DASpeech: Directed Acyclic Transformer for Fast and High-quality Speech-to-Speech Translation

Qingkai Fang[1,2], **Yan Zhou**[1,2], **Yang Feng**[1,2]*
[1]Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences (ICT/CAS)
[2]University of Chinese Academy of Sciences, Beijing, China
{fangqingkai21b,zhouyan23z,fengyang}@ict.ac.cn

Abstract

Direct speech-to-speech translation (S2ST) translates speech from one language into another using a single model. However, due to the presence of linguistic and acoustic diversity, the target speech follows a complex multimodal distribution, posing challenges to achieving both high-quality translations and fast decoding speeds for S2ST models. In this paper, we propose DASpeech, a non-autoregressive direct S2ST model which realizes both *fast* and *high-quality* S2ST. To better capture the complex distribution of the target speech, DASpeech adopts the two-pass architecture to decompose the generation process into two steps, where a linguistic decoder first generates the target text, and an acoustic decoder then generates the target speech based on the hidden states of the linguistic decoder. Specifically, we use the decoder of DA-Transformer as the linguistic decoder, and use FastSpeech 2 as the acoustic decoder. DA-Transformer models translations with a directed acyclic graph (DAG). To consider all potential paths in the DAG during training, we calculate the expected hidden states for each target token via dynamic programming, and feed them into the acoustic decoder to predict the target mel-spectrogram. During inference, we select the most probable path and take hidden states on that path as input to the acoustic decoder. Experiments on the CVSS Fr$\rightarrow$En benchmark demonstrate that DASpeech can achieve comparable or even better performance than the state-of-the-art S2ST model Translatotron 2, while preserving up to 18.53$\times$ speedup compared to the autoregressive baseline. Compared with the previous non-autoregressive S2ST model, DASpeech does not rely on knowledge distillation and iterative decoding, achieving significant improvements in both translation quality and decoding speed. Furthermore, DASpeech shows the ability to preserve the speaker's voice of the source speech during translation.[2][3]

1 Introduction

Direct speech-to-speech translation (S2ST) directly translates speech of the source language into the target language, which can break the communication barriers between different language groups and has broad application prospects. Traditional S2ST usually consists of cascaded automatic speech recognition (ASR), machine translation (MT), and text-to-speech (TTS) models [1, 2]. In contrast, direct S2ST achieves source-to-target speech conversion with a unified model [3], which can (1) avoid error propagation across sub-models [4]; (2) reduce the decoding latency [5]; and (3) preserve

*Corresponding author: Yang Feng.
[2]Audio samples are available at https://ictnlp.github.io/daspeech-demo/.
[3]Code is publicly available at https://github.com/ictnlp/DASpeech.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

non-linguistic information (e.g., the speaker's voice) during translation [6]. Recent works show that direct S2ST can achieve comparable or even better performance than cascaded systems [7, 8].

Despite the theoretical advantages of direct S2ST, it is still very challenging to train a direct S2ST model in practice. Due to the linguistic diversity during translation, as well as the diverse acoustic variations (e.g., duration, pitch, energy, etc.), the target speech follows a complex multimodal distribution. To address this issue, Jia et al. [6], Inaguma et al. [7] propose the two-pass architecture, which first generates the target text with a linguistic decoder, and then uses an acoustic decoder to generate the target speech based on the hidden states of the linguistic decoder. The two-pass architecture decomposes the generation process into two steps: content translation and speech synthesis, making it easier to model the complex distribution of the target speech and achieving state-of-the-art performance among direct S2ST models.

Although the two-pass architecture achieves better translation quality, two passes of autoregressive decoding also incur high decoding latency. To reduce the decoding latency, Huang et al. [9] recently proposes non-autoregressive (NAR) S2ST that generates target speech in parallel. However, due to the conditional independence assumption of NAR models, it becomes more difficult to capture the multimodal distribution of the target speech compared with autoregressive models[4]. Therefore, the trade-off between translation quality and decoding speed of S2ST remains a pressing issue.

In this paper, we introduce an S2ST model with both high-quality translations and fast decoding speeds: DASpeech, a non-autoregressive two-pass direct S2ST model. Like previous two-pass models, DASpeech includes a speech encoder, a linguistic decoder, and an acoustic decoder. Specifically, the linguistic decoder uses the structure of DA-Transformer [11] decoder, which models translations via a directed acyclic graph (DAG). The acoustic decoder adopts the design of FastSpeech 2 [12], which takes the hidden states of the linguistic decoder as input and generates the target mel-spectrogram. During training, we consider all possible paths in the DAG by calculating the expected hidden state for each target token via dynamic programming, which are fed to the acoustic decoder to predict the target mel-spectrogram. During inference, we first find the most probable path in DAG and take hidden states on that path as input to the acoustic decoder. Due to the task decomposition of two-pass architecture, as well as the ability of DA-Transformer and FastSpeech 2 themselves to model linguistic diversity and acoustic diversity, DASpeech is able to capture the multimodal distribution of the target speech. Experiments on the CVSS Fr$\rightarrow$En benchmark show that: (1) DASpeech achieves comparable or even better performance than the state-of-the-art S2ST model Translatotron 2, while maintaining up to 18.53$\times$ speedup compared to the autoregressive model. (2) Compared with the previous NAR S2ST model TranSpeech [9], DASpeech no longer relies on knowledge distillation and iterative decoding, achieving significant advantages in both translation quality and decoding speed. (3) When training on speech-to-speech translation pairs of the same speaker, DASpeech emerges with the ability to preserve the source speaker's voice during translation.

2 Background

2.1 Directed Acyclic Transformer

Directed Acyclic Transformer (DA-Transformer) [11, 13] is proposed for non-autoregressive machine translation (NAT), which achieves comparable results to autoregressive Transformer [14] without relying on knowledge distillation. DA-Transformer consists of a Transformer encoder and an NAT decoder. The hidden states of the last decoder layer are organized as a directed acyclic graph (DAG). The hidden states correspond to vertices of the DAG, and there are unidirectional edges that connect vertices with small indices to those with large indices. DA-Transformer successfully alleviates the linguistic multi-modality problem since DAG can capture multiple translations simultaneously by assigning different translations to different paths in DAG.

Formally, given a source sequence $X = (x_1, ..., x_N)$ and a target sequence $Y = (y_1, ..., y_M)$, the encoder takes X as input and the decoder takes learnable positional embeddings $\mathbf{G} = (\mathbf{g}_1, ..., \mathbf{g}_L)$ as input. Here L is the graph size, which is set to λ times the source length, i.e., $L = \lambda \cdot N$, and λ is a hyperparameter. DA-Transformer models the *translation probability* $P_\theta(Y|X)$ by marginalizing all

[4]It is known as the multi-modality problem [10] in non-autoregressive sequence generation.

possible paths in DAG:

$$P_\theta(Y|X) = \sum_{A \in \Gamma} P_\theta(Y|A,X) P_\theta(A|X), \tag{1}$$

where $A = (a_1, ..., a_M)$ is a path represented by a sequence of vertex indexes with $1 = a_1 < \cdots < a_M = L$, and Γ contains all paths with the same length as the target sequence Y. The probability of path A is defined as:

$$P_\theta(A|X) = \prod_{i=1}^{M-1} P_\theta(a_{i+1}|a_i, X) = \prod_{i=1}^{M-1} \mathbf{E}_{a_i, a_{i+1}}, \tag{2}$$

where $\mathbf{E} \in \mathbb{R}^{L \times L}$ is the *transition probability matrix*. We apply lower triangular masking on $\mathbf{E}$ to allow only forward transitions. With the selected path A, all target tokens are predicted in parallel:

$$P_\theta(Y|A,X) = \prod_{i=1}^{M} P_\theta(y_i|a_i, X) = \prod_{i=1}^{M} \mathbf{P}_{a_i, y_i}, \tag{3}$$

where $\mathbf{P} \in \mathbb{R}^{L \times |\mathbb{V}|}$ is the *prediction probability matrix*, and $\mathbb{V}$ indicates the vocabulary. Finally, we train the DA-Transformer by minimizing the negative log-likelihood loss:

$$\mathcal{L}_{\mathrm{DAT}} = -\log P_\theta(Y|X) = -\log \sum_{A \in \Gamma} P_\theta(Y|A,X) P_\theta(A|X), \tag{4}$$

which can be calculated with dynamic programming.

