[S2ST] DASpeech

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

Contents

1.Instruction

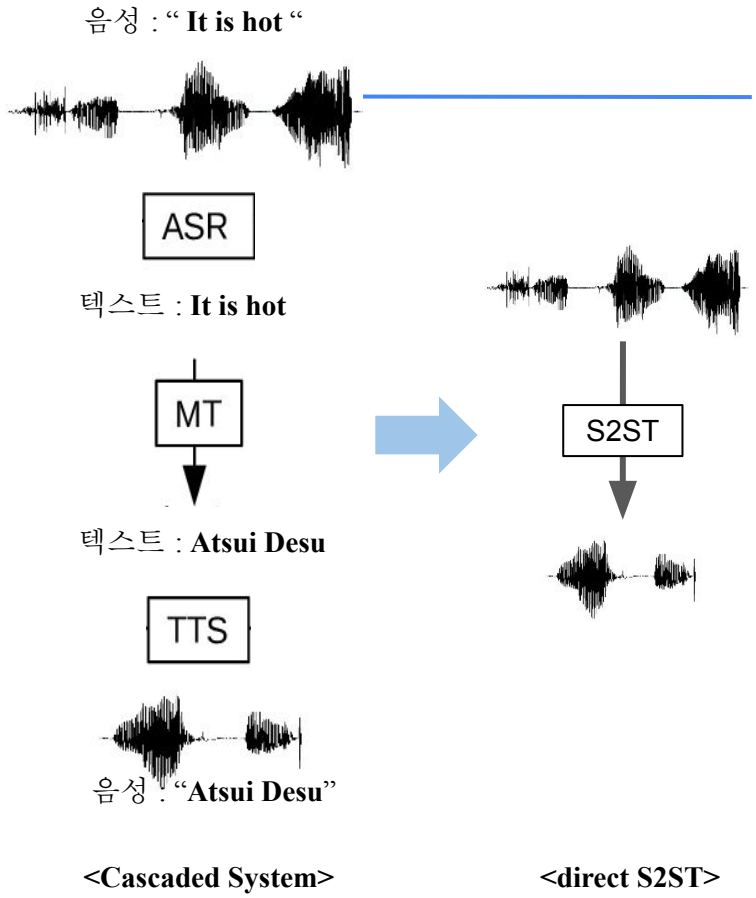
1. Introduction

* S2ST(Speech to Speech Translation) 연구의 동향)

전통적인 "**ASR** + **MT** + **TTS**" 의 Cascaded System 에서
=> direct S2ST 의 구조가 발표

* direct S2ST의 장점)

- 여러 개가 아닌 하나의 모델을 훈련하고 결과를 생성
- 비교적 디코딩 속도가 빠름
- 화자의 speech 특징, 단어의 음성 특징을 보존 가능



1. Introduction

- direct S2ST에서 해결해야할 문제 정의)

  direct S2ST를 훈련하는 과정에 대한 방법론+성능 개선
  복잡한 멀티 모달리티 (언어적 정보+어쿠스틱 정보(ex. duration, pitch, energy))

- 해결 제안)

(1) **Two-pass 구조** : 생성 프로세스를 **two step**으로 나누는 것
  a) linguistic decoder에서 target text 생성
  b) (hidden states)를 참고하여 acoustic decoder에서 target speech를 생성

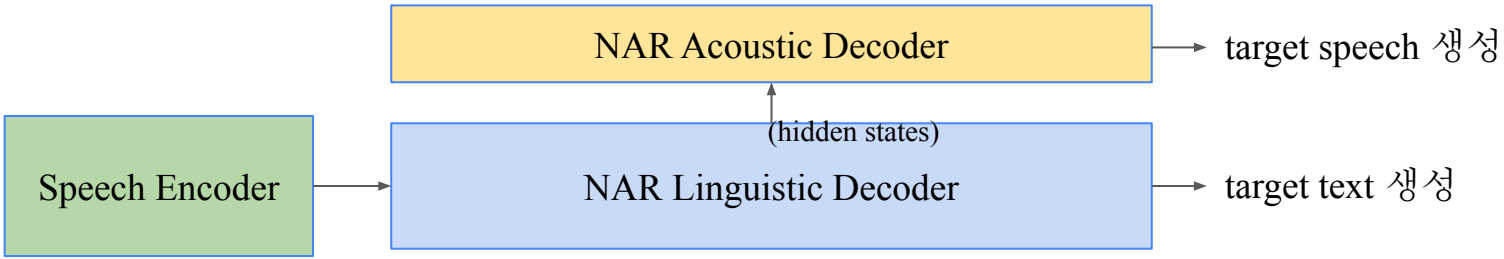
(2) **NAR S2ST** : target speech를 병렬적으로 생성
  => 디코딩 지연 시간 축소(단, AR모델에 비해 멀티모달리티를 잡기 어려워지는 점을 감안)

1. Introduction

- **Two pass** 구조란?
  - A speech encoder, A linguistic decoder, An acoustic decoder 세 개의 블록 구조를 가지는 구조

Linguistic decoder)
- DA(Directed Acyclic) - Transformer 구조를 적용(**DAG**)
- NAR(비자동회귀) 번역 모델로 병렬적으로 생성
- 다이나믹 프로그래밍으로 모든 가능한 구조를 연산

Acoustic Decoder)
- FastSpeech 2 구조를 적용
- Mel-spectrogram을 생성하는 NAR(비자동회귀) TTS 모델



1. Introduction

- **NAR S2ST** 의 병렬성이란?

