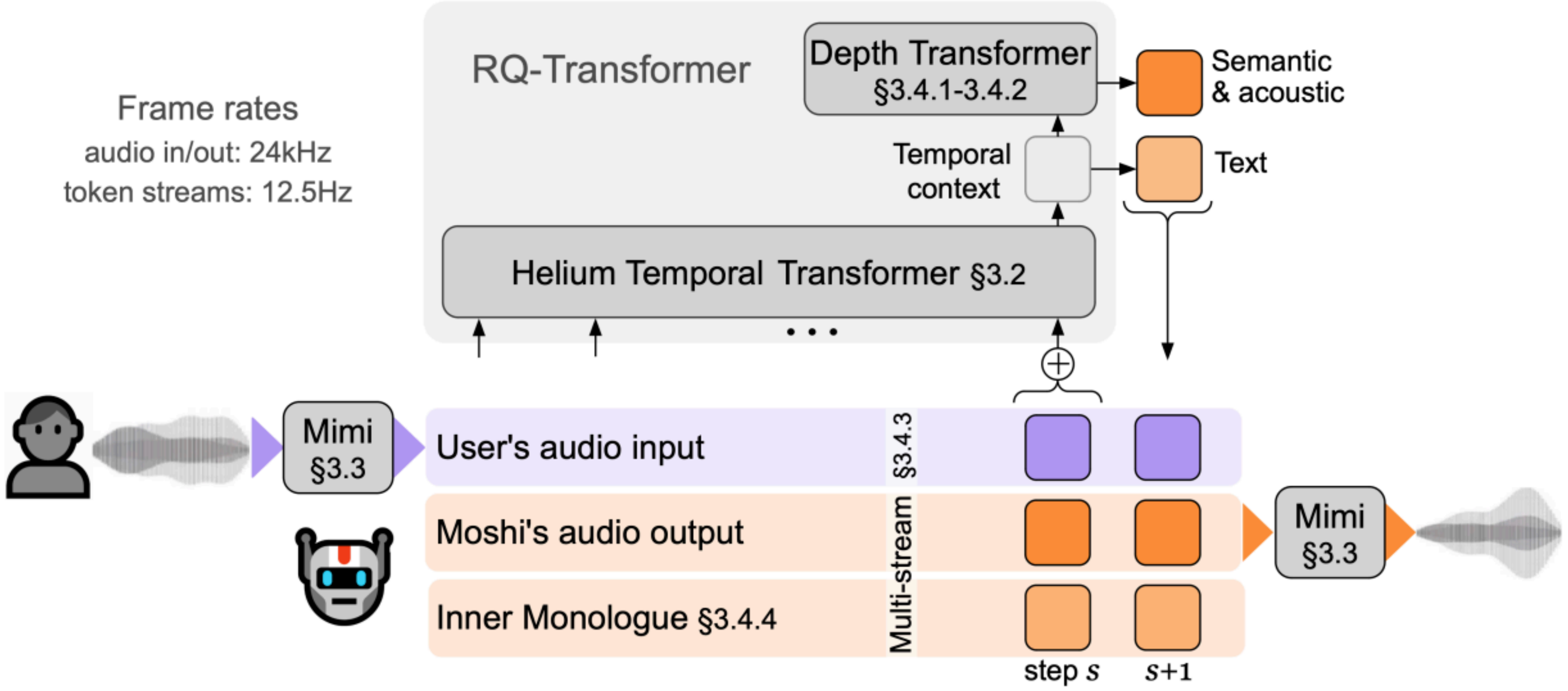
What is full-duplex SLM?

A system that can listen and speak “simultaneously”

- **Modular full-duplex**
  - Using an external orchestrator: FlexDuo, Semantic VAD, etc.
  - Using internal state prediction: FreezeOmni, NeuralFSM, etc.
- **End-to-end full-duplex**
  - Single-stream modeling: SyncLLM, OmniFlatten, SALM-Omni, etc.
  - Multi-stream modeling: dGSLM, **Moshi**, Voila, etc.

