations. We also use semi-autoregressive training (SMART) to alleviate mismatches between training and testing conditions [29]. However, any NAR decoder can be used in Orthros, conceptually. Moreover, effective length prediction methods and the impact of vocabulary sizes are also studied.

Experiments on four benchmark corpora show that the proposed framework achieves comparable translation quality to state-of-the-art AR E2E- and cascaded-ST models with approximately 2.31× and 4.61× decoding speed-ups, respectively. Interestingly, the best NAR model can even outperform AR models in terms of the BLEU score in some cases. This work is the first study of NAR models for the E2E-ST task to the best of our knowledge.

2. BACKGROUND

2.1. End-to-end Speech Translation (E2E-ST)

E2E-ST is formulated as a direct mapping problem from input speech $X = (x_1, \dots, x_U)$ in a source language to the target translation text $Y = (y_1, \dots, y_N)$. E2E models can be implemented with any encoder-decoder architectures, and we adopt the state-of-the-art Transformer model [30]. A conventional E2E-ST model is composed of a speech encoder and an *autoregressive* (AR) translation decoder, which decomposes a probability distribution of Y into a chain of conditional probabilities from left to right as:

$$P(Y|X) = \prod_{i=1}^{N} P_{\mathrm{ar}}(y_i|y_{<i}, X). \quad (1)$$

Parameters are optimized with a single translation objective $\mathcal{L}_{\mathrm{ar}} = -\log P_{\mathrm{ar}}(Y|X)$ after pre-training with the ASR and MT tasks [13].

2.2. Conditional masked language model (CMLM)

Since AR models generate target tokens incrementally during inference, they suffer from high inference latency and do not fully leverage the computational power of modern hardware such as GPU. In order to tackle this problem, parallel sequence generation with *non-autoregressive* (NAR) models have been investigated in a wide range

Table 1: An example of Mask-Predict decoding on the Fisher-dev set. Highlighted tokens are masked for the next iteration.

Reference	but with that and and when we bought it i thought who's gonna put a you know a walkman or something and that's what i'm doing now
$t = 4$	he came with that and when we bought it i thought who going to put you know know ak or or something and now i'm doing it
$t = 7$	he came with that and when we bought it i thought who he going to put you know akman or something and now i'm doing it
$t = 10$	he came with that and when we bought it i thought who is going to put you know a walkman or something and now i'm doing it

of tasks such as text-to-speech synthesis [31, 32], machine translation [19], and speech recognition [33–38]. NAR models factorize conditional probabilities in Eq. (1) by assuming conditional independence for every target position. However, iterative refinement methods generally achieve better quality than pure NAR methods at the cost of speed because of multiple forward passes [20, 23, 39]. Among them, the conditional masked language model (CMLM) [23] is a natural choice because of its simplicity and good performance. Moreover, we can flexibly trade the speed during inference by changing the number of iterations T.

In CMLM, the partial decoder inputs are masked by replacing with a unique token `[MASK]` based on confidence. Intermediate discrete variables at the t-th iteration $Y^{(t)}$ $(1 \leq t \leq T)$ are iteratively refined given the rest observed tokens $Y^{(t)}_{\text{obs}} \subset Y^{(t-1)}$ as:

$$P(Y|X) = \prod_{t=1}^{T} P_{\text{nar}}(Y^{(t)}|Y^{(t)}_{\text{obs}}, X). \tag{2}$$

The target length distribution for N is typically modeled by a linear classifier stacked on the encoder.

However, NAR models suffer from a *multimodality* problem, where a distribution of multiple correct translations must be modeled given the same source sentence. Recent studies reveal that sequence-level knowledge distillation [40] makes training data deterministic and mitigates this problem [19, 41, 42].