2.2 FastSpeech 2

FastSpeech 2 [12] is a non-autoregressive text-to-speech (TTS) model that generates mel-spectrograms from input phoneme sequences in parallel. It is composed of three stacked modules: encoder, variance adaptor, and mel-spectrogram decoder. The encoder and mel-spectrogram decoder consist of several feed-forward Transformer blocks, each containing a self-attention layer followed by a 1D-convolutional layer. The variance adaptor contains three variance predictors including duration predictor, pitch predictor, and energy predictor, which are used to reduce the information gap between input phoneme sequences and output mel-spectrograms. During training, the ground truth duration, pitch and energy are used to train these variance predictors and also as conditional inputs to generate the mel-spectrogram. During inference, we use the predicted values of these variance predictors. The introduction of variation information greatly alleviates the acoustic multi-modality problem, which leads to better voice quality. The training objective of FastSpeech 2 consists of four terms:

$$\mathcal{L}_{\mathrm{TTS}} = \mathcal{L}_{\mathrm{L1}} + \mathcal{L}_{\mathrm{dur}} + \mathcal{L}_{\mathrm{pitch}} + \mathcal{L}_{\mathrm{energy}}, \tag{5}$$

where $\mathcal{L}_{\mathrm{L1}}$ measures the L1 distance between the predicted and ground truth mel-spectrograms, $\mathcal{L}_{\mathrm{dur}}$, $\mathcal{L}_{\mathrm{pitch}}$ and $\mathcal{L}_{\mathrm{energy}}$ compute the mean square error (MSE) loss between predictions and ground truth for duration, pitch, and energy, respectively.

3 DASpeech

In this section, we introduce DASpeech, a non-autoregressive two-pass direct S2ST model that generates target phonemes and target mel-spectrograms successively. Formally, the source speech sequence is denoted as $X = (x_1, ..., x_N)$, where N is the number of frames in the source speech. The sequences of target phoneme and target mel-spectrogram are represented by $Y = (y_1, ..., y_M)$ and $S = (s_1, ..., s_T)$, respectively. DASpeech first generates Y from X with a speech-to-text translation (S2TT)[5] DA-Transformer. Subsequently, it generates S with a FastSpeech 2-style decoder conditioned on the last-layer hidden states of the DA-Transformer. We first overview the model architecture of DASpeech in Section 3.1. In Section 3.2, we introduce our proposed training techniques that leverage pretrained S2TT DA-Transformer and FastSpeech 2 models and finetune the entire model for S2ST end-to-end. Finally, we present several decoding algorithms for DASpeech in Section 3.3.

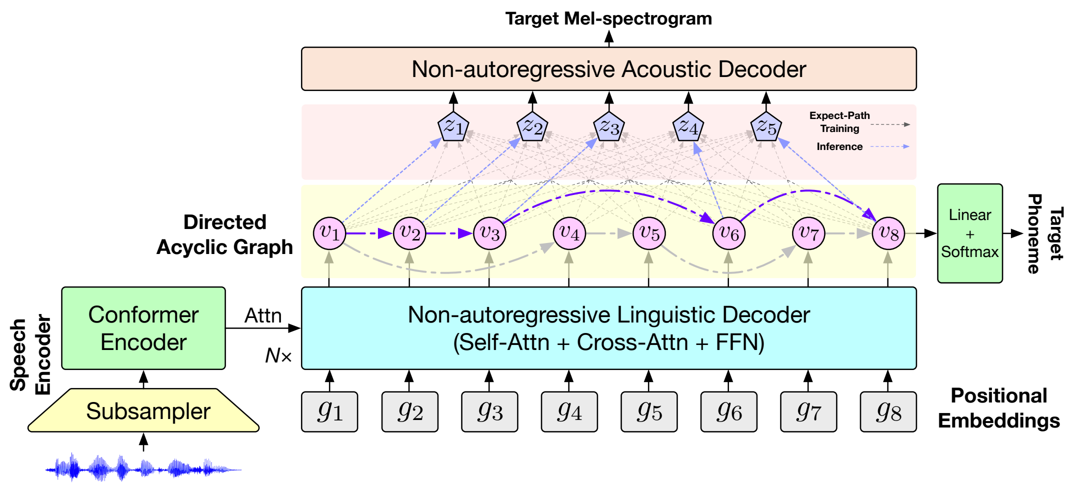


Figure 1: Overview of DASpeech. The last-layer hidden states of the linguistic decoder are organized as a DAG. During training, the input to the acoustic decoder is the sequence of expected hidden states. During inference, it is the sequence of hidden states on the most probable path.

3.1 Model Architecture

As shown in Figure 1, DASpeech consists of three parts: a speech encoder, a non-autoregressive linguistic decoder, and a non-autoregressive acoustic decoder. Below are the details of each part.

Speech Encoder Since our model takes speech features as input, we replace the Transformer encoder in the original DA-Transformer with a speech encoder. The speech encoder contains a subsampler followed by several Conformer blocks [15]. Specifically, the subsampler consists of two 1D-convolutional layers which shrink the length of input sequences by a factor of 4. Conformer combines multi-head attention modules and convolutional layers together to capture both global and local features. We use relative positional encoding [16] in the multi-head attention module.

Non-autoregressive Linguistic Decoder The linguistic decoder is identical to the decoder of DA-Transformer, which generates the target phoneme sequence from the source speech in parallel. Each decoder layer comprises a self-attention layer, a cross-attention layer, and a feed-forward layer. The decoder takes learnable positional embeddings as input, and the last-layer hidden states are organized as a DAG to model translations, as we described in Section 2.1.

Non-autoregressive Acoustic Decoder The acoustic decoder adopts the design of FastSpeech 2, which generates target mel-spectrograms from the last-layer hidden states of DA-Transformer in parallel. The model architecture is the same as that introduced in Section 2.2, except that the embedding matrix of the input phonemes is removed, since the input has changed from the phoneme sequence to the hidden state sequence.

3.2 Training

DASpeech has the advantage of easily utilizing pretrained S2TT DA-Transformer and FastSpeech 2 models. We use the pretrained S2TT DA-Transformer to initialize the speech encoder and the non-autoregressive linguistic decoder, and use the pretrained FastSpeech 2 model to initialize the non-autoregressive acoustic decoder. Finally, the entire model is finetuned for direct S2ST. This pretraining-finetuning pipeline simplifies model training and enables the use of additional S2TT and TTS data. However, end-to-end finetuning presents a major challenge due to the length discrepancy between the output from the linguistic decoder and the input to the acoustic decoder. Specifically, the hidden state sequence output by the linguistic decoder has a length of $L = \lambda \cdot N$, while the input sequence required by the acoustic decoder should have a length of M, which is the length of the ground truth phoneme sequence. Therefore, it is necessary to determine how to obtain the input

[5]In this work, speech-to-text translation refers to speech-to-phoneme translation if not otherwise specified.

sequence of acoustic decoder $\mathbf{Z} = (\mathbf{z}_1, ..., \mathbf{z}_M)$ from the last-layer hidden states of linguistic decoder $\mathbf{V} = (\mathbf{v}_1, ..., \mathbf{v}_L)$. The following introduces our proposed approach: *Expect-Path Training*.

Expect-Path Training Intuitively, the i-th input element $\mathbf{z}_i$ should be the hidden state of the vertex responsible for generating y_i. However, since there may be multiple vertices capable of generating each y_i due to numerous possible paths, we would like to consider all potential paths. To address this issue, we define $\mathbf{z}_i$ as the expected hidden state under the posterior distribution $P_\theta(a_i|X, Y)$:

$$\mathbf{z}_i = \sum_{j=1}^{L} P_\theta(a_i = j|X, Y) \cdot \mathbf{v}_j, \tag{6}$$

where $P_\theta(a_i = j|X, Y)$ refers to the probability of vertex j being the i-th vertex on path A, which means that y_i is generated by vertex j. We can compute $P_\theta(a_i = j|X, Y)$ as follows:

$$\begin{aligned} P_\theta(a_i = j|X, Y) &= \sum_{A\in\Gamma} \mathbb{1}(a_i = j) \cdot P_\theta(A|X, Y) && (7)\\ &= \sum_{A\in\Gamma} \mathbb{1}(a_i = j) \cdot \frac{P_\theta(Y, A|X)}{\sum_{A'\in\Gamma} P_\theta(Y, A'|X)} && (8)\\ &= \frac{\sum_{A\in\Gamma} \mathbb{1}(a_i = j) \cdot P_\theta(Y, A|X)}{\sum_{A\in\Gamma} P_\theta(Y, A|X)}, && (9) \end{aligned}$$

where $\mathbb{1}(a_i = j)$ is an indicator function to indicate whether the i-th vertex of path A is vertex j. To calculate $\sum_{A\in\Gamma} \mathbb{1}(a_i = j) \cdot P_\theta(Y, A|X)$ and $\sum_{A\in\Gamma} P_\theta(Y, A|X)$ in Equation (9), we employ the *forward-backward algorithm* [17], which involves two passes of dynamic programming.