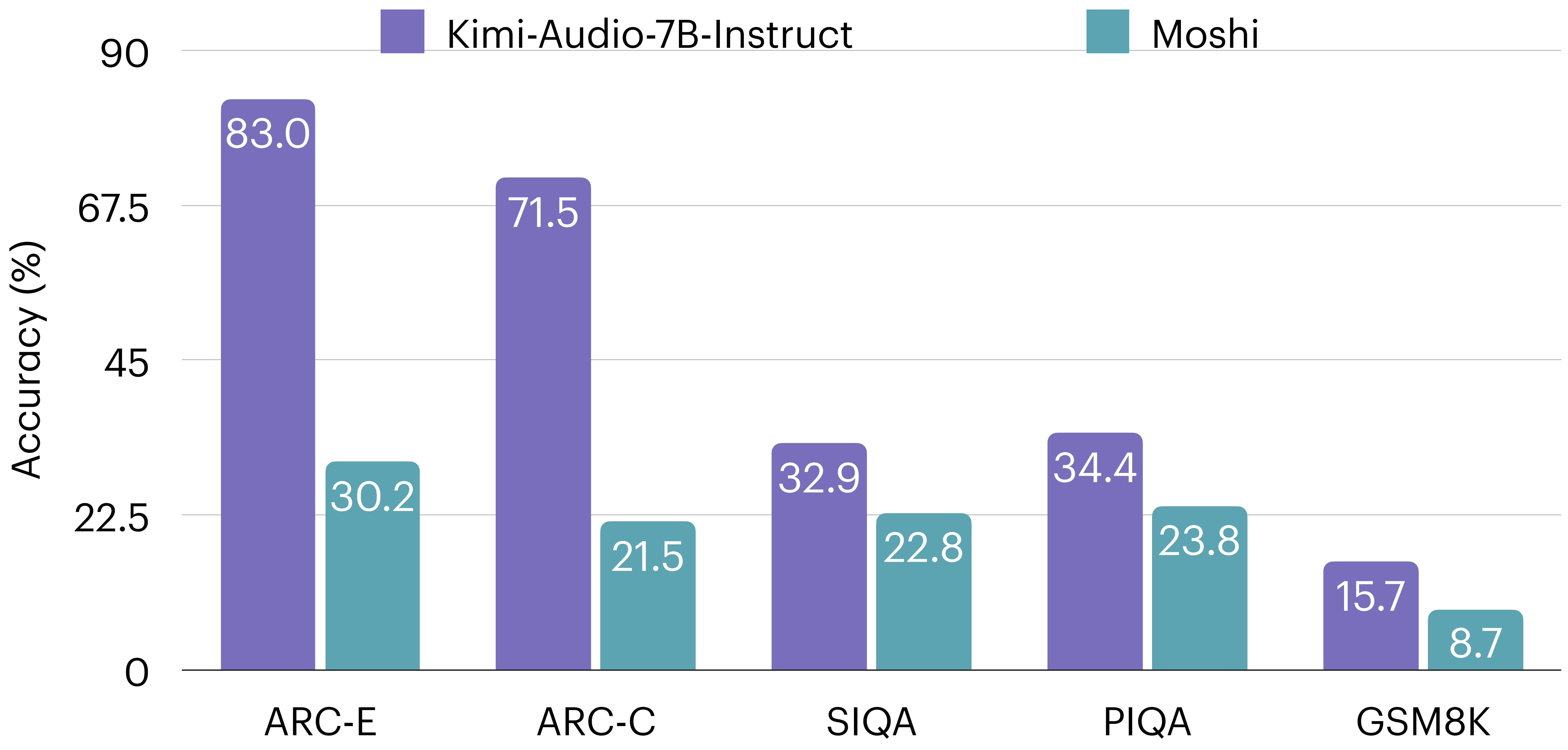
Moshi

Open-source FD model by Kyutai



D'efossez, Alexandre et al. "Moshi: a speech-text foundation model for real-time dialogue." *ArXiv*.

