Self-supervised learning and universal modeling for speech and audio processing

Benchmarking and open-source toolkits

Panelists

Shinji Watanabe

- Associate Professor
- Carnegie Mellon University

Titouan Parcollet

- Associate Professor
- Avignon Université

Shang-Wen Li (Daniel)

- Engineering and Science Manager
- Facebook AI

Mirco Ravanelli

- Post-doc Researcher
- MILA, Université de Montréal

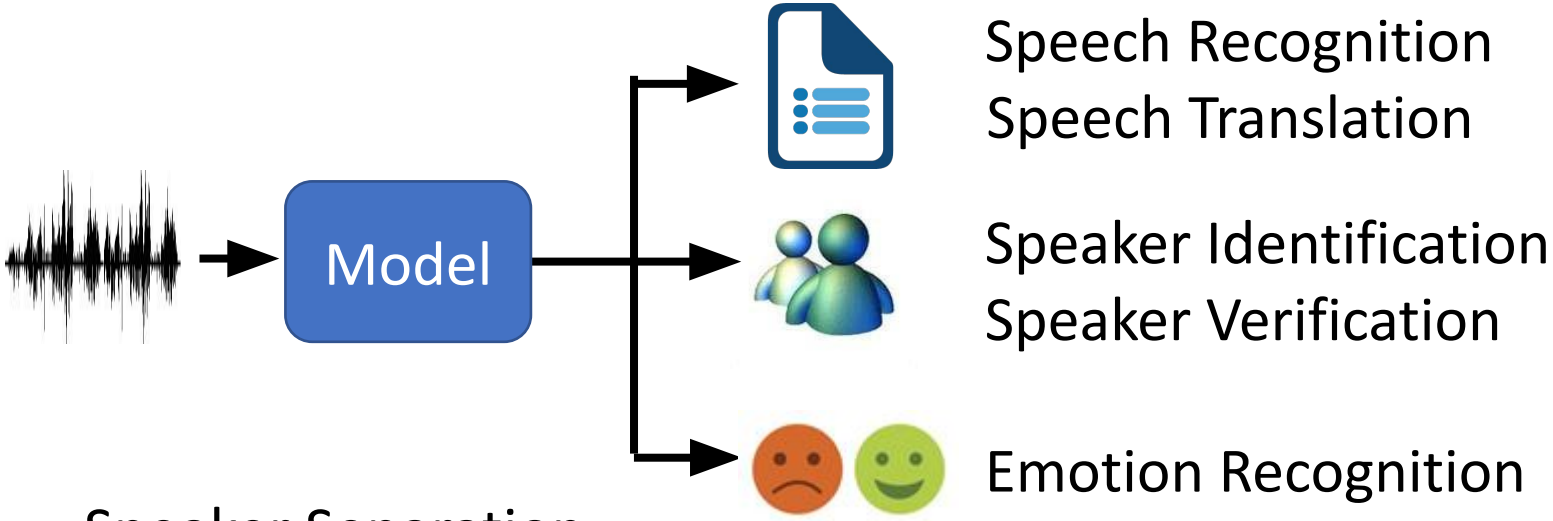
Speech Processing

Week 2

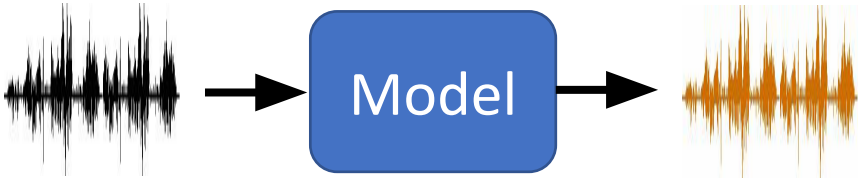
Week 2					
Date	2021/8/9	2021/8/10	2021/8/11	2021/8/12	2021/8/13
Weekday	Mon	Tue	Wed	Thur	Fri
09:00-10:00 (GMT+8)		**Speaker:** Ming-Wei Chang **Title:** Pre-training for Natural Language Processing [Lecture Info] [Course Link]	**Speaker:** Philipp Krähenbuhl **Title:** Computer Vision [Lecture Info] [Course Link]	Poster Session 2 [Poster]	
10:00-10:30 (GMT+8)					
10:30-11:00 (GMT+8)					**Speaker:** Been Kim **Title:** Interpretable machine learning [Lecture Info] [Course Link]
11:00-12:00 (GMT+8)					
12:00-20:00 (GMT+8)	Break				
20:00-21:00 (GMT+8)					
21:00-22:00 (GMT+8)	**Speaker:** John Shawe-Taylor **Title:** An introduction to Statistical Learning Theory and PAC-Bayes Analysis [Lecture Info] [Course Link]	**Speaker:** Hung-Yi Lee **Title:** Deep Learning for Speech Processing [Lecture Info] [Course Link]	**Speaker:** Song Han **Title:** TinyML and Efficient Deep Learning [Lecture Info] [Course Link]	**Speakers:** Thang Vu, Shang-Wen Li **Title:** Meta Learning for Human Language Processing [Lecture Info1] [Lecture Info2] [Course Link]	**Speaker:** Srinivasan Arunachalam **Title:** Overview of learning quantum states [Lecture Info] [Course Link]
22:00-23:00 (GMT+8)					

What we have learned

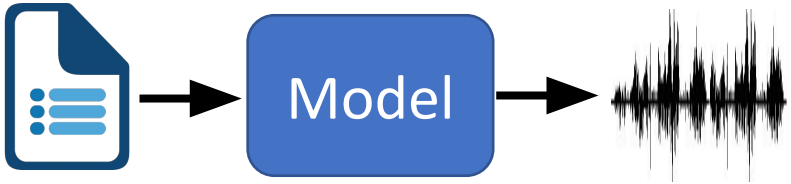
How to implement the models?



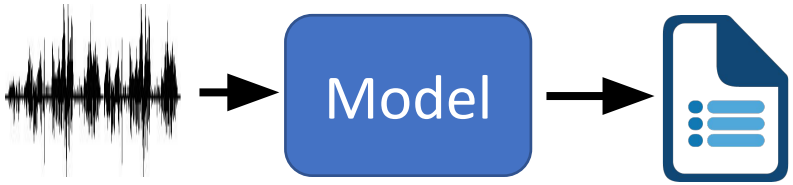
Speaker Separation
Voice Conversion



Text-to-Speech

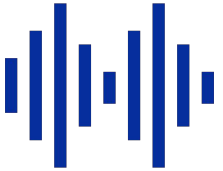


Lack of Training Data



Impossible to have a large amount of paired data for all tasks in all languages

“hello”



“how are you”

“good morning”

100,000 hours of paired data

Babies learn their first language with little supervision.

Can AI have the same ability?

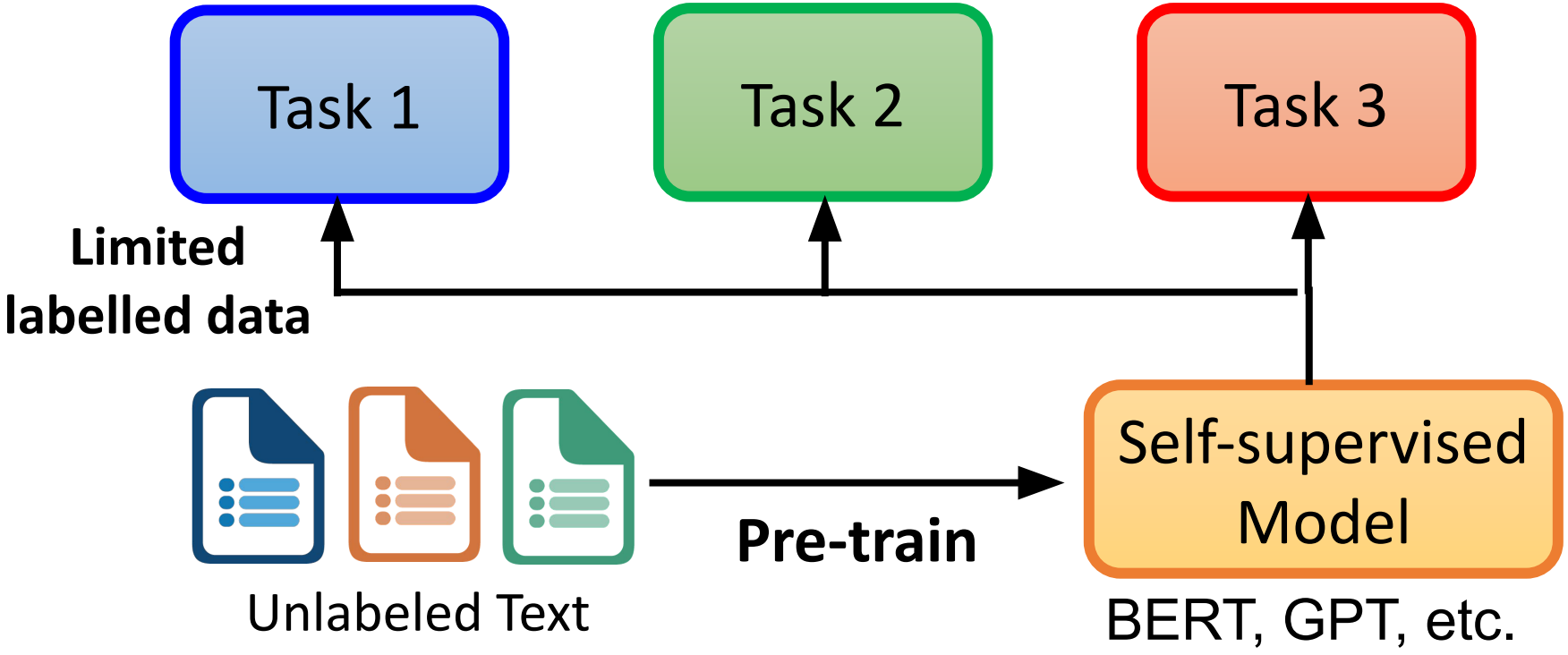
Self-supervised learning / Pre-training

Week 2

Week 2					
Date	2021/8/9	2021/8/10	2021/8/11	2021/8/12	2021/8/13
Weekday	Mon	Tue	Wed	Thur	Fri
09:00-10:00 (GMT+8)		**Speaker:** Ming-Wei Chang **Title:** Pre-training for Natural Language Processing Lecture Info Course Link	**Speaker:** Philipp Krähenbuhl **Title:** Computer Vision Lecture Info Course Link	Poster Session 2 Poster	
10:00-10:30 (GMT+8)					
10:30-11:00 (GMT+8)					**Speaker:** Been Kim **Title:** Interpretable machine learning Lecture Info Course Link
11:00-12:00 (GMT+8)					
12:00-20:00 (GMT+8)	Break				
20:00-21:00 (GMT+8)					
21:00-22:00 (GMT+8)	**Speaker:** John Shawe-Taylor **Title:** An introduction to Statistical Learning Theory and PAC-Bayes Analysis Lecture Info Course Link	**Speaker:** Hung-Yi Lee **Title:** Deep Learning for Speech Processing Lecture Info Course Link	**Speaker:** Song Han **Title:** TinyML and Efficient Deep Learning Lecture Info Course Link	**Speakers:** Thang Vu, Shang-Wen Li **Title:** Meta Learning for Human Language Processing Lecture Info1 Lecture Info2 Course Link	**Speaker:** Srinivasan Arunachalam **Title:** Overview of learning quantum states Lecture Info Course Link
22:00-23:00 (GMT+8)					

What we have learned

How about speech?



Outline

- Part I
  - Benchmarking for Self-supervised Learning
    - SUPERB (speaker: Hung-yi Lee)
    - LeBenchmark (speaker: Titouan Parcollet)
  - Open-source toolkits for wide variety of tasks
    - Speech Brain (speaker: Mirco Ravanelli)
    - ESPNet (speaker: Shinji Watanabe)
  - Type your questions during the talks.
- Part II:
  - We will discuss your questions.

Benchmarking for Self-supervised Learning

To Learn More .....

- SUPERB
- LeBenchmark

} uses speech representations in a wide variety of downstream tasks

- Zero Speech → no linguistic labels available

Speech

- HEAR → Audio (music, sound event, etc.)

Zero Speech: https://www.zerospeech.com/

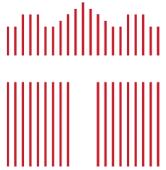
HEAR: https://neuralaudio.ai/hear2021-holistic-evaluation-of-audio-representations.html

Benchmarking for Self-supervised Learning

SUPERB

SUPERB

Speech processing **U**niversal **PER**formance **B**enchmark



National Taiwan University 國立臺灣大學

Carnegie Mellon University

Massachusetts Institute of Technology

JOHNS HOPKINS UNIVERSITY

FACEBOOK AI

SUPERB: Speech processing Universal PERformance Benchmark

Shu-wen Yang[1], *Po-Han Chi*[1*], *Yung-Sung Chuang*[1*], *Cheng-I Jeff Lai*[2*], *Kushal Lakhotia*[3*], *Yist Y. Lin*[1*], *Andy T. Liu*[1*], *Jiatong Shi*[4*], *Xuankai Chang*[6], *Guan-Ting Lin*[1], *Tzu-Hsien Huang*[1], *Wei-Cheng Tseng*[1], *Ko-tik Lee*[1], *Da-Rong Liu*[1], *Zili Huang*[4], *Shuyan Dong*[5†], *Shang-Wen Li*[5†], *Shinji Watanabe*[6], *Abdelrahman Mohamed*[3], *Hung-yi Lee*[1]

Will be published at INTERSPEECH 2021

Typical Self-supervised Learning for Speech

Speech Recognition

2. More downstream tasks?

1. No side-by-side comparison

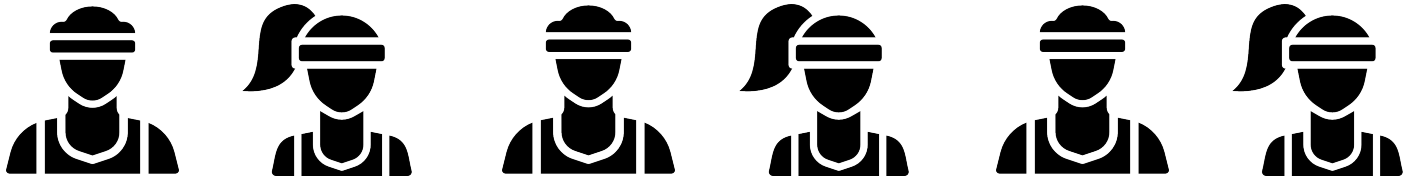
Unlabeled Speech

Pre-train

Self-supervised Model

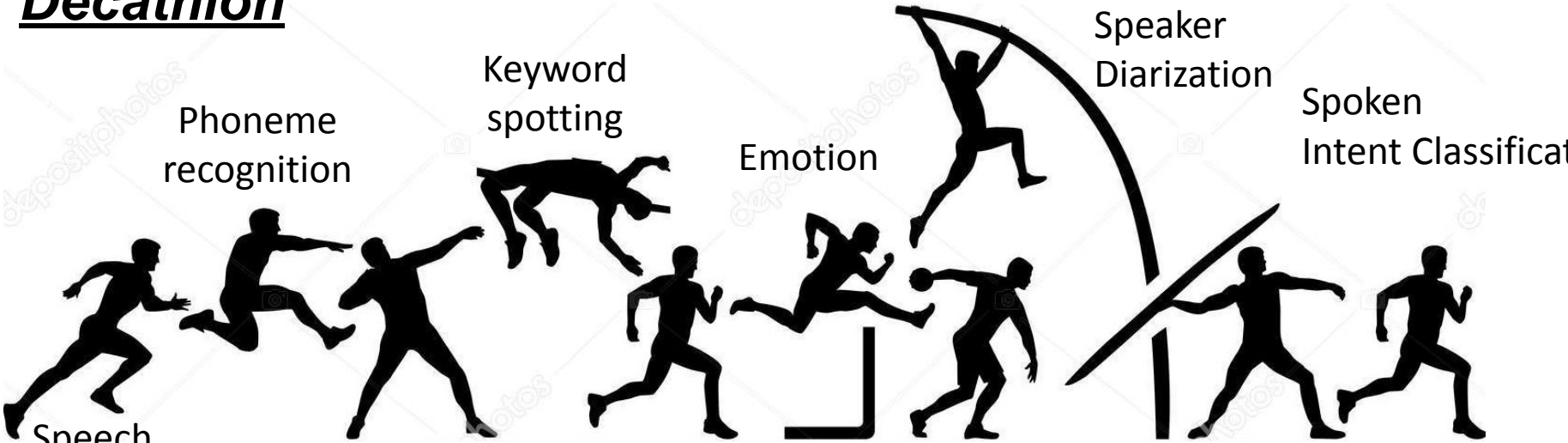
CPC, PASE, APC, Mockingjay,
wav2vec series, HuBERT series, etc

SUPERB



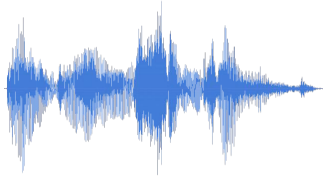
Self-supervised Speech Models

Decathlon



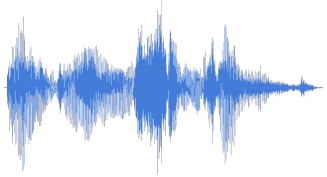
Content

Phoneme Recognition



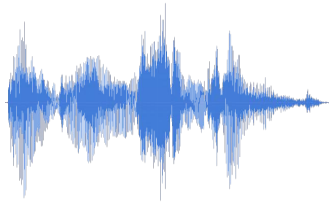
transcribe → /b/ /d/ /f/ /g/ ...

Keyword Spotting



classify → **Left / Right / Go ...**

Speech Recogition

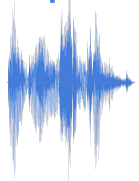


transcribe → **I want to pet a cat**

Query-by-Example

Document:
I joined MLSS 2021 Taipei.

Query:
"Tapei"

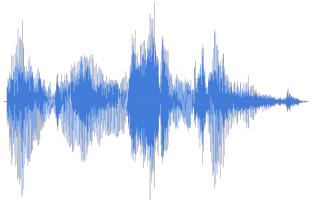


Query in Document?

Yes / No

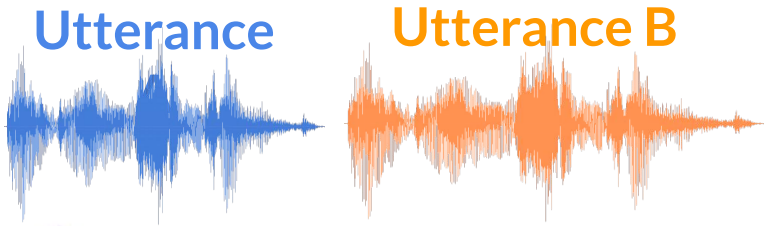
Speaker

Speaker Identification



classify → **Speaker ID**

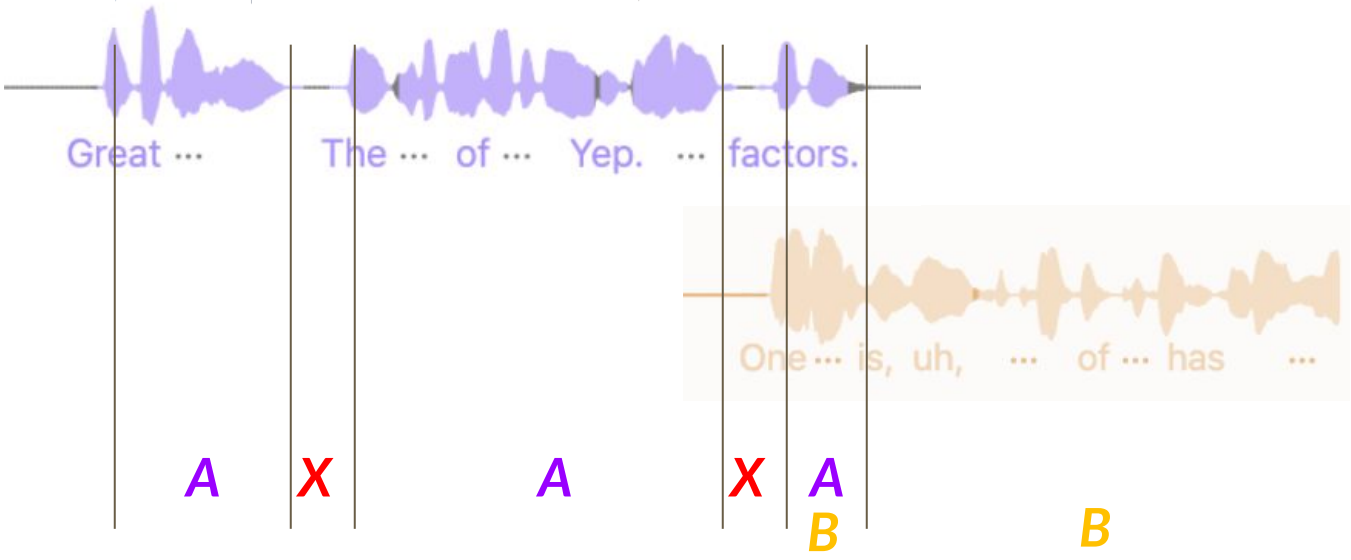
Speaker Verification



A & B *same speaker*?

