Progress in Singing Voice Synthesis with Muskits

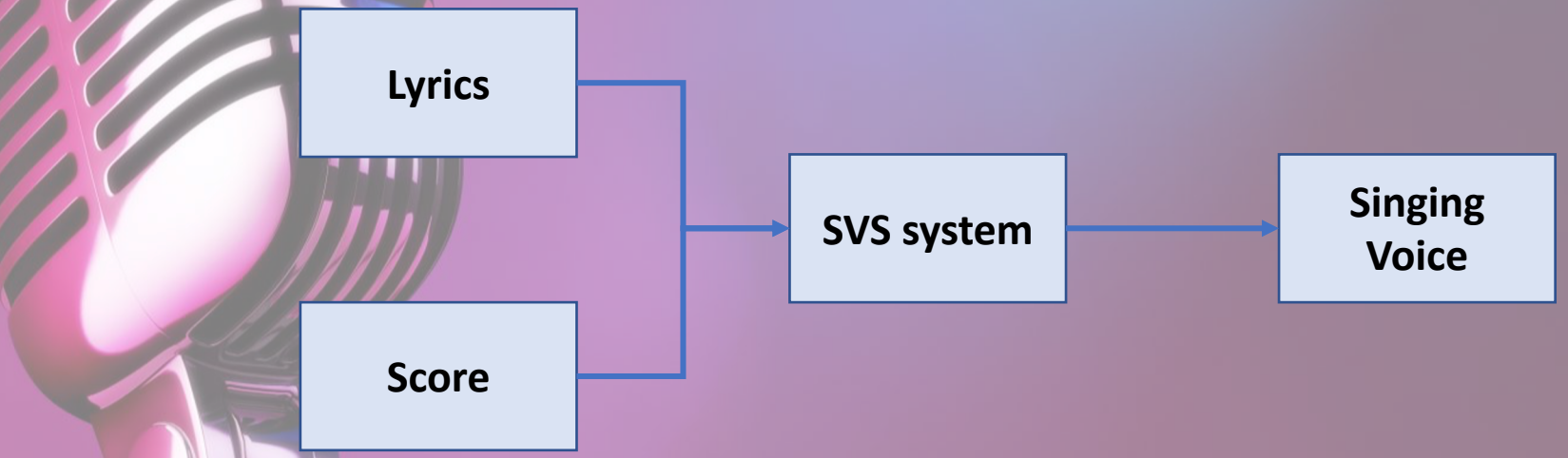
Jiatong Shi

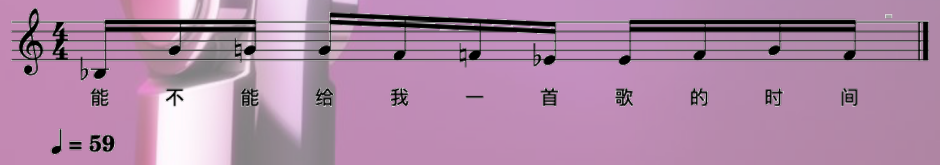
jiatongs@cs.cmu.edu

Content

- Singing voice synthesis (SVS): background and challenges
- Muskits: an end-to-end singing voice synthesis toolkit
- Research with Muskits
  - Perceptual entropy loss
  - SingAug: data augmentation strategies
  - PHONEix: enhancement to singing pronunciation
- Future works

Singing Voice Synthesis





Challenges in SVS

- Compared to text-to-speech (TTS), SVS has its difficulty in data collection
  - Has more strict usage guidelines and copyright concerns

Challenges in SVS

- Compared to text-to-speech (TTS), SVS has its difficulty in data collection
  - Has more strict usage guidelines and copyright concerns
  - Requires a professional high-quality recording environment

Challenges in SVS

- Compared to text-to-speech (TTS), SVS has its difficulty in data collection
  - Has more strict usage guidelines and copyright concerns
  - Requires a professional high-quality recording environment
  - Needs longer time to record

Challenges in SVS

- Compared to text-to-speech (TTS), SVS has its difficulty in data collection
  - Has more strict usage guidelines and copyright concerns
  - Requires a professional high-quality recording environment
  - Needs longer time to record
  - Needs detailed annotation (e.g., music score information, lyrics alignment)

Fortunately…

- We still have some great efforts in releasing databases and code bases for some SVS models.
  - Sinsy, known as one of the initial open-source works (HMM-architecture)
    - https://www.sinsy.jp/
  - NN-SVS, supports 11 recipes until 11/27/2022 (cascaded DNN-architecture)
    - https://github.com/nnsvs/nnsvs
  - Others for specifics models or datasets
    - https://github.com/MoonInTheRiver/DiffSinger
    - https://github.com/neosapience/mlp-singer
    - https://github.com/NVIDIA/mellotron
    - SoonbeomChoi/BEGANSing

However…

- Difficulties in providing a fair comparison of different models
  - Differences in datasets
  - Differences in feature representations
  - Differences in vocoders

The Goal of Muskits

- A platform for fair comparison of different models
- A platform for end-to-end SVS researchers
- A platform for different types of singing voice synthesis
  - Single-singer
  - Multi-singer
  - Multilingual
  - Transfer learning

Muskits at a Glance

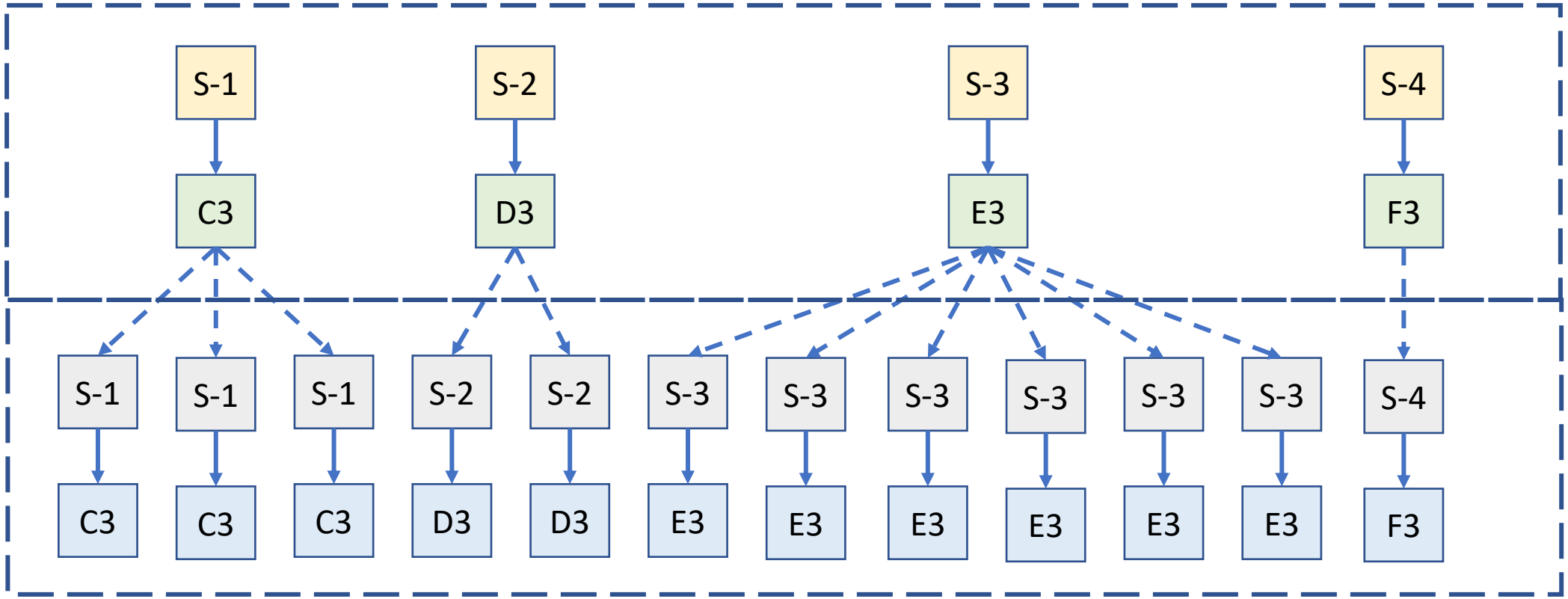
- Python library
  - Neural network architectures
  - A uniform training/decoding framework

- Shell recipes
  - All-in-one style scripts (following Kaldi and ESPnet)

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., ... & Vesely, K. (2011). The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding* (No. CONF). IEEE Signal Processing Society

Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., ... & Ochiai, T. (2018). Espnet: End-to-end speech processing toolkit. *arXiv preprint arXiv:1804.00015*.

Music Score Representation



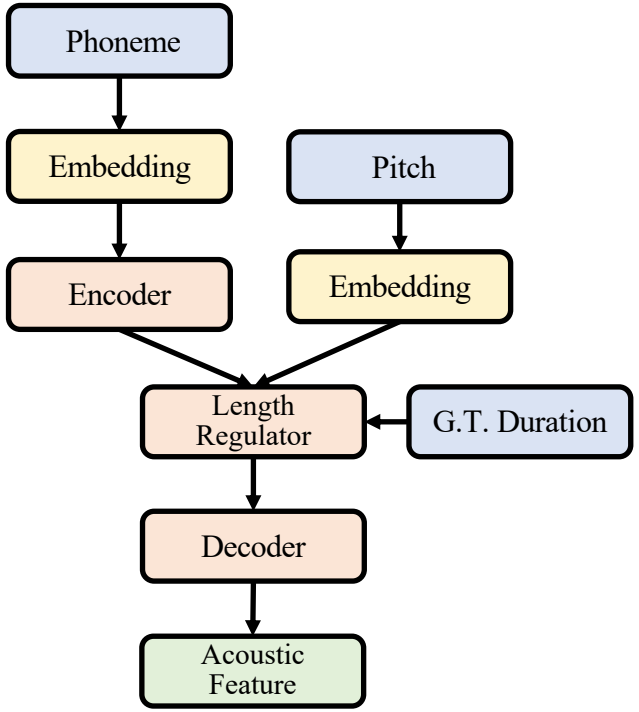
Syllable-level

+
Duration
Information
=

Frame-level

Models

Without ground truth duration (G.T. Dur)