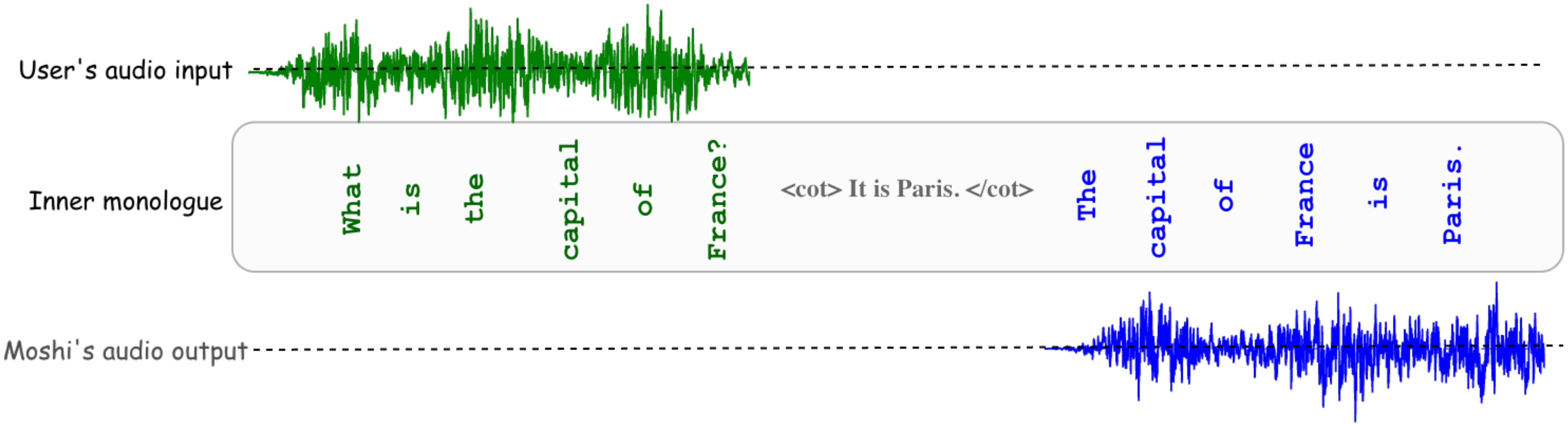
Moshi performs poorly on spoken reasoning tasks



