SimulSpeech: End-to-End Simultaneous Speech to Text Translation

Yi Ren [*]
Zhejiang University
rayeren@zju.edu.cn

Jinglin Liu [*]
Zhejiang University
jinglinliu@zju.edu.cn

Xu Tan [*]
Microsoft Research
xuta@microsoft.com

Chen Zhang [*]
Zhejiang University
zc99@zju.edu.cn

Tao Qin
Microsoft Research
taoqin@microsoft.com

Zhou Zhao[†]
Zhejiang University
zhaozhou@zju.edu.cn

Tie-Yan Liu
Microsoft Research
tyliu@microsoft.com

Abstract

In this work, we develop SimulSpeech, an end-to-end simultaneous speech to text translation system which translates speech in source language to text in target language concurrently. SimulSpeech consists of a speech encoder, a speech segmenter and a text decoder, where 1) the segmenter builds upon the encoder and leverages a connectionist temporal classification (CTC) loss to split the input streaming speech in real time, 2) the encoder-decoder attention adopts a wait-k strategy for simultaneous translation. SimulSpeech is more challenging than previous cascaded systems (with simultaneous automatic speech recognition (ASR) and simultaneous neural machine translation (NMT)). We introduce two novel knowledge distillation methods to ensure the performance: 1) Attention-level knowledge distillation transfers the knowledge from the multiplication of the attention matrices of simultaneous NMT and ASR models to help the training of the attention mechanism in SimulSpeech; 2) Data-level knowledge distillation transfers the knowledge from the full-sentence NMT model and also reduces the complexity of data distribution to help on the optimization of SimulSpeech. Experiments on MuST-C English-Spanish and English-German spoken language translation datasets show that SimulSpeech achieves reasonable BLEU scores and lower delay compared to full-sentence end-to-end speech to text translation (without simultaneous translation), and better performance than the two-stage cascaded simultaneous translation model in terms of BLEU scores and translation delay.

1 Introduction

Simultaneous speech to text translation (Fügen et al., 2007; Oda et al., 2014; Dalvi et al., 2018), which translates source-language speech into target-language text concurrently, is of great importance to the real-time understanding of spoken lectures or conversations and now widely used in many scenarios including live video streaming and international conferences. However, it is widely considered as one of the challenging tasks in machine translation domain because simultaneous speech to text translation has to understand the speech and trade off translation accuracy and delay. Conventional approaches to simultaneous speech to text translation (Fügen et al., 2007; Oda et al., 2014; Dalvi et al., 2018) divide the translation process into two stages: simultaneous automatic speech recognition (ASR) (Rao et al., 2017) and simultaneous neural machine translation (NMT) (Gu et al., 2016), which cannot be optimized jointly and result in inferior accuracy, and also incurs more translation delay due to two stages.

In this paper, we move a step further to translate the source speech to target text simultaneously, and develop SimulSpeech, an end-to-end simultaneous speech to text translation system. The SimulSpeech model consists of 1) a speech encoder where each speech frame can only see its previous frames to simulate streaming speech inputs; 2) a text decoder where the encoder-decoder attention follows the wait-k strategy (Ma et al., 2018) to decide when to listen and write on the source speech and target text respectively (see Figure 1); 3) a speech segmenter that builds upon the encoder and leverages a CTC loss to detect the word boundary, which is used to decide when to stop listening according to the

[*] Equal contribution.
[†] Corresponding author

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3787–3796
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

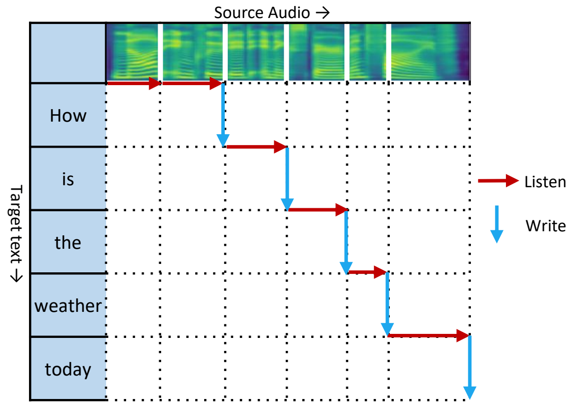


Figure 1: The wait-k strategy for simultaneous speech to text translation. The model will wait for the first k source speech segments and then start to translate a target word. After that, once receiving a new source segment, the decoder generates a new target word until there is no more source word, and then the translation degrades to the full-sentence translation. The example shows the case with $k = 2$.

wait-k strategy.