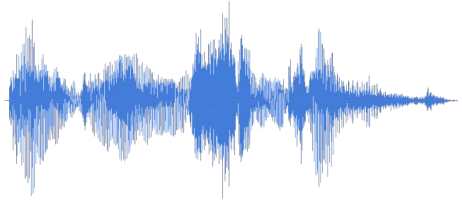
Yes / No

Speaker Diarization



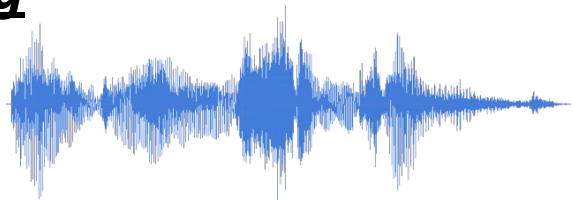
Semantic

Intent Classification



classify → **Intent classes**

Slot Filling



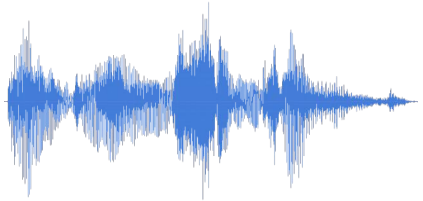
I fly from **Taipei** to **New York**

extract →

Slot type	*Slot value*
from_location	Taipei
to_location	New York

Emotion

Emotion Recognition



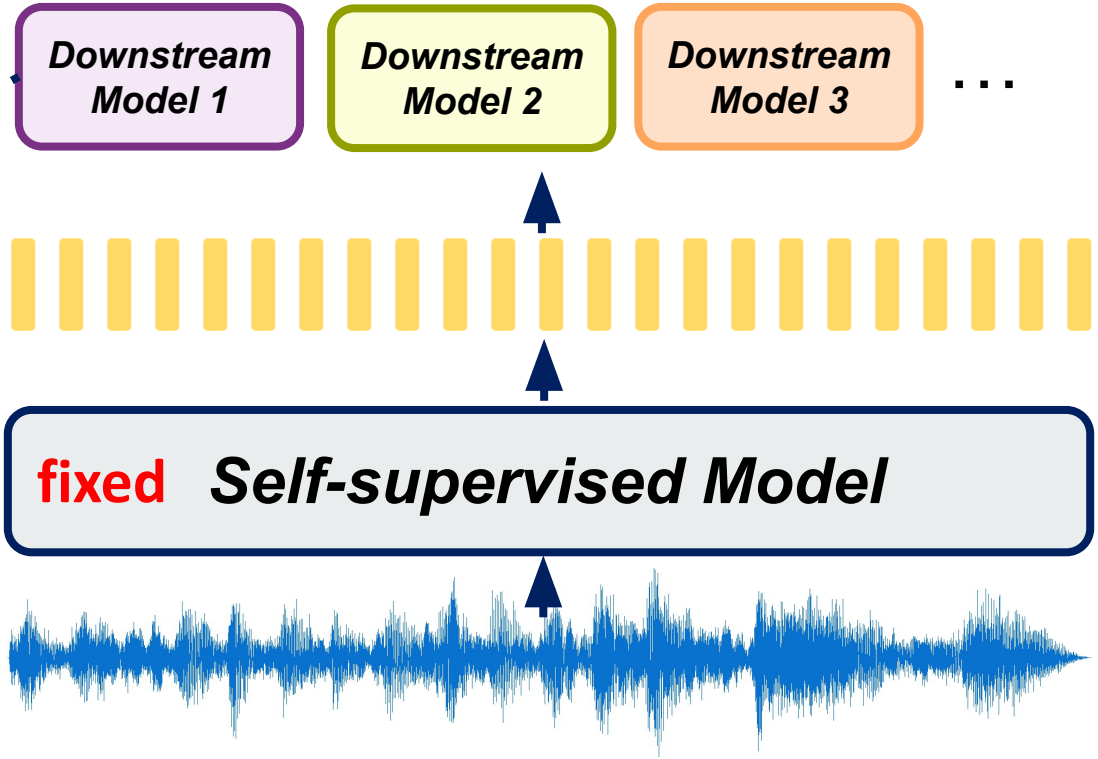
classify → **Happy / Angry / Sad .....**

How to use Self-supervised Model - Constrained Track

All the self-supervised models use the same network architecture.

Keep it simple

Shared across tasks



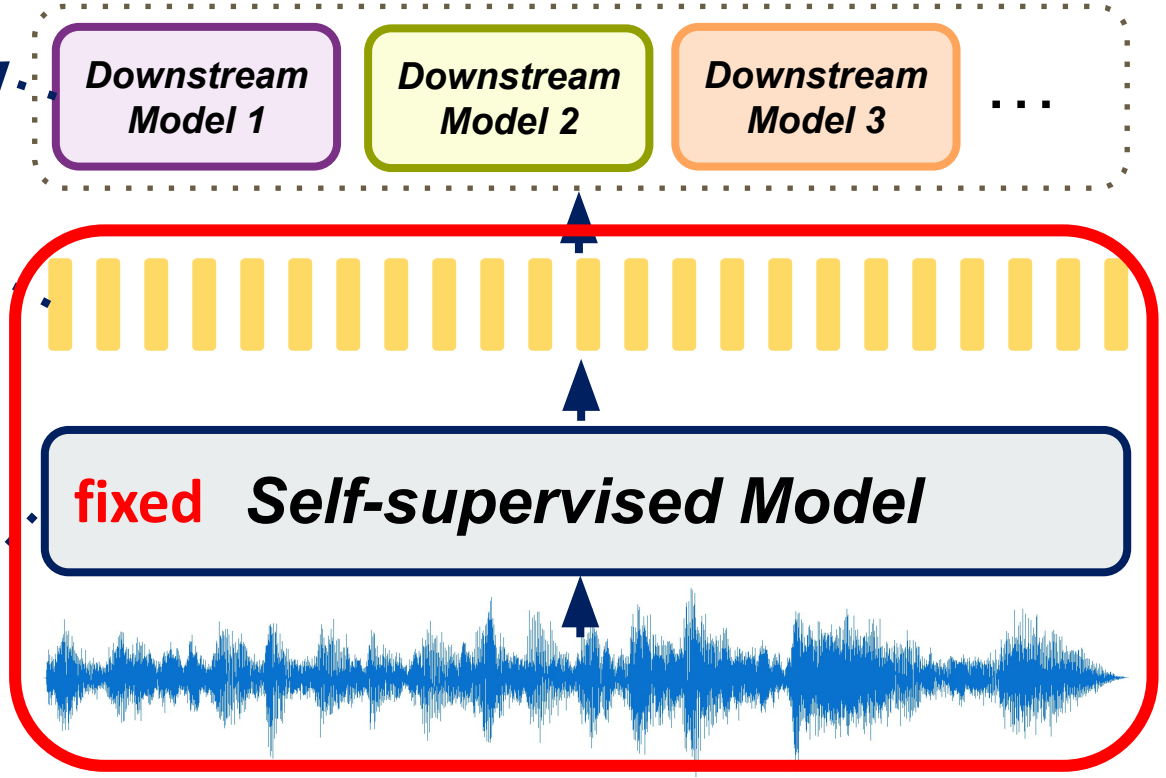
How to use Self-supervised Model - Constrained Track

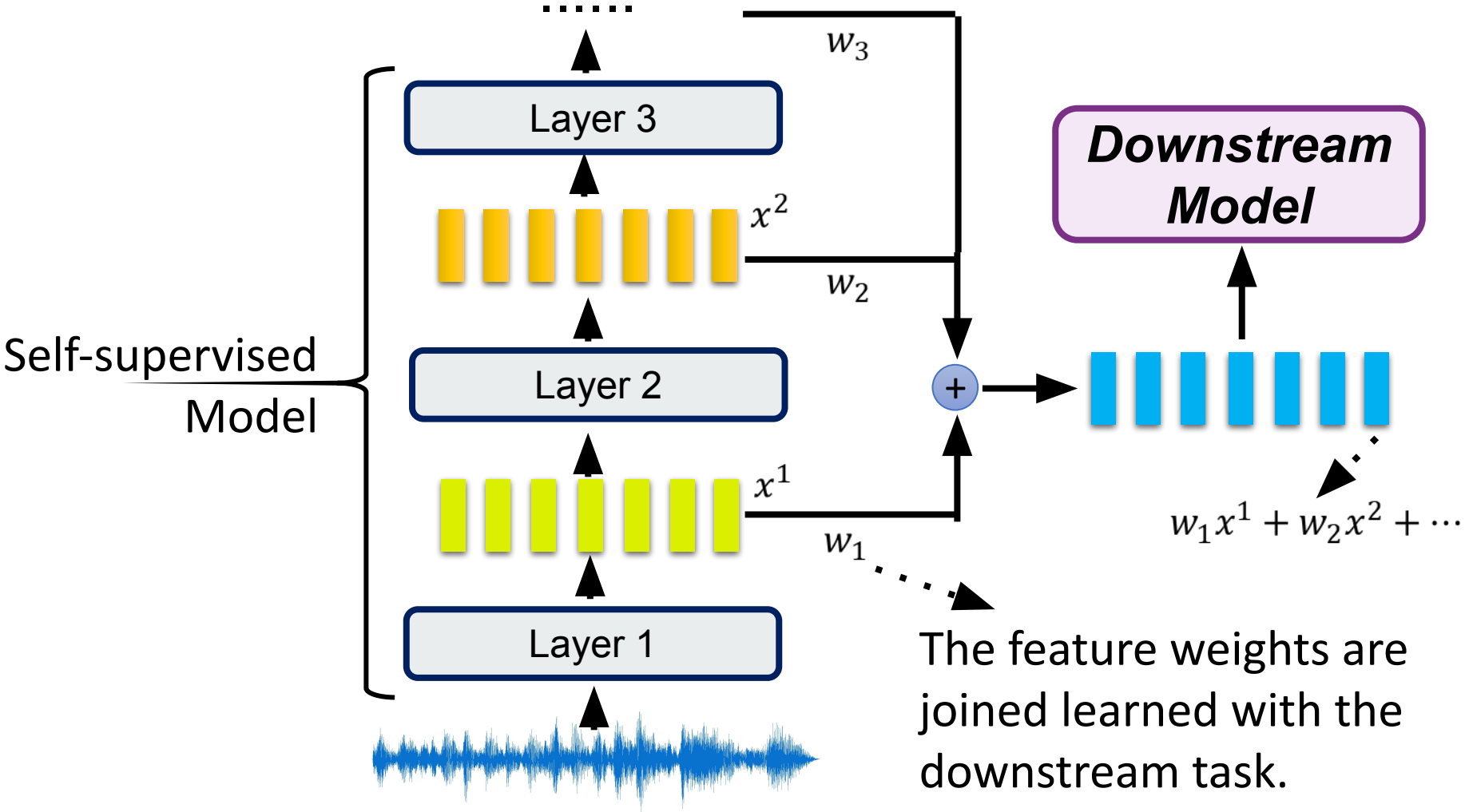
Limited capacity

Universal features

- General-purpose knowledge for speech processing, from acoustic signals to semantics
- Easy to add new tasks
- Storage saving

task independent




Self-supervised Model
Layer 3
Layer 2
Layer 1
x^2
x^1
w_3
w_2
w_1
+
Downstream Model
$w_1x^1 + w_2x^2 + \cdots$
The feature weights are joined learned with the downstream task.

Hand-crafted features

tasks (open data, easy to reproduce)

	PR	KS	IC	SID	ER	ASR (WER)		QbE	SF		ASV	SD
	PER ↓	Acc ↑	Acc ↑	Acc ↑	Acc ↑	w/o ↓	w/ LM ↓	MTWV ↑	F1 ↑	CER ↓	EER ↓	DER ↓
FBANK	82.01	8.63	9.10	8.5E-4	35.39	23.18	15.21	0.0058	69.64	52.94	9.56	10.05
PASE+ [16]	58.87	82.54	29.82	37.99	57.86	25.11	16.62	0.0072	62.14	60.17	11.61	8.68
APC [7]	41.98	91.01	74.69	60.42	59.33	21.28	14.74	0.0310	70.46	50.89	8.56	10.53
VQ-APC [32]	41.08	91.11	74.48	60.15	59.66	21.20	15.21	0.0251	68.53	52.91	8.72	10.45
NPC [33]	43.81	88.96	69.44	55.92	59.08	20.20	13.91	0.0246	72.79	48.44	9.4	9.34
Mockingjay [8]	70.19	83.67	34.33	32.29	50.28	22.82	15.48	6.6E-04	61.59	58.89	11.66	10.54
TERA [9]	49.17	89.48	58.42	57.57	56.27	18.17	12.16	0.0013	67.50	54.17	15.89	9.96
modified CPC [34]	42.54	91.88	64.09	39.63	60.96	20.18	13.53	0.0326	71.19	49.91	12.86	10.38
wav2vec [12]	31.58	95.59	84.92	56.56	59.79	15.86	11.00	0.0485	76.37	43.71	7.99	9.9
vq-wav2vec [13]	33.48	93.38	85.68	38.80	58.24	17.71	12.80	0.0410	77.68	41.54	10.38	9.93
wav2vec 2.0 Base [14]	5.74	96.23	92.35	75.18	63.43	6.43	4.79	0.0233	88.30	24.77	6.02	6.08
wav2vec 2.0 Large [14]	4.75	**96.66**	95.28	86.14	65.64	3.75	3.10	0.0489	87.11	27.31	5.65	**5.62**
HuBERT Base [35]	5.41	96.30	98.34	81.42	64.92	6.42	4.79	**0.0736**	88.53	25.20	**5.11**	5.88
HuBERT Large [35]	**3.53**	95.29	**98.76**	**90.33**	**67.62**	**3.62**	**2.94**	0.0353	**89.81**	**21.76**	5.98	5.75

Self-supervised Speech Model

	PR	KS	IC	SID	ER	ASR (WER)		QbE	SF		ASV	SD
	PER ↓	Acc ↑	Acc ↑	Acc ↑	Acc ↑	w/o ↓	w/ LM ↓	MTWV ↑	F1 ↑	CER ↓	EER ↓	DER ↓
FBANK	82.01	8.63	9.10	8.5E-4	35.39	23.18	15.21	0.0058	69.64	52.94	9.56	10.05
PASE+ [16]	58.87	82.54	29.82	37.99	57.86	25.11	16.62	0.0072	62.14	60.17	11.61	8.68
APC [7]	41.98	91.01	74.69	60.42	59.33	21.28	14.74	0.0310	70.46	50.89	8.56	10.53
VQ-APC [32]	41.08	91.11	74.48	60.15	59.66	21.20	15.21	0.0251	68.53	52.91	8.72	10.45
NPC [33]	43.81	88.96	69.44	55.92	59.08	20.20	13.91	0.0246	72.79	48.44	9.4	9.34
Mockingjay [8]	70.19	83.67	34.33	32.29	50.28	22.82	15.48	6.6E-04	61.59	58.89	11.66	10.54
TERA [9]	49.17	89.48	58.42	57.57	56.27	18.17	12.16	0.0013	67.50	54.17	15.89	9.96
modified CPC [34]	42.54	91.88	64.09	39.63	60.96	20.18	13.53	0.0326	71.19	49.91	12.86	10.38
wav2vec [12]	31.58	95.59	84.92	56.56	59.79	15.86	11.00	0.0485	76.37	43.71	7.99	9.9
vq-wav2vec [13]	33.48	93.38	85.68	38.80	58.24	17.71	12.80	0.0410	77.68	41.54	10.38	9.93
wav2vec 2.0 Base [14]	5.74	96.23	92.35	75.18	63.43	6.43	4.79	0.0233	88.30	24.77	6.02	6.08
wav2vec 2.0 Large [14]	4.75	**96.66**	95.28	86.14	65.64	3.75	3.10	0.0489	87.11	27.31	5.65	**5.62**
HuBERT Base [35]	5.41	96.30	98.34	81.42	64.92	6.42	4.79	**0.0736**	88.53	25.20	**5.11**	5.88
HuBERT Large [35]	**3.53**	95.29	**98.76**	**90.33**	**67.62**	**3.62**	**2.94**	0.0353	**89.81**	**21.76**	5.98	5.75

Self-supervised learning outperforms fbank in most cases.

	PR	KS	IC	SID	ER	ASR (WER)		QbE	SF		ASV	SD
	PER ↓	Acc ↑	Acc ↑	Acc ↑	Acc ↑	w/o ↓	w/ LM ↓	MTWV ↑	F1 ↑	CER ↓	EER ↓	DER ↓
FBANK	82.01	8.63	9.10	8.5E-4	35.39	23.18	15.21	0.0058	69.64	52.94	9.56	10.05
PASE+ [16]	58.87	82.54	29.82	37.99	57.86	25.11	16.62	0.0072	62.14	60.17	11.61	8.68
APC [7]	41.98	91.01	74.69	60.42	59.33	21.28	14.74	0.0310	70.46	50.89	8.56	10.53
VQ-APC [32]	41.08	91.11	74.48	60.15	59.66	21.20	15.21	0.0251	68.53	52.91	8.72	10.45
NPC [33]	43.81	88.96	69.44	55.92	59.08	20.20	13.91	0.0246	72.79	48.44	9.4	9.34
Mockingjay [8]	70.19	83.67	34.33	32.29	50.28	22.82	15.48	6.6E-04	61.59	58.89	11.66	10.54
TERA [9]	49.17	89.48	58.42	57.57	56.27	18.17	12.16	0.0013	67.50	54.17	15.89	9.96
modified CPC [34]	42.54	91.88	64.09	39.63	60.96	20.18	13.53	0.0326	71.19	49.91	12.86	10.38
wav2vec [12]	31.58	95.59	84.92	56.56	59.79	15.86	11.00	0.0485	76.37	43.71	7.99	9.9
vq-wav2vec [13]	33.48	93.38	85.68	38.80	58.24	17.71	12.80	0.0410	77.68	41.54	10.38	9.93
wav2vec 2.0 Base [14]	5.74	96.23	92.35	75.18	63.43	6.43	4.79	0.0233	88.30	24.77	6.02	6.08
wav2vec 2.0 Large [14]	4.75	**96.66**	95.28	86.14	65.64	3.75	3.10	0.0489	87.11	27.31	5.65	**5.62**
HuBERT Base [35]	5.41	96.30	98.34	81.42	64.92	6.42	4.79	**0.0736**	88.53	25.20	**5.11**	5.88
HuBERT Large [35]	**3.53**	95.29	**98.76**	**90.33**	**67.62**	**3.62**	**2.94**	0.0353	**89.81**	**21.76**	5.98	5.75

- Some self-supervised models outperform fbank in all tasks.
- HuBERT and wav2vec 2.0 outperform other models

	PR	KS	IC	SID	ER	ASR (WER)		QbE	SF		ASV	SD
	PER ↓	Acc ↑	Acc ↑	Acc ↑	Acc ↑	w/o ↓	w/ LM ↓	MTWV ↑	F1 ↑	CER ↓	EER ↓	DER ↓
FBANK	82.01	8.63	9.10	8.5E-4	35.39	23.18	15.21	0.0058	69.64	52.94	9.56	10.05
PASE+ [16]	58.87	82.54	29.82	37.99	57.86	25.11	16.62	0.0072	62.14	60.17	11.61	8.68
APC [7]	41.98	91.01	74.69	60.42	59.33	21.28	14.74	0.0310	70.46	50.89	8.56	10.53
VQ-APC [32]	41.08	91.11	74.48	60.15	59.66	21.20	15.21	0.0251	68.53	52.91	8.72	10.45
NPC [33]	43.81	88.96	69.44	55.92	59.08	20.20	13.91	0.0246	72.79	48.44	9.4	9.34
Mockingjay [8]	70.19	83.67	34.33	32.29	50.28	22.82	15.48	6.6E-04	61.59	58.89	11.66	10.54
TERA [9]	49.17	89.48	58.42	57.57	56.27	18.17	12.16	0.0013	67.50	54.17	15.89	9.96
modified CPC [34]	42.54	91.88	64.09	39.63	60.96	20.18	13.53	0.0326	71.19	49.91	12.86	10.38
wav2vec [12]	31.58	95.59	84.92	56.56	59.79	15.86	11.00	0.0485	76.37	43.71	7.99	9.9
vq-wav2vec [13]	33.48	93.38	85.68	38.80	58.24	17.71	12.80	0.0410	77.68	41.54	10.38	9.93
wav2vec 2.0 Base [14]	5.74	96.23	92.35	75.18	63.43	6.43	4.79	0.0233	88.30	24.77	6.02	6.08
wav2vec 2.0 Large [14]	4.75	**96.66**	95.28	86.14	65.64	3.75	3.10	0.0489	87.11	27.31	5.65	**5.62**
HuBERT Base [35]	5.41	96.30	98.34	81.42	64.92	6.42	4.79	**0.0736**	88.53	25.20	**5.11**	5.88
HuBERT Large [35]	**3.53**	95.29	**98.76**	**90.33**	**67.62**	**3.62**	**2.94**	0.0353	**89.81**	**21.76**	5.98	5.75

Leaderboard: https://superbbenchmark.org/

Welcome to submit! ☺

https://github.com/s3prl/s3prl

S3PRL toolkit: **S**elf-**S**upervised **S**peech **P**re-training and **R**epresentation **L**earning

- Upstream Pre-training
- Upstream Hub
- Downstream fine-tuning
- SUPERB Challenge

Speech Brain

ESPNet

https://github.com/s3prl/s3prl

https://github.com/s3prl/s3prl

S3PRL toolkit: **S**elf-**S**upervised **S**peech **P**re-training and **R**epresentation **L**earning

Shu-wen Yang (Leo)

Andy T. Liu

Link to recording: https://youtu.be/PkMFnS6cjAc

Next Step

- Adding new tasks
  - Speech Translation
  - Spoken Question Answering
  - Speech Enhancement
  - Speaker Separation
  - Voice Conversion
- Other ways to use self-supervised models

Next Step

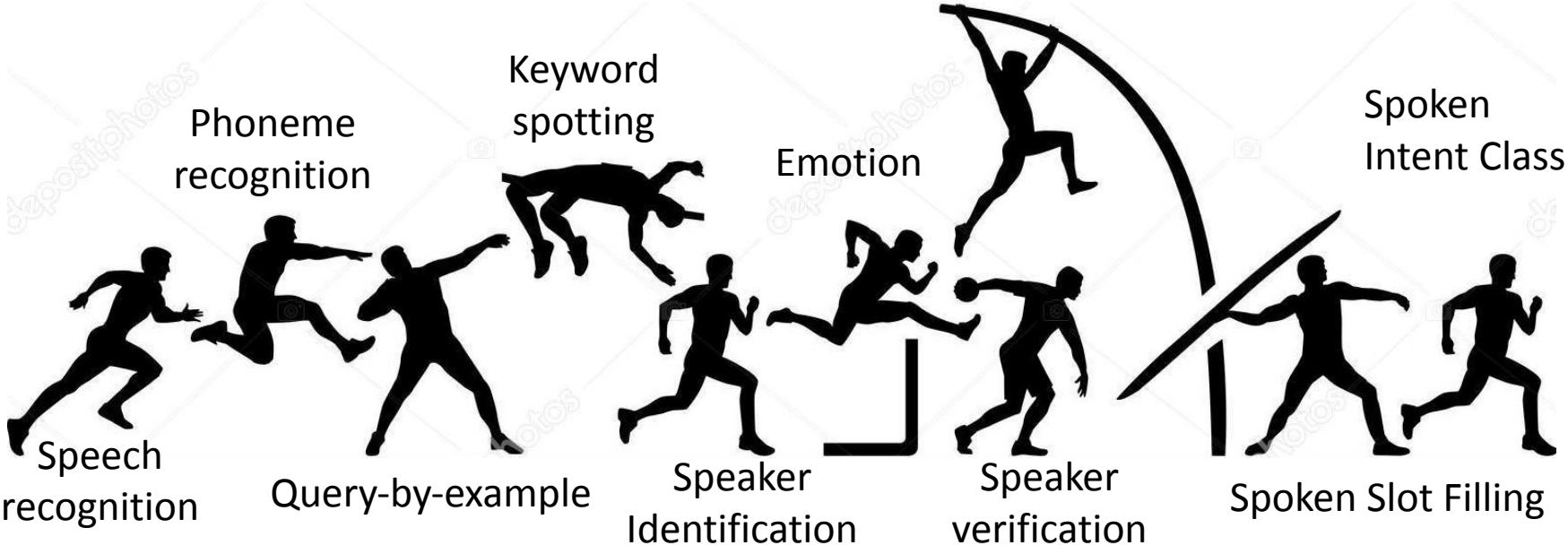
- Near future: Challenge
  - Private data for the tasks
  - (Perhaps) You only submit self-supervised model. We train your model on downstream tasks by our scripts for fair comparison.

