S3PRL introduction & recent update in JSALT

Shu-wen (Leo) Yang
Andy T. Liu, Heng-Jui Chang, Haibin Wu, Cheng Liang

S3PRL

Self-**S**upervised **S**peech **P**re-training
and **R**epresentation **L**earning

https://github.com/s3prl/s3prl/

s3prl

s3prl

Self-Supervised Speech Pre-training and Representation Learning Toolkit.

youtu.be/PkMFnS6cjAc

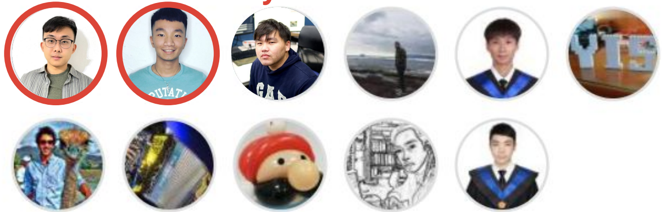
1.4k stars 315 forks

https://github.com/s3prl/s3prl/

Used by 14

+ 6

Contributors 38



+ 27 contributors

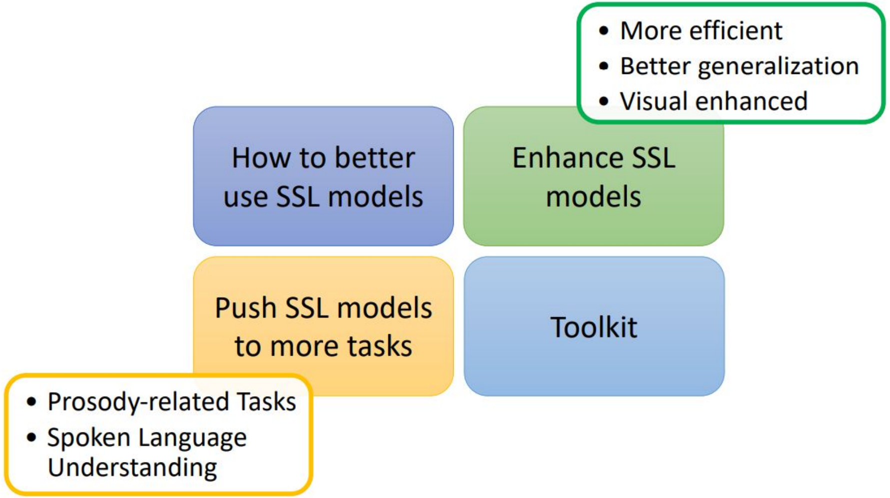
Prof. Hung-yi Lee, Advisor & Sponsor

Major functionality

No version	v0.2	v0.3	
Pre-training	Pre-trained model collection	Downstream Benchmarking & SUPERB	
2019	**2020**	**2021**	
Andy T. Liu	Shu-wen Yang	Shu-wen Yang	Heng-Jui Chang
Shu-wen Yang	Andy T. Liu	Andy T. Liu	Xuankai Chang
Po-Han Chi		Po-Han Chi	Yung-Sung Chuang
			Zili Huang
			Wen-Chin Huang
			Tzu-Hsien Huang
			Kushal Lakhotia
			Yist Lin Y.
			Guan-Ting Lin
			Jiatong Shi
			Hsiang-Sheng Tsai
			Wei-Cheng Tseng

Feedback ←→ Improvement during JSALT workshop

- It is intensively used in the JSALT pre-training team for evaluating new techniques
  - Bugs reported
  - Thank all the users for reporting the error
  - Thank JSALT for providing the platform to have so many users to help the open-source



Feedback ←→ Improvement during JSALT workshop

- We receive lots of feedback and continuously improve it:
  - How to change the corpus for XXX task?
  - How to change the probing model for the XXX task?
  - The steps to benchmark a new SSL model is too complicated
  - Connection to the HuggingFace models
  - How to benchmark with just a subset of the corpus?
  - The latest SSL models?

S3PRL was not designed as a flexible/reusable library
but as the recipes to reproduce papers

Feedback ⟷ Improvement during JSALT workshop

- The mostly asked issue is…

Fairseq installation issues

  - At some commits, some pre-trained models work, but the others fail
  - At some commits, all the pre-trained models work, but the new models can't be supported
  - At some commits, you can't even `torch.load` the some checkpoints, since the checkpoints contain deprecated pickled Fairseq object

Too Difficult…

Feedback ⟵⟶ Improvement during JSALT workshop

- Fixing these issues → saving lots of users' time
- So the users can use S3PRL under more settings without tracing the code

s3prl

s3prl

Self-Supervised Speech Pre-training and Representation Learning Toolkit.

youtu.be/PkMFnS6cjAc

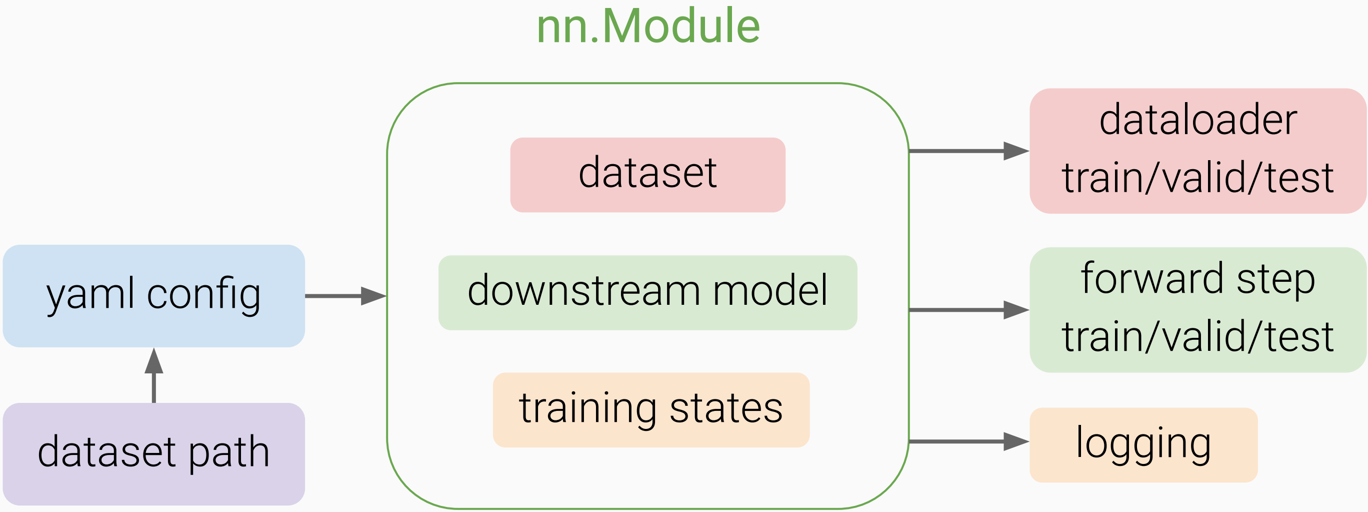
1.4k stars 315 forks

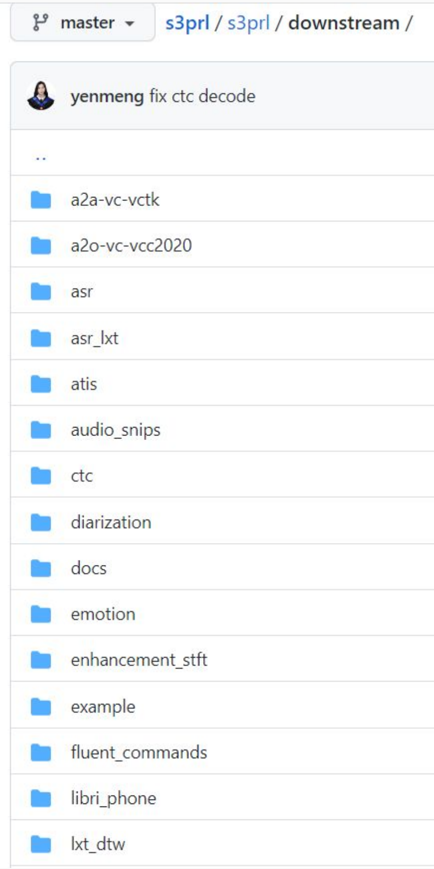
Major Updates for v0.4

- Deprecate tasks' God Classes
- Hooks for changing corpus, downstream, and upstream
- More upstream models
- Remove Fairseq dependency
- HugggingFace connection
- Audio / Sound connection