CoT fine-tuning for Moshi

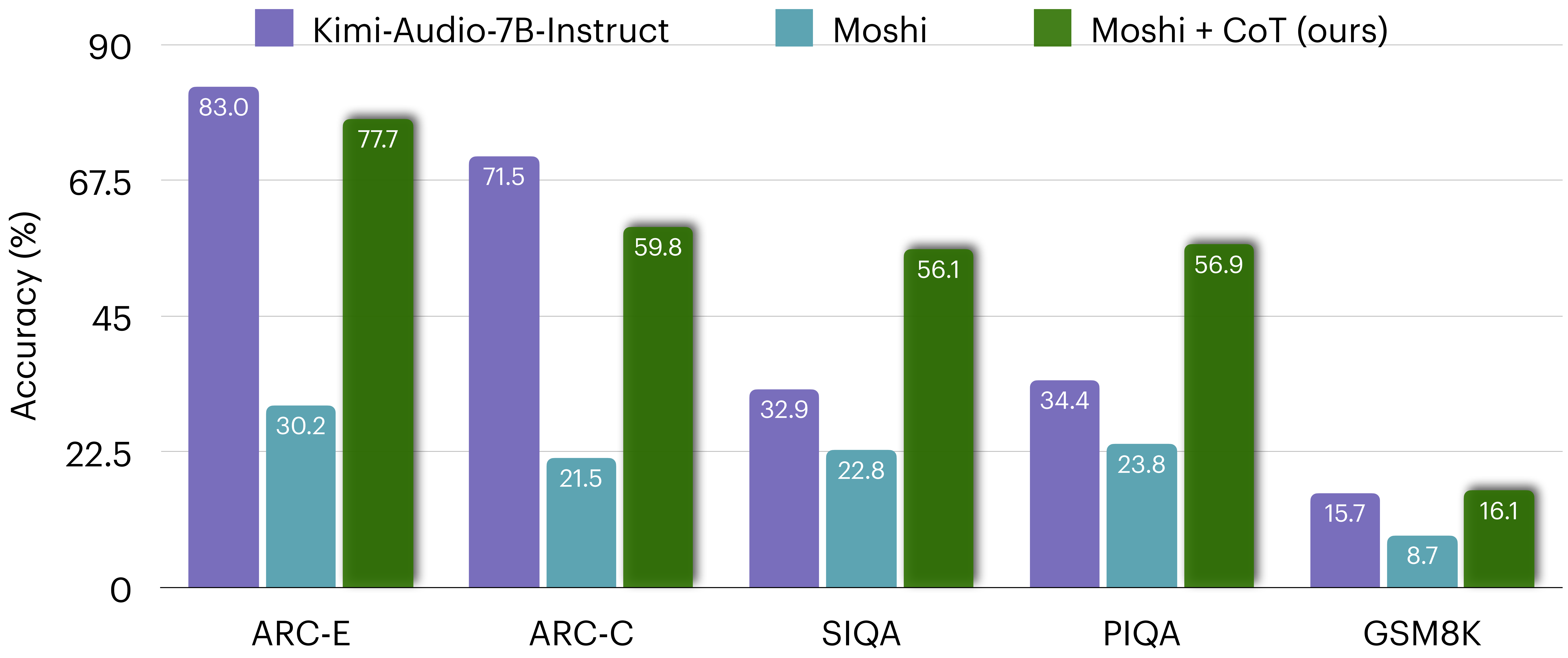
CoT = chain-of-thought

- Fine-tune the model to additionally generate the following on the text monologue channel:
  - **Streaming user audio transcripts**
  - **Chain of thought** (reasoning) between <cot> and </cot>