Stayed tuned: https://superbbenchmark.org/

Far Future Work

life-long learning: learning the tasks sequentially

meta learning: learning a learning algorithm on some training tasks



Call for Paper

- IEEE JSTSP Special Issue on **Self-Supervised Learning for Speech and Audio Processing**
- Deadline: December 30, 2021
- Link https://signalprocessingsociety.org/blog/ieee-jstsp-special-issue-self-supervised-learning-speech-and-audio-processing

Benchmarking for Self-supervised Learning

LeBenchmark

A bit of context – LeBenchmark

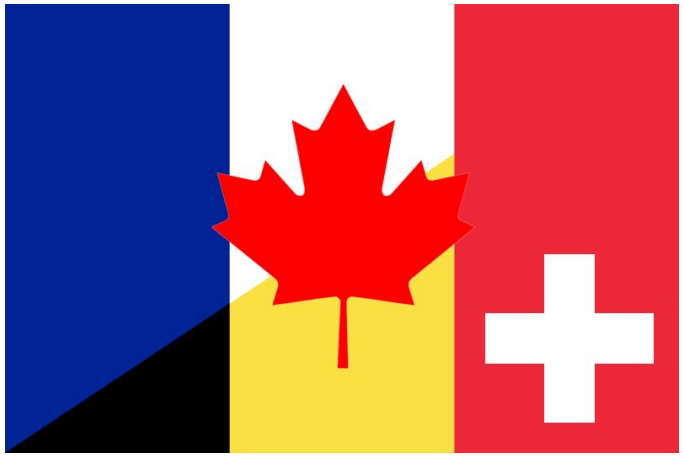
LeBenchmark: A Reproducible Framework for Assessing Self-Supervised Representation Learning from Speech

Solène Evain[1,], Ha Nguyen[1,2,*], Hang Le[1,*], Marcely Zanon Boito[1,*], Salima Mdhaffar[2,*], Sina Alisamir[1,3], Ziyi Tong[1], Natalia Tomashenko[2], Marco Dinarelli[1,*], Titouan Parcollet[2,*], Alexandre Allauzen[4], Yannick Estève[2], Benjamin Lecouteux[1], François Portet[1], Solange Rossato[1], Fabien Ringeval[1], Didier Schwab[1] and Laurent Besacier[1,5]*

Published at INTERSPEECH 2021

A bit of context – LeBenchmark

... with one thing in common:



SSL for speech looks great but ...

Lack of scientific standardisation

SSL for speech looks great but ...

Lack of scientific standardisation **+ mostly evaluated on English.**

SSL for speech looks great but ...

Lack of scientific standardisation + mostly evaluated on English.

=

How do we even know if a new method works better in a multilingual setup?

SSL for speech looks great but ...

Lack of scientific standardisation + mostly evaluated on English.

=

How do we even know if a new method works better in a multilingual setup?

LeBenchmark = std(dataset && evaluation) + French.

First step – standardised dataset

Remember: we want to evaluate new SSL models for Speech.

First step – standardised dataset

Remember: we want to evaluate new SSL models for Speech.

We should not waste time in investigating the impact of different data.

First step – standardised dataset

Remember: we want to evaluate new SSL models for Speech.

Offer a large-enough and heterogeneous-enough dataset.

First step – standardised dataset

Remember: we want to evaluate new SSL models for Speech.

Offer a **large-enough** and heterogeneous-enough dataset.

SSL needs **a lot** of data (e.g. VoxPopuli > 100K Hours)

First step – standardised dataset

Remember: we want to evaluate new SSL models for Speech.

Offer a large-enough and **heterogeneous-enough** dataset.

Gender balance, influence of the environment, type of speech (spontaneous, broadcasted, read ...) ...

First step – standardised dataset

Problem: getting the data.
(everything that is not english is hard to obtain)

First step – standardised dataset

Problem: getting the data.
(everything that is not english is hard to obtain)

Gather well-known French datasets and **document them**.

First step – standardised dataset

Problem: getting the data.
(everything that is not english is hard to obtain)

Gather well-known French datasets and **document them**.

LeBenchmark datasets = **1K** hours or **3K** hours or **7K** hours or **10K*** hours

***not available yet (Submitted to NeurIPS Datasets and Benchmarks track)**

First step – standardised dataset

Corpus$_{License}$	# Utterances	Duration	# Speakers	Mean Utt. Duration	Speech type
		Small dataset			
MLS French$_{CCBY4.0}$ (75)	**263,055** 124,590 / 138,465 / –	**1,096:43** 520:13 / 576:29 / –	**178** 80 / 98 / –	**15 s** 15 s / 15 s / –	Read
		Medium dataset			
African Accented French$_{Apache2.0}$ (3)	**16,402** 373 / 102 / 15,927	**18:56** – / – / 18:56	**232** 48 / 36 / 148	**4 s** – / – / –	Read
Att-Hack$_{CCBYNCND}$ (44)	**36,339** 16,564 / 19,775 / –	**27:02** 12:07 / 14:54 / –	**20** 9 / 11 / –	**2.7 s** 2.6 s / 2.7 s / –	Acted Emotional
CaFE$_{CCNC}$ (35)	**936** 468 / 468 / –	**1:09** 0:32 / 0:36 / –	**12** 6 / 6 / –	**4.4 s** 4.2 s / 4.7 s / –	Acted Emotional
CFPP2000$_{CCBYNCSA}$* (15), (cfp)	**9853** 166 / 1,184 / 8,503	**16:26** 0:14 / 1:56 / 14:16	**49** 2 / 4 / 43	**6 s** 5 s / 5 s / 6 s	Spontaneous
ESLO2$_{NC}$ (29), (esl)	**62,918** 30,440 / 32,147 / 331	**34:12** 17:06 / 16:57 / 0:09	**190** 68 / 120 / 2	**1.9 s** 2 s / 1.9 s / 1.7 s	Spontaneous
EPAC**$_{NC}$ (30)	**623,250** 465,859 / 157,391 / –	**1,626:02** 1,240:10 / 385:52 / –	**1,935** – / – / –	**9 s** – / – / –	Radio Broadcasts
GEMEP$_{NC}$ (16)	**1,236** 616 / 620 / –	**0:50** 0:24 / 0:26 / –	**10** 5 / 5 / –	**2.5 s** 2.4 s / 2.5 s / –	Acted Emotional
MPF (32), (4)	**19,527** 5,326 / 4,649 / 9,552	**19:06** 5:26 / 4:36 / 9:03	**114** 36 / 29 / 49	**3.5 s** 3.7 s / 3.6 s / 3.4 s	Spontaneous
PORTMEDIA$_{NC}$ (French) (45)	**19,627** 9,294 / 10,333 / –	**38:59** 19:08 / 19:50 / –	**193** 84 / 109 / –	**7.1 s** 7.4 s / 6.9 s / –	Acted telephone dialogue
TCOF (Adults) (7)	**58,722** 10,377 / 14,763 / 33,582	**53:59** 9:33 / 12:39 / 31:46	**749** 119 / 162 / 468	**3.3 s** 3.3 s / 3.1 s / 3.4 s	Spontaneous
Medium dataset total	**1,111,865** 664,073 / 379,897 / 67,895	**2,933:24** 1,824:53 / 1,034:15 / 74:10	-	-	-
		Large dataset			
MaSS (13)	**8,219** 8,219 / – / –	**19:40** 19:40 / – / –	**Unk** – / – / –	**8.6 s** 8.6 s / – / –	Read
NCCFr$_{NC}$ (72)	**29,421** 14,570 / 13,922 / 929	**26:35** 12:44 / 12:59 / 00:50	**46** 24 / 21 / 1	**3 s** 3 s / 3 s / 3 s	Spontaneous
Voxpopuli$_{CCBYNC4.0}$ (78) *Unlabeled****	**568,338** – / – / –	**4,532:17** – / – / 4,532:17	**Unk** – / – / –	**29 s** – / – / –	Professional speech
Voxpopuli$_{CCBYNC4.0}$ (78) *transcribed****	**76.281** – / – / –	**211:57** – / – / 211:57	**327** – / – / –	**10 s** – / – / –	Professional speech
Large dataset total****	**1,814,242** 682,322 / 388,217 / 99,084	**7,739:22** 1,853:02 / 1,041:07 / 4,845:07	-	-	-

Influence of the data availability (e.g. low-resources languages).

Are 1000H enough to train a SSL system?

Is it worth adding thousands of hours of speech and compute?

Does it depend on the downstream task?

First step – standardised dataset

Corpus$_{License}$	# Utterances	Duration	# Speakers	Mean Utt. Duration	Speech type
		Small dataset			
MLS French$_{CCBY4.0}$ (75)	**263,055** 124,590 / 138,465 / –	**1,096:43** 520:13 / 576:29 / –	**178** 80 / 98 / –	**15 s** 15 s / 15 s / –	Read
		Medium dataset			
African Accented French$_{Apache2.0}$ (3)	**16,402** 373 / 102 / 15,927	**18:56** – / – / 18:56	**232** 48 / 36 / 148	**4 s** – / – / –	Read
Att-Hack$_{CCBYNCND}$ (44)	**36,339** 16,564 / 19,775 / –	**27:02** 12:07 / 14:54 / –	**20** 9 / 11 / –	**2.7 s** 2.6 s / 2.7 s / –	Acted Emotional
CaFE$_{CCNC}$ (35)	**936** 468 / 468 / –	**1:09** 0:32 / 0:36 / –	**12** 6 / 6 / –	**4.4 s** 4.2 s / 4.7 s / –	Acted Emotional
CFPP2000$_{CCBYNCSA}$* (15), (cfp)	**9853** 166 / 1,184 / 8,503	**16:26** 0:14 / 1:56 / 14:16	**49** 2 / 4 / 43	**6 s** 5 s / 5 s / 6 s	Spontaneous
ESLO2$_{NC}$ (29), (esl)	**62,918** 30,440 / 32,147 / 331	**34:12** 17:06 / 16:57 / 0:09	**190** 68 / 120 / 2	**1.9 s** 2 s / 1.9 s / 1.7 s	Spontaneous
EPAC**$_{NC}$ (30)	**623,250** 465,859 / 157,391 / –	**1,626:02** 1,240:10 / 385:52 / –	**1,935** – / – / –	**9 s** – / – / –	Radio Broadcasts
GEMEP$_{NC}$ (16)	**1,236** 616 / 620 / –	**0:50** 0:24 / 0:26 / –	**10** 5 / 5 / –	**2.5 s** 2.4 s / 2.5 s / –	Acted Emotional
MPF (32), (4)	**19,527** 5,326 / 4,649 / 9,552	**19:06** 5:26 / 4:36 / 9:03	**114** 36 / 29 / 49	**3.5 s** 3.7 s / 3.6 s / 3.4 s	Spontaneous
PORTMEDIA$_{NC}$ (French) (45)	**19,627** 9,294 / 10,333 / –	**38:59** 19:08 / 19:50 / –	**193** 84 / 109 / –	**7.1 s** 7.4 s / 6.9 s / –	Acted telephone dialogue
TCOF (Adults) (7)	**58,722** 10,377 / 14,763 / 33,582	**53:59** 9:33 / 12:39 / 31:46	**749** 119 / 162 / 468	**3.3 s** 3.3 s / 3.1 s / 3.4 s	Spontaneous
Medium dataset total	**1,111,865** 664,073 / 379,897 / 67,895	**2,933:24** 1,824:53 / 1,034:15 / 74:10	-	-	-
		Large dataset			
MaSS (13)	**8,219** 8,219 / – / –	**19:40** 19:40 / – / –	**Unk** – / – / –	**8.6 s** 8.6 s / – / –	Read
NCCFr$_{NC}$ (72)	**29,421** 14,570 / 13,922 / 929	**26:35** 12:44 / 12:59 / 00:50	**46** 24 / 21 / 1	**3 s** 3 s / 3 s / 3 s	Spontaneous
Voxpopuli$_{CCBYNC4.0}$ (78) *Unlabeled****	**568,338** – / – / –	**4,532:17** – / – / 4,532:17	**Unk** – / – / –	**29 s** – / – / –	Professional speech
Voxpopuli$_{CCBYNC4.0}$ (78) *transcribed****	**76.281** – / – / –	**211:57** – / – / 211:57	**327** – / – / –	**10 s** – / – / –	Professional speech
Large dataset total**	**1,814,242** 682,322 / 388,217 / 99,084	**7,739:22** 1,853:02 / 1,041:07 / 4,845:07	-	-	-

Influence of the data type (e.g. on-the-wild).

Does spontaneous speech help with SSL?

Are noisy data helping with robustness?

Can we just train a big model with read speech only?

First step – standardised dataset

Corpus$_{License}$	# Utterances	Duration	# Speakers	Mean Utt. Duration	Speech type
		Small dataset			
MLS French$_{CCBY4.0}$ (75)	**263,055** 124,590 / 138,465 / –	**1,096:43** 520:13 / 576:29 / –	**178** 80 / 98 / –	**15 s** 15 s / 15 s / –	Read
		Medium dataset			
African Accented French$_{Apache2.0}$ (3)	**16,402** 373 / 102 / 15,927	**18:56** – / – / 18:56	**232** 48 / 36 / 148	**4 s** – / – / –	Read
Att-Hack$_{CCBYNCND}$ (44)	**36,339** 16,564 / 19,775 / –	**27:02** 12:07 / 14:54 / –	**20** 9 / 11 / –	**2.7 s** 2.6 s / 2.7 s / –	Acted Emotional
CaFE$_{CCNC}$ (35)	**936** 468 / 468 / –	**1:09** 0:32 / 0:36 / –	**12** 6 / 6 / –	**4.4 s** 4.2 s / 4.7 s / –	Acted Emotional
CFPP2000$_{CCBYNCSA}$* (15), (cfp)	**9853** 166 / 1,184 / 8,503	**16:26** 0:14 / 1:56 / 14:16	**49** 2 / 4 / 43	**6 s** 5 s / 5 s / 6 s	Spontaneous
ESLO2$_{NC}$ (29), (esl)	**62,918** 30,440 / 32,147 / 331	**34:12** 17:06 / 16:57 / 0:09	**190** 68 / 120 / 2	**1.9 s** 2 s / 1.9 s / 1.7 s	Spontaneous
EPAC**$_{NC}$ (30)	**623,250** 465,859 / 157,391 / –	**1,626:02** 1,240:10 / 385:52 / –	**1,935** – / – / –	**9 s** – / – / –	Radio Broadcasts
GEMEP$_{NC}$ (16)	**1,236** 616 / 620 / –	**0:50** 0:24 / 0:26 / –	**10** 5 / 5 / –	**2.5 s** 2.4 s / 2.5 s / –	Acted Emotional
MPF (32), (4)	**19,527** 5,326 / 4,649 / 9,552	**19:06** 5:26 / 4:36 / 9:03	**114** 36 / 29 / 49	**3.5 s** 3.7 s / 3.6 s / 3.4 s	Spontaneous
PORTMEDIA$_{NC}$ (French) (45)	**19,627** 9,294 / 10,333 / –	**38:59** 19:08 / 19:50 / –	**193** 84 / 109 / –	**7.1 s** 7.4 s / 6.9 s / –	Acted telephone dialogue
TCOF (Adults) (7)	**58,722** 10,377 / 14,763 / 33,582	**53:59** 9:33 / 12:39 / 31:46	**749** 119 / 162 / 468	**3.3 s** 3.3 s / 3.1 s / 3.4 s	Spontaneous
Medium dataset total	**1,111,865** 664,073 / 379,897 / 67,895	**2,933:24** 1,824:53 / 1,034:15 / 74:10	-	-	-
		Large dataset			
MaSS (13)	**8,219** 8,219 / – / –	**19:40** 19:40 / – / –	**Unk** – / – / –	**8.6 s** 8.6 s / – / –	Read
NCCFr$_{NC}$ (72)	**29,421** 14,570 / 13,922 / 929	**26:35** 12:44 / 12:59 / 00:50	**46** 24 / 21 / 1	**3 s** 3 s / 3 s / 3 s	Spontaneous
Voxpopuli$_{CCBYNC4.0}$ (78) *Unlabeled****	**568,338** – / – / –	**4,532:17** – / – / 4,532:17	**Unk** – / – / –	**29 s** – / – / –	Professional speech
Voxpopuli$_{CCBYNC4.0}$ (78) *transcribed****	**76.281** – / – / –	**211:57** – / – / 211:57	**327** – / – / –	**10 s** – / – / –	Professional speech
Large dataset total**	**1,814,242** 682,322 / 388,217 / 99,084	**7,739:22** 1,853:02 / 1,041:07 / 4,845:07	-	-	-

Research remains slowed down by the distribution scheme of the data …

Second step – baseline SSL models

Give the community a set of baselines to compare with.

Second step – baseline SSL models

Give the community a set of baselines to compare with.

Research about SSL is not only about finding the best SSL model.

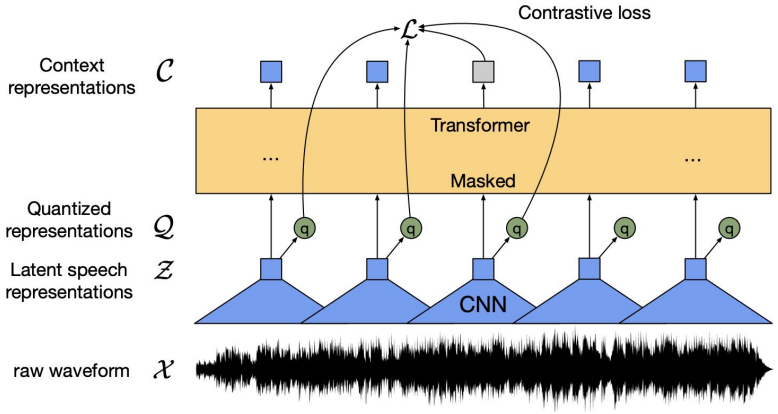
Second step – baseline SSL models

Give the community a set of baselines to compare with.

Research about SSL is not only about finding the best SSL model.

Not everyone can train a SSL model on 7K hours **(64 Tesla V100/14 days).**

Second step – baseline SSL models



wav2vec 2.0*

Model Type: Transformer
SSL Type: Contrastive Learning

Training these models **really** is challenging …

*Baevski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *Advances in Neural Information Processing Systems*, *33*.

Second step – baseline SSL models

Release the models to the community!

wav2vec-FR-7K-Large
wav2vec-FR-3K-Large
wav2vec-FR-1K-Large

wav2vec-FR-7K-Base
wav2vec-FR-3K-Base
wav2vec-FR-2.6K-Base
wav2vec-FR-1K-Base

https://huggingface.co/LeBenchmark

Can be **easily** used with well-know toolkits: SpeechBrain, ESPnet, S3PRL, FairSeq ...

Second step – baseline SSL models

Release the models to the community!

wav2vec-FR-7K-Large
wav2vec-FR-3K-Large
wav2vec-FR-1K-Large

wav2vec-FR-7K-Base
wav2vec-FR-3K-Base
wav2vec-FR-2.6K-Base
wav2vec-FR-1K-Base

https://huggingface.co/LeBenchmark

wav2vec-FR-2.6K-Base **does not contain spontaneous speech.**

Third step – Introduce the benchmark

Problem: designing a benchmark close to the real world is **extremely hard**.
(almost impossible)

Third step – Introduce the benchmark

Problem: designing a benchmark close to the real world is **extremely hard**.
(almost impossible)

diversity.

Third step – Introduce the benchmark

Diversity of problems: classification (AER), sequence labelling (SLU) and conditional natural language generation (ASR, AST).

Third step – Introduce the benchmark

Diversity of problems: classification (AER), sequence labelling (SLU) and conditional natural language generation (ASR, AST).

Diversity of information extracted: transcript (ASR), semantics (SLU), translation (AST) and paralinguistics (AER).

Third step – Introduce the benchmark

Diversity of problems: classification (AER), sequence labelling (SLU) and conditional natural language generation (ASR, AST).

Diversity of information extracted: transcript (ASR), semantics (SLU), translation (AST) and paralinguistics (AER).

Diversity of annotated resources available for downstream tasks: large (ASR), medium (SLU, AST) or small (AER).

Third step – Introduce the benchmark

Diversity of problems: classification (AER), sequence labelling (SLU) and conditional natural language generation (ASR, AST).

Diversity of information extracted: transcript (ASR), semantics (SLU), translation (AST) and paralinguistics (AER).

Diversity of annotated resources available for downstream tasks: large (ASR), medium (SLU, AST) or small (AER).

Provide **French standardised downstream baselines with training scripts** for:

Automatic **E**motion **R**ecognition,
Spoken **L**anguage **U**nderstanding,
Automatic **S**peech **T**ranslation,
Automatic **S**peech **R**ecognition.

https://github.com/LeBenchmark/Interspeech2021

Third step – Introduce the benchmark

Too much tables and numbers **=** let’s jump to the outcomes.

