Investigating the Reordering Capability in CTC-based Non-Autoregressive End-to-End Speech Translation

Shun-Po Chuang[*]   **Yung-Sung Chuang**[*]   **Chih-Chiang Chang**[*]   **Hung-yi Lee**

National Taiwan University, Taipei, Taiwan

{f04942141,b05901033,r09922057,hungyilee}@ntu.edu.tw

Abstract

We study the possibilities of building a non-autoregressive speech-to-text translation model using connectionist temporal classification (CTC), and use CTC-based automatic speech recognition as an auxiliary task to improve the performance. CTC's success on translation is counter-intuitive due to its monotonicity assumption, so we analyze its reordering capability. Kendall's tau distance is introduced as the quantitative metric, and gradient-based visualization provides an intuitive way to take a closer look into the model. Our analysis shows that transformer encoders have the ability to change the word order and points out the future research direction that worth being explored more on non-autoregressive speech translation.[1]

1 Introduction

Recently, there are more and more research works focusing on end-to-end speech translation (ST) (Bérard et al., 2016; Weiss et al., 2017; Bérard et al., 2018; Vila et al., 2018; Di Gangi et al., 2019; Ran et al., 2019; Chuang et al., 2020). Instead of cascading machine translation (MT) models to an automatic speech recognition (ASR) system, end-to-end models can skip the error bottleneck caused by ASR and be more computationally efficient. However, in the inference time, an autoregressive (AR) decoder is needed to decode the output sequence, causing the latency issue.

In MT, non-autoregressive (NAR) models have been heavily explored recently (Gu et al., 2018; Lee et al., 2018; Ghazvininejad et al., 2019; Stern et al., 2019; Gu et al., 2019; Saharia et al., 2020) by leveraging the parallel nature of transformer (Vaswani et al., 2017). In contrast, such kind of models is rarely explored in the field of speech translation, except for a concurrent work (Inaguma et al.,

---

[*]Contributed equally.

[1]The source code is available. See Appendix A.

2020). In this work, we use connectionist temporal classification (CTC) (Graves et al., 2006) to train NAR models for ST, without an explicit decoder module. Our entire model is merely a transformer encoder. Multitask learning (Anastasopoulos and Chiang, 2018; Kano et al., 2021) on ASR, which is often used in speech translation, can also be applied in our transformer encoder architecture to further push the performance. We achieve initial results on NAR speech translation by using a single speech encoder.

CTC's success on the translation task is counter-intuitive because of its monotonicity assumption. Previous works directly adopt the CTC loss on NAR translation without further verification on the reordering capability of CTC (Libovický and Helcl, 2018; Saharia et al., 2020; Inaguma et al., 2020). To further understand the reason that the CTC-based model can achieve ST task, we analyze the ordering capabilities of ST models by leveraging Kendall's tau distance (Birch and Osborne, 2011; Kendall, 1938), and a gradient-based visualization is introduced to provide additional evidence. To the best of our knowledge, this is the first time to examine the ordering capabilities on the ST task.

We found that after applying multitask training, our model can have more tendency to re-arrange the positions of the target words to better positions that are not aligned with audio inputs. We highlight that our contribution is to 1) take the first step on translating pure speech signal to target language text in a NAR end-to-end manner and 2) take a closer look at the reason that NAR model with CTC loss can achieve non-monotonic mapping.

2 Approaches

2.1 CTC-based NAR-ST Model

We adopt transformer architecture for non-autoregressive speech-to-text translation (NAR-ST). The NAR-ST model consists of convolutional

layers and self-attention layers. The audio sequence X is downsampled by convolutional layers, and self-attention layers will generate the final translation token sequence Y based on the downsampled acoustic features. We use CTC loss as the objective function to optimize the NAR-ST model. The CTC decoding algorithm allows the model to generate translation in a single step.

CTC predicts an alignment between the input audio sequence X and the output target sequence Y by considering the probability distribution marginalized over all possible alignments.

