How to better leverage pre-trained speech models: prompt & adapter

Kai-Wei Chang[1], Allen Fu[1], Zih-Ching Chen[1], Chih-Ying Liu[1], Hua Shen[2], Shang-Wen Li[3], Hung-yi Lee[1]
[1]National Taiwan University
[2]Penn State University
[3]Meta AI

Outline

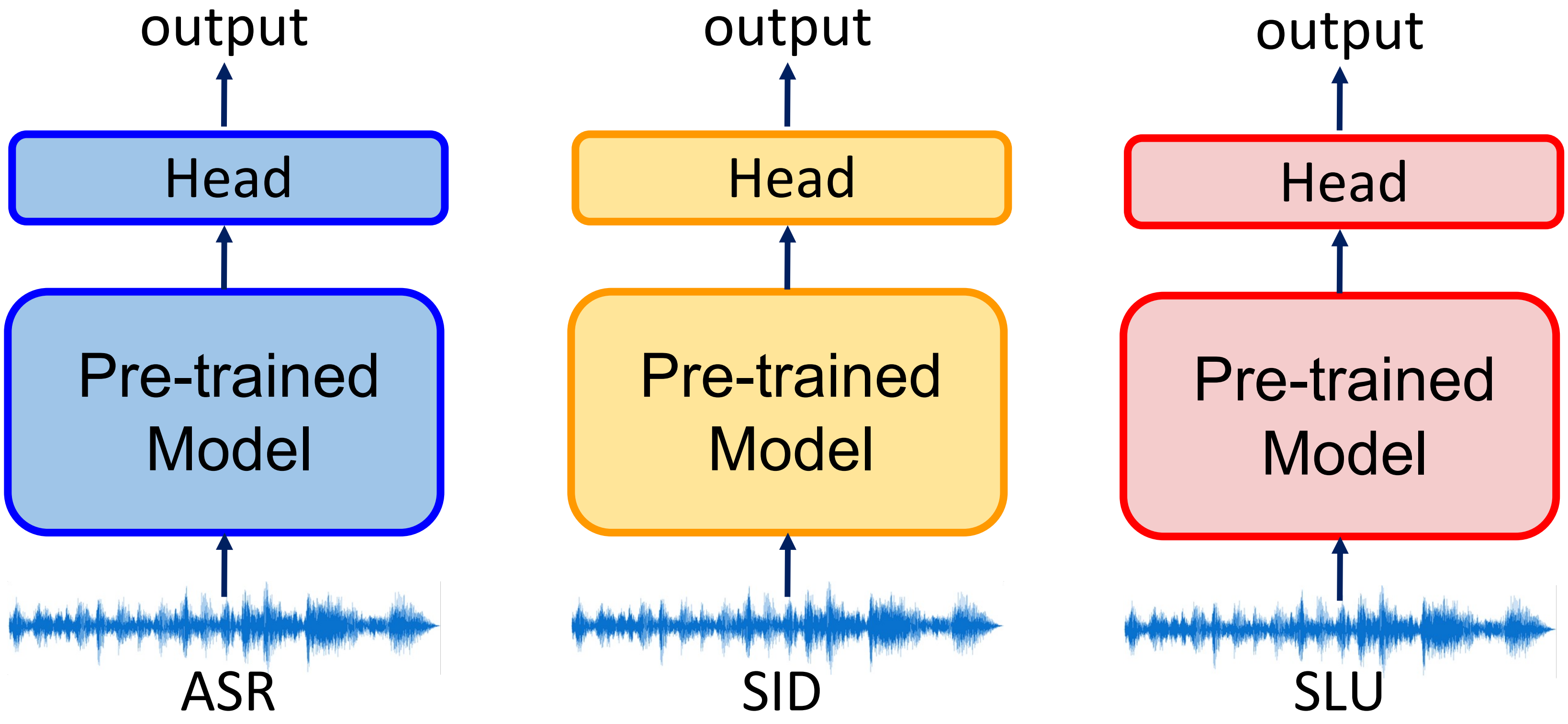
- Recap: Motivations, prompt and adapter background
- Prompting on GSLM (Kai-Wei Chang, Hua Shen)
- Adapter for SSL speech models (Allen Fu, Zih-Ching Chen)

Outline

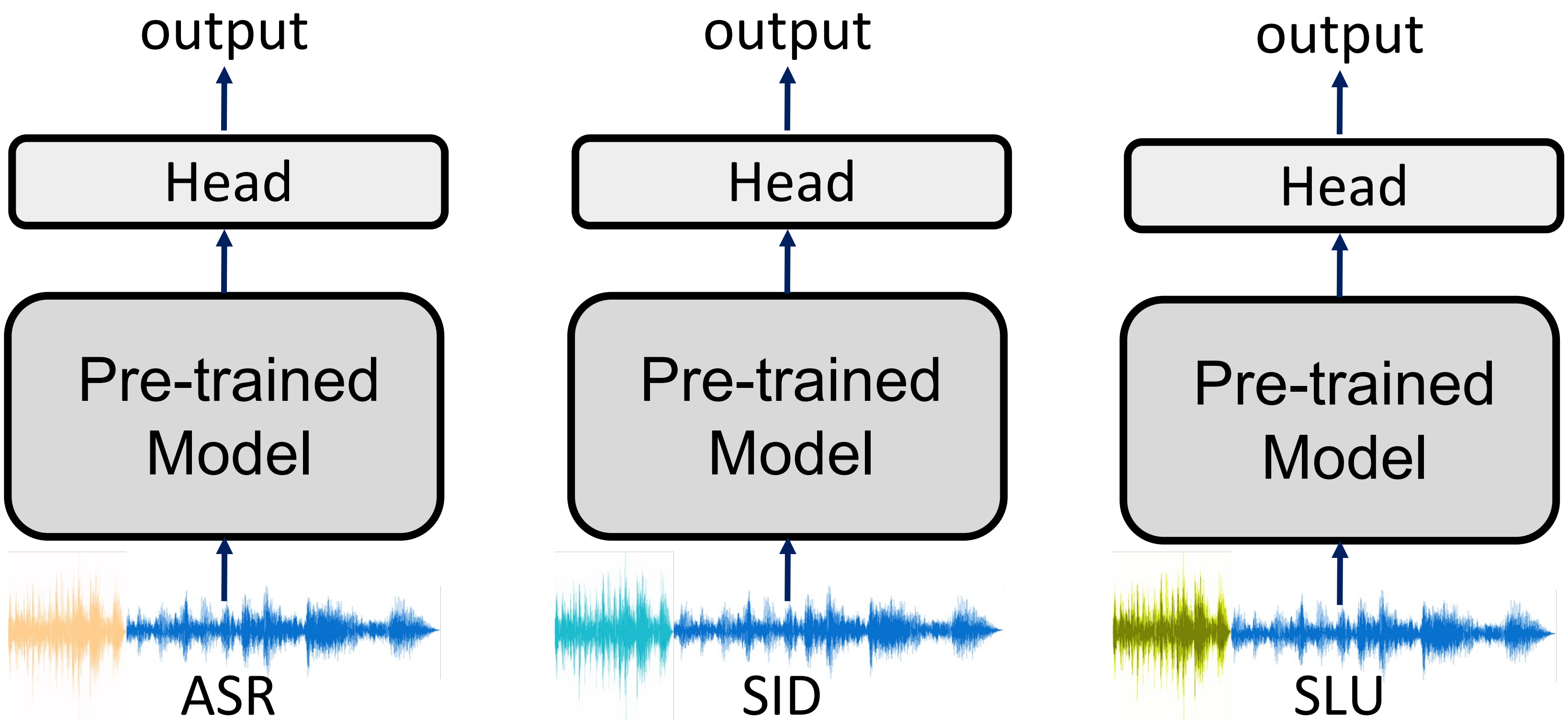
- Recap: Motivations, prompt and adapter background
- Prompting on GSLM (Kai-Wei Chang, Hua Shen)
- Adapter for SSL speech models (Allen Fu, Zih-Ching Chen)