Third step – Introduce the benchmark

Too much **tables** and **numbers =** let’s jump to the outcomes.

Good for the community!

Third step – Introduce the benchmark

Automatic **E**motion **R**ecognition
(RECOLA — **3.8h** / AlloSat **37h**)

Expected or Unexpected

Third step – Introduce the benchmark

Automatic **E**motion **R**ecognition
(RECOLA — **3.8h** / AlloSat **37h**)

Multilingual wav2vec 2.0 (XLSR) trained on **way more data** is not as good as our French models.

Smaller wav2vec 2.0 (base) lead to **better performance**.

SSL features are better than traditional acoustic ones (FBANKs).

Expected or Unexpected

Third step – Introduce the benchmark

Spoken **L**anguage **U**nderstanding
(MEDIA — **17h**)

Multilingual wav2vec 2.0 (XLSR) trained on **way more data** is not as good as our French models.

More hours of pretraining do not improve the performance.

Larger wav2vec 2.0 (large) lead to **better performance**.

SSL features are better than traditional acoustic ones (FBANKs).

Expected or Unexpected

Third step – Introduce the benchmark

Automatic **S**peech **T**ranslation
(French to {English, Portugese, Spanish} — TEDx and CoVoST2 — [**25h, 180h**])

Multilingual wav2vec 2.0 (XLSR) trained on **way more data** is not as good as our French models.

Gains are reducing with **the increase of supervised data**.

Larger wav2vec 2.0 (large) lead to **better performance**.

SSL features are better than traditional acoustic ones (FBANKs).

Expected or Unexpected

Third step – Introduce the benchmark

Automatic **S**peech **R**ecognition
(CommonVoice — **477h** / ETAPE — **36h**)

Multilingual wav2vec 2.0 (XLSR) trained on **way more data** is not as good as our French models.

Gains are reducing with **the increase of supervised data**.

Larger wav2vec 2.0 (large) lead to **better performance**.

SSL features are better than traditional acoustic ones (FBANKs).

Expected or Unexpected

LeBenchmark

Outcomes:

SSL models can not be deployed and used from intuitions only.

We still do not know exactly what will work or not.

SSL models behaviors vary a lot.

Benchmarks are needed to frame the development of these technologies!

open-source toolkits

Open-Source Toolkits for Speech Processing

- **Open-source** toolkits have played a critical role in the development of speech processing technology:

CMU Sphinx

- With the raise of **deep learning**, some general-purpose libraries have been developed:

Open-Source Toolkits for Speech Processing

- Thanks to these general framework, more flexible **python-based** speech processing toolkits have quickly appeared.

- Some of them are **task-specific**:

Speech Separation

Diarization

SIDEKIT

Speaker Recognition

DeepSpeech

Speech Recognition

RETURNN

Speech Recognition

Speech Recognition

ESPRESSO

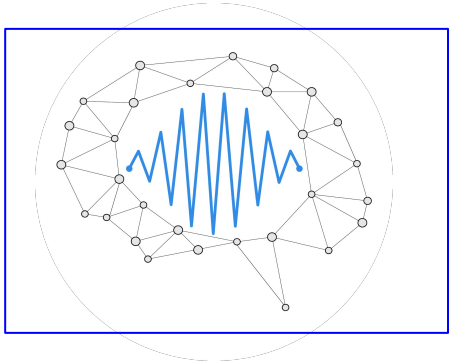
Speech Recognition

LINGVO

Speech Recognition

Open-Source Toolkits for Speech Processing

- Some others support multiple speech tasks:



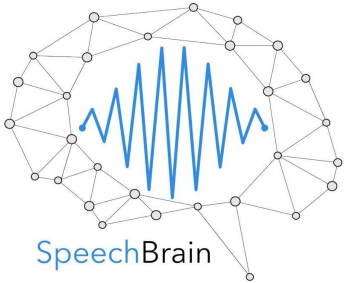
open-source toolkits

Speech Brain

What is SpeechBrain?

- **SpeechBrain** is an *open-source* and *all-in-one* speech toolkit based on PyTorch.

- **Goal:** speed up **research** and **development** of speech and audio processing techniques.



Key features:

- *Flexibility*
- *Easy-to-use*
- *Modularity*
- *Efficiency*
- *Good documentation*

python™

PyTorch

Website: speechbrain.github.io/

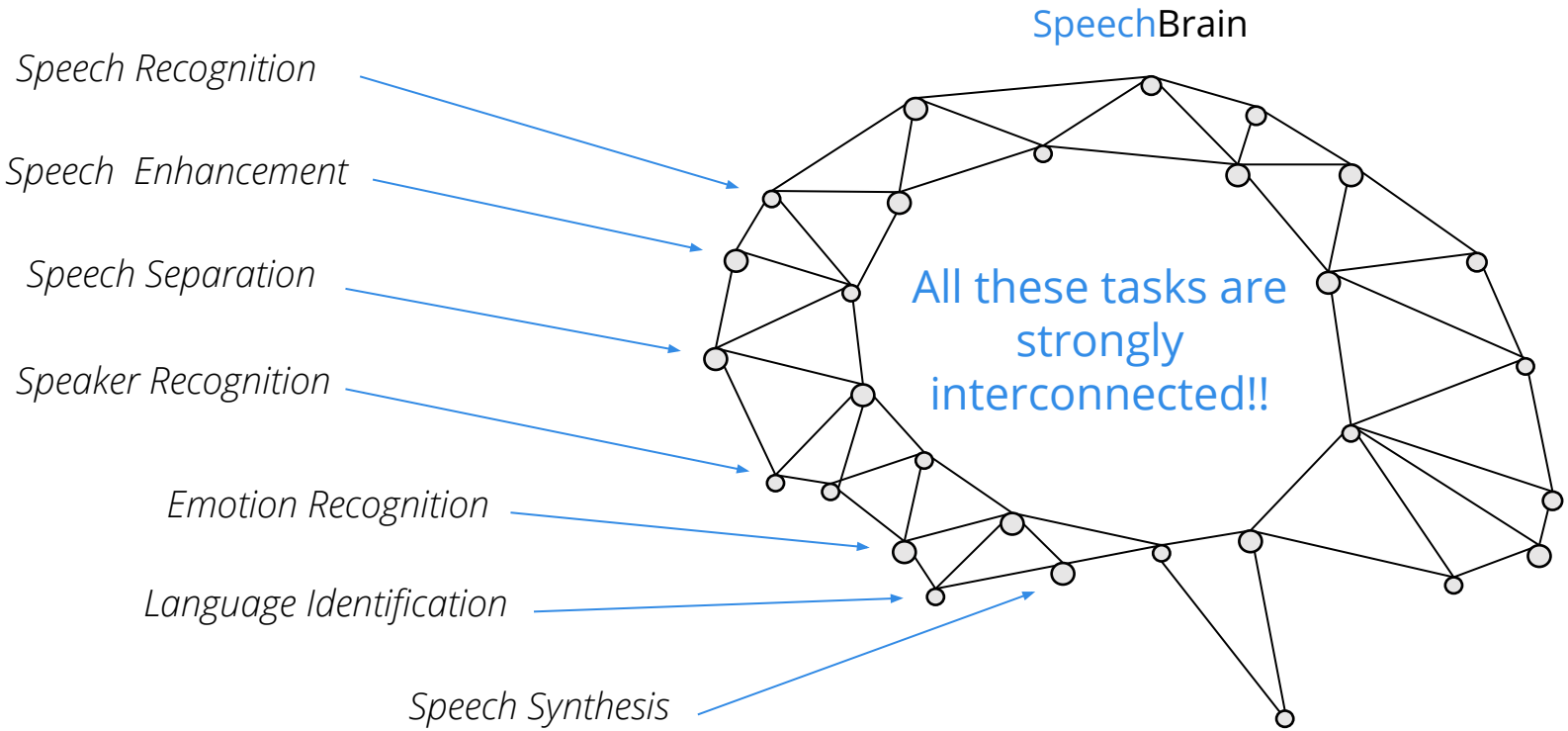
Code: github.com/speechbrain/speechbrain

Tutorials: github.com/speechbrain/speechbrain

Pretrained models: huggingface.co/speechbrain

What is SpeechBrain?

SpeechBrain is designed from scratch to support **multiple speech processing tasks.**



What can I do with SpeechBrain?

Speech Recognition

Neural Network

Seq2seq RNN
RNN Transducers
Transformers
...

Text

Speech Enhancement

Neural Network

MetricGAN+
MimicLoss
...

Speech Separation

Neural Network

ConvTasnet
DualPath RNN
SepFormers
....

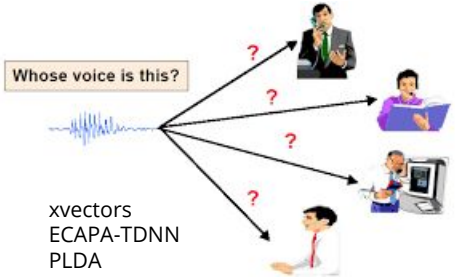
Spoken Language Understanding

Neural Network

-E2E Direct
- Multi-Stage
- Decoupled

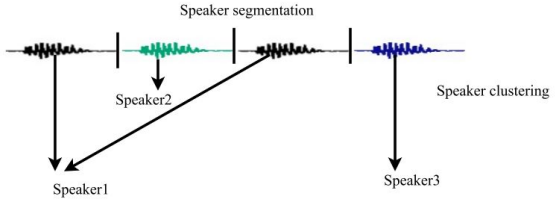
Semantics

Speaker Recognition



xvectors
ECAPA-TDNN
PLDA
....

Speaker Diarization



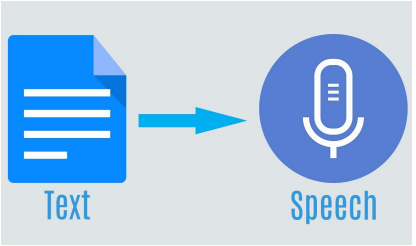
And many others....

SpectralClustering
.....

What can I do with SpeechBrain?

SpeechBrain has also recipes for:

- Language Modeling
- Language Identification
- Sound Classification
- Grapheme-to-phoneme
- Multi-microphone signal processing
- EEG Decoding of Brain Signals



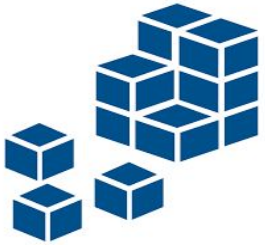
….. and ongoing work for:

- Text-to-Speech
- Music Generation
- Speech Translation
- Voice Activity Detection
- Finite State Transducers (Integration with k2)

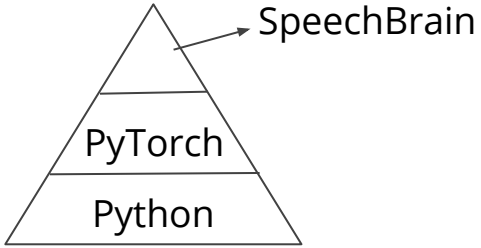
Design Principles

Ease of use

Simplicity & Modularity: we develop intuitive **modules that are easy to interconnect with each other**. The code is pythonic and maximizes the use PyTorch routines.



Lean software stack: SpeechBrain employs a **simple software stack** (i.e., *Python → PyTorch → SpeechBrain*) to avoid dealing with too many levels of abstractions. PyTorch-compatible code works in our toolkit without any further modification



Minimal external dependencies: SpeechBrain has a **minimal list of external dependencies** that are all installable via PyPI. The installation process simply requires running the command pip install speechbrain and is done within a few minutes.

Design Principles

Replicability & Transparency

- SpeechBrain promotes **open** and **transparent science**.
- We trained most of our models with **publicly available data**. This way, our results can be easily **replicated** by the community.
- Several **pre-trained models**, which only require a few lines of code to use, are distributed via **Hugging Face**.
- Besides sharing the code and the trained models, we also share the whole experiment folder, which **contains all the needed details** (e.g., logs) to reproduce our results.

Design Principles

Accessibility

- We have released SpechBrain under a very permissive license (Apache 2.0).
- SpeechBrain is designed to be easily understandable by a **large user base**, including early **students** and **practitioners.**
- We want SpeechBrain to serve **educational purposes** as well.
- We put several efforts on writing comprehensive **documentation**.

How to Train a Model

- For all recipes and tasks, users can train a model simply with:

```
python train.py hparams.yaml
```

```
Brain.fit()
self.make_dataloader()
self.on_fit_start()
for epoch in epoch_counter:
    self.on_stage_start(Stage.TRAIN)
    for batch in train_set:
        self.fit_batch(batch)
            p = self.compute_forward(batch)
            loss = self.compute_objectives(p, batch)
            # perform parameter update
    self.on_stage_end(Stage.TRAIN)
    self.on_stage_start(Stage.VALID)
    for batch in valid_set:
        self.evaluate_batch(batch)
            p = self.compute_forward(batch)
            loss = self.compute_objectives(p, batch)
    self.on_stage_end(Stage.VALID)

self._compile_jit()
self._wrap_distributed()
self.init_optimizers()
```

Important computations (e.g., forward step, objectives, data-io transformation) **are directly visible** in train.py

HyperPyYAML

```
dropout: 0.2
features: !new:speechbrain.lobes.features.MFCC
    n_mels: 40
    left_frames: 5
    right_frames: 5

model: !new:torch.nn.LSTM
    input_size: 440
    hidden_size: 256
    num_layers: 4
    dropout:  !ref <dropout>
    bidirectional: True
```

Not a plain list of hyperparameters, we declare the objects that we use.

There is a **visible connection between the hyperparameter and the object** using it.

Inference

- SpeechBrain also provides functions for performing easy inference on pre-trained models:

Speech Recognition

```
from speechbrain.pretrained import EncoderDecoderASR
asr model =
EncoderDecoderASR.from hparams(source= "speechbrain/asr-crdnn-rnnlm-librispeech" )
asr_model.transcribe_file( 'your_file.wav' )
```

huggingface.co/speechbrain

Speaker Verification

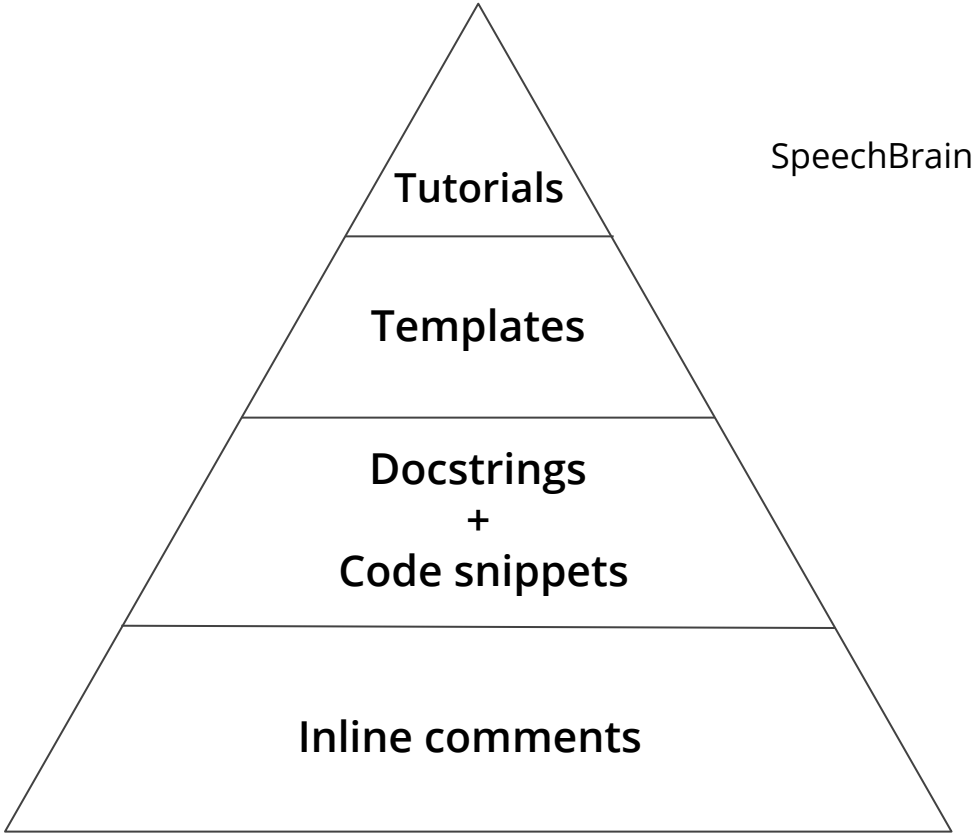
```
from speechbrain.pretrained import SpeakerRecognition
verification = SpeakerRecognition.from hparams(source= "speechbrain/spkrec-ecapa-voxceleb" )
score, prediction = verification.verify_files( "file_1.wav","file2_2.wav")
```

Speech Separation

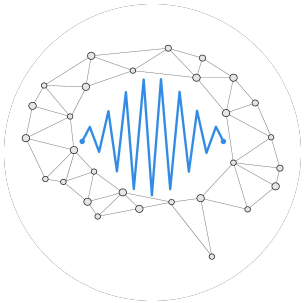
```
from speechbrain.pretrained import SepformerSeparation as separator
model = separator.from hparams(source="speechbrain/sepformer-wsj02mix")
est_sources = model.separate_file(path='mixture.wav'
```

*SpeechBrain currently has **25 pretrained models** on HuggingFace*

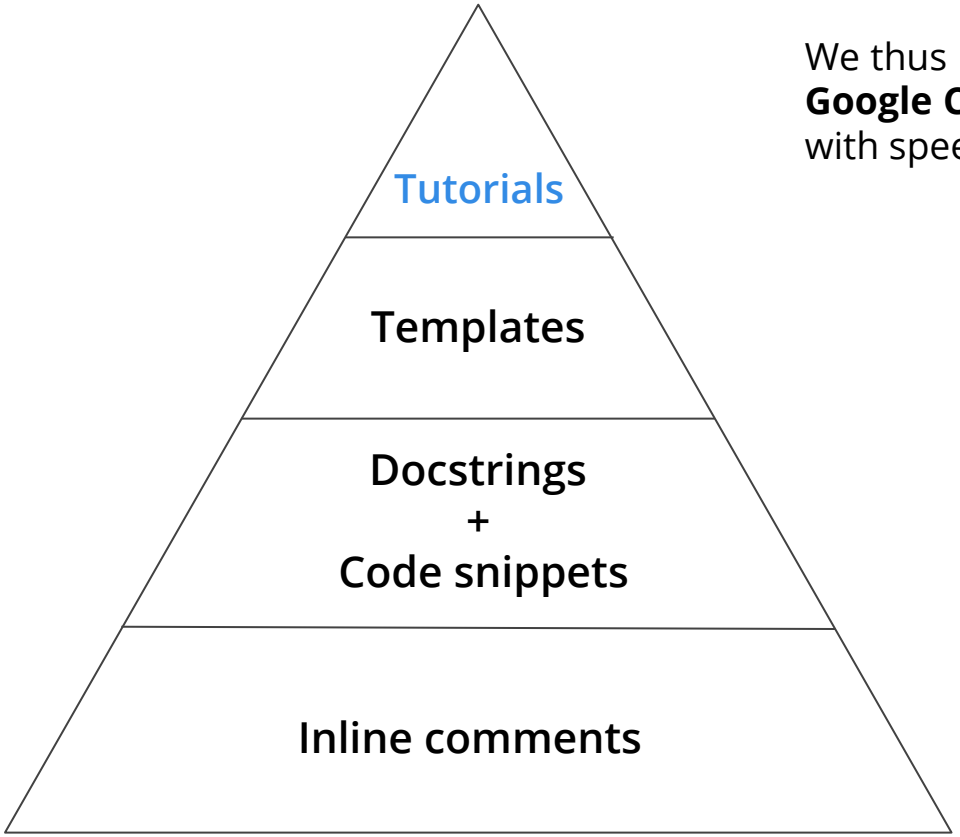
Documentation



SpeechBrain provides documentation at **different levels**.



Documentation



Tutorials

We thus have written several tutorials (currently **22**) with **Google Colab** to help newcomers become more familiar with speech technologies.

