The Volctrans Neural Speech Translation System for IWSLT 2021

Chengqi Zhao Zhicheng Liu Jian Tong Tao Wang Mingxuan Wang
Rong Ye Qianqian Dong Jun Cao Lei Li
ByteDance AI Lab
{zhaochengqi.d,liuzhicheng.lzc,tongjian,wangtao.960826
wangmingxuan.89,yerong,dongqianqian
caojun.sh,lileilab}@bytedance.com

Abstract

This paper describes the systems submitted to IWSLT 2021 by the Volctrans team. We participate in the offline speech translation and text-to-text simultaneous translation tracks. For offline speech translation, our best end-to-end model achieves 7.9 BLEU improvements over the benchmark on the MuST-C test set and is even approaching the results of a strong cascade solution. For text-to-text simultaneous translation, we explore the best practice to optimize the `wait-k` model. As a result, our final submitted systems exceed the benchmark at around 7 BLEU on the same latency regime. We release our code and model to facilitate both future research works and industrial applications[1].

1 Introduction

This paper describes the neural speech translation systems submitted to IWSLT 2021 by the Volctrans team (also known as the team from ByteDance AI Lab), including cascade and end-to-end speech translation (ST) systems for the offline ST track and a simultaneous neural machine translation (NMT) system. We aim at finding the best practice for these two tracks.

For offline ST, the cascaded system often outperforms the fully end-to-end approach. Recent studies on the fully end-to-end approaches obtain promising results and attract a lot of interest. Last year's results have shown that an end-to-end model achieves an even better performance (Ansari et al., 2020) compared with the cascaded competitors. However, they introduce pre-training (Bansal et al., 2019; Stoian et al., 2020; Wang et al., 2020; Alinejad and Sarkar, 2020) and data augmentation techniques (Jia et al., 2019; Pino et al., 2020) to end-to-end models, while the cascaded is not that strong enough. Hence, in this paper, we would like to optimize the speech translation model in two aspects. First, we are devoted to building a strong cascade competitor and learns the best practice from WMT evaluation campaigns (Li et al., 2019; Wu et al., 2020), such as back translation (Sennrich et al., 2016a) and ensemble. Second, we explore various self-supervised learning methods and introduce as much semi-supervised data as possible towards finding the best practice of training end-to-end ST models. In our settings, ASR data, MT data, and monolingual text data are all considered in a progressively training framework. The results are very promising, and the final performance on the MuST-C test set surpasses the end-to-end baseline by 7.9 BLUE scores, while it is still lagging behind our cascade model by 1.5 BLUE scores. It is not surprising since some well-optimized methods for MT can not be easily used on ST, such as back translation. However, our experience shows that the external data can effectively close the gap between end-to-end models and cascade models.

In parallel, we also participate in the simultaneous NMT track, which translates in real-time. Our system is based on an efficient `wait-k` model (Elbayad et al., 2020). We investigate large-scale knowledge distillation (Kim and Rush, 2016; Freitag et al., 2017) and back translation methods. Specially, we develop a `multi-path` training strategy, which enables a unified model serving different `wait-k` paths. Our target is to obtain the best translation quality at different latency levels.

The remaining part of the paper proceeds as follows. Section 2 and section 3 describe our cascade and end-to-end systems respectively. Section 4 presents the implementation of simultaneous NMT models. Each section starts from the training sources and how we synthesize large-scale data. And then, we give details about the model structure and techniques for training and inference. We con-

[1]Code and models are available at https://github.com/bytedance/neurst/tree/master/examples/iwslt21

Proceedings of the 18th International Conference on Spoken Language Translation, pages 64–74
Bangkok, Thailand (Online), August 5–6, 2021.

Dataset	#samples	#hours
MuST-C	250,942	450
LibriSpeech	281,241	961
Common Voice	562,517	899
iwslt-corpus	157,909	231
TED-LIUM 3	111,600	165

Table 1: The statistics of audio datasets to train the ASR model. The *iwslt-corpus* and TED-LIUM 3 are filtered by an ASR model trained on MuST-C, LibriSpeech and Common Voice.

duct experiments using only the provided datasets by IWSLT 2021, and results are shown in Section 5.

2 Cascaded Speech Translation

2.1 Automatic Speech Recognition

The ASR model is transformer-like and trained on paired speech and transcript data

Datasets and Preprocessing We divide the allowed ASR datasets into two parts: clean and noisy and consider MuST-C[2], LibriSpeech (Panayotov et al., 2015), and Mozilla Common Voice as the clean datasets, and use them for training an ASR system to filter the noisy part, i.e., *iwslt-corpus*[3] and TED-LIUM 3 (Hernandez et al., 2018). We remove the training samples where the word error rate (WER) score between the ASR output and English transcript exceeds 75%. The statistics of the ASR datasets are shown in Table 1.