Recap - Typical way

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

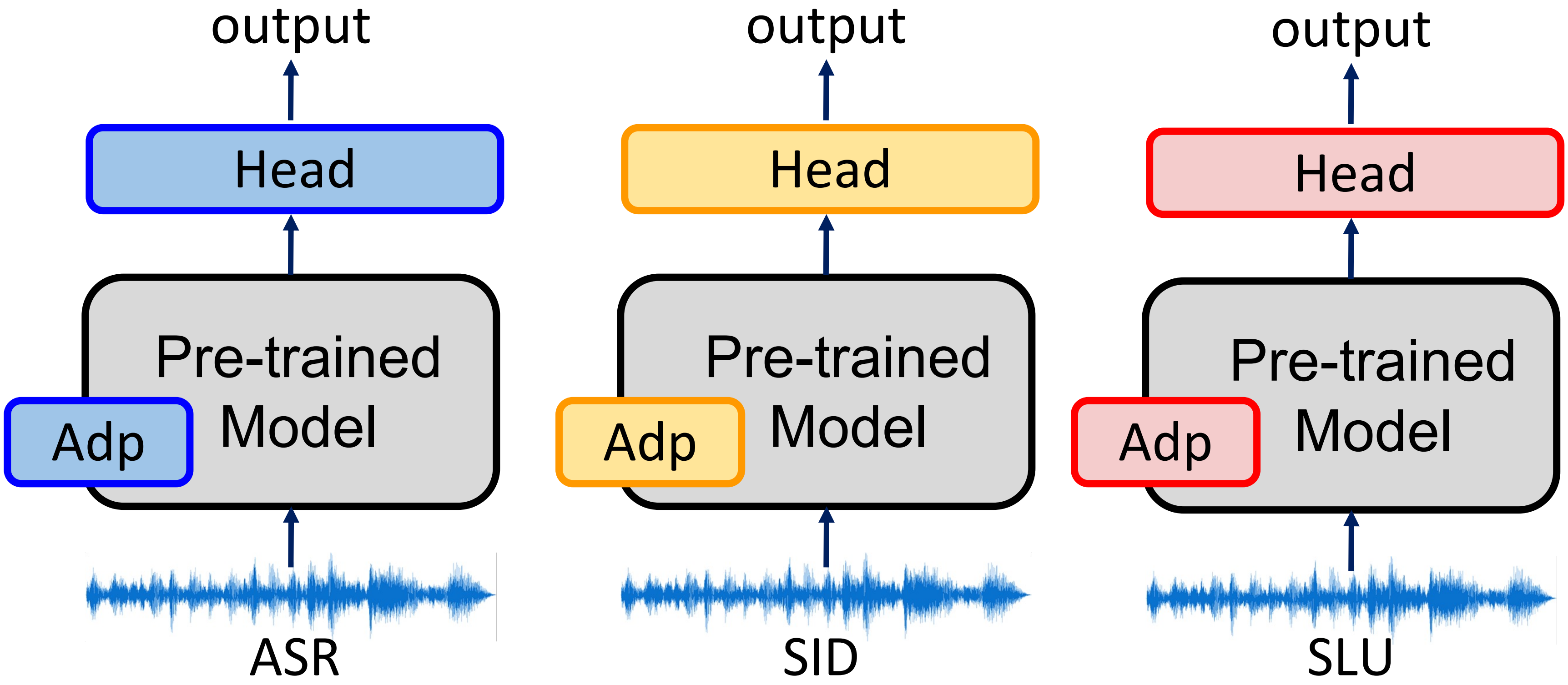


Recap - Prompting



Recap - Adapter

only have to store adapters for each task



Recap - Research questions

- Does adapter/prompting yield reasonable **performance**?
- What **tasks** adapter/prompting improves/doesn't improve
- What scenarios adapter/prompting improves
    - **Parameter** efficiency
    - **Robustness**
    - **Few-shot** adaptation
    - Other directions in SSL team
    - …

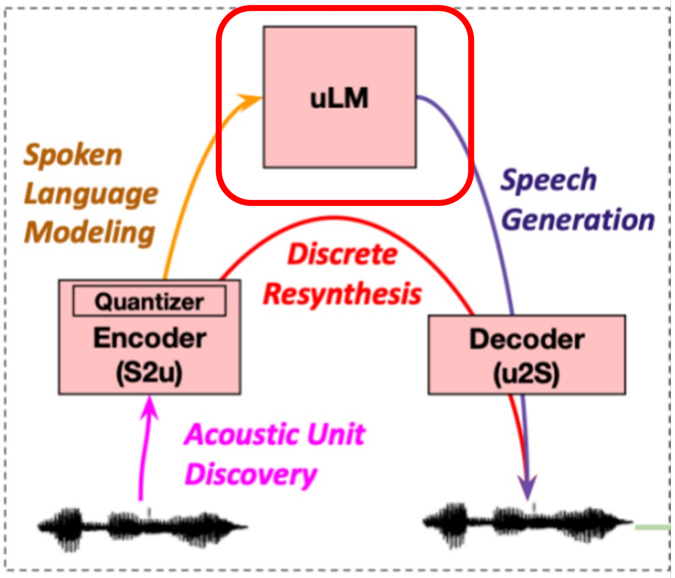
Outline

- Recap: Motivations, prompt and adapter background
- Prompting on GSLM (Kai-Wei Chang, Hua Shen)
- Adapter for SSL speech models (Allen Fu, Zih-Ching Chen)

Recap - GSLM

1. Speech to Unit (S2u)
2. unit Language Model (uLM)
3. unit to Speech (u2S)

- Prompting on the uLM of GSLM
- Take advantage of the uLM pre-trained on a large corpus