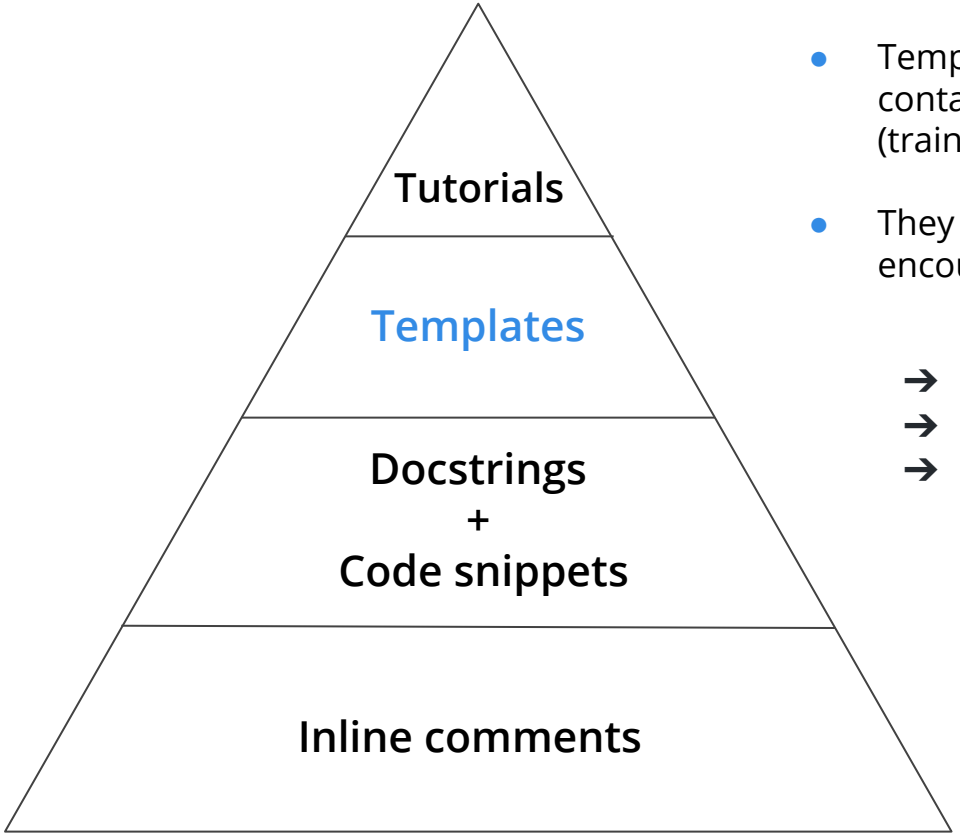
colab

https://speechbrain.github.io/

Examples of tutorials:

- Speech Recognition from Scratch
- Speech Classification from Scratch
- Speech Enhancement from Scratch

Documentation

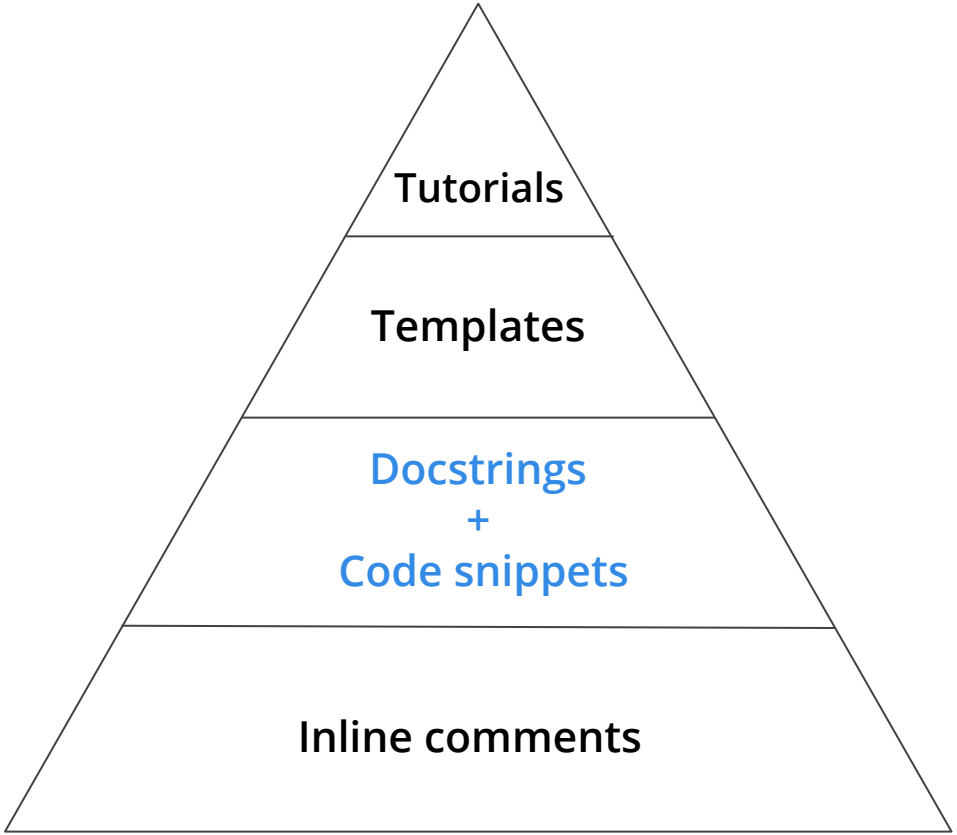


Templates

- Templates are **simple**, **well-documented recipes** that contain all the parts necessary for a **working system** (training, validation, evaluation, inference, ..).
- They cover a broad spectrum of types of tasks that are encountered in speech research, such as:

➔ *sequence regression* (enhancement)
➔ *sequence to sequence* (speech recognition)
➔ *sequence classification* (speaker ID)

Documentation




```python
class SincConv(nn.Module):
    """This function implements SincConv (SincNet).

    M. Ravanelli, Y. Bengio, "Speaker Recognition from raw waveform with
    SincNet", in Proc. of  SLT 2018 (https://arxiv.org/abs/1808.00158)

    Arguments
    ---------
    input_shape : tuple
        The shape of the input. Alternatively use ``in_channels``.
    in_channels : int
        The number of input channels. Alternatively use ``input_shape``.
    out_channels : int
        It is the number of output channels.
    kernel_size: int
        Kernel size of the convolutional filters.
    stride : int
        Stride factor of the convolutional filters. When the stride factor > 1,
        a decimation in time is performed.
    dilation : int
        Dilation factor of the convolutional filters.
    padding : str
        (same, valid, causal). If "valid", no padding is performed.
        If "same" and stride is 1, output shape is the same as the input shape.
        "causal" results in causal (dilated) convolutions.
    padding_mode : str
        This flag specifies the type of padding. See torch.nn documentation
        for more information.
    groups : int
        This option specifies the convolutional groups. See torch.nn
        documentation for more information.
    bias : bool
        If True, the additive bias b is adopted.
    sample_rate : int,
        Sampling rate of the input signals. It is only used for sinc_conv.
    min_low_hz : float
        Lowest possible frequency (in Hz) for a filter. It is only used for
        sinc_conv.
    min_low_hz : float
        Lowest possible value (in Hz) for a filter bandwidth.

    Example
    -------
    >>> inp_tensor = torch.rand([10, 16000])
    >>> conv = SincConv(input_shape=inp_tensor.shape, out_channels=25, kernel_size=11)
    >>> out_tensor = conv(inp_tensor)
    >>> out_tensor.shape
    torch.Size([10, 16000, 25])
    """
```

Performance

- SpeechBrain is released with many recipes on popular datasets that achieve **competitive** or **state-of-the-art results** in many tasks.

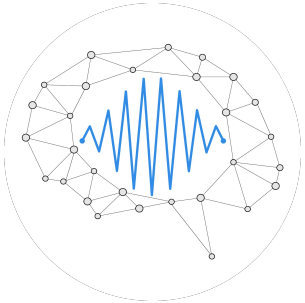
Dataset	Task	System	Performance
LibriSpeech	Speech Recognition	CNN + Transformer	**WER=2.35% (test-clean)**
LibriSpeech	Speech Recognition	wav2vec2 + CTC	**WER=1.90% (test-clean)**
TIMIT	Speech Recognition	CRDNN + distillation	**PER=13.1% (test)**
TIMIT	Speech Recognition	wav2vec2 + CTC/Att.	**PER=8.04% (test)**
VoxCeleb2	Speaker Verification	ECAPA-TDNN	**EER=0.69% (vox1-test)**
AMI	Speaker Diarization	ECAPA-TDNN	**DER=2.13% (lapel-mix)**
VoiceBank	Speech Enhancement	MetricGAN+	**PESQ=3.08 (test)**
WSJ2MIX	Speech Separation	SepFormer	**SDRi=22.6 dB (test)**
WSJ3MIX	Speech Separation	SepFormer	**SDRi=20.0 dB (test)**

Other Features

- Multi-GPU Training
- Dynamic Batching (based on bucketing)
- Large Scale Experiments with Webdataset
- On-the-fly augmentation and feature computation (on both CPU and GPU)
- Interface with HuggingFace for pre-training with large models (e.g, wav2vec)

Project History

- **Project Announcement:** September 2019
- **Development Started:** February 2020



- **Beta testing:** February 2021

- **Public Release:** March 2021
- **Now**

46 recipes for 20 datasets.
2.8k stars on github (in just 5 months)

SpeechBrain is growing very fast!

Reference

https://arxiv.org/abs/2106.04624

SpeechBrain: A General-Purpose Speech Toolkit

Mirco Ravanelli[1,2], Titouan Parcollet[3,16], Peter Plantinga[4], Aku Rouhe[5], Samuele Cornell[6], Loren Lugosch[1,7], Cem Subakan[1], Nauman Dawalatabad[8], Abdelwahab Heba[9], Jianyuan Zhong[1], Ju-Chieh Chou[10*], Sung-Lin Yeh[11*], Szu-Wei Fu[12], Chien-Feng Liao[12], Elena Rastorgueva[13†], François Grondin[14], William Aris[14], Hwidong Na[15], Yan Gao[16], Renato De Mori[3,7], and Yoshua Bengio[1,2]

Website: speechbrain.github.io/

Code: github.com/speechbrain/speechbrain

Tutorials: github.com/speechbrain/speechbrain

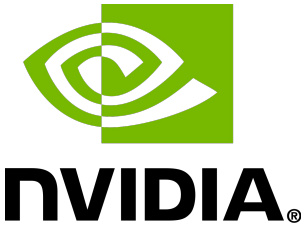
Pretrained models: huggingface.co/speechbrain

Sponsors


Mila


SAMSUNG


DOLBY

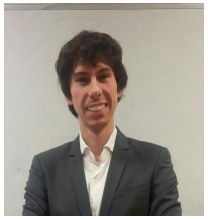

NVIDIA


NUANCE


ViaDialog
Customer Interactions Management

Institutional Partners

Core Development Team


Mirco Ravanelli


Titouan Parcollet


Peter Plantinga


Cem Sübakan


Chien-Feng Liao

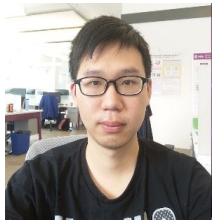

Szu-Wei Fu


Elena Rastorgueva


Loren Lugosch


Nauman Dawalatabad


Aku Rouhe


Ju-Chieh Chou


Hwidong Na


Abdel Heba


Samuele Cornell


Sung-Lin Yeh


Francois Grondin


William Aris


Yan Gao

open-source toolkits

ESPNet

Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Note that this presentation is a summary/collection of recent activities in the ESPnet toolkit

Please also check individual reports

ESPnet launched in December 2017

Our initial report at Interspeech 2018

ESPnet: End-to-End Speech Processing Toolkit

Shinji Watanabe[1], *Takaaki Hori*[2], *Shigeki Karita*[3], *Tomoki Hayashi*[4], *Jiro Nishitoba*[5], *Yuya Unno*[6], *Nelson Enrique Yalta Soplin*[7], *Jahn Heymann*[8], *Matthew Wiesner*[1], *Nanxin Chen*[1], *Adithya Renduchintala*[1], *Tsubasa Ochiai*[9],

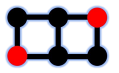
[1]Johns Hopkins University, [2]Mitsubishi Electric Research Laboratories, [3]NTT Communication Science Laboratories, [4]Nagoya University, [5]Retrieva, Inc., [6]Preferred Networks, Inc., [7]Waseda University, [8]Paderborn University, [9]Doshisha University

shinjiw@jhu.edu

Abstract

This paper introduces a new open source platform for end-to-end speech processing named ESPnet. ESPnet mainly focuses on end-to-end automatic speech recognition (ASR), and adopts widely-used dynamic neural network toolkits, Chainer and PyTorch, as a main deep learning engine. ESPnet also follows the Kaldi ASR toolkit style for data processing, feature extraction/format, and recipes to provide a complete setup for speech

network [13, 14, 15, 16]. Attention-based methods use an attention mechanism to perform alignment between acoustic frames and recognized symbols, while CTC uses Markov assumptions to efficiently solve sequential problems by dynamic programming. ESPnet adopts hybrid CTC/attention end-to-end ASR [17], which effectively utilizes the advantages of both architectures in training and decoding. During training, we employ the multiobjective learning framework to improve robustness on irregular alignments and achieve fast convergence. During de-

ESPnet  ESPnet launched in December 2017

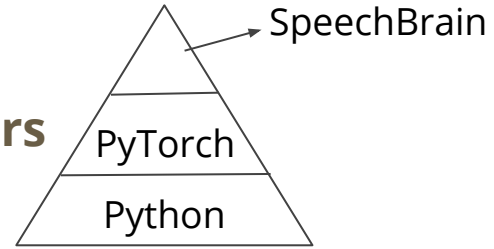
https://github.com/espnet/espnet

- **Open source** (Apache2.0) end-to-end speech processing toolkit
- Major concept: **Reproducibility**
  - Leverage our end-to-end ASR experience to the community
  - Accelerates end-to-end research for speech researchers
- PyTorch based dynamic neural network toolkit as an engine
  - Easily develop novel neural network architecture
- Follows the **famous speech recognition toolkit, Kaldi, style**
  - Smoothly port ASR experiments from Kaldi to ESPnet with the common data processing, feature extraction/format
  - Recipes to provide a complete setup for speech processing experiments
- **The project is greatly accelerated in these three years**

ESPnet ESPnet launched in December 2017

https://github.com/espnet/espnet

- **Open source** (Apache2.0) end-to-end speech processing toolkit
- Major concept: **Reproducibility**
  - Leverage our end-to-end ASR experience to the community
  - Accelerates end-to-end research for speech researchers
- PyTorch based dynamic neural network toolkit as an engine
  - Easily develop novel neural network architecture
- Follows the **famous speech recognition toolkit, Kaldi, style**
  - Smoothly port ASR experiments from Kaldi to ESPnet with the common data processing, feature extraction/format
  - Recipes to provide a complete setup for speech processing experiments
- **The project is greatly accelerated in these three years**

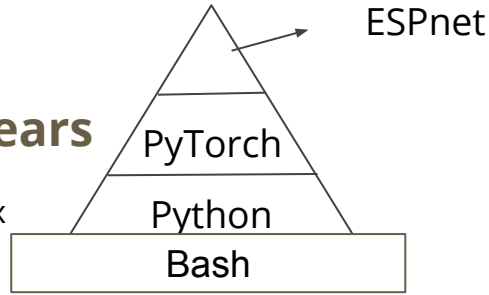


ESPnet ESPnet launched in December 2017

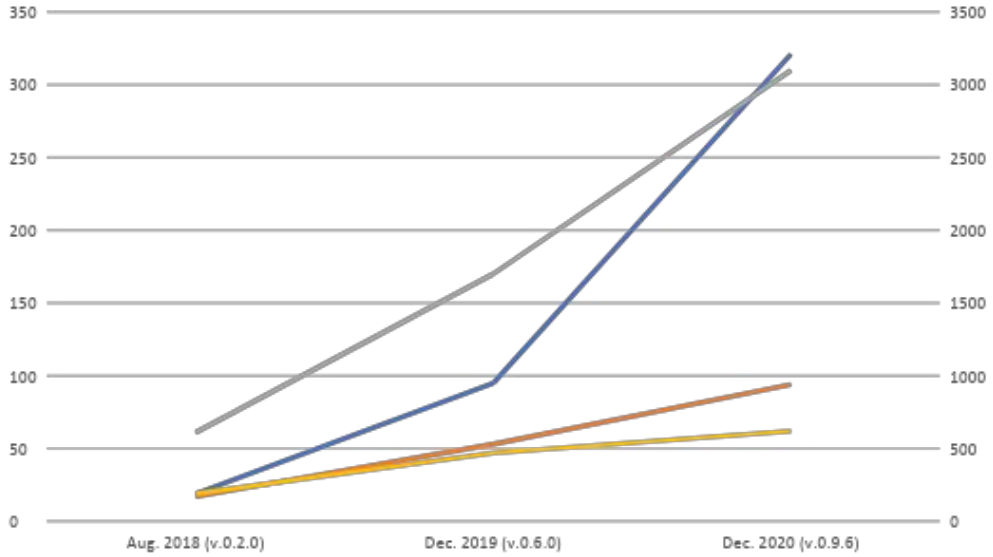
https://github.com/espnet/espnet

- **Open source** (Apache2.0) end-to-end speech processing toolkit
- Major concept: **Reproducibility**
  - Leverage our end-to-end ASR experience to the community
  - Accelerates end-to-end research for speech researchers
- PyTorch based dynamic neural network toolkit as an engine
  - Easily develop novel neural network architecture
- Follows the **famous speech recognition toolkit, Kaldi, style**
  - Smoothly port ASR experiments from Kaldi to ESPnet with the common data processing, feature extraction/format
  - Recipes to provide a complete setup for speech processing experiments
- **The project is greatly accelerated in these three years**

Good compatibility/scalability with Linux systems and job schedular but not pythonic and hard for begginers



Activity statistics (from 2018 to 2020)



- **Citations**, **contributors**, **recipes** (examples), and **stars** are all growing

i.e.,

- Developers have increasingly supported the development of ESPnet
- has been used in various research groups and contributed a lot to speech research activities
- ESPnet 4.1 stars (today)
  - Kaldi 10.7K stars
  - Nvidia NeMo 3.1K stars
  - SpeechBrain 2.8K stars
  - Google Lingvo 2.3K stars
  - FairSeq: 13.6K stars

Major change in the internal framework
From ESPnet1 to ESPnet2

- ESPnet2: a new system for DNN training to extend our system from v.0.7.0
- Mostly refactoring, but which enables major update to deal with
  - **Distributed training**
  - **On-the-fly feature extraction** from the raw waveform
  - Improved the **scalability**
- Improved software workflow by enhancing
  - **continuous integration**, **enriching documentation**, supporting the **docker**, **pip install**
  - **model zoo** (mostly zenodo compared with hugging face HUB used in SpeechBrain. hugging face HUB migration is on-going)
  - **WandB** based experiment monitoring and sharing
- The migration is ongoing (ASR and TTS are already finished)


Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

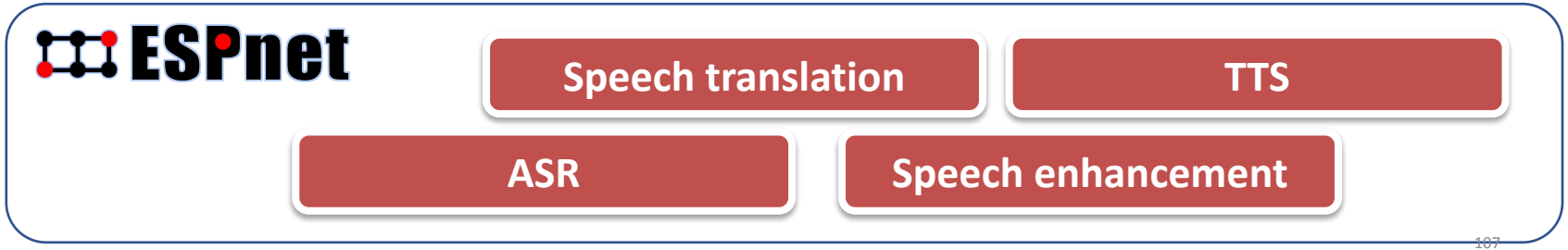
Broadened Applications

- ESPnet (**ASR+X**) covers the following topics complementally

- Why can one toolkit support such wide-ranges of applications?

Broadened Applications

- ESPnet (**ASR+X**) covers the following topics complementally



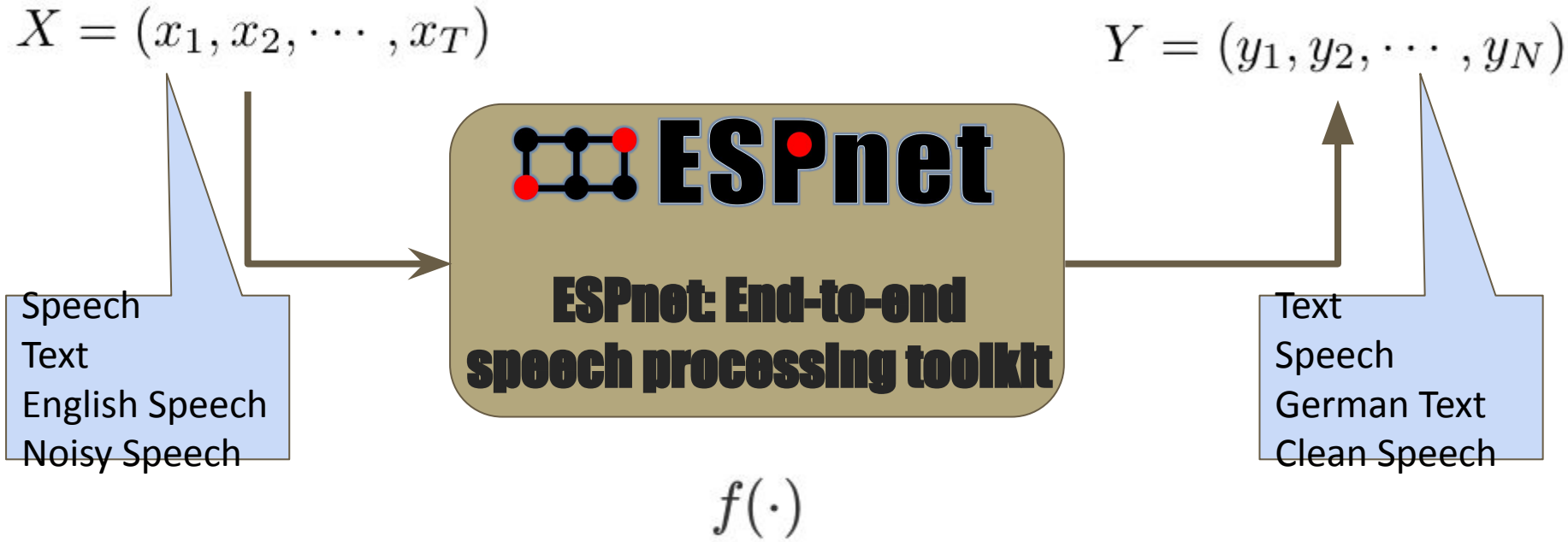
- Why can one toolkit support such wide-ranges of applications?