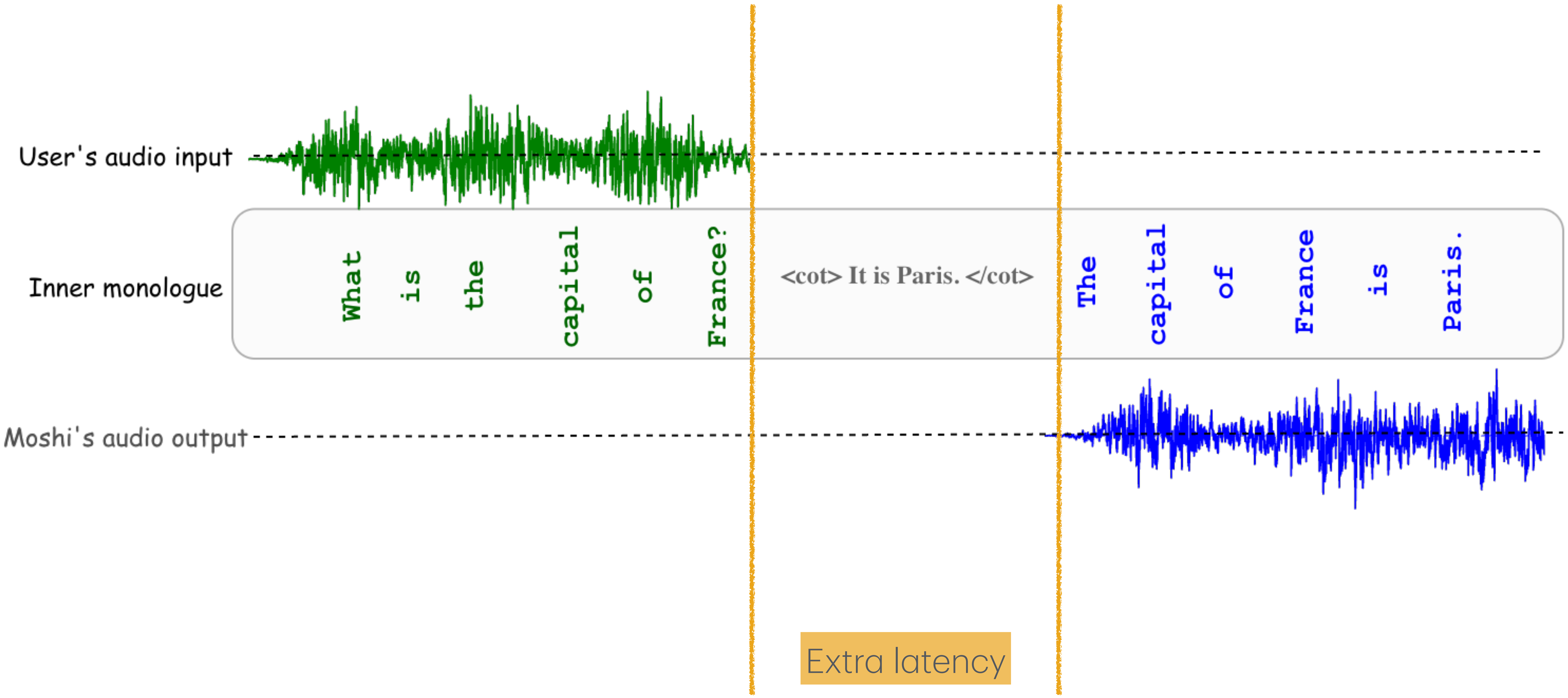


CoT improves reasoning by 2-3x…

We use synthesized CoT-Collection training data



…but it increases response latency!



How can we reduce this latency?

- Two possible approaches:

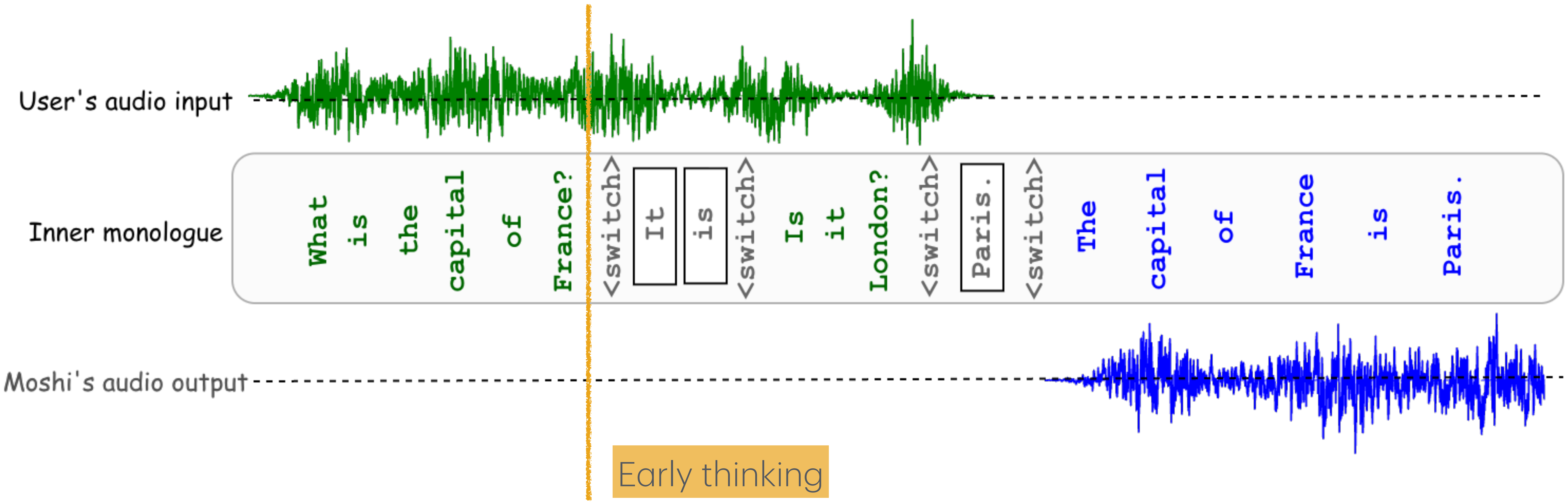
  1. **Thinking while speaking**: interleave thinking with response generation, e.g. Mini-Omni-Reasoner (Xie et al.)

