ESPnet-ST IWSLT 2021 Offline Speech Translation System

Hirofumi Inaguma[1*] **Brian Yan**[2*] **Siddharth Dalmia**[2] **Pengcheng Guo**[3]
Jiatong Shi[4] **Kevin Duh**[4] **Shinji Watanabe**[2,4]
[1]Kyoto University, Japan   [2]Carnegie Mellon University, USA
[3]Northwestern Polytechnical University, China   [4]Johns Hopkins University, USA
inaguma@sap.ist.i.kyoto-u.ac.jp
byan@cs.cmu.edu

Abstract

This paper describes the ESPnet-ST group's IWSLT 2021 submission in the offline speech translation track. This year we made various efforts on training data, architecture, and audio segmentation. On the data side, we investigated sequence-level knowledge distillation (SeqKD) for end-to-end (E2E) speech translation. Specifically, we used multi-referenced SeqKD from multiple teachers trained on different amounts of bitext. On the architecture side, we adopted the Conformer encoder and the Multi-Decoder architecture, which equips dedicated decoders for speech recognition and translation tasks in a unified encoder-decoder model and enables search in both source and target language spaces during inference. We also significantly improved audio segmentation by using the pyannote.audio toolkit and merging multiple short segments for long context modeling. Experimental evaluations showed that each of them contributed to large improvements in translation performance. Our best E2E system combined all the above techniques with model ensembling and achieved 31.4 BLEU on the 2-ref of tst2021 and 21.2 BLEU and 19.3 BLEU on the two single references of tst2021.

1 Introduction

This paper presents the ESPnet-ST group's English→German speech translation (ST) system submitted to the IWSLT 2021 offline speech translation track. ESPnet (Watanabe et al., 2018) has been widely used for many speech applications; automatic speech recognition (ASR), text-to-speech (Hayashi et al., 2020), speech translation (Inaguma et al., 2020), machine translation (MT), and speech separation/enhancement (Li et al., 2021). The purpose of this submission is not only to show the recent progress on ST researches, but also to encourage future research by building strong systems along with the open-sourced project.

This year we focused on (1) sequence-level knowledge distillation (SeqKD) (Kim and Rush, 2016), (2) Conformer encoder (Gulati et al., 2020), (3) Multi-Decoder architecture (Dalmia et al., 2021), (4) model ensembling, and (5) better segmentation with a neural network-based voice activity (VAD) system (Bredin et al., 2020) and a novel algorithm to merge multiple short segments for long context modeling. Our primary focus was E2E models, although we also compared them with cascade systems with our best effort. All experiments were conducted with the ESPnet-ST toolkit (Inaguma et al., 2020), and the recipe is publicly available at https://github.com/espnet/espnet/tree/master/egs/iwslt21.

2 Data preparation

In this section, we describe data preparation for each task. The corpus statistics are listed in Table 1. We removed the off-limit talks following previous evaluation campaigns[1]. To fit the GPU memory, we excluded utterances having more than 3000 speech frames or more than 400 characters. All sentences were tokenized with the tokenizer.perl script in the Moses toolkit (Koehn et al., 2007).

2.1 ASR

We used Must-C (Di Gangi et al., 2019), Must-C v2[2], ST-TED (Jan et al., 2018), Librispeech (Panayotov et al., 2015), and TEDLIUM2 (Rousseau et al., 2012) corpora. We used the cleaned version of ST-TED following (Inaguma et al., 2019). The speech

---

*Equal contribution

[1]https://sites.google.com/view/iwslt-evaluation-2019/speech-translation/off-limit-ted-talks
[2]https://ict.fbk.eu/must-c-release-v2-0/

	#Hour	#Sentence
ASR		
Must-C	408 × 3	0.68M
Must-C v2	458 × 3	0.74M
ST-TED (cleaned)	200 × 3	0.40M
Librispeech	960	0.28M
TEDLIUM2	210 × 3	0.27M
E2E-ST		
Must-C	408 × 3	0.68M
Must-C v2	458 × 3	0.74M
ST-TED (cleaned)	200 × 3	0.40M
MT		
Must-C		0.68M
Must-C v2		0.74M
ST-TED (cleaned)		0.40M
Europarl		1.82M
Commoncrawl	-	2.39M
Paracrawl		34.37M
NewsCommentary		0.37M
WikiTitles		1.38M
RAPID		1.63M
WikiMatrix		1.57M