The CTC loss function is defined as:

$$\mathcal{L}_{\text{CTC}} = - \sum_{(x,y) \in D} \log p_\theta(y|x), \qquad (1)$$

where x is audio frame sequence, y is target sequence, and D is the training set. CTC uses dynamic programming to marginalize out the latent alignments to compute the log-likelihood:

$$\log p_\theta(y|x) = \log \sum_{a \in \beta(y)} \prod_t p(a_t|x; \theta) \quad (2)$$

where $a = \{a_t\}_{t=0}^{|x|}$ is an alignment between x and y and is allowed to include a special "blank" token that should be removed when converting a to the target sequence y. $\beta^{-1}(a)$ is a collapsing function such that $\beta^{-1}(a) = y$ if $a \in \beta(y)$.

CTC has a strong conditional independence and monotonicity assumption. It means that the tokens in Y can be generated independently, and there exists a monotonic alignment between X and Y. The monotonicity property is suitable for tasks such as ASR. However, in translation tasks, there is no guarantee that the output sequence should follow the assumption, as word orders differ in different languages. In this work, we want to examine whether the powerful self-attention based transformer model can overcome this problem to some degree or not.

2.2 CTC-based Multitask NAR-ST Model

Multitask learning improves data efficiency and performance across various tasks (Zhang and Yang, 2017). In AR end-to-end ST, multitask learning technique is often applied using ASR as an auxiliary task (Anastasopoulos and Chiang, 2018; Sperber and Paulik, 2020). It requires an ASR decoder in addition to the ST decoder to learn to predict transcriptions while sharing the encoder.
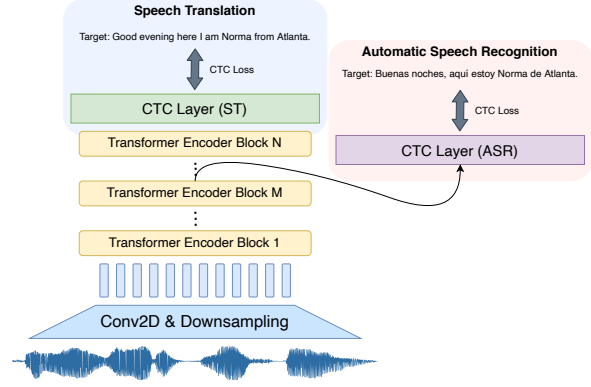

Figure 1: Multitask CTC model for ST and ASR.

To perform multitask learning on NAR-ST model, we propose to apply CTC-based ASR on a single M-th layer in the model, as illustrated in Figure 1. It helps the NAR-ST model capture more information with a single CTC layer in an end-to-end manner. And the ASR output will not be involved in the translation decoding process.

2.3 Reordering Evaluation – Kendall's Tau Distance

We measure reordering degree by Kendall's tau distance (Kendall, 1938). LRscore (Birch and Osborne, 2011) also introduced the distance with consideration of lexical correctness. Different from LRscore, we purely analyze the reordering capability rather than lexical correctness in this work.

Given a sentence triplet (T, H, Y), where $T = \langle t_1, ..., t_{|T|} \rangle$ is the audio transcription. $H = \langle h_1, ..., h_{|H|} \rangle$ and $Y = \langle y_1, ..., y_{|Y|} \rangle$ are hypothesis and reference translation, respectively. An external aligner provides two alignments: $\boldsymbol{\pi} = \langle \pi(1), ..., \pi(|T|) \rangle$ and $\boldsymbol{\sigma} = \langle \sigma(1), ..., \sigma(|T|) \rangle$. $\boldsymbol{\pi}$ maps each source token t_k to a reference token $y_{\pi(k)}$, and $\boldsymbol{\sigma}$ maps t_k to a hypothesis token $h_{\sigma(k)}$. We follow the simplifications proposed in LRscore to reduce the alignments to a bijective relationship. The proportion of disagreements between $\boldsymbol{\pi}$ and $\boldsymbol{\sigma}$ is:

$$\text{R}(\boldsymbol{\pi}, \boldsymbol{\sigma}) = \frac{\sum_{i=1}^{|T|} \sum_{j=1}^{|T|} z_{ij}}{|T|(|T| - 1)/2}$$

where $z_{ij} = \begin{cases} 1 & \text{if } \pi(i) < \pi(j) \text{ and } \sigma(i) > \sigma(j) \\ 0 & \text{otherwise} \end{cases}$