Considering the difficulty of this task, we elaborately design two techniques to boost the performance of SimulSpeech: 1) attention-level knowledge distillation that transfers the knowledge from the multiplication of the attention matrices of simultaneous NMT and ASR model to SimulSpeech to help the training of its attention mechanism; 2) data-level knowledge distillation that transfers the knowledge from a full-sentence NMT model to SimulSpeech and also reduces the complexity of data distribution (Zhou et al., 2019) to help on the optimization of SimulSpeech model.

Compared with the cascaded pipeline that trains simultaneous ASR and NMT models separately, SimulSpeech can alleviate the error propagation problem and optimize all model parameters jointly towards the end goal, as well as reduce the delay of simultaneous translation. Experiments on MuST-C[1] English-Spanish and English-German spoken language translation datasets demonstrate that SimulSpeech: 1) achieves reasonable BLEU scores and lower delay compared to full-sentence end-to-end speech to text translation (without simultaneous translation), and 2) obtains better performance than the two-stage cascaded simultaneous translation model in terms of BLEU scores and translation delay.

2 Preliminaries

In this section, we briefly review some basic knowledge for simultaneous speech to text translation, including speech to text translation, simultaneous translation based on wait-k strategy, and CTC loss for segmentation.

Speech to Text Translation Given a set of bilingual speech-text sentence pairs $D = \{(x, y) \in (\mathcal{X} \times \mathcal{Y})\}$, an speech to text machine translation model learns the parameter θ by minimizing the negative log-likelihood $-\sum_{(x,y)\in D} \log P(y|x;\theta)$. $P(y|x;\theta)$ is calculated based on the chain rule $\prod_{t=1}^{T_y} P(y_t|y_{<t}, x; \theta)$, where $y_{<t}$ represents the text tokens preceding position t, and T_y is the length of text sentence y. An encoder-attention-decoder framework is usually adopted to model the conditional probability $P(y|x;\theta)$, where the encoder maps the input audio to a set of hidden representations h and the decoder generates each target token y_t using the previously generated tokens $y_{<t}$ as well as the speech representations h. Previous works (Bérard et al., 2016; Weiss et al., 2017; Liu et al., 2019) on speech to text translation focus on the full-sentence translation where the full source speech can be seen when predicting each target token.

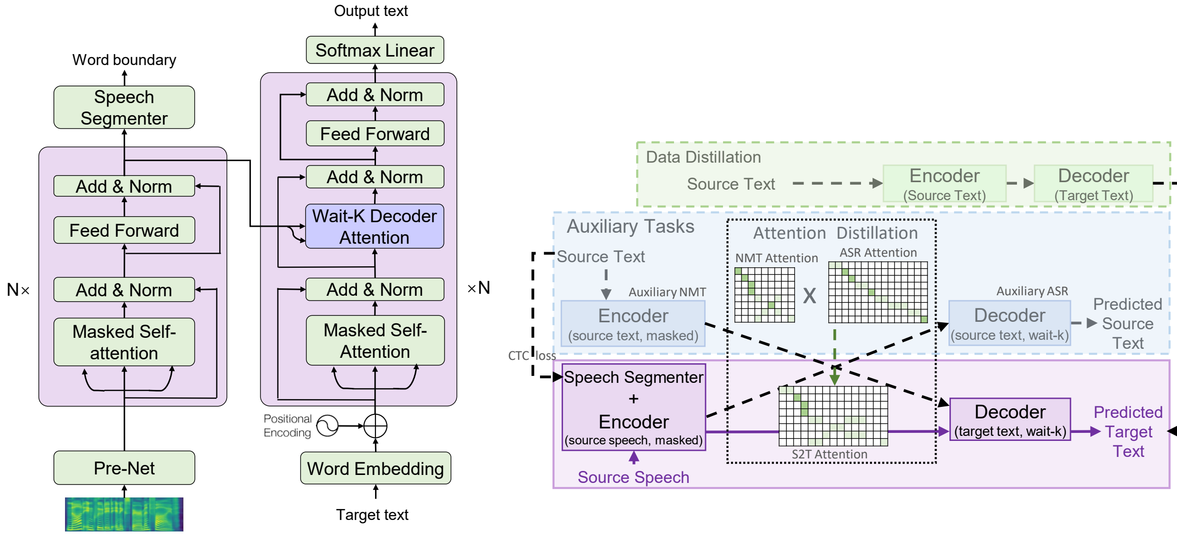
Simultaneous Translation Based on Wait-k Simultaneous translation aims to translate sentences before they are finished according to certain strategies. We use wait-k strategy (Ma et al., 2018) in this work: given a set of speech and text pairs $D = \{(x, y) \in (\mathcal{X} \times \mathcal{Y})\}$, the model with the wait-k strategy learns the parameter θ by minimizing the negative log-likelihood loss $-\sum_{(x,y)\in D} \log P(y|x;k;\theta)$, where k corresponds to the wait-k strategy. $P(y|x;k;\theta)$ is calculated based on the chain rule