Model architecture and tasks

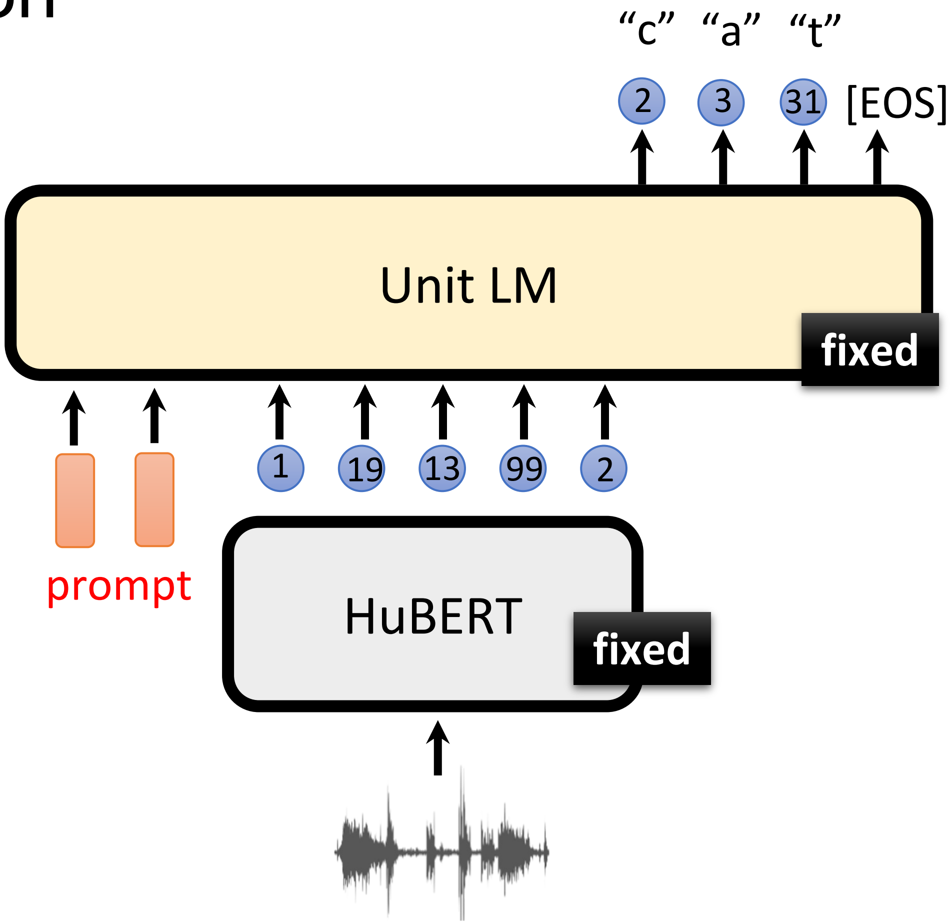
- [GSLM] Lakhotia et.al., Generative Spoken Language Modeling from Raw Audio

Recap: **Prompting** on Unit LM (uLM)

Speech Recognition (ASR)

Unit ID	Character
1	“m”
2	“c”
3	“a”
4	“g”
…	

label mapping (Verbalizer)



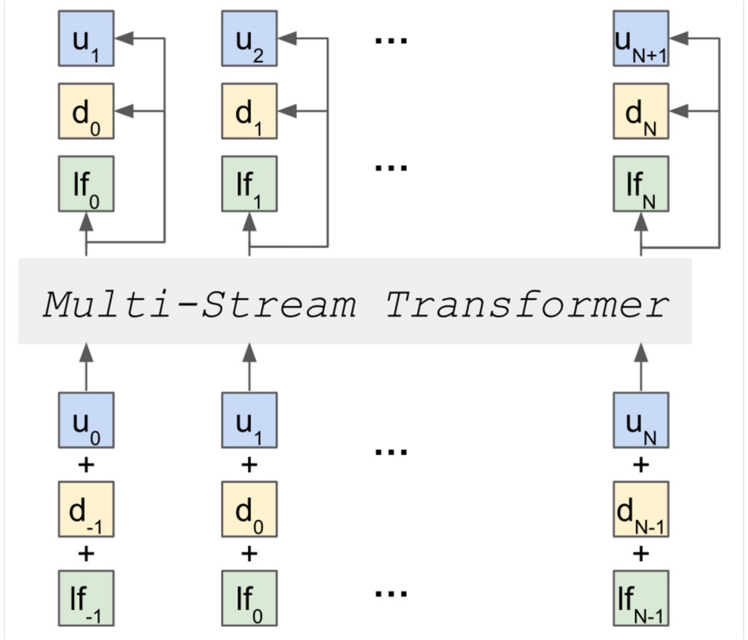
Prompting on uLM: analysis

- Performance is **improved** on speech classification tasks
  - e.g. Keyword Spotting, Intent Classification
- Performance is **not improved** on sequence decoding tasks
  - e.g. ASR, Slot Filling
  - The performance is limited by the architecture of the pre-trained model itself
  - The framework suffers from long input sequence

Prompting on pGSLM

- To better leveage the LM's knowledge on "prosody"
- Perform on prosody related tasks e.g. emotion recognition

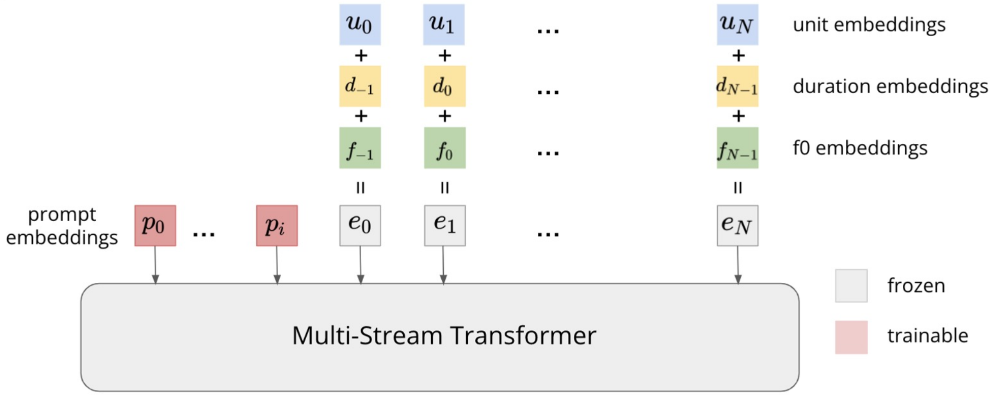
- u: discrete unit
- d: duration
- lf: log f0

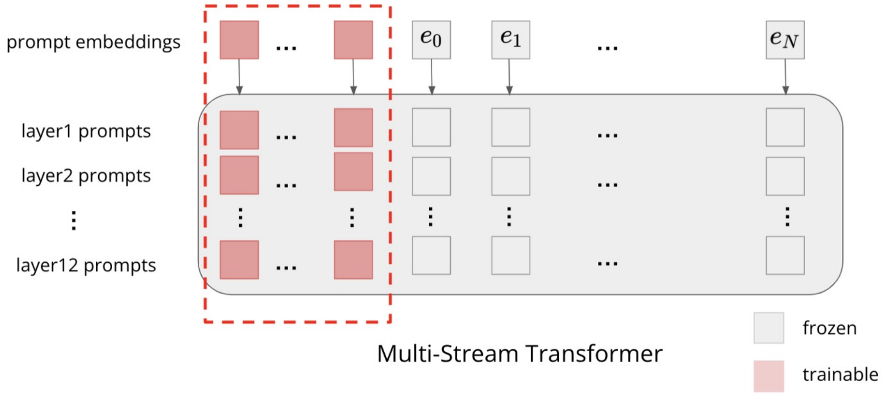


Deep prompt tuning

Input Prompt Tuning

- add trainable prompts on the input embeddings





Deep Prompt Tuning

- add trainable prompts on the input embedding and input of each transformer layer