Then, we define the term **reordering correctness** R_{acc} by introducing the brevity penalty (BP):

$$\text{R}_{acc} = (1 - \sqrt{\text{R}(\boldsymbol{\pi}, \boldsymbol{\sigma})}) * BP$$

where $BP = e^{1-|Y|/|H|}$ if $|H| \leq |Y|$ else 1

Method	fisher_dev	fisher_dev2	fisher_test	CH_devtest	CH_evltest	Speed-Up
(A) *Autoregressive Models*						
(a) Transformer (*b*=10)	38.90	39.78	38.92	12.93	13.05	× 1.0
(b) Transformer + MTL (*b*=10)	46.38	47.54	46.28	**17.66**	17.17	× 1.0
(c) Transformer + MTL + ASR init. (*b*=10)	**48.27**	**49.17**	**48.40**	17.26	17.45	× 1.0
(d) Transformer + MTL + ASR init. (*b*=5)	48.18	48.91	48.21	17.34	**17.55**	× 2.0
(e) Transformer + MTL + ASR init. (*b*=1)	46.05	47.04	46.14	16.48	16.33	× 8.5
(B) *Non-Autoregressive Models (Ours)*						
(f) CTC	42.61	43.91	43.50	13.02	13.52	
(g) CTC + MTL at 4-th layer	42.26	43.70	43.58	13.10	13.17	
(h) CTC + MTL at 6-th layer	42.06	44.05	43.56	13.19	13.38	× 28.9
(i) CTC + MTL at 8-th layer	**44.45**	**45.23**	**44.92**	**14.20**	**14.19**	
(j) CTC + MTL at 10-th layer	42.86	44.18	43.59	13.65	13.28	

Table 1: BLEU on Fisher Spanish dataset and CALLHOME (CH) dataset, including autoregressive and non-autoregressive models. The abbreviation *b* stands for the beam size for beam search decoding. Multitask learning (MTL) represents using ASR as the auxiliary task trained with ST. In *Autoregressive Models*, the auxiliary loss always applied on the final encoder output in MTL, and we applied it on different layers in NAR models.

The higher the value, the more similar between two given alignments. Ideally, a well-trained model could handle the reordering problem by making σ close to π and result in $R_{acc} = 1$.

3 Experiments

3.1 Experimental Setup

We use the ESPnet toolkit (Watanabe et al., 2018) for experiments. We perform Spanish speech to English text translation with Fisher Spanish corpus. The test sets of CALLHOME corpus are also included for evaluation. The dataset details and download links are listed in Appendix B.

The NAR-ST model consists of two convolutional layers followed by twelve transformer encoder layers. Knowledge distillation (Kim and Rush, 2016) is also applied. More training details and parameter settings can be found in Appendix C.

3.2 Translation Quality and Speed

We use BLEU (Papineni et al., 2002) to evaluate the translation quality, as shown in Table 1. Beam-search decoding with beam-size *b* is considered for the AR models in this experiments. Greedy decoding is always used for the NAR models.

In the results of AR models (part (A)), multitask learning (MTL) can get better performance compared to the model without jointly training with an auxiliary task (row (b) v.s (a)). Further improvement can be brought by using a pre-trained ASR encoder as the initialization weight (row (c) v.s (b)). It shows that using ASR data for MTL and initialization are the essential steps to achieve exceptional performance. The performance drops when beam-size decreases, which shows a trade-off between the decoding speed and the performance (row (c) v.s (d)(e)).

To better optimize the decoding speed, NAR-ST provide a great solution to reach a shorter decoding time (part (B)). NAR-ST models is ×28.9 faster than the AR model with beam-size 10 (part (B) v.s rows (a)-(c)) and ×3.4 faster than the AR model with greedy decoding (part (B) v.s row (e)). We initialize the NAR-ST models with the weight pre-trained on ASR task and applied the proposed MTL approach on different intermediate layers (rows (g)-(j)). As the results showed in part (B), applying MTL on the higher layers improves the performance (rows (i)(j) v.s (f)). It shows that speech signal needs more layers to model the complexity, and sheds light on selecting the intermediate layer to apply MTL is essential. We also evaluate the ASR results of the MTL models in Appendix D.