The major cause of most of the issues

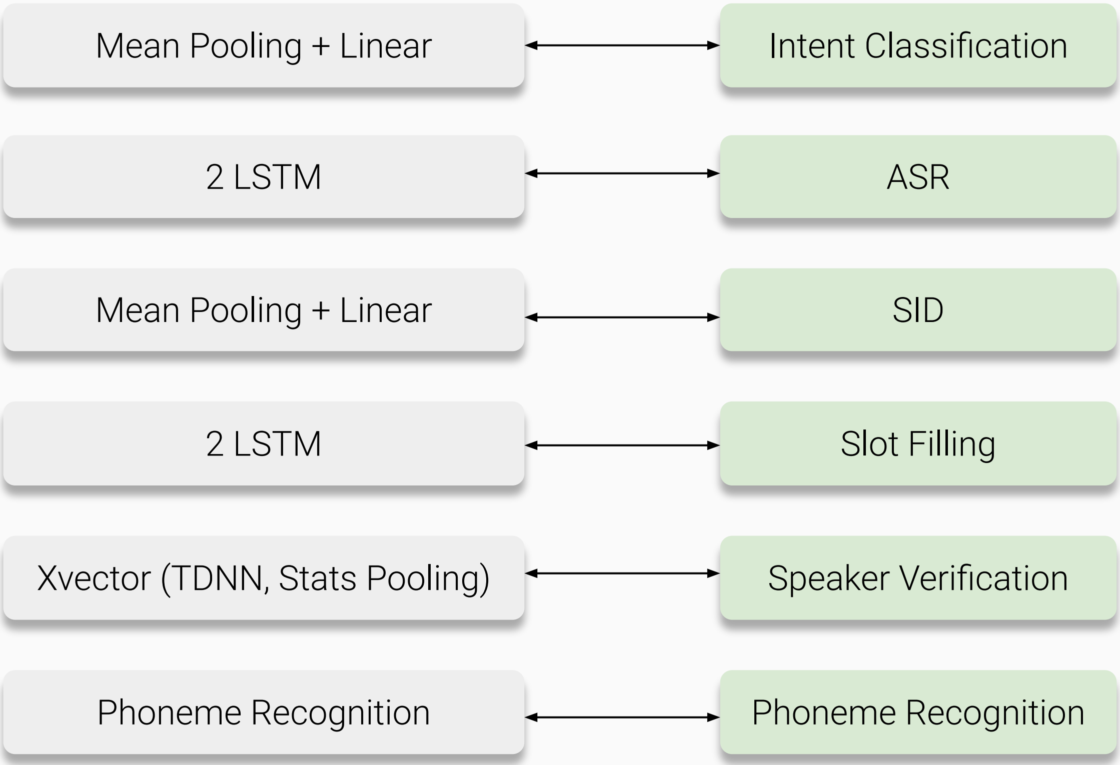
- A single God Class handling all the details for a task





In the God Class - Model entanglement

A specific model

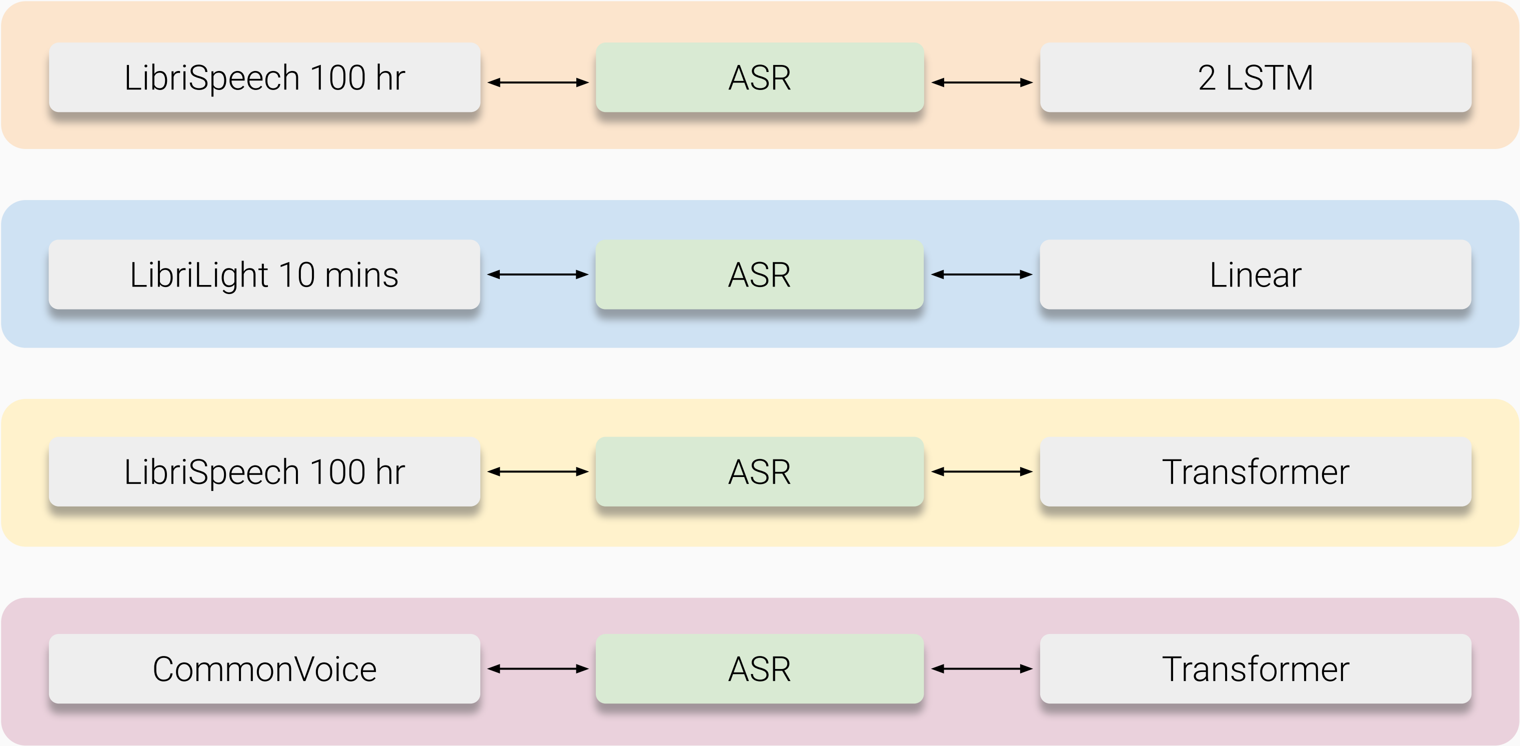


In the God Class - Corpus entanglement

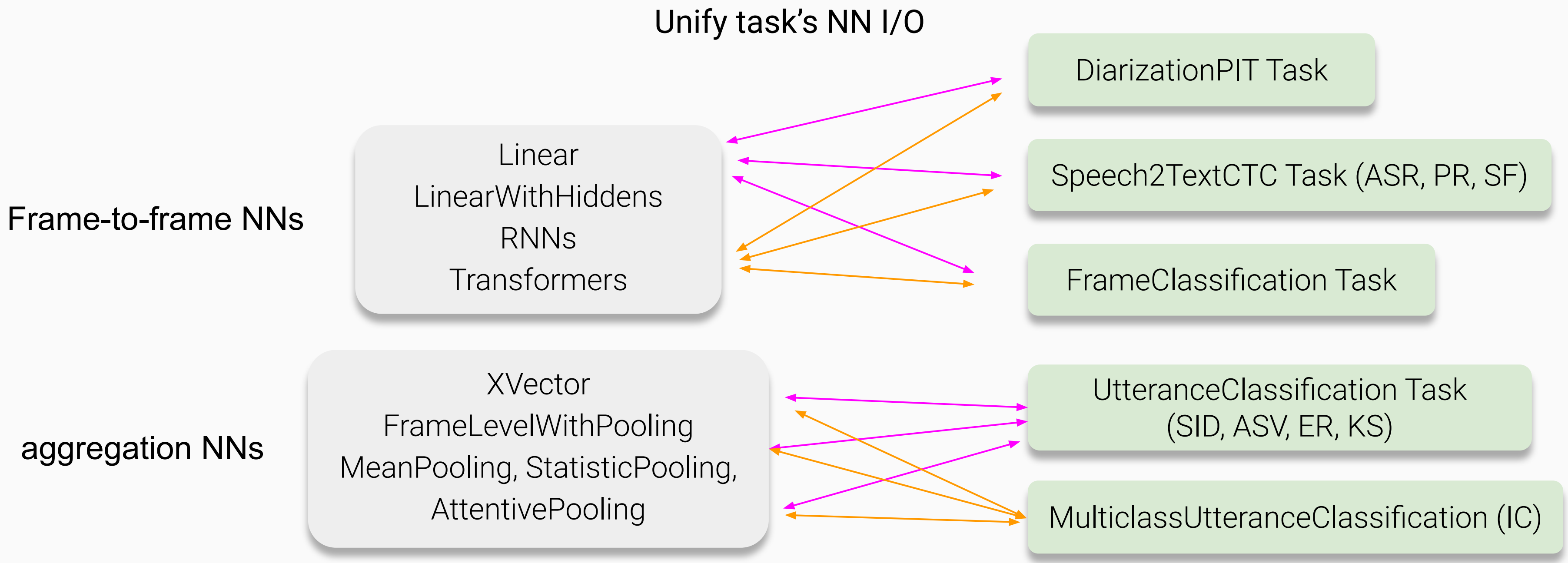
A specific corpus

Fluent Speech Command	Intent Classification
LibriSpeech	ASR
VoxCeleb1	SID
SNIPS	Slot Filling
VoxCeleb1	Speaker Verification
LibriSpeech	Phoneme Recognition