$$P(y|x;k;\theta) = \prod_{t=1}^{T_y} P(y_t|y_{<t}, x_{<t+k}; \theta), \qquad (1)$$

where $y_{<t}$ represents the tokens preceding position t and T_y is the length of target sentence y, $x_{<t+k}$ represents the speech segments preceding position $t + k$. The wait-k strategy ensures that the model can see $t + k - 1$ source segments when generating the target token y_t, while can see the whole sentence if there is no more source segments.

CTC for Alignment and Segmentation The connectionist temporal classification (CTC)

[1] https://ict.fbk.eu/must-c/



(a) SimulSpeech model structure.

(b) The training pipeline for SimulSpeech model.

Figure 2: (a) The model structure of SimulSpeech. (b) The training pipeline for SimulSpeech model. The SimulSpeech model is shown in purple box, and the auxiliary training techniques are in other boxes.

loss (Graves et al., 2006) is widely used for alignment and segmentation, which maps the frame-level classification outputs of a speech sequence to a text sequence (with a different length from the speech sequence). For a text sequence y, CTC introduces a set of intermediate representation paths $\phi(y)$ called CTC paths, which has a many-to-one mapping to y since multiple CTC paths can correspond to the same text sequence. For example, both the frame-level classification outputs (CTC paths) "$HHE\emptyset L\emptyset LOO$" and "$\emptyset HHEEL\emptyset LO$" are mapped to text sequence "$HELLO$", where $\emptyset$ is the blank symbol. The likelihood of y can thus be evaluated as a sum of the probabilities of its CTC paths:

$$P(y|x) = \sum_{z \in \phi(y)} P(z|x), \qquad (2)$$

where x is the utterance consisting of speech frames and z is one of the CTC path.

3 The SimulSpeech Model

Similar to many sequence to sequence generation tasks, SimulSpeech adopts the encoder-decoder framework. As shown in Figure 2a, both the encoder and decoder follow the basic network structure of Transformer (Vaswani et al., 2017a) for neural machine translation. SimulSpeech is different from Transformer in several aspects:

- To handle speech inputs, we employ a speech pre-net (Shen et al., 2018) to extract speech features, which consists of multiple convolutional layers with the same hidden size as Transformer.

- To enable simultaneous translation, we design different attention mechanisms for the encoder and decoder. The encoder adopts masked self-attention, which masks the future frames of a speech frame when encoding it and ensures that each speech frame can only see its previous frames to simulate the real-time streaming inputs. The decoder adopts the wait-k strategy (Ma et al., 2018), as shown in Equation 1, which guarantees that each target token can only see the source segments following the wait-k strategy.

- As the wait-k strategy requires source speech to be discrete segments, we introduce a speech segmenter to split a speech sequence into discrete segments, each representing a word or phrase. The segmenter takes the outputs of the speech encoder as inputs, passes through multiple non-linear dense layers and then a softmax linear layer to predict the character in frame level. When a word boundary token (the space character in our case) is predicted by the segmenter, SimulSpeech knows a word is ended. Multiple consecutive word boundary tokens are merged into one boundary.

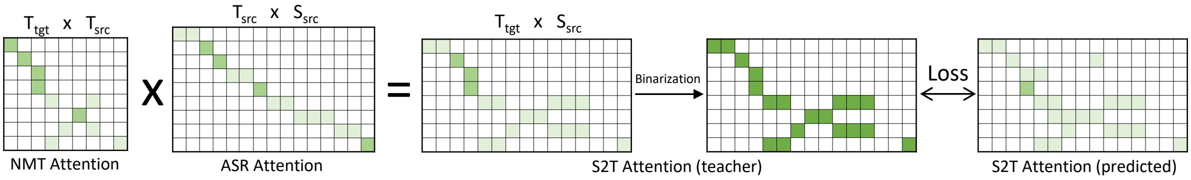


Figure 3: Details of attention-level knowledge distillation.

4 Training of SimulSpeech

The training of the SimulSpeech model is more difficult than that of an NMT model or an ASR model, since SimulSpeech involves multiple modalities (i.e., speech and text) and multiple languages. In this section, we discuss how to train the SimulSpeech model. As shown in Figure 2b, we introduce the CTC loss for the training of the speech segmenter, and attention-level and data-level knowledge distillation for the training of the overall SimulSpeech model. In SimulSpeech training, the training data are provided in the format of (source speech, source text, target text) tuples.