Forward Algorithm The *forward probability* is defined as $\alpha_i(j) = P_\theta(y_1, ..., y_i, a_i = j|X)$, which is the probability of generating the partial target sequence $(y_1, ..., y_i)$ and ending in vertex j at the i-th step. By definition, we have $\alpha_1(1) = \mathbf{P}_{1,y_1}$ and $\alpha_1(1 < j \leq L) = 0$. Due to the Markov property, we can sequentially calculate $\alpha_i(\cdot)$ from its previous step $\alpha_{i-1}(\cdot)$ as follows:

$$\alpha_i(j) = \mathbf{P}_{j,y_i} \sum_{k=1}^{j-1} \alpha_{i-1}(k) \cdot \mathbf{E}_{k,j}. \tag{10}$$

Backward Algorithm The *backward probability* is defined as $\beta_i(j) = P_\theta(y_{i+1}, ..., y_M|a_i = j, X)$, which is the probability of starting from vertex j at the i-th step and generating the rest of the target sequence $(y_{i+1}, ..., y_M)$. By definition, we have $\beta_M(L) = 1$ and $\beta_M(1 \leq j < L) = 0$. Similar to the forward algorithm, we can sequentially calculate $\beta_i(j)$ from its next step $\beta_{i+1}(j)$ as follows:

$$\beta_i(j) = \sum_{k=j+1}^{L} \mathbf{E}_{j,k} \cdot \beta_{i+1}(k) \cdot \mathbf{P}_{k,y_{i+1}}. \tag{11}$$

Recalling Equation (9), the denominator is the sum of the probabilities of all valid paths, which is equal to $\alpha_M(L)$. The numerator is the sum of the probabilities of all paths with $a_i = j$, which is equal to $\alpha_i(j) \cdot \beta_i(j)$. Therefore, the Equation (6) can be calculated as:

$$\mathbf{z}_i = \sum_{j=1}^{L} P_\theta(a_i = j|X, Y) \cdot \mathbf{v}_j = \sum_{j=1}^{L} \frac{\alpha_i(j) \cdot \beta_i(j)}{\alpha_M(L)} \cdot \mathbf{v}_j. \tag{12}$$

The time complexity of the forward-backward algorithm is $\mathcal{O}(ML^2)$. Finally, the training objective of DASpeech is as follows:

$$\mathcal{L}_{\text{DASpeech}} = \mathcal{L}_{\text{DAT}} + \mu \cdot \mathcal{L}_{\text{TTS}}, \tag{13}$$

where μ is the weight of TTS loss. The definitions of $\mathcal{L}_{\text{DAT}}$ and $\mathcal{L}_{\text{TTS}}$ are the same as those in Equations (4) and (5).

3.3 Inference

During inference, we perform two-pass parallel decoding. First, we find the most probable path A^* in DAG with one of the decoding strategies proposed for DA-Transformer (see details below). We then feed the last-layer hidden states on path A^* to the non-autoregressive acoustic decoder to generate the mel-spectrogram. Finally, the predicted mel-spectrogram will be converted into waveform using a pretrained HiFi-GAN vocoder [18]. Since both DAG and TTS decoding are fully parallel, DASpeech achieves significant improvements in decoding efficiency compared to previous two-pass models which rely on two passes of autoregressive decoding. Considering the trade-off between translation quality and decoding efficiency, we use the following two decoding strategies for DA-Transformer in our experiments: *Lookahead* and *Joint-Viterbi*.

Lookahead Lookahead decoding sequentially chooses a_i and y_i in a greedy way. At each decoding step, it jointly considers the transition probability and the prediction probability:

$$a_i^*, y_i^* = \underset{a_i, y_i}{\arg\max}\, P_\theta(y_i|a_i, X) P_\theta(a_i|a_{i-1}, X). \tag{14}$$

Joint-Viterbi Joint-Viterbi decoding [19] finds the global joint optimal solution of the translation and decoding path via Viterbi decoding [20]:

$$A^*, Y^* = \underset{A, Y}{\arg\max}\, P_\theta(Y, A|X). \tag{15}$$

After Viterbi decoding, we first decide the target length M and obtain the optimal path by backtracking from $a_M^* = L$. More details can be found in the original paper.

4 Experiments

4.1 Experimental Setup

Dataset We conduct experiments on the CVSS dataset [4], a large-scale S2ST corpus containing speech-to-speech translation pairs from 21 languages to English. It is extended from the CoVoST 2 [21] S2TT corpus by synthesizing the target text into speech with state-of-the-art TTS models. It includes two versions: CVSS-C and CVSS-T. For CVSS-C, all target speeches are in a single speaker's voice. For CVSS-T, the target speeches are in voices transferred from the corresponding source speeches. We evaluate the models on the CVSS-C French→English (Fr→En) and CVSS-T French→English (Fr→En) datasets. We also conduct a multilingual experiment by combining all 21 language directions in CVSS-C together to train a single many-to-English S2ST model.

Pre-processing We convert the source speech to 16000Hz and generate target speech with 22050Hz. We compute the 80-dimensional mel-filterbank features for the source speech, and transform the target waveform into mel-spectrograms following Ren et al. [12]. We apply utterance-level and global-level cepstral mean-variance normalization for source speech and target speech, respectively. We follow Ren et al. [12] to extract the duration, pitch, and energy information of the target speech.

Model Configurations The speech encoder, linguistic decoder, and acoustic decoder contain 12 Conformer layers, 4 Transformer decoder layers, and 8 feed-forward Transformer blocks, respectively. The detailed configurations can be found in Table 5 in Appendix A. For model regularization, we set dropout to 0.1 and weight decay to 0.01, and no label smoothing is used. We use the HiFi-GAN vocoder pretrained on the VCTK dataset[6] [22] to convert the mel-spectrogram into waveform.

Training DASpeech follows the pretraining-finetuning pipeline. During pretraining, the speech encoder and the linguistic decoder are trained on the S2TT task for 100k updates with a batch of 320k audio frames. The learning rate warms up to 5e-4 within 10k steps. The acoustic decoder is pretrained on the TTS task for 100k updates with a batch size of 512. The learning rate warms up to 5e-4 within 4k steps. During finetuning, we train the entire model for 50k updates with a batch of 320k audio frames. The learning rate warms up to 1e-3 within 4k steps. We use Adam optimizer [23] for both pretraining and finetuning. For the weight of TTS loss μ, we experiment with $\mu \in \{1.0, 2.0, 5.0, 10.0\}$ and choose $\mu = 5.0$ according to results on the `dev` set. We implement our model with the open-source toolkit *fairseq* [24]. All models are trained on 4 RTX 3090 GPUs.

In the multilingual experiment, the presence of languages with limited data or substantial interlingual variations makes the mapping from source speech to target phonemes particularly challenging. To address this, we adopt a two-stage pretraining strategy. Initially, we pretrain the speech encoder and

[6]See VCTK_V1 in `https://github.com/jik876/hifi-gan`.

Table 1: Results on CVSS-C Fr→En and CVSS-T Fr→En `test` sets. ♣ indicates results quoted from Huang et al. [9]. ♠ indicates results of our re-implementation. †: target length beam=15 and noisy parallel decoding (NPD). T_{phone}, T_{unit}, and T_{mel} indicate the sequence length of phonemes, discrete units, and mel-spectrograms, respectively. ** means the improvements over S2UT are statistically significant ($p < 0.01$).

ID	Models	Decoding	#Iter	#Param	ASR-BLEU (Fr→En) CVSS-C	ASR-BLEU (Fr→En) CVSS-T	Speedup
	Ground Truth	/	/	/	84.52	81.48	/
	Single-pass autoregressive decoding						
A1	S2UT [5]	Beam=10	T_{unit}	73M	22.23	22.28	1.00×
A2	Translatotron [3]	Autoregressive	T_{mel}	79M	16.96	11.25	2.32×
	Two-pass autoregressive decoding						
B1	UnitY [7]	Beam=(10, 1)	$T_{\text{phone}} + T_{\text{unit}}$	64M	24.09	24.29	1.43×
B2	Translatotron 2 [6]	Beam=10	$T_{\text{phone}} + T_{\text{mel}}$	87M	**25.21**	**24.39**	1.42×
	Single-pass non-autoregressive decoding						
C1♣	TranSpeech [9]	Iteration	5	67M	17.24	/	11.04×
C2♣	+ b=15 + NPD†	Iteration	15	67M	18.39	/	2.53×
C3♠	TranSpeech [9]	Iteration	5	67M	16.38	16.49	12.45×
C4♠	+ b=15 + NPD†	Iteration	15	67M	19.05	18.60	3.35×
	Two-pass non-autoregressive decoding						
D1	**DASpeech**	Lookahead	$1+1$	93M	24.71**	24.45**	**18.53**×
D2	($\lambda = 0.5$)	Joint-Viterbi	$1+1$	93M	**25.03****	**25.26****	16.29×
D3	**DASpeech**	Lookahead	$1+1$	93M	24.41**	24.17**	18.45×
D4	($\lambda = 1.0$)	Joint-Viterbi	$1+1$	93M	24.80**	24.48**	15.65×
	Cascaded systems						
E1	S2T + FastSpeech 2	Beam=10	$T_{\text{phone}} + 1$	49M+41M	24.71	24.49	/
E2	DAT + FastSpeech 2	Lookahead	$1+1$	51M+41M	22.19	22.10	/
E3	($\lambda = 0.5$)	Joint-Viterbi	$1+1$	51M+41M	22.80	22.75	/
E4	DAT + FastSpeech 2	Lookahead	$1+1$	51M+41M	22.68	22.57	/
E5	($\lambda = 1.0$)	Joint-Viterbi	$1+1$	51M+41M	23.20	23.15	/

the linguistic decoder using the speech-to-subword task, followed by pretraining on the speech-to-phoneme task. In the second stage of pretraining, the embedding and output projection matrices of the decoder are replaced and trained from scratch to accommodate changes in the vocabulary. We employ this pretraining strategy for DASpeech, UnitY and Translatotron 2 in the multilingual experiment. We learn the subword vocabulary with a size of 6K using the SentencePiece toolkit.