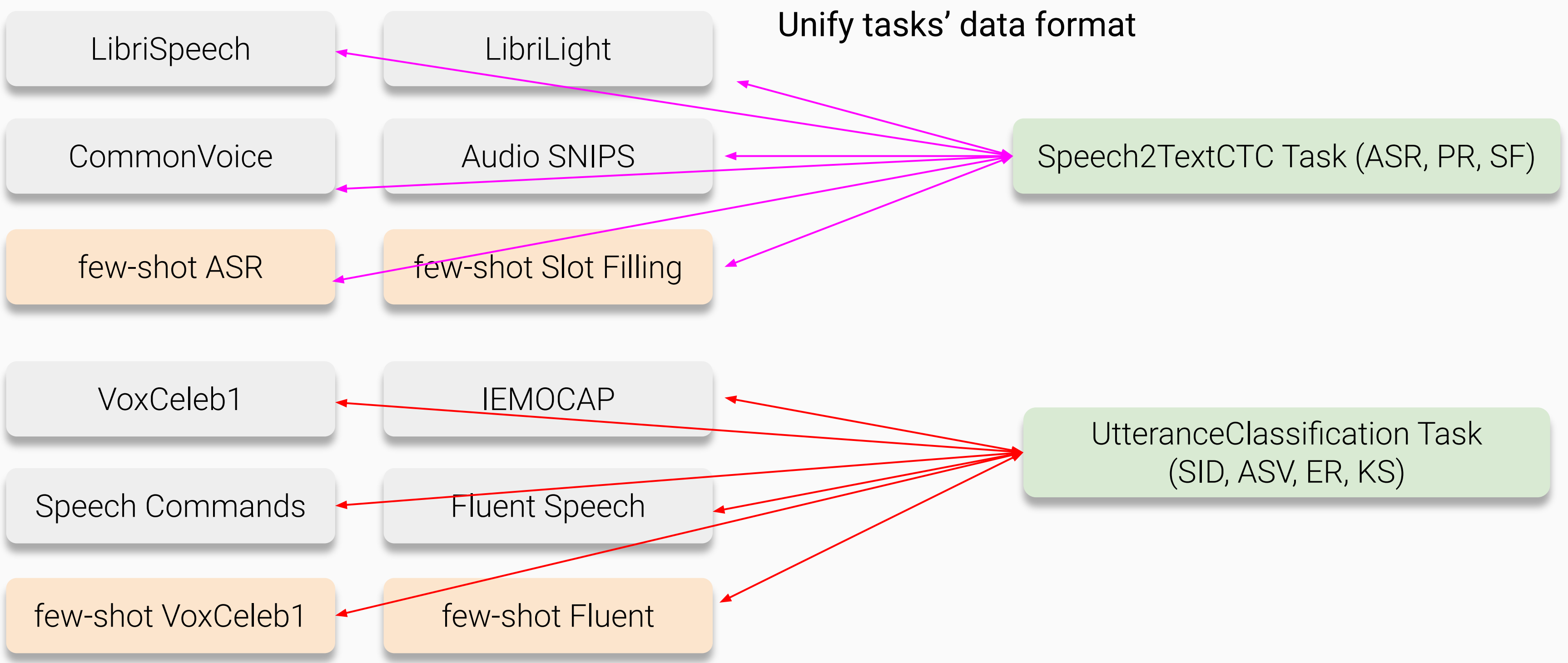
Every slightly change requires library code change



Disentangle model from the Task (LightningModule)



Disentangle corpus from the task



Hooks to customize the behavior for each task

corpus hook	Change the data
upstream hook	Change the feature model (SSL)
downstream hook	Change the task model
train/valid/test dataset hook	Change the __getitem__ (add noise, reverb… etc)
train/valid/test sampler hook	Change the batching behavior

Default usage

- Default hooks reproduce the exact SUPERB setting

```
classmethod run(**cfg)
```

- *Necessary Config:*

```yaml
workspace: ???    # (str) The workspace shared across stages
setup:
  corpus:
    dataset_root: ???    # (str) The root path of the corpus
  upstream:
    name: ???
```

```python
SuperbASR.run(
    workspace="result/pseudo_asr",
    setup=dict(
        corpus=newdict(
            dataset_root="/home/leo/d/datasets/LibriSpeech",
        ),
        upstream=dict(
            name="hubert",
        ),
    ),
    train=dict(
        optimizer=dict(
            lr=1.0e-2,
        )
    )
)
```

```yaml
train:
  n_jobs: 4    # (int) The number of jobs when multiprocessing on CPU
  seed: 1337    # (int) The seed
  device: cuda:0    # (str) The device used for training
  rank: 0    # (int) The global rank when distributed training
  world_size: 1    # (int) The total number of processes when distributed training
  optimizer:
    CLS: torch.optim.adam.Adam    # (str) The class used to create the optimizer. The below
    lr: 0.0001
```

Hooks to customize data

```yaml
setup:
  corpus:
    CLS: librispeech_for_speech2text    # (str)
  #  The corpus class. You can add the **kwargs right below this CLS key
    dataset_root: ???    # (str) The root path of the corpus
```

```python
from pathlib import Path
from s3prl import newdict
from s3prl.problem import SuperbASR
from s3prl.util.pseudo_data import pseudo_audio

N_SAMPLES, N_TRAIN, N_VALID, N_TEST = 40, 20, 10, 10

def prepare_pseudo_data(wav_paths):
    train_paths = wav_paths[:N_TRAIN]
    valid_paths = wav_paths[N_TRAIN : N_TRAIN + N_VALID]
    test_paths = wav_paths[N_TRAIN + N_VALID :]

    def path_to_datapoint(path):
        return {
            "wav_path": path,
            "transcription": "Hello World",
        }

    train_data = {Path(path).stem: path_to_datapoint(path) for path in train_paths}
    valid_data = {Path(path).stem: path_to_datapoint(path) for path in valid_paths}
    test_data = {Path(path).stem: path_to_datapoint(path) for path in test_paths}

    return {
        "train_data": train_data,
        "valid_data": valid_data,
        "test_data": test_data,
    }
```

```python
with pseudo_audio(secs=range(1, N_SAMPLES + 1), sample_rate=16000) as (
    wav_paths,
    num_samples,
):

    SuperbASR.run(
        workspace="result/pseudo_asr",
        setup=dict(
            corpus=newdict(
                CLS=prepare_pseudo_data,  # hook for changing data
                wav_paths=wav_paths,  # arguments to the hook
            ),
            upstream=dict(name="hubert"),
        ),
    )
```

Note: It is also easy to load Kaldi based data directory by using a directory parser hook

Hooks to customize downstream

```python
import torch.nn as nn
from s3prl import newdict
from s3prl.problem import SuperbASR

class CustomizedModel(nn.Module):
    def __init__(self, input_size, output_size, hidden_size: int) -> None:
        super().__init__()
        self.model = nn.Sequential(
            nn.Linear(input_size, hidden_size),
            nn.Linear(hidden_size, output_size),
        )

    def forward(self, x, x_len):
        x = self.model(x)
        return x, x_len

SuperbASR.run(
    workspace="result/pseudo_asr",
    setup=dict(
        corpus=dict(
            dataset_root="/home/leo/d/datasets/LibriSpeech",
        ),
        upstream=dict(
            name="hubert",
        ),
        downstream=newdict(
            CLS=CustomizedModel,
            hidden_size=256,
        ),
    ),
)
```

Hooks to customize upstream

```python
import torch
import torch.nn as nn
from s3prl import newdict
from s3prl.problem import SuperbASR

class CustomizedUpstream(nn.Module):
    def __init__(self, ckpt_path: str) -> None:
        super().__init__()
        ckpt = torch.load(ckpt_path, map_location="cpu")
        hidden_size = ckpt["config"]["hidden_size"]
        self.model = nn.Sequential(
            nn.Linear(1, hidden_size),
            nn.Linear(hidden_size, hidden_size),
        )
        self.model.load_state_dict(ckpt["model_weights"])

    def forward(self, x, x_len):
        x = self.model(x)
        return x, x_len

SuperbASR.run(
    workspace="result/pseudo_asr",
    setup=dict(
        corpus=dict(
            dataset_root="/home/leo/d/datasets/LibriSpeech",
        ),
        upstream=newdict(
            CLS=CustomizedUpstream,
            ckpt_path="./ckpts/ssl_val_best.ckpt",
        ),
    ),
)
```

More upstream models

In SUPERB

Mockingjay | TERA | HuBERT | APC | VQ-APC | NPC | DeCoAR

DeCoAR 2.0 | Modified CPC | wav2vec | vq-wav2vec | wav2vec 2.0 | PASE+

New models

discreteBERT | HuBERT-MGR | LightHuBERT | FitHuBERT

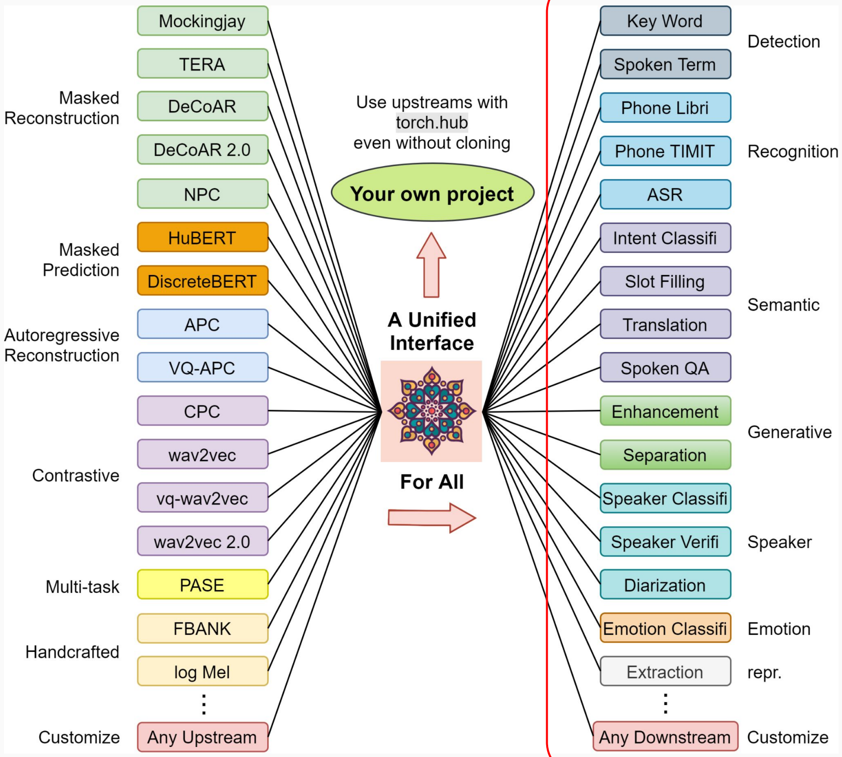
Unispeech-SAT | WavLM | data2vec | AudioAlbert | DistilHuBERT