4.1 Training Segmenter with CTC Loss

In SimulSpeech, the speech segmenter is used to detect word boundaries, and detected boundaries are used to determine when to stop listening and switch to translation, which is critical for the performance of simultaneous translation. As it is hard to find frame-level label to guide the output of the softmax linear layer in speech segmenter, we leverage connectionist temporal classification (CTC) loss to train the speech segmenter. According to Equation 2, the CTC loss is formulated as

$$\mathcal{L}_{\text{ctc}} = -\sum_{(x,y)\in(\mathcal{X}\times\mathcal{Y}^{\text{src}})} \sum_{z\in\phi(y)} P(z|x), \quad (3)$$

where $(\mathcal{X}\times\mathcal{Y}^{\text{src}})$ denotes the set of source speech and source text sequence pairs, and $\phi(y)$ denotes the set of CTC paths for y.

During inference, we simply use the best path decoding (Graves et al., 2006) to decide the word boundary without seeing subsequent speech frames, which is consistent with the masked self-attention in speech encoder, i.e., the output of segmenter for position i depends only on the inputs at positions preceding i.

4.2 Attention-Level Knowledge Distillation

To better train the SimulSpeech model, we propose a novel attention-level knowledge distillation that is specially designed for speech to text translation, which transfers the knowledge from the multiplication of attention weights matrices of simultaneous ASR and NMT models, into the attention of the SimulSpeech model. In order to obtain the attention weights of simultaneous ASR and NMT, we add auxiliary simultaneous ASR and NMT tasks which share the same encoder or decoder with SimulSpeech model respectively, as shown in Figure 2b. The two auxiliary tasks both leverage a wait-k strategy similar to that used in SimulSpeech model.

Denote the sequence length of the source speech, source text and target text as S_{src}, T_{src} and T_{tgt} respectively. Denote the attention weights of simultaneous ASR and NMT as $A^{T_{\text{src}}\times S_{\text{src}}}$ and $A^{T_{\text{tgt}}\times T_{\text{src}}}$ respectively. Ideally, the attention weights of SimulSpeech $A^{T_{\text{tgt}}\times S_{\text{src}}}$ should satisfy

$$A^{T_{\text{tgt}}\times S_{\text{src}}} = A^{T_{\text{tgt}}\times T_{\text{src}}} \times A^{T_{\text{src}}\times S_{\text{src}}}. \quad (4)$$

However, the attention weights are difficult to learn, and the attention weights of SimulSpeech model are more difficult to learn than that of the simultaneous ASR and NMT models since SimulSpeech is much more challenging. Therefore, we propose to distill the knowledge from the multiplication of the attention weights of the simultaneous ASR and NMT, as shown in Figure 2b and Figure 3. We first multiply the attention matrix of simultaneous NMT by that of simultaneous ASR, and then binarize the attention matrix with a threshold. We then match the attention weights that is predicted by the SimulSpeech model to the binarized attention matrix, with the loss function

$$\mathcal{L}_{\text{att\_kd}} = -\mathcal{B}(A^{T_{\text{tgt}}\times T_{\text{src}}} \times A^{T_{\text{src}}\times S_{\text{src}}}) \times A^{T_{\text{tgt}}\times S_{\text{src}}}, \quad (5)$$

where $\mathcal{B}$ is the binarization operation which set the element of the matrix to 1 if above the threshold of 0.05, and otherwise to 0.

4.3 Data-Level Knowledge Distillation

Data-level knowledge distillation is widely used to help model training in various tasks and situations (Kim and Rush, 2016; Tan et al., 2019) and can boost the performance of a student model. In this work, we leverage knowledge distillation to transfer the knowledge from a full-sentence NMT teacher model to a SimulSpeech model. We train a full-sentence NMT teacher model first and then generate target text y' given source text y that is paired with source speech x. Finally, we train the student SimulSpeech with the generated target text y' which is paired with the source speech x. The loss function is formulated as

$$\mathcal{L}_{\text{data\_kd}} = -\sum_{(x,y')\in(\mathcal{X}\times\mathcal{Y}^{\text{tgt}'})} \log P(y'|x), \quad (6)$$

where $(\mathcal{X} \times \mathcal{Y}^{\text{tgt}'})$ denotes the set of speech-text sequence pairs where text is generated by the NMT teacher model.