3. PROPOSED METHOD: ORTHROS

3.1. Model architecture

Typical text-based NAR models generate target sentences with multiple lengths in parallel to improve quality in a stochastic [19] or deterministic [23] way, followed by an optional rescoring step with a separate AR model [19]. However, a spoken utterance consists of hundreds of acoustic frames even after downsampling, and its length varies significantly based on the speaking rate and silence duration. Therefore, it is challenging to estimate the target length given a speech in advance accurately. Moreover, extra computation and memory consumption for feature encoding with the separate AR model in rescoring are not negligible. This motivated us to propose *Orthros*, having dual decoders on top of the *shared* encoder: an NAR decoder for fast token generation and an AR decoder for candidate selection from the NAR decoder. This way, re-encoding speech frames is unnecessary, and the AR decoder greatly improves the effectiveness of using a large length beam. The speech encoder is identical to that of AR models. A length predictor and a CTC ASR layer for the auxiliary ASR task are also built on the same encoder.

Our NAR decoder is based on the conditional masked language model (CMLM) [23]. One of the distinguished advantages over pure NAR models [19] is that CMLM removes the necessity of a copy of the source sentence to initialize decoder inputs. This could be achieved by using a predicted transcription from the auxiliary ASR sub-module, but this contradicts a motivation to avoid ASR errors.

3.2. Inference

The inference of the CMLM is based on the *Mask-Predict* algorithm [23]. Let T be the number of iterations, $\hat{N}$ be a predicted target sequence length, and $\hat{Y}^{(t)}_{\text{mask}}$ and $\hat{Y}^{(t)}_{\text{obs}}$ be masked and observed tokens in the prediction $\hat{Y}^{(t-1)}$ ($|\hat{Y}^{(t-1)}| = \hat{N}$) at the t-th iteration ($1 \leq t \leq T$), respectively. At the initial iteration $t = 0$, all tokens are initialized with `[MASK]`. An example is shown in Table 1.

3.2.1. Mask-Predict

The mask-predict algorithm performs two operations, *mask* and *predict*, at every iteration t. In the *mask* operation, given predicted tokens at the previous iteration, $\hat{Y}^{(t-1)}$, we mask k_t tokens having the lowest confidence scores, where k_t is a linear decay function $k_t = \lfloor \hat{N} \cdot \frac{T-t}{t} \rfloor$. In the *predict* operation, we take the most probable token from a posterior probability distribution P_{cmlm} at every masked position i and update $\hat{y}^{(t)}_i \in \hat{Y}^{(t)}_{\text{mask}}$ as:

$$\hat{y}^{(t)}_i = \underset{w_i \in V}{\text{argmax}}\, P_{\text{cmlm}}(w_i|\hat{Y}^{(t)}_{\text{obs}}, X), \tag{3}$$

where V is the vocabulary. When using SMART, described in Section 3.3.1, all tokens $\hat{y}^{(t)}_i \in \hat{Y}^{(t)}$ are updated in Eq. (3) if they differ from those at the previous iteration. Furthermore, we generate l target sentences having different lengths in parallel and select the most probable candidate by calculating the average log probability over all tokens at the last iteration: $\frac{1}{N}\sum_i \log P^{(T)}_{i,\text{cmlm}}$.

For target length prediction, we sample top-l ($l \geq 1$) length candidates from a linear classifier conditioned on time-averaged encoder outputs. We also study a simple scaling method using CTC outputs used for the auxiliary ASR task.

3.2.2. Candidate selection with AR decoder

Using multiple length candidates is effective for improving quality, but it is sub-optimal to directly use sequence-level scores from $P^{(T)}_{i,\text{cmlm}}$ in Eq. (3) because they are stale [23]. Therefore, we propose to select the most probable translation among l candidates after the last iteration by using log probability scores from the AR decoder averaged over all tokens. Note that we do not use scores from the NAR decoder here. Since the AR decoder can rescore all tokens in a candidate in parallel, it can still maintain the advantage of parallelism in self-attention.

3.3. Training

The training objective of the CMLM can be formulated as follows:

$$\mathcal{L}_{\text{cmlm}} = -\sum_{y \in Y_{\text{mask}}} \log P_{\text{cmlm}}(y|Y_{\text{obs}}, X), \tag{4}$$

where $Y_{\text{mask}} \subset Y$ and $Y_{\text{obs}} = Y \setminus Y_{\text{mask}}$. We sample the number of masked tokens from a uniform distribution, $\mathcal{U}(1, N)$, following [23].