For model training, we extract 80-channel log Mel-filterbank coefficients with windows of 25ms and steps of 10ms on the audio input. The transcripts are lowercased and we remove all punctuation marks. Then, we apply Moses tokenizer[4] and byte pair encoding (BPE) (Sennrich et al., 2016b)[5] to the transcripts with 8,000 merge operations.

End-to-End ASR Model We refer to the recent progress of transformer-based ASR (Dong et al., 2018; Karita et al., 2019) and implement the speech transformer model, as illustrated in Figure 1 a). The feature extractor consists of two-layer CNN with 256 channels, 3×3 kernel, and stride size

[2]In this paper, MuST-C denotes the newly released English-German ST dataset (v2) by IWSLT 2021.

[3]The training corpus for IWSLT evaluation campaign over the last years.

[4]https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

[5]https://github.com/rsennrich/subword-nmt

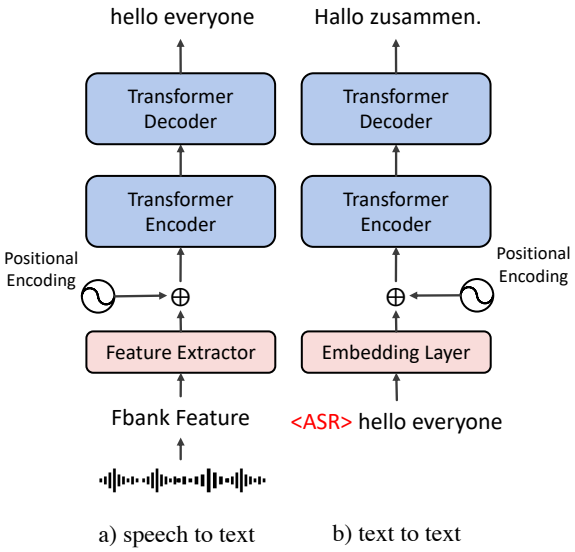


Figure 1: Overview of the cascaded speech translation model.

2, each of which is followed by a layer normalization and ReLU activation. The major architecture is the same as the transformer model, including 12 layers for the encoder and 6 layers for the decoder. The model width is 768, and the hidden size of the feed-forward layer is 3,072. The attention head is set to 12 for both self-attention and cross-attention. To train the model, we use Adam optimizer (Kingma and Ba, 2015) and set the warmup steps to 25,000. Empirically, we scale up the learning rate by 5.0 to accelerate the convergence. The ASR model is trained on 8 NVIDIA Tesla V100 GPUs with 320,000 frames per batch. And we truncate the audio frames to 3,000 and remove training samples whose transcript length exceeds 120 for GPU memory efficiency. To further improve the performance, we apply SpecAugment technique (Park et al., 2019) with frequency masking ($mF = 2, F = 27$) and time masking ($mT = 2, T = 70, p = 0.2$).

2.2 Neural Machine Translation

All MT models are based on transformer (Vaswani et al., 2017). We employ data augmentation and model ensemble techniques to improve the final performance.

Datasets and Preprocessing We utilize English-German (EN-DE) parallel sentences from WMT

2020[6], OpenSubtitles 2018[7], MuST-C and *iwslt-corpus* for training. We filter the parallel corpora following the rules listed in Li et al. (2019), with a much stricter constrain on word alignment. Additionally, we randomly select 10% sentences separately from both sides of the original WMT and OpenSubtitles corpus for data augmentation (see below), along with the transcripts in ASR datasets described in sec 2.1.

As for text preprocessing, we apply Moses tokenizer and BPE with 32,000 merge operations on each side.

Tagged Back-Translation Back-translation (Sennrich et al., 2016a) is an effective way to improve the translation quality by leveraging a large amount of monolingual data and has been widely used in WMT evaluation campaigns. In our setting, we add a "<BT>" tag to the source side of back-translated data to prevent overfitting on the synthetic data, which is also known as tagged back-translation (Caswell et al., 2019; Marie et al., 2020).

Knowledge Distillation Sequence-level knowledge distillation (Kim and Rush, 2016; Freitag et al., 2017) is another useful technique to improve performance. In this way, we enlarge the training data by translating English sentences to German using a good teacher model.

ASR Output Adaptation Traditionally, the output of ASR systems is lowercased with no punctuation marks, while the MT systems receive natural texts. In our system, we attempt to make the MT systems robust to these irregular texts. A simple way to do so is to apply the same rules on the source side of the MT training set. However, empirical study shows it causes performance degradation. Inspired by the tagged back-translation method, we enhance the regular MT models with transcripts from both ASR systems and the ASR datasets, as illustrated in Figure 1 b). An extra tag "<ASR>" indicates the irregular input. Note that the basic idea to bridge the gap between the ASR output and the MT input involves additional sub-systems, like case and punctuation restoration. In our cascade system, we prefer to use fewer sub-systems, and the detailed comparison would be our future work.

