FastAdaSP: Multitask-Adapted Efficient Inference for Large Speech Language Model

Yichen Lu*, Jiaqi Song*, Chao-Han Huck Yang, Shinji Watanabe

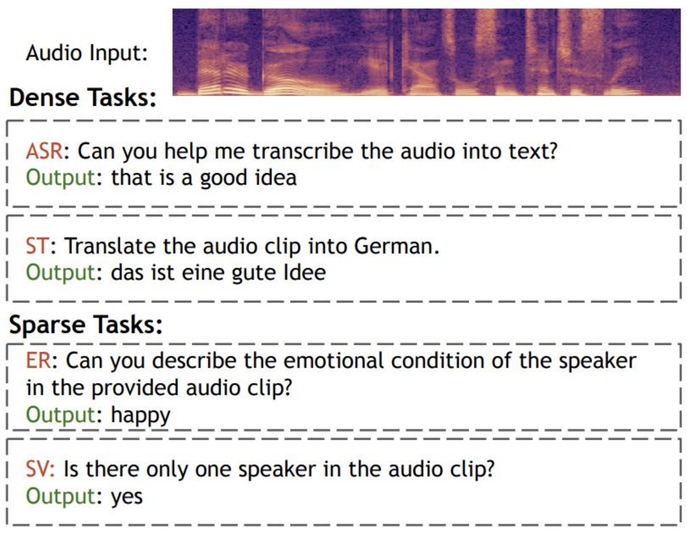
yichenl5@andrew.cmu.edu, jiaqison@andrew.cmu.edu, hucky@nvidia.com, swatanab@andrew.cmu.edu

Carnegie Mellon University Language Technologies Institute

Watanabe's Audio and Voice Lab

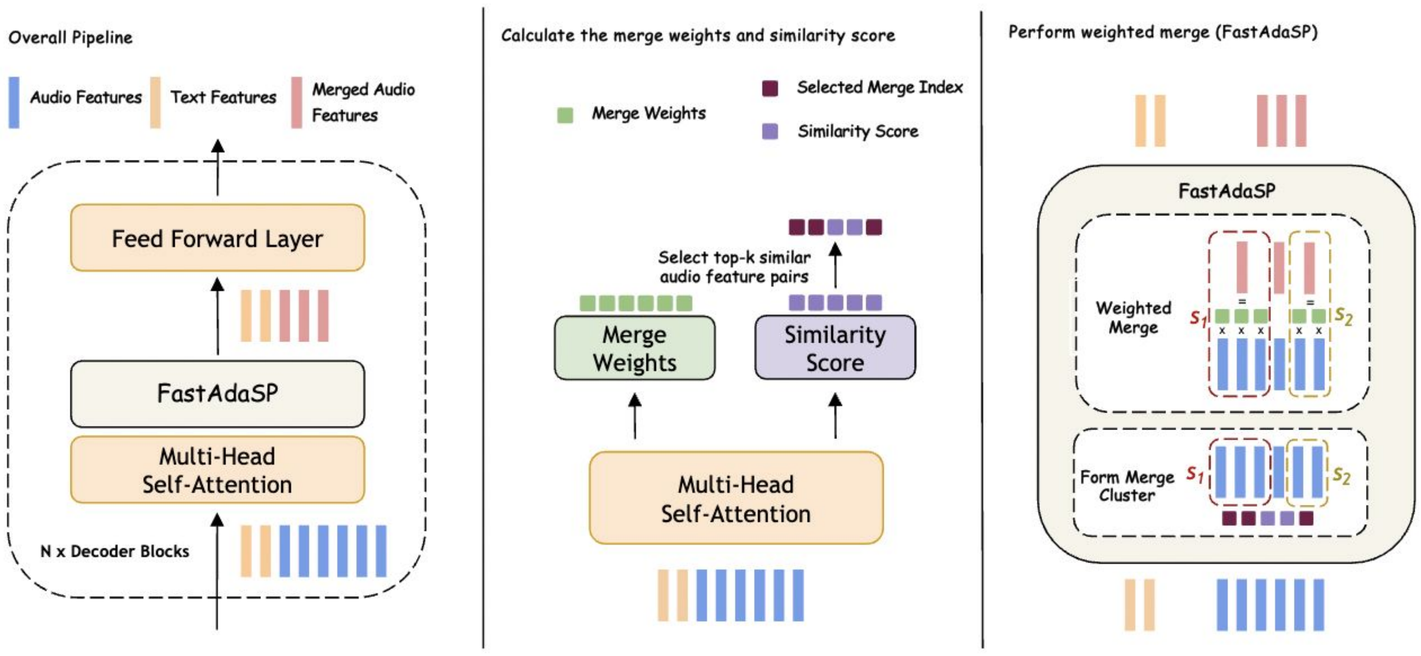
Scan here!

