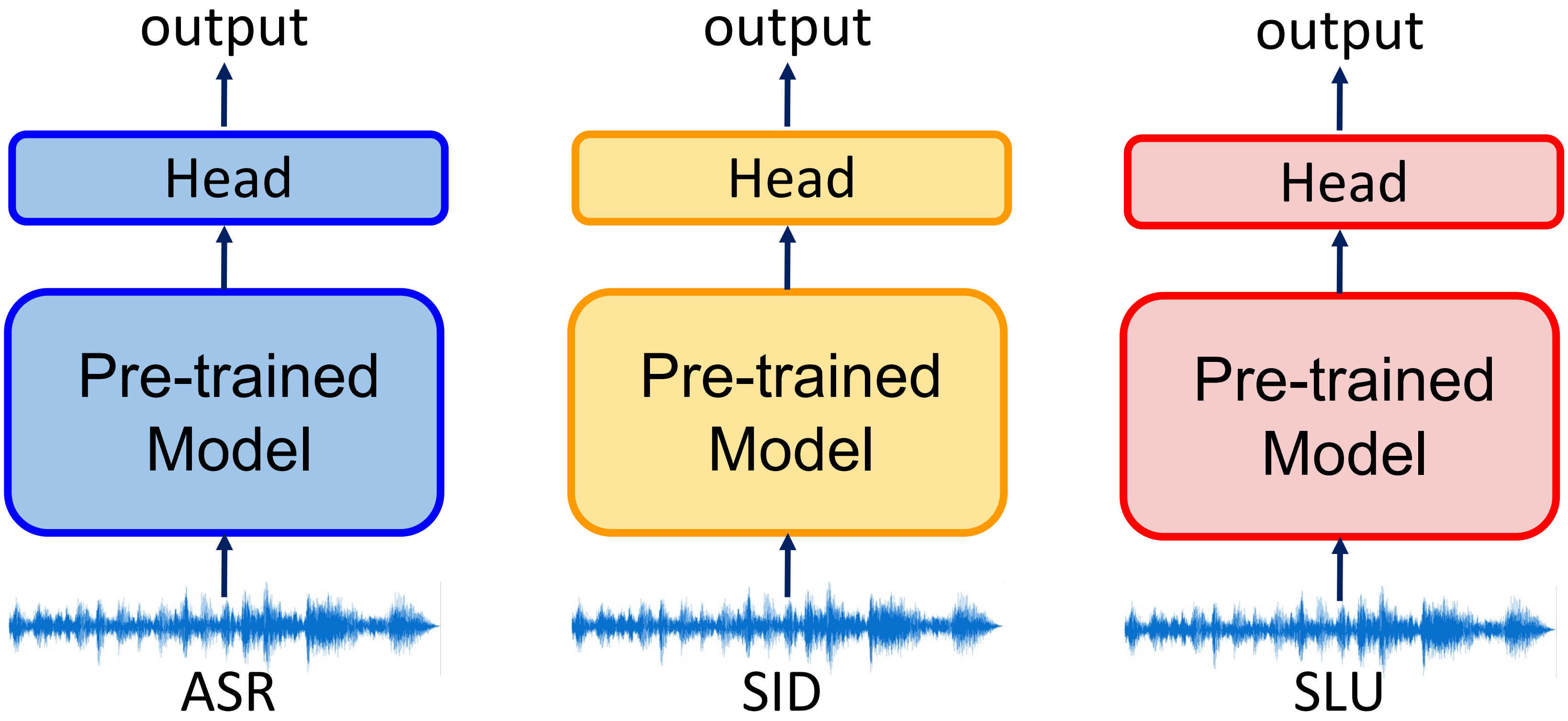
How to better leverage pre-trained speech models

Kai-Wei Chang[1], Allen Fu[1], Zih-Ching Chen[1], Shang-Wen Li[2], Hung-yi Lee[1]
[1]National Taiwan University
[2]Meta AI

Typical way

We have to store a ginantic pre-trained models for each task.