3.3.1. Semi-autoregressive training (SMART)

To bridge the gap between training and test conditions in the CMLM, we adopt semi-autoregressive training (SMART) [29]. SMART uses two forward passes to calculate the cross-entropy (CE) loss. In the first pass, the CMLM generates predictions at all positions, $\hat{Y}$, given partially-observed ground-truth tokens Y_{obs} as in the original training process. The gradient flow is truncated in the first pass [29]. Then, a subset of tokens in $\hat{Y}$ are masked again with a new mask.

Table 2: BLEU scores of AR and NAR methods on the tst-COMMON sets of Must-C (En→De and En→Fr), Fisher-test (Fsh-test) and CallHome-evltest (CH-evltest) sets of Fisher-CallHome Spanish (Es→En), and the test set of Libri-trans (En→Fr). Seq-KD represents sequence-level knowledge distillation. Latency is measured as average decoding time per sentence on Must-C En→De, with batch size 1.

		ID	Model	BLEU					Latency (ms)	Speedup
				Must-C		Fisher-CallHome		Libri-trans		
				De	Fr	Fsh-test	CH-evltest			
E2E	AR	A1	Transformer ($b = 1$)	21.54	32.26	48.38	18.07	16.52	175	1.54 ×
			Transformer ($b = 4$)	**23.12**	**33.84**	**48.49**	**18.90**	**16.84**	271	1.00 ×
		A2	Transformer + Seq-KD ($b = 1$)	23.88	33.92	50.34	19.09	15.91	–	–
			Transformer + Seq-KD ($b = 4$)	**24.43**	**34.57**	**50.32**	**19.81**	16.44	–	–
	NAR	N1	CTC ($b = 1$)	19.40	27.38	45.97	15.91	12.10	13	20.84 ×
		N2	Orthros (CMLM, $T = 4$)	18.78	25.99	46.03	16.71	12.90	–	–
			Orthros (CMLM, $T = 4$ +AR)	19.62	27.77	47.80	18.28	13.69	–	–
			Orthros (CMLM, $T = 10$)	20.89	28.74	48.56	18.60	14.68	–	–
			Orthros (CMLM, $T = 10$ +AR)	21.79	30.31	49.98	19.71	15.43	–	–
		N3	Orthros (SMART, $T = 4$)	20.03	27.22	45.89	17.39	14.17	46	5.89 ×
			Orthros (SMART, $T = 4$ +AR)	21.08	29.30	48.73	19.25	14.99	61	4.44 ×
			Orthros (SMART, $T = 10$)	21.25	29.31	47.09	18.25	15.11	99	2.73 ×
			Orthros (SMART, $T = 10$ +AR)	**22.27**	**31.07**	**50.07**	**20.10**	**16.08**	111	2.44 ×
		N4	+ BPE8k→16k	**22.88**	**32.20**	**50.18**	19.88	**16.22**	117	2.31 ×
		N5	+ large (SMART, $T = 4$ +AR, $l = 7$)	22.54	31.24	–	–	–	59	4.59 ×
			+ large (SMART, $T = 10$ +AR, $l = 7$)	**23.92**	**33.05**	–	–	–	113	2.39 ×
Cascade	AR	A3	AR ASR ($b = 1$) → AR MT ($b = 1$)	22.20	31.67	40.94	19.15	16.44	154→166	0.84 ×
			AR ASR ($b = 4$) → AR MT ($b = 4$)	23.30	33.40	42.05	19.77	16.52	333→207	0.50 ×

The resulting observed tokens $\hat{Y}_{\rm obs}$ are fed into the decoder as inputs in the second pass. The CE loss is calculated with predictions at *all* positions in the second pass, unlike the original training.