Experiments - IC

Speech Classification Task

Method	ACC	# Params
SUPERB Fine Tuning	98.34	0.2 M
GSLM Deep Prompt Tuning	98.40	0.15 M
pGSLM Input Prompt Tuning	98.25	0.1 M
pGSLM Deep Prompt Tuning	98.15	0.1 M

Experiments - ER

Speech Classification Task

Method	ACC	# Params
SUPERB Fine Tuning	**65.92**	0.2 M
GSLM Deep Prompt Tuning	< 50 %	0.3 M
pGSLM Finetuning	49.88	151 M
pGSLM Input Prompt Tuning	**56.11**	0.3 M
pGSLM Deep Prompt Tuning	**54.50**	0.3 M

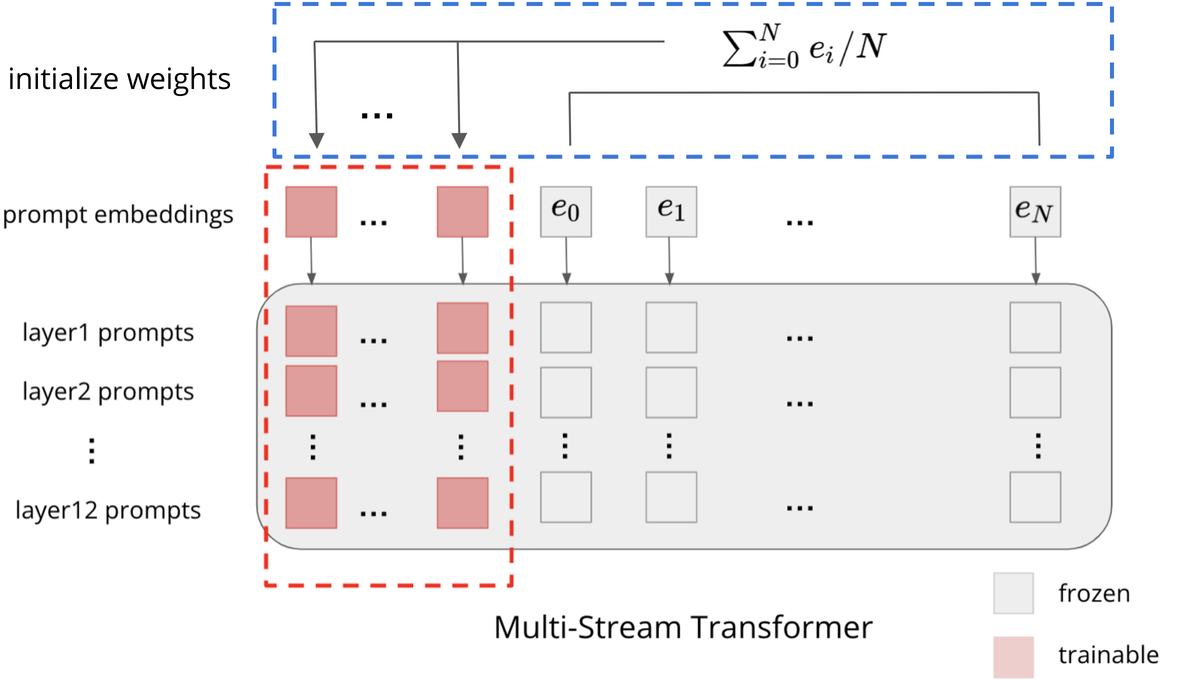
Experiments - ASR

Sequence Decoding Task

Method	WER	# Params
SUPERB Dowstream	**6.42**	43 M
GSLM Deep Prompt Tuning	34.17	4.5 M
pGSLM Input Prompt Tuning	76.72	4.5 M
pGSLM Deep Prompt Tuning	73.93	4.5 M
GSLM Finetuning	26.19	151 M
pGSLM Finetuning	13.20	151 M

Prompt Initialization

Initialize the prompts' weights with the embedding of the first batch of data



Experiments - ASR

Method	Init	WER	# Params
Input Prompt Tuning	False	76.72	0.45M
Input Prompt Tuning	True	72.65(↓4.07)	0.45M
Deep Prompt Tuning	False	73.93	4.5M
Deep Prompt Tuning	True	73.90(↓0.03)	4.5M

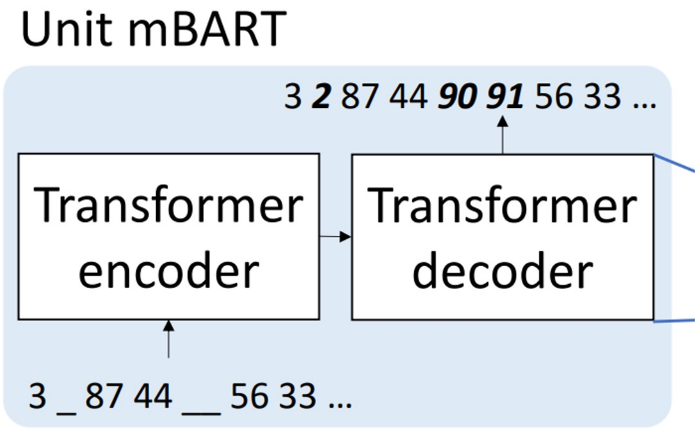
- Prompt init. help the performance to some extent

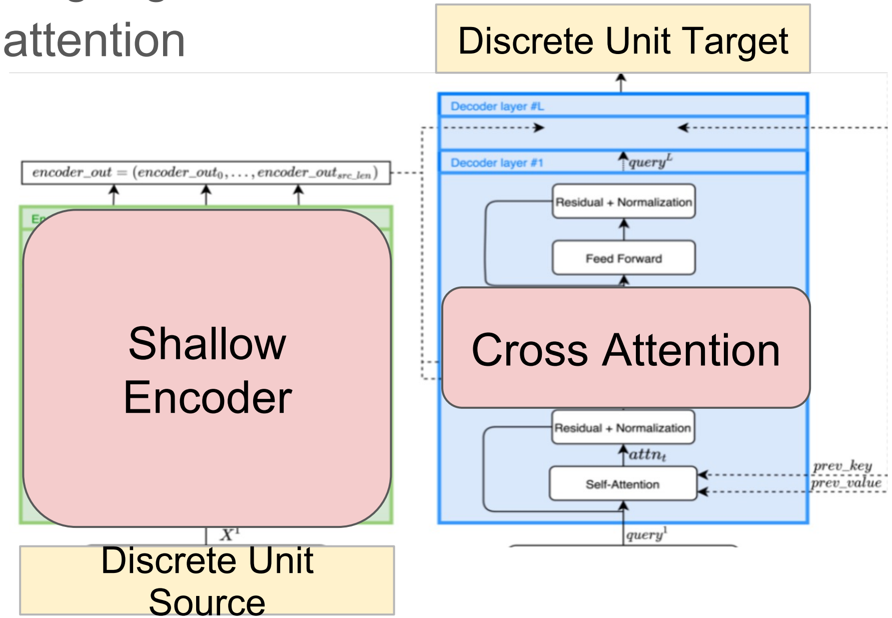
Observations

1. For prosody related task: pGSLM > GSLM
2. For sequence decoding tasks: GSLM > pGSLM with the same number of trainable parameters.
3. Prompt initialization can help the performance
4. Currently, sequence decoding tasks are still very challenging for prompting