The total loss function to train SimulSpeech model is

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{ctc}} + \lambda_2\mathcal{L}_{\text{att\_kd}} + \lambda_3\mathcal{L}_{\text{data\_kd}}, \quad (7)$$

where λ_1, λ_2, λ_3 are hyperparameters to trade off the three losses.

5 Experiments and Results

In this section, we evaluate SimulSpeech on MuST-C corpus (Di Gangi et al., 2019). First we describe experimental settings and details, then we show the experiment results, and further conduct some analyses on our model.

5.1 Experiment Settings

Datasets We use the MuST-C English-Spanish (En→Es) and English-German (En→De) speech translation corpus in our experiments. Both two datasets contain audio clips in source language, and the corresponding source-language transcripts and target-language translated text. The official data statistics and splits for train/dev/test set are shown in Table 1. For the speech data, we transform the raw audio into mel-spectrograms following Shen et al. (2018) with 50 ms frame size and 12.5 ms hop size. To simplify the model training, we remove some non-verbal annotation in the text, such as "(Laughing)", "(Music)". All the sentences are first tokenized with moses tokenizer[2] and then segmented into subword symbols using Byte Pair Encoding (BPE) (Sennrich et al., 2016), except for the label to train the speech segmenter, where we use character sequence of source text. We learn the BPE merge operations across source and target languages. We use the speech segmenter proposed in Section 3 to split the speech mel-spectrograms into segments, where each segment is regarded as discrete tokens and represents a word or short phrase.

Task	Train	Dev	Test
En→Es	229703 (496h)	1316 (2.5h)	2502 (4h)
En→De	265625 (400h)	1423 (2.5h)	2641 (4h)

Table 1: The number of sentences and the duration of audio in MuST-C dataset.

Model Configuration We use the Transformer (Vaswani et al., 2017b) as the basic SimulSpeech model structure since it achieves state-of-the-art accuracy and becomes a popular choice for recent NMT research. The model hidden size, number of heads, number of encoder and decoder-layers are set to 384, 4, 6 and 4 respectively. Considering that the adjacent hidden states are closely related in speech task, we replace the feed-forward network in Transformer with a 2-layer 1D convolutional network (Gehring et al., 2017) with ReLU activation. Left padding is used in the 1D convolutional network in the target side (Ren et al., 2019) to avoid the output token seeing its subsequent tokens in the training stage. The kernel size and filter size of 1D convolution are set to 1536 and 9 respectively. The pre-net (bottom left in Figure 2a) is a 3-layer convolutional network with left padding, whose output dimension is same as the hidden size of the transformer encoder. The decoder of the auxiliary ASR model and the encoder of the auxiliary NMT model, as well as the encoder and decoder of the NMT teacher model share the same model structures described above.

Training and Inference SimulSpeech is trained on 2 NVIDIA Tesla V100 GPUs, with totally batch size of 64 sentences. We use the Adam optimizer with the default parameters (Kingma and Ba, 2014) and learning rate schedule in Vaswani et al. (2017a). We train the SimulSpeech with auxiliary simultaneous ASR and NMT tasks by default. We set the λ_1, λ_2, λ_3 in Equation 7 as 1.0, 0.1, 1.0 respectively, according to the validation performance. SimulSpeech is trained and tested with the same k unless otherwise stated. The translation quality is evaluated by tokenized case sensitive BLEU (Papineni

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

et al., 2002) with the perl scripts[3]. Our code is based on tensor2tensor (Vaswani et al., 2018)[4].

The Metric of Translation Delay Many previous works focus on proposing the metrics of translation delay for simultaneous text to text translation, such as average proportion (AP) (Cho and Esipova, 2016) and average latency (AL) (Ma et al., 2018). The former calculates the mean absolute delay cost by each target token, while the latter measures the degree of out of sync with the speaker. In this work, we extend the AP and AL metric that are originally calculated on word sequence to speech sequence for simultaneous speech to text translation task. Our extended AP is defined as follows:

$$AP(x, y) = \frac{1}{|x|^{\text{time}}|y|} \sum_{i=1}^{|y|} t(i), \tag{8}$$

where x and y are the source speech and target text, $|x|^{\text{time}}$ is the total time duration of source speech, $|y|$ is the length of target text, $t(i)$ is real-time delay in terms of source speech when generating the i-th word in target sequence, i.e., the duration of source speech listened by the model before writing the i-th target token. Our extended AL is defined as follows:

$$AL(x, y) = \frac{1}{\tau(|x|^{\text{seg}})} \sum_{i=1}^{\tau(|x|^{\text{seg}})} g(i) - \frac{i-1}{r}, \tag{9}$$

where $|x|^{\text{seg}}$ is length of speech segments, $g(i)$ is the delay at step i, i.e., the number of source segments listened by the model before writing the i-th target token. $\tau(|x|^{\text{seg}})$ denotes the earliest timestep where our model has consumed the entire source sequence:

$$\tau(|x|^{\text{seg}}) = \arg\min_{t}(g(t) = |x|^{\text{seg}}), \tag{10}$$

and $r = |y|/|x|^{\text{seg}}$ is the length ratio between target and source sequence.