  2. **Thinking while listening**: start reasoning early during user's question!

We train the model to start reasoning early

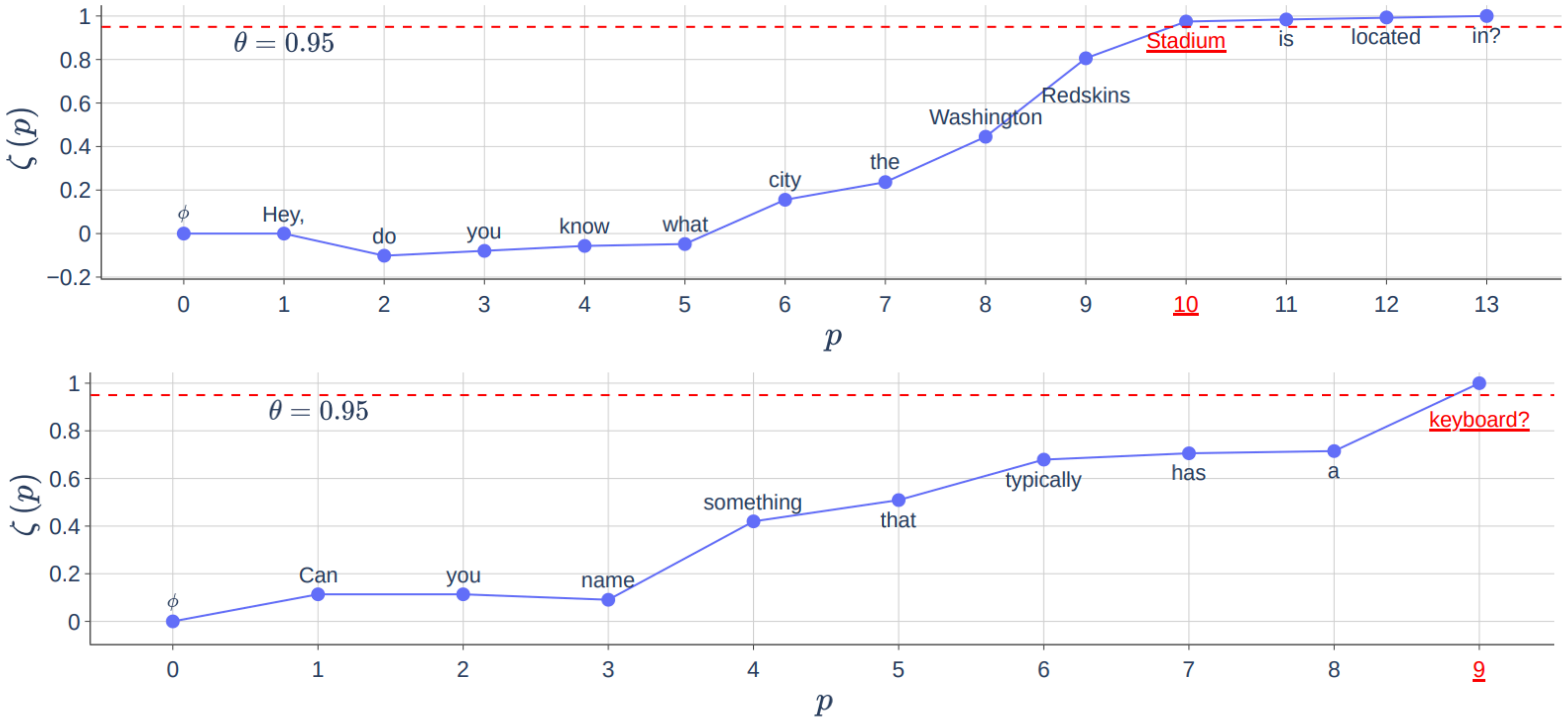
CoT fine-tuning with left shifted tokens