Some text-based refinement approaches can further improve the translation quality (Libovický and Helcl, 2018; Lee et al., 2018; Ghazvininejad et al., 2019; Gu et al., 2019; Chan et al., 2020). We leave it as the future work and focus on analyzing the reordering capability of the CTC-based model.

3.3 Word Order Analysis

In this section, we discuss the word ordering problem in the translation task. We use R_{acc} defined in section 2.3 to systematically evaluate the reordering correctness across the corpora. Besides, we examine the gradient norm in models to visualize the reordering process.

Quantitative Analysis We use SimAlign (Sabet et al., 2020) to align the transcriptions and translations with details in Appendix E. Table 2 shows

	Method	R_{acc}		
		dev	dev2	test
(a)	random permutaion	41.37	41.84	42.42
Autoregress Models				
(b)	Tr. (b=10)	77.79	79.16	79.09
(c)	Tr. + MTL (b=10)	79.32	80.18	80.30
Non-Autoregress Models (Ours)				
(d)	CTC	71.69	74.00	74.57
(e)	CTC+MTL@8	71.91	74.35	74.96

Table 2: The reordering correctness evaluated on Fisher Spanish dataset. The values are selected as the best among 4 references. Tr. stands for Transformer. All models are initialized with pretrained ASR.

R_{acc} evaluated on ST models. We included the correctness of random permutation as a baseline.

The AR models obtain high R_{acc} scores (rows (b)(c)), it shows that the AR model can handle a complex word order. The NAR models also have the ability to rearrange words (rows (d)(e) v.s. (a)) but are weaker than AR models due to the independent assumption brought by CTC. An interesting observation is that applying MTL tends to improve R_{acc} (rows (c)(e) v.s. (b)(d)). We conclude that the monotonic natural in ASR improves the stability in training ST (Sperber et al., 2019).

To investigate the relation between model performance and the reordering difficulty, we measure the reordering difficulty by $R_{\boldsymbol{\pi}} = R(\boldsymbol{\pi}, \boldsymbol{m})$, where $\boldsymbol{m} = \langle 1, ..., |T| \rangle$ is a dummy monotonic alignment. We split all the testing data (dev/dev2/test) into smaller equal-size groups by different reference $R_{\boldsymbol{\pi}}$. The BLEU scores for these groups were plotted in Figure 2. Obviously, AR models are more robust to higher reordering difficulty. Nonetheless, we observed that when MTL is applied at layer 8, CTC model is more robust to reordering difficulty, in some cases ($R_{\boldsymbol{\pi}}<0.07$) even come close to the AR model without ASR pretraining.

Gradient-based Visualization We consider the gradient norm as an approximated indicator of reordering in our model. For each output token h_i, we concatenate the relative influence on it across all layers, which yields a matrix $\mathbf{O}^i \in \mathbb{R}^{|X| \times L}$, where each row is a frame and each column is a layer. We refer to this as the reordering matrix for token h_i. We leave the computational details in Appendix H.

Figure 3 shows a reordering matrix for token _thing. We can observe that the single-task CTC model (Figure 3, right) tends to keep focusing on
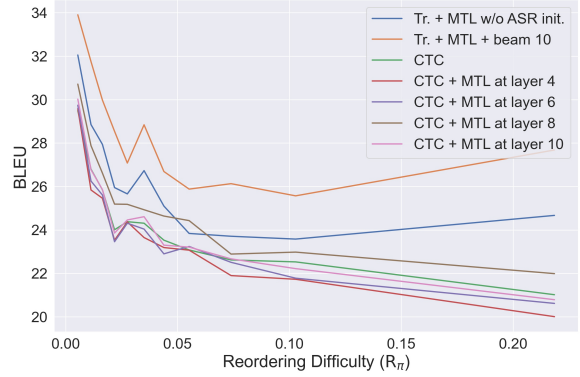