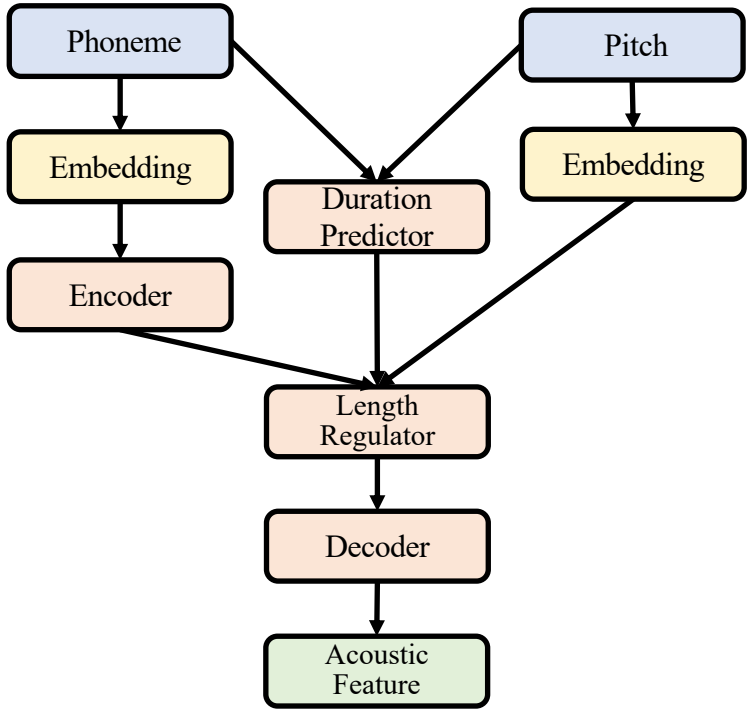


Current Supported Models

RNN
Transformer
XiaoiceSing

Check more details either from the paper or the Github repository

With ground truth duration (G.T. Dur)



Training, Inference, and Evaluation

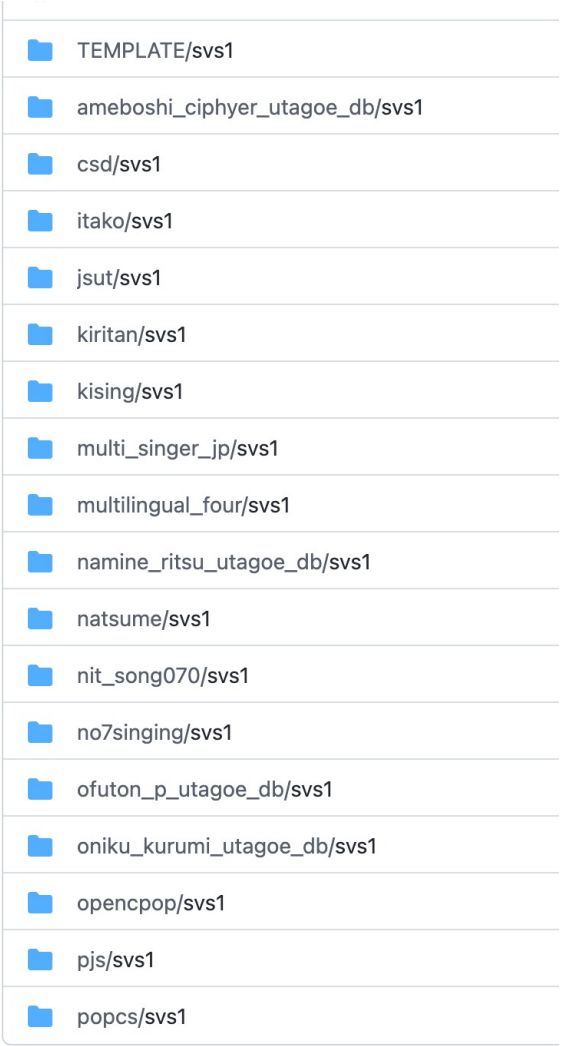
- Training
  - On-the-fly data augmentation (pitch augmentation and mixup-augmentation)
- Inference
  - Various neural vocoders
- Evaluation (Objective)
  - Mean cepstral distortion (MCD), voice/unvoiced error rate (VUV_E), rooted mean square error of the fundamental frequency (F0RMSE)
  - Calculated by dynamic time-warping

Shuai Guo∗, Jiatong Shi∗, Tao Qian+, Shinji Watanabe and Qin Jin. ”SingAug: Data Augmentation for Singing Voice Synthesis with Cycle-consistent Training Strategy ” (Interspeech 2022)

How is Muskits

Recipes

- 17 recipes and 1 template
- 15 corpora
  - 11 Japanese
  - 3 Mandarin
  - 1 Korean + English
- 2 special settings
  - Multi-singer
    - Four singer
  - Multilingual
    - Four-languages



Experiments (data)

- Four different scenarios
  - Singler-singer
  - Multi-singer
  - Multilingual
  - Transfer learning

Table 1: *Details of experimental data: N_{src}, N_{lang}, N_{singer} stands for number of databases, languages, and singers, respectively.*

Task	N_{src}, N_{lang}, N_{singer}	Duration(h)		
		Train	Dev	Test
Single-singer	1,1,1	0.63	0.08	0.07
Multi-singer	4,1,4	3.01	0.34	0.38
Multilingual	6,4,6	11.72	0.49	1.01
Transfer	1,1,1	0.71	0.02	0.05

(All from open-source databases)

Single/multi-singer experiment

Table 2: *: Single- and multi-singer experiments: The experimental databases are Ofuton-P database for single-singer experiments and a combination of four databases as described in Sec 4.1; the models are discussed in Sec 4.2; the metrics are defined in Sec 3.3.*

Model	G.T. Dur.	Single-Singer MCD↓	Single-Singer VUV_E↓	Single-Singer F_0RMSE↓	Single-Singer MOS↑	Multi-Singer MCD↓	Multi-Singer VUV_E↓	Multi-Singer F_0RMSE↓	Multi-Singer MOS↑
RNN [10]	✓	6.52	2.30	0.105	**3.47 ± 0.11**	7.15	3.87	0.195	3.25 ± 0.13
Transformer [20]	✓	**6.46**	2.67	0.118	3.23 ± 0.11	**6.68**	3.90	**0.184**	**3.40 ± 0.12**
XiaoiceSing [22]	✓	6.88	**2.22**	**0.103**	3.11 ± 0.11	6.81	**3.86**	0.190	3.23 ± 0.12
RNN [10]	/	**6.19**	**2.18**	**0.109**	**3.43 ± 0.11**	7.05	**3.55**	0.226	2.84 ± 0.12
Transformer [20]	/	6.95	5.61	0.172	2.39 ± 0.11	6.87	3.70	0.186	**3.33 ± 0.12**
XiaoiceSing [22]	/	7.19	2.68	0.120	1.96 ± 0.08	**6.79**	3.85	**0.154**	2.83 ± 0.11
G.T.	/	/	/	/	4.89 ± 0.05	/	/	/	4.65 ± 0.10

The results are kind of mix, but surprisingly, RNN-based model has better performances over Transformer and XiaoiceSing in single-singer + G.T. duration.

Does multilingual pre-training help?

Table 3: *Multilingual experiments on Ofuton-P. All the models use the same XiaoiceSing architecture. The pre-trained models are conducted over the multilingual data described in Sec. 4.1.*

Model	MCD↓	VUV_E↓	F_0RMSE↓	MOS↑
Baseline	6.88	**2.22**	**0.103**	3.05 ± 0.12
Pre-train	**6.17**	2.51	0.113	2.99 ± 0.12
+ LangID	6.32	2.94	0.123	2.90 ± 0.12
Fine-tune	6.23	2.75	0.106	$\mathbf{3.49 \pm 0.12}$
+ LangID	6.34	2.55	0.110	3.09 ± 0.12
G.T.	/	/	/	4.77 ± 0.07

Pre-training model may not direct offer better performances, but fine-tuning could help

Does transfer learning help?

Table 4: *Objective evaluation for transfer learning experiments on KiSing (a low-resource multi-style singing database). All the models use the same XiaoiceSing architecture. Models with ∗ are first pre-trained on the corresponding database and then fine-tuned on KiSing.*

Model	MCD↓	VUV_E↓	F_0RMSE↓
Baseline	8.74	3.74	0.214
Opencpop∗	8.52	**3.54**	0.190
Multilingual∗	**8.25**	3.66	**0.177**

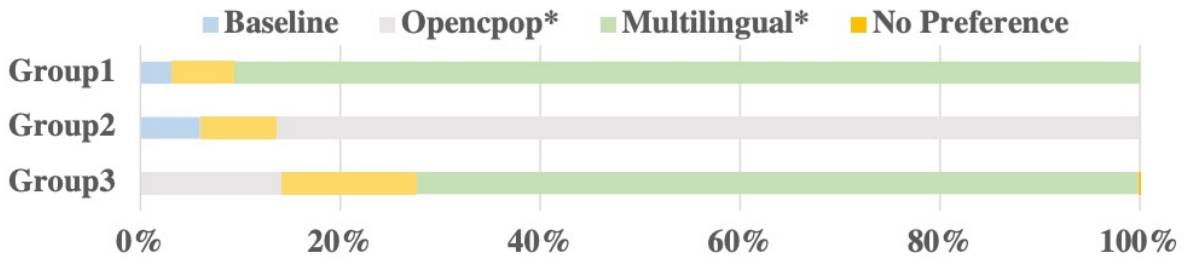


Figure 1: *A/B test for transfer learning experiments on KiSing.*

- KiSing, a low-resource multi-style database
  - Limited variances in terms of phoneme sequences and melodies
  - Both Chinese folk style and a pop style, resulting in frequent changes

Real-time Demonstration

Muskit real time singing voice synthesis demonstration

This notebook provides a demonstration of the realtime singing voice synthesis using Muskits and Hifi-GAN.