We also adopt the glancing strategy [25] during training, which shows effectiveness in alleviating the multi-modality problem for NAT. It first assigns target tokens to appropriate vertices following the most probable path $\hat{A} = \arg\max_{A \in \Gamma} P_\theta(Y, A|X)$, and then masks some tokens. We linearly anneal the unmasking ratio τ from 0.5 to 0.1 during pretraining and fix τ to 0.1 during finetuning.

Evaluation During finetuning, we save checkpoints every 2000 steps and average the last 5 checkpoints for evaluation. We use the open-source ASR-BLEU toolkit[7] to evaluate the translation quality. The translated speech is first transcribed into text using a pretrained ASR model. SacreBLEU [26] is then used to compute the BLEU score [27] and the statistical significance of translation results. The decoding speedup is measured on the `test` set using 1 RTX 3090 GPU with a batch size of 1.

Baseline Systems We implement the following baseline systems for comparison. More details about the model architectures and hyperparameters can be found in Appendix A.

- **S2UT** [5] Speech-to-unit translation (S2UT) model generates discrete units corresponding to the target speech with a sequence-to-sequence model. We introduce the auxiliary task of predicting target phonemes to help the model converge.

[7] `https://github.com/facebookresearch/fairseq/tree/ust/examples/speech_to_speech/asr_bleu`

- **Translatotron** [3] Translatotron generates the target mel-spectrogram with a sequence-to-sequence model. We also introduce the auxiliary task of predicting the target phonemes.
- **UnitY** [7] UnitY is a two-pass model which generates target phonemes and discrete units successively[8]. We remove the R-Drop training [28] for simplification. We pretrain the speech encoder and first-pass decoder on the S2TT task.
- **Translatotron 2** [6] Translatotron 2 is a two-pass model which generates target phonemes and mel-spectrograms successively. We enhance Translatotron 2 by replacing LSTM with Transformer, and introducing an additional encoder between two decoders following Inaguma et al. [7]. The speech encoder and first-pass decoder are pretrained on the S2TT task.
- **TranSpeech** [9] TranSpeech is the first non-autoregressive S2ST model that generates target discrete units in parallel. To alleviate the acoustic multi-modality problem, TranSpeech introduces bilateral perturbation (BiP) to disentangle the acoustic variations from the discrete units. We re-implement TranSpeech following the configurations in the original paper.
- **S2T + FastSpeech 2** The cascaded system that combines an autoregressive S2TT model and FastSpeech 2. The S2T model contains 12 Conformer layers and 4 Transformer decoder layers, which is also used in UnitY and Translatotron 2 pretraining.
- **DAT + FastSpeech 2** The cascaded system that combines the S2TT DA-Transformer model and FastSpeech 2. Both models are used in DASpeech pretraining.

4.2 Main Results

Table 1 summarizes the results on the CVSS-C Fr→En and CVSS-T Fr→En datasets. **(1)** Compared with previous autoregressive models, DASpeech (`D1-D4`) obviously surpasses single-pass models (`A1`, `A2`) and achieves comparable or even better performance than two-pass models (`B1`, `B2`), while preserving up to 18.53 times decoding speedup compared to S2UT. **(2)** Compared with the previous NAR model TranSpeech (`C1-C4`), DASpeech does not rely on knowledge distillation and iterative decoding, achieving significant advantages in both translation quality and decoding speedup. **(3)** DASpeech obviously outperforms the corresponding cascaded systems (`D1-D4` vs. `E2-E5`), demonstrating the effectiveness of our expect-path training approach. We also find that the cascaded model prefers larger graph size ($\lambda = 1.0$ is better) while DASpeech prefers smaller graph size ($\lambda = 0.5$ is better). We think the reason is that a larger graph size can improve S2TT performance, but it also makes end-to-end training more challenging. We further study the effects of the graph size in Appendix C. **(4)** On the CVSS-T dataset, which includes target speeches from various speakers, we observe a performance degradation in Translatotron and Translatotron 2 as the target mel-spectrogram becomes more difficult to predict. In contrast, DASpeech still performs well since its acoustic decoder explicitly incorporates variation information to alleviate the acoustic multi-modality, demonstrating the potential of DASpeech in handling complex and diverse target speeches.

Table 2 shows the results on CVSS-C dataset of the multilingual X→En S2ST model. We report the average ASR-BLEU scores on all languages, as well as the average scores on high/middle/low-resource languages[9]. We find that DASpeech still obviously outperforms S2UT but performs slightly worse than Translatotron 2 and UnitY in the multilingual setting, with an average gap of about 1.3 ASR-BLEU compared to Translatotron 2. However, DASpeech has about 13 times decoding speedup compared to Translatotron 2, achieving a better quality-speed trade-off.

Table 2: ASR-BLEU scores on CVSS-C `test` sets of the multilingual X→En S2ST model.

Models		**Avg.**	**High**	**Mid**	**Low**
S2UT [5]		5.15	16.74	6.24	0.84
UnitY [7]		8.15	24.97	9.78	1.86
Translatotron 2 [6]		**8.74**	**25.92**	**11.07**	**2.04**
DASpeech ($\lambda = 0.5$)	+ Lookahead	7.42	22.84	9.51	1.41
	+ Joint-Viterbi	7.43	22.80	9.49	1.45

Table 3: ASR-BLEU scores on the CVSS-C Fr→En `test` set with best-path training and expect-path training.

Models		**Best**	**Expect**	**Δ**
DASpeech ($\lambda = 0.5$)	+ Lookahead	24.45	24.71	+0.26
	+ Joint-Viterbi	24.84	25.03	+0.19
DASpeech ($\lambda = 1.0$)	+ Lookahead	24.18	24.41	+0.23
	+ Joint-Viterbi	24.46	24.80	+0.34

[8]Note that the original UnitY uses subwords instead of phonemes. Here we use phonemes just for consistency with other systems.

[9]The detailed results of each language pair can be found in Table 6 in Appendix B.

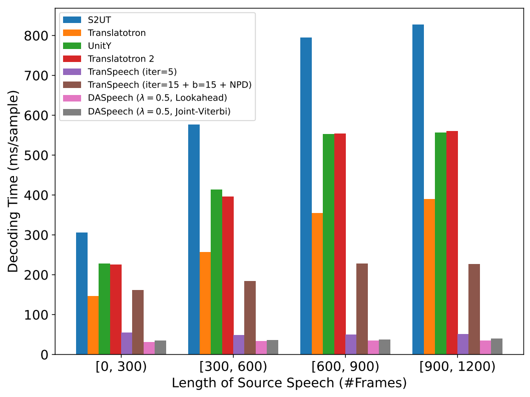


Figure 2: Translation latency of different models across different source speech lengths, categorized into 4 groups based on source speech frame counts. The translation latency is computed as the decoding time on 1 RTX 3090 GPU.

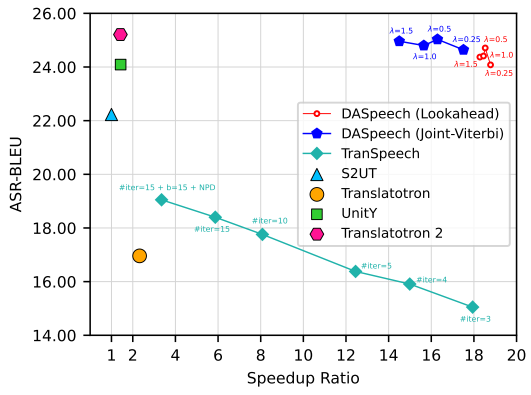


Figure 3: Trade-off between translation quality and decoding speed. X-axis represents the speedup ratio relative to S2UT, and Y-axis represents ASR-BLEU. The upper right represents better trade-off.