Future work 1: Prompting on Different Architecutres

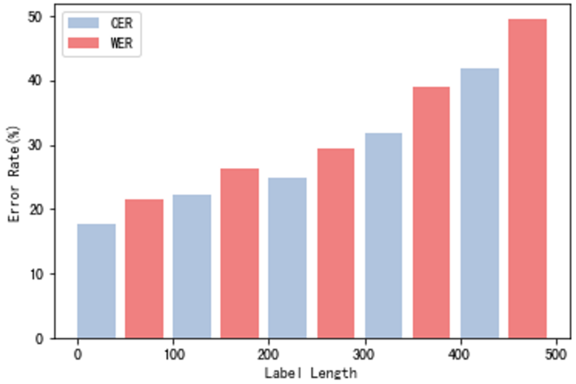
- uLM is a “decoder-only” language model
  - performs poor on sequence decoding tasks
- **Unit mBART** is an “encoder-decoder” language model
- **Light weight adapter**: trainable cross attention



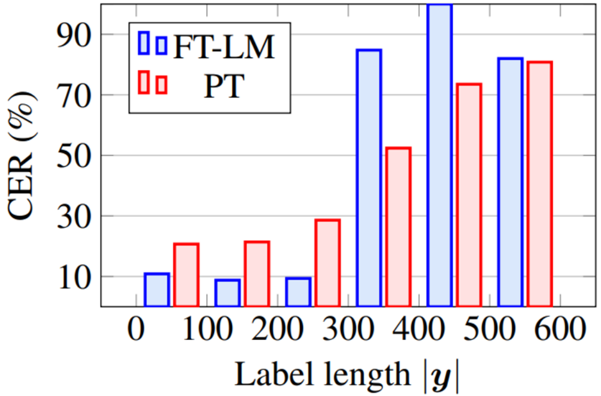


Future work 2: Length Adapter

- The current framework suffers from long sequence
- Combine the research from the compression team



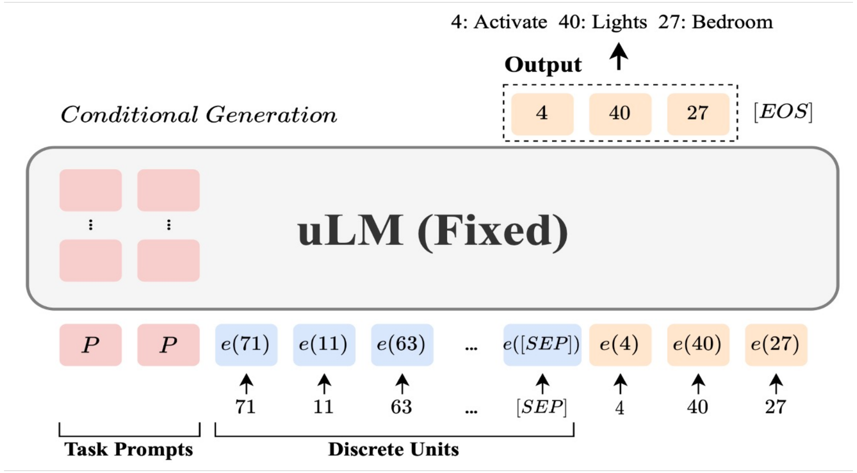
pGSLM



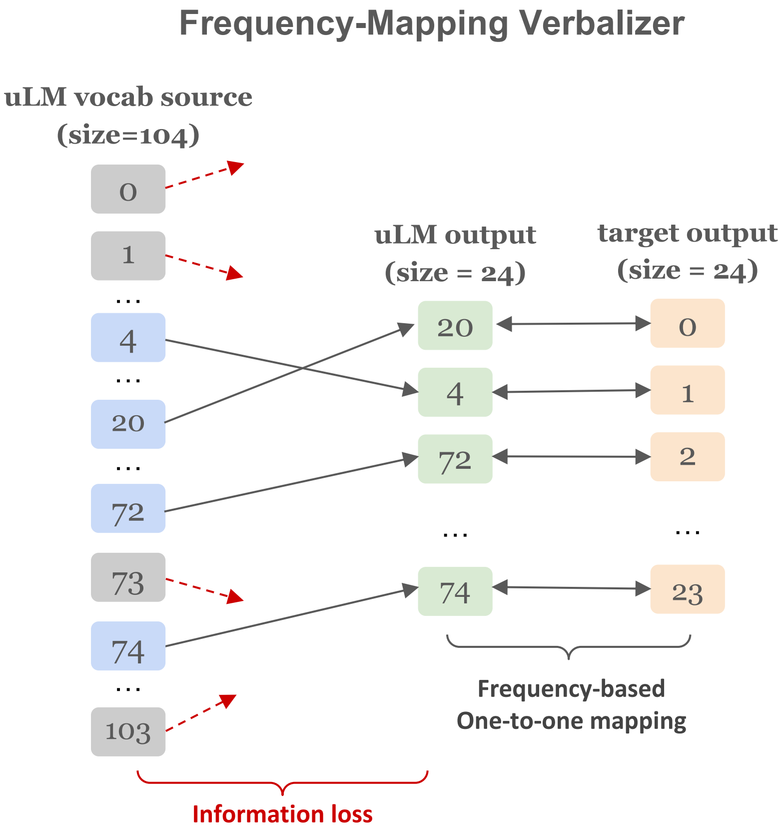
GSLM

Future work 3: Linear Verbalizer

How to better define the mapping between the model's output and task labels?