Unified form → Unified software design

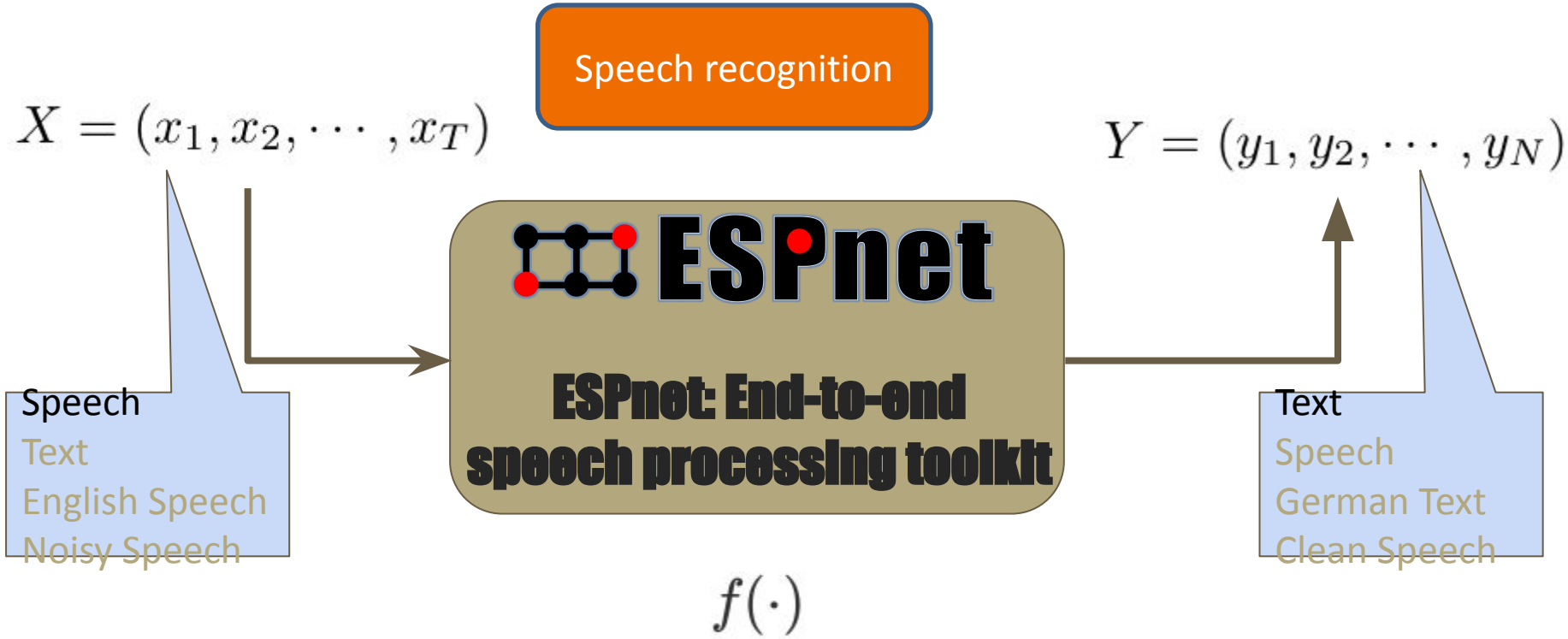
- We design ESPnet by leveraging a **unified** mathematical **form** of **sequence (X) to sequence (Y) transformation f**

$$X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N)$$

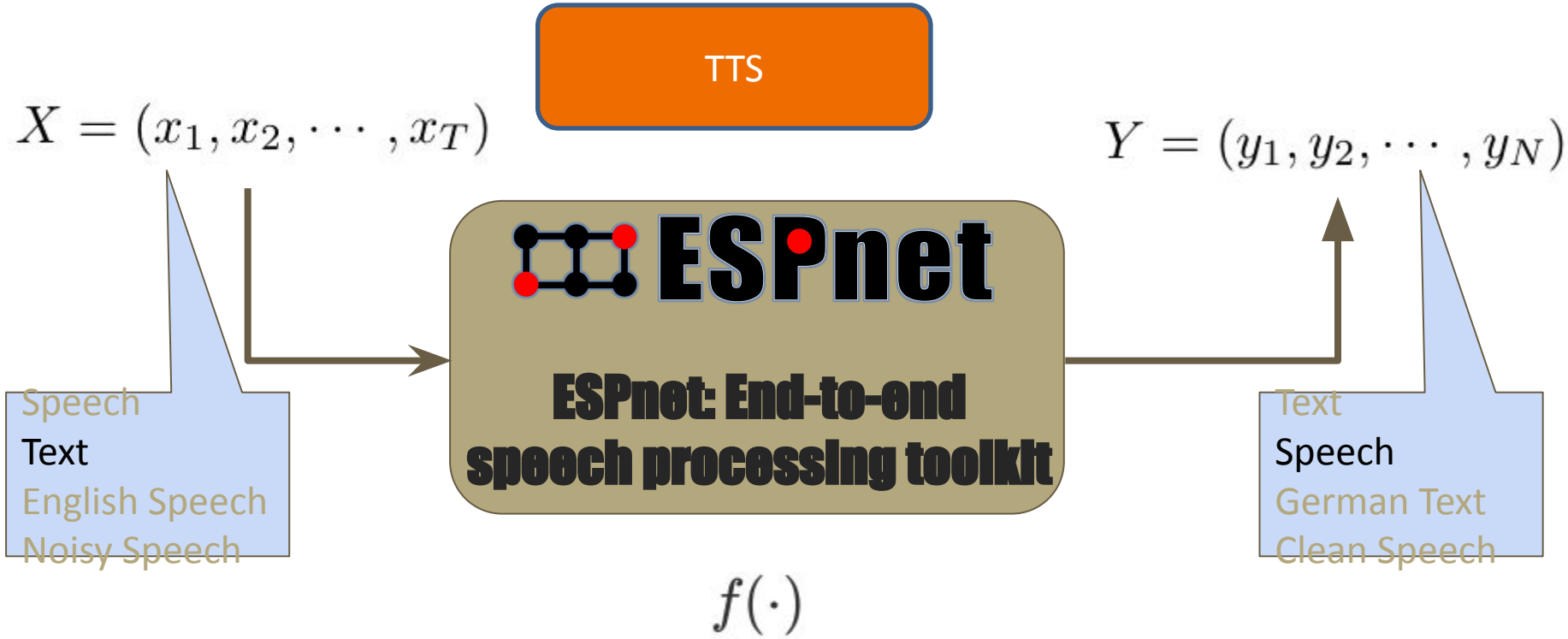
Unified form → Unified software design



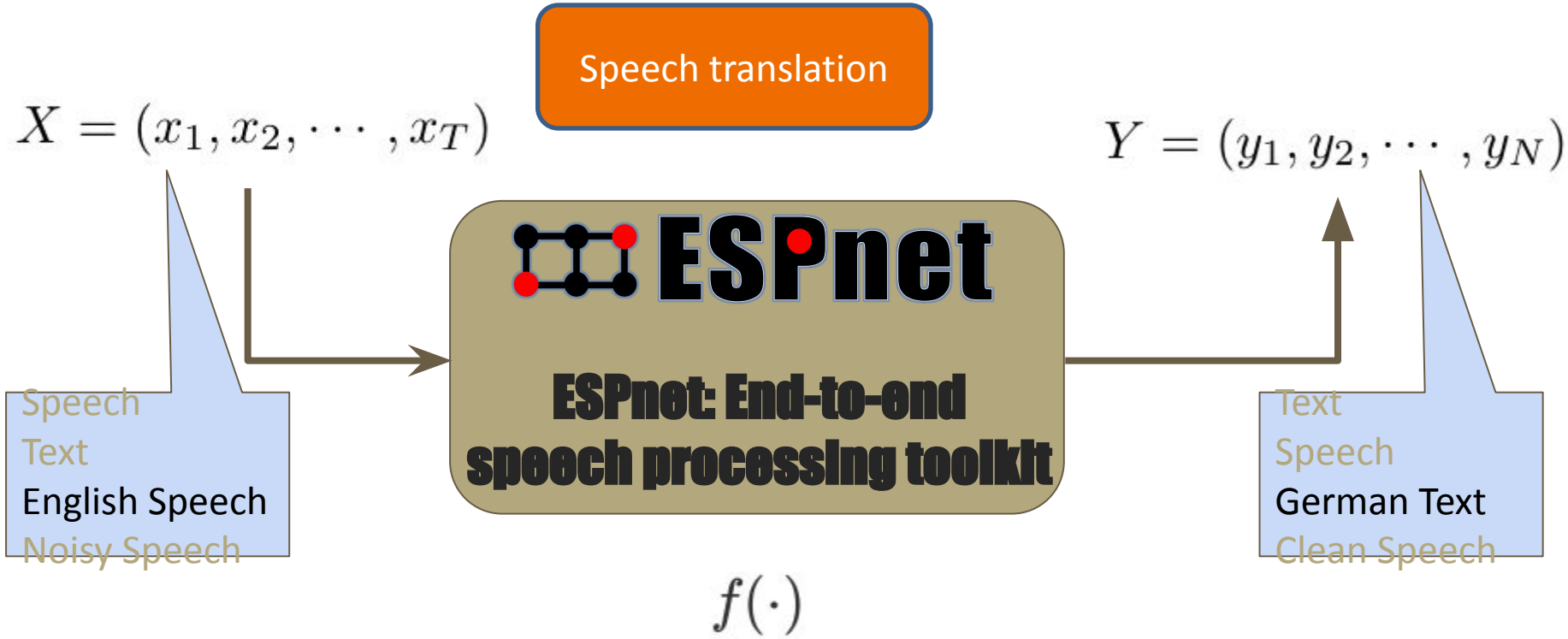
Unified form → Unified software design



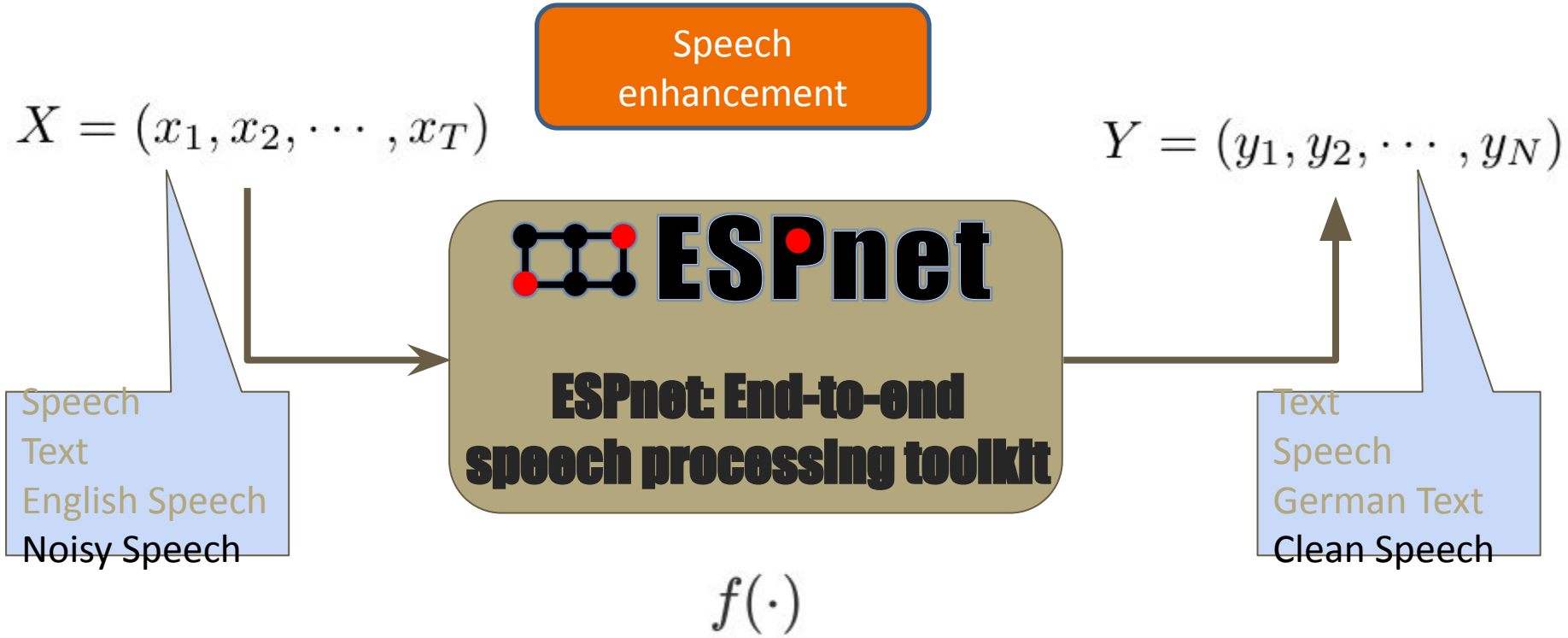
Unified form → Unified software design



Unified form → Unified software design



Unified form → Unified software design



Unified form → Unified software design

$X = (x_1, x_2, \cdots, x_T)$

$Y = (y_1, y_2, \cdots, y_N)$

CTC
Attention
Joint
CTC/Attention
RNN-T
Transformer, etc.

$f(\cdot)$

Unified form → Unified software design

$$X = (x_1, x_2, \cdots, x_T)$$

$$Y = (y_1, y_2, \cdots, y_N)$$

CTC
Attention
Joint
CTC/Attention
RNN-T
Transformer, etc.

$f(\cdot)$

Especially **Joint CTC/Attention**
significantly improves ASR results
(but makes inference complicated)

Unified form → Unified software design

$$X = (x_1, x_2, \cdots, x_T)$$

$$Y = (y_1, y_2, \cdots, y_N)$$

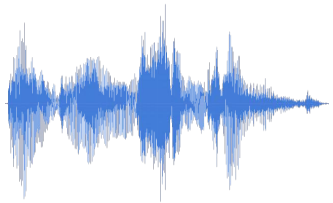
- Many speech processing applications can be **unified** based on seq2seq
- **Nemo, Fairseq, Lingvo, Espresso, SpeechBrain, Asteroid** and other toolkits also fully make use of these functions
- We are closely collaborating/interacting with them

Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Automatic speech recognition (ASR)

Speech Recogition

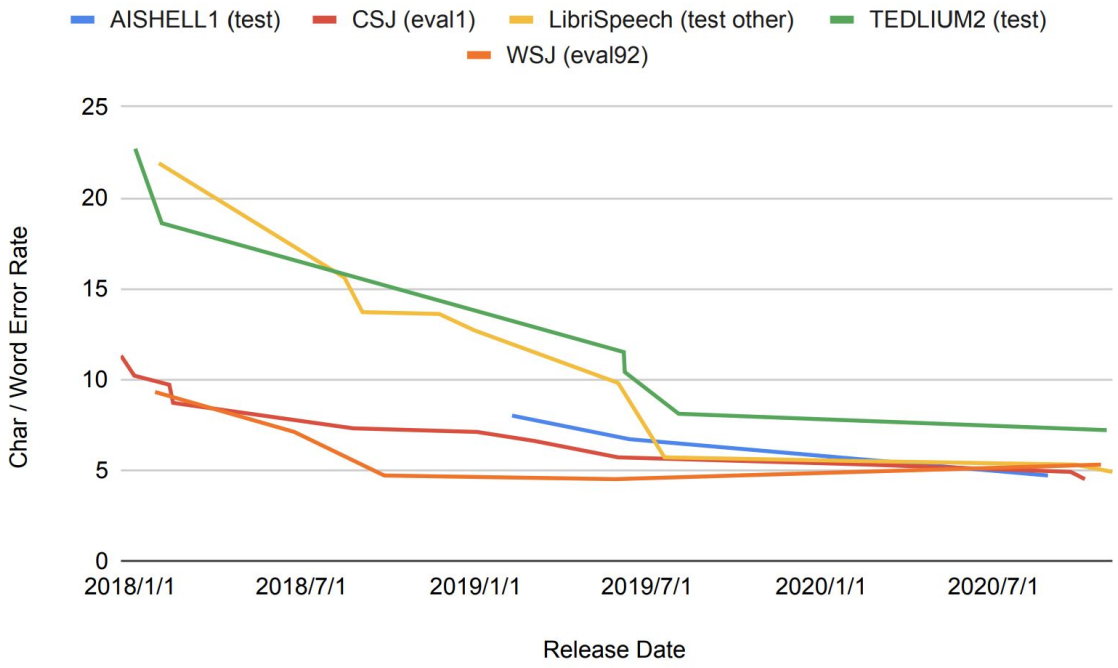


transcribe

I want to pet a cat

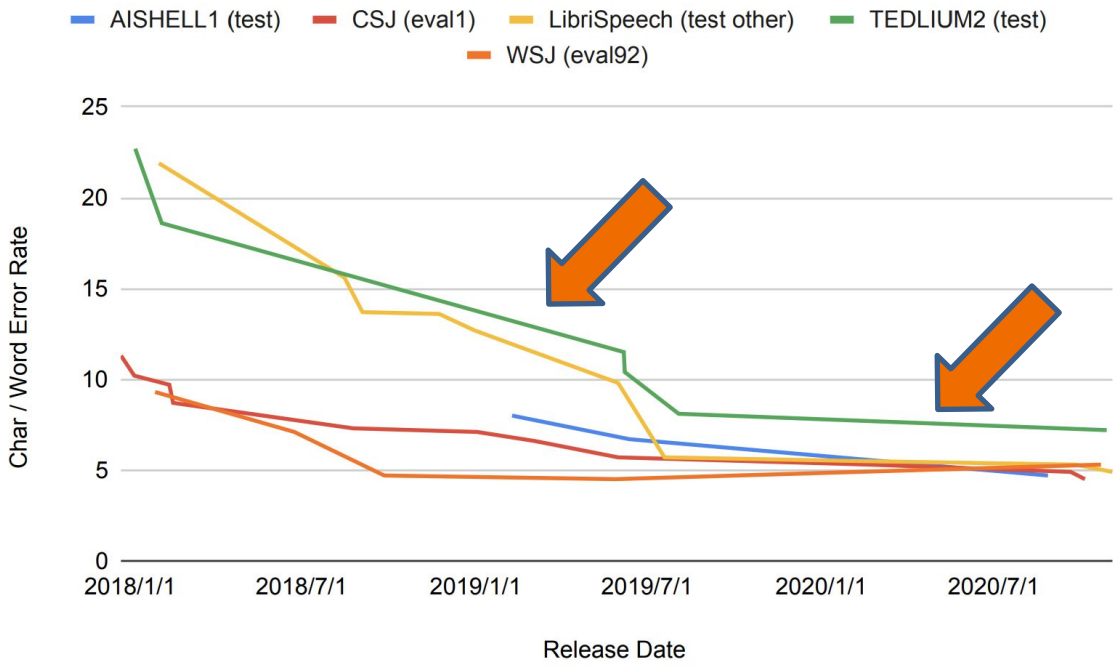
Maintaining state-of-the-art performance in ASR

Error rate: Lower is better

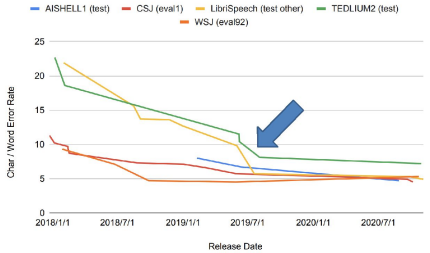


Maintaining state-of-the-art performance in ASR

Error rate: Lower is better



ESPnet Transformer



A COMPARATIVE STUDY ON TRANSFORMER VS RNN IN SPEECH APPLICATIONS

Shigeki Karita[1],
(Alphabetical Order) *Nanxin Chen*[3], *Tomoki Hayashi*[5,6], *Takaaki Hori*[7], *Hirofumi Inaguma*[8], *Ziyan Jiang*[3], *Masao Someki*[5], *Nelson Enrique Yalta Soplin*[2], *Ryuichi Yamamoto*[4], *Xiaofei Wang*[3], *Shinji Watanabe*[3], *Takenori Yoshimura*[5,6], *Wangyou Zhang*[9]
[1]NTT Communication Science Laboratories, [2]Waseda University, [3]Johns Hopkins University,
[4]LINE Corporation, [5]Nagoya University, [6]Human Dataware Lab. Co., Ltd.,
[7]Mitsubishi Electric Research Laboratories, [8]Kyoto University, [9]Shanghai Jiao Tong University

ABSTRACT

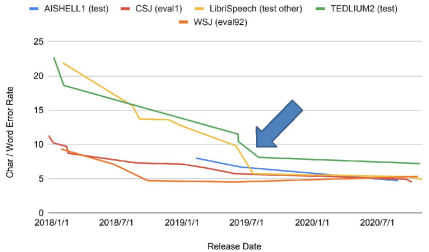
Sequence-to-sequence models have been widely used in end-to-end speech processing, for example, automatic speech recognition (ASR), speech translation (ST), and text-to-speech (TTS). This paper focuses on an emergent sequence-to-sequence model called Transformer, which achieves state-of-the-art performance in neural machine translation and other natural language processing applications. We undertook intensive studies in which we experimentally compared and analyzed Transformer and conventional recurrent

In our speech application experiments, we investigate several aspects of Transformer and RNN-based systems. For example, we measure the word/character/regression error from the ground truth, training curve, and scalability for multiple GPUs.

The contributions of this work are:

- We conduct a larges-scale comparative study on Transformer and RNN with significant performance gains especially for the ASR related tasks.
- We explain our training tips for Transformer in speech applica-

ESPnet Transformer



A COMPARATIVE STUDY ON TRANSFORMER VS RNN IN SPEECH APPLICATIONS

Shigeki Karita[1],
(Alphabetical Order) *Nanxin Chen*[3], *Tomoki Hayashi*[5,6], *Takaaki Hori*[7], *Hirofumi Inaguma*[8], *Ziyan Jiang*[3], *Masao Someki*[5], *Nelson Enrique Yalta Soplin*[2], *Ryuichi Yamamoto*[4], *Xiaofei Wang*[3], *Shinji Watanabe*[3], *Takenori Yoshimura*[5,6], *Wangyou Zhang*[9]
[1]NTT Communication Science Laboratories, [2]Waseda University, [3]Johns Hopkins University,
[4]LINE Corporation, [5]Nagoya University, [6]Human Dataware Lab. Co., Ltd.,
[7]Mitsubishi Electric Research Laboratories, [8]Kyoto University, [9]Shanghai Jiao Tong University

ABSTRACT

Sequence-to-sequence models have been wid
end speech processing, for example, automatic
(ASR), speech translation (ST), and text-to-
paper focuses on an emergent sequence-to-seq
Transformer, which achieves state-of-the-art per
machine translation and other natural language processing applications. We undertook intensive studies in which we experimentally compared and analyzed Transformer and conventional recurrent

and RNN with significant performance gains especially for the ASR related tasks.