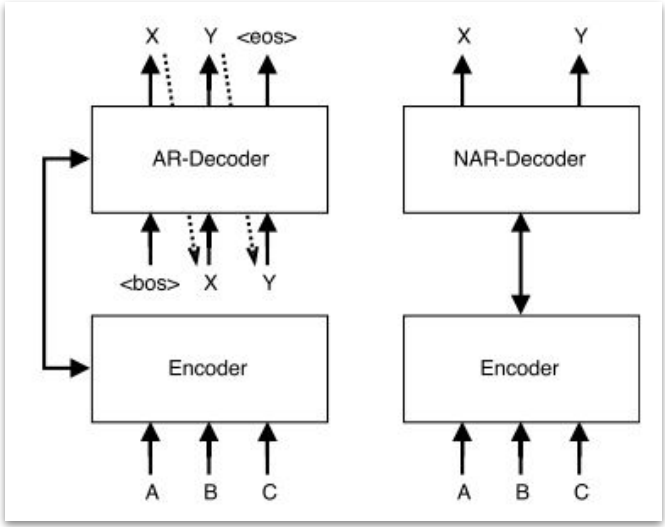
  cf. 기존에 사용하는 AR 모델과의 차이점

  pros) AR 모델은 큰 corpora에서 학습시키기 쉽고 beam search(ex. beam=5)로 적절한 output을 찾을 수 있음.

  cons) 이전 토큰이 다음 토큰에 영향을 주기 때문에 순차적으로 실행되어 오래 걸림.

=> Decoding 과정에서 병렬적(parallel)으로 토큰을 배치하기 때문에 디코딩 지연이 줄어듦.

=> 주로 Encoder는 transformer와 동일한 Encoder 구조 사용.



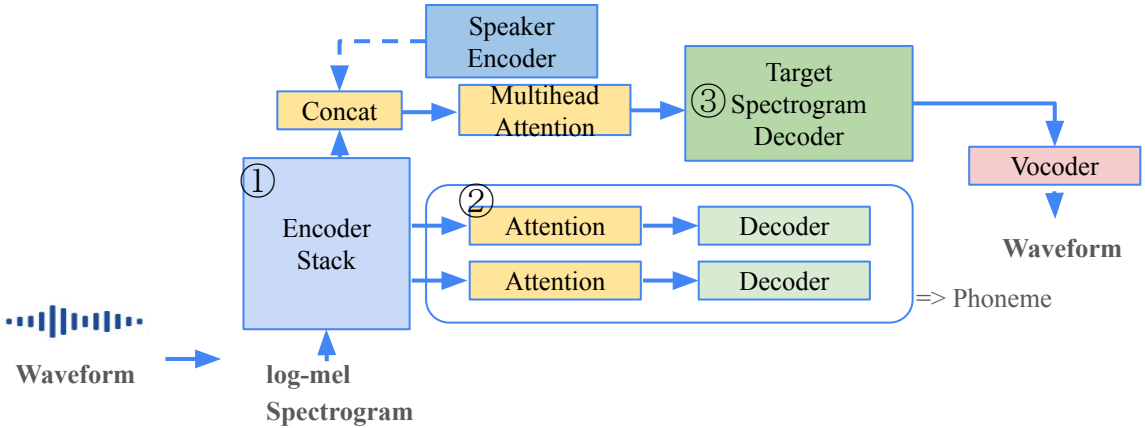
<좌> AR디코딩 <우> NAR디코딩

2. Background

2. Background

**DASpeech Model = “It is a Two-pass / Non-Autoregressive / direct S2ST model ”

- cf. direct S2ST(Translatotron)의 기본적인 구조
  : LSTM layer를 기반으로 한 Encoder / Decoder 구조