S3PRL SPEECH TOOLKIT

ESPnet

Upstream SSL models (Frontend)

Downstream Tasks



Task	Corpus
ESPnet-ASR	WSJ, Switchboard, CHiME-4/5, Librispeech, TED, CSJ, AMI, HKUST, Voxforge
ESPnet-SLU intent / slot filling	SLURP, Fluent Speech Commands, Audio SNIPS, HarperValleyBank, Grabo, IEMOCAP
ESPnet-ST	Fisher-CallHome Spanish, Libri-trans, IWSLT'18, How2, Must-C, Mboshi-French

To give a more stable support for users and ESPNet

- **Remove all the fairseq dependencies**
  - All the upstream can be used without installing fairseq
- **Add unit-tests for the forward and backward for all upstreams**
  - Test the representation and gradient’s numerical values
  - Guarantee the same representation across S3PRL versions

Reproduced results

- A complete re-build to
  - Get away all the old dirty design at once
  - Ensure we always have the exact old codebase for SUPERB for reproducibllity
  - Results on Hubert Base

Task	PR	IC	SID	KS	ER	ASR	QBE	SD	SF	SV
Metric	PER	ACC	ACC	ACC	ACC	WER	MTWV	DER	CER	EER
Old	**5.41**	**98.34**	**81.19**	**96.3**	**64.92**	**6.11**	**7.37**	**5.88**	**25.2**	**5.11**
New	**5.483**	**98.207**	**80.69**	**96.62**	**64.76**	**6.14**	**7.37**	**5.8**	**24.22**	**5.15**
Relative	**-1%**	**-0.1%**	**-0.6%**	**0.3%**	**-0.2%**	**-0.4%**	**0%**	**1.3%**	**3%**	**-0.7%**

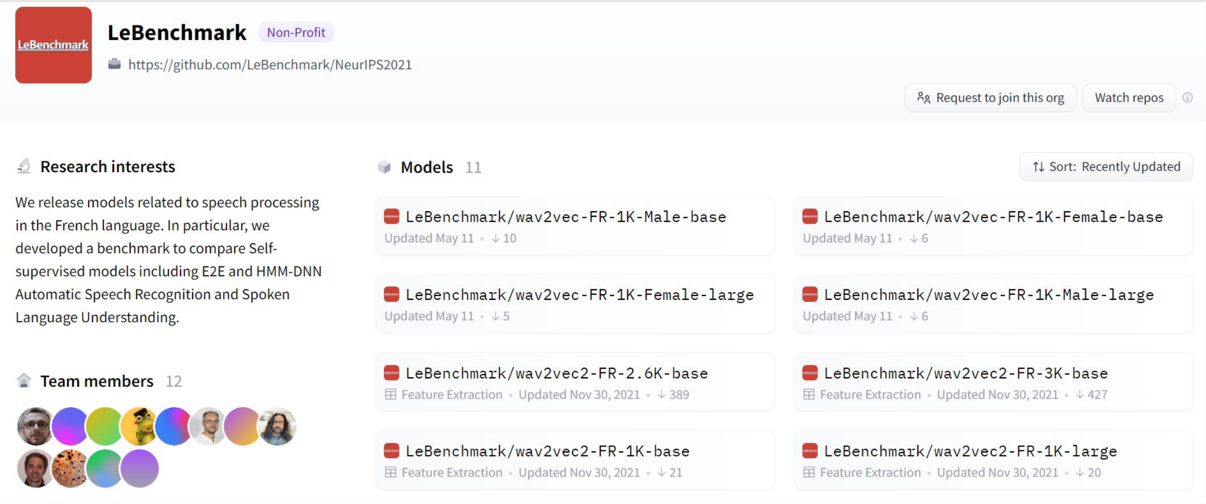
For the last week in JSALT

- Connection to HuggingFace
- Connection to the HEAR Benchmark

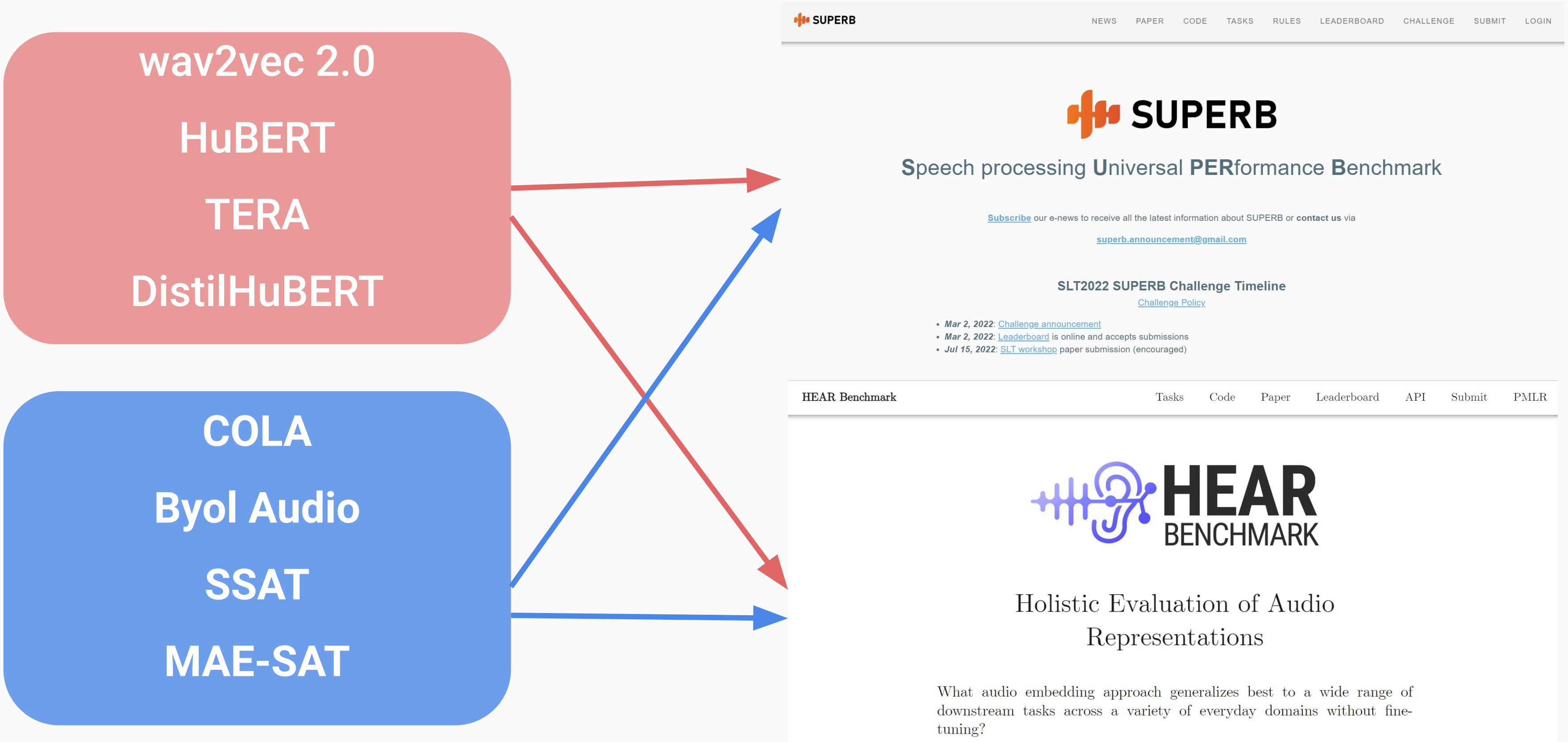
Thanks the users in JSALT for the great suggestion

Connection to HuggingFace

- There are fewer model architectures available in Huggingface
  - wav2vec 2.0, HuBERT, data2vec, UniSpeech, WavLM
- But a lot more pre-trained checkpoints available, e.g. LeBenchmark



Connection to Audio SSL



Connection to HEAR Benchmark

- Transferbility of the speech SSL SOTA to the more sounds
- Exisiting HEAR codebase: official, faster
  - Dump a single layer frozen representation
  - Train the downstream model
- S3PRL, on-the-fly feature extraction: slower, more flexible
  - Enable examination for lots of speech SSL models
  - Enable weighted-sum over all layers
  - Enable finetuning SSL models on audio tasks

Connection to HEAR Benchmark

HEAR Benchmark Tasks Code Paper Leaderboard API Submit PMLR

Task Name	Embed Type	Predictor Type	Split Method	Duration (sec)	# clips	Evaluation Metric	Novel
DCASE 2016 Task 2	T	L	TVT	120.0	72	Onset FMS	✓
NSynth Pitch 5hr	S	C	TVT	4.0	5000	Pitch Acc.	✓
NSynth Pitch 50hr	S	C	TVT	4.0	49060	Pitch Acc.	✓
Speech Commands 5hr	S	C	TVT	1.0	22890	Accuracy	✓
Speech Commands Full	S	C	TVT	1.0	100503	Accuracy	
Beehive States	S	C	TVT	600.0	576	AUCROC	
Beijing Opera Percussion	S	C	5-fold	4.77	236	Accuracy	✓
CREMA-D	S	C	5-fold	5.0	7438	Accuracy	
ESC-50	S	C	5-fold	5.0	2000	Accuracy	
FSD50K	S	L	TVT	0.3-30.0	51185	mAP	
Gunshot Triangulation	S	C	7-fold	1.5	88	Accuracy	✓
GTZAN Genre	S	C	10-fold	30.0	1000	Accuracy	