Data Combination and Sampling Strategy We train transformer models with different combinations of data sets because increasing the model's diversity can benefit the model ensemble. The detailed setups are listed in Table 2. We over-sample the in-domain datasets (i.e., MuST-C/*iwslt-corpus*-related portions) to improve the in-domain performance. Specifically, to control the ratio of samples from different data sources, we sample a fixed number of sentences being proportional to $(\frac{N_s}{\sum_s N_s})^{\frac{1}{T}}$, where N_s is the number of sentences from data source s, and sampling temperature T is set to 5. Note that the MT#1 is trained on lowercased source texts without punctuation marks, while MT#2-5 use the tagged transcripts.

Model Setups We follow the transformer big setting, except that

- we deepen the encoder layers to 16.
- the dropout rate is set 0.15.
- the model width is changed to 768, the hidden size of the feed-forward layer is 3,072, and the attention head is 12 for MT#5 only.

We use Adam optimizer with the same schedule algorithm as Vaswani et al. (2017). All models are trained with a global batch size of 65,536.

2.3 Inference

We average the latest 10 checkpoints of a single training process for all the above experiments. And during inference, the "<ASR>" tag is added to the front of the ASR output. The beamwidth is set to 10 for both ASR and MT tasks.

3 End-to-End Speech Translation

Recent studies show that the fully end-to-end solution achieves promising performance when compared with the cascaded models (Ansari et al., 2020). This section will introduce how we build our end-to-end models for the offline ST task.

3.1 Training Data

The end-to-end model is trained on paired speech and translation data. We collect MuST-C and *iwslt-corpus* (after filtering described in section 2), with a total of only 681 hours transcribed and translated speech. To address the data scarcity problem, we explore the knowledge distillation technique to augment the data by leveraging ASR datasets and MT models, also known as pseudo labeling. In detail, we distill from four MT models: MT#1,

[6] http://www.statmt.org/wmt20/translation-task.html, including Common Crawl, Europarl v10, News Commentary v15, and ParaCrawl v5.1

[7] https://opus.nlpl.eu/OpenSubtitles2018.php

Dataset	Size	MT#1	MT#2		MT#3	MT#4	MT#5
			pretrain	fine-tune			
WMT 2020	13.7M	P	P	/	P	/	P
OpenSubtitles 2018	10.7M	P	P	/	P	P	/
MuST-C	0.25M	P	P/BT/SR	P/BT/SR	P/SR/KD	P/BT/SR	P/BT/SR
iwslt-corpus	0.16M	/	P/BT/SR	P/BT/SR	P/SR/KD	P/SR	P/BT/SR
TED-LIUM 3 (EN)	0.11M	/	/	/	KD	/	/
Common Voice (EN)	0.56M	/	/	/	KD	/	/
extra monolingual (EN/DE)	6.77M	/	/	BT	KD	BT	BT

Table 2: The statistics of MT datasets after data filtering and the detailed combination modes of datasets for difference MT models (MT#1-5). The MT#1 setting is used for training both DE→EN and EN→DE directions. "P" denotes the parallel corpus. "BT" is the back-translated data using MT#1 (DE→EN). "SR" indicates the irregular data from both ASR datasets and the ASR model. "KD" is the synthetic data generated by MT#2.

Dataset	#samples	#hours
MuST-C	1,198,056	2,186
iwslt-corpus	746,714	1,112
LibriSpeech	1,117,394	3,833
Common Voice	2,212,581	3,546
TED-LIUM 3	384,389	577

Table 3: The size of audio datasets with data augmentation to train the end-to-end ST model.

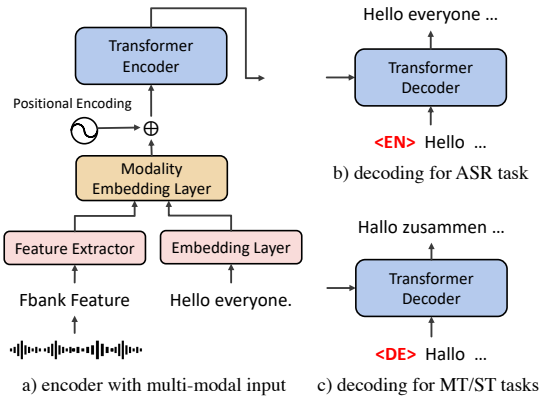


Figure 2: Overview of the end-to-end ST model with progressive multi-task learning. Note that the audio and text inputs are unnecessary to be aligned during training.

MT#2, an ensemble of MT#3-5, and MT#3-R2L which is trained with the same setting as MT#3 and generates the target translations in the right to left fashion. We filter the augmented samples with bad alignment scores as the same as data filtering in MT. The statistics of training data is shown in Table 3.