3.3.2. Total training objective

The speech encoder and all four branches (NAR/AR decoders, length predictor, and CTC ASR layer) are optimized jointly. The total objective function is formulated as:

$$\mathcal{L}_{\rm total} = (1-\lambda_{\rm asr})\mathcal{L}_{\rm cmlm}(Y|X) + \lambda_{\rm ar}\mathcal{L}_{\rm ar}(Y|X) + \lambda_{\rm lp}\mathcal{L}_{\rm lp}(N|X) + \lambda_{\rm asr}\mathcal{L}_{\rm asr}(Y^{\rm src}|X),$$

where $\mathcal{L}_{\rm ar}$, $\mathcal{L}_{\rm lp}$, and $\mathcal{L}_{\rm asr}$ are losses in AR E2E-ST, length prediction, and ASR tasks, $Y^{\rm src}$ is the corresponding transcription, and λ_* are the corresponding tunable hyperparameters. We set ($\lambda_{\rm ar}$, $\lambda_{\rm lp}$, $\lambda_{\rm asr}$) to (0.3, 0.1, 0.3) throughout the experiments.

4. EXPERIMENTAL EVALUATION

4.1. Datasets

We used En-De (229k pairs, 408 hours) and En-Fr (275k pairs, 492 hours) language directions on Must-C [43], Fisher-CallHome Spanish (Es-En, 138k pairs, 170 hours, hereafter Fisher-CH) [44], and Libri-trans (En-Fr, 45k pairs, 100 hours) [45]. All corpora contain a triplet of source speech and the corresponding transcription and translation, and we used the same preprocessing as [46]. For Must-C, we report case-sensitive detokenized BLEU [47] on the `tst-COMMON` set. Non-verbal speech labels such as "*(Applause)*" were removed during evaluation. For Fisher-CH, we report case-insensitive detokenized BLEU on the Fisher-`test` (four references) and CallHome-`evltest` sets. For Libri-trans, we report case-insensitive BLEU on the `test` set. We removed case information and all punctuation marks except for apostrophe in both transcriptions of all corpora and translations of Fisher-CH.

We extracted 80-channel log-mel filterbank coefficients with 3-dimensional pitch features using Kaldi [48] as input speech features, which was augmented by a factor of 3 with speed perturbation [49] and SpecAugment [50] to avoid overfitting. All sentences were tokenized with the `tokenizer.perl` script in Moses [51]. We built vocabularies based on byte pair encoding (BPE) algorithm [52] implemented with Sentencepiece [53]. The joint source and target vocabularies were used in the ST/MT tasks, while the ASR vocabularies were constructed with the transcriptions only. For Must-C, we used 5k and 8k vocabularies for ASR and E2E-ST/MT models, respectively. For Fisher-CH and Libri-trans, we used 8k and 1k vocabularies for NAR E2E-ST models and the others, respectively.

4.2. Model configurations

We used the Transformer architecture implemented in ESPnet-ST [46] for all tasks. All ASR and E2E-ST models consisted of stacked 12 encoder layers and 6 decoder layers. Speech encoders had 2 CNN layers before the self-attention layers, which performed 4-fold downsampling. The text encoders in the MT models consisted of 6 layers. The dimension of self-attention layer $d_{\rm model}$ and feed-forward network $d_{\rm ff}$, and the number of heads H were set to 256, 2048, and 4, respectively. For the large model on Must-C, we set $d_{\rm model} = 512$ and $H = 8$. The Adam optimizer [54] with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$ was used for training with Noam learning rate schedule [30]. Warmup steps and a learning rate constant were set to 25000 and 5.0, respectively. A mini-batch was constructed with 32 utterances, and gradients were accumulated for 8 steps in NAR E2E-ST models. The last 5 best checkpoints based on the validation performance were used for model averaging. Following the standard practice in NAR models [19, 23], we used sequence-level knowledge distillation (Seq-KD) [40] with the corresponding AR Transformer MT model, except for Libri-trans. During inference, we used a beam width $b \in \{1, 4\}$ for AR ASR/ST/MT models, and a length beam width $l = 9$ for NAR models. The language model was used for the ASR model on Libri-trans only. Joint CTC/Attention decoding [55] was performed for ASR models. Decoding time was measured with a batch size 1 on a single NVIDIA TITAN RTX GPU by averaging on five runs. We initialized encoder parameters of E2E-ST models with those of the corresponding pre-trained ASR model and AR decoder parameters with those of the corresponding pre-trained AR MT model trained on the same triplets, respectively [13]. However, NAR decoder parameters were initialized based on the weight initialization scheme in BERT [23, 56].