4.3 Alternative Training Approach: Best-Path Training

In addition to the expect-path training approach proposed in Section 3.2, we also experiment with a simpler approach: *Best-Path Training*. The core idea is to select the most probable path $\hat{A} = \arg\max_{A\in\Gamma} P_\theta(Y, A|X)$ via Viterbi algorithm [20], and take the hidden states on path $\hat{A} = (\hat{a}_1, ..., \hat{a}_M)$ as input to the acoustic decoder, i.e., $\mathbf{z}_i = \mathbf{v}_{\hat{a}_i}$. As shown in Table 3, best-path training also performs well but is inferior to expect-path training. We attribute this to the fact that when using best-path training, only hidden states on the most probable path participate in TTS training, which may result in insufficient training for the remaining hidden states. In contrast, all hidden states participate in TTS training with our expect-path training, which achieves better performance. The time complexity of Viterbi algorithm used in the best-path training is also $\mathcal{O}(ML^2)$. More details about the best-path training can be found in Appendix E.

Table 4: Average speaker similarity on CVSS-T Fr→En `test` set.

Models		**Speaker Similarity**
Ground Truth		0.48
Unit-based S2ST		
S2UT [5]		0.03
UnitY [7]		0.03
TranSpeech [9]		0.03
Mel-spectrogram-based S2ST		
Translatotron [3]		0.04
Translatotron 2 [6]		0.05
DASpeech ($\lambda = 0.5$)	+ Lookahead	**0.14**
	+ Joint-Viterbi	0.10

4.4 Analysis of Decoding Speed

In this section, we provide more detailed analysis of decoding speed. Figure 2 shows the translation latency of different models for speech inputs of different lengths. The results indicate that the translation latency of autoregressive models (S2UT, Translatotron, UnitY, and Translatotron 2) significantly increases with the length of the source speech. In contrast, the translation latency of non-autoregressive models (TranSpeech and DASpeech) are hardly affected by the source speech length. When translating longer speech inputs, DASpeech's decoding speed can reach more than 20 times that of S2UT. Furthermore, we illustrate the quality-speed trade-off of different models in Figure 3. By adjusting the hyperparameter λ, the translation quality and decoding latency will change. Specifically, as λ increases, the decoding latency of the model will increase, and it achieves the best translation quality when $\lambda = 0.5$. It is evident that DASpeech achieves the best trade-off between translation quality and decoding latency among all models. We further study the speedup under batch decoding in Appendix D.

4.5 Voice Preservation

In this section, we investigate the voice preservation ability of direct S2ST models on the CVSS-T Fr→En dataset, where target speeches are in voices transferred from source speeches. Specifically, we use a pretrained speaker verification model[10] [29] to extract the speaker embedding of the source

[10] `https://github.com/yistLin/dvector`

speech and generated target speech. We define the cosine similarity between source and target speaker embeddings as *speaker similarity*, and report the average speaker similarity on the `test` set in Table 4. We find that: **(1)** unit-based S2ST model can not preserve the speaker's voice since discrete units contain little speaker information; and **(2)** DASpeech can better preserve the speaker's voice than Translatotron and Translatotron 2, since its acoustic decoder explicitly introduces variation information of the target speech, allowing the model to learn more complex target distribution.

5 Related Work

Direct Speech-to-Speech Translation Speech-to-speech translation (S2ST) extends speech-to-text translation [30–33] which further synthesizes the target speech. Translatotron [3] is the first S2ST model that directly generates target mel-spectrograms from the source speech. Since continuous speech features contain a lot of variance information that makes training challenging, Tjandra et al. [34], Zhang et al. [35] use the discrete tokens derived from a VQ-VAE model [36] as the target. Lee et al. [5, 37] extend this research line by leveraging discrete units derived from the pretrained HuBERT model [38] as the target. To further reduce the learning difficulty, Inaguma et al. [7], Jia et al. [6], Chen et al. [39] introduce a two-pass architecture that generates target text and target speech successively. To address the data scarcity issue, some techniques like pretraining and data augmentation are used to enhance S2ST [8, 40–44]. Huang et al. [9] proposes the first non-autoregressive S2ST model which achieves faster decoding speed. Our DASpeech extends this line of research and achieves better translation quality and faster decoding speed.

Non-autoregressive Machine Translation Machine translation based on autoregressive decoding usually has a high decoding latency [45]. Gu et al. [10] first proposes NAT for faster decoding speed. To alleviate the multi-modality problem in NAT, many approaches have been proposed [46] like knowledge distillation [47–49], latent-variable models [50, 51], learning latent alignments [52–56], sequence-level training [57, 58], and curriculum learning [25]. Recently, Huang et al. [11] introduce DA-Transformer, which models different translations with DAG to alleviate the multi-modality problem, achieving competitive results with autoregressive models. Ma et al. [59], Gui et al. [60] further enhance DA-Transformer with fuzzy alignment and probabilistic context-free grammar.

Non-autoregressive Text-to-Speech Ren et al. [61], Peng et al. [62] first propose non-autoregressive TTS that generates mel-spectrograms in parallel. FastSpeech 2 [12] explicitly models variance information to alleviate the issue of acoustic multi-modality. Many subsequent works enhance non-autoregressive TTS with more powerful generative models like variational auto-encoder (VAE) [63, 64], normalizing flows [65–67], and denoising diffusion probabilistic models (DDPM) [68, 69]. DASpeech adopts the design of FastSpeech 2 for training stability and good voice quality.

6 Conclusion

In this paper, we introduce DASpeech, a non-autoregressive two-pass direct S2ST model. DASpeech is built upon DA-Transformer and FastSpeech 2, and we propose an expect-path training approach to train the model end-to-end. DASpeech achieves comparable or even better performance than the state-of-the-art S2ST model Translatotron 2, while maintaining up to 18.53× speedup compared to the autoregressive model. DASpeech also significantly outperforms previous non-autoregressive model in both translation quality and decoding speed. In the future, we will investigate how to enhance DASpeech using techniques like pretraining and data augmentation.

7 Limitations & Broader Impacts

Limitations Although DASpeech achieves impressive performance in both translation quality and decoding speed, it still has some limitations: (1) the translation quality of DASpeech still lags behind Translatotron 2 in the multilingual setting; (2) the training cost of DASpeech is higher than Translatotron 2 (96 vs. 18 GPU hours) since it requires dynamic programming during training; and (3) the outputs of DASpeech are not always reliable, especially for some low-resource languages.

Broader Impacts In our experiments, we find that DASpeech emerges with the ability to maintain the speaker identity during translation. It raises potential risks in terms of model misuse, such as mimicking a particular speaker or voice identification spoofing.

Acknowledgements

We thank all the anonymous reviewers for their insightful and valuable comments.

References

[1] A. Lavie, A. Waibel, L. Levin, M. Finke, D. Gates, M. Gavalda, T. Zeppenfeld, and Puming Zhan. Janus-iii: speech-to-speech translation in multiple languages. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 99–102 vol.1, 1997. doi: 10.1109/ICASSP.1997.599557.

[2] S. Nakamura, K. Markov, H. Nakaiwa, G. Kikui, H. Kawai, T. Jitsuhiro, J.-S. Zhang, H. Yamamoto, E. Sumita, and S. Yamamoto. The atr multilingual speech-to-speech translation system. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(2):365–376, 2006. doi: 10.1109/TSA.2005.860774.

[3] Ye Jia, Ron J. Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. Direct speech-to-speech translation with a sequence-to-sequence model. In Gernot Kubin and Zdravko Kacic, editors, *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 1123–1127. ISCA, 2019. doi: 10.21437/Interspeech.2019-1951. URL `https://doi.org/10.21437/Interspeech.2019-1951`.

[4] Ye Jia, Michelle Tadmor Ramanovich, Quan Wang, and Heiga Zen. CVSS corpus and massively multilingual speech-to-speech translation. In *Proceedings of Language Resources and Evaluation Conference (LREC)*, pages 6691–6703, 2022.

[5] Ann Lee, Peng-Jen Chen, Changhan Wang, Jiatao Gu, Sravya Popuri, Xutai Ma, Adam Polyak, Yossi Adi, Qing He, Yun Tang, Juan Pino, and Wei-Ning Hsu. Direct speech-to-speech translation with discrete units. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3327–3339, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.235. URL `https://aclanthology.org/2022.acl-long.235`.

[6] Ye Jia, Michelle Tadmor Ramanovich, Tal Remez, and Roi Pomerantz. Translatotron 2: High-quality direct speech-to-speech translation with voice preservation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 10120–10134. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/jia22b.html`.

[7] Hirofumi Inaguma, Sravya Popuri, Ilia Kulikov, Peng-Jen Chen, Changhan Wang, Yu-An Chung, Yun Tang, Ann Lee, Shinji Watanabe, and Juan Pino. Unity: Two-pass direct speech-to-speech translation with discrete units, 2022.