- Muskits: https://github.com/SJTMusicTeam/Muskits
- ParallelWaveGAN: https://github.com/kan-bayashi/ParallelWaveGAN

Author: Jiatong Shi (@ftshijt)

- https://colab.research.google.com/github/SJTMusicTeam/svs_demo/blob/master/muskit_svs_realtime.ipynb

Muskits: an End-to-End Music Processing Toolkit for Singing Voice Synthesis

Jiatong Shi[1], *Shuai Guo*[2], *Tao Qian*[2], *Nan Huo*[3], *Tomoki Hayashi*[4,5], *Yuning Wu*[2], *Frank Xu*[1], *Xuankai Chang*[1], *Huazhe Li*[6], *Peter Wu*[7], *Shinji Watanabe*[1], *Qin Jin*[2†]

[1]Carnegie Mellon University, [2]Renmin University of China, [3]The University of Hong Kong, [4]Human Dataware Lab. Co., Ltd., [5]Nagoya University, [6]Tsinghua University, [7]University of California, Berkeley

jiatongs@cs.cmu.edu, {shuaiguo, qiantao, qjin}@ruc.edu.cn, shinjiw@ieee.org

Research with Muskits

- Low-resource SVS
  - Perceptual Entropy loss (ICASSP2021)
  - Data augmentation strategies (Interspeech 2022)
- Syllable-phoneme alignment in SVS
  - PHONEix: enhancement to singing pronunciation (ICASSP2023 Submission)

Recap the challenges in SVS

- Lots of difficulties in data collection. As a result, SVS dataset is usually much smaller than TTS's.
  - Typical size of a SVS dataset is around 1hr to 5hrs.

Recap the challenges in SVS

- Lots of difficulties in data collection. As a result, SVS dataset is usually much smaller than TTS's.
  - Typical size of a SVS dataset is around 1hr to 5hrs.

- Problem:
  - Data scarcity
    - Overfitting issue → Perceptual entropy loss
    - Need to use data in more efficient manners → SingAug: augmentation strategies

SEQUENCE-TO-SEQUENCE SINGING VOICE SYNTHESIS WITH PERCEPTUAL ENTROPY LOSS

Jiatong Shi[1*], *Shuai Guo*[2*], *Nan Huo*[1], *Yuekai Zhang*[1], *Qin Jin*[2†]

[1] Johns Hopkins University, USA
[2] Renmin University of China, P.R.China

- The synthesised sound examples can be found at https://peterguoruc.github.io/SVS_pe.github.io/

Motivation of Perceptual Entropy Loss

- Perceptual Entropy as a regularization factor
  - Psycho-acoustic model
  - Perceptual Entropy ↑ = human perceptive details ↑

Perceptual Entropy Loss - Calculation

$$PE(t) = \sum_{i=1}^{n} \sum_{\omega=l_{ti}}^{h_{ti}} [\log_2 \left(2 \cdot \left| \frac{Re(\omega)}{\sqrt{\frac{6T_i'}{k_i}}} \right| + 1 \right) + \log_2 \left(2 \cdot \left| \frac{Im(\omega)}{\sqrt{\frac{6T_i'}{k_i}}} \right| + 1 \right)] \quad (1)$$

$$\mathcal{L}_{pe} = \frac{1}{1 + PE} \quad (2)$$

$$\mathcal{L} = \mathcal{L}_{sing} + \lambda \cdot \mathcal{L}_{pe} \quad (3)$$

Variable Description:

i is the index of critical band;

n is the number of bands given the sampling rate of the system;

l_{ti} and h_{ti} are the upper and lower bounds of band i at time t;

k_i is the number of frequency components in band i,

$T_{i'}$ is the masking threshold in band i.

Re(ω) and *Im(ω)* denote the real and imaginary parts of the predicted spectrum.

References:

1.T.Painterand and A.Spanias,"Perceptual coding of digital audio," in *Proc. IEEE*, 2000, pp. 451–515.

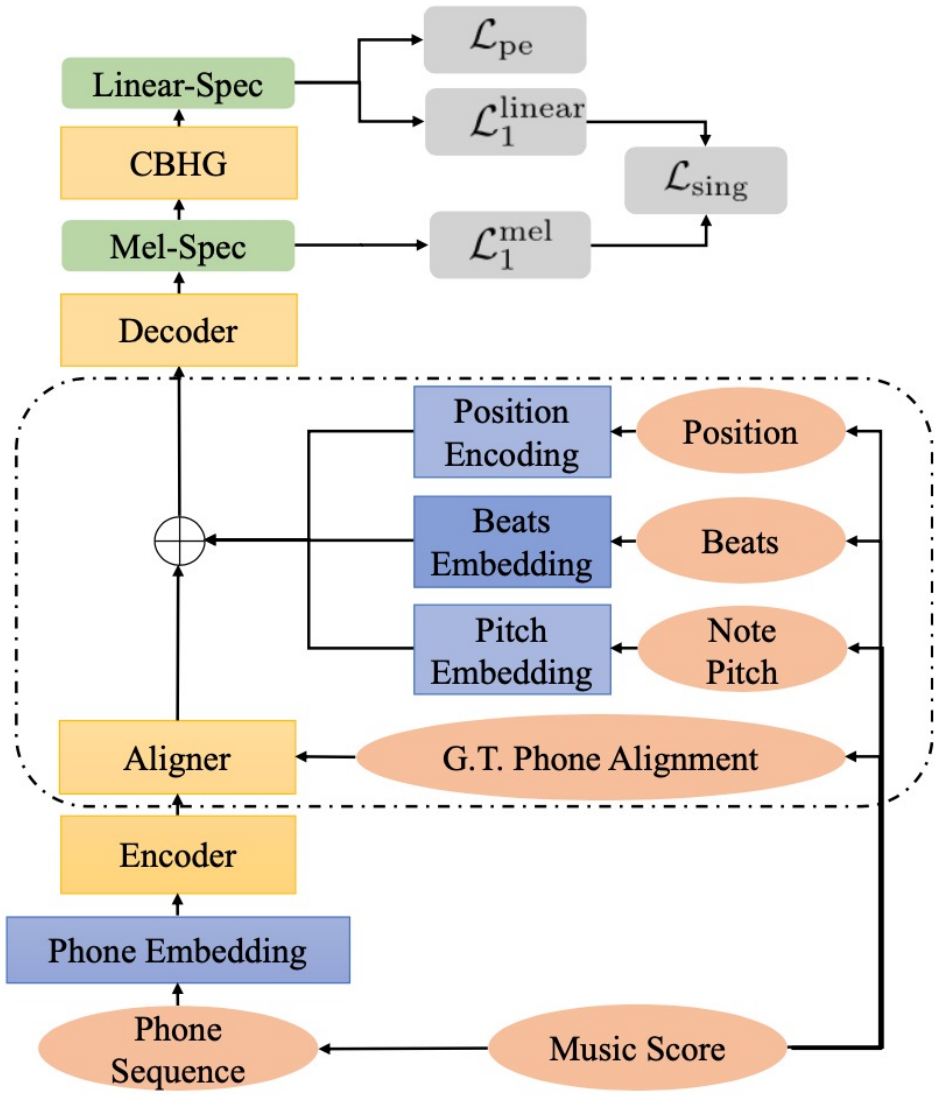
2.N. S Jayant and P. Noll, "Digital coding of waveforms: principles and applications to speech and video," *Englewood Cliffs, NJ*, pp. 115–251, 1984.

3.N. Jayant, et al., "Signal compression based on models of human perception," *Proc. the IEEE*, vol. 81, no. 10, pp. 1385–1422, 1993.

4.H. Fletcher, "Auditory patterns," *Reviews of modern physics*, vol. 12, no. 1, pp. 47, 1940.

Experiments on Kiritan

- 65 min Japanese SVS corpus
- Tested on three encoder/decoder architectures (i.e., RNN, Transformer, Conformer)



Objective Evaluation

Table 5. Evaluation of the synthesized singing quality based on different models with and without PE loss on the Kiritan dataset. * means that the model is trained with the PE loss.

Model	VAL SET MCD(dB)	TEST SET MCD(dB) ↓	F0_RMSE(Hz) ↓	VUV_ERROR(%) ↓	F0_CORR ↑
RNN	3.41	7.19	52.05	9.40	0.79
RNN*	3.39	**5.99**	45.53	5.26	0.85
Transformer	3.38	6.48	44.99	4.95	0.85
Transformer*	3.52	6.60	48.07	4.60	0.84
Conformer	3.49	7.01	47.82	6.69	0.83
Conformer*	3.77	6.47	**40.79**	**4.53**	**0.89**

Subjective Evaluation

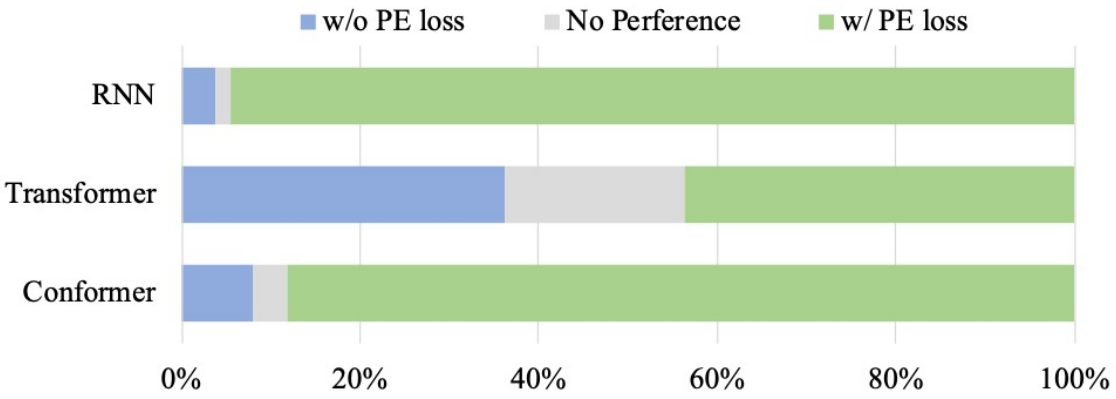


Fig. 2. The first set of Subjective Evaluation on the impact of the PE loss.