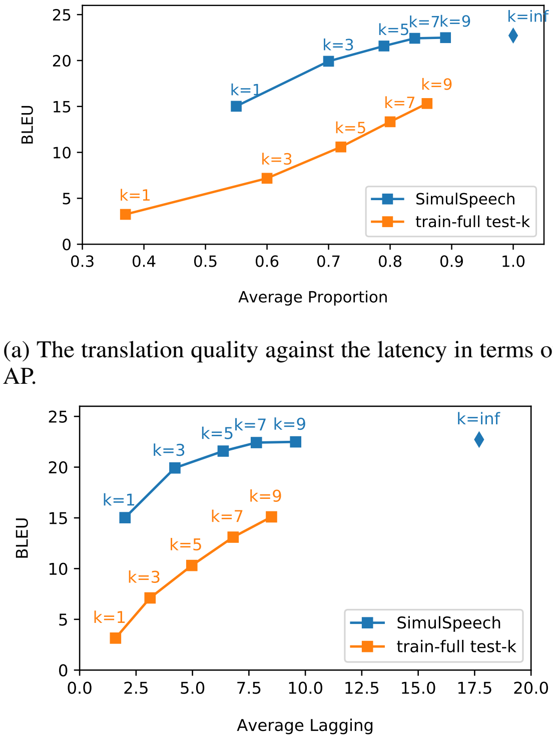
5.2 Experiment Results

Translation Accuracy First, we evaluate the performance of SimulSpeech model under different k. The BLEU scores of En-Es and En-De are shown in Table 2. We can see that the performance of our model does not drop a lot when k is small, compared to the full-sentence translation (training with k=inf).

k	1	3	5	7	9	inf
En-Es	15.02	19.92	21.58	22.42	22.49	22.72
En-Es (*FS*)	3.25	7.18	10.52	13.33	15.32	22.72
En-De	10.73	15.52	16.90	17.46	17.87	18.29
En-De (*FS*)	2.58	6.89	9.65	11.70	13.15	18.29

Table 2: The BLEU scores of SimulSpeech on the test set of the MuST-C En→Es and En →De dataset. *FS* denotes training with k=inf.

Translation Delay We plot the translation quality (in terms of BLEU score) against delay metrics (AP and AL) of our SimulSpeech model and test-time wait-k model (trained with full-sentence translation but only test with wait-k, denoted as "train-full test-k") in Figure 4a and 4b. We can see that the BLEU scores increase as k increases, with the sacrifice of translation delay. The accuracy of SimulSpeech model is always better than the test-time wait-k, which demonstrates the effectiveness of the SimulSpeech.



(a) The translation quality against the latency in terms of AP.

(b) The translation quality against the latency in terms of AL.

Figure 4: The translation quality against the latency metrics (AP and AL) on En→Es dataset.

Comparison with Cascaded Models Finally, we implement the cascaded simultaneous speech to text translation pipeline and compare the accuracy of SimulSpeech with it under the same translation

[3] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

[4] https://github.com/tensorflow/tensor2tensor

	1	2	3	4	5	6	7	8	9	10	11	12
En (source)	the	first	on	here	is	the	classic	apple.				
Es (target)	la	primera	aquí	es	la	clásica	manzana.					
ASR (wait-1)	the	first	on	here	is	the	class	sake	apple.			
ASR (wait-1) + NMT (wait-3)			pero	la	primera	vez	es	una	manzana	motivo	de	clase.
SimulSpeech (wait-3)			la	primera	es	una	manzana	clásica.				