[8] Sravya Popuri, Peng-Jen Chen, Changhan Wang, Juan Pino, Yossi Adi, Jiatao Gu, Wei-Ning Hsu, and Ann Lee. Enhanced direct speech-to-speech translation using self-supervised pre-training and data augmentation. In Hanseok Ko and John H. L. Hansen, editors, *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 5195–5199. ISCA, 2022. doi: 10.21437/Interspeech.2022-11032. URL `https://doi.org/10.21437/Interspeech.2022-11032`.

[9] Rongjie Huang, Jinglin Liu, Huadai Liu, Yi Ren, Lichao Zhang, Jinzheng He, and Zhou Zhao. Transpeech: Speech-to-speech translation with bilateral perturbation. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=UVAmFAtC5ye`.

[10] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=B1l8BtlCb`.

[11] Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. Directed acyclic transformer for non-autoregressive machine translation. In *Proceedings of the 39th International Conference on Machine Learning, ICML 2022*, 2022.

[12] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=piLPYqxtWuA`.

[13] Fei Huang, Pei Ke, and Minlie Huang. Directed acyclic transformer pre-training for high-quality non-autoregressive text generation. *Transactions of the Association for Computational Linguistics*, 2023.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[15] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040, 2020. doi: 10.21437/Interspeech.2020-3015.

[16] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL `https://aclanthology.org/P19-1285`.

[17] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi: 10.1109/5.18626.

[18] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[19] Chenze Shao, Zhengrui Ma, and Yang Feng. Viterbi decoding of directed acyclic transformer for non-autoregressive machine translation. In *Findings of EMNLP 2022*, 2022.

[20] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967. doi: 10.1109/TIT.1967.1054010.

[21] Changhan Wang, Anne Wu, and Juan Pino. Covost 2: A massively multilingual speech-to-text translation corpus, 2020.

[22] Christophe Veaux, Junichi Yamagishi, and Kirsten MacDonald. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. 2017.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

[24] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[25] Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. Glancing transformer for non-autoregressive neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International*

Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1993–2003, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.155. URL https://aclanthology.org/2021.acl-long.155.

[26] Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6319. URL https://www.aclweb.org/anthology/W18-6319.

[27] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

[28] xiaobo liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. R-drop: Regularized dropout for neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10890–10905. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/5a66b9200f29ac3fa0ae244cc2a51b39-Paper.pdf.

[29] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, page 4879–4883. IEEE Press, 2018. doi: 10.1109/ICASSP.2018.8462665. URL https://doi.org/10.1109/ICASSP.2018.8462665.

[30] Qingkai Fang, Rong Ye, Lei Li, Yang Feng, and Mingxuan Wang. Stemm: Self-learning with speech-text manifold mixup for speech translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

[31] Qingkai Fang and Yang Feng. Back translation for speech-to-text translation without transcripts. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

[32] Qingkai Fang and Yang Feng. Understanding and bridging the modality gap for speech translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

[33] Yan Zhou, Qingkai Fang, and Yang Feng. CMOT: Cross-modal mixup via optimal transport for speech translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7873–7887, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.436. URL https://aclanthology.org/2023.acl-long.436.

[34] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Speech-to-speech translation between untranscribed unknown languages. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 593–600, 2019. doi: 10.1109/ASRU46091.2019.9003853.

[35] Chen Zhang, Xu Tan, Yi Ren, Tao Qin, Kejun Zhang, and Tie-Yan Liu. Uwspeech: Speech to speech translation for unwritten languages. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14319–14327, May 2021. doi: 10.1609/aaai.v35i16.17684. URL https://ojs.aaai.org/index.php/AAAI/article/view/17684.

[36] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

[37] Ann Lee, Hongyu Gong, Paul-Ambroise Duquenne, Holger Schwenk, Peng-Jen Chen, Changhan Wang, Sravya Popuri, Yossi Adi, Juan Pino, Jiatao Gu, and Wei-Ning Hsu. Textless speech-to-speech translation on real data. In *Proceedings of the 2022 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 860–872, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.63. URL https://aclanthology.org/2022.naacl-main.63.

[38] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29: 3451–3460, oct 2021. ISSN 2329-9290. doi: 10.1109/TASLP.2021.3122291. URL https://doi.org/10.1109/TASLP.2021.3122291.

[39] Peng-Jen Chen, Kevin Tran, Yilin Yang, Jingfei Du, Justine Kao, Yu-An Chung, Paden Tomasello, Paul-Ambroise Duquenne, Holger Schwenk, Hongyu Gong, Hirofumi Inaguma, Sravya Popuri, Changhan Wang, Juan Miguel Pino, Wei-Ning Hsu, and Ann Lee. Speech-to-speech translation for A real-world unwritten language. *CoRR*, abs/2211.06474, 2022. doi: 10.48550/arXiv.2211.06474. URL https://doi.org/10.48550/arXiv.2211.06474.

[40] Ye Jia, Yifan Ding, Ankur Bapna, Colin Cherry, Yu Zhang, Alexis Conneau, and Nobu Morioka. Leveraging unsupervised and weakly-supervised data to improve direct speech-to-speech translation. In Hanseok Ko and John H. L. Hansen, editors, *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 1721–1725. ISCA, 2022. doi: 10.21437/Interspeech.2022-10938. URL https://doi.org/10.21437/Interspeech.2022-10938.

[41] Qianqian Dong, Fengpeng Yue, Tom Ko, Mingxuan Wang, Qibing Bai, and Yu Zhang. Leveraging pseudo-labeled data to improve direct speech-to-speech translation. In Hanseok Ko and John H. L. Hansen, editors, *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 1781–1785. ISCA, 2022. doi: 10.21437/Interspeech.2022-10011. URL https://doi.org/10.21437/Interspeech.2022-10011.

[42] Xuan-Phi Nguyen, Sravya Popuri, Changhan Wang, Yun Tang, Ilia Kulikov, and Hongyu Gong. Improving speech-to-speech translation through unlabeled text. *arXiv preprint arXiv:2210.14514*, 2022.

[43] Kun Wei, Long Zhou, Ziqiang Zhang, Liping Chen, Shujie Liu, Lei He, Jinyu Li, and Furu Wei. Joint pre-training with speech and bilingual text for direct speech to speech translation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10095616.

[44] Ziqiang Zhang, Long Zhou, Chengyi Wang, Sanyuan Chen, Yu Wu, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. Speak foreign languages with your own voice: Cross-lingual neural codec language modeling. *CoRR*, abs/2303.03926, 2023. doi: 10.48550/arXiv.2303.03926. URL https://doi.org/10.48550/arXiv.2303.03926.

[45] Shaolei Zhang, Qingkai Fang, Zhuocheng Zhang, Zhengrui Ma, Yan Zhou, Langlin Huang, Mengyu Bu, Shangtong Gui, Yunji Chen, Xilin Chen, and Yang Feng. Bayling: Bridging cross-lingual alignment and instruction following through interactive translation for large language models, 2023.

[46] Jiatao Gu and Xiang Kong. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.11. URL https://aclanthology.org/2021.findings-acl.11.

[47] Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139.

[48] Chunting Zhou, Jiatao Gu, and Graham Neubig. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BygFVAEKDH.

[49] Chenze Shao, Xuanfu Wu, and Yang Feng. One reference is not enough: Diverse distillation with reference selection for non-autoregressive translation. In *Proceedings of NAACL 2022*, 2022.

[50] Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8846–8853. AAAI Press, 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/6413.

[51] Yu Bao, Hao Zhou, Shujian Huang, Dongqi Wang, Lihua Qian, Xinyu Dai, Jiajun Chen, and Lei Li. latent-glat: Glancing at latent variables for parallel text generation. In *ACL*. 2022. URL https://arxiv.org/abs/2204.02030.

[52] Jindřich Libovický and Jindřich Helcl. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1336. URL https://aclanthology.org/D18-1336.

[53] Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.83. URL https://aclanthology.org/2020.emnlp-main.83.

[54] Cunxiao Du, Zhaopeng Tu, and Jing Jiang. Order-agnostic cross entropy for non-autoregressive machine translation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2849–2859. PMLR, 2021. URL http://proceedings.mlr.press/v139/du21c.html.

[55] Chenze Shao and Yang Feng. Non-monotonic latent alignments for CTC-based non-autoregressive machine translation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Qvh0SAPrYzH.

[56] Zhengrui Ma, Shaolei Zhang, Shoutao Guo, Chenze Shao, Min Zhang, and Yang Feng. Non-autoregressive streaming transformer for simultaneous translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[57] Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):198–205, Apr. 2020. doi: 10.1609/aaai.v34i01.5351. URL https://ojs.aaai.org/index.php/AAAI/article/view/5351.

[58] Chenze Shao, Yang Feng, Jinchao Zhang, Fandong Meng, and Jie Zhou. Sequence-level training for non-autoregressive neural machine translation. *Computational Linguistics*, 47(4):891–925, December 2021. doi: 10.1162/coli_a_00421. URL https://aclanthology.org/2021.cl-4.29.