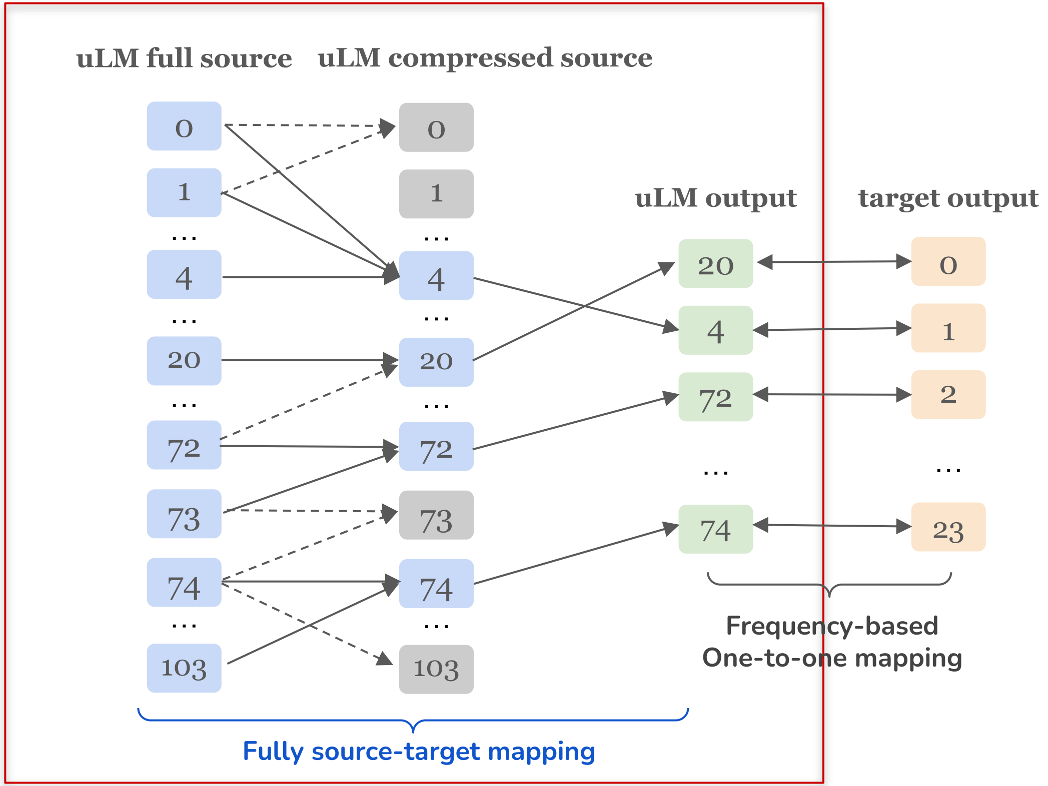
Current Approach: frequency-based algorithm (Simply counting the frequency of the units and the tasks labels)



Future work 3: Linear Verbalizer

Linear Verbalizer

Adding linear fully connected layers to make fully source-target label mapping without information loss.



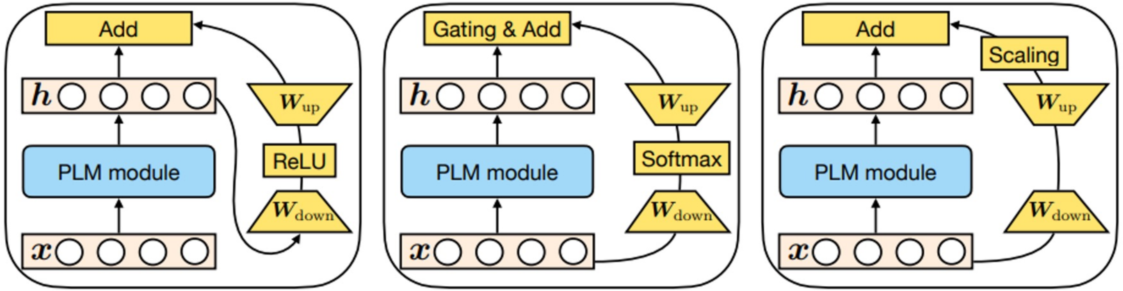
Yang et.al., Voice2Series: Reprogramming Acoustic Models for Time Series Classification

Outline

- Recap: Motivations, prompt and adapter background
- Prompting on GSLM (Kai-Wei Chang, Hua Shen)
- Adapter for SSL speech models (Allen Fu, Zih-Ching Chen)

Adapters for SSL models

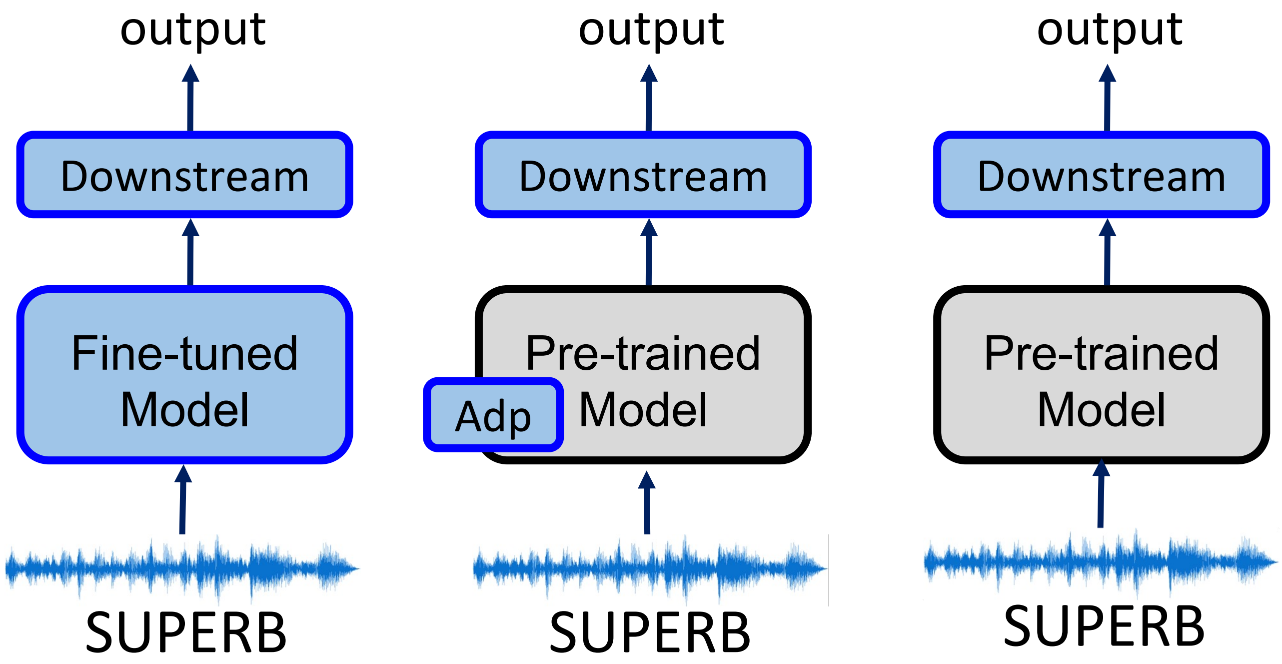
- How adapters work on different self-supervised speech models
- Add adapters to upstream models