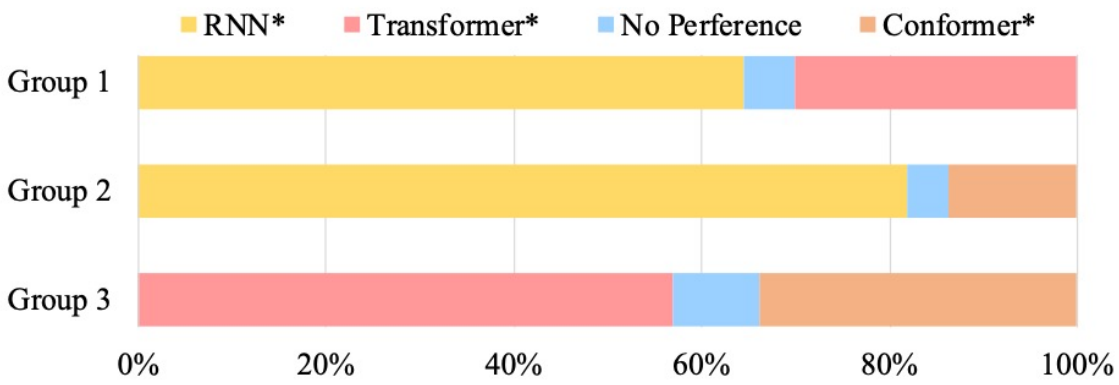


Fig. 3. The second set of Subjective Evaluation on the impact of different model architectures.

How PE loss improves the modeling

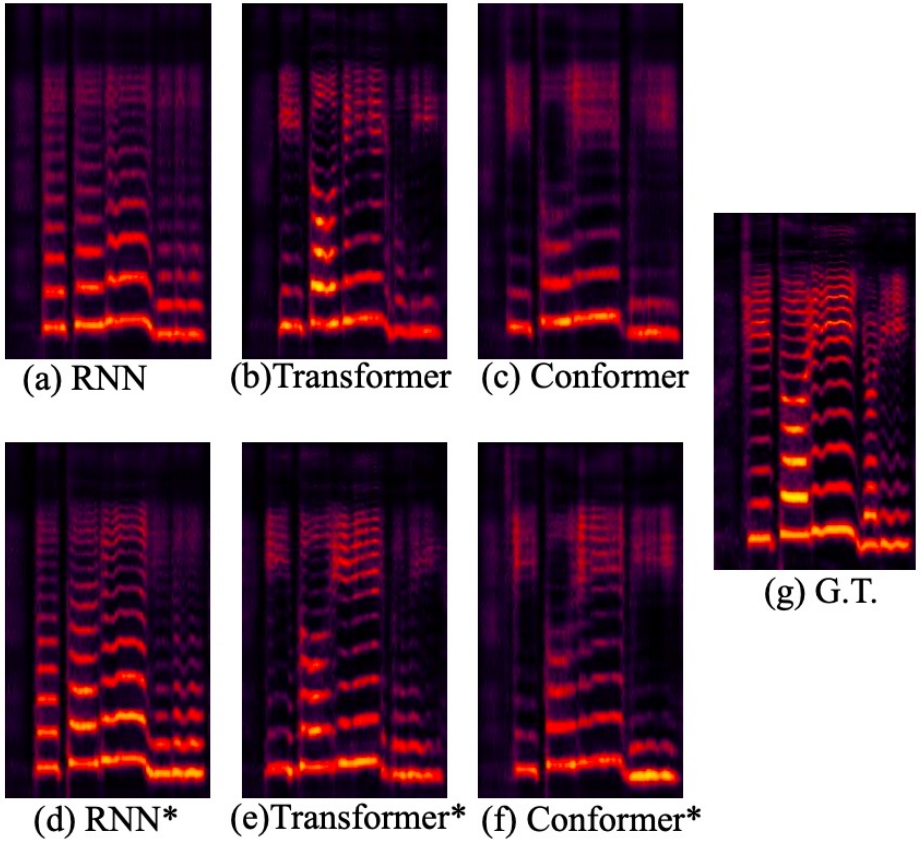


Fig. 4. Time-Frequency spectrum of an example synthesized singing from different models. * stands for models trained using the PE loss. G.T. refers to the ground truth singing voice.

Better F0 Modeling

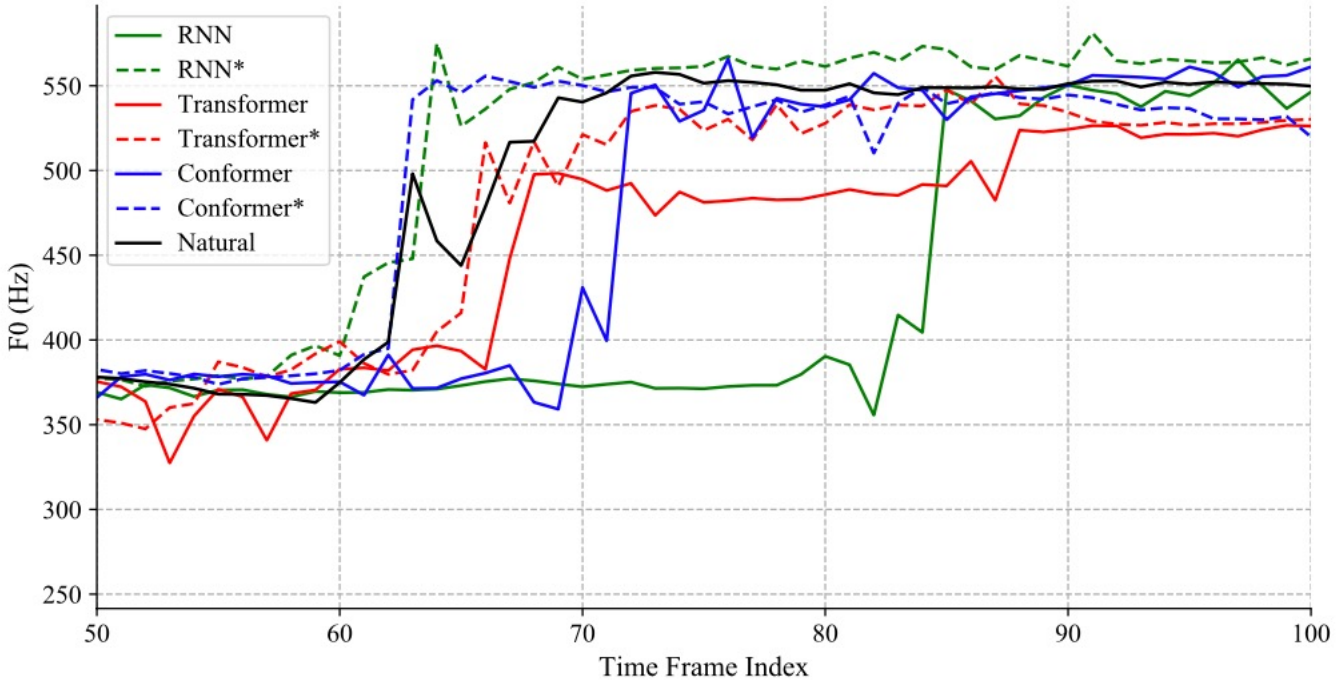


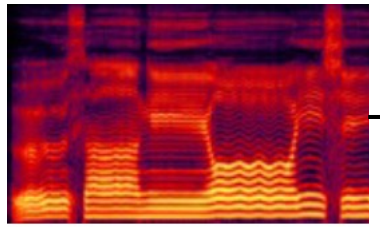
Fig. 5. F0 Contour prediction based on different models. * stands for models trained using PE loss. Natural is the ground truth singing.

SingAug: Data Augmentation for Singing Voice Synthesis with Cycle-consistent Training Strategy

Shuai Guo[1*], *Jiatong Shi*[2*], *Tao Qian*[1], *Shinji Watanabe*[2], *Qin Jin*[1†]

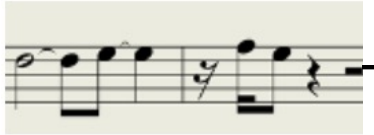
[1]School of Information, Renmin University of China, P.R.China
[2]Language Technologies Institute, Carnegie Mellon University, U.S.A.

Recap of the SVS task



Acoustic Feature

$$Y \in \mathbb{R}^{T \times D^a}$$



Music Score

Phoneme sequence $X^{\text{ph}} \in \mathbb{N}^{T'}$

Pitch sequence $X^{\text{pi}} \in \mathbb{N}^{T'}$

Duration sequence $X^{\text{beats}} \in \mathbb{N}^{T'}$

- Original SVS loss

$$\mathcal{L}_{\text{svs}}^{\text{ori}} = L_1\left(f(X;\theta_{xy}), Y\right)$$

Variable Description:

T is the number of frames in the sequence;

T' is the number of tokens;

D^a is the acoustic feature dimension;

θ_{xy} refers to the trainable parameters of the acoustic model;

$f(X;\theta_{xy})$ denotes the acoustic model;

$L_1(\cdot)$ denotes the L1 loss function;

Base Framework

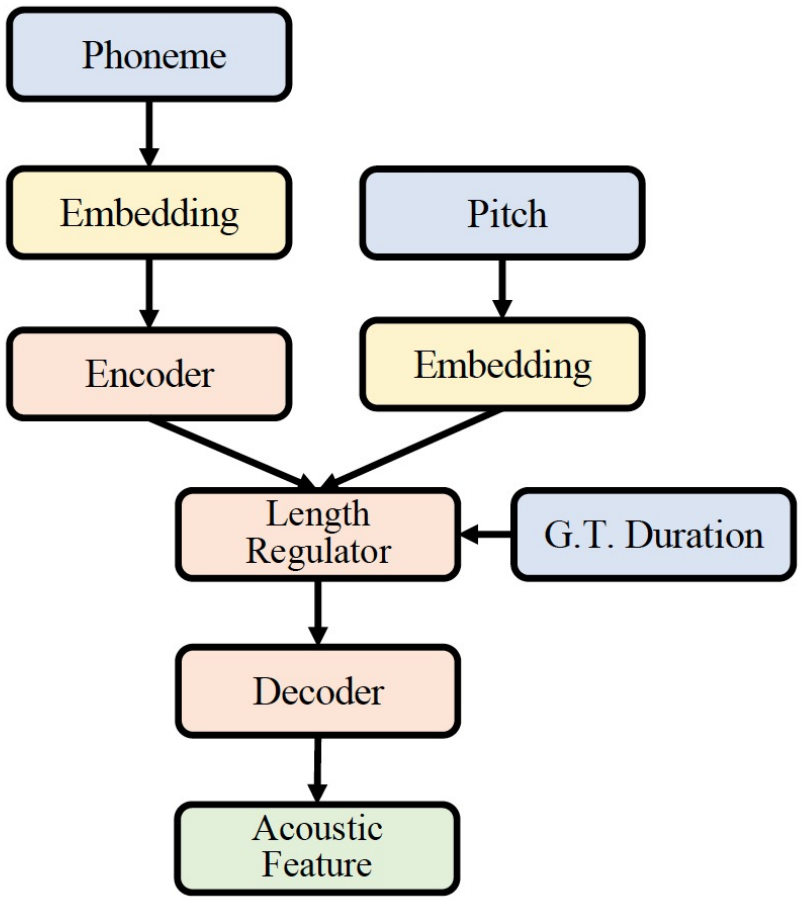
The Architecture of the base SVS System:

blue blocks: the model inputs;

yellow blocks: the embedding layers;