2. Background - baseline

- Baseline 목록

(a) S2UT
: mel-spectrogram 대신 자체 이산 (discrete units) 단위 활용

(b) Translatotron
: mel-spectrogram을 사용한 기본적인 S2ST 구조

(c) UnitY
: two-pass model로, target 음소와 음성에 대한 units을 생성

(d) Translatotron 2
: two-pass model로, target 음소와 음성에 대한 mel-spectrogram을 생성

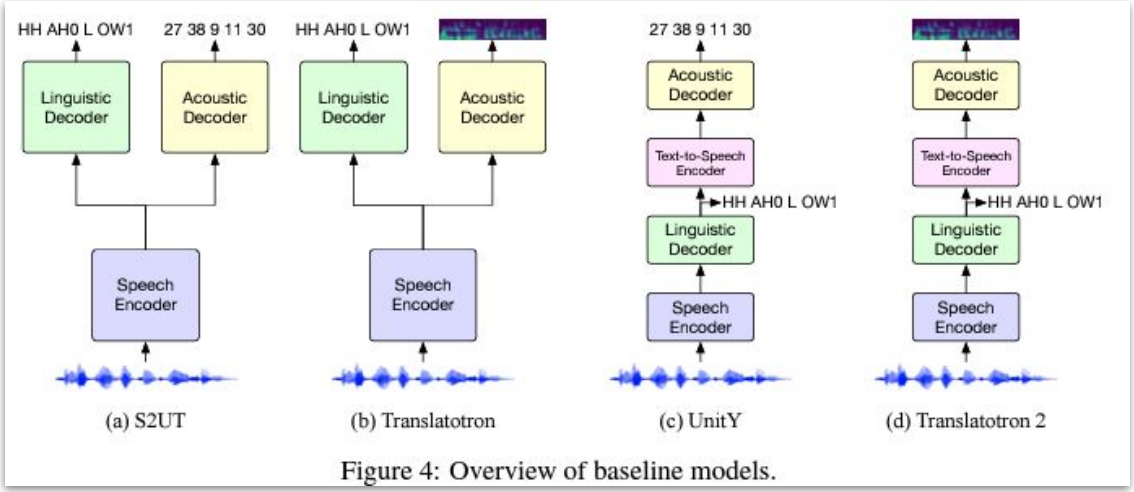


Figure 4: Overview of baseline models.

2. Background - baseline

- Decoder에 사용된 구조

Linguistic Decoder => **DA-Transformer** 구조를 사용

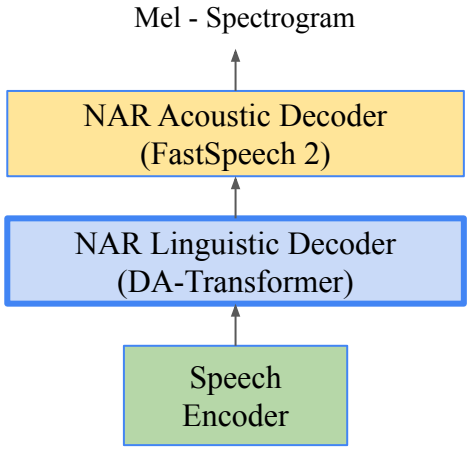
DA-Transformer 특징)

- Directed Acyclic Transformer로, NAR(비자기회귀) Transformer 모델이다.

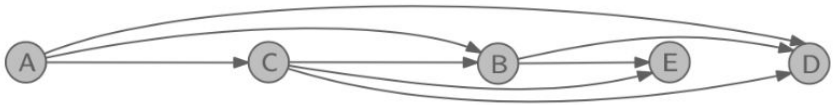
- 마지막 디코더 레이어의 hidden states 는 DAG 그래프 구조로 구성되어 있다.
  hidden states는 DAG의 정점에 해당.

- DAG는 여러 번역을 동시에 다룰 수 있기 때문에 언어적 다중 모달리티 문제를 완화.

- DAG의 다른 경로에 다른 번역을 할당



<DASpeech 의 Overview>



<DAG(비순환 일방향 그래프)구조 예시>

2.1. DA-Transformer(Directed Acyclic Transformer)

DA-Transformer : Non-autoregressive Machine Translation(NAT; 비자기번역모델)

(1) 지난 디코더 레이어에서 hidden states를 DAG(Directed Acyclic Graph)로 구성
(A : path, X : source data, Y : output)

(2) Linguistic Multi-modality 문제를 해결.
여러 개의 번역을 다룰 수 있다.
(DAG의 paths에 할당)

(3) 번역 확률 P(Y|X)은 모든 가능한 경우의 곱으로 계산

(4) Loss는 다이나믹 프로그래밍으로 연산

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

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

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

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

2.1. FastSpeech 2

FastSpeech 2 = non-autoregressive TTS 모델.

: 음소(phoneme) 시퀀스 → mel-spectrogram으로 생성하는 역할

특징)

- feed-forward Transformer, self-attention 레이어와 1d-conv 레이어로 되어있다.
- 3 variance predictor로 구성 (= duration predictor, pitch predictor, energy predictor)
  → 인풋의 시퀀스와 아웃풋의 멜-스펙트로그램 간의 정보 갭차이를 줄일 수 있다.
  → 추론 시 사용
- 보이스의 퀄리티를 개선, acoustic 멀티 모달리티를 개선

Training 할 때) 각 predictor의 Loss를 더하여 계산 (Loss : MSE(prediction, ground truth))
L_1 : mel-spectrogram의 L1 길이

$$\mathcal{L}_{\text{TTS}} = \mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{dur}} + \mathcal{L}_{\text{pitch}} + \mathcal{L}_{\text{energy}},$$

3. Architecture

3.1. Architecture

구조는 크게 3개의 파트로 구분:

1) Speech Encoder (Conformer Encoder)

2) NAR Linguistic Decoder (DA Transformer)

3) NAR Acoustic Decoder (FastSpeech 2)