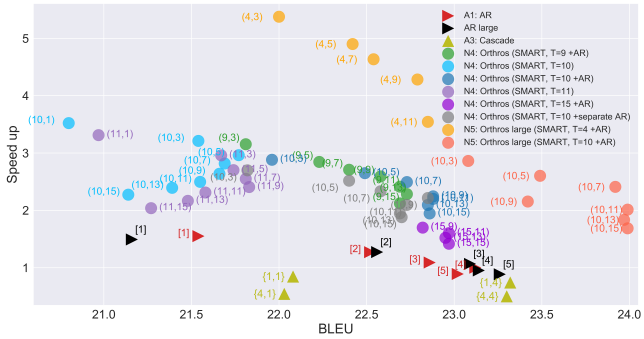

Fig. 1: Trade-off between decoding speed-up and BLEU scores on the Must-C En-De tst-COMMON set. Parentheses, square brackets, and curly brackets represent (T, l), [b], and {b (ASR), b (MT)}, respectively.

Table 3: Ablation study on the Fisher-dev set

Model	BLEU			
	$T = 4$		$T = 10$	
	w/o AR	w/ AR	w/o AR	w/ AR
Orthros (N3)	**45.76**	**49.01**	46.88	**50.28**
- Seq-KD	44.36	47.42	44.25	49.50
- AR decoder	45.53	N/A	**46.94**	N/A
+ length prediction w/ CTC	45.41	48.18	46.79	50.05

4.3. Main results

The main results are shown in Table 2. Iterative refinement based on CMLM (N2) significantly outperformed the pure NAR CTC model (N1) in translation quality.[1] Increasing the number of iterations T was effective for improving quality at the cost of latency. SMART also boosted the BLEU scores with no extra latency during inference, except for Fisher-CH (N3). This is probably because sequence lengths in Fisher-CH are relatively short. Candidate selection with the AR decoder greatly improved the quality with a negligible increase of latency, which corresponds to performing one more iteration. We also found that NAR models prefer the large vocabulary size for better quality. Using BPE16k, BLEU scores were improved while keeping latency (N4). We note that the BPE size was tuned separately for AR and NAR models. We will analyze this phenomenon in Section 4.5. Increasing the model capacity also improved the quality of NAR models while it did not hurt the speed so much when using GPU. AR models did not benefit from the larger capacity on this corpus though not shown in the table (see Fig. 1).

For a comparison with AR models, Orthros achieved comparable quality to both strong AR E2E (A1, A2) and cascaded systems (A3) with smaller latency. Interestingly, N4 and N5 even outperformed A1 in quality by a large margin on Fisher-CH and Must-C En-De, respectively. Seq-KD was very effective for AR models as well except for Libri-trans. Although relative speed-ups are smaller than those in the MT literature [23, 39], this is probably because we used the smaller vocabulary and the baseline AR models have much smaller latency (e.g., 607ms in [39] vs. 271ms in ours).

Fig. 1 shows the trade-off between relative speed-ups and BLEU scores on the Must-C En-De tst-COMMON set. Consistent with text-based CMLM models [23], a large length beam width l was not effective. However, the proposed candidate selection significantly improved the performances with a larger l. This way, a similar BLEU score can be achieved with a smaller iteration. Moreover,

[1]CTC in the E2E-ST task has the speech encoder and the output classifier. It was optimized with a single CTC objective with a pair of (X, Y). Since input speech lengths are generally longer than target sequence lengths in the E2E-ST task, we did not use the upsampling technique in [22].