Table 1: Corpus statistics

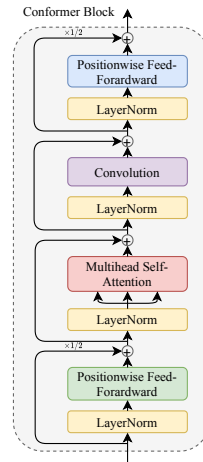| Filtering method	#Sentence		
	WMT5M	WMT10M	WMT20M
In-domain LM	5.00M	10.00M	20.00M
+ `langid`	3.42M	7.90M	15.33M
+ length/character	3.15M	7.77M	15.01M

Table 2: MT bitext filtering

Figure 1: Block diagram of Conformer architecture

data was augmented by three-fold speed pertur-bation (Ko et al., 2015) with speed ratios of 0.9, 1.0, and 1.1 except for Librispeech. We removed case information and punctuation marks except for apostrophes from the transcripts. The 5k unit vo-cabulary was constructed based on the byte pair encoding (BPE) algorithm (Sennrich et al., 2016) with the `sentencepiece` toolkit[3] using the En-glish transcripts only.

2.2 E2E-ST

We used Must-C, Must-C v2, and ST-TED only. The shared source and target vocabulary of BPE16k units was constructed using cased and punctuated transcripts and translations.

2.3 MT

We used available bitext for WMT20[4] in addition to the in-domain TED data used for E2E-ST systems. We first performed perplexity-based filtering with an in-domain n-gram language model (LM) (Moore and Lewis, 2010). We controlled the WMT data size by thresholding and obtained three data pools: 5M, 10M, and 20M sentences. Next, we removed non-printing characters and performed language identification with the `langid.py` toolkit (Lui and Baldwin, 2012)[5] and kept sentences whose lan-

guage IDs were identified correctly on both English and German sides. We also removed sentences hav-ing more than 250 tokens in either language or a source-target length ratio of more than 1.5 with the `clean-corpus-n.perl` script in Moses. Fi-nally, we removed sentences having CJK and other unrelated characters in either language with the built-in `regex` module in Python. The resulting data size is shown in Table 2. We found that our filtering strategy removed 22-37% of data. Note that the above filtering process was performed over the WMT data only. For each data size, the joint source and target vocabulary of BPE32k units was constructed using cased and punctuated sentences after the filtering. We did not use additional mono-lingual data.

3 System

3.1 Conformer encoder

Conformer encoder (Gulati et al., 2020) is a stacked multi-block architecture and has shown consistent improvement over a wide range of E2E speech processing applications (Guo et al., 2020). The architecture of each block is depicted in Figure 1. It includes a multi-head self-attention module, a convolution module, and a pair of position-wise feed-forward modules in the Macaron-Net style. While the self-attention module learns the long-

---

[3] `https://github.com/google/sentencepiece`

[4] Europarl, Commoncrawl, Paracrawl, NewsCommentary, WikiTitles, RAPID and WikiMatrix

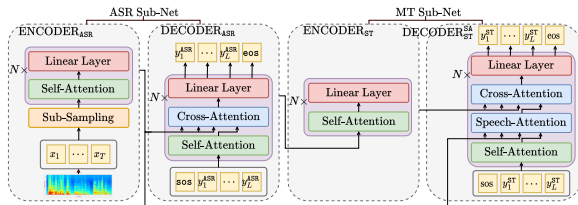[5] `https://github.com/saffsd/langid.py`

Figure 2: The Multi-Decoder (MD) architecture decomposes the overall ST task with ASR and MT sub-nets while maintaining E2E differentiability.

range global context, the convolution module aims to model the local feature patterns synchronously. Recent studies have shown improvements by introducing Conformer in the E2E-ST task (Guo et al., 2020; Inaguma et al., 2021b), which motivated us to adopt this architecture as our system.

3.2 SeqKD

Sequence-level knowledge distillation (SeqKD) (Kim and Rush, 2016) is an effective method to transfer knowledge in a teacher model to a student model via discrete symbols. Our recent studies (Inaguma et al., 2021a,b) showed a large improvement in ST performance with this technique. Unlike the previous studies, however, we used more training data than bitext in ST training data to train teacher MT models. We translated source transcripts in the ST training data by the teacher MT models with a beam width of 5 and then replaced the original ground-truth translation with the generated translation. We used cased and punctuated transcripts as inputs to the MT teachers. We also combined both the original and pseudo translations as data augmentation (*multi-referenced training*) (Gordon and Duh, 2019).