Figure 2: The BLEU score curve under different reordering difficulties (R_{π}). Details are in Appendix G.
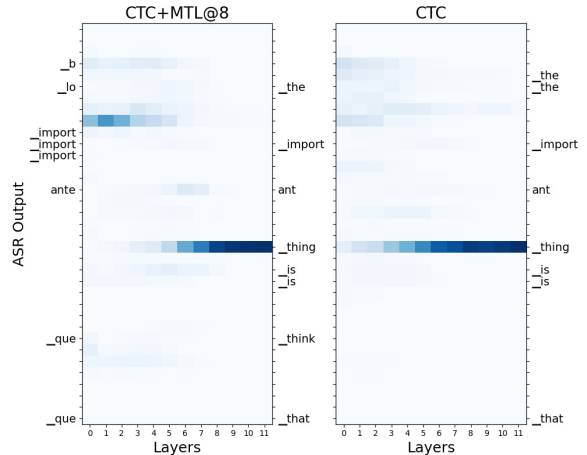

Figure 3: The re-ordering matrix of the token "thing". The ASR output from MTL model is included as a proxy for word positioning in source speech.

the same position of the output token _thing at the lower layers. In contrast, the model with MTL (Figure 3, left) does not focus on the same position at the lower layers. It can focus more on the other position where the corresponding speech segment appears, and aggregate the information to the output position at the higher layers. Additional examples can be found in Appendix I.

4 Concluding Remarks

We propose a CTC-based NAR-ST model with an auxiliary CTC-based ASR task and are the first to study the reordering capability in CTC-based NAR-ST model. R_{acc} is adopted to analyze reordering in the ST task, and gradient-based visualizations reveal the internal manipulation of the models. Besides trying to improve BLEU scores, we encourage future research on NAR models to also evaluate whether the NAR models have inferior reordering capabilities in order to close the gap between AR and NAR models.

Broader Impact and Ethical Considerations

We believe that our work can help researchers in the NLP community understand more about the non-autoregressive speech translation models, and we envision that the model proposed in this paper will equip the researchers with a new technique to perform better and faster speech translation. We do not see ourselves violating the code of ethics of ACL-IJCNLP 2021.

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6224–6228. IEEE.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *arXiv preprint arXiv:1612.01744*.

Alexandra Birch and Miles Osborne. 2011. Reordering metrics for mt. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1027–1035.

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. 2020. Imputer: Sequence modelling via imputation and dynamic programming. *arXiv preprint arXiv:2002.08926*.

Shun-Po Chuang, Tzu-Wei Sung, Alexander H. Liu, and Hung-yi Lee. 2020. Worse WER, but better BLEU? leveraging word embedding as intermediate in multitask end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5998–6003, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Mattia A Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting transformer to end-to-end spoken language translation. *Proc. Interspeech 2019*, pages 1133–1137.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6114–6123.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, pages 11181–11191.

Hirofumi Inaguma, Yosuke Higuchi, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2020. Orthros: Non-autoregressive end-to-end speech translation with dual-decoder. *arXiv preprint arXiv:2010.13047*.

Takatomo Kano, Sakriani Sakti, and Satoshi Nakamura. 2021. Transformer-based direct speech-to-speech translation with transcoder. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 958–965.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Proc. Interspeech 2019*, pages 2613–2617.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2019. Guiding non-autoregressive neural machine translation decoding with reordering information. *arXiv preprint arXiv:1911.02215*.

Masoud Jalili Sabet, Philipp Dufter, and Hinrich Schütze. 2020. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. *arXiv preprint arXiv:2004.08728*.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. *arXiv preprint arXiv:2004.07437*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-Passing Models for Robust and Data-Efficient End-to-End Speech Translation. *Transactions of the Association for Computational Linguistics (TACL)*.

Matthias Sperber and Matthias Paulik. 2020. Speech translation and the end-to-end promise: Taking stock of where we are. *arXiv preprint arXiv:2004.06358*.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985.

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*, pages 5998–6008.

Laura-Cross Vila, Carlos Escolano, José AR Fonollosa, and Marta-R Costa-Jussà. 2018. End-to-end speech translation with the transformer. *Proc. IberSPEECH 2018*, pages 60–63.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson-Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al. 2018. Espnet: End-to-end speech processing toolkit. *Proc. Interspeech 2018*, pages 2207–2211.

Ron J Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. *arXiv preprint arXiv:1703.08581*.

Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*.

A Source Code

Please download our code at `https://github.com/voidism/NAR-ST` and follow the instructions written in `README.md` to reproduce the results.