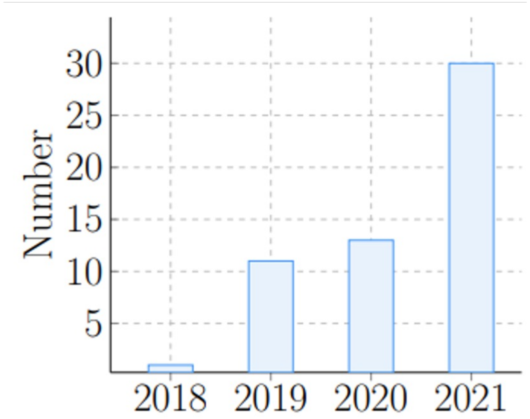
Adapter / Prompting

Adapter / Prompting gains popularity in NLP and yields promising results

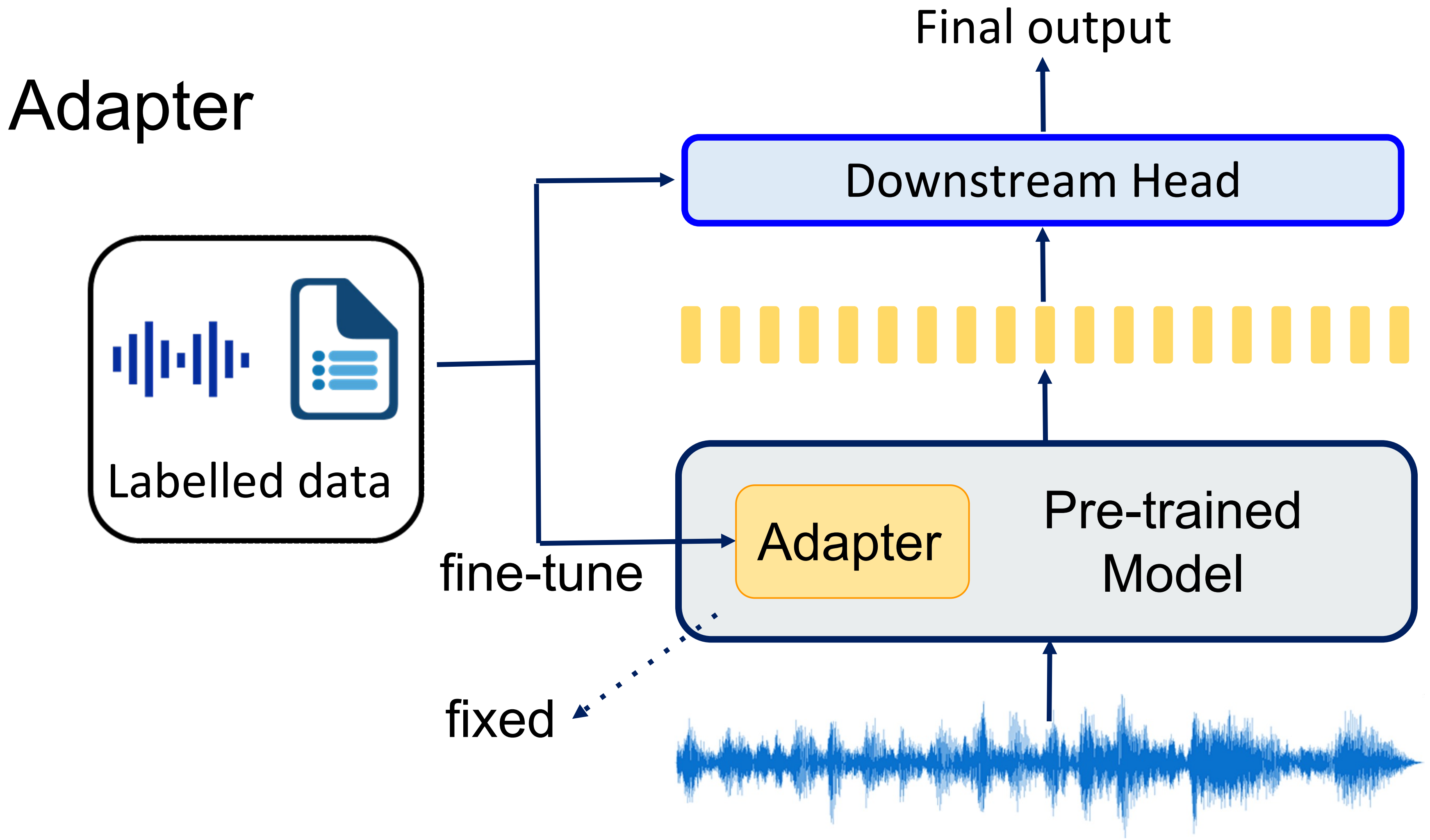
https://adapterhub.ml/

No. of papers about prompting

https://arxiv.org/abs/2107.13586