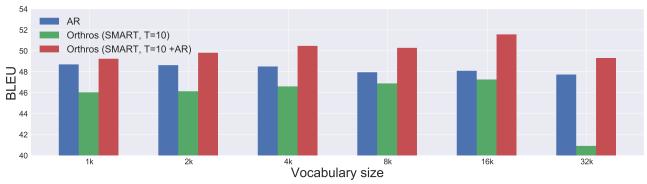

Fig. 2: Impact of vocabulary size on the Fisher-dev set

Orthros (N4) can obtain the same BLEU as a baseline AR (A1) with more than 3 times speed-up for greedy decoding and 1.5 times for beam search. The large Orthros (N5) achieved better BLEU scores than N4 with similar latency and outperformed the AR models with beam search both in quality and latency. Although the cascaded models showed reasonable BLEU scores, they were much slower than the E2E models. We also compared AR models for candidate selection: the AR decoder on the unified encoder (proposal) vs. the separate AR encoder-decoder. The unified encoder showed smaller latency with better quality. We suspect that this is because sharing the encoder has a positive effect on candidate selection, or the AR decoder in Orthros was trained with Seq-KD. We will analyze this in future work. Although the overhead for additional speech encoding was relatively small here, this would be enlarged when using a more complicated encoder architecture. One more advantage of Orthros is that the memory consumption for model parameters and encoder output caching is much smaller.

4.4. Ablation study

To see individual contributions of the proposed techniques, we conducted the ablation study on the Fisher-dev set in Table 3. Seq-KD was beneficial for boosting BLEU scores consistent with the NAR MT task [23]. Joint training with the AR decoder did not hurt BLEU scores when candidate selection was not used. For length prediction, we also investigated a simple approach by scaling the transcription length $\hat{N}_{\mathrm{src}}$, which was obtained from the CTC ASR layer with greedy decoding, by a constant value α, i.e., $\hat{N} = \lfloor \alpha \hat{N}_{\mathrm{src}} \rfloor$. Although this works as well, we needed to tune α on the dev set on each corpus, and therefore we adopted the classification approach.

4.5. Effect of vocabulary size

Finally, we investigated the impact of vocabulary size. Fig. 2 shows BLEU scores of AR and NAR E2E models as a function of the vocabulary size on the Fisher-dev set. We observed AR models have a peak around 1k BPE because the data size is relatively small (170-hours). However, the performance of NAR models continued to improve according to the vocabulary size until 16k. The candidate selection with the AR decoder was beneficial for all the vocabulary sizes, especially for 32k. This is probably because misspelling was alleviated thanks to many *complete* words in the large vocabulary, which had a complementary effect on the conditional independence assumption made in the NAR models. We also observed similar trends in other corpora and CTC models.

5. CONCLUSION

In this work, we proposed a unified NAR decoding framework to speed-up inference in the E2E-ST task, *Orthros*, with NAR and AR decoders on the shared encoder. Selecting the better candidate with the AR decoder greatly improved the effectiveness of a large length beam in the NAR decoder. We also presented that using a large vocabulary and parameters is effective for NAR E2E-ST models. The best NAR E2E model reached a level of state-of-the-art AR Transformer model in the BLEU score while reducing inference latency more than twice.

6. REFERENCES

[1] Hermann Ney, "Speech translation: Coupling of recognition and translation," in *Proc. of ICASSP*, 1999, pp. 517–520.

[2] Alexandre Bérard et al., "Listen and translate: A proof of concept for end-to-end speech-to-text translation," in *Proc. of NeurIPS 2016 End-to-end Learning for Speech and Audio Processing Workshop*, 2016.

[3] Marcely Zanon Boito et al., "Unwritten languages demand attention too! word discovery with encoder-decoder models," in *Proc. of ASRU*, 2017, pp. 458–465.

[4] Ye Jia et al., "Leveraging weakly supervised data to improve end-to-end speech-to-text translation," in *Proc. of ICASSP*, 2019, pp. 7180–7184.

[5] Juan Pino et al., "Harnessing indirect training data for end-to-end automatic speech translation: Tricks of the trade," *arXiv*, pp. arXiv–1909, 2019.