- We explain our training tips for Transformer in speech applications: ASR, TTS and ST

- One of the first success in the speech areas
- The performance was boosted

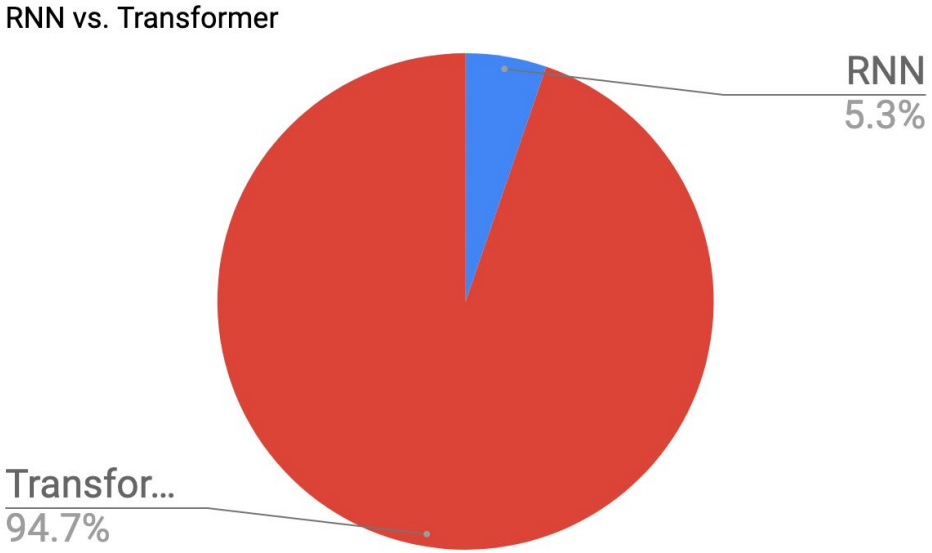
Transformer boosted the performance

- Improve the performance from RNN with **13** ASR tasks among 15 tasks
- Reaching the Kaldi performance (state-of-the-art **non** end-to-end ASR) in half of tasks

dataset	token	error	Kaldi	Our RNN	Our Transformer
AISHELL	char	CER	N/A / 7.4	6.8 / 8.0	**6.0 / 6.7**
AURORA4	char	WER	(*) 3.6 / 7.7 / 10.0 / 22.3	3.5 / 6.4 / 5.1 / 12.3	**3.3 / 6.0 / 4.5 / 10.6**
CSJ	char	CER	(*) 7.5 / 6.3 / 6.9	6.6 / 4.8 / 5.0	**5.7 / 4.1 / 4.5**
CHiME4	char	WER	**6.8 / 5.6 / 12.1 / 11.4**	9.5 / 8.9 / 18.3 / 16.6	9.6 / 8.2 / 15.7 / 14.5
CHiME5	char	WER	**47.9 / 81.3**	59.3 / 88.1	60.2 / 87.1
Fisher-CALLHOME Spanish	char	WER	N/A	27.9 / 27.8 / 25.4 / 47.2 / 47.9	**27.0 / 26.3 / 24.4 / 45.3 / 46.2**
HKUST	char	CER	23.7	27.4	**23.5**
JSUT	char	CER	N/A	20.6	**18.7**
LibriSpeech	BPE	WER	3.9 / 10.4 / 4.3 / 10.8	3.1 / 9.9 / 3.3 / 10.8	**2.2 / 5.6 / 2.6 / 5.7**
REVERB	char	WER	18.2 / 19.9	24.1 / 27.2	**15.5 / 19.0**
SWITCHBOARD	BPE	WER	**18.1 / 8.8**	28.5 / 15.6	26.0 / 14.0
TED-LIUM2	BPE	WER	**9.0** / 9.0	11.2 / 11.0	9.3 / **8.1**
TED-LIUM3	BPE	WER	**6.2 / 6.8**	14.3 / 15.0	9.7 / 8.0
VoxForge	char	CER	N/A	12.9 / 12.6	**9.4 / 9.1**
WSJ	char	WER	**4.3 / 2.3**	7.0 / 4.7	6.8 / 4.4

Transformer boosted the performance

- Improve the performance from RNN with **13** ASR tasks among 15 tasks
- Reaching the Ka ASR) in half of



dataset					Our Transformer
AISHELL					**6.0 / 6.7**
AURORA4					**3.3 / 6.0 / 4.5 / 10.6**
CSJ					**5.7 / 4.1 / 4.5**
CHiME4					9.6 / 8.2 / 15.7 / 14.5
CHiME5					60.2 / 87.1
Fisher-CALLHOME Spanish					**7.0 / 26.3 / 24.4 / 45.3 / 46.2**
HKUST					**23.5**
JSUT					**18.7**
LibriSpeech					**2.2 / 5.6 / 2.6 / 5.7**
REVERB					**15.5 / 19.0**
SWITCHBOARD					26.0 / 14.0
TED-LIUM2					9.3 / **8.1**
TED-LIUM3					9.7 / 8.0
VoxForge	char	CER	N/A	12.9 / 12.6	**9.4 / 9.1**
WSJ	char	WER	**4.3 / 2.3**	7.0 / 4.7	6.8 / 4.4

Experiments (~ 1000 hours)
Librispeech at April 2019

Toolkit	dev_clean	dev_other	test_clean	test_other
Facebook wav2letter++	3.1	10.1	3.4	11.2
RWTH RASR	2.9	8.8	3.1	9.8
Nvidia Jasper	2.6	7.6	2.8	7.8
Google SpecAug.	N/A	N/A	**2.5**	5.8

- Very impressive results by Google

Experiments (~ 1000 hours)
Librispeech at August 2019

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Facebook wav2letter++	3.1	10.1	3.4	11.2
RWTH RASR	2.9	8.8	3.1	9.8
Nvidia Jasper	2.6	7.6	2.8	7.8
Google SpecAug.	N/A	N/A	**2.5**	5.8
ESPnet	**2.2**	**5.6**	2.6	**5.7**

- **Reached Google's best performance by community-driven efforts**

GAFAM

GAFAM

ESPnet

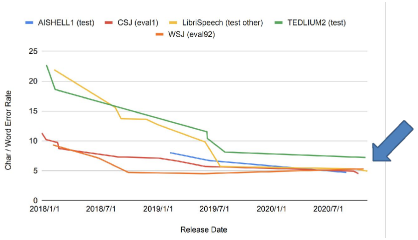
Experiments (~ 1000 hours) in August 2019
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Facebook wav2letter++	3.1	10.1	3.4	11.2
RWTH RASR	2.9	8.8	3.1	9.8
Nvidia Jasper	2.6	7.6	2.8	7.8
Google SpecAug.	N/A	N/A	**2.5**	5.8
ESPnet	**2.2**	**5.6**	2.6	**5.7**

Experiments (~ 1000 hours) in March 2020
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Facebook wav2letter++	3.1	10.1	3.4	11.2
RWTH RASR	2.9	8.8	3.1	9.8
Nvidia Jasper	2.6	7.6	2.8	7.8
Google SpecAug.	N/A	N/A	2.5	5.8
ESPnet	2.2	5.6	2.6	5.7
Google **Conformer**	**1.9**	**4.4**	**1.9**	**3.9**

ESPnet Conformer



RECENT DEVELOPMENTS ON ESPNET TOOLKIT BOOSTED BY CONFORMER

Pengcheng Guo[1,4], *Florian Boyer*[2,3], *Xuankai Chang*[4], *Tomoki Hayashi*[5], *Yosuke Higuchi*[6]
Hirofumi Inaguma[7], *Naoyuki Kamo*[8], *Chenda Li*[9], *Daniel Garcia-Romero*[4], *Jiatong Shi*[4]
Jing Shi[4,10], *Shinji Watanabe*[4], *Kun Wei*[1], *Wangyou Zhang*[9], *Yuekai Zhang*[4]

[1]Northwestern Polytechnical University, [2]LaBRI, University of Bordeaux, [3] Airudit
[4]Johns Hopkins University, [5]Human Dataware Lab. Co., Ltd.
[6]Waseda University, [7]Kyoto University, [8]NTT Corporation [9]Shanghai Jiao Tong University
[10]Institute of Automation, Chinese Academy of Sciences

ABSTRACT

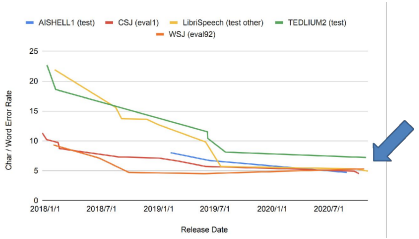
In this study, we present recent developments on ESPnet: End-to-End Speech Processing toolkit, which mainly involves a recently proposed architecture called Conformer, Convolution-augmented Transformer. This paper shows the results for a wide range of end-to-end speech processing applications, such as automatic speech recognition (ASR), speech translations (ST), speech separation (SS) and text-to-speech (TTS). Our experiments reveal various training tips and significant performance benefits obtained with the Conformer on different tasks. These results are competitive or even outperform the current state-of-art Transformer models. We are preparing to release all-in-one recipes using open source and publicly available corpora for all the above tasks with pre-trained

of publicly available corpora and try our best to share the practical guides (e.g., learning rate, hyper-parameters, network structure) on the use of Conformer. We also prepare to release the reproducible recipes and state-of-the-art setups to the community to succeed our exciting outcomes.

The contributions of this study include:

- We extend the Conformer architecture to various end-to-end speech processing applications and conduct comparative experiments with Transformer.
- We share our practical guides for the training of Conformer, like learning rate, kernel size of Conformer block, and model architectures, etc.

ESPnet Conformer



RECENT DEVELOPMENTS ON ESPNET TOOLKIT BOOSTED BY CONFORMER

Pengcheng Guo[1,4], *Florian Boyer*[2,3], *Xuankai Chang*[4], *Tomoki Hayashi*[5], *Yosuke Higuchi*[6]
Hirofumi Inaguma[7], *Naoyuki Kamo*[8], *Chenda Li*[9], *Daniel Garcia-Romero*[4], *Jiatong Shi*[4]
Jing Shi[4,10], *Shinji Watanabe*[4], *Kun Wei*[1], *Wangyou Zhang*[9], *Yuekai Zhang*[4]

[1]Northwestern Polytechnical University, [2]LaBRI, University of Bordeaux, [3] Airudit
[4]Johns Hopkins University, [5]Human Dataware Lab. Co., Ltd.
[6]Waseda University, [7]Kyoto University, [8]NTT Corporation [9]Shanghai Jiao Tong University
[10]Institute of Automation, Chinese Academy of Sciences

ABSTRACT

In this study, we present recent developments
End Speech Processing toolkit, which mainly
proposed architecture called Conformer, Co
Transformer. This paper shows the results for
to-end speech processing applications, such
recognition (ASR), speech translations (ST), sp
and text-to-speech (TTS). Our experiments r
ing tips and significant performance benefits obtained with the Conformer on different tasks. These results are competitive or even outperform the current state-of-art Transformer models. We are preparing to release all-in-one recipes using open source and

of publicly available corpora and try our best to share the practical

- We try to follow Google's conformer work
- Also apply conformer to ST, TTS, as well as ASR

periments with Transformer.

- We share our practical guides for the training of Conformer, like learning rate, kernel size of Conformer block, and model architectures, etc.

Experiments (~ 1000 hours) in October 2020
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Facebook wav2letter++	3.1	10.1	3.4	11.2
RWTH RASR	2.9	8.8	3.1	9.8
Nvidia Jasper	2.6	7.6	2.8	7.8
Google SpecAug.	N/A	N/A	2.5	5.8
ESPnet	2.2	5.6	2.6	5.7
Google Conformer	**1.9**	**4.4**	**1.9**	**3.9**
ESPnet Conformer	**1.9**	4.6	2.1	4.7

We continue to work on catching up SOTA.

Is the story ended? → No...

New weapon (Wav2vec2.0, HuBERT)

Experiments (~ 1000 hours) in March 2021
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Google SpecAug.	N/A	N/A	2.5	5.8
ESPnet	2.2	5.6	2.6	5.7
Google Conformer	1.9	4.4	1.9	3.9
ESPnet Conformer	1.9	4.6	2.1	4.7
Facebook wav2vec2.0 (60k LibriVox)	**1.6**	**3.0**	**1.8**	**3.3**
Facebook Hubert (60k LibriVox)	1.7	**3.0**	1.9	3.5

Self-supervised training further improves the performance

SUPERB

S3PRL toolkit

+

ESPnet

Experiments (~ 1000 hours) in July 2021
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Google SpecAug.	N/A	N/A	2.5	5.8
ESPnet	2.2	5.6	2.6	5.7
Google Conformer	1.9	4.4	1.9	3.9
ESPnet Conformer	1.9	4.6	2.1	4.7
Facebook wav2vec2.0 (60k LibriVox)	**1.6**	**3.0**	**1.8**	**3.3**
Facebook Hubert (60k LibriVox)	1.7	**3.0**	1.9	3.5
ESPnet + S3PRL Hubert (60k LibriVox)	1.7	3.4	**1.8**	3.6

Reaching SOTA again!!!

Experiments (~ 1000 hours) in July 2021
Librispeech

Toolkit/Method	dev_clean	dev_other	test_clean	test_other
Google SpecAug.	N/A	N/A	2.5	5.8
ESPnet	2.2	5.6	2.6	5.7
Google Conformer	1.9	4.4	1.9	3.9
ESPnet Conformer	1.9	4.6	2.1	4.7
Facebook wav2vec2.0 (60k LibriVox)	**1.6**	**3.0**	**1.8**	**3.3**
Facebook Hubert (60k LibriVox)	1.7	**3.0**	1.9	3.5
ESPnet + S3PRL Hubert (60k LibriVox)	1.7	3.4	**1.8**	3.6

Note that this was achieved by great helps from Google, Facebook, and SUPERB

Good example of “Collapetition”

= Collaboration + Competition

Discussions for catching SOTA

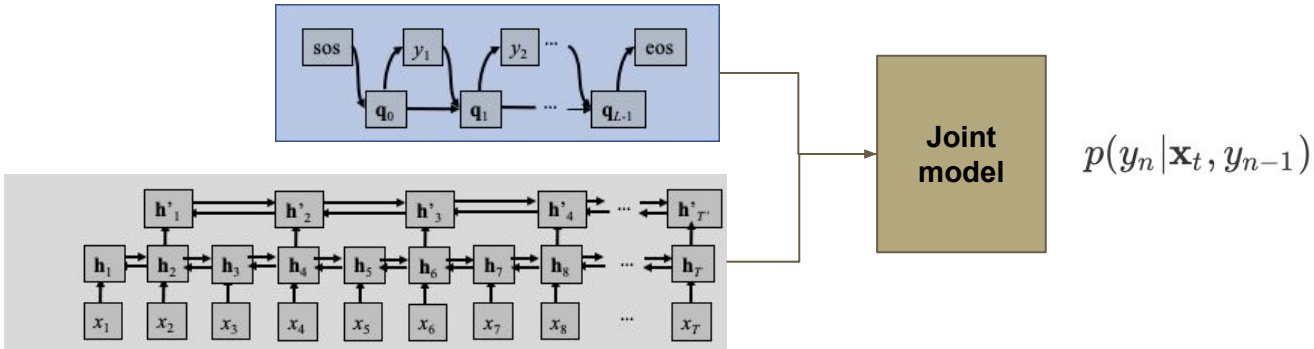
- We should not give up to catch up SOTA
- We developed Transformer -> Conformer -> Hubert
  - All activities have been supported by various collaborators (Collapetition)
  - We have further interactions with others including **SpeechBrain, K2, etc.**
- Now self-supervised models are ready in ESPnet through **SUPERB S3PRL**!!!
  - ESPnet can become one of the downstream tasks in SUPERB
  - It can be applied to **LeBenchmark**

Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

RNN/Transformer Transducer [Boyer+(2021)]

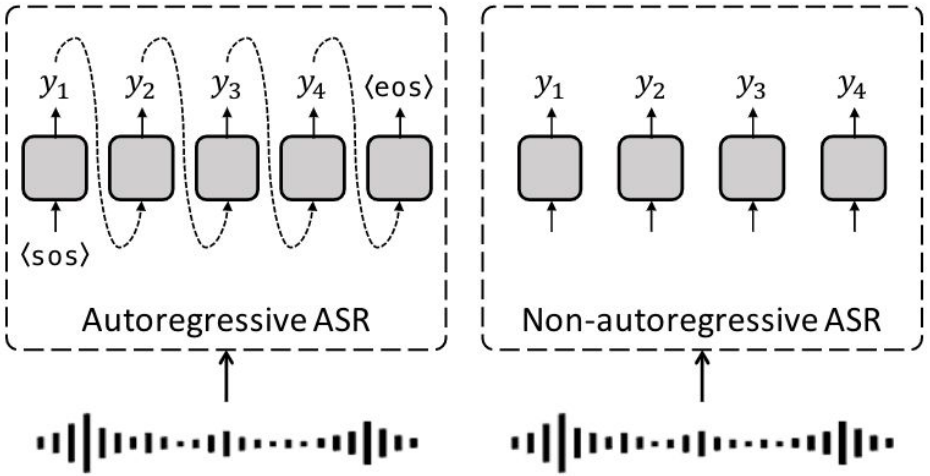
- RNN or transformer transducer (widely used in industry)
  - Good for **streaming**



  - ESPnet has **various architecture supports** (LSTM, CNN, Transformer, conformer)
  - Also supports **various beam search algorithms, including K2 FST beam search (similar to SpeechBrain)**

Non-Autoregressive modeling [Higuchi+(2020)]

- The most complicated part in ASR: **Left-to-right beam search (several hundreds of lines)**
- BERT-like iterative mask predict
- Achieved 0.07 real time factor!
  (Takes only 70ms to decode
  - No software optimization
  - Just CPU
- **Only 20 lines for coding**



Today's talk

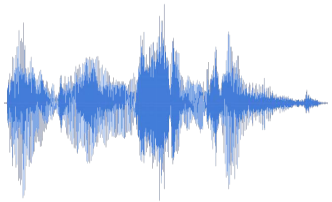
- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Text to speech (TTS)

Speech Synthesis or Text to speech

I want to pet a cat

synthesize



ESPnet TTS

ESPNET-TTS: UNIFIED, REPRODUCIBLE, AND INTEGRATABLE OPEN SOURCE END-TO-END TEXT-TO-SPEECH TOOLKIT

Tomoki Hayashi[1,2], *Ryuichi Yamamoto*[3], *Katsuki Inoue*[4], *Takenori Yoshimura*[1,2], *Shinji Watanabe*[5], *Tomoki Toda*[1], *Kazuya Takeda*[1], *Yu Zhang*[6], and *Xu Tan*[7]
[1]Nagoya University, [2]Human Dataware Lab. Co., Ltd., [3]LINE Corp.,
[4]Okayama University, [5]Johns Hopkins University, [6]Google AI, [7]Microsoft Research

ABSTRACT

This paper introduces a new end-to-end text-to-speech (E2E-TTS) toolkit named ESPnet-TTS, which is an extension of the open-source speech processing toolkit ESPnet. The toolkit supports state-of-the-art E2E-TTS models, including Tacotron 2, Transformer TTS, and FastSpeech, and also provides recipes inspired by the Kaldi automatic speech recognition (ASR) toolkit. The recipes are based on the design unified with the ESPnet ASR recipe, providing high reproducibility. The toolkit also provides pre-trained models and samples of all of the recipes so that users can use it as a baseline. Furthermore, the unified design enables the integration of ASR functions with TTS, e.g., ASR-based objective evaluation and semi-supervised learning with both ASR and TTS models. This paper describes the research purpose to make E2E-TTS systems more user-friendly and to accelerate research in this field. The toolkit not only supports state-of-the-art E2E-TTS models such as Tacotron 2 [6], Transformer TTS [8], and FastSpeech [9] but also provides Kaldi automatic speech recognition (ASR) toolkit [17] style recipes. The recipe is based on the design unified with the ASR recipe and includes all of the procedures required to reproduce the results. The toolkit provides a number of recipes for more than ten languages, which include single-speaker TTS as well as multi-speaker one and speaker adaptation. Pre-trained models and generated samples of all of the recipes are also provided so that users can easily use it as a baseline or perform TTS demonstrations. Furthermore, thanks to the unified design among TTS and ASR, we can easily integrate ASR functions with TTS, for example, ASR-based objective evaluation

ESPnet TTS

- Mainly focuses on the development of **text to mel-spectrogram** (text2mel) models.
  - It supports Tacotron2 and Transformer-TTS (AR), and FastSpeech and FastSpeech2 (non-AR)
  - Of course, we can easily switch to RNN, transformer, or conformer
- **Multi-speaker extensions** with X-vector and global style token.
- Users can quickly develop the **state-of-the-art baseline systems** for the research purpose
- A lot of **examples and demonstration systems**, which works in **real-time** for various languages, including English, Mandarin, and Japanese

https://colab.research.google.com/github/espnet/notebook/blob/master/espnet2_tts_realtime_demo.ipynb

- Again supported by a lot of helps from Google (Dr. Heiga Zen)!!!