Motivation of FastAdaSP

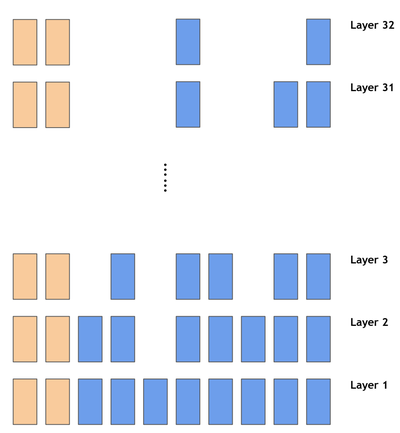


- Large Speech Language Model like GPT-4o have powerful conversational speech processing abilities.
- However, challenges related to inference latency and memory efficiency remain major bottlenecks as the model grows larger.
- Previous methods for optimizing large language model (LLM) inference, such as H2O, cannot universally applicable across all speech or audio-related tasks.
- We want to develop:
  - A fast inference method design for speech modality in SpeenLMs
  - It could adaptively speed up all speech related tasks like dense and sparse tasks
  - It could apply to all SpeechLMs

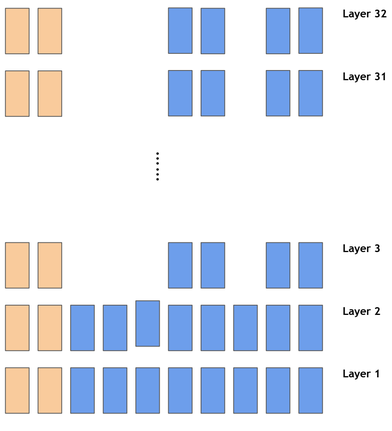
Methodology



Dense Task - Scheduler



Sparse Task - Layer Selection



- For **dense** tasks, we designed an operation scheduler that smoothly merges tokens layer by layer to prevent aggressive token dropping in SpeechLM.
- For **sparse** tasks, we use Transfer Entropy to guide layer selection for token reduction

We use Transfer Entropy to guide layer selection for token reduction;

TE defined as: $|H(\Phi(F_{\text{final}}; \mathbb{W}_{\text{final}})) - H(F_{\text{final}} \mid \Phi(F_i; \mathbb{W}_i))|$

Which:

$\Phi(\cdot;\cdot)$ is the token reduction operation

F is the embedding output

H(.) is the entropy calculation

FLOPs Reduce	TE	TE Rank	10%	20%	30%	40%	50%
Layer 2	2.20	4	54.78	54.30	54.06	52.91	52.10
Layer 9	2.17	3	**55.51**	54.30	53.61	53.30	51.50
Layer 12	2.29	5	54.75	53.96	53.44	52.72	48.35
Layer 15	2.11	2	53.98	54.06	53.02	50.57	-
Layer 3 (Selected)	2.06	1	55.17	**55.05**	**54.40**	**53.86**	**52.14**

Table 6: **Layer Selection Experiments:** Comparison on the performance between different layers on Qwen-Audio ER task (Full token baseline accuracy: 54.80%)

Experiments & Results

- In the performance experiment, FastAdaSP reduces FLOPs by up to **50%** on Qwen-Audio with minimal performance impact in both sparse and dense tasks.
- In the speed experiments, at a 50% reduction ratio, FastAdaSP can achieve a **1.84x** throughput speedup on A100 GPUs.