3.3 Multi-Decoder architecture

The Multi-Decoder is an E2E-ST model using Searchable Hidden Intermediates to decompose the overall ST task into ASR and MT sub-tasks (Dalmia et al., 2021). As shown in Figure 2, the Multi-Decoder consists of two encoder-decoder models, an ASR sub-net and a subsequent MT sub-net, where the hidden representations of the ASR decoder are passed as inputs to the encoder of the MT sub-net. During inference, the best ASR decoder hidden representations are retrieved using beam search decoding at this intermediate stage.

Since this framework decomposes the overall ST task, it brings several advantages of cascaded

approaches into the E2E setting. For instance, the Multi-Decoder allows for greater search capabilities and separation of speech and text encoding. However, one trade-off is a greater risk of error propagation from the ASR sub-net to the downstream MT sub-net. To alleviate this issue, we condition the decoder of the MT sub-net on the ASR encoder hidden representations in addition to the MT encoder hidden representations using multi-source cross-attention. This improved variant of the architecture is called the Multi-Decoder with Speech Attention.

3.4 Model ensembling

We use posterior probability combination to ensemble models trained with different data and architectures. During inference, we perform a posterior combination at each step of beam search decoding by first computing the softmax normalized posterior probabilities for each model in the ensemble and then taking the mean value. In this ensembling approach, a single unified beam search operates over the combined posteriors of the models to find the most likely decoded sequence.

3.5 Segmentation

How to segment audio during inference significantly impacts ST performances (Gaido et al., 2020; Pham et al., 2020; Potapczyk and Przybysz, 2020; Gaido et al., 2021). This is because the ST systems are usually trained with utterances segmented based on punctuation marks (Di Gangi et al., 2019) while the audio segmentation by voice activity detection (VAD) at test time does not access such meta information. Since VAD splits a long speech recording into chunks by silence regions, it would prevent models from extracting semantically coherent contextual information. Therefore, it is very important to seek a better segmentation strategy in order to minimize this gap in training and test conditions and evaluate models correctly. In fact, the last year's winner obtained huge improvements by using their own segmentation strategy.

Motivated by this fact, we investigated two VAD systems apart from the provided segmentation. Specifically, we used WebRTC[6] and pyannote.audio (Bredin et al., 2020)[7] toolkits. For We-

---

[6] https://github.com/wiseman/py-webrtcvad

[7] https://github.com/pyannote/pyannote-audio

102

Algorithm 1 Merge short segments after VAD for long context modeling

---

1: **function** MERGESEGMENT($\boldsymbol{x}$, M_{dur}, M_{int})
2:     $Q \leftarrow VAD(\boldsymbol{x})$ ▷ $\{(s_1, e_1), \cdots, (s_M, e_M)\}$
3:     **while** True **do**
4:         $N_{\text{merge}} \leftarrow 0$
5:         $Q_{\text{next}} \leftarrow \{\}$ ▷ Queue
6:         $S, T \leftarrow s_1, e_1$ ▷ Start/End time
7:         **for** $(s_m, e_m) \in Q$ **do**
8:             **if** $e_m - S < M_{\text{dur}}$ and $s_m - E < M_{\text{int}}$ **then**
9:                 $N_{\text{merge}} \leftarrow N_{\text{merge}} + 1$ ▷ Merge segments
10:            **else**
11:                $Q_{\text{next}}.enqueue((S, E))$
12:                $S \leftarrow s_m$ ▷ Reset
13:            **end if**
14:            $E \leftarrow e_m$
15:        **end for**
16:        $Q \leftarrow Q_{\text{next}}$
17:        **if** $N_{\text{merge}} = 0$ **then**
18:            **break**
19:        **end if**
20:    **end while**
21:    **return** Q
22: **end function**

---

bRTC, we set the frame duration, padding duration, and aggressive mode to 10ms, 150ms, and 3, respectively. For pyannote.audio, we used a publicly available model pre-trained on the DIHARD corpus (Ryant et al., 2019).

However, we observed that VAD systems are more likely to generate short segments because they do not take contextual information into account. Therefore, we propose a novel algorithm to merge multiple short segments into a single chunk to enable long context modeling by self-attention in both encoder and decoder modules. The proposed algorithm is shown in Algorithm 1. We first perform VAD and obtain multiple segments. Then, we check the segments in a greedy way from left to right and merge adjacent segments if (1) the total utterance duration is below a threshold M_{dur} [10ms] and (2) the time interval of the two segments is below a threshold M_{int} [10ms]. This process continues until no segment is merged in an iteration. Although recent studies proposed similar methods (Potapczyk and Przybysz, 2020; Gaido et al., 2021), our algorithm is a bottom-up approach while theirs are top-down.