Synthesis

```
# decide the input sentence by yourself
print(f"Input your favorite sentence in {lang}.")
x = input()

# synthesis
with torch.no_grad():
    start = time.time()
    wav, c, *_ = text2speech(x)
    wav = vocoder.inference(c)
rtf = (time.time() - start) / (len(wav) / fs)
print(f"RTF = {rtf:5f}")

# let us listen to generated samples
from IPython.display import display, Audio
display(Audio(wav.view(-1).cpu().numpy(), rate=fs))
```

```
Input your favorite sentence in English.
I'll talk about T T S  functions in E S P net
RTF = 0.025100
```

0:03 / 0:03

Voice conversion challenge 2020 baseline system

The Sequence-to-Sequence Baseline for the Voice Conversion Challenge 2020: Cascading ASR and TTS

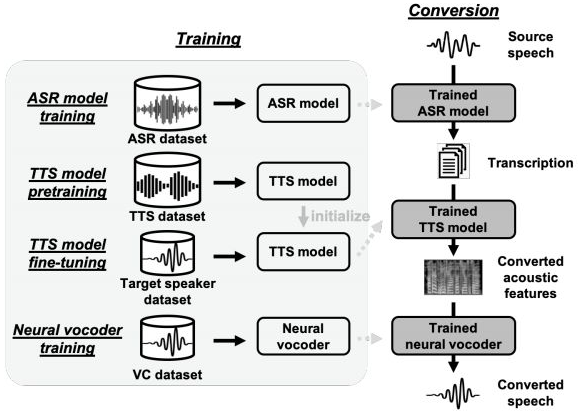
Wen-Chin Huang[1], *Tomoki Hayashi*[1], *Shinji Watanabe*[2], *Tomoki Toda*[1]

[1]Nagoya University, Japan
[2]Johns Hopkins University, USA

wen.chinhuang@g.sp.m.is.nagoya-u.ac.jp

Abstract

This paper presents the sequence-to-sequence (seq2seq) baseline system for the voice conversion challenge (VCC) 2020. We consider a naive approach for voice conversion (VC), which is to first transcribe the input speech with an automatic speech recognition (ASR) model, followed using the transcriptions to generate the voice of the target with a text-to-speech (TTS) model. We revisit this method under a sequence-to-sequence (seq2seq) framework by utilizing ESPnet, an open-source end-to-end speech processing toolkit, and the many well-configured pretrained models provided by the community. Official evaluation results show that our system comes out top among the participating systems in terms of conversion similarity, demonstrating the promising ability of seq2seq models to convert speaker identity. The implementation is



Voice conversion challenge 2020 baseline system

The Sequence-to-Sequence Baseline for the Voice Conversion Challenge 2020: Cascading ASR and TTS

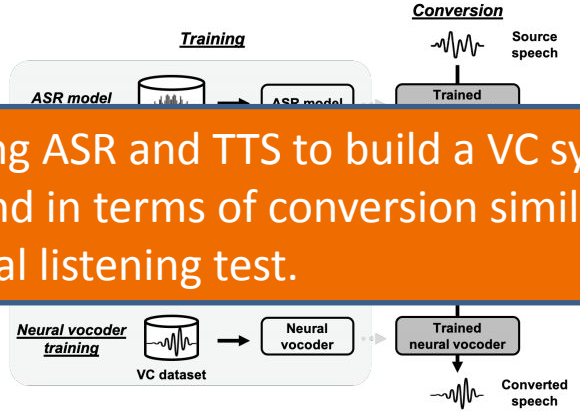
Wen-Chin Huang[1], *Tomoki Hayashi*[1], *Shinji Watanabe*[2], *Tomoki Toda*[1]

[1]Nagoya University, Japan
[2]Johns Hopkins University, USA

wen.chinhuang@g.sp.m.is.nagoya-u.ac.jp

Abstract

This paper presents the sequence-to-sequence (seq2seq) baseline system for the voice conversion challenge (VCC) 2020. We consider a naive approach for voice c… which is to first transcribe the input speech … speech recognition (ASR) model, followed u… tions to generate the voice of the target wit… (TTS) model. We revisit this method und… sequence (seq2seq) framework by utilizing … source end-to-end speech processing toolk… well-configured pretrained models provided by the community. Official evaluation results show that our system comes out top among the participating systems in terms of conversion similarity, demonstrating the promising ability of seq2seq models to convert speaker identity. The implementation is



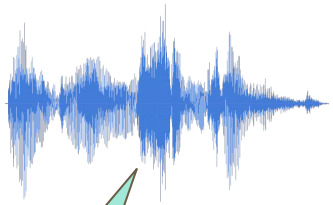
- Combining ASR and TTS to build a VC system
- Placed 2nd in terms of conversion similarity in the official listening test.

Today's talk

- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Speech to text translation (ST)

Speech Translation



translate

Das ist eine andere Geschichte (Germany)

That's another story **(English)**

ESPnet-ST

ESPnet-ST: All-in-One Speech Translation Toolkit

Hirofumi Inaguma[1] **Shun Kiyono**[2] **Kevin Duh**[3] **Shigeki Karita**[4]
Nelson Yalta[5] **Tomoki Hayashi**[6,7] **Shinji Watanabe**[3]
[1] Kyoto University [2] RIKEN AIP [3] Johns Hopkins University
[4] NTT Communication Science Laboratories [5] Waseda University
[6] Nagoya University [7] Human Dataware Lab. Co., Ltd.
inaguma@sap.ist.i.kyoto-u.ac.jp

Abstract

We present *ESPnet-ST*, which is designed for the quick development of speech-to-speech translation systems in a single framework. *ESPnet-ST* is a new project inside end-to-end speech processing toolkit, ESPnet, which

can reduce latency at inference time, which is useful for time-critical use cases like simultaneous interpretation. (2) A single model enables back-propagation training in an end-to-end fashion, which mitigates the risk of error propagation by cascaded modules. (3) In certain use cases

ESPnet-ST

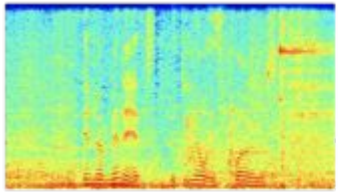
- Support the speech translation (ST) task with both the traditional **pipeline** approach (ASR + NMT) and **end-to-end (E2E)** approach
  - ESPnet also supports NMT and comparable performance to the other toolkit
- We demonstrated the **state-of-the-art translation performance** in standard ST benchmarks
  - MUST-C, IWSLT, Fisher Callhome Spanish
- Again, new progresses in ASR can be easily transferred to ST performance improvement, e.g., conformer, non-AR modeling

Today's talk

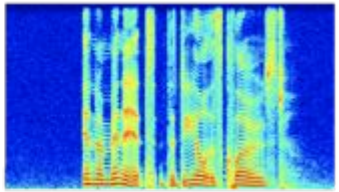
- Introduction of ESPnet, end-to-end speech processing toolkit
- Broadened applications
- Automatic speech recognition (ASR)
  - Performance improvement
  - New features
    - RNN-transducer
    - Non-autoregressive modeling
- Text to speech (TTS)
- Voice conversion
- Speech translation
- Speech enhancement

Speech enhancement (SE)

Speech Enhancement



denoise
dereverberate
separate



ESPnet-SE

ESPNET-SE: END-TO-END SPEECH ENHANCEMENT AND SEPARATION TOOLKIT DESIGNED FOR ASR INTEGRATION

Chenda Li[1*], *Jing Shi*[2,3*], *Wangyou Zhang*[1*], *Aswin Shanmugam Subramanian*[3], *Xuankai Chang*[3], *Naoyuki Kamo, Moto Hira*[4], *Tomoki Hayashi*[5,6], *Christoph Boeddeker*[7], *Zhuo Chen*[8], *Shinji Watanabe*[3]
[1]Shanghai Jiao Tong University, [2]Institute of Automation, Chinese Academy of Sciences,
[3]Johns Hopkins University, [4]Facebook AI, [5]Nagoya University,
[6]Human Dataware Lab. Co., Ltd., [7]Paderborn University, [8]Microsoft Research

ABSTRACT

We present ESPnet-SE, which is designed for the quick development of speech enhancement and speech separation systems in a single framework, along with the optional downstream speech recognition module. ESPnet-SE is a new project which integrates rich automatic speech recognition related models, resources and systems to support and validate the proposed front-end implementation (i.e. speech enhancement and separation). It is capable of processing both single-channel and multi-channel data, with various functionalities including dereverberation, denoising and source separation. We provide all-in-one recipes including data pre-processing, feature extraction, training and evaluation pipelines for a wide range of benchmark

In this paper, we introduce a new E2E-SE toolkit named ESPnet-SE[1], which is an extension of the open-source speech processing toolkit ESPnet [8]. ESPnet-SE fully considers the various forms of speech input in the front-end scenes and meanwhile flexibly and organically integrated with the downstream automatic speech recognition (ASR) task, making it a user-friendly toolkit to easily build totally end-to-end robust ASR systems, even without need for clean speech signals. The toolkit provides adaptability to different speech data, including (1) single and multiple speakers, (2) single and multiple channels, (3) anechoic and reverberant conditions. Moreover, thanks to the ripe and efficient ASR modules in ESPnet, rich speech recognition related models, resources and systems can be optionally concatenated after the E2E-SE system, enabling evaluation and joint

ESPnet-SE

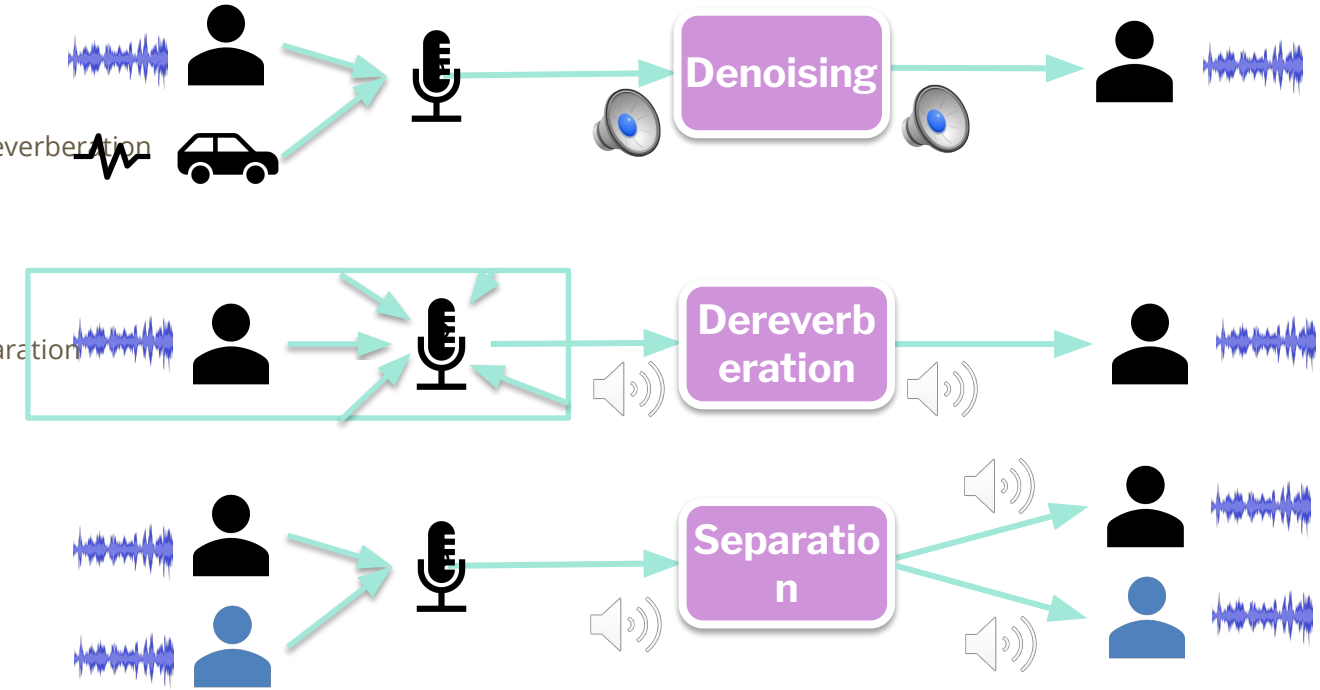
- One of the biggest changes in ESPnet
- Include **ALL** speech enhancement functions
- Demonstration

https://colab.research.google.com/drive/1fjRJCh96SoYLZPRxsjF9VDv4Q2VoIckI?usp=sharing

- Many help from **torchaudio**
- **SpeechBrain** also has the strong SE functions

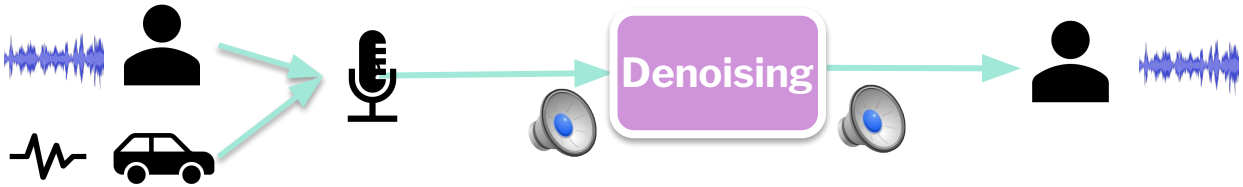
Speech enhancement
Several types of problems

- Denoising (people mainly call it speech enhancement)
- Dereverberation
- Separation



Speech enhancement
Several types of problems

- Denoising (people mainly call it speech enhancement)



Enhance the single-channel real noisy speech in CHiME4

```
# play the enhanced single-channel speech
wave = enh_model_sc(mixwav_sc[None, ...], sr)
print("Input real noisy speech", flush=True)
display(Audio(mixwav_sc, rate=sr))
print("Enhanced speech", flush=True)
display(Audio(wave[0].squeeze(), rate=sr))
```

```
Input real noisy speech
```

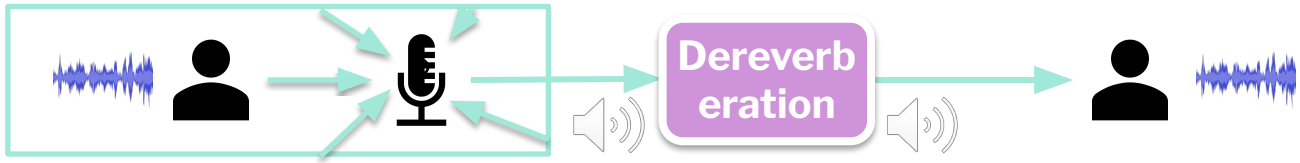
0:00 / 0:05

```
Enhanced speech
```

0:00 / 0:05

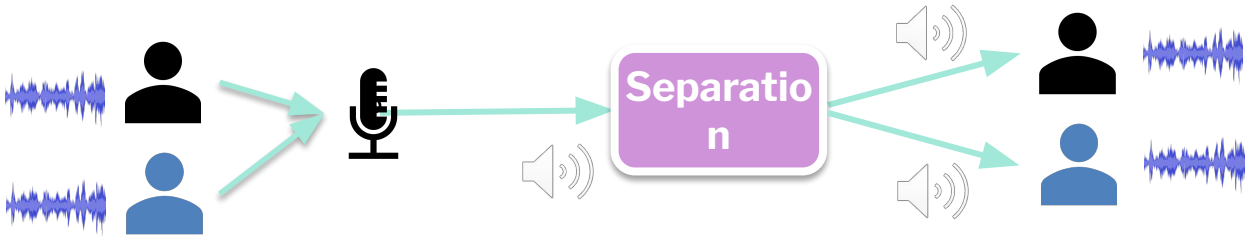
Speech enhancement
Several types of problems

- Dereverberation



Speech enhancement
Several types of problems

- Separation



Separate the example in wsj0_2mix testing set

```
!gdown --id 1ZCUkd_Lb7pO2rpPr4FqYdtJBZ7JMiInx -O /content/447c020t_1.2106_422a0112_-1.2106.wav

import os
import soundfile
from IPython.display import display, Audio

mixwav, sr = soundfile.read("447c020t_1.2106_422a0112_-1.2106.wav")
waves_wsj = separate_speech(mixwav[None, ...], fs=sr)

print("Input mixture", flush=True)
display(Audio(mixwav, rate=sr))
print(f"========= Separated speech with model {tag} =========", flush=True)
print("Separated spk1", flush=True)
display(Audio(waves_wsj[0].squeeze(), rate=sr))
print("Separated spk2", flush=True)
display(Audio(waves_wsj[1].squeeze(), rate=sr))
```

```
Downloading...
From: https://drive.google.com/uc?id=1ZCUkd_Lb7pO2rpPr4FqYdtJBZ7JMiInx
To: /content/447c020t_1.2106_422a0112_-1.2106.wav
100% 184k/184k [00:00<00:00, 5.73MB/s]
100%|██████████| 13/13 [00:00<00:00, 28.27it/s]Input mixture
```

0:00 / 0:11

```
========= Separated speech with model Chenda Li/wsj0_2mix_enh_train_enh_conv_tasnet_raw_valid.si_snr.
Separated spk1
```

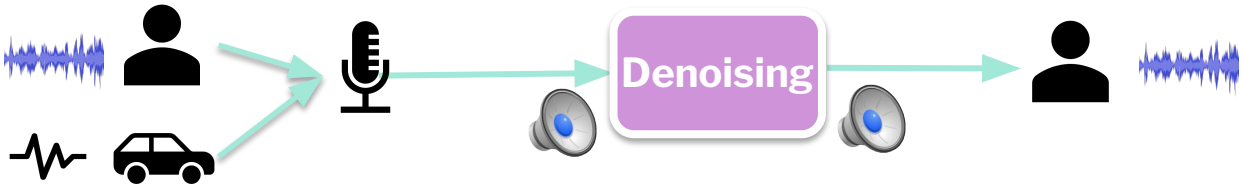
0:00 / 0:11

```
Separated spk2
```

0:00 / 0:11

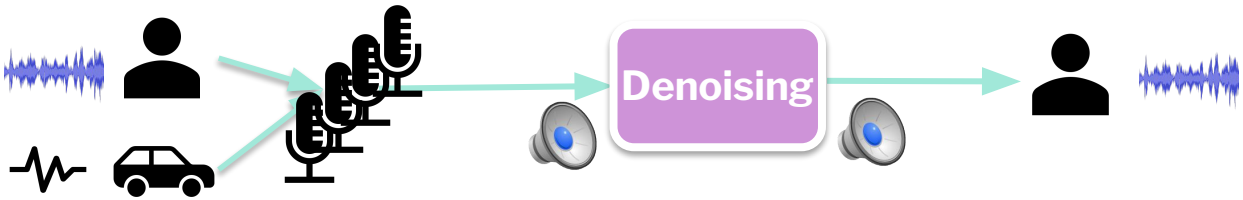
Microphone array processing
Single to multiple microphones

- Denoising (people mainly call it speech enhancement)



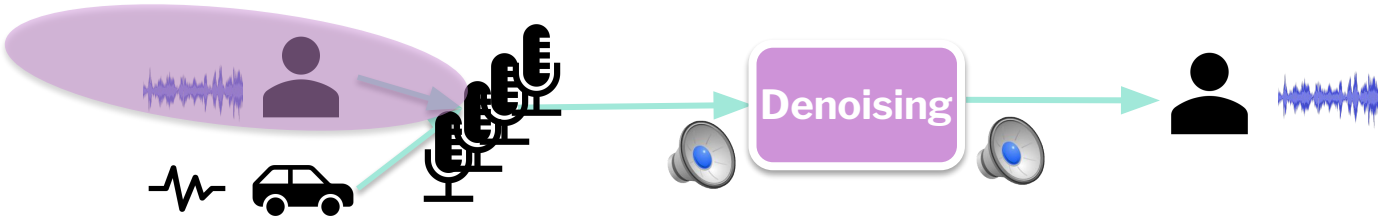
Microphone array processing
Single to multiple microphones

- Denoising (people mainly call it speech enhancement)



Microphone array processing
Single to multiple microphones

- Denoising (people mainly call it speech enhancement)

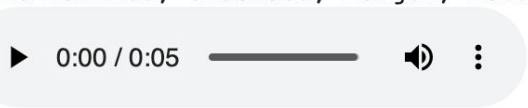


Make a spatial **beam** (beamforming)
to only pick up desired signals

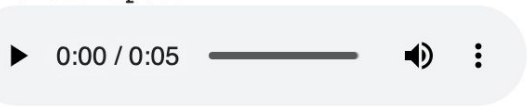
Enhance the multi-channel real noisy speech in CHiME4

```
wave = enh_model_mc(mixwav_mc[None, ...], sr)
print("Input real noisy speech", flush=True)
display(Audio(mixwav_mc.T, rate=sr))
print("Enhanced speech", flush=True)
display(Audio(wave[0].squeeze(), rate=sr))
```

```
/usr/local/lib/python3.7/dist-packages/torch/functi
  normalized, onesided, return_complex)
Input real noisy speech
/usr/local/lib/python3.7/dist-packages/torch/functi
  normalized, onesided, length, return_complex)
```

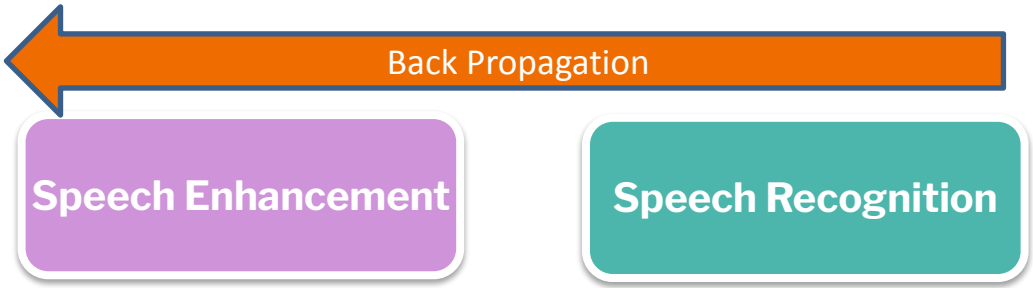


```
Enhanced speech
```



Differentiable speech enhancement frontend

- ESPnet SE can be used as an **independent** enhancement module
  - Denoising, Dereveberation, Separation
  - Single channel or multichannel
  - Can port **Asteroid** (audio source separation toolkit) pre-trained models
- ESPnet SE can be used as a **differentiable** enhancement module

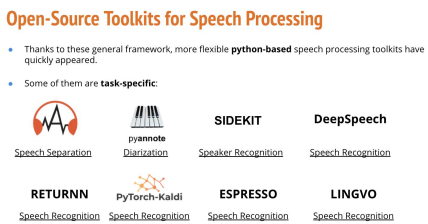


- We can realize a cocktail party effect by a machine

Summary

ESPnet

- End-to-end speech processing has a lot of potentials
- ESPnet provides **state-of-the-art** and **reproducible** research in various speech processing applications
- We are working closely with other benchmarks (SUPERB, LeBenchmark, CHiME, VCC, etc.) and toolkit



We need further interactions with machine leaning community!
Let's work together for the community contribution!!

Discussions
(these are commonly shared with SpeechBrain project)

- **Complementary task covering with multiple toolkits**
  - ESPnet
    - More focus on ASR/TTS/Speech translation with SOTA
  - SpeechBrain
    - More applications, and SOTA for ASR, speaker identification, spoken language understanding, and speech separation
  - Other tools also have unique functions ;)
- **ESPnet and SpeechBrain are academic driven**
  - Less biased for techniques
  - Any collaborations based on academic freedom!
  - Weak for the production support in general e.g., streaming and intensive tuning
  - Difficulties on maintainability and scalability
- **Speech is one of the most important machine learning applications**
  - How should these toolkits be evolved for machine learning researchers?
- We also want to discuss these items during the following sessions!!!

Concluding Remarks

Concluding Remarks

- Self-supervised learning for speech
- Open-source toolkits for wide variety of tasks

Start to engage in developing Speech Technology