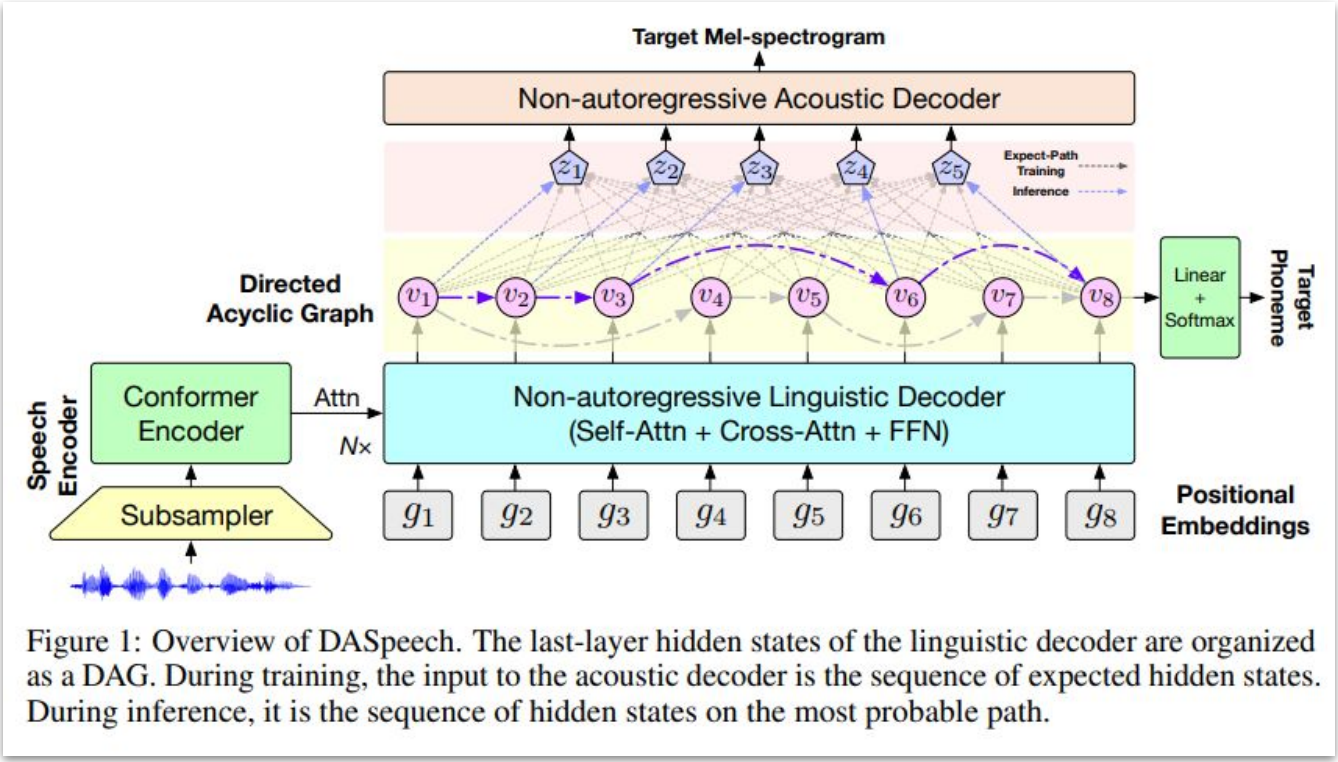


Figure 1: Overview of DASpeech. The last-layer hidden states of the linguistic decoder are organized as a DAG. During training, the input to the acoustic decoder is the sequence of expected hidden states. During inference, it is the sequence of hidden states on the most probable path.

<DASpeech Architecture>

3.1. Architecture

Subsampler
: 1차원 Conv 레이어 2개로 구성

Conformer Encoder
: Conformer block으로 이루어짐
multi-head attention 모듈과 Conv 레이어가 같이 결합

=> local 변수와 global 변수를 한번에 잡기 용이함.

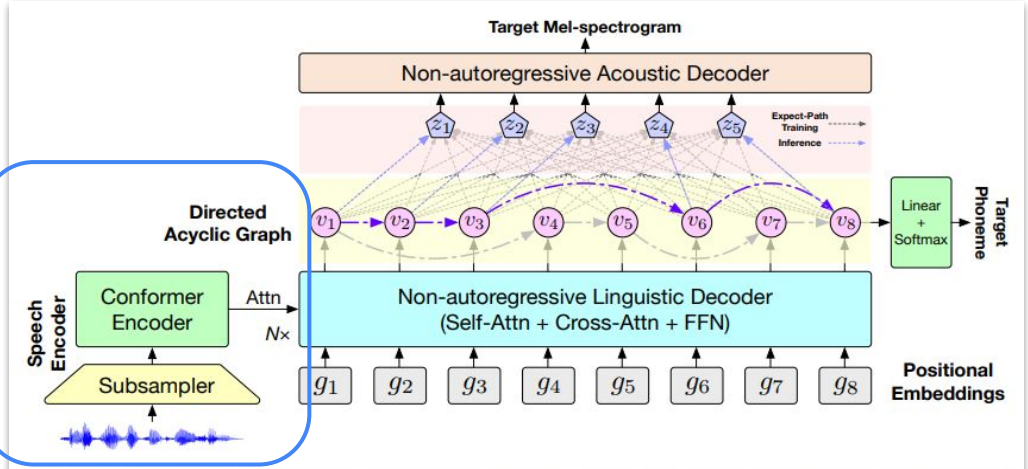


Figure 1: Overview of DASpeech. The last-layer hidden states of the linguistic decoder are organized as a DAG. During training, the input to the acoustic decoder is the sequence of expected hidden states. During inference, it is the sequence of hidden states on the most probable path.