- We use special `<switch>` tokens to interleave the ASR and reasoning tokens.



How much should we shift?

We want to identify an *inflection point* in the question



Question Completeness (QC)

- We use an external LLM (LLaMA3-8B-Chat) to get the probability of the reasoning (**R**) + answer (**A**) after each word of the question, and then compute the **Question Completeness** $\zeta(p)$:

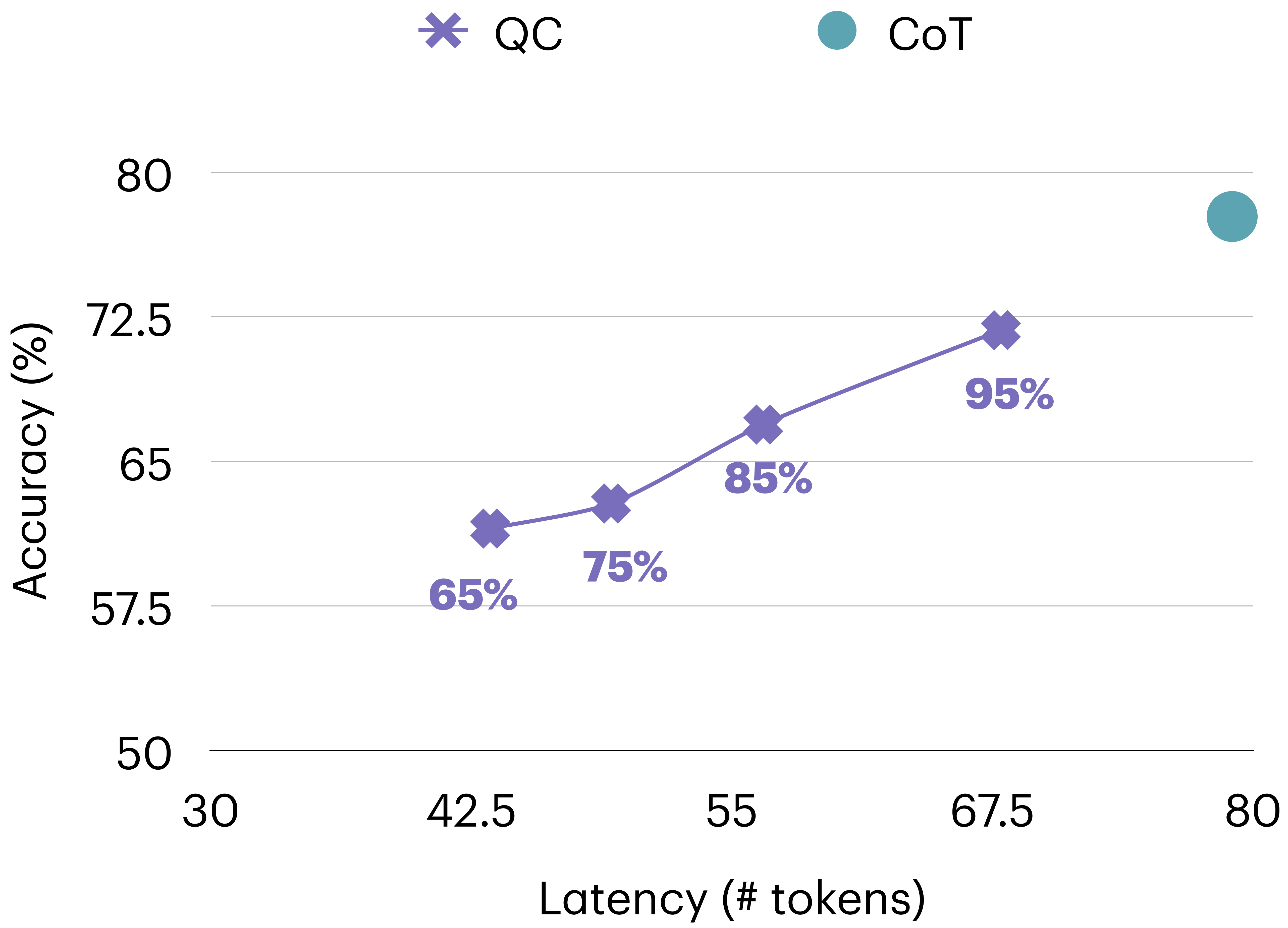
$$\zeta(p) = 1 - \frac{\mathscr{D}_{\mathrm{KL}}(\mathbf{X}_N \,||\, \mathbf{X}_p)}{\mathscr{D}_{\mathrm{KL}}(\mathbf{X}_N \,||\, \mathbf{X}_0)}, \quad \text{where} \quad \mathbf{X}_p = \Pr(\mathbf{R}, \mathbf{A} \,|\, \mathbf{Q}_{0:p})$$