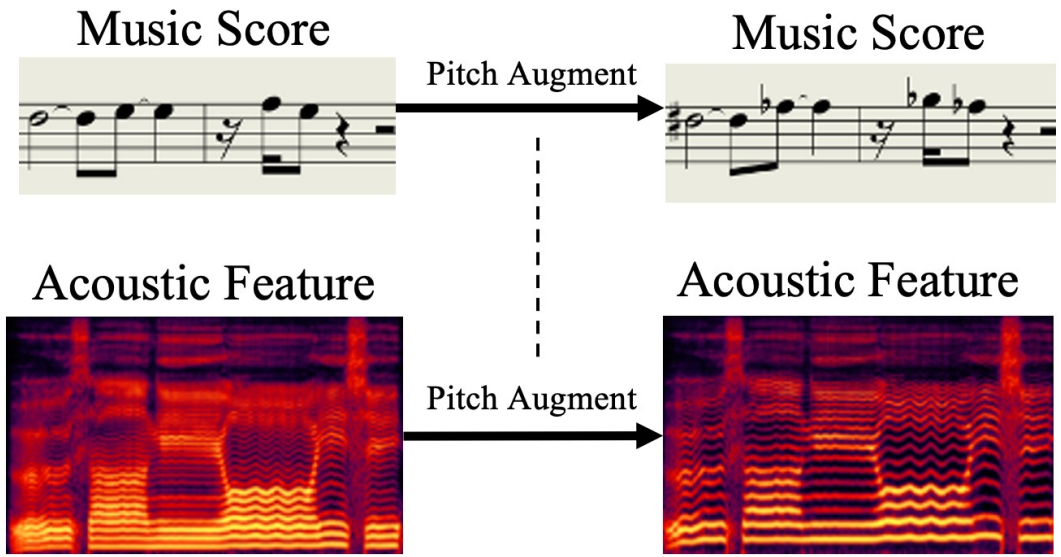
orange blocks: the main components of the SVS system;

green blocks: the model outputs;



References:
1.J.Wu and J. Luan, “Adversarially Trained Multi-Singer Sequenceto- Sequence Singing Synthesizer,” Interspeech, pp. 1296–1300, 2020.
2.J. Shi, S. Guo, N. Huo, Y. Zhang, and Q. Jin, “Sequence-tosequence singing voice synthesis with perceptual entropy loss,” in ICASSP, 2021, pp. 76–80.
3.Y.-H. Yi, Y. Ai, Z.-H. Ling, and L.-R. Dai, “Singing Voice Synthesis Using Deep Autoregressive Neural Networks for Acoustic Modeling,” Interspeech, pp. 2593–2597, 2019.

Pitch Augmentation (PA)



References:
1.M. Blaauw and J. Bonada, “A Neural Parametric Singing Synthesizer Modeling Timbre and Expression from Natural Songs,” Applied Sciences, vol. 7, no. 12, p. 1313, 2017.
2.C. Zhang, J. Yu, L. Chang, X. Tan, J. Chen, T. Qin, and K. Zhang, “PDAugment: Data Augmentation by Pitch and Duration Adjustments for Automatic Lyrics Transcription,” arXiv preprint arXiv:2109.07940, 2021.
3.Z. Zhang, Y. Zheng, X. Li, and L. Lu, “Wesinger: Dataaugmented singing voice synthesis with auxiliary losses,” arXiv preprint arXiv:2203.10750, 2022.

Mix-up Augmentation (MA)

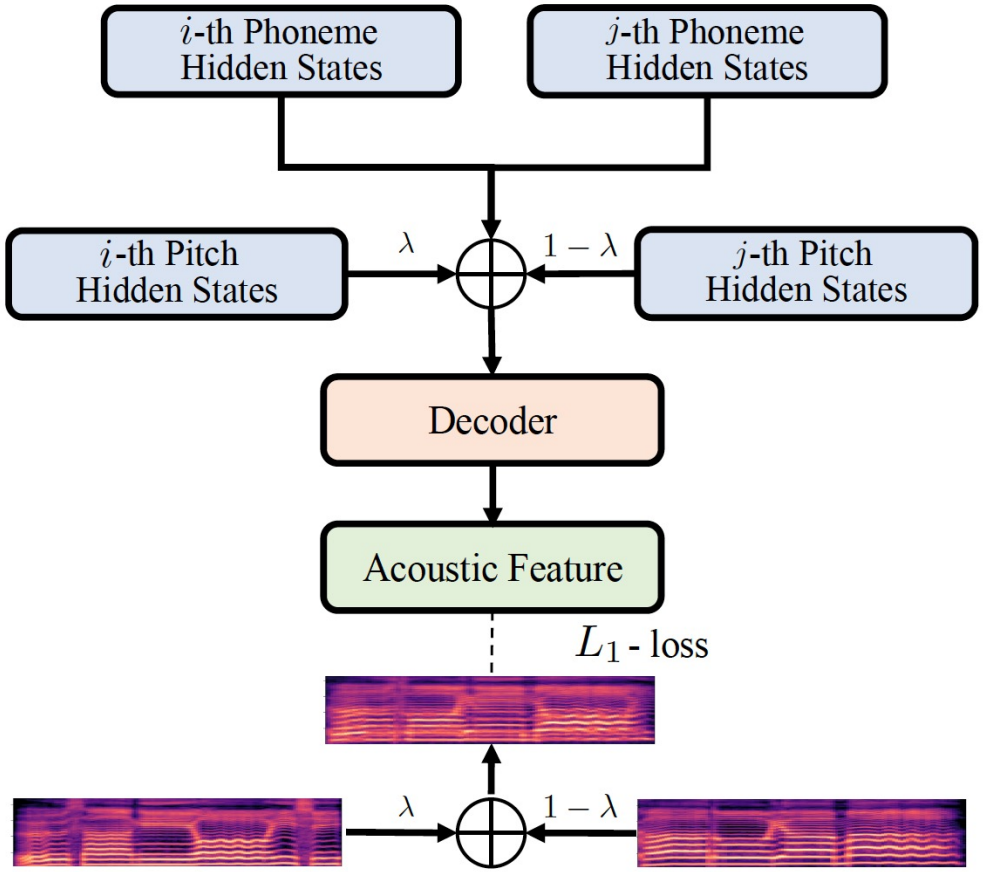
The Architecture of the base SVS System:

blue blocks: the model inputs;

yellow blocks: the embedding layers;

orange blocks: the main components of the SVS system;

green blocks: the model outputs;



References:

1.L. Meng, J. Xu, X. Tan, J.Wang, T. Qin, and B. Xu, “MixSpeech: Data augmentation for low-resource automatic speech recognition,” in ICASSP, 2021, pp. 7008–7012.

2.H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond Empirical Risk Minimization,” in ICLR, 2018.

3.Y. Cheng, L. Jiang, W. Macherey, and J. Eisenstein, “Advaug: Robust adversarial augmentation for neural machine translation,” in ACL, 2020, pp. 5961–5970.

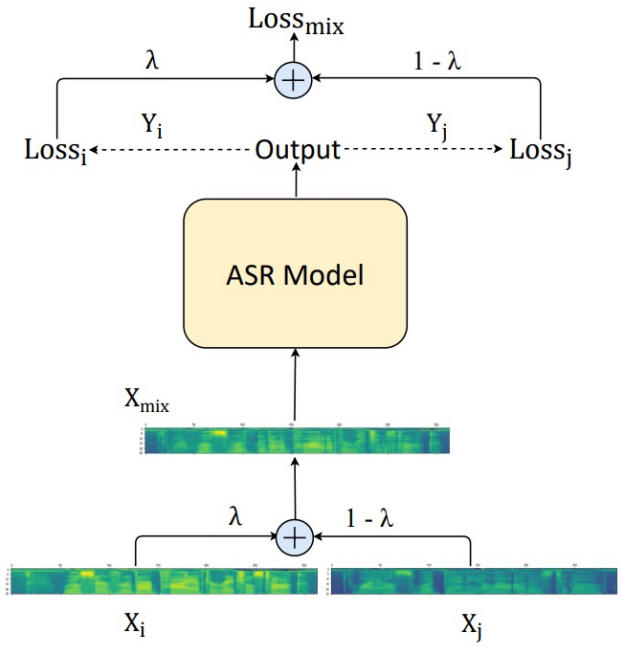
Mix-up Augmentation (Cont'd)

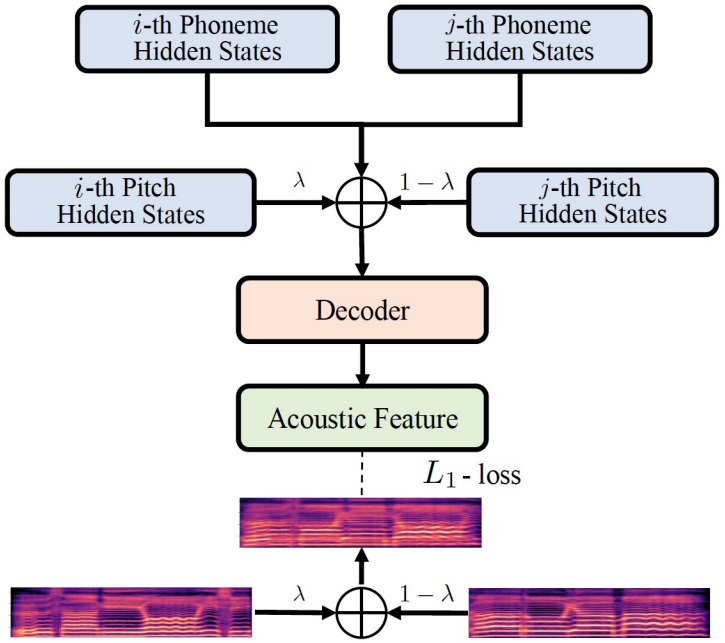
- Mix-up of hidden states $$H_{\text{mix}} = \lambda * H_i + (1 - \lambda) * H_j$$
- Mixture loss $$\mathcal{L}_{\text{mix}} = \lambda * L_1(\hat{Y}_{\text{mix}}, Y_i) + (1 - \lambda) * L_1(\hat{Y}_{\text{mix}}, Y_j)$$
- SVS loss with mix-up $$\mathcal{L}_{\text{svs}} = (1 - w_{\text{mix}}) * \mathcal{L}_{\text{svs}}^{\text{ori}} + w_{\text{mix}} * \mathcal{L}_{\text{mix}}.$$