[59] Zhengrui Ma, Chenze Shao, Shangtong Gui, Min Zhang, and Yang Feng. Fuzzy alignments in directed acyclic graph for non-autoregressive machine translation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=LSz-gQyd0zE.

[60] Shangtong Gui, Chenze Shao, Zhengrui Ma, Xishan Zhang, Yunji Chen, and Yang Feng. Non-autoregressive machine translation with probabilistic context-free grammar. In *Advances in Neural Information Processing Systems*, 2023.

[61] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3165–3174, 2019. URL `https://proceedings.neurips.cc/paper/2019/hash/f63f65b503e22cb970527f23c9ad7db1-Abstract.html`.

[62] Kainan Peng, Wei Ping, Zhao Song, and Kexin Zhao. Non-autoregressive neural text-to-speech. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7586–7598. PMLR, 2020. URL `http://proceedings.mlr.press/v119/peng20a.html`.

[63] Yoonhyung Lee, Joongbo Shin, and Kyomin Jung. Bidirectional variational inference for non-autoregressive text-to-speech. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=o3iritJHLfO`.

[64] Xu Tan, Jiawei Chen, Haohe Liu, Jian Cong, Chen Zhang, Yanqing Liu, Xi Wang, Yichong Leng, Yuanhao Yi, Lei He, Frank K. Soong, Tao Qin, Sheng Zhao, and Tie-Yan Liu. Naturalspeech: End-to-end text to speech synthesis with human-level quality. *CoRR*, abs/2205.04421, 2022. doi: 10.48550/arXiv.2205.04421. URL `https://doi.org/10.48550/arXiv.2205.04421`.

[65] Chenfeng Miao, Shuang Liang, Minchuan Chen, Jun Ma, Shaojun Wang, and Jing Xiao. Flow-tts: A non-autoregressive network for text to speech based on flow. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 7209–7213. IEEE, 2020. doi: 10.1109/ICASSP40776.2020.9054484. URL `https://doi.org/10.1109/ICASSP40776.2020.9054484`.

[66] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[67] Yi Ren, Jinglin Liu, and Zhou Zhao. Portaspeech: Portable and high-quality generative text-to-speech. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 13963–13974, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/748d6b6ed8e13f857ceaa6cfbdca14b8-Abstract.html`.

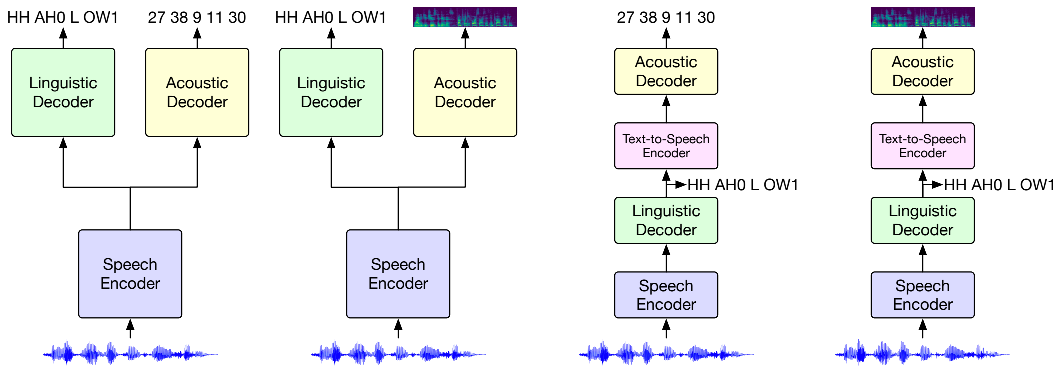
[68] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail A. Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8599–8608. PMLR, 2021. URL `http://proceedings.mlr.press/v139/popov21a.html`.

[69] Rongjie Huang, Max W. Y. Lam, Jun Wang, Dan Su, Dong Yu, Yi Ren, and Zhou Zhao. Fastdiff: A fast conditional diffusion model for high-quality speech synthesis. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4157–4163. ijcai.org, 2022. doi: 10.24963/ijcai.2022/577. URL `https://doi.org/10.24963/ijcai.2022/577`.

A Details of Baseline Models

In our experiments, we implement five baseline systems using *fairseq*: S2UT, Translatotron, UnitY, Translatotron 2, and TranSpeech. We reproduce TranSpeech with their open-source implementations[11]. In this section, we mainly introduce the configurations of the other four baseline systems.

Figure 4 shows the model architectures of these models. In terms of model architecture, S2UT and Translatotron are single-pass S2ST models while UnitY and Translatotron 2 are two-pass S2ST models. In terms of predicted targets, S2UT and UnitY predict discrete units while Translatotron and Translatotron 2 predict mel-spectrograms. Below we describe the details of each model. The detailed hyperparameters can be found in Table 5.



(a) S2UT (b) Translatotron (c) UnitY (d) Translatotron 2

Figure 4: Overview of baseline models.

S2UT Our implemented S2UT model includes three parts: a speech encoder, a linguistic decoder, and an acoustic decoder. The speech encoder is the same as DASpeech. The linguistic decoder is appended to the top layer of the speech encoder for multi-task learning, which predicts the target phonemes during training. The acoustic decoder generates the reduced discrete units derived from the 11-th layer of the pretrained mHuBERT model[12]. We do not include other auxiliary tasks and remove CTC decoding in Lee et al. [5] for simplification. The model is trained from scratch for 100k steps. We use beam search with a beam size of 10.

Translatotron The speech encoder and linguistic decoder of Translatotron are the same as S2UT. The acoustic decoder generates mel-spectrograms autoregressively. The pre-net dimension is 32 and the reduction factor of the acoustic decoder is 5. The model is trained from scratch for 100k steps.

UnitY UnitY is a two-pass model that includes four parts: a speech encoder, a linguistic decoder, a text-to-speech encoder, and an acoustic decoder. The architecture of the speech encoder, linguistic decoder, and acoustic decoder are the same as S2UT. The additional text-to-speech encoder is used to bridge the gap in representations between two decoders. We remove R-Drop training for simplification. We first conduct S2TT pretraining and finetune the model for 50k steps. We set the beam size of the first-pass and second-pass decoder to 10 and 1, respectively.

Translatotron 2 The model architecture of Translatotron 2 is similar to UnitY except that the second decoder generates mel-spectrograms rather than discrete units. The reduction factor of the acoustic decoder is set to 5. We first conduct S2TT pretraining and finetune the model for 50k steps. The beam size is set to 10 for the first-pass decoder.

For all the above models, we save checkpoints every 2000 steps and average the last 5 checkpoints for evaluation, which is the same as DASpeech. For S2UT and UnitY, we use the pretrained unit-based HiFi-GAN[13] vocoder to synthesize waveform. For Translatotron and Translatotron 2, we use the same pretrained HiFi-GAN vocoder as DASpeech.

[11]`https://github.com/Rongjiehuang/TranSpeech`

[12]`https://dl.fbaipublicfiles.com/hubert/mhubert_base_vp_en_es_fr_it3_L11_km1000.bin`

[13]`https://dl.fbaipublicfiles.com/fairseq/speech_to_speech/vocoder/code_hifigan/mhubert_vp_en_es_fr_it3_400k_layer11_km1000_lj/g_00500000`

Table 5: Hyperparameters of DASpeech and baseline models.

Hyperparameters		**S2UT**	**Translatotron**	**UnitY**	**Translatotron 2**	**DASpeech**
Speech Encoder	conv_kernel_sizes	(5, 5)	(5, 5)	(5, 5)	(5, 5)	(5, 5)
	encoder_type	conformer	conformer	conformer	conformer	conformer
	encoder_layers	12	12	12	12	12
	encoder_embed_dim	256	256	256	256	256
	encoder_ffn_embed_dim	2048	2048	2048	2048	2048
	encoder_attention_heads	4	4	4	4	4
	encoder_pos_enc_type	relative	relative	relative	relative	relative
	depthwise_conv_kernel_size	31	31	31	31	31
Linguistic Decoder	decoder_layers	4	4	4	4	4
	decoder_embed_dim	512	512	512	512	512
	decoder_ffn_embed_dim	2048	2048	2048	2048	2048
	decoder_attention_heads	8	8	8	8	8
	label_smoothing	0.1	0.1	0.1	0.1	0.0
	s2t_loss_weight	8.0	0.1	8.0	0.1	1.0
Text-to-Speech Encoder	encoder_layers	-	-	2	2	-
	encoder_embed_dim	-	-	512	512	-
	encoder_ffn_embed_dim	-	-	2048	2048	-
	encoder_attention_heads	-	-	8	8	-
Acoustic Decoder	decoder_layers	6	6	2	6	8
	decoder_embed_dim	512	512	512	512	256
	decoder_ffn_embed_dim	2048	2048	2048	2048	1024
	decoder_attention_heads	8	8	8	8	4
	label_smoothing	0.1	-	0.1	-	-
	n_frames_per_step	1	5	1	5	1
	unit_dictionary_size	1000	-	1000	-	-
	var_pred_hidden_dim	-	-	-	-	256
	var_pred_kernel_size	-	-	-	-	3
	var_pred_dropout	-	-	-	-	0.5
	s2s_loss_weight	1.0	1.0	1.0	1.0	5.0
Training	lr	1e-3	1e-3	1e-3	1e-3	1e-3
	lr_scheduler	inverse_sqrt	inverse_sqrt	inverse_sqrt	inverse_sqrt	inverse_sqrt
	warmup_updates	4000	4000	4000	4000	4000
	warmup_init_lr	1e-7	1e-7	1e-7	1e-7	1e-7
	optimizer	Adam	Adam	Adam	Adam	Adam
	dropout	0.1	0.1	0.1	0.1	0.1
	max_tokens	40k×4	40k×4	40k×4	40k×4	40k×8
	weight_decay	0.0	0.0	0.0	0.0	0.01
	clip_norm	1.0	1.0	1.0	1.0	1.0
	max_update	100k	100k	50k	50k	50k