	ASR (WER% ↓)					ST (BLEU ↑)				
Full Token Baseline	2.21					41.46				
FLOPs Reduce	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Random Merge	2.43	3.39	8.21	27.53	169.96	40.63	39.35	37.01	32.39	24.3
Random Evict	5.70	21.42	61.04	184.59	342.88	38.39	28.22	14.98	6.29	-
A-ToMe (Li et al., 2023)	2.20	3.26	13.91	71.56	273.49	41.24	39.87	36.52	25.35	8.64
FastV (Chen et al., 2024)	12.54	54.40	110.42	179.58	258.78	41.12	40.31	38.45	34.74	27.14
FastAdaSP-Dense Decay Schedule	**2.19**	2.23	2.51	4.37	**15.24**	41.41	41.05	40.51	39.02	35.79
FastAdaSP-Dense Constant Schedule	2.22	**2.21**	**2.30**	**3.57**	16.01	**41.47**	**41.30**	**40.83**	**39.81**	**37.04**

Table 9: Comparison between FastAdaSP with other token reduction methods on Qwen-Audio **dense tasks**

	ER (ACC% ↑)					AC (CIDEr ↑ | SPICE ↑ | SPIDEr ↑)				
Full Token Baseline	54.80					0.45 | 0.13 | 0.29				
FLOPs Reduce	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Random Merge	51.80	48.00	43.80	39.20	32.30	0.44 | 0.13 | 0.29	0.43 | 0.13 | 0.28	0.41 | 0.13 | 0.27	0.41 | 0.12 | 0.26	0.38 | 0.12 | 0.25
Random Evict	52.80	48.20	42.00	34.61	23.14	0.43 | 0.13 | 0.28	0.42 | 0.13 | 0.27	0.38 | 0.12 | 0.25	0.31 | 0.10 | 0.20	0.12 | 0.07 | 0.14
A-ToMe (Li et al., 2023)	54.91	54.70	54.20	**53.90**	51.60	0.44 | 0.13 | 0.29	0.44 | 0.13 | 0.29	0.43 | 0.13 | 0.28	0.41 | 0.13 | 0.27	0.39 | 0.12 | 0.28
FastV (Chen et al., 2024)	54.80	53.80	53.50	52.10	50.38	0.44 | 0.13 | 0.29	**0.45 | 0.13 | 0.29**	0.45 | 0.13 | 0.29	0.44 | 0.13 | 0.28	0.43 | 0.13 | 0.28
FastAdaSP-Sparse	**55.17**	**55.05**	**54.40**	53.86	**52.14**	**0.45 | 0.13 | 0.29**	0.44 | 0.13 | 0.29	**0.45 | 0.13 | 0.29**	**0.44 | 0.13 | 0.28**	**0.43 | 0.13 | 0.28**

Table 10: Comparison between FastAdaSP with other token reduction methods on Qwen-Audio **sparse task**

Beam Size	Audio Length (s)	Token Reduce %	FLOPs Reduction % ↑	Real Time Factor ↓	Pre-filling Latency (s) ↓	Decoding Latency (s) ↓	Throughput (token/s) ↑
1	120	Full Token	0.00	0.054	0.79	5.75	12.86
		50	48.62	0.044	0.77	4.57	13.57 (1.05x)
5	120	Full Token	0.00	0.137	3.11	13.32	5.48
		50	48.40	0.092	3.09	8.01	8.87 (1.61x)
1	240	Full Token	0.00	0.044	1.70	8.90	8.09
		50	49.21	0.036	1.59	7.02	9.69 (1.20x)
5	240	Full Token	0.00	0.126	6.72	23.55	3.10
		50	49.21	0.077	6.48	11.89	5.72 (**1.84x**)

Table 12: **Long Sequence Computational cost experiments on A100.** Long sequence audio samples (120s and 240s) input on WavLLM using one A100 80GB GPU