Variable Description:

H is the sum of expanded hidden states H^{ph}, H^{ph} and positional information;
λ is the mix-up weight and follows $Beta(\alpha, \alpha)$;
$\hat{Y}_{\text{mix}}$ denotes the acoustic feature predicted from H_{mix};
w_{mix} is the weight of mixture loss;

The Difference of Our MA to Others





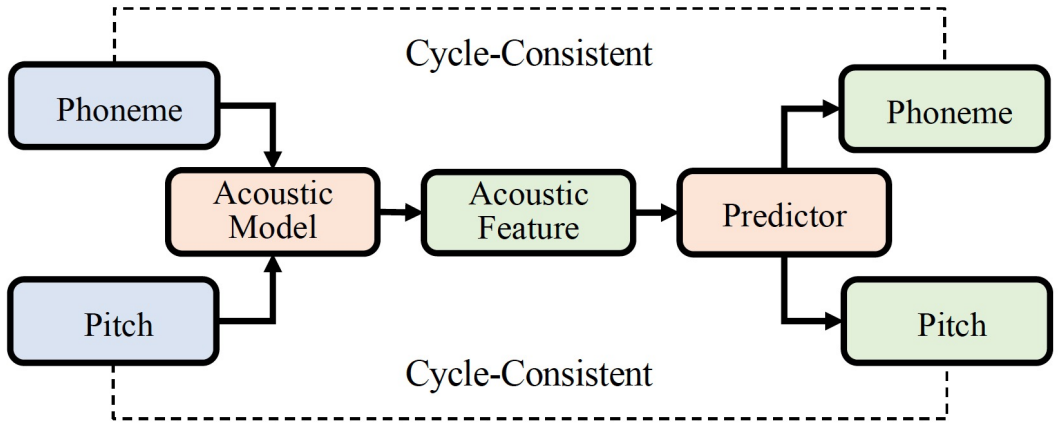
Meng, L., Xu, J., Tan, X., Wang, J., Qin, T., & Xu, B. (2021, June). MixSpeech: Data augmentation for low-resource automatic speech recognition. In *ICASSP.* (pp. 7008-7012).

Cycle-consistent training strategy (CC)

$$\mathcal{L}_{\text{si}} = \text{CrossEntropy}\left(g(Y;\theta_{yx}), X\right)$$

$$\mathcal{L}_{\text{pd}} = \text{CrossEntropy}\left(g(f(X;\theta_{xy});\theta_{yx}), X\right)$$

$$\mathcal{L} = w_{\text{svs}} * \mathcal{L}_{\text{svs}} + w_{\text{si}} * \mathcal{L}_{\text{si}} + w_{\text{pd}} * \mathcal{L}_{\text{pd}}$$



Variable Description:

θ_{yx} refers to the trainable parameters of the predictor module;

$g(Y;\theta_{yx})$ denotes the predictor module;

Some Experimental setups

Dataset – Ofuton:

- a male singer
- 56 Japanese songs, 61 mins in total
- 547 training, 58 validation, 70 testing

Dataset – Opencpop:

- a female singer
- 100 popular Mandarin songs (5.2 hours)
- follow the official split

Data processing:

- down-sample to 24k HZ
- 12.5 ms frame-shift & 50 ms frame-length

Augmentation Settings:

- PA: {-1, 0, 1}, MA proportion: 0.15
- $w_{mix} = 0.1$
- CC weight
  - Ofuton: 0.7, 0.2, 0.1
  - Opencpop: 0.85, 0.1, 0.05

Results on Ofuton

Dataset	Method	MCD↓	LF0-RMSE↓	F0-CORR↑	ST_ACC↑	VUV_ERR↓	MOS↑
Ofuton	Baseline	6.88	0.103	0.75	69.74	2.24	2.65 ± 0.08
	+ PA	6.55	0.106	0.86	71.09	2.38	2.87 ± 0.09
	+ MA	6.65	0.101	0.75	70.29	**2.07**	2.89 ± 0.09
	+ PA&MA	6.55	0.107	0.87	71.74	2.32	2.98 ± 0.09
	+ PA&MA&CC	**6.44**	**0.100**	**0.91**	**71.88**	2.45	**3.24 ± 0.09**
	G.T.	-	-	-	-	-	4.60 ± 0.07

- Compared to Baseline

  PA, MA, PA&MA&CC improve the performance on both subjective and objective scores

Evaluation on Opencpop

Dataset	Method	MCD↓	LF0-RMSE↓	F0-CORR↑	ST_ACC↑	VUV_ERR↓	MOS↑
Opencpop	Baseline	8.15	0.226	0.87	**74.46**	6.04	2.89 ± 0.11
	+ PA	7.85	0.219	0.87	74.37	**5.68**	3.06 ± 0.11
	+ MA	8.11	0.241	0.88	74.41	6.43	2.92 ± 0.10
	+ PA&MA	8.23	0.265	**0.89**	73.40	6.59	2.43 ± 0.09
	+ PA&MA&CC	**7.76**	**0.214**	0.86	74.30	5.71	**3.26 ± 0.11**
	G.T.	-	-	-	-	-	4.60 ± 0.09

- Compared to Baseline

  PA, MA, PA&MA&CC improve the performance on both subjective and objective scores
- More results fluctuations in F0-based measures
- Compared to single augmentation (PA or MA), PA & MA perform not good, but applying with CC reaches the best MOS

Evaluation on Opencpop

Dataset	Method	MCD↓	LF0-RMSE↓	F0-CORR↑	ST_ACC↑	VUV_ERR↓	MOS↑
Opencpop	Baseline	8.15	0.226	0.87	**74.46**	6.04	2.89 ± 0.11
	+ PA	7.85	0.219	0.87	74.37	**5.68**	3.06 ± 0.11
	+ MA	8.11	0.241	0.88	74.41	6.43	2.92 ± 0.10
	+ PA&MA	8.23	0.265	**0.89**	73.40	6.59	2.43 ± 0.09
	+ PA&MA&CC	**7.76**	**0.214**	0.86	74.30	5.71	**3.26 ± 0.11**
	G.T.	-	-	-	-	-	4.60 ± 0.09

- Compared to Baseline

  PA, MA, PA&MA&CC improve the performance on both subjective and objective scores
- More results fluctuations in F0-based measures
- Compared to single augmentation (PA or MA), PA & MA perform not good, but applying with CC reaches the best MOS

Evaluation on Opencpop

Dataset	Method	MCD↓	LF0-RMSE↓	F0-CORR↑	ST_ACC↑	VUV_ERR↓	MOS↑
Opencpop	Baseline	8.15	0.226	0.87	**74.46**	6.04	2.89 ± 0.11
	+ PA	7.85	0.219	0.87	74.37	**5.68**	3.06 ± 0.11
	+ MA	8.11	0.241	0.88	74.41	6.43	2.92 ± 0.10
	+ PA&MA	8.23	0.265	**0.89**	73.40	6.59	2.43 ± 0.09
	+ PA&MA&CC	**7.76**	**0.214**	0.86	74.30	5.71	**3.26 ± 0.11**
	G.T.	-	-	-	-	-	4.60 ± 0.09

- Compared to Baseline

  PA, MA, PA&MA&CC improve the performance on both subjective and objective scores
- More results fluctuations in F0-based measures
- Compared to single augmentation (PA or MA), PA & MA perform not good, but applying with CC reaches the best MOS

Ablation Studies (PA Shifting Factor)

Method	MCD↓	ST_ACC↑	VUV_ERR↓
P-adaptive	8.26	73.29	6.77
P1	**7.85**	**74.37**	**5.68**
P2	8.08	73.92	6.06

- P1: {-1, 0, 1}, P2: {-2, -1, 0, 1, 2}
- P1 achieves the best
- Too much variation in PA may reduce the performance

Ablation Studies (Different weights in MA)

Method	MCD↓	ST_ACC↑	VUV_ERR↓
$w_{mix} = 0.1$	**8.11**	**74.41**	6.43
$w_{mix} = 0.2$	8.15	73.60	6.43
$w_{mix} = 0.3$	8.20	74.25	**6.36**

- Large weights in MA may reduce the performance

Ablation Studies (Different weights in CC)

- Different weights in Cycle-consistent training strategy
  - CC1 - $w_{svs} = 0.7, \quad w_{si} = 0.2, \quad w_{pd} = 0.1$
  - CC2 - $w_{svs} = 0.85, w_{si} = 0.1, \quad w_{pd} = 0.05$
  - CC3 - $w_{svs} = 1, \quad w_{si} = 1, \quad w_{pd} = 1$

Method	MCD↓	ST_ACC↑	VUV_ERR↓
CC1 + PA + MA	8.25	73.89	6.62
CC2 + PA + MA	**7.76**	**74.30**	**5.71**
CC3 + PA + MA	8.71	73.83	6.45

Ablation Studies (Augmentation Methods)

Method	MCD↓	ST_ACC↑	VUV_ERR↓
CC2	8.06	73.67	6.63
CC2 + PA	8.11	73.46	6.44
CC2 + MA	8.04	74.28	6.20
CC2 + PA + MA	**7.76**	**74.30**	**5.71**

- The methods with CC improve the MCD metric regardless of whether PA or MA is used
- CC+PA+MA could bring further improvements on synthetic quality