3.1. Architecture

NAR Linguistic Decoder
: DA-Transformer의 디코더 구조를 그대로 쓰고 있음. (DA-Transformer는 음소 시퀀스로부터 speech를 병렬적으로 생성)

각 decoder는 self-attention 레이어, cross-attention 레이어, feed-forward 레이어로 구성

Positional Embeddings도 인풋으로 받음.
hidden states는 DAG 구조로 구성.

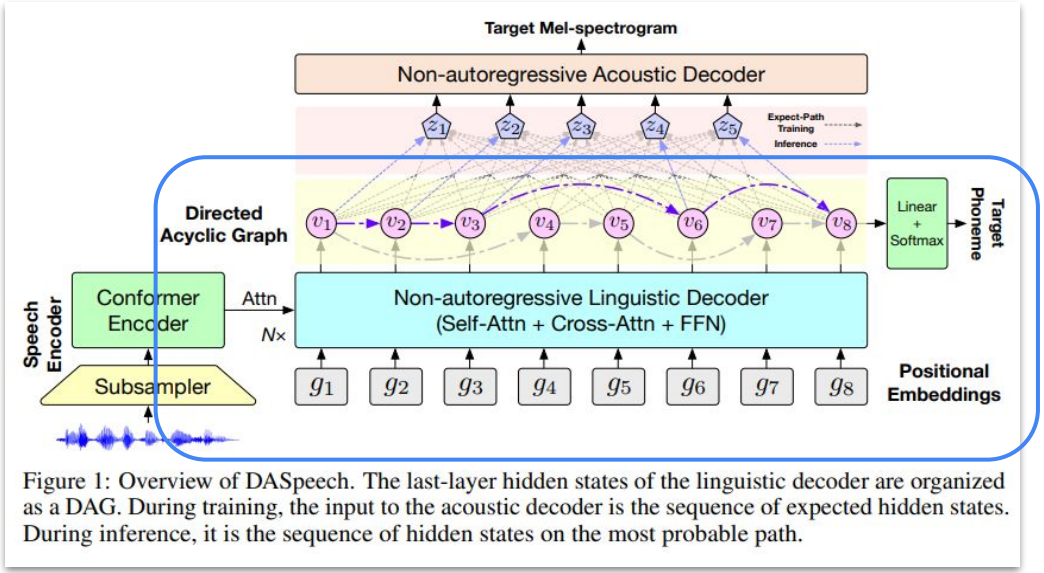


Figure 1: Overview of DASpeech. The last-layer hidden states of the linguistic decoder are organized as a DAG. During training, the input to the acoustic decoder is the sequence of expected hidden states. During inference, it is the sequence of hidden states on the most probable path.

3.1. Architecture

NAR Acoustic Decoder
: FastSpeech 2의 구조를 가져옴.

DA-Transformer의 마지막 레이어 hidden states를 가지고 mel spectrogram을 생성

predictor를 훈련하는 중에 시간(duration), 피치(pitch), 에너지(energy)정보가 사용되고, mel spectrogram을 생성하기 위한 조건부 입력으로도 사용됨.