4 Experimental setting

In this section, we describe the experimental setting for each task. The detailed configurations for each task are summarized in Table 3.

Configuration	ASR	E2E-ST		MT
		non-MD	MD	
Warmup step	25k	25k	25k	8k
Learning rate factor	10.0	2.5	12.5	1.0
Batch size	200 utt	128 utt	120 utt	65k tok
Epoch	30	30	30	40
Validation metric	Accuracy	BLEU	BLEU	BLEU
Model average	5	5	5	5
Beam width	10	4	16, 10	4

Table 3: Summary of training configuration

4.1 Feature extraction

We extracted 80-channel log-mel filterbank coefficients computed with 25-ms window size and shifted every 10-ms with 3-dimensional pitch features using the Kaldi toolkit (Povey et al., 2011). The features were normalized by the mean and the standard deviation calculated on the entire training set. We applied SpecAugment (Park et al., 2019) with mask parameters $(m_T, m_F, T, F) = (2, 2, 40, 30)$ and time-warping for both ASR and E2E-ST tasks.

4.2 ASR

We used both Transformer and Conformer architectures. The encoder had two CNN blocks followed by 12 Transformer/Conformer blocks following (Karita et al., 2019; Guo et al., 2020). Each CNN block consisted of a channel size of 256 and a kernel size of 3 with a stride of 2×2, which resulted in time reduction by a factor of 4. Both architectures had six Transformer blocks in the decoder. In both encoder and decoder blocks, the dimensions of the self-attention layer d_{model} and feed-forward network d_{ff} were set to 512 and 2048, respectively. The number of attention heads H was set to 8. The kernel size of depthwise separable convolution in Conformer blocks was set to 31. We optimized the model with the joint CTC/attention objective (Watanabe et al., 2017) with a CTC weight of 0.3. We also used CTC scores during decoding but did not use any external LM for simplicity. We adopted the best model configuration from the Librispeech ASR recipe in ESPnet.

4.3 MT

We used the Transformer-Base and -Big configurations in (Vaswani et al., 2017).

4.4 E2E-ST

We used the same Conformer architecture as ASR except for the vocabulary. We initialized the en-

Model	WER ($\downarrow$)		
	Librispeech `test-other`	TEDLIUM2 `test`	Must-C `tst-COMMON`
Transformer	9.4	6.4	7.0
Conformer	**7.1**	**6.2**	**5.6**

Table 4: Word error rate (WER) of ASR systems

VAD	M_{dur}	M_{int}	WER ($\downarrow$)				
			tst2010	tst2015	tst2018	tst2019	Avg.
Provided	–	–	18.2	32.1	23.5	20.8	23.65
	1500	200	14.4	29.3	18.4	15.5	19.40
	2000	200	12.7	27.7	16.4	11.5	17.08
	2500	200	14.5	29.9	15.1	12.2	17.93
WebRTC	–	–	35.3	35.1	44.0	22.7	34.28
	1500	200	19.4	26.7	27.7	13.8	21.90
	2000	200	19.8	27.7	27.1	11.9	21.63
	2500	200	22.9	29.5	27.1	11.6	22.78
pyannote	–	–	9.5	24.0	15.5	7.3	14.08
	1500	200	8.0	23.0	12.4	7.3	12.68
	1500	100	7.5	22.2	12.4	6.5	12.15
	2000	200	10.3	22.5	12.2	6.5	12.88
	2000	150	9.6	21.8	12.3	6.1	12.45
	2000	100	8.1	**21.5**	**12.0**	5.8	11.90
	2000	50	**7.3**	21.9	12.4	**5.9**	**11.88**

Table 5: Impact of audio segmentation for ASR

coder parameters with those of the Conformer ASR. On the decoder side, we initialized parameters like BERT (Devlin et al., 2019), where weight parameters were sampled from $\mathcal{N}(0, 0.02)$, biases were set to zero, and layer normalization parameters were set to $\beta = 0$, $\gamma = 1$. This technique led to better translation performance and faster convergence.

5 Results

5.1 ASR

5.1.1 Architecture

We compared Transformer and Conformer ASR architectures in Table 4. We observed that Conformer significantly outperformed Transformer. Therefore, we use the Conformer encoder in the following experiments.