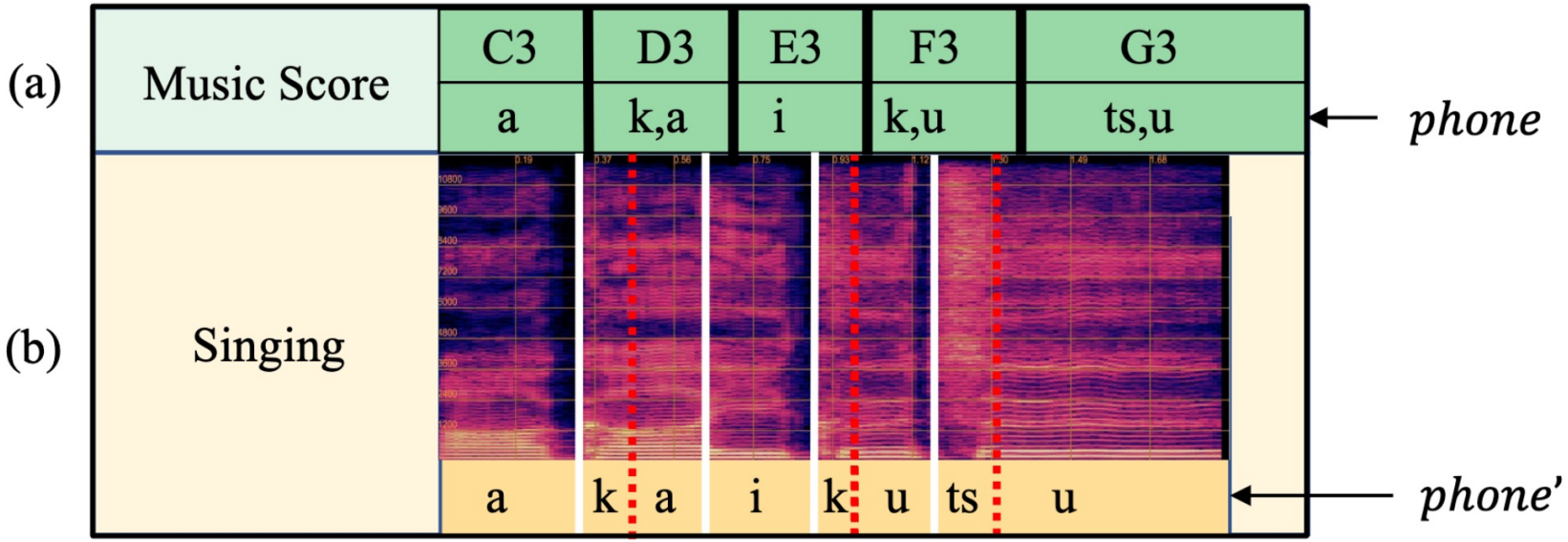
The Difference between SVS and TTS?

- TTS
  - A relatively less-constraint generation
    - no strict duration control
    - no strict pitch control
    - less concern to high-frequency spectrum
- SVS
  - A need to follow the music score
    - music score has clear duration information
    - explicit pitch requirement
    - need high-frequency to make the song sounding

The Difference between SVS and TTS?

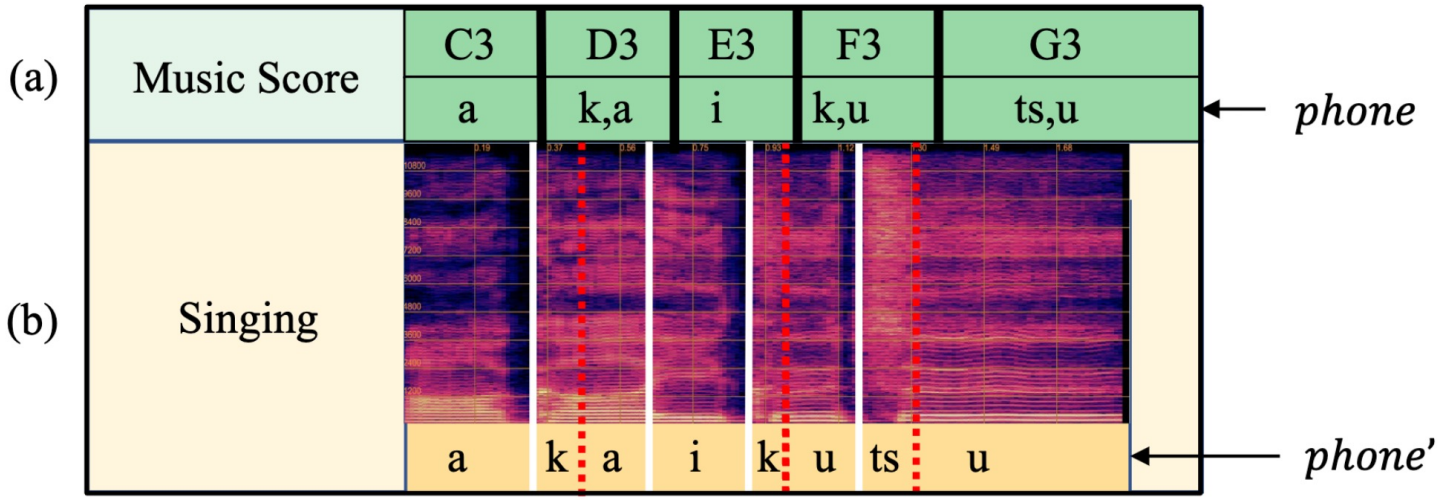
- TTS
  - A relatively less-constraint generation
    - no strict duration control
    - no strict pitch control
    - less concern to high-frequency spectrum
- SVS
  - A need to follow the music score
    - music score has clear duration information
    - explicit pitch requirement
    - need high-frequency to make the song sounding
  - (The singers still have their own choices of flexibility)

Example of SVS



(White bold lines indicate syllable splits and red dotted lines indicate phoneme splits in each syllable)

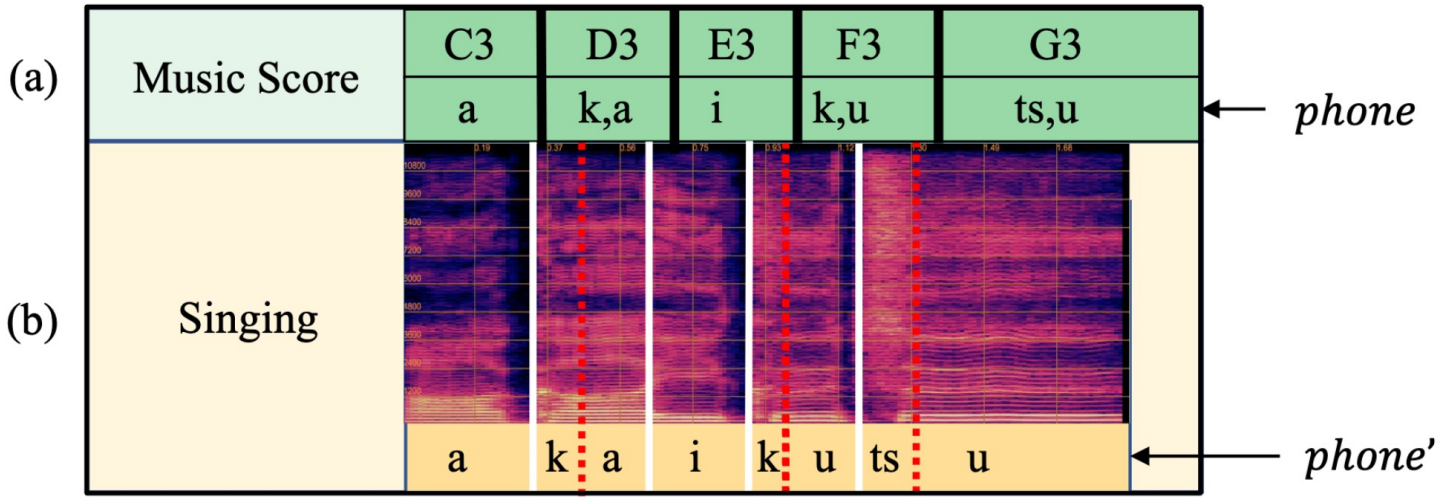
How Current SVS Systems Handle the Features



Type 1: use the music score as pitch and duration input (i.e., Fig. (a)) and then apply a duration predictor to generate phoneme-level duration (can be obtained by force-alignment/heuristic rules) for further acoustic modeling.

Issues: cannot adapt to the mismatch from the singer's voice, as it usually cannot strictly follow the score

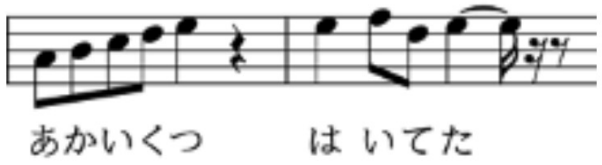
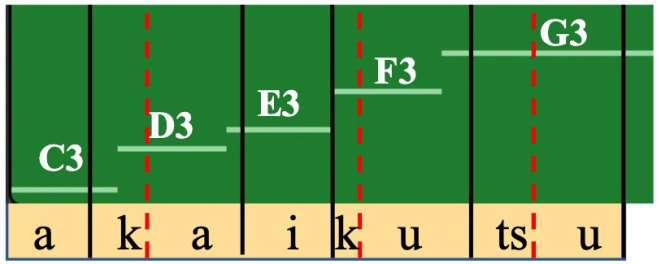
How Current SVS Systems Handle the Features