Moreover, two additional copies of the original and the augmented training data are created by modifying the speed to 110% and 90% of the initial rate, which makes a 3-fold training set.

3.2 Speech Transformer for End-to-End ST

As a baseline system, the model architecture and training configurations are the same as the end-to-end ASR in our cascade system, except for the learning rate, which is scaled up by 3.0 for ST. We initialize the feature extractor and encoder from the corresponding component of ASR.

We keep the cases and punctuation marks on the target side and apply Moses tokenizer and BPE to the translations with 32,000 merge operations.

3.3 Progressive Multi-task Learning

Inspired by the multi-task learning framework for ST and the progressive training strategy (Tang et al., 2020; Ye et al., 2021), we introduce PMTL-ST, a progressive multi-task learning framework for speech translation, which can leverage additional ASR and MT data for training. As illustrated in Figure 2 a), the encoder accepts both audio and text inputs. Then we add a modality embedding to the representation to indicate audio input or text before passing to the shared transformer encoder. For decoding, we involve "<EN>" and "<DE>" tokens to make the decoder compatible with ASR and translation (MT/ST) tasks, as shown in 2 b)/c).

For progressive training, we separately train an ASR model and an MT model via different branches in Figure 2. Then, we initialize the feature extractor and the audio modality embedding from the ASR model, and the rest of the model parameters are initialized by the MT model. The final model is trained jointly with ASR, MT, and ST.

All other training configurations, such as batch size and learning rate, are the same as the corresponding single task described before. Additionally, for the PMTL-ST models, we jointly learn the

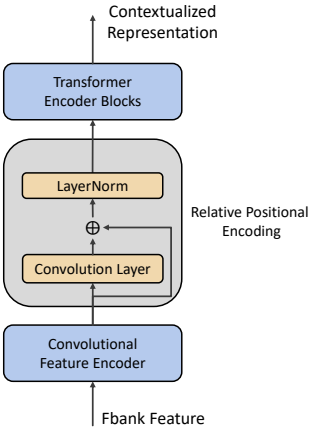


Figure 3: The proposed fbank2vec network for audio feature encoding.

sentencepiece[8] model with 16,000 tokens on the mixture of English and German texts.

3.4 Fbank2vec

Inspired by the recent progress of speech representation learning, like wav2vec 2.0 (Baevski et al., 2020), we introduce a fbank2vec network to learn contextualized audio representations from log Mel-filterbank features, as shown in Figure 3.

Convolutional Feature Encoder The encoder consists of two blocks containing a convolution followed by layer normalization and a GELU activation (Hendrycks and Gimpel, 2016). The convolution in each block has 512 channels with 3×3 kernel and stride size 2.

Relative Positional Encoding We use a group convolution layer to model the relative positional embeddings as Baevski et al. (2020) does. The kernel size is 128, and the number of groups is 16.

Contextualized Encoder The final contextualized audio representations are generated by several transformer encoder blocks. In our setting, we stack 6 layers of the post-norm transformer, and the inner activation function for the feed-forward layers is GELU. In turn, the number of shared encoder layers in Figure 2 is changed to 6.

We insert the fbank2vec network in the front of the feature extractor. The feature extractor further reduces the dimension of audio representations by one convolution layer with 5×5 kernel and stride size 2. The number of channels keeps the same as the dimension of fbank2vec output.

We experiment with two setups, fbank2vec-768 and fbank2vec-512. The fbank2vec-768 means that

- the dimension of fbank2vec output is 768;
- inner the contextualized encoder, the hidden size of feed-forward layers is 3,072, and the head of the self-attention layers is 12.

For the fbank2vec-512, the numbers are 512, 2,048, and 8, respectively. Note that the fbank2vec module is pretrained by an ASR task and the overall model follows the progressive multi-task learning framework, so the configurations of word embeddings, the shared encoder and decoder vary accordingly.

4 Simultaneous Translation

This section describes our submissions to the text-to-text simultaneous speech translation track for English to German (EN2DE) and English to Japanese (EN2JA). For versatility, we adopt identical methods for these two language pairs.

4.1 Training Data

The training data for EN→DE is from MuST-C, OpenSubtitles 2018, and WMT 2020 datasets. And for EN→JA, we use the parallel and monolingual data from the WMT 2020 news task.

Data Preprocessing We follow the data filtering process proposed in WMT works (Li et al., 2019; Wu et al., 2020), including language detection, length ratio filtering, dictionary alignment, and so on. For pre-processing, we first apply MeCab[9] tokenizer to the Japanese sentences. Then, words are segmented into subword units using sentencepiece toolkit for both language pairs. We jointly learn on the source and target side with a vocabulary of 10,000 tokens.