B Detailed Results on CVSS-C X→En Datasets

Table 6 summarizes the detailed results of each language pair on CVSS-C `test` sets of the multilingual X→En S2ST models.

Table 6: Results on CVSS-C `test` sets of the multilingual X→En S2ST models.

Models		Avg.	High				Mid				
			Fr	De	Ca	Es	Fa	It	Ru	Zh	Pt
S2UT [5]		5.15	19.65	13.35	15.37	18.58	1.43	14.47	7.94	0.93	6.42
UnitY [7]		8.15	27.27	20.81	24.22	27.58	3.63	21.68	10.86	4.16	8.56
Translatotron 2 [6]		8.74	28.04	21.54	25.34	28.77	4.23	23.66	13.41	4.49	9.54
DASpeech	+ Lookahead	7.42	25.43	17.87	22.58	25.49	3.01	20.80	12.96	2.86	7.90
($\lambda = 0.5$)	+ Joint-Viterbi	7.43	25.39	18.36	22.33	25.10	2.81	20.76	12.94	3.05	7.89

Models		Low											
		Nl	Tr	Et	Mn	Ar	Lv	Sl	Sv	Cy	Ta	Ja	Id
S2UT [5]		4.67	0.52	0.36	0.14	0.56	0.39	0.73	1.28	0.66	0.17	0.20	0.38
UnitY [7]		10.60	3.79	1.07	0.12	0.78	1.50	0.81	1.38	1.74	0.10	0.15	0.27
Translatotron 2 [6]		11.17	4.58	1.12	0.32	1.35	1.37	0.93	1.49	1.50	0.10	0.22	0.33
DASpeech	+ Lookahead	9.04	1.75	0.04	0.08	0.64	1.43	1.20	1.33	0.70	0.09	0.29	0.29
($\lambda = 0.5$)	+ Joint-Viterbi	9.43	1.66	0.07	0.08	0.48	1.48	1.30	1.30	0.85	0.09	0.31	0.32

C Effects of the Graph Size

In this section, we investigate how the graph size affects the performance. We vary the size factor λ from 0.25 to 1.5, and measure the translation quality of both the S2TT DA-Transformer model and DASpeech on the CVSS-C Fr→En `test` set. As shown in Figures 5 and 6, we observe that the performance of S2TT DA-Transformer keeps increasing as the graph size gets larger, which is consistent with the observations in machine translation [11, 59]. However, DASpeech performs best at $\lambda = 0.5$ and shows a performance drop at larger λ. We speculate that this is because larger graph size makes end-to-end training more challenging. We will investigate this issue in the future.

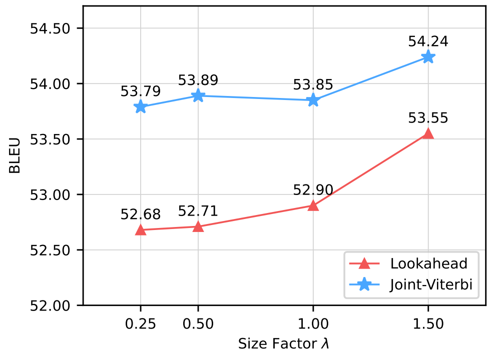


Figure 5: Phoneme-level BLEU scores of the S2TT DA-Transformer under different size factor λ.

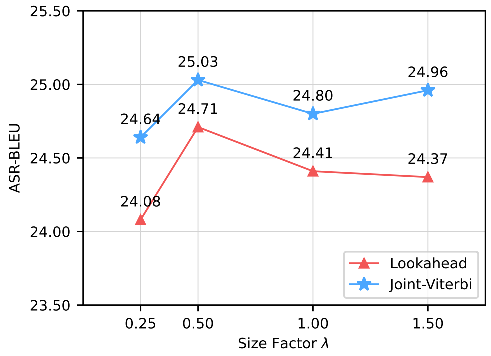


Figure 6: ASR-BLEU scores of DASpeech under different size factor λ.

D Speedup Under Batch Decoding

As Gu and Kong [46] pointed out, the speed benefits of non-autoregressive models may degrade during batch decoding. To better understand this problem, we evaluate the speedup ratio under different decoding batch sizes. As shown in Figure 7, the speedup ratio keeps dropping as the decoding batch size increases. Nevertheless, DASpeech ($\lambda = 0.5$ with Joint-Viterbi decoding) still achieves more than 6× speedup with a decoding batch size of 64 and maintains comparable performance with Translatotron 2.

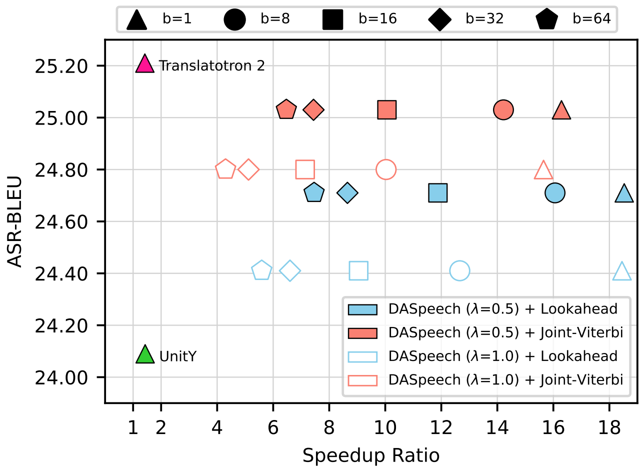


Figure 7: Speedup ratio compared to S2UT baseline (not shown in the figure) and ASR-BLEU score on the CVSS-C Fr→En `test` set with different batch decoding sizes ($b \in \{1, 8, 16, 32, 64\}$).

E Best-Path Training

Best-path-training selects the most probable path $\hat{A} = (\hat{a}_1, ..., \hat{a}_M)$ and takes the hidden states on path $\hat{A}$ as input to the acoustic decoder. Formally, given the target phoneme sequence Y, we can find the most probable path $\hat{A} = \arg\max_{A \in \Gamma} P_\theta(Y, A|X)$ via Viterbi algorithm [20]. Specifically, we use $\delta_i(j)$ to denote the probability of the most probable path so far $(\hat{a}_1, ..., \hat{a}_i)$ with $\hat{a}_i = j$ that generates $(y_1, ..., y_i)$. Considering the definition of $a_1 = 1$, we have $\delta_1(1) = \mathbf{P}_{1,y_1}$ and $\delta_1(1 < j \leq L) = 0$. For $i > 1$, we can sequentially calculate $\delta_i(\cdot)$ from its previous step $\delta_{i-1}(\cdot)$ due to the Markov property:

$$\delta_i(j) = \max_{k<j}(\delta_{i-1}(k) \cdot \mathbf{E}_{k,j} \cdot \mathbf{P}_{j,y_i}), \tag{16}$$

$$\phi_i(j) = \arg\max_{k<j}(\delta_{i-1}(k) \cdot \mathbf{E}_{k,j} \cdot \mathbf{P}_{j,y_i}), \tag{17}$$

where $\phi_i(j)$ stores $\hat{a}_{i-1}$ of the most probable path so far $(\hat{a}_1, ..., \hat{a}_{i-1}, \hat{a}_i = j)$. After M iterations, we can obtain the most probable path by backtracking from $\hat{a}_M = L$:

$$\hat{a}_i = \phi_{i+1}(\hat{a}_{i+1}). \tag{18}$$

Finally, we select the hidden states on the most probable path, i.e., $\mathbf{z}_i = \mathbf{v}_{\hat{a}_i}$, as the input sequence of the acoustic decoder.