5.1.2 Segmentation

Next, we investigated the VAD systems and the proposed segment merging algorithm for long context modeling in Table 5. We used the same decoding hyperparameters tuned on Must-C. We firstly observed that merging short segments was very effective probably because it alleviated frame classification errors in the VAD systems. Among three audio segmentation methods, we confirmed that pyannote.audio significantly reduced the WER while WebRTC had negative impacts compared to the provided segmentation. Specifically, we found that

the `dihard` option in pyannote.audio worked very well while the rest options did not. The optimal maximum duration M_{dur} was around 2000 frames (i.e., 20 seconds). In the last experiments, we tuned the maximum interval M_{int} among $\{50, 100, 150, 200\}$ and found 50 and 100 (i.e., 0.5 and 1 second) was best on average. Compared to the provided segmentation, we obtained a 49.6% improvement on average.

5.2 MT

In this section, we show the results of our MT systems used for cascade systems and pseudo labeling in SeqKD. We report case-sensitive detokenized BLEU scores (Papineni et al., 2002) with the `multi-bleu-detok.perl` script in Moses. We carefully investigated the effective amount of WMT training data to improve the performance of the TED domain. The results are shown in Table 6. We confirmed that adding the WMT data improved the performance by more than 4 BLEU. Regarding the WMT data size, using up to 10M sentences was helpful, but 20M did not show clear improvements, probably because of the undersampling of the TED data. Oversampling as in multilingual NMT (Arivazhagan et al., 2019) could alleviate this problem, but this is beyond our scope.

After training with a mix of the WMT and TED data, we also tried to finetune the model with the TED data only, but this did not lead to clear improvement, especially for the IWSLT test sets. Increasing the model capacity was not helpful, although the conclusion might change by adding more training data and evaluating the model in other domains such as news. Because our primary focus to use MT systems was pseudo labeling for SeqKD, we decided to use the Base configuration to speed up decoding.

Finally, we checked the BLEU scores on the Must-C training data used for SeqKD. We observed that adding more WMT data decreased the BLEU score, from which we can conclude that using more WMT data gradually changed the MT output from the TED style. Therefore, we decided to use the models trained on `WMT5M` and `WMT10M` as teachers for SeqKD.

5.3 Speech translation

5.3.1 E2E-ST

SeqKD The results are shown in Table 7. We first observed the baseline Conformer model

| Model | BLEU (↑) | | | | | | | |
| | Must-C | | Must-C v2 | tst2010 | tst2015 | tst2018 | tst2019 | Must-C |
	dev	tst-COMMON	tst-COMMON					Train
Base (Must-C only)	–	30.02	29.86	27.28	24.92	21.13	20.37	
Base (WMT5M)	31.31	34.13	33.85	31.61	32.44	28.30	28.28	45.68
+ Big	27.32	29.11	28.85	27.61	28.44	24.42	23.92	–
Base (WMT10M)	**33.28**	**35.09**	34.80	**33.58**	33.26	29.24	28.87	38.31
+ In-domain finetune	30.67	**35.50**	**35.30**	30.79	31.43	25.35	26.10	–
Base (WMT20M)	33.15	35.06	**34.87**	33.26	**33.56**	**29.94**	**29.08**	33.60

Table 6: BLEU scores of text-based MT systems

| ID | Model | BLEU (↑) | | | | | | | |
| | | Must-C | | | Must-C v2 | tst2010 | tst2015 | tst2018 | tst2019 |
		dev	tst-COMMON	tst-HE	tst-COMMON				
-	Bidir SeqKD (E2E) (Inaguma et al., 2021b)	25.67	27.01	25.36	–	–	–	–	–
	Multi-Decoder (E2E) (Dalmia et al., 2021)	–	26.4	–	–	–	–	–	–
	RWTH (Cascade) (Bahar et al., 2021)	–	26.50	26.80	–	–	28.4	–	–
	KIT (E2E) (Pham et al., 2020)	–	30.60	–	–	24.27	21.82	–	–
	KIT (Cascade) (Pham et al., 2020)	–	–	–	–	26.68	24.95	–	–
	SRPOL (E2E) (Potapczyk and Przybysz, 2020)	–	–	–	–	29.44	24.6	–	23.96
A1	Baseline (X)	25.14	**35.63**	22.63	**36.07**	21.40	18.18	16.69	17.39
A2	+ SeqKD (Y)	26.31	29.29	26.33	29.50	23.34	21.24	21.09	22.25
A3	+ 2ref SeqKD (X+Y)	26.50	30.59	26.21	30.92	23.00	22.18	20.38	21.59
A4	+ 3ref SeqKD (X+Y+Z)	**27.66**	30.90	**27.44**	31.07	**24.97**	**22.66**	**22.20**	**23.41**
B1	MD + 2ref SeqKD	–	30.78	–	–	–	–	–	23.78
C1	Conformer ASR → Base MT (WMT10M)	27.01	29.42	26.13	29.75	**25.04**	**23.17**	**23.05**	23.19

Table 7: BLEU scores of ST systems. X: original, Y: WMT5M, Z: WMT10M. For unsegmented test sets, we used pyannote.audio with $M_{\mathrm{dur}} = 2000$ and $M_{\mathrm{int}} = 100$.