정보의 도입으로 음향 다중 modality 문제를 완화

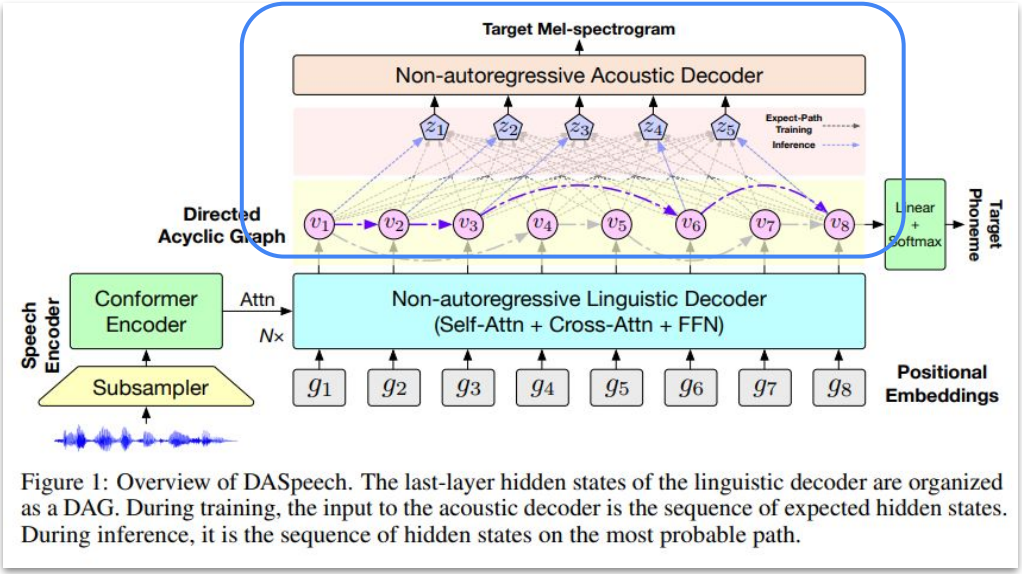


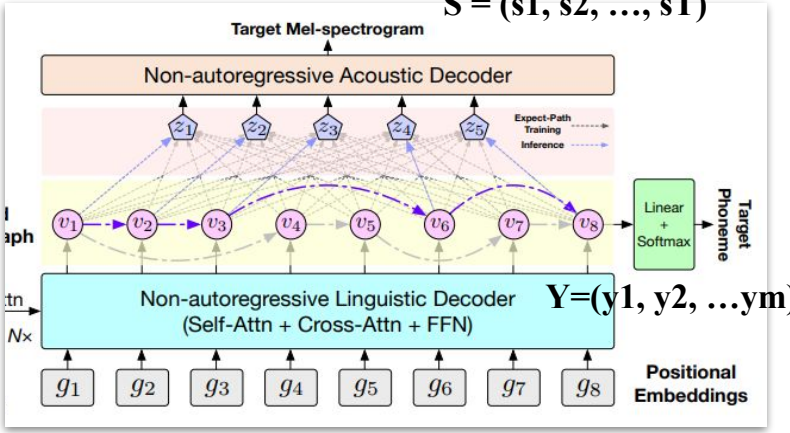
Figure 1: Overview of DASpeech. The last-layer hidden states of the linguistic decoder are organized as a DAG. During training, the input to the acoustic decoder is the sequence of expected hidden states. During inference, it is the sequence of hidden states on the most probable path.

3.2. Training

DASpeech : a Non-autoregressive two-pass direct S2ST model

- 데이터)
  스피치 : X=(x1, x2… xn)
  target 음소 : Y=(y1, y2, …ym)
  target 멜 스펙트로그램 : S = (s1, s2, …, sT)

S = (s1, s2, …, sT)



X=(x1, x2… xn)

Y=(y1, y2, …ym)

- 과정
  : target 음소에서 => target 멜-스펙트로그램 생성

(1) DA-Transformer(S2TT) 과정을 통해 X로부터 Y 생성
(2) DA-Transformer의 hidden states를 참고하여 FastSpeech 2에서 S 생성
(3) 사전훈련된 S2TT DA-Transformer와 FastSpeech 2를 갖고=>S2ST 전체 모델을 finetune

3.2. Training

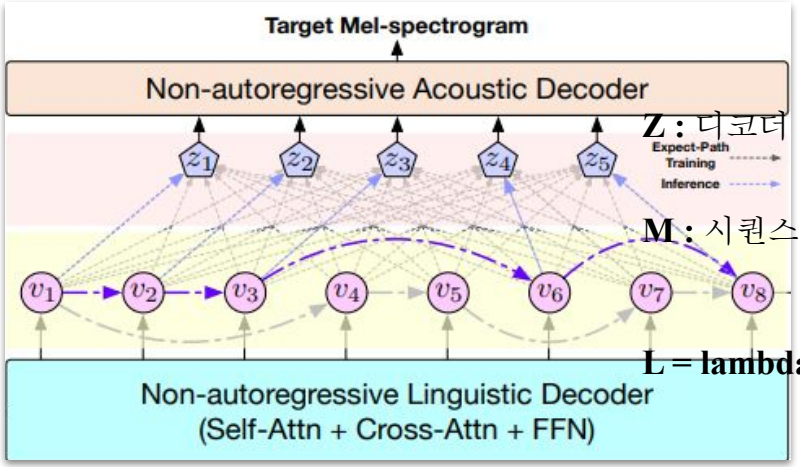
[사전훈련 - 미세조정 pipeline]

: 모델 훈련을 단순화한다는 장점이 있고, S2TT 데이터와 TTS 데이터를 쓸 수 있다.
단, linguistic decoder의 아웃풋과 acoustic decoder 의 인풋 사이에 길이 차이 이슈 있음.

linguistic 디코더의 hidden states 길이 : L = lambda * N
acoustic 디코더의 input sequence 길이 : M
(phoneme 시퀀스 길이 : M)

linguistic 디코더의 hidden states : V=(v1,...,vL)
acoustic 디코더의 input sequence : Z=(z1,...,zM)

길이 차이 해결 => **Expect-Path Training**



Z : 디코더 인풋

M : 시퀀스 길이

L = lambda*N

3.2. Training

[Expect-Path Training]