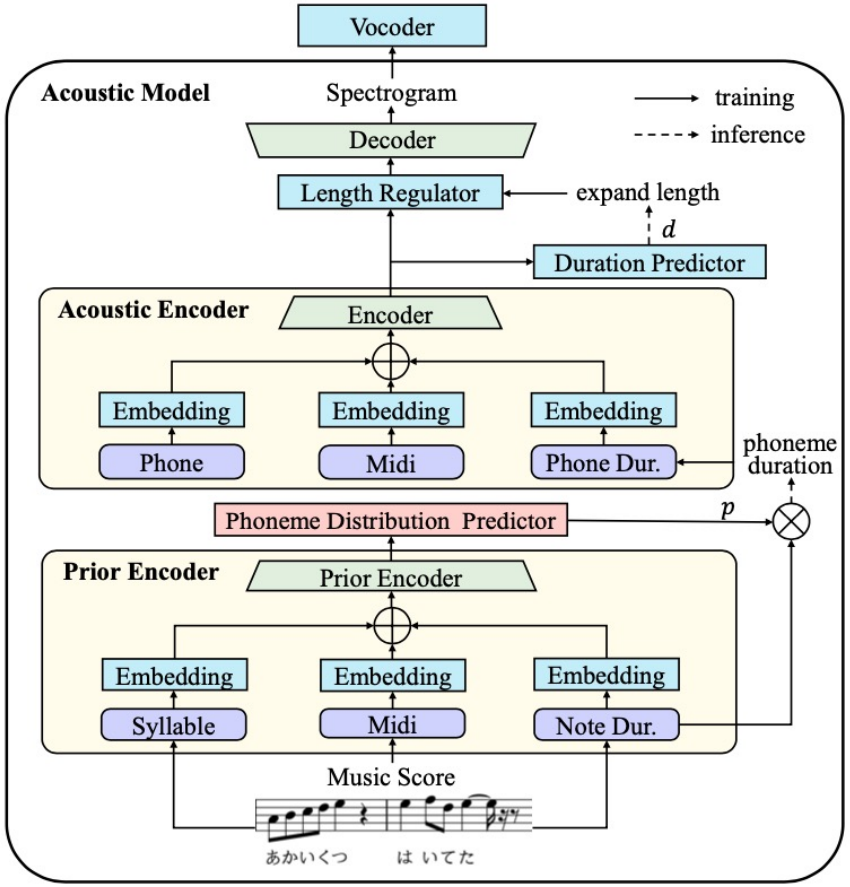
Type 2: use the labeled phoneme duration sequence (i.e., figure (b) phone') and the note sequence of music score (i.e., figure (a))

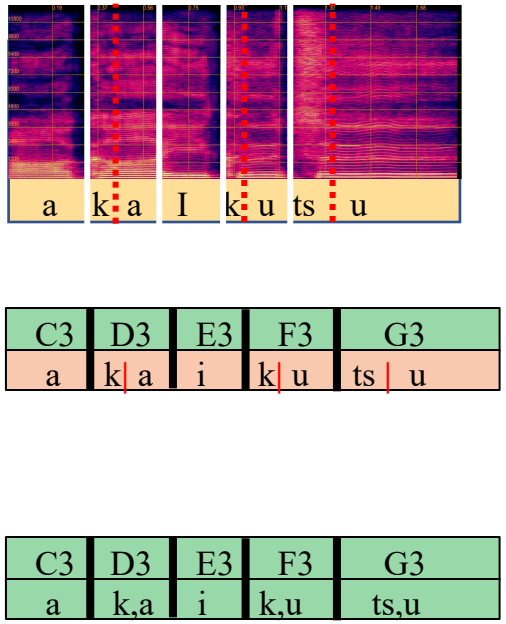
Issues: cause mismatched alignment between music score and phoneme sequence

Comparison of Type1 and Type2

AFP Strategy		*Type 1*	*Type 2*
Raw Data			
Feats. Input	Frame Level	C3 | D3 | E3 | F3 | G3 a | k,a | i | k,u | ts,u	C3 | D3 | E3 | F3 | G3 a | k a | i | k u | ts u
	Phone Level	[C3, D3, D3, E3, F3, F3, G3, G3] [a, k, a, i, k, u, ts, u]	[C3, C3, D3, D3, E3, F3, G3, G3, G3] [a, k, k, a, i, u, u, ts, u]
Length Regulator	GT Input	a | k | a | i | k | u | ts | u obtained by HMM-based forced alignment	a | k a | i | k u | ts u guided by annotation

PHONEix


Vocoder
Acoustic Model
Spectrogram
training
inference
Decoder
Length Regulator
expand length
d
Duration Predictor
Acoustic Encoder
Encoder
Embedding
Embedding
Embedding
Phone
Midi
Phone Dur.
phoneme duration
Phoneme Distribution Predictor
p
Prior Encoder
Prior Encoder
Embedding
Embedding
Embedding
Syllable
Midi
Note Dur.
Music Score
あかいくつ
は いてた

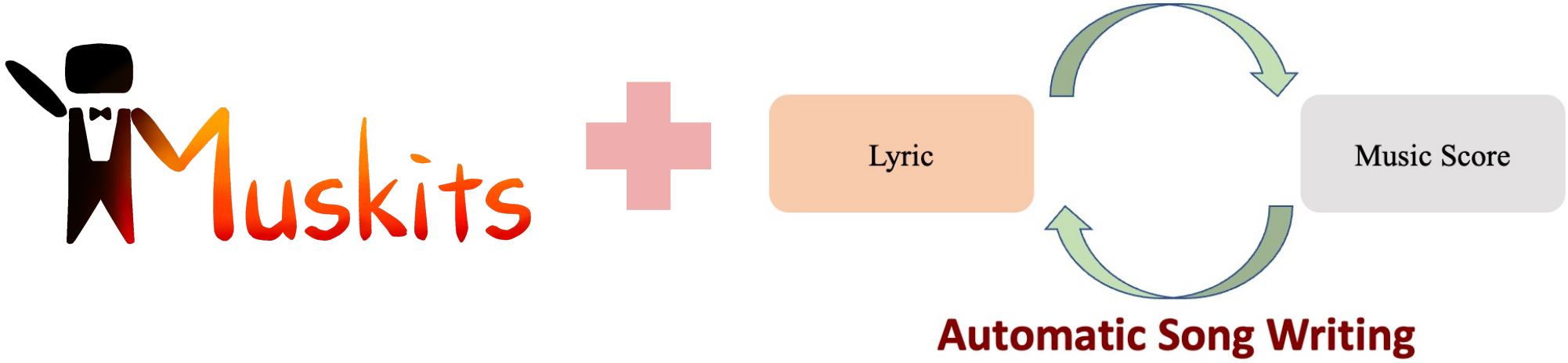

a k a I k u ts u
C3 D3 E3 F3 G3
a k a i k u ts u
C3 D3 E3 F3 G3
a k,a i k,u ts,u

Experiments on Ofuton and Opencpop

Dataset	Model	Method	MCD ↓	VUV E ↓	SA ↑	MOS ↑
Ofuton	LSTM	***Type 1***	6.74	2.49%	56.65%	2.54 ± 0.05
		Type 2	6.34	2.53%	57.58%	2.58 ± 0.05
		PHONEix	**6.28**	**2.41%**	**61.17%**	**3.03** ± 0.06
	Transformer	***Type 1***	6.95	1.93%	61.09%	2.60 ± 0.05
		Type 2	6.78	**1.71%**	58.44%	2.26 ± 0.05
		PHONEix	**6.52**	2.15%	**62.47%**	**3.01** ± 0.06
		Ground Truth	-	-	-	4.47 ± 0.04
Opencpop	LSTM	***Type 1***	9.33	6.79%	46.92%	2.09 ± 0.05
		Type 2	8.62	11.32%	53.93%	2.48 ± 0.05
		PHONEix	**7.90**	**6.05%**	**60.97%**	**3.56** ± 0.06
	Transformer	***Type 1***	8.77	6.19%	52.16%	2.52 ± 0.05
		Type 2	9.05	10.59%	52.90%	1.89 ± 0.04
		PHONEix	**8.42**	**5.97%**	**60.47%**	**3.03** ± 0.05
		Ground Truth	-	-	-	4.72 ± 0.03

Future Works

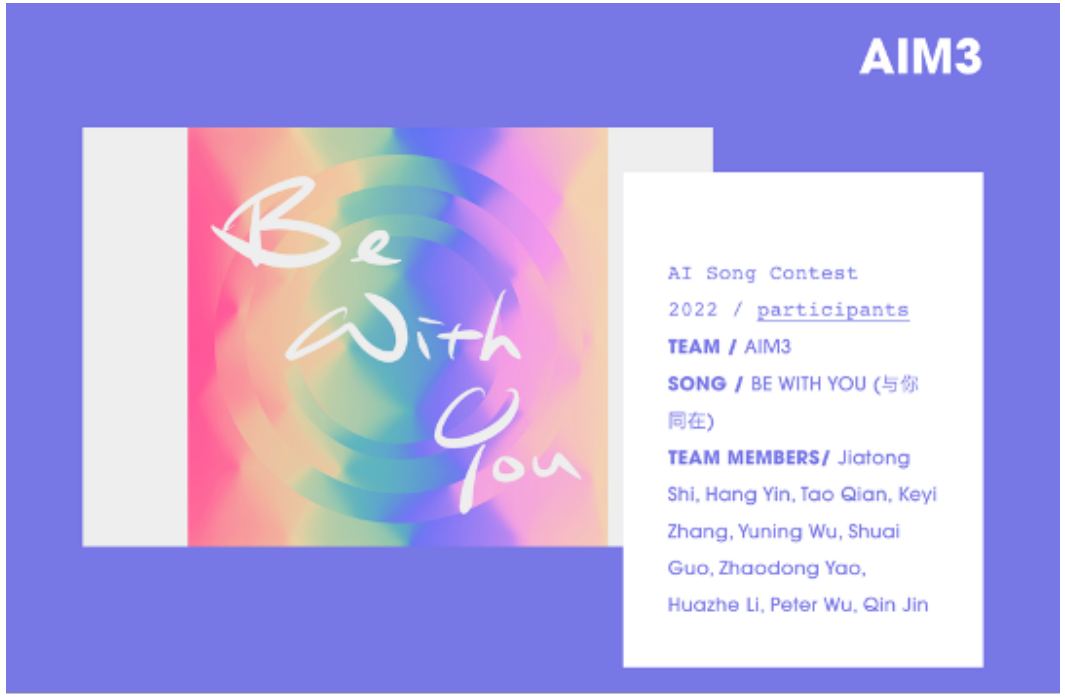
- Muskits → not limited to SVS but other music tasks
  - music generation, music transcription, automatic song writing, etc.



Future Works

- Modeling Advancement
  - More acoustic models
  - More vocoders
  - Joint-training of acoustic model and vocoders
- ESPnet integration
  - We already finished our very initial merge!
  - Take benefit from software supports (CI testing, strong maintaining teams)
  - Take benefits from various speech processing method

Be with you – AI Song Contest



Our Demo Page:
https://sjtmusicteam.github.io/MuskitsPage/

More teams:
https://www.aisongcontest.com/participants-2022

Thanks for listening!