B Dataset Details

We use the Fisher and CALLHOME Spanish dataset (a Spanish-to-English speech-to-text translation dataset), which can be downloaded in the following links: 1) Fisher Spanish Speech `https://catalog.ldc.upenn.edu/LDC2010S01` 2) CALLHOME Spanish Speech `https://catalog.ldc.upenn.edu/LDC96S35` 3) Fisher and CALLHOME Spanish–English Speech Translation `https://catalog.ldc.upenn.edu/LDC2014T23`.

B.1 Statistics

The data statistics are listed in Table 3.

Data Split	# of Utterance	Duration (hours)
Training sets		
fisher_train	138792	171.61
Validation sets		
fisher_dev	3973	4.59
fisher_dev2	3957	4.70
callhome_devtest	3956	3.82
Testing sets		
fisher_test	3638	4.48
callhome_evltest	1825	1.83

Table 3: The data statistics of the Fisher and CALLHOME Spanish dataset.

B.2 Preprocessing

We use ESPnet to preprocess our data. For text, we use Byte Pair Encoding (BPE) (Sennrich et al., 2016) with vocabulary size 8,000. We convert all text to lowercase with punctuation removal. For audio, we convert all audio files into wav file with a sample frequency of 16,000. We extract 80-dim fbank without delta. We use SpecAugment (Park et al., 2019) to augment our data. More details can be found in our source code in Appendix A.

C Training Details

C.1 Computing Infrastructure and Runtime

We use a single NVIDIA TITAN RTX (24G) for each experiment. The average runtime of experiments in Table 1 is 2-3 days for both autoregressive and non-autoregressive models.

C.2 Hyperparameters

Our training hyperparameters are listed in Table 4. We do not conduct hyperparameter search, but follow the autoregressive ST best setting in ESPnet toolkit (Watanabe et al., 2018), and use the same hyperparameter for our non-autoregressive models. Due to the limited budget, we run each experiment for once. For inference stage of CTC-based models, we simply use greedy decode to produce the output sequences.

Hyperparameter	Value
encoder layers	12
hidden units	2048
attention dimension	256
attention heads	4
label smoothing weight	0.1
batch size	64
optimizer	noam
learning rate	2.5
warmup steps	25000
attention dropout rate	0.0
gradient accumulate step	2
gradient clipping	5.0
epoch	30
dropout rate	0.1

Table 4: The main hyperparameters in the experiment.

C.3 Knowledge Distillation

To perform sequence-level knowledge distillation (Seq-KD) (Kim and Rush, 2016) to improve the performance of NAR models, we firstly trained an autoregressive transformer-based MT model on the transcriptions and translations in same training set with ESPnet. Then we used the trained model to produce the hypotheses with beam search size of 1 for the whole training set. We swapped the ground truth sequences with the hypotheses for all NAR ST model training. We also show the ablation results on knowledge distillation in Table 5.

We also try the possibility of using the autoregressive ST model to produce the hypotheses for Seq-KD, but the results are not as good as using a MT model. The results are shown in the second row in Table 5. The download links to MT/ST decode results for conducting Seq-KD are also provided in the `README.md` file in our source code (See Appendix A).

Method	Fisher			CALLHOME	
	dev	dev2	test	devtest	evltest
NAR w/ MT Seq-KD	**44.45**	**45.23**	**44.92**	**14.20**	**14.19**
NAR w/ ST Seq-KD	41.08	41.28	40.85	13.63	13.27
NAR w/o Seq-KD	41.59	43.13	42.25	13.05	12.81

Table 5: Ablation test on sequence-level knowledge distillation (Seq-KD) on Fisher Spanish dataset and CALLHOME (CH) dataset. The NAR model we used is CTC+MTL at 8-th layer.

C.4 Model Selection

When evaluating the models, we average the model checkpoints of the final 10 epochs to obtain our final model for NAR experiments. For AR experiments, we follow the original setting in ESPnet to average the 5 best-validated model checkpoints to obtain the final model.

C.5 Model Size

The number of parameters of our CTC model is 18.2M. The number of parameters of the autoregressive model is 27.9M.