Data Augmentation Similar to section 2.2, we utilize tagged back-translation (BT) and knowledge distillation (KD) strategies to improve the performance of simultaneous NMT. We experiment with both LightConv (Wu et al., 2018) and transformer models. The model with the best BLEU score on the development set is chosen for data augmentation. The statistics of all training data and model settings are presented in Table 4 and Table 5 respectively.

[8] https://github.com/google/sentencepiece

[9] https://github.com/taku910/mecab

Dataset	Size	MT#0	MT#1	MT#2	MT#3	MT#4	MT#5
EN → DE							
WMT 2020(EN → DE)	41.14M	P	P	P	P	P/FT	FT
OpenSubtitles 2018	13.84M	P	P	P	P	P/BT/FT	FT/BT
MuST-C	0.23M	P	P/BT	P/BT	P/BT	P/BT/FT	FT/BT
monolingual(EN/DE)	10.25M	P	BT	BT	BT	BT	BT
EN → JA							
WMT 2020(EN → JA)	18.19M	P	P/BT	P/BT	P/BT	P/BT/FT	BT/FT

Table 4: The statistics of MT datasets and the combination modes of datasets for simultaneous NMT models. "P" indicates the parallel corpus. "BT" means the back-translated data generated by MT#0. "FT" is the forward-translated data generated by MT#1-3.

#	Model Arch	Enc	Dec	Emb
0	Transformer	6	6	1024
1	Transformer	6	6	1024
2	Transformer	50	6	1024
3	LightConv	6	6	1024
4	Transformer	16	3	768
5	Transformer	16	3	768

Table 5: The model setups. "Enc", "Dec" denote the number of encoder and decoder layers. "Emb" means the embedding size and the hidden size.

4.2 Efficient `wait-k` Model

Our simultaneous NMT systems are based on transformer `wait-k` models, which first read k source tokens and then alternate between reading and writing (translating). Formally, when decoding the sentence $\mathbf{x}$, the number of visible source tokens is constrained within $\min(k+t-1, |\mathbf{x}|)$ at decoding step t, where k is the hyper-parameter controlling the latency. Furthermore, to avoid recomputing the hidden states of the encoder each time a token is read, we implement incremental unidirectional encoders (Elbayad et al., 2020). And `multi-path` training is also applied to leverage more possible `wait-k` paths which refers that hyper-parameter $k \in [3, 9]$ is random selected at each batch during training.

Models are trained with a batch size of 32,000 tokens on Tesla V100 GPUs. We average the last 6 checkpoints once the model converges.

4.3 Inference

We explore the look-ahead beam search strategy for inference. Specifically, we apply beam search to generate $M (M > 1)$ tokens at each decoding step and pick the first token in the one with the highest log-probability out of multiple decoding paths. The look-ahead beam search achieves consistent performance improvement when k_{eval} is small while its performance improvement is insignificant with a large k_{eval}. This search method is excluded from our final submissions due to its higher latency, and we choose the greedy search instead.

Additionally, we split the source sentences into sub-sentences once the end-of-sentence punctuation is recognized. Though it may result in a slight performance drop due to the lack of context, we can obtain a much lower latency.

For the final submissions, we use ensemble models. We train several models with different k_{train} values and disjoint subsets of training data for data diversity. Each model produces different latency-quality trade-offs.

5 Experimental Results

We conduct all our experiments using NeurST (Zhao et al., 2020) and report results for the submitted speech translation tasks in this section. It is worth noting that all transcripts and translations in the test sets are removed from the training data.

When evaluating the offline ST models, tags such as applause and laughing are removed from both hypothesis and reference. We use word error rate (WER) to evaluate the ASR model and report case-sensitive detokenized BLEU[10] for MT. No other data segmentation techniques are applied to the dev/test sets. Results on MuST-C *dev* and *tst-COMMON*, as well as *dev(v1)* and *tst-COMMON(v1)* from MuST-C v1 (Gangi et al., 2019) are listed together, which serve as strong baselines for comparison purpose in the end-to-end speech translation field.

When evaluating the simultaneous translation, we use the official SimulEval (Ma et al., 2020) toolkit and report case-sensitive detokenized BLEU (Post, 2018) and Average Lagging (Ma et al., 2019)