Figure 5: An example from the test set in En→Es dataset, which demonstrates that SimulSpeech outperforms cascaded models under same delay (the delay of wait-1 for ASR plus wait-3 for NMT is equal to the delay of wait-3 for SimulSpeech). In this case, wait-1 ASR model in cascaded method does not recognize the word "classic" correctly, and results in the wrong translation in NMT model.

delay by using the same k. For cascaded method, we try all possible combinations of wait-k ASR and wait-k NMT models and report the best one. The accuracy of the two methods is shown in Table 3. It can be seen that 1) SimulSpeech outperforms the cascaded method when $k < 9$ which covers most simultaneous translation scenarios. 2) Cascaded model only outperforms SimulSpeech in larger k[5]. These results demonstrate the advantages of SimulSpeech specifically for simultaneous translation scenario. We further plot the BLEU scores of the two methods in Figure 6. It can be seen that SimulSpeech with wait-3 can achieve the same BLEU score with the cascaded method under wait-5. To sum up, SimulSpeech achieves higher translation accuracy than cascaded method under the same translation delay, and achieves lower translation delay with the same translation accuracy.

Model	k=1	k=3	k=5	k=7	k=9	k=inf
Cascaded	12.77	16.91	19.66	21.05	**23.43**	**25.60**
SimulSpeech	**15.02**	**19.92**	**21.58**	**22.42**	22.49	22.72

Table 3: The comparison between two-stage cascaded method and SimulSpeech under different wait-k on En→Es dataset.

5.3 Ablation Study

We evaluate the effectiveness of each component and show the results in Table 4. From the BLEU scores in Row 2 and Row 3, it can be seen that the translation accuracy with different wait-k can be boosted by adding auxiliary task to naive simultaneous speech to text translation model (denoted as Naive S2T).

The Effectiveness of data-level knowledge distillation We further evaluate the effectiveness of data-level knowledge distillation (Row 4 vs Row 3). The result shows that data-level knowledge distillation can achieve a large accuracy improvement.

Model	k=1	k=5	k=9
Naive S2T	9.02	14.90	15.90
+Aux	12.98	19.41	20.39
+Aux+DataKD	13.77	20.98	21.52
+Aux+AttnKD	13.74	20.64	20.90
+Aux+DataKD+AttKD (SimulSpeech)	**15.02**	**21.58**	**22.49**

Table 4: The ablation studies on En→Es dataset. The baseline model (Naive S2T) is the naive simultaneous speech to text translation model with wait-k policy. We gradually add our techniques on it to evaluate their effectiveness.

The Effectiveness of attention-level knowledge distillation We further evaluate the effectiveness of attention-level knowledge distillation. We add attention-level knowledge distillation (Row 5 vs. Row 3) to the model and find that the accuracy can also be improved. As a result, we combine all the techniques together (Row 6, SimulSpeech) and obtain the best BLEU scores across different wait-k, which demonstrates the effectiveness of all techniques we proposed for the training of SimulSpeech.

The Effectiveness of Speech Segmenter To evaluate the effectiveness of our segmenter, we compare the accuracy of SimulSpeech model using our segmentation method and the ground-truth segmentation, where we extract the segmentation from the ground-truth speech and corresponding transcripts using the alignment tool[6] and regard it as the ground-truth segmentation. As shown in Table 5, the BLEU scores of SimulSpeech using our segmentation method is close to that using ground-truth segmentation[7], which demonstrates the effectiveness of our speech segmenter.

[5] In a typical simultaneous translation scenario, k should be as small as possible, otherwise large delay is incurred.

[6] https://github.com/lowerquality/gentle

[7] Note that we cannot obtain the ground-truth segmentation during inference. Therefore the accuracy gap in Table 5 is reasonable.

Method	k=1	k=3	k=5	k=7	k=9
Ground-Truth	18.04	22.61	23.76	23.36	23.14
Our Method	15.02	19.92	21.58	22.42	22.49

Table 5: The BLEU scores of SimulSpeech on En→Es using our speech segmentation method and ground-truth segmentation.

Case Analysis We further conduct case studies to demonstrate the advantages of our end-to-end translation over the previous cascaded models. As shown in Figure 5, simultaneous ASR model makes a mistake which further affects the accuracy of downstream simultaneous NMT model, while SimulSpeech is not suffered by this problem. As a result, SimulSpeech outperforms cascaded models.

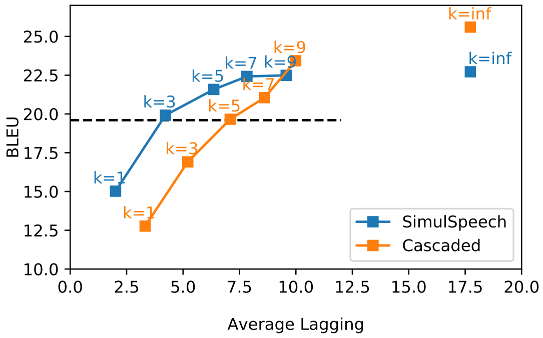


Figure 6: The comparison between SimulSpeech and the cascaded method in terms of translation accuracy and delay on En→Es dataset.