D Automatic Speech Recognition Evaluation

Method	Fisher		
	dev	dev2	test
CTC+MTL@4	47.84	46.79	45.97
CTC+MTL@6	40.21	39.02	38.08
CTC+MTL@8	32.87	31.99	30.88
CTC+MTL@10	**29.31**	**28.36**	**27.10**

Table 6: The Word Error Rate (WER) on Fisher Spanish dataset of CTC results of the intermediate layers in multitask CTC models.

We compute the Word Error Rate (WER) for ASR output obtained from the intermediate ASR branch of our proposed models. The results are shown in Table 6. We can observe that when applying multitask learning in the higher layers, the WER becomes lower. It indicates that ASR need more layers to perform better. However, the best ST scores are achieved by CTC+MTL@8 instead of CTC+MTL@10. It may be caused by the fact that there are only two transformer encoder layers for CTC+MTL@10 to perform ST. It may be too difficult for the model to convert the information from source language to target language in two encoder layers, even though the lower WER indicates useful information is provided to perform ST.

E SimAlign setup

We use the released code[2] by the authors of SimAlign as the external aligner to obtain word alignments used for calculating reordering metrics. SimAlign uses contextualized embeddings from pretrained language models, and there are several proposed algorithms to do word alignments. We use XLM-R (Conneau et al., 2019) as the underlying contextualized word embeddings with the itermax matching algorithm.

F Reordering Difficulty

We provide reordering difficulty measured on all en-xx language pairs in CoVoST2 dataset in Table 7.

en-xx	R_π	en-xx	R_π
de	5.92	tr	15.80
zh-CN	10.63	fa	13.93
ja	20.86	sv-SE	2.94
ar	6.04	mn	16.28
et	5.98	cy	5.78
ca	4.39	id	3.41
sl	5.13	ta	15.57
lv	5.34		

Table 7: The training set reordering difficulty evaluated on each pairs of languages in the CoVoST2 dataset.

G Details on Figure 2

In Figure 2, the primary goal is to view the relation between reordering difficulty and the model's performance. We describe the method used in (Birch and Osborne, 2011) to represent the reordering difficulty as follows: For each example in the fisher dev and test set, calculate Kendall's tau distance between 1) its reference alignment (alignment between source transcription and reference translation) and 2) a dummy monotonic alignment, which is just the sequence 1...m. Intuitively this shows how much the reference alignment disagrees with monotonic alignment, and hence the reordering difficulty. Next, we divide all examples into 10 bins, where each bin contains examples with similar reordering difficulty, and all bins have an equal number of examples. Finally, we calculate the BLEU score of the hypotheses in each bin. The result is plotted in Figure 2.

---

[2]https://github.com/cisnlp/simalign

H Gradient-based Visualization

We first obtain a saliency matrix $\mathbf{J}^M \in \mathbb{R}^{|X| \times |X|}$ for the M-th transformer layer by computing the gradient norm of output logits w.r.t. the latent representations of each timestep in that layer. An example is shown in Figure 4. Then, we normalize $\mathbf{J}^M$ across the dimension corresponding to the source audio sequence. Intuitively, the i-th column of $\mathbf{J}^M$ can be interpreted as the relative influence of the representations at each position on the i-th output token.

Consequently, we proceed to re-arrange $\mathbf{J}^M$ in the following way: for each output token h_i, we concatenate the relative influence on it across all layers, which yields the reordering matrix for token h_i, denoted as $\mathbf{O}^i \in \mathbb{R}^{|X| \times L}$.
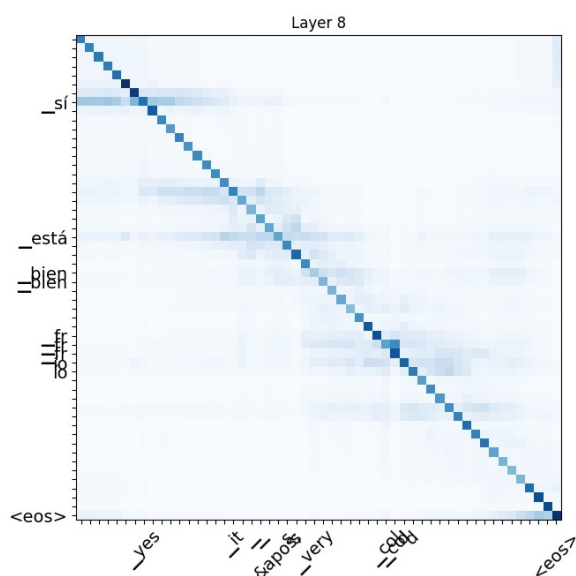

Figure 5: The re-ordering matrix of the token "_we".