[6] Chengyi Wang et al., "Bridging the gap between pre-training and fine-tuning for end-to-end speech translation," in *Proc. of AAAI*, 2020, pp. 9161–9168.

[7] Chengyi Wang et al., "Curriculum pre-training for end-to-end speech translation," in *Proc. of ACL*, 2020, pp. 3728–3738.

[8] Hirofumi Inaguma et al., "Multilingual end-to-end speech translation," in *Proc. of ASRU*, 2019, pp. 570–577.

[9] Mattia Antonino Di Gangi et al., "One-to-many multilingual end-to-end speech translation," in *Proc. of ASRU*, 2019, pp. 585–592.

[10] Juan Pino et al., "Self-training for end-to-end speech translation," in *Proc. of Interspeech*, 2020.

[11] Elizabeth Salesky et al., "Exploring phoneme-level speech representations for end-to-end speech translation," in *Proc. of ACL*, 2019, pp. 1835–1841.

[12] Elizabeth Salesky et al., "Phone features improve speech translation," in *Proc. of ACL*, 2020, pp. 2388–2397.

[13] Alexandre Bérard et al., "End-to-end automatic speech translation of audiobooks," in *Proc. of ICASSP*, 2018, pp. 6224–6228.

[14] Sameer Bansal et al., "Pre-training on high-resource speech recognition improves low-resource speech-to-text translation," in *Proc. of NAACL-HLT*, 2019, pp. 58–68.

[15] Sathish Indurthi et al., "End-end speech-to-text translation with modality agnostic meta-learning," in *Proc. of ICASSP*, 2020, pp. 7904–7908.

[16] Ron J Weiss et al., "Sequence-to-sequence models can directly translate foreign speech," in *Proc. of Interspeech*, 2017, pp. 2625–2629.

[17] Yuchen Liu et al., "End-to-end speech translation with knowledge distillation," in *Proc. of Interspeech*, 2019, pp. 1128–1132.

[18] Jan Niehues et al., "Dynamic transcription for low-latency speech translation.," in *Proc. of Interspeech*, 2016, pp. 2513–2517.

[19] Jiatao Gu et al., "Non-autoregressive neural machine translation," in *Proc. of ICLR*, 2018.

[20] Jason Lee et al., "Deterministic non-autoregressive neural sequence modeling by iterative refinement," in *Proc. of EMNLP*, 2018, pp. 1173–1182.

[21] Lukasz Kaiser et al., "Fast decoding in sequence models using discrete latent variables," in *Proc. of ICML*, 2018, pp. 2390–2399.

[22] Jindřich Libovickỳ et al., "End-to-end non-autoregressive neural machine translation with connectionist temporal classification," in *Proc. of EMNLP*, 2018, pp. 3016–3021.

[23] Marjan Ghazvininejad et al., "Mask-predict: Parallel decoding of conditional masked language models," in *Proc. of EMNLP*, 2019, pp. 6114–6123.

[24] Jiatao Gu et al., "Levenshtein Transformer," in *Proc. of NeurIPS*, 2019, pp. 11181–11191.

[25] Mitchell Stern et al., "Insertion Transformer: Flexible sequence generation via insertion operations," in *Proc. of ICML*, 2019, pp. 5976–5985.

[26] Xuezhe Ma et al., "Flowseq: Non-autoregressive conditional sequence generation with generative flow," in *Proc. of EMNLP*, 2019, pp. 4273–4283.

[27] Marjan Ghazvininejad et al., "Aligned cross entropy for non-autoregressive machine translation," in *Proc. of ICML*, 2020.

[28] Chitwan Saharia et al., "Non-autoregressive machine translation with latent alignments," *arXiv preprint arXiv:2004.07437*, 2020.

[29] Marjan Ghazvininejad et al., "Semi-autoregressive training improves mask-predict decoding," *arXiv preprint arXiv:2001.08785*, 2020.

[30] Ashish Vaswani et al., "Attention is all you need," in *Proc. of NeurIPS*, 2017, pp. 5998–6008.