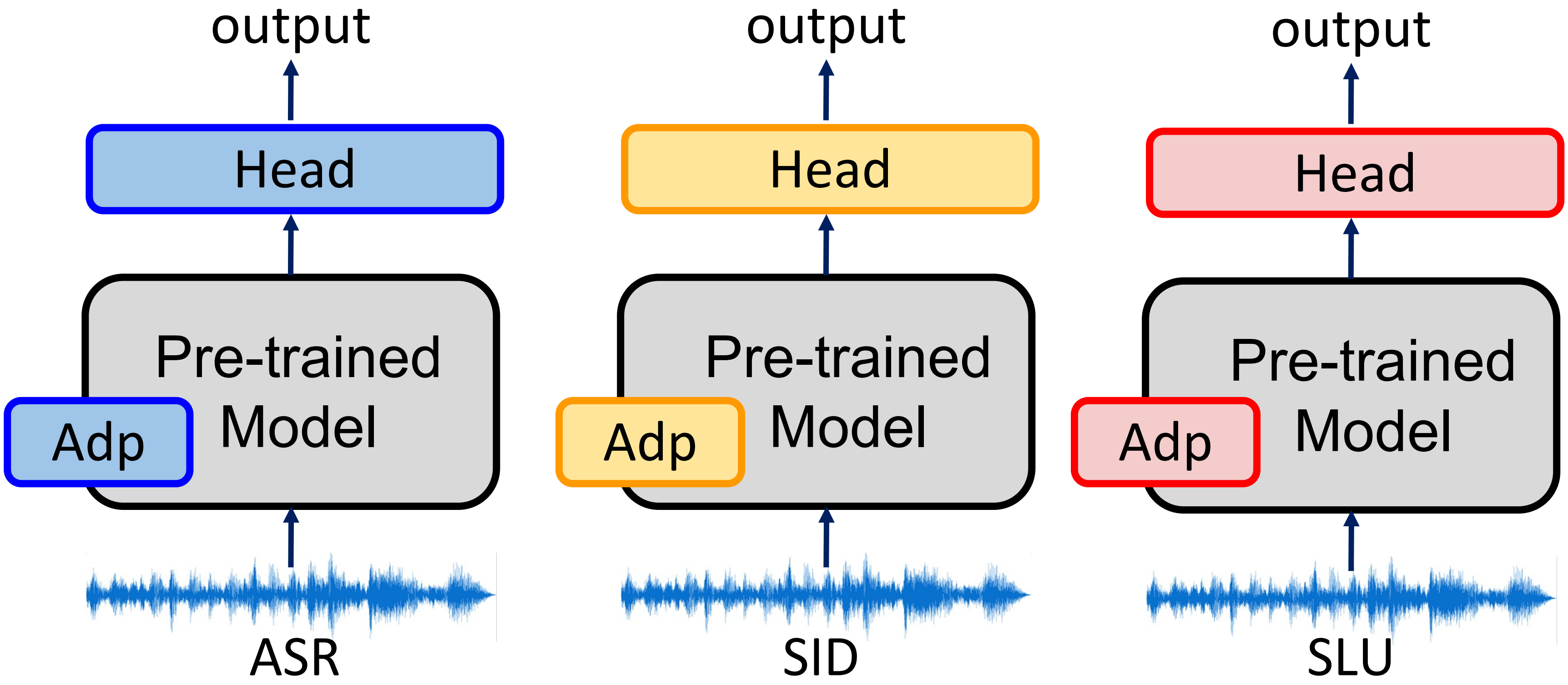


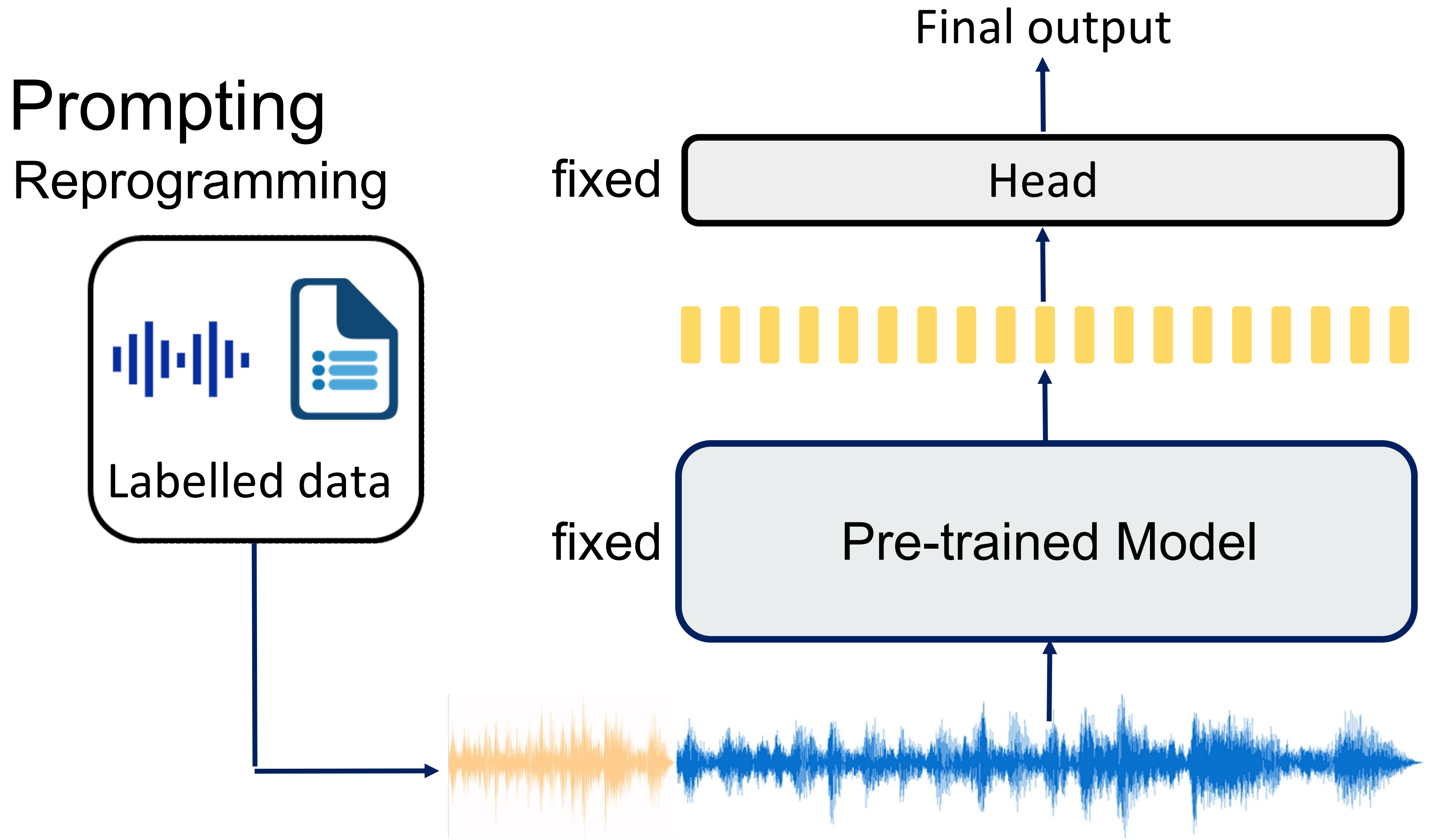
Does Adapter/Prompting also work on speech?