[10] https://github.com/jniehues-kit/sacrebleu

#	System	*dev*	*tst-COM*	*dev(v1)*	*tst-COM(v1)*	Training data composition
Pure MT						
1	MT (w/o punc. & lc)	32.0	34.1	32.2	34.0	MT (see Table 2)
2	MT (w/ punc. & tc)	33.8	36.2	33.7	35.9	
3	ensemble MT (w/o punc. & lc)	33.8	35.2	33.8	35.3	
4	ensemble MT (w/ punc. & tc)	34.7	36.7	34.6	36.2	
Cascaded ASR → MT						
5	AppTek/RWTH (Bahar et al., 2020)	-	-	-	29.7	/
6	ASR → MT	29.9	32.1	28.4	31.3	ASR+MT
7	ASR → ensemble MT	**31.7**	**33.3**	**30.1**	**32.3**	/
End-to-End ST						
8	direct ST baseline	23.9	23.9	-	-	MuST-C ONLY
9	direct ST	28.9	29.9	27.9	29.5	ST+ST Augm. by MT#1&2
10	direct ST++	29.6	**30.4**	**28.3**	**29.7**	ST All
11	direct ST++*	**30.0**	30.2	28.2	29.6	ST All
12	XSTNet-768 (Ye et al., 2021)	30.4	**31.1**	-	**30.3**	ASR+MT+ST All
13	direct ST + fbank2vec-512	28.7	29.1	26.7	27.6	ST All
14	PMTL-ST + fbank2vec-768	29.6	29.6	26.9	28.1	ASR+MT+ST All
15	PMTL-ST + fbank2vec-768 ++	30.8	**31.1**	**28.8**	30.1	ASR+MT+ST All+speed pertub
16	PMTL-ST + fbank2vec-768 ++*	**30.9**	**31.1**	**28.8**	30.1	ASR+MT+ST All+speed pertub
17	ensemble (9, 10, 11)	30.4	31.2	29.0	30.6	/
18	ensemble (15, 16)	31.0	31.1	28.8	30.1	/
19	ensemble (14, 15, 16)	31.4	31.5	29.3	30.6	/
20	ensemble (13, 14, 15, 16)	**31.6**	**31.8**	**29.5**	**30.8**	/

Table 6: The overall results of the offline speech translation. The MT model used in the cascade approach is MT#2 and the ensemble MT model is formed by MT#2-MT#5. The direct ST++* is the same as direct ST++ with different random seed for in-domain data over-sampling. The PMTL-ST + fbank2vec-768 ++* is continuously trained from PMTL-ST + fbank2vec-768 ++. *tst-COM* is the abbreviation for *tst-COMMON*.

Testset	WER
dev	5.2
tst-COMMON	5.7
dev(v1)	10.6
tst-COMMON(v1)	7.4

Table 7: The WER of the ASR system for the offline ST.

on MuST-C *tst-COMMON* (EN2DE) and IWSLT21 dev set (EN2JA).

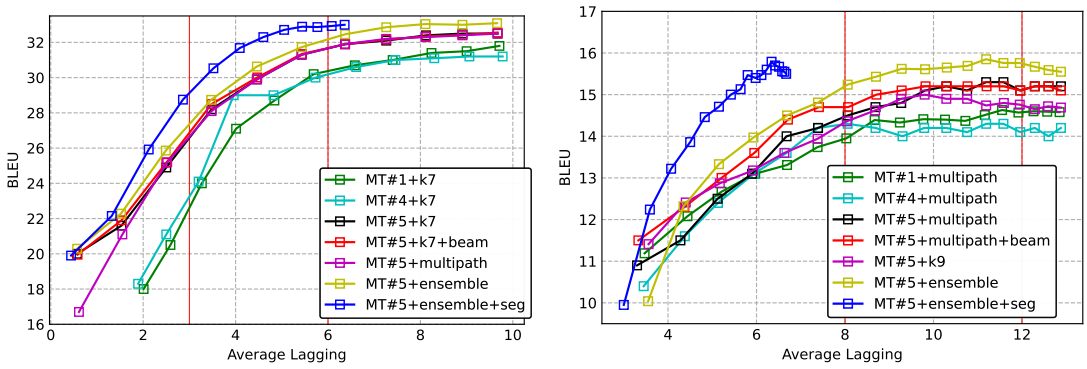
5.1 Offline Speech Translation

The overall performance of the offline ST and the ASR component used in the cascade system are listed in Table 6 and Table 7 respectively.

In Table 6, line 1-4 show the performance of our pure MT systems, which translate the lowercased ground truth transcripts with no punctuation marks, and the natural texts. As seen, there may be no essential improvements with the "<ASR>" tag on the irregular input (up to 2 BLEU gap on the single model), and it suggests that text restoration has the potential to narrow the gap. Line 6-7 present the results of translating the ASR output, and we see our cascaded approach surpasses last year's best cascade system (line 5) by 2.6 BLEU. However, there is still a significant loss of up to 3 BLEU scores than line 1/3 due to ASR errors.