(a) Adapter (b) Prefix Tuning (c) LoRA

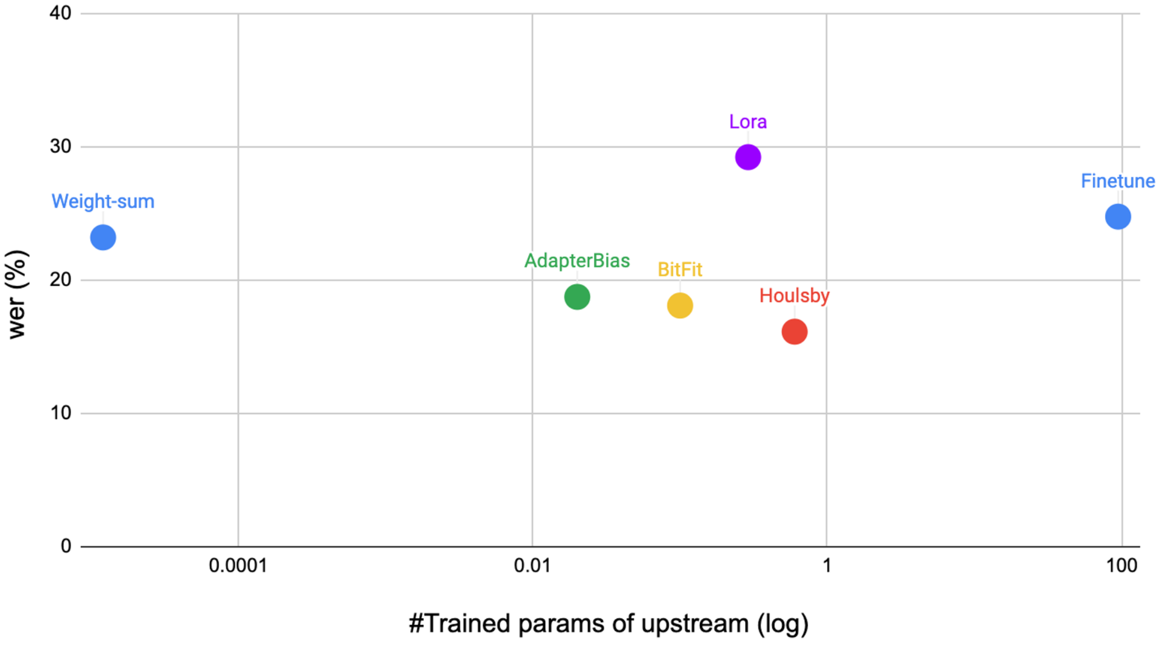
- [Unified Adapter] Towards a Unified View of Parameter-Efficient Transfer Learning

Fine-tuning SSL models in speech



#training parameters of speech tasks

- **Goal:** Explore different adapter methods compared with FT



Adapters w/ limited labels for downstream tasks

- **Experiment settings**
  - Measuring performance on SUPERB tasks (ASR, PR)
  - Varying in the size of training data (1hr, 10hr, 100hr)
  - Trying different upstream models (hubert, decoar2)
  - Examining different adapter methods

- **Ablation study**
  - Does different adapter architectures perform differently on different tasks?

Adapters w/ limited downstream data (ASR)

- downstream model : 2 layers of LSTM (43M)

hubert	#Params of Adp (M)	**ASR (WER ↓)**		
		1hr	10hr	100hr
FT	94.7	42.25	24.77	-
baseline	0	48.08	28.86	7.09
Houlsby	0.6	**30.96**	**16.14**	5.88
AdapterBias	0.02	37.05	18.75	**5.54**
BitFit	0.1	34.43	18.1	9.34
LoRA	0.29	49.59	29.27	6.94
Weighted-sum	just 12	43.31	23.21	6.42

decoar2	**ASR (WER ↓)**		
	1hr	10hr	100hr
FT	53.26	33.62	25.46
baseline	65.79	45.16	39.06
Houlsby	**46.71**	**31.64**	**28.63**
AdapterBias	49.89	33.01	29.89
BitFit	49.06	32.65	29.4
LoRA	63.25	42.79	39.52
Weighted-sum	61.32	41.73	36.26

Adapters w/ limited downstream data (PR)

- downstream model : Linear layer (80k)

hubert	#Params of Adp (M)	PR (PER ↓)		
		1hr	10hr	100hr
FT	94.7	-	-	2.68
baseline	0	16.21	13.7	7.74
Houlsby	0.6	10.64	**6.44**	**2.997**
AdapterBias	0.02	**8.64**	7.41	4.19
BitFit	0.1	9.07	6.97	4.23
LoRA	0.29	15.6	14.39	8.74
Weighted-sum	just 12	-	-	5.41

decoar2	PR (PER ↓)		
	1hr	10hr	100hr
FT	-	-	-
baseline	26.85	22.86	-
Houlsby	**14.12**	**11.1**	-
AdapterBias	15.45	13.72	-
BitFit	15.64	14.41	-
LoRA	27.74	26.97	-
Weighted-sum	-	-	14.93

Analysis - training w/ limited downstream labels

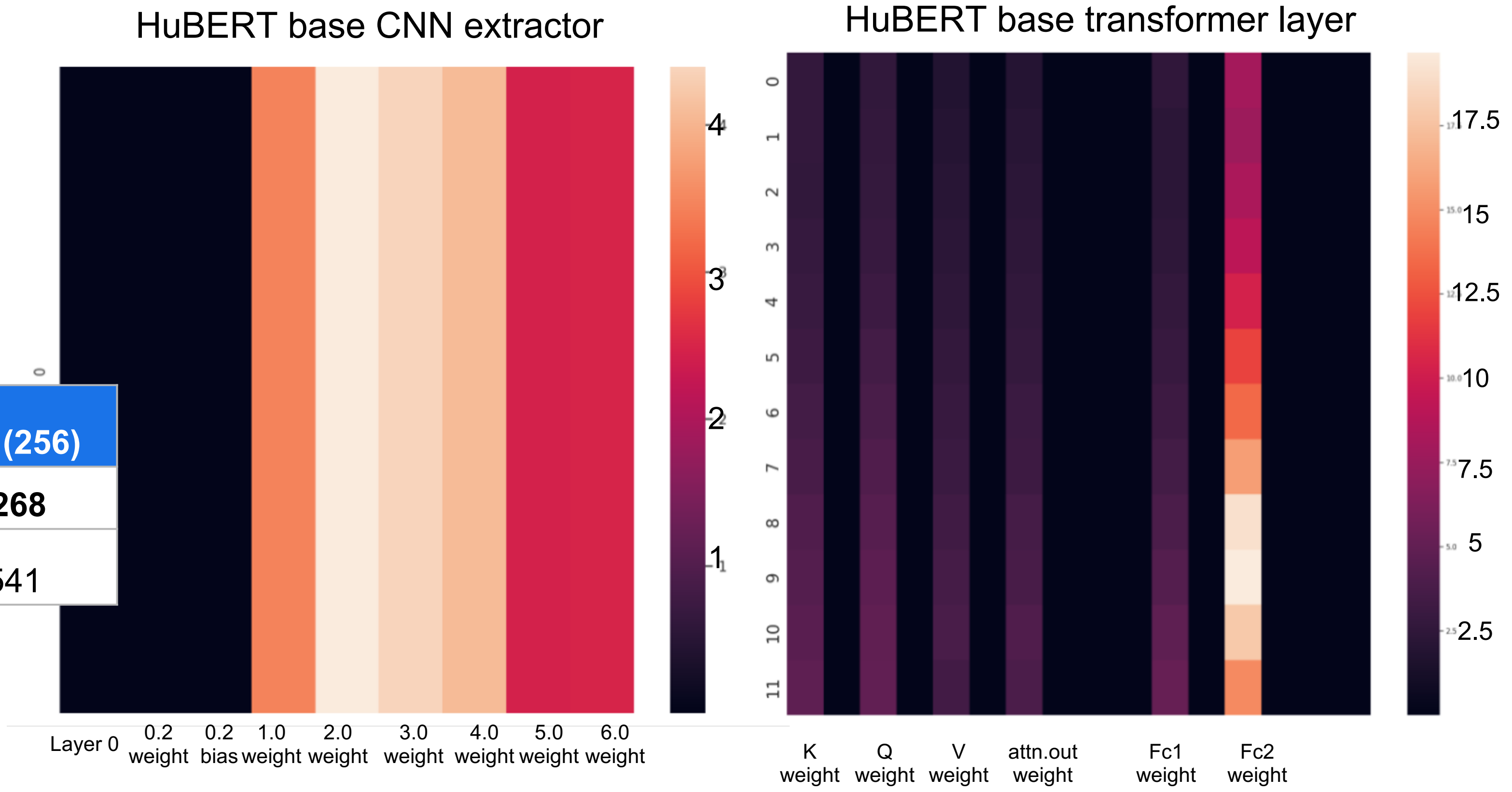
- ASR
  - Houlsby performs best
  - Adapter method improves better on small data and hubert
- PR
  - Houlsby performs best
  - Adapter method improves better on small data and decoar2
  - FT is hard to tune
- Performance ranking: Houlsby, AdapterBias&BitFit, Weight-sum, Lora