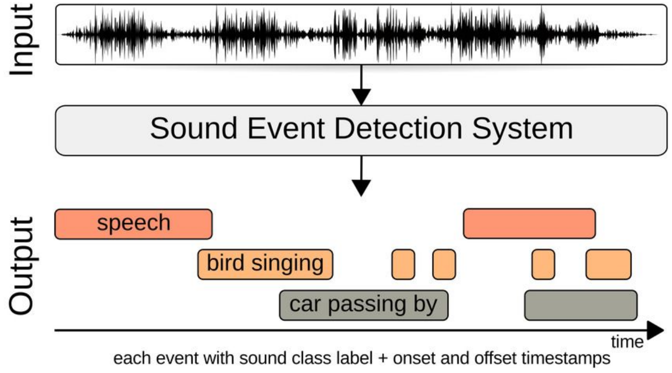
Connection to HEAR Benchmark

- **The best demonstrastion on the benefit of codebase refactoring**
- 11 new audio tasks in HEAR are immediately supported
- Task design: k-fold, accuracy
    - Reuse the template of IEMOCAP emotion classification in SUPERB
- Change the hook configuration:
    - Corpus hook
    - Downstream model hook

Connection to HEAR Benchmark

- 2 new tasks required to be supported in HEAR Benchmark
  - multilabel classifcation (WIP)
  - sound event detection (done)

Rank	Model	Onset FMS
1	PaSST 2lvl+mel	0.9254
2	PaSST 2lvl	0.9132
3	wav2vec2 WS (S3PRL)	0.8641
12	wav2vec2 baseline	0.6630
	wav2vec2 baseline (S3PRL)	0.6624

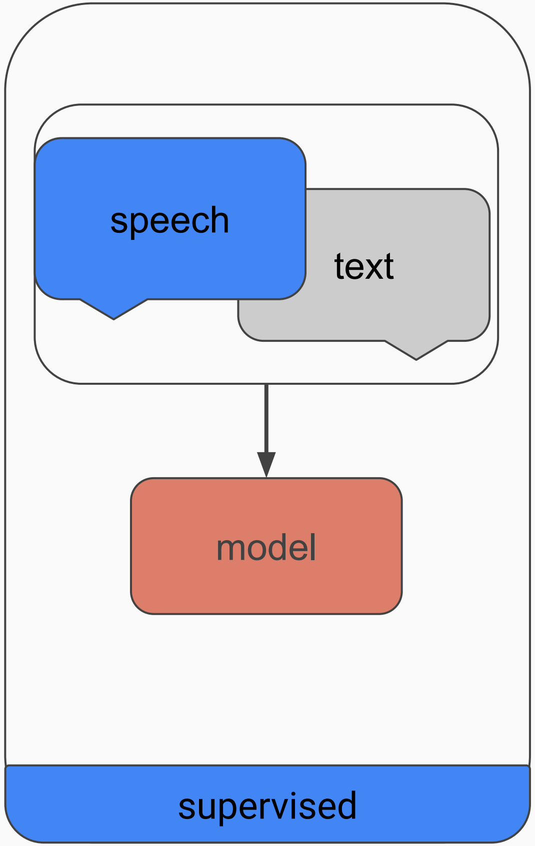


Release v0.4.0 in JSALT

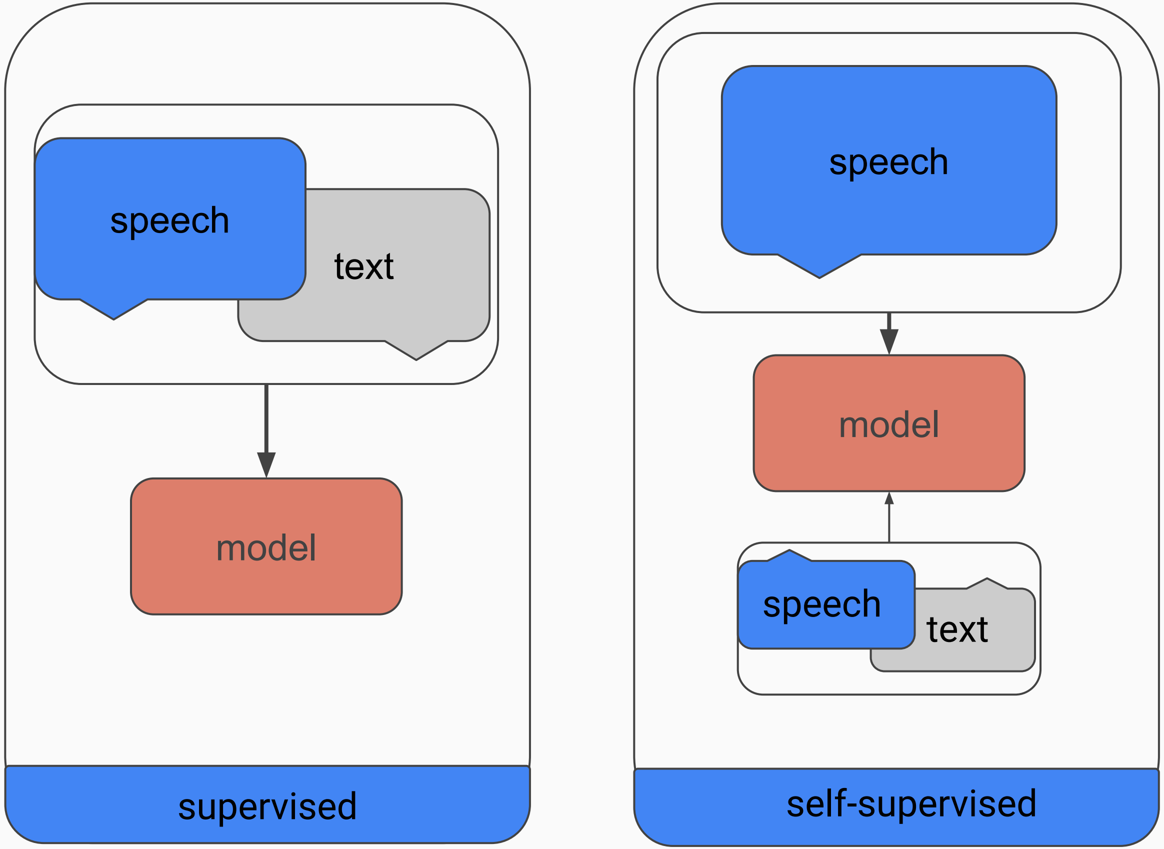
Unsupervised Automatic Speech Recognition

Dongji Gao

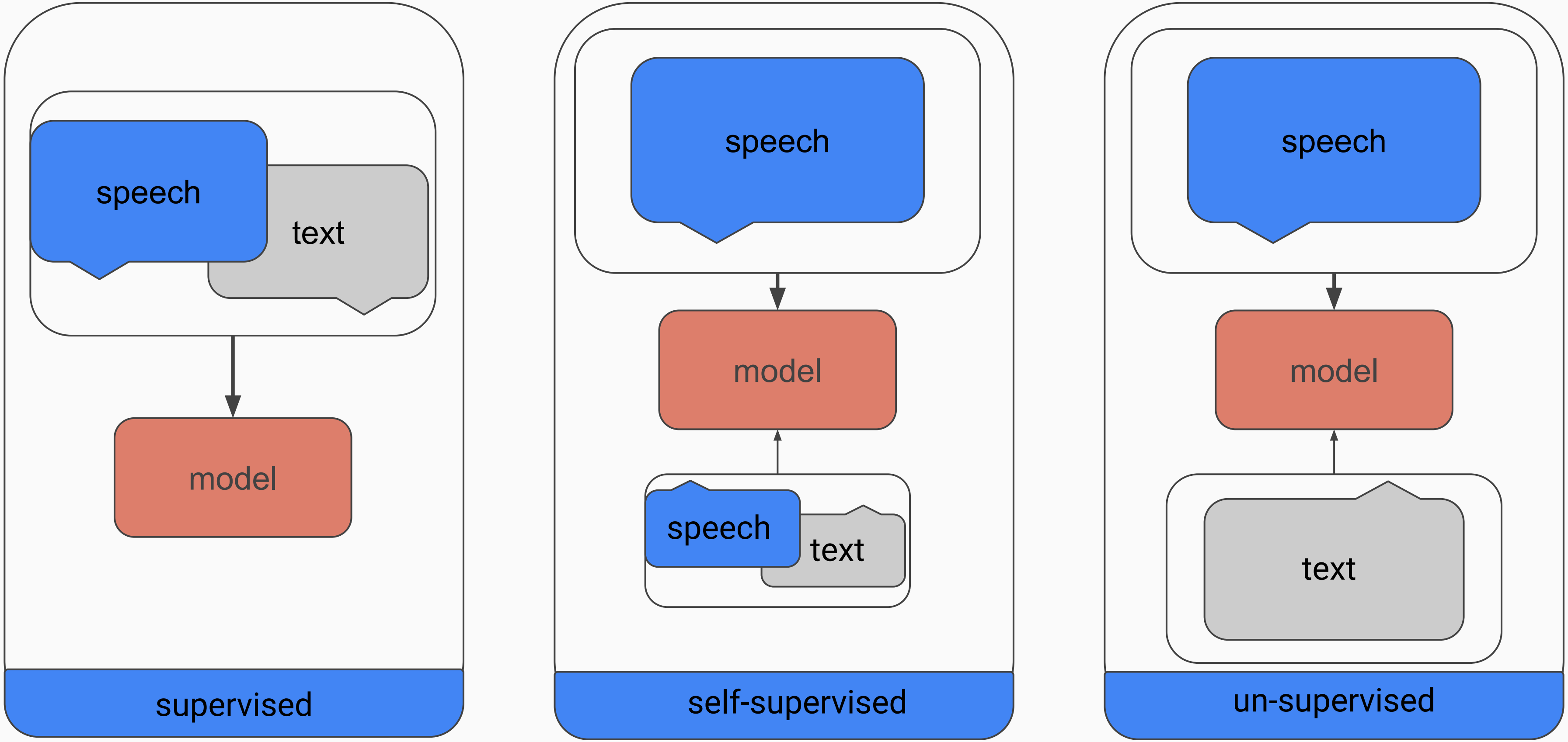
Supervised -> self-supervised -> unsupervised