The results of our end-to-end solutions are presented in line 8-20, where line 8 is a benchmark model (Zhao et al., 2020) trained on the MuST-C dataset only. With the growth of model capacity (256d→768d) and data augmentation, we obtain 6 BLEU improvement on the *tst-COMMON* over the benchmark (line 8). Then, increasing the size of augmented data gains slight improvement, as comparing line 9 to line 10/11 (+0.3∼0.5 BLEU scores). Line 13-16 show the results of our proposed fbank2vec. As shown in line 15, we achieve 31.1 BLEU on *tst-COMMON*, the best single model with fbank2vec, progressive multi-task learning, and speed perturbation. We obtain 31.8 BLEU (line 20) for the final ensemble model, which surpasses the end-to-end benchmark by 7.9 BLEU scores and is approaching the cascade system with a nearly 1.5 BLEU gap.

Lastly, our primary cascade system is line 7, and the primary end-to-end system is line 20 for submission, which achieves higher performance via model ensemble.



(a) EN→DE (b) EN→JA

Figure 4: Latency-quality trade-offs of the simultaneous NMT. $k7/9$ means $k_{\text{train}} = 7/9$. MT#X indicate the aforementioned training datasets and model settings in Table 4 and 5. `beam` refers to our look-ahead beam search strategy. `seg` means that the sentences are pre-splited during inference. `multipath` means that k is random selected during training.

		Low	Medium	High
EN → DE	Ensemble	25.86	31.73	33.21
	+seg	28.75	32.87	32.97
EN → JA	Ensemble	14.81	15.85	15.85
	+seg	15.79	15.79	15.79

Table 8: Performance of our final submissions models on MuST-C *tst-COMMON* for English-German and IWSLT21 dev set for English-Japanese.

5.2 Simultaneous Translation

We evaluate the simultaneous NMT systems with different combinations of strategies and present our results in Figure 4. Then we report the performance on different latency regimes in Table 8.

As shown in Figure 4, we can obtain remarkable BLEU improvements by training with only the knowledge distilled data (black) comparing to the filtered parallel data (green) and back-translated data (magenta), on average 1.0 BLEU improvement on EN→DE and 0.5 on EN→JA. The possible reasons may be: 1) Noise in origin data is migrated, like non-parallel sentence pairs. 2) Complex sentences with diverging word order are excluded, and the machine-translated texts, i.e., translationese, sometimes have simpler expressions.

We can see that the proposed look-ahead beam search (red) is competitive when k_{eval} is relatively small but is comparable with the greedy search when k_{eval} is large. So overall considering translation latency, we use the greedy search for our final submissions. As for `multi-path` training, we see it achieves limited BLEU improvement in our experiments.

# - System	tst2020	tst2021 ref2	tst2021 ref1	tst2021 both
7 - Cascade (ensemble)	22.2	21.8	17.1	29.5
6 - Cascade (single)	21.0	20.3	16.4	27.7
20 - Direct (ensemble)	24.3	21.7	18.7	31.3
16 - Direct (single)	23.5	21.6	18.2	30.6
17 - Direct (ensemble)	22.4	21.1	17.5	29.2
10 - Direct (single)	21.6	20.4	17.0	28.1

Table 9: BLEU of the IWSLT 2021 submissions for offline speech translation task. The rows in bold are our primary systems. The `ref1` of tst2021 is originally from the TED website, while the `ref2` is newly created for this year's campaign.

For our final submission of EN→DE, we use the ensemble model, which consists of three transformer models trained on different dataset combinations, with $k_{\text{train}} = 7$. For EN→JA, the submitted model is formed by two transformer models, with $k_{\text{train}} = \infty$ (trained on full sentences) and `multi-path` training respectively. As presented in Figure 4, the model ensemble technique leads to at least 0.5 BLEU improvement on average (yellow). Additionally, with the sentence segmentation (bleu), the average lagging is significantly reduced. As a result, our final submitted systems exceed the baseline system at around 7 BLEU on the same latency regime.

6 Final Results

Table 9 lists the final results of the IWSLT 2021 offline ST track. Surprisingly, we find that our end-to-end models significantly surpass the cascade systems, which is different from our conclusions on

System	BLEU	AL	AP	DAL
EN → DE				
MT(Low Latency)	23.24	3.08	0.68	4.25
MT(Mid Latency)	27.22	6.30	0.81	9.24
MT(High Latency)	26.82	12.03	0.92	12.39
EN → JA				
MT(Low Latency)	16.91	6.54	0.89	11.26
MT(Mid Latency)	16.91	6.54	0.89	11.26
MT(High Latency)	16.97	11.27	0.97	11.90

Table 10: Performance of the IWSLT 2021 submissions for simultaneous NMT on the blind test set.

the MuST-C test sets. We think this may be caused by the reference of tst2021. Since the `ref1` of tst2021 is the original one from the TED website, the translations could be much shorter for subtitling, and our end-to-end models may fit well on it.

Table 10 shows the official evaluation for our simultaneous NMT systems.