Adapter
Final output
Downstream Head
Labelled data
fine-tune
Adapter
Pre-trained Model
fixed

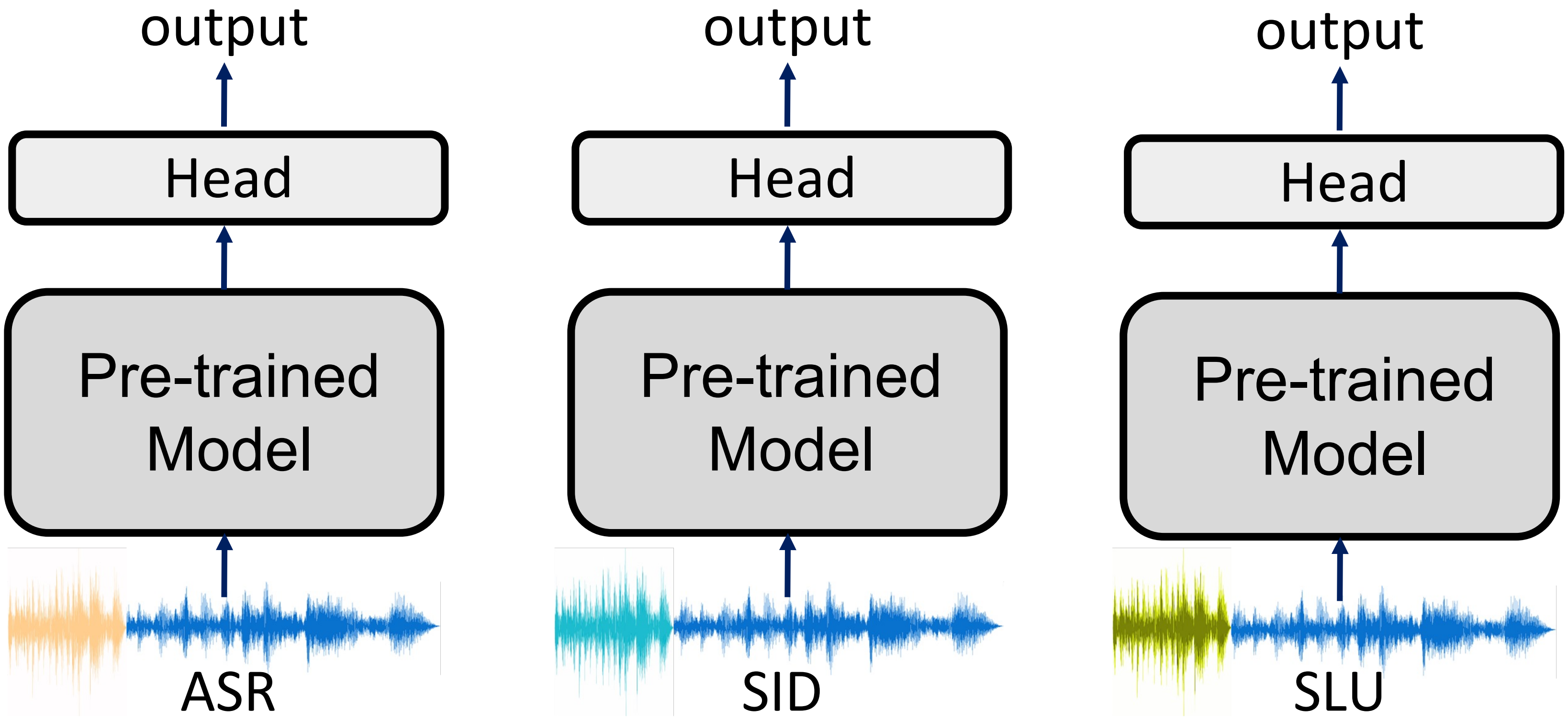
Adapter

only have to store adapters for each task




Prompting
Reprogramming
Labelled data
fixed
Head
fixed
Pre-trained Model
Final output