Figure 4: The saliency matrix of layer 8 input.

I Reordering Matrix

We provide some additional examples of visible reordering in our CTC-based models. In Figure 5, "_we" is heavily influenced by the position of audio signal "amos", even though the ASR output is incorrectly predicted as "as". In Figure 6, "_you" also influenced by audio signal "usted". It is interesting to observe that in some cases the pure CTC-based model appears more capable of reordering, while in others it does not.

J Higher Reordering Difficulty

We address instances of higher difficulty by analyzing Figure 7. In the figure, the horizontal axis
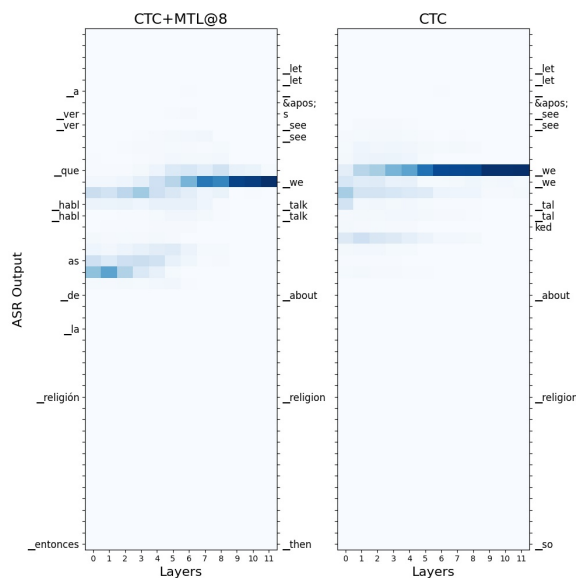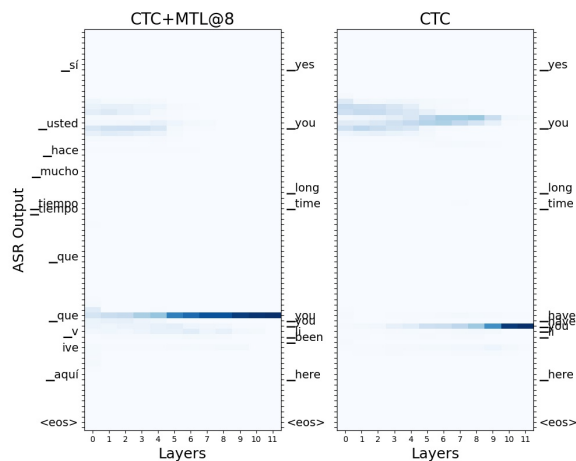

Figure 6: The re-ordering matrix of the token "_you".

corresponds to the reordering difficulty, and the vertical axis corresponds to the reordering correctness. In this figure, there is a very consistent decrease in reordering correctness when reordering difficulty increases, and the rate of decrease is very similar between NAR and AR models. This observation reveals that when evaluated on distant language pairs, that the reordering difficulty is large, the gap between NAR and AR will probably remain roughly the same. We will conduct experiments on different language pairs to verify the above claim in future work.
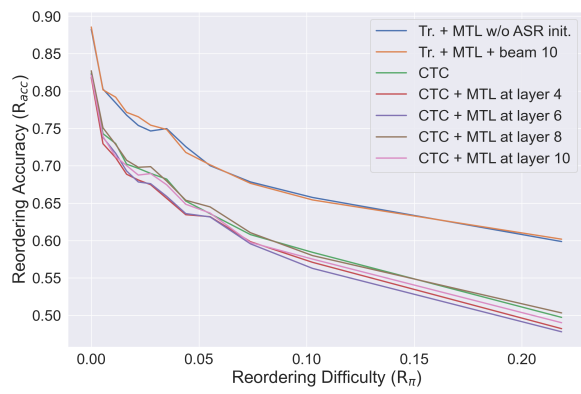
1076

Figure 7: The reordering accuracy (R_{acc}) curve under different reordering difficulties (R_π).