7 Conclusion

This paper summarizes the results of the shared tasks in the IWSLT 2021 produced by the Volctrans team. We investigate the performance of the end-to-end solutions with data augmentation and progressively training framework for the offline ST task. Our end-to-end approach surpasses the last year's best cascaded system by 1 BLEU, but it is still lagging behind our cascade model by 1.5 BLEU scores on MuST-C test sets. However, our end-to-end solutions achieve promising performance on tst2020 and tst2021. Afterwards, we develop the efficient `wait-k` model with `multi-path` training, and large-scale knowledge distillation and back translation methods. The final submitted systems exceed the baseline systems at 7 BLEU on the same regime. We see the data augmentation technique plays the most important role in these tasks. In the future, we would like to explore a more extensive data condition on both modality and quantity. We hope our practice could facilitate batch research works and industrial applications.

References

Ashkan Alinejad and Anoop Sarkar. 2020. Effectively pretraining a speech translation decoder with machine translation data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8014–8020.

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondrej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander H. Waibel, and Changhan Wang. 2020. FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN. In *Proceedings of the 17th International Conference on Spoken Language Translation, IWSLT 2020*, pages 1–34.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*.

Parnia Bahar, Patrick Wilken, Tamer Alkhouli, Andreas Guta, Pavel Golik, Evgeny Matusov, and Christian Herold. 2020. Start-before-end and end-to-end: Neural speech translation by apptek and RWTH aachen university. In *IWSLT*, pages 44–54. Association for Computational Linguistics.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *NAACL-HLT (1)*, pages 58–68. Association for Computational Linguistics.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged back-translation. In *Proceedings of the Fourth Conference on Machine Translation, WMT 2019*.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pages 5884–5888.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2020. Efficient wait-k models for simultaneous machine translation. In *INTERSPEECH*, pages 1461–1465. ISCA.

Markus Freitag, Yaser Al-Onaizan, and Baskaran Sankaran. 2017. Ensemble distillation for neural machine translation. *CoRR*, abs/1702.01802.

Mattia Antonino 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 (NAACL-HLT 2019)*, pages 2012–2017.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

François Hernandez, Vincent Nguyen, Sahar Ghannay, Natalia A. Tomashenko, and Yannick Estève. 2018. TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation. In

Speech and Computer - 20th International Conference, SPECOM 2018, Proceedings, Lecture Notes in Computer Science.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J. Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *ICASSP*, pages 7180–7184. IEEE.

Shigeki Karita, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, Wangyou Zhang, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, and Ryuichi Yamamoto. 2019. A comparative study on transformer vs RNN in speech applications. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019*, pages 449–456.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*.

Bei Li, Yinqiao Li, Chen Xu, Ye Lin, Jiqiang Liu, Hui Liu, Ziyang Wang, Yuhao Zhang, Nuo Xu, Zeyang Wang, Kai Feng, Hexuan Chen, Tengbo Liu, Yanyang Li, Qiang Wang, Tong Xiao, and Jingbo Zhu. 2019. The NiuTrans machine translation systems for WMT19. In *Proceedings of the Fourth Conference on Machine Translation*.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *ACL (1)*, pages 3025–3036. Association for Computational Linguistics.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020. Simuleval: An evaluation toolkit for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 144–150.

Benjamin Marie, Raphael Rubino, and Atsushi Fujita. 2020. Tagged back-translation revisited: Why does it really work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, pages 5206–5210.

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 *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 2613–2617.

Juan Pino, Qiantong Xu, Xutai Ma, Mohammad Javad Dousti, and Yun Tang. 2020. Self-training for end-to-end speech translation. In *Proc. Interspeech 2020*, pages 1476–1480.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Berlin, Germany, Volume 1: Long Papers*.

Mihaela C. Stoian, Sameer Bansal, and Sharon Goldwater. 2020. Analyzing ASR pretraining for low-resource speech-to-text translation. In *ICASSP*, pages 7909–7913. IEEE.

Yun Tang, Juan Pino, Changhan Wang, Xutai Ma, and Dmitriy Genzel. 2020. A general multi-task learning framework to leverage text data for speech to text tasks. *CoRR*, abs/2010.11338.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020. Curriculum pre-training for end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3728–3738.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2018. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*.

Liwei Wu, Xiao Pan, Zehui Lin, Yaoming Zhu, Mingxuan Wang, and Lei Li. 2020. The volctrans machine translation system for WMT20. In *WMT@EMNLP*, pages 305–312. Association for Computational Linguistics.

Rong Ye, Mingxuan Wang, and Lei Li. 2021. End-to-end speech translation via cross-modal progressive training. *CoRR*, abs/2104.10380.

Chengqi Zhao, Mingxuan Wang, and Lei Li. 2020. Neurst: Neural speech translation toolkit. *CoRR*, abs/2012.10018.