: sequence Zi 는 output Yi 와 대응 / Y를 생성할 수 있는 모든 잠재적 경로를 고려할 때,
=> Zi를 사후 분포 P로 정의

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

: 경로(path) A에 대한 정점 j의 확률 P를 정의 / P(Y,A|X) 의 모든 합을 계산 => **forward-backward**

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

3.2. Training

[Forward & Backward Training]

(순방향 확률) αi(j) = Pθ(y1, ..., yi , ai = j|X)
: target sequence의 일부 (Y1, Y2, … Yi) 를 생성할 확률

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

(역방향 확률) βi(j) = Pθ(yi+1, ..., yM|ai = j, X)
: i번째에서 시작하여 target sequence의 나머지를 생성할 확률

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

시간복잡도는 O(ML^2), 최종 loss는 $L_{DASpeech} = L_{DAT} + \mu \cdot L_{TTS}$ 로 계산한다.

3.3. Inference

[**Inference 과정**] : 추론 과정에서 2-pass 병렬 decoding을 수행.

- DA-Transformer에서 DAG의 가장 가능성이 높은 경로 A*를 찾는다.
- hidden states를 acoustic decoder에 공급하여 Mel-Spectrogram 생성
- 음성으로 변환 하는 과정은 HiFi-GAN 보코더 사용

=> DAG, TTS 모두 병렬 과정이기 때문에 디코딩 효율성이 크게 향상

[**Inference 과정**] : DAG의 디코딩 전략)

Lookahead : 그리디 방식을 적용 / 각 디코딩 단계에서 전이 확률과 예측 확률을 동시에 고려

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

Joint-Viterbi : Viterbi 디코딩 / 타겟의 길이 M을 정하고 a * M = L을 역추적하며 최선의 경로를 찾음

$$A^*, Y^* = \arg\max_{A,Y} P_\theta(Y, A|X).$$

4. Experiment

Experiment

Dataset : CVSS-C dataset (Fr->En), CVSS-T dataset(Fr->En) (from CoVoST 2)

Vocoder : HiFi-GAN 보코더 사용 (VCTK dataset으로 사전훈련된 모델)

Training : 사전 훈련 -> fine tuning

- 사전 훈련
  - (1)Speech Encoder & (2)Linguistic Decoder사전 훈련 => 320k audio frames.
  - (3)Acoustic Decoder => 100k updates

- fine tuning
  - 미세 조정 훈련 : 50k updates with a batch of 320k audio frames.
  - 코드 구현은 fairseq 오픈 소스 사용
  - 훈련 시 python 3.8, 4 RTX 3090 GPU 사용

Evaluation : 번역) BLEU score 사용 스피드) 1 GPU, 1 batch, test set으로 측정

Experiment

Baseline

TranSpeech
DASpeech

DA-T.
FastSpeech 2

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	/

Experiment

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 ($p < 0.01$).

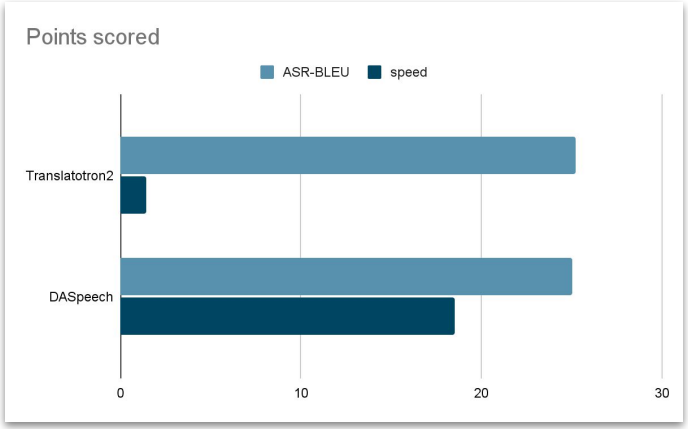
ID	Models	Decoding	#Iter	#Param	ASR-BLEU (Fr→En) CVSS-C	CVSS-T	Speedup
	Ground Truth	/	/	/	84.52	81.48	/
	Single-pass autoregressive decoding						
A1	S2UT [5]	Beam=10	T_{unit}	73M	22.23	22.28	1.00×
A2	Translatotron [3]	Autoregressive	T_{mel}	79M	16.96	11.25	2.32×
	Two-pass autoregressive decoding						
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×

[Result 1]