(A1) achieved 35.63 BLEU on the Must-C tst-COMMON set, and it is the new state-of-the-art record to the best of our knowledge. Surprisingly, it even outperformed text-based MT systems in Table 6. On the other hand, unlike our observations in (Inaguma et al., 2021a,b), SeqKD (A2-4) degraded the performance on the Must-C tst-COMMON set. However, the results on the Must-C dev and tst-HE sets showed completely different trends, where we observed better BLEU scores by SeqKD in proportion to the WMT data used for training the teachers. Therefore, after tuning audio segmentation, we also evaluated the models on the unsegmented IWSLT test sets. Here, we used the pyannote.audio based segmentation with $(M_{\mathrm{dur}}, M_{\mathrm{int}}) = (2000, 100)$ as described in §5.1.2. Then, we confirmed large improvements with SeqKD by 2-6 BLEU, and therefore we decided to determine the best model based on the IWSLT test sets. Multi-referenced training consistently improved the BLEU scores on the IWSLT sets. For example, A4 outperformed A1 by 6.02 BLEU on tst2019 although the tst2019 set was well-segmented (WER: 6.0%). Given these observations, we recommend evaluating ST models on

ID	Ensembled Models	tst2019
-	B1	21.06
E1	B1, A4	22.51
E2	B1, A4, A1	22.83
E3	B1, A4, A1, A3	23.36
E4	B1, A4, A1, A3, A2	**23.61**

Table 8: BLEU (↑) scores of ensembled E2E-ST systems on tst2019, using the provided segmentation with $M_{\mathrm{dur}} = 2000$ and $M_{\mathrm{int}} = 100$

multiple test sets for future research.

Multi-Decoder architecture We combined the SeqKD and Multi-Decoder techniques in our B1 system. B1, which used a conformer ASR encoder and 2ref SeqKD, showed an improvement of 2.19 BLEU on tst2019 over A3, the encoder-decoder which also used 2ref SeqKD. B1 also achieved a slightly higher result on tst2019 compared to A4 which used 3ref SeqKD. These results suggest that the Multi-Decoder architecture is indeed compatible with SeqKD.

Model ensemble As shown in Table 8, ensembling our various ST systems using the posterior combination method described in §3.4 showed im-

VAD	M_{dur}	M_{int}	BLEU (↑)				
			tst2010	tst2015	tst2018	tst2019	Avg.
Provided†	–	–	–	–	–	20.1	–
Provided (E2E)	–	–	21.99	19.94	19.29	19.70	20.23
	1000	200	22.62	20.54	19.80	20.54	20.88
	1500	200	23.00	21.66	20.14	21.50	21.58
	2000	200	22.95	21.58	20.03	21.34	21.48
WebRTC (E2E)	–	–	13.13	12.97	11.07	13.32	12.62
	1000	200	20.95	20.66	17.09	20.87	19.89
	1500	200	21.00	20.99	17.67	21.05	20.18
	2000	200	20.25	21.81	17.08	20.71	19.96
pyannote (E2E)	–	–	22.26	16.84	17.78	19.98	19.22
	1500	200	25.00	22.22	21.97	22.67	22.97
	1500	100	**25.92**	**22.81**	**22.51**	22.88	**23.53**
	2000	200	24.10	21.98	21.00	22.71	22.45
	2000	150	24.25	22.26	21.41	22.99	22.73
	2000	100	24.97	22.66	22.20	**23.41**	23.31
	2000	50	24.50	20.67	22.14	22.89	22.55
pyannote (Cascade)	1500	200	25.06	22.65	23.01	22.51	23.31
	1500	100	**25.56**	22.85	23.03	22.82	23.57
	2000	200	24.41	22.76	22.15	22.08	22.85
	2000	150	24.50	23.03	**23.12**	23.11	23.44
	2000	100	25.04	**23.17**	23.05	**23.19**	**23.61**
	2000	50	24.33	20.79	**23.12**	23.11	22.84

Table 9: Impact of audio segmentation for ST. `A4` was used for the E2E model. † (Potapczyk and Przybysz, 2020)

provements over the best single model, `B1`. We found that an ensemble of all of our models, `A1–4` and `B1`, achieved the best result of 23.61 BLEU on tst2019 and outperformed `B1` by 2.55 BLEU. Although `A1` as a single system performs worse on tst2019 than the other single systems as shown in Table 7, including it in an ensemble with the two best single systems, `B1` and `A4`, still yielded a slight gain of 0.32 BLEU (`E2`). Therefore, we can conclude that weak models are still beneficial for ensembling.