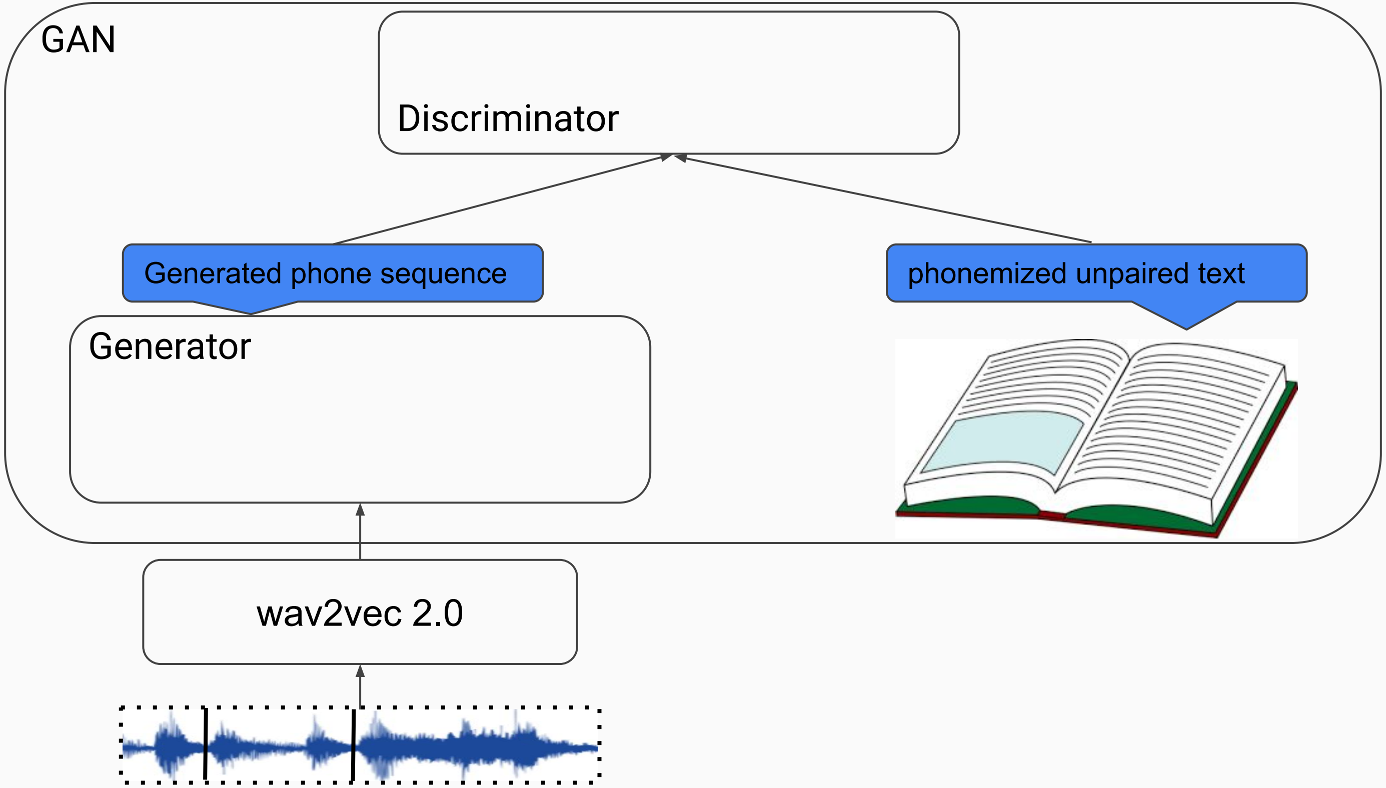
Supervised -> self-supervised -> unsupervised



Supervised -> self-supervised -> unsupervised

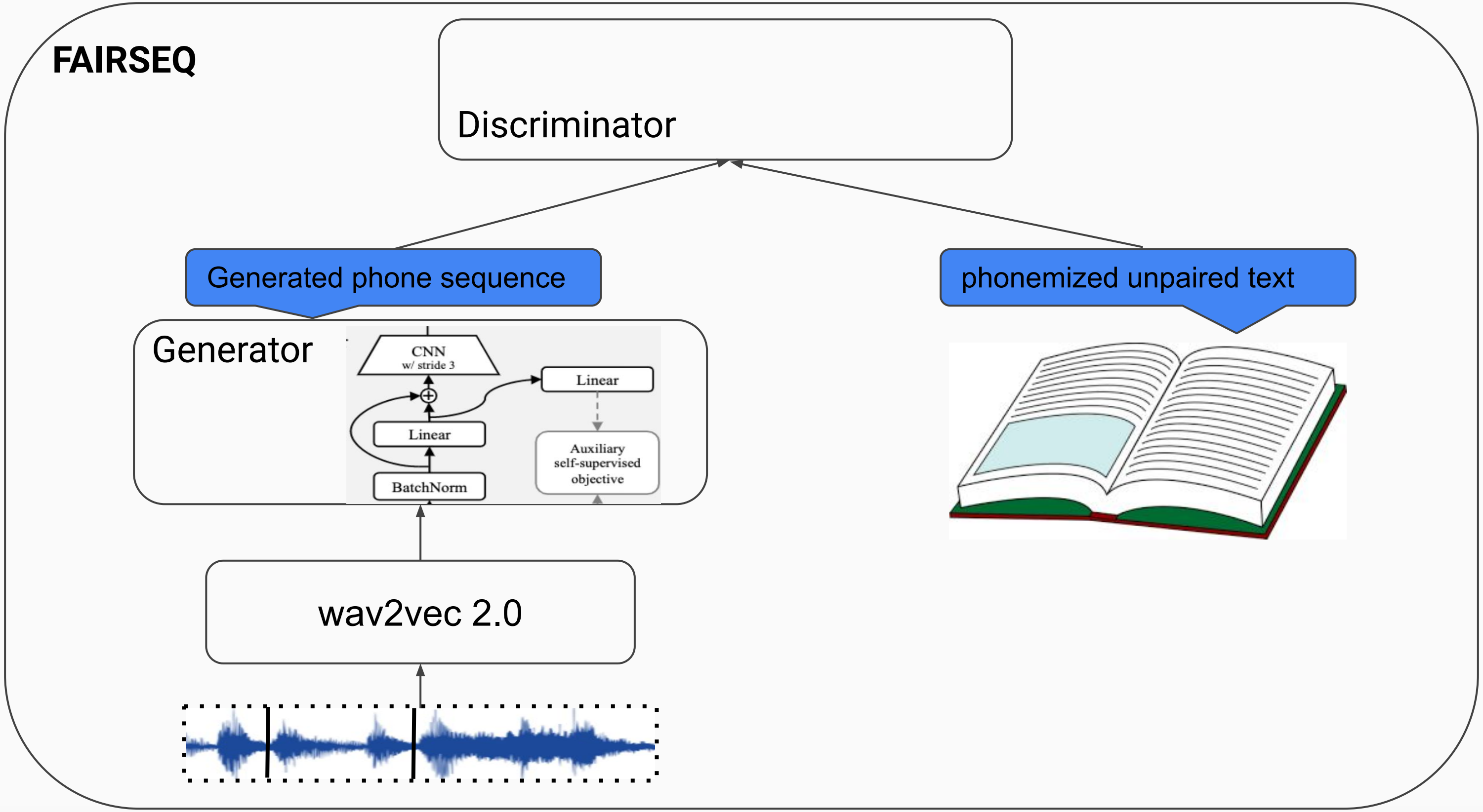


wav2vec–u2



Liu, Alexander H., Wei-Ning Hsu, Michael Auli, and Alexei Baevski. "Towards End-to-end Unsupervised Speech Recognition." (2022).