[31] Aaron Oord et al., "Parallel wavenet: Fast high-fidelity speech synthesis," in *Proc. of ICML*, 2018, pp. 3918–3926.

[32] Yi Ren et al., "FastSpeech: Fast, robust and controllable text to speech," in *Proc. of NeurIPS*, 2019, pp. 3171–3180.

[33] N Chen et al., "Listen and fill in the missing letters: Non-autoregressive Transformer for speech recognition," *arXiv preprint arXiv:1911.04908*, 2019.

[34] William Chan et al., "Imputer: Sequence modelling via imputation and dynamic programming," *arXiv preprint arXiv:2002.08926*, 2020.

[35] Yosuke Higuchi et al., "Mask CTC: Non-autoregressive end-to-end ASR with CTC and mask predict," in *Proc. of Interspeech*, 2020, pp. 3655–3659.

[36] Yuya Fujita et al., "Insertion-based modeling for end-to-end automatic speech recognition," in *Proc. of Interspeech*, 2020, pp. 3660–3664.

[37] Ye Bai et al., "Listen attentively, and spell once: Whole sentence generation via a non-autoregressive architecture for low-latency speech recognition," in *Proc. of Interspeech*, 2020, pp. 3381–3385.

[38] Zhengkun Tian et al., "Spike-triggered non-autoregressive Transformer for end-to-end speech recognition," in *Proc. of Interspeech*, 2020, pp. 5026–5030.

[39] Junliang Guo et al., "Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation," in *Proc. of ACL*, 2020, pp. 376–385.

[40] Yoon Kim et al., "Sequence-level knowledge distillation," in *Proc. of EMNLP*, 2016, pp. 1317–1327.

[41] Chunting Zhou et al., "Understanding knowledge distillation in non-autoregressive machine translation," in *Proc. of ICLR*, 2019.

[42] Yi Ren et al., "A study of non-autoregressive model for sequence generation," in *Proc. of ACL*, 2020, pp. 149–159.

[43] Mattia A. Di Gangi et al., "MuST-C: a Multilingual Speech Translation Corpus," in *Proc. of NAACL-HLT*, 2019, pp. 2012–2017.

[44] Matt Post et al., "Improved speech-to-text translation with the Fisher and Callhome Spanish–English speech translation corpus," in *Proc. of IWSLT*, 2013.

[45] Ali Can Kocabiyikoglu et al., "Augmenting Librispeech with French translations: A multimodal corpus for direct speech translation evaluation," in *Proc. of LREC*, 2018.

[46] Hirofumi Inaguma et al., "ESPnet-ST: All-in-one speech translation toolkit," in *Proc. of ACL: System Demonstrations*, 2020, pp. 302–311.

[47] Kishore Papineni et al., "Bleu: a method for automatic evaluation of machine translation," in *Proc. of ACL*, 2002, pp. 311–318.

[48] Daniel Povey et al., "The kaldi speech recognition toolkit," in *Proc. of ASRU*, 2011.

[49] Tom Ko et al., "Audio augmentation for speech recognition," in *Proc. of Interspeech*, 2015, pp. 3586–3589.

[50] Daniel S Park et al., "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. of Interspeech*, 2019, pp. 2613–2617.

[51] Philipp Koehn et al., "Moses: Open source toolkit for statistical machine translation," in *Proc. of ACL: Demo and Poster Sessions*, 2007, pp. 177–180.

[52] Rico Sennrich et al., "Neural machine translation of rare words with subword units," in *Proc. of ACL*, 2016, pp. 1715–1725.

[53] Taku Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proc. of ACL*, 2018, pp. 66–75.

[54] Diederik Kingma et al., "Adam: A method for stochastic optimization," *Proc. of ICLR*, 2015.

[55] Shinji Watanabe et al., "Hybrid CTC/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[56] Jacob Devlin et al., "BERT: Pre-training of deep bidirectional Transformers for language understanding," in *Proc. of NAACL-HLT*, 2019, pp. 4171–4186.