5.3.2 Segmentation

Similar to §5.1.2, we also investigated the impact of audio segmentation for E2E-ST models. To this end, we used the `A4` model. Note that we used the same decoding hyperparameters tuned on Must-C. The results are shown in Table 9. We confirmed a similar trend to ASR. Although $(M_{dur}, M_{int}) = (1500, 100)$ showed the best performance on average, we decided to use $(M_{dur}, M_{int}) = (2000, 100)$ for submission considering the best performance on the latest IWSLT test, tst2019.

5.3.3 Cascade system

We also evaluated the cascade system with the Conformer ASR and the Transformer-Base MT trained on the `WMT10M` data (`C1`). The MT model was trained by feeding source sentences without case

information and punctuation marks. The results in Table 9 showed that the BLEU scores correlated to the WER in Table ,5 and the performance was comparable with that of `A4`. Although there is some room for improving the performance of the cascade system further by using in-domain English LM, it is difficult to conclude which modeling (cascade or E2E) is effective because the cascade system had more model parameters in the ASR decoder and MT encoder. This means that the E2E model could also be enhanced by using a similar amount of parameters.

5.3.4 Final system

Our final system was the best ensemble system `E4`, using the pyannote.audio based segmentation with $(M_{dur}, M_{int}) = (2000, 200)$[8]. This system, which was our primary submission, scored 24.14 BLEU on tst2019 as shown in Table 10. Compared to the result in Table 8, it improved by 0.53 BLEU thanks to better audio segmentation. It was also slightly higher than the IWSLT20 winner's submission by SPROL (Potapczyk and Przybysz, 2020).

We also present the results on tst2020 and tst2021 in Table 10. Our primary submission `E4` outperformed the result of last year's winner system on tst2020.

6 Conclusion

In this paper, we have presented the ESPnet-ST group's offline systems on the IWSLT 2021 submission. We significantly improved the baseline Conformer performance with multi-referenced SeqKD, Multi-Decoder architecture, segment merging algorithm, and model ensembling. Our future work includes scaling training data and careful analysis of the performance gap in different test sets.

7 Acknowledgement

---

[8]Because of time limitation, we submitted the systems before completing tuning segmentation hyperparameters.

System	Segmentation	Segment merging	M_{int}	BLEU (↑)				
				tst2019	tst2020	tst2021 ref1	ref2	both
IWSLT'20 winner♣	given	–	–	20.1	21.5	–	–	–
	own	–	–	23.96	25.3	–	–	–
E4 (primary)	pyannote	✓	200	**24.14**	**25.6**	19.3	21.2	31.4
E4+*	pyannote	✓	200	24.41	25.5	**19.7**	20.6	30.8
E4+*	pyannote	✓	100	**24.87**	**26.0**	19.5	21.1	31.3
E4+*	given	✓	100	23.72	25.1	19.4	**21.4**	**31.5**
E4+*	given	✗	–	21.10	22.3	17.4	18.4	27.7
B1	pyannote	✓	100	23.78	25.0	18.9	20.9	31.1

Table 10: BLEU scores of submitted systems on tst2020 and tst2021. ♣ (Potapczyk and Przybysz, 2020). $M_{\text{dur}} = 2000$ was used for the segment merging algorithm. *Late submission (not official). E4+ denotes E4 trained for more steps.

References

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Parnia Bahar, Tobias Bieschke, Ralf Schlüter, and Hermann Ney. 2021. Tight integrated end-to-end training for cascaded speech translation. In *Proceedings of SLT*, pages 950–957. IEEE.

Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. 2020. Pyannote.audio: neural building blocks for speaker diarization. In *Proceedings of ICASSP*, pages 7124–7128. IEEE.

Siddharth Dalmia, Brian Yan, Vikas Raunak, Florian Metze, and Shinji Watanabe. 2021. Searchable hidden intermediates for end-to-end models of decomposable sequence tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1882–1896, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. 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 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, Mauro Cettolo, and Marco Turchi. 2020. Contextualized translation of automatically segmented speech. In *Proceedings of Interspeech*, pages 1471–1475.