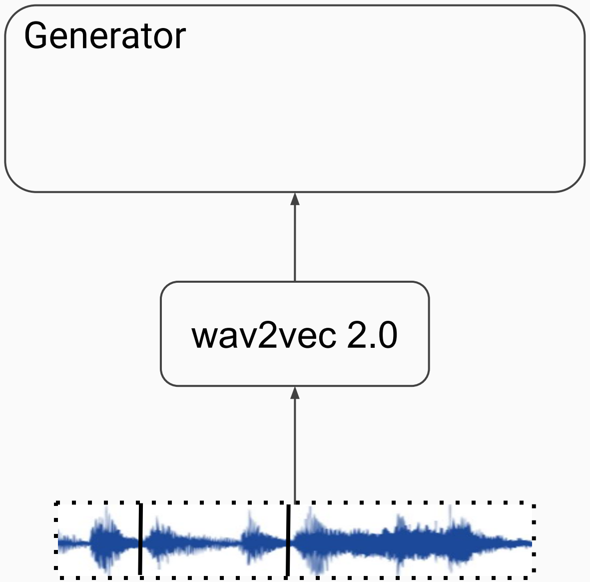
wav2vec–u2



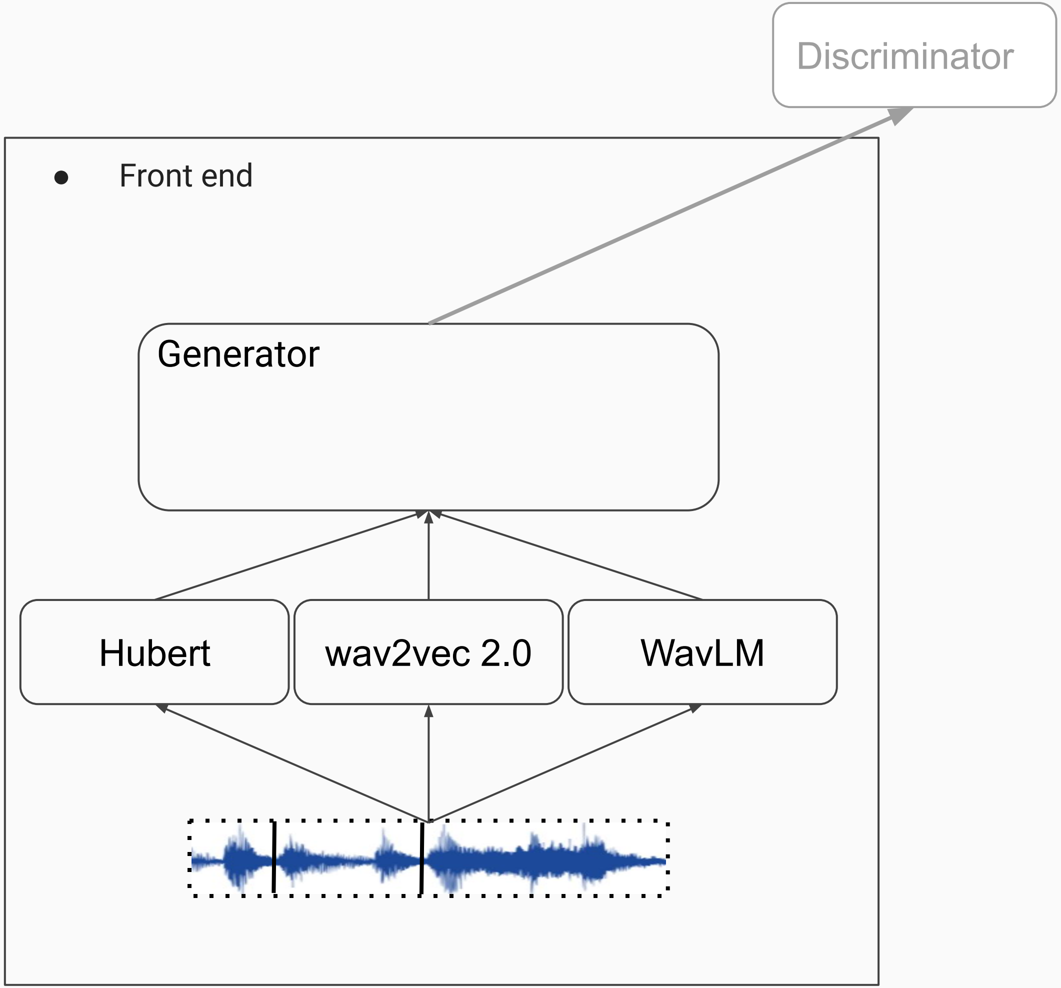
Liu, Alexander H., Wei-Ning Hsu, Michael Auli, and Alexei Baevski. "Towards End-to-end Unsupervised Speech Recognition." (2022).

ESPNET

- Front end

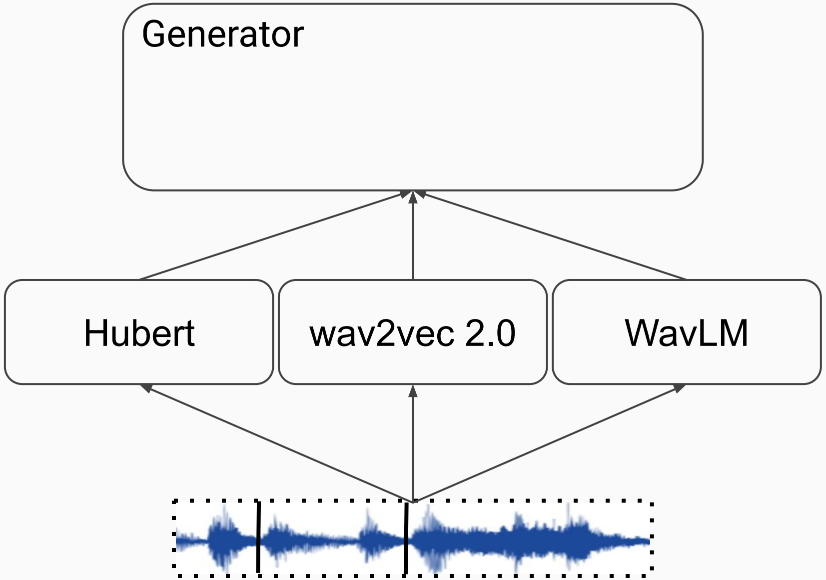


ESPNET


Discriminator
Front end
Generator
Hubert
wav2vec 2.0
WavLM

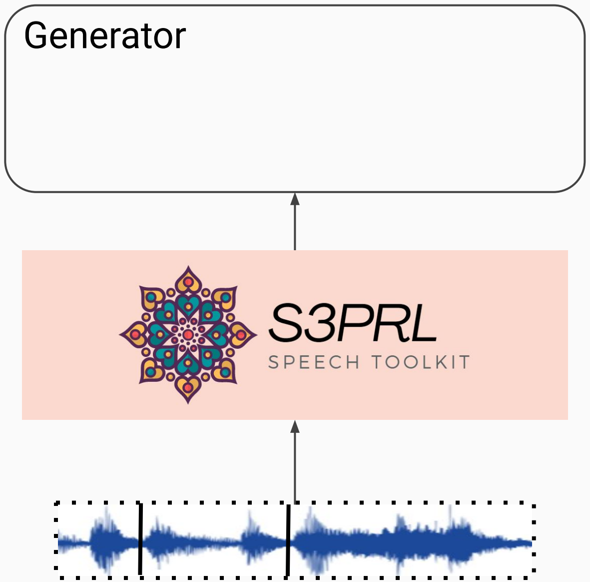
ESPNET

- Front end



ESPNET

- Front end



ESPNET

- Front end

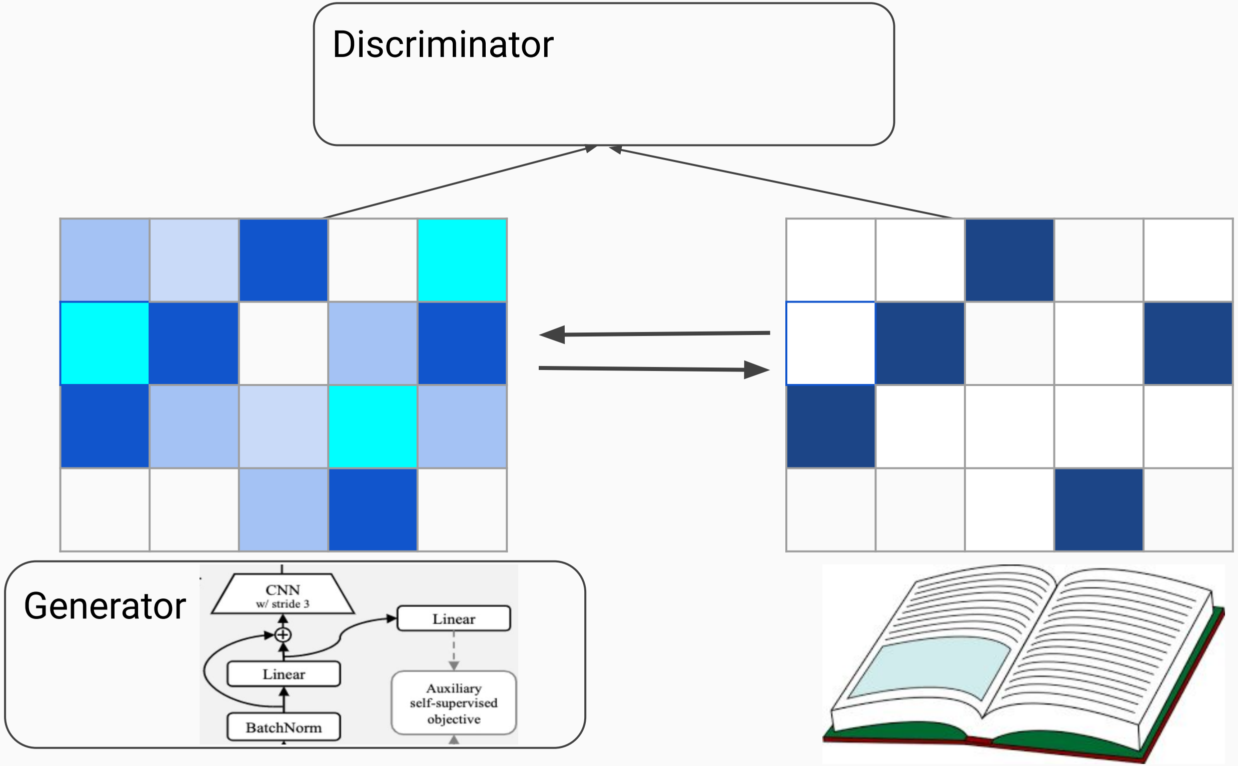
S3PRL
SPEECH TOOLKIT

- Faster data preprocessing
  - Parallel
    - VAD
    - Remove silence
    - MFCC clustering
  - On-the-fly feature extraction
    - Trainable weighted sum of features from different layer