Prompting


output
Head
Pre-trained
Model
ASR
output
Head
Pre-trained
Model
SID
output
Head
Pre-trained
Model
SLU

Research questions

- Does it work?
  - Additional adapter/prompting layers improve **performance**
- What **tasks** it improves/doesn’t improve
- What scenarios it improves
  - **Parameter** efficiency
  - **Robustness**
  - **Few-shot** adaptation

Current exploration

	SSL models in s3prl	GSLM	others (e.g., pGSLM)
Prompt		ongoing	
Adapter	ongoing		

Does it work - **performance**

What **tasks** it improves/doesn’t improve

GSLM

1. speech to unit
2. unit language model
3. unit to speech

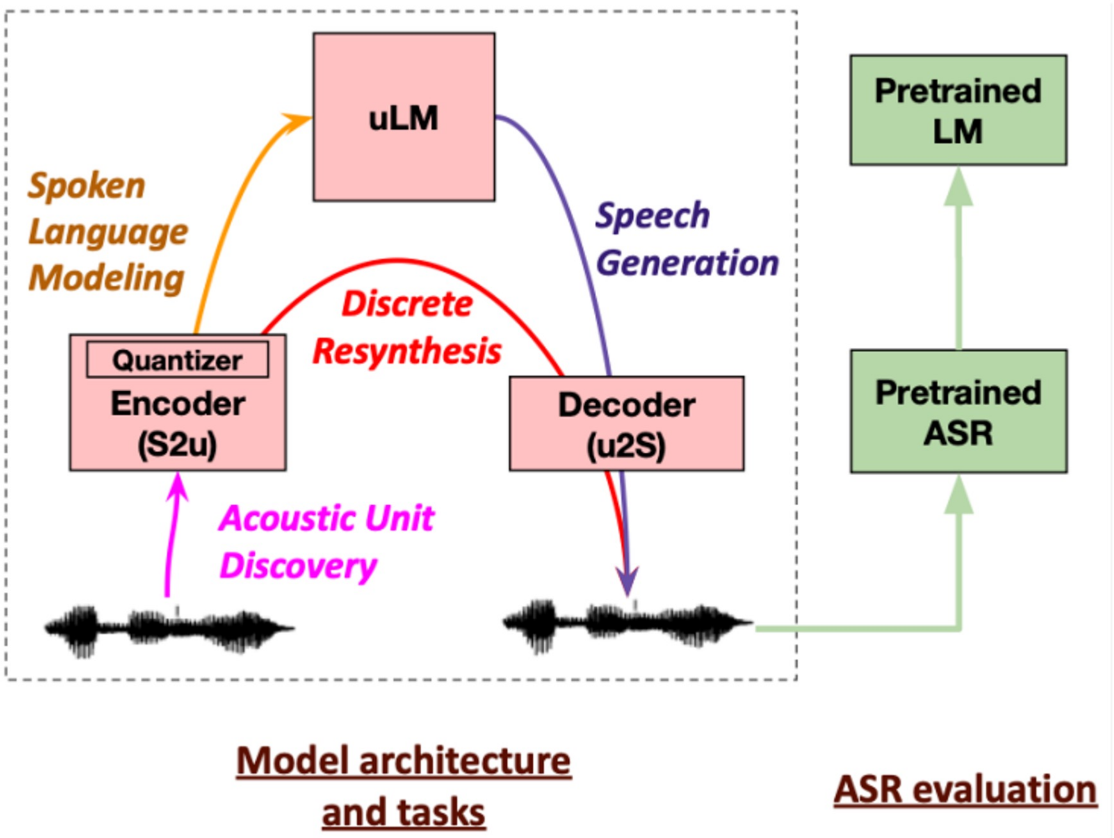
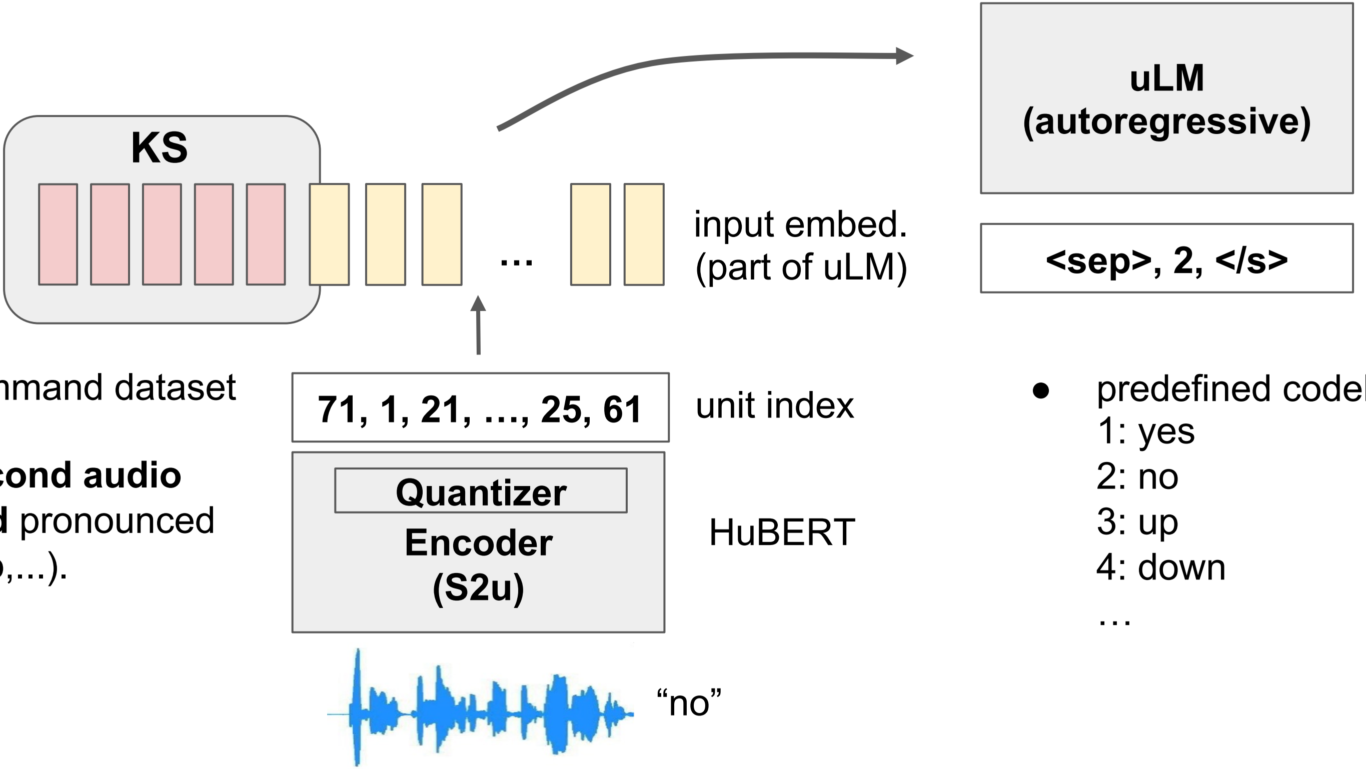


Figure 1: **Setup of the baseline model architecture, tasks and metrics.**

Prompting for GSLM: Keyword Spotting



- speech command dataset
- 12-classes
- input: **1 second audio** only **1 word** pronounced (yes, no, up,...).