Marco Gaido, Matteo Negri, Mauro Cettolo, and Marco Turchi. 2021. Beyond voice activity detection: Hybrid audio segmentation for direct speech translation. *arXiv preprint arXiv:2104.11710*.

Mitchell A Gordon and Kevin Duh. 2019. Explaining sequence-level knowledge distillation as data-augmentation for neural machine translation. *arXiv preprint arXiv:1912.03334*.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented Transformer for speech recognition. In *Proceedings of Interspeech*, pages 5036–5040.

Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, et al. 2020. Recent developments on ESPnet toolkit boosted by Conformer. *arXiv preprint arXiv:2010.13956*.

Tomoki Hayashi, Ryuichi Yamamoto, Katsuki Inoue, Takenori Yoshimura, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Yu Zhang, and Xu Tan. 2020. Espnet-TTS: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit. In *Proceedings of ICASSP*, pages 7654–7658. IEEE.

Hirofumi Inaguma, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2019. Multilingual end-to-end speech translation. In *Proceedings of ASRU*, pages 570–577.

Hirofumi Inaguma, Yosuke Higuchi, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2021a. Orthros: Non-autoregressive end-to-end speech translation with dual-decoder. In *Proceedings of ICASSP*.

Hirofumi Inaguma, Tatsuya Kawahara, and Shinji Watanabe. 2021b. Source and target bidirectional knowledge distillation for end-to-end speech translation. *arXiv preprint arXiv:2104.06457*.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Niehues Jan, Roldano Cattoni, Stüker Sebastian, Mauro Cettolo, Marco Turchi, and Marcello Federico. 2018. The iwslt 2018 evaluation campaign. In *Proceedings of IWSLT*, pages 2–6.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. 2019. A comparative study on Transformer vs RNN in speech applications. In *Proceedings of ASRU*, pages 499–456.

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, Austin, Texas. Association for Computational Linguistics.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Proceedings of Interspeech*, pages 3586–3589.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Chenda Li, Jing Shi, Wangyou Zhang, Aswin Shanmugam Subramanian, Xuankai Chang, Naoyuki Kamo, Moto Hira, Tomoki Hayashi, Christoph Boeddeker, Zhuo Chen, and Shinji Watanabe. 2021. ESPnet-SE: End-to-end speech enhancement and separation toolkit designed for ASR integration. In *Proceedings of SLT*, pages 785–792. IEEE.

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*,

pages 25–30, Jeju Island, Korea. Association for Computational Linguistics.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Nicholas A Nystrom, Michael J Levine, Ralph Z Roskies, and J Ray Scott. 2015. Bridges: a uniquely flexible hpc resource for new communities and data analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, pages 1–8.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an ASR corpus based on public domain audio books. In *Proceedings of ICASSP*, pages 5206–5210. IEEE.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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. In *Proceedings of Interspeech*, pages 2613–2617.

Ngoc-Quan Pham, Thanh-Le Ha, Tuan-Nam Nguyen, Thai-Son Nguyen, Elizabeth Salesky, Sebastian Stüker, Jan Niehues, and Alex Waibel. 2020. Relative positional encoding for speech recognition and direct translation. In *Proceedings of Interspeech*, pages 31–35.

Tomasz Potapczyk and Pawel Przybysz. 2020. SR-POL's system for the IWSLT 2020 end-to-end speech translation task. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 89–94, Online. Association for Computational Linguistics.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *Proceedings of ASRU*.

Anthony Rousseau, Paul Deléglise, and Yannick Estève. 2012. TED-LIUM: An automatic speech recognition dedicated corpus. In *Proceedings of LREC*, pages 125–129.

Neville Ryant, Kenneth Church, Christopher Cieri, Alejandrina Cristia, Jun Du, Sriram Ganapathy, and Mark Liberman. 2019. The second DIHARD diarization challenge: Dataset, task, and baselines. In *Proceedings of Interspeech*, pages 978–982.

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, Berlin, Germany. Association for Computational Linguistics.

John Towns, Timothy Cockerill, Maytal Dahan, Ian Foster, Kelly Gaither, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D Peterson, et al. 2014. Xsede: Accelerating scientific discovery computing in science & engineering, 16 (5): 62–74, sep 2014. *URL https://doi.org/10.1109/mcse*, 128.

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 *Proceedings of NeurIPS*, pages 5998–6008.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. 2017. Hybrid CTC/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.