6 Related Works

6.1 Speech to Text Translation

Speech to text translation has been a hot research topic in the field of artificial intelligence recently (Bérard et al., 2016; Weiss et al., 2017; Liu et al., 2019). Early works on speech to text translation rely on a two-stage method by cascaded ASR and NMT models. Bérard et al. (2016) proposed an end-to-end speech to text translation system, which does not leverage source language text during training or inference. Weiss et al. (2017) further leveraged an auxiliary ASR model with a shared encoder with the speech to text model, regarding it as a multi-task problem. Vila et al. (2018) applied Transformer (Vaswani et al., 2017b) architecture to this task and achieved good accuracy. Bansal et al. (2018) explored speech to text translation in the low-resource setting where both data and computation are limited. Sperber et al. (2019) proposed a novel attention-passing model for end-to-end speech to text translation and achieved comparable accuracy to the cascaded models.

6.2 Simultaneous Translation

Simultaneous translation aims to translate sentences before they are finished (Fügen et al., 2007; Oda et al., 2014; Dalvi et al., 2018). Traditional speech to text simultaneous translation system usually first recognizes and segments the incoming speech stream based on an automatic speech recognition (ASR) system, and then translates it to the text in target language. And most of the previous works focus on the simultaneous machine translation part (Zheng et al., 2019): Gu et al. (2016) proposed a framework for simultaneous NMT in which an agent learns to make decisions on when to translate from the interaction with a pre-trained NMT environment. Ma et al. (2018) introduced a very simple but effective wait-k strategy for simultaneous NMT based on a prefix-to-prefix framework, which predicts the next target word conditioned on the partial source sequence the model has seen, instead of the full source sequence. The wait-k strategy will wait for the first k source words and then start to generate a target word. After that, once receiving a new source word, the decoder generates a new target word until there is no more source word, and then the translation degrades to full-sentence translation.

7 Conclusion

In this work, we developed SimulSpeech, an end-to-end simultaneous speech to text translation system that directly translates source speech into target text concurrently. SimulSpeech consists of a speech encoder, a speech segmenter, and a text decoder with wait-k strategy for simultaneous translation. We further introduced several techniques including data-level and attention-level knowledge distillation to boost the accuracy of SimulSpeech. Experiments on MuST-C spoken language translation datasets demonstrate the advantages of SimulSpeech in terms of both translation accuracy and delay.

For future work, we will design more flexible policies to achieve better translation quality and lower delay in simultaneous spoken language translation. We will also investigate simultaneous translation from the speech in a source language to the speech in a target.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (Grant No.2018AAA0100603), Zhejiang Natural Science Foundation (LR19F020006), National Natural Science Foundation of China (Grant No.61836002), National Natural Science Foundation of China (Grant No.U1611461), and National Natural Science Foundation of China (Grant No.61751209). This work was also partially funded by Microsoft Research Asia.

References

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2018. Low-resource speech-to-text translation. *arXiv preprint arXiv:1803.09164*.

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*.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. *arXiv preprint arXiv:1806.03661*.

Mattia Antonino Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. In *NAACL-HLT*, Minneapolis, MN, USA.

Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine translation*, 21(4):209–252.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*, pages 1243–1252. JMLR. org.

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. ACM.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor OK Li. 2016. Learning to translate in real-time with neural machine translation. *arXiv preprint arXiv:1610.00388*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yuchen Liu, Hao Xiong, Zhongjun He, Jiajun Zhang, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019. End-to-end speech translation with knowledge distillation. *arXiv preprint arXiv:1904.08075*.

Mingbo Ma, Liang Huang, Hao Xiong, Kaibo Liu, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, and Haifeng Wang. 2018. Stacl: Simultaneous translation with integrated anticipation and controllable latency. *arXiv preprint arXiv:1810.08398*.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *ACL*, pages 551–556.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Kanishka Rao, Haşim Sak, and Rohit Prabhavalkar. 2017. Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. In *ASRU*, pages 193–199. IEEE.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. *arXiv preprint arXiv:1905.09263*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *ICASSP*, pages 4779–4783. IEEE.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *TACL*, 7:313–325.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *NIPS 2017*, pages 6000–6010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *NIPS*, 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. In *IberSPEECH*, pages 60–63.

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*.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simultaneous translation with flexible policy via restricted imitation learning. *arXiv preprint arXiv:1906.01135*.

Chunting Zhou, Graham Neubig, and Jiatao Gu. 2019. Understanding knowledge distillation in non-autoregressive machine translation. *arXiv preprint arXiv:1911.02727*.