The difference in the weight of a fine-tuned model : **PR**

Phoneme Recognition

Difference in weight

PR	0.22M (256)
Fully FT	**0.0268**
Weighted sum	0.0541



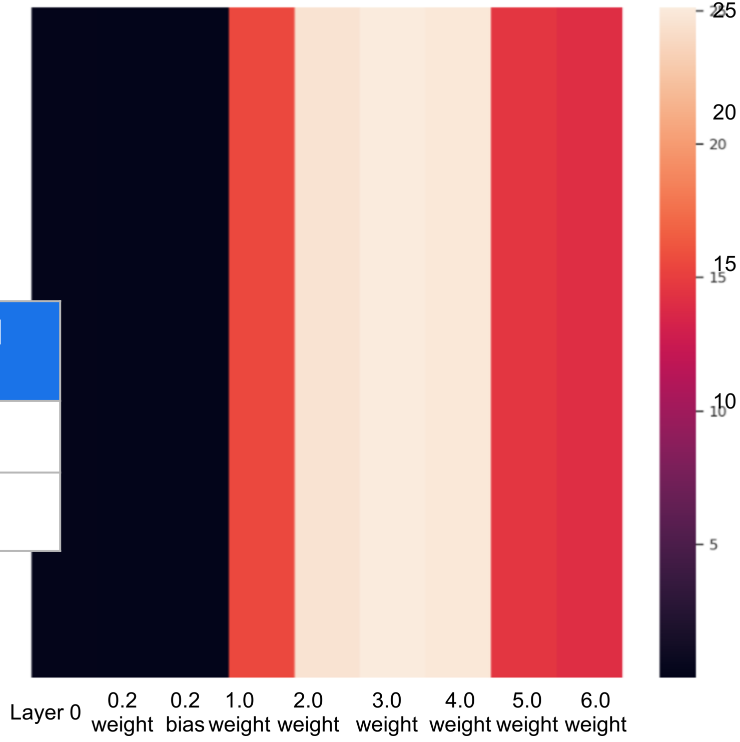
The difference in the weight of a fine-tuned model : **SF**

Slot filling

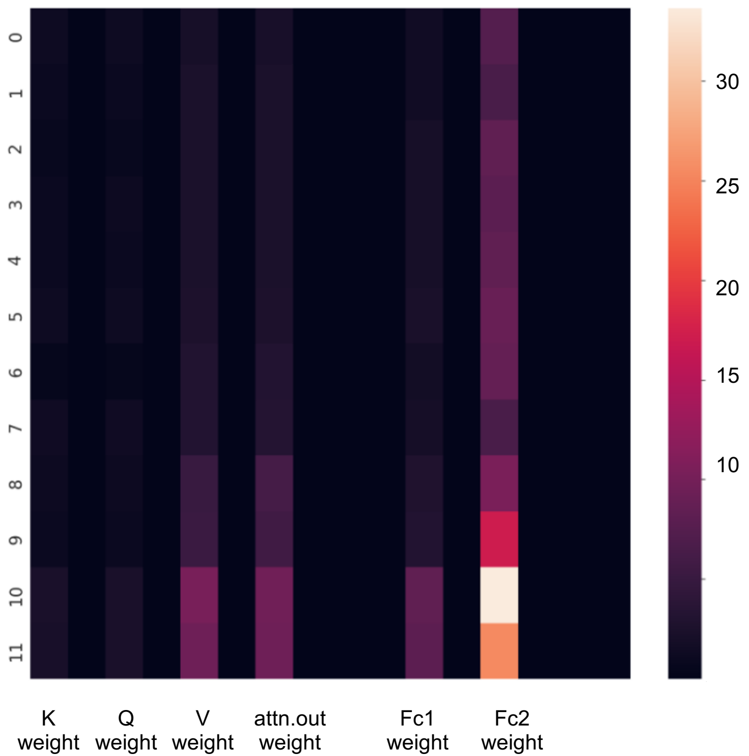
Difference in weight

SF	34.38 M (128)
Fully FT	**0.865**
Weighted sum	0.851

HuBERT base CNN extractor



HuBERT base transformer layer



Analysis - Parameters changed after finetuning

For PR

- Mostly focus on **FC2** of the **7th-11th transformer layers**.
- The CNN extractor is not tuned as much

For SF

- Mostly focus on FC2 of the **10th transformer layer.**
- The parameters of CNN extractor is changed compared with PR

We could analyze the performance of different adapter methods based on these findings.

Adapters - next steps

- Expand experiment settings
  - performance on SUPERB task (ASR, PR, SF, IC, SV)
  - training data size (1hr, 10hr, 100hr)
  - different model (hubert, decoar2, wav2vec)
  - different efficient method (adapter, prompt, weighted sum)
- Explore stability of methods
  - learning rate
  - random seed

Adapters - next steps (cont'd)

- Result interpretation, ablation study
  - Why did adapters work?
  - Does different adapter architectures performs differently on different tasks?
- SLT submissions

Summary

- Various ways to improve prompting performance
  - Prompting on pGSLM
  - Deep prompt tuning
  - Prompt Initialization
- Will explore more ideas for performance
  - Prompting on unit BART or other architecures
  - Prompting w/ adaptor
  - Prompting w/ Verbalizer

Summary (cont'd)

- Investigate various aspects of adapters
  - Performance gain over various up-/down-stream models (w/ limited training data)
  - Will expand exp settings and explore stabilities
  - Will do SLT submissions
- Explore compound topics with other workstreams
  - Robustness of adapters
  - Adapting compressed/multimodal pre-trained networks for better performance
  - …

Timeline

- ~6/18 release of prompting for GSLM and detailed usage (https://github.com/ga642381/SpeechPrompt)
- ~ Workshop begins: release of s3prompt / s3adapter including detailed usage of adding prompt and adapter modules.