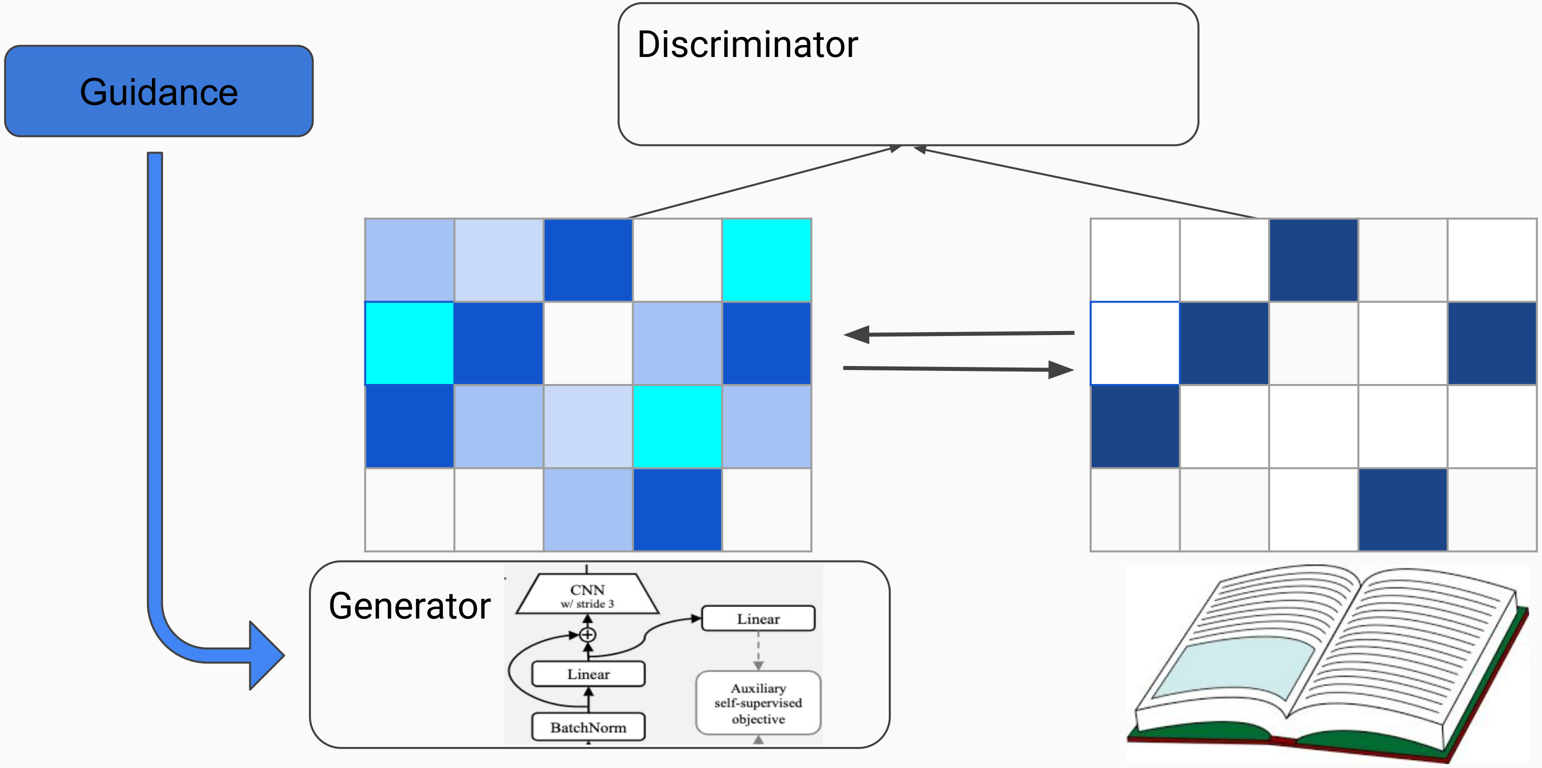
ESPNET

- Front end
- Faster data preprocessing
  - Parallel
    - VAD
    - Remove silence
    - MFCC clustering
  - On-the-fly feature extraction
    - Trainable weighted sum of features from different layer
- Training
  - Reproducibility
  - Efficiency
  - Performance

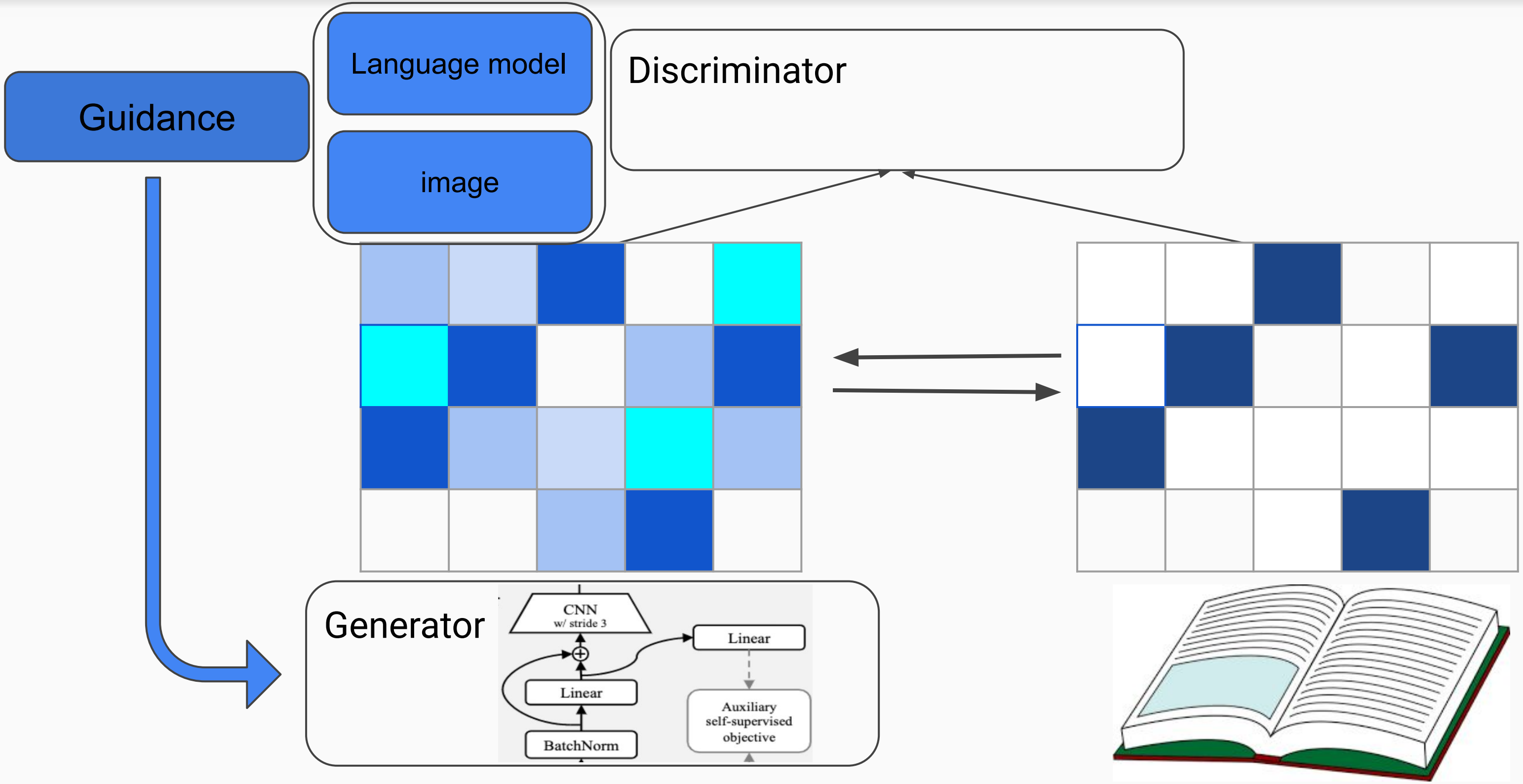
Text: “C B A D B”, phoneme set {A, B, C, D}



Text: “C B A D B”, phoneme set {A, B, C, D}



Text: “C B A D B”, phoneme set {A, B, C, D}



ESPNET

- Front end
- Faster data preprocessing
  - Parallel
    - VAD
    - Remove silence
    - MFCC clustering
  - On-the-fly feature extraction
- Training

Thanks!

Dongji Gao
Jiatong Shi
Ann Lee
Paola Garcia
Shinji Watanabe
Hung-yi Lee

Results on Librispeech

Pseudo Label	# of class	Avg. PER
None	-	15.9 ± 1.1
wav2vec2.0 VQ indices[2]	320×2	16.6 ± 2.2
k-means clustering wav2vec2.0 features	32	16.4 ± 1.4
	64	15.5 ± 1.8
	128	15.9 ± 0.9
k-means clustering MFCC audio features	50	15.2 ± 0.9
	64	$\mathbf{13.6\pm0.9}$
	100	14.8 ± 1.3
	128	16.8 ± 1.7

Text: “C B A D B”, phoneme set {A, B, C, D}