** Baseline 목록에서 two-pass 구조를 갖는 UnitY, Translatotron2가 더 높은 번역 성능을 보임.

Experiment

[Result 2]
** Translatotron 2와 비교했을 때, 번역 성능은 거의 비슷하면서 속도는 약 18.53x의 속도로 매우 개선됨



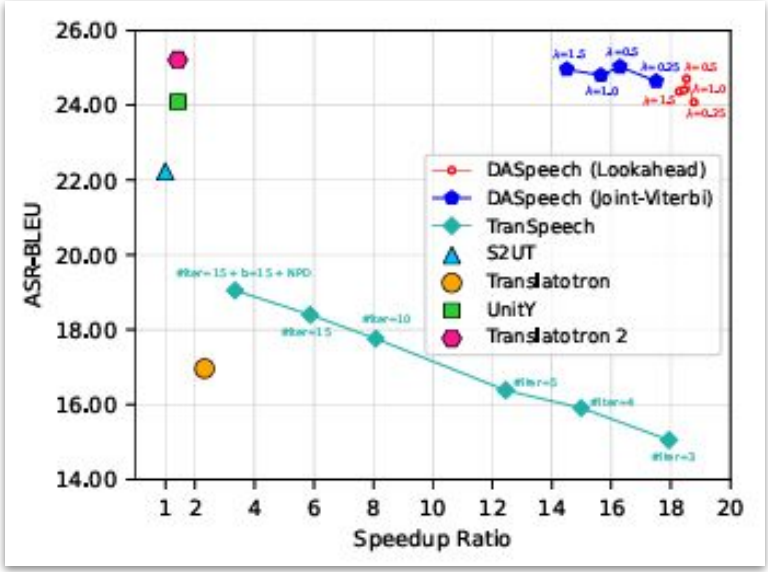
[Result 3]
** TranSpeech의 NAR 모델이랑 비교했을 때, 지식 증류(knowledge distillation)을 적용하지 않아도 성능과 속도 면에서 개선됨

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	/

Experiment

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



** speaker 목소리의 유사성 지표

=> 높은 similarity

** 번역 성능과 디코딩 속도의 trade-off

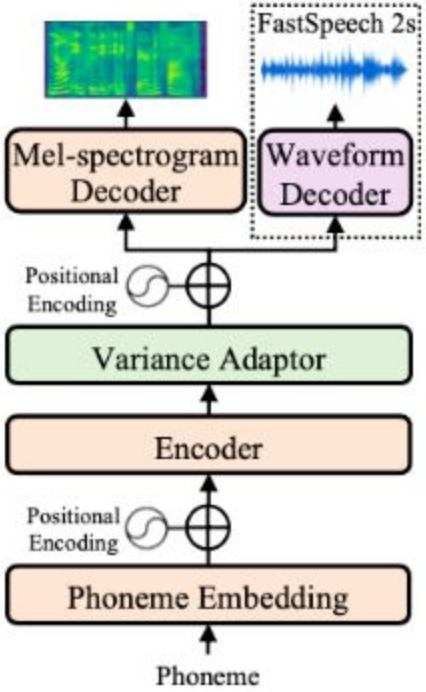
=> 우측 상향으로 갈수록 번역 성능이 좋고 속도가 빠른 편. (DASpeech 위치)

5. Conclusion

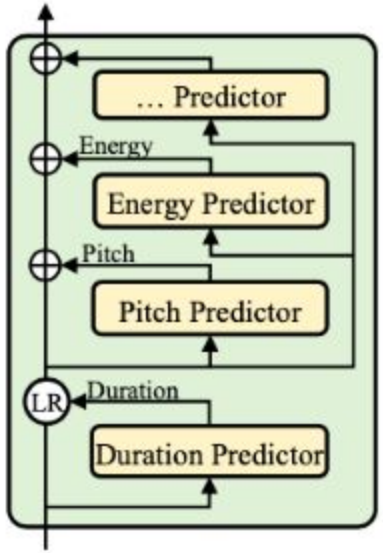
Conclusion

- 새로 Non-autoregressive two-pass direct S2ST 모델을 소개했다는 의의.
- DASpeech는 Translatotron2와 비슷한 성능을 유지하면서도 AR 모델에 비해 최대 18배 더 빠른 속도를 가진다.
- DASpeech는 AR 모델보다 번역 성능이 좋고 디코딩 속도가 빠르다.

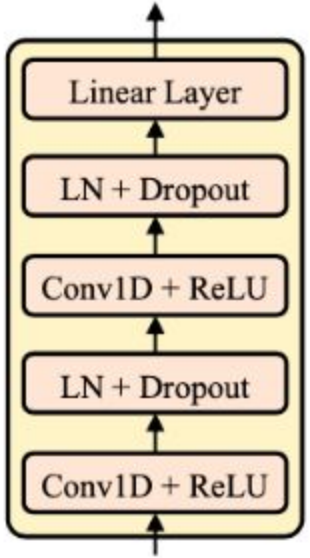
QnA



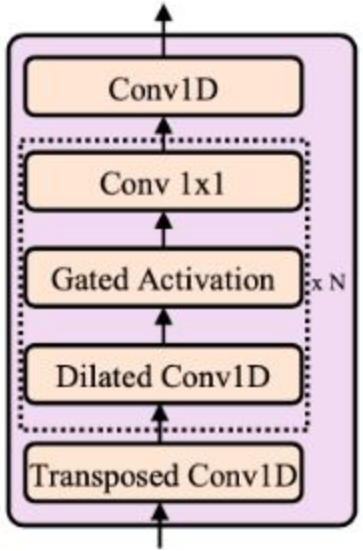
(a) FastSpeech 2



(b) Variance adaptor



(c) Duration/pitch/energy predictor



(d) Waveform decoder