- predefined codebook:
  1: yes
  2: no
  3: up
  4: down
  …

Prompting for GSLM: results

settings \ tasks	KS (↑)	IC (↑)	PR (↓)	ASR (↓)
prefix-length	6	6	180	180
trainable params	0.15M	0.15M	4.5M	4.5M
SUPERB downstream params	0.2M	0.2M	0.22M	42.6M
best valid loss	0.012	0.008	0.185	0.19
Prompt	94.6%	98.4%	21.1%	45.2%
HuBERT (SUPERB)	96.3%	98.3%	5.4%	6.4%

Current exploration - **prompting**

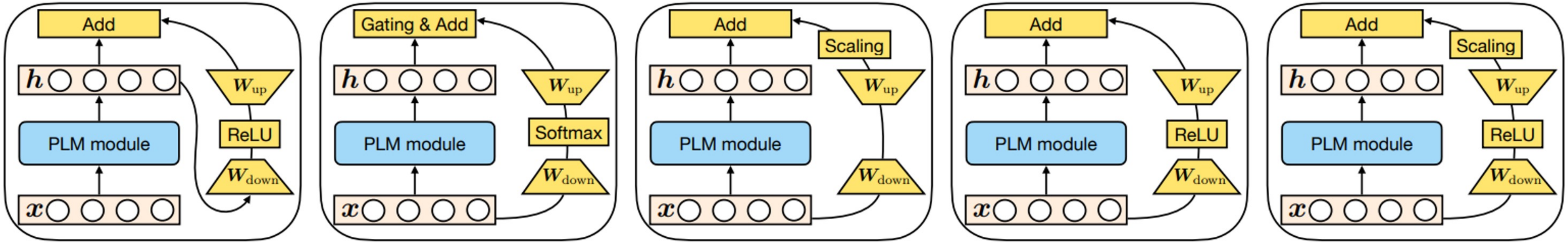
	SSL models in s3prl	GSLM	others (e.g., pGSLM)
Prompt		ongoing	
Adapter	ongoing		

Does it work? promising **performance**

What **tasks** it improves/doesn’t improve? clssification / sequence decoding

Adapters for SSL models

- How adapters work on different self-supervised speech models
- Add adapters to upstream models



(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

- [Unified Adapter] Towards a Unified View of Parameter-Efficient Transfer Learning

Adapters for SSL models: results

	ASR (WER ↓)			IC (acc. ↑)		
	HuBERT	DeCoAR2	TERA	Hubert	DeCoAR2	TERA
Baseline	6.42	14.45	21.53	98.3	90.8	60.7
Houlsby	6.50	13.28	20.81	**98.4**	91.0	62.0
AdapterBias	6.41	**13.28**	21.09	**98.4**	91.2	60.7
BitFit	6.42	13.64	**20.65**	**98.4**	**91.2**	**62.4**
LoRA	**6.28**	13.30	20.91	**98.4**	91.1	62.0

Adapters for SSL models: results (cont'd)

	PR (PER ↓)			**SF (F1 ↑)**		
	HuBERT	DeCoAR2	TERA	Hubert	DeCoAR2	TeRA
Baseline	5.41	15.31	53.23	88.53	83.27	66.92
Houlsby	5.53	17.22	55.12	**88.76**	82.17	**69.20**
AdapterBias	5.44	18.22	54.26	88.55	**83.41**	67.70
BitFit	**5.38**	18.62	54.59	88.23	81.54	68.42
LoRA	5.54	18.36	54.39	88.53	83.13	68.52

Current exploration - **adapter**

	SSL models in s3prl	GSLM	others (e.g., pGSLM)
Prompt		ongoing	
Adapter	ongoing		

Does it work? promising **performance (especially small models)**

What **tasks** it improves/doesn't improve? ASR, SF, or less performant models / tasks with performance saturated

TODO (before workshop)

	SSL models in s3prl	GSLM	others (e.g., pGSLM)
Prompt		ongoing	
Adapter	ongoing		

- Improve experiment settings
- Integrate implementation with s3prl
- Fill out the table
- Establish experiments for more **scenarios**

TODO (at workshop)

	SSL models in s3prl	GSLM	others (e.g., pGSLM)
Prompt		ongoing	
Adapter	ongoing		

- Experiment on more
  - **scenarios**
  - **tasks** (in SUPERB)
  - other tasks (e.g., prompting for dialog response generation)
- Synergy with other directions