- This quantity measures how "complete" is the question at word p from the point-of-view of generating **R** and **A**.

- The QC curve is monotonically increasing in p.

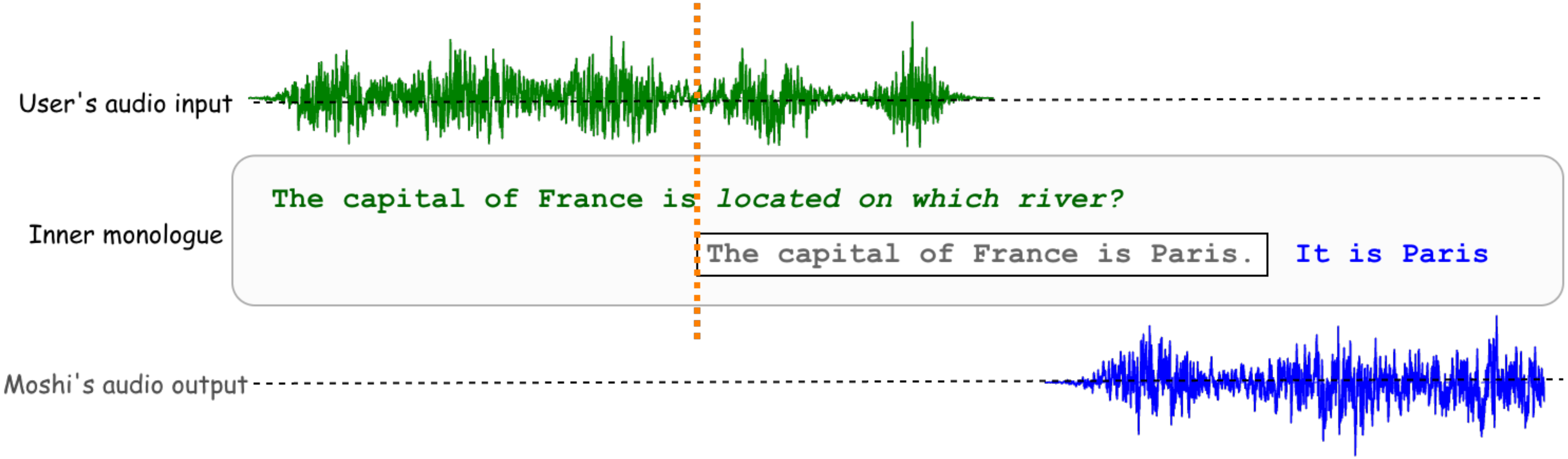
Training with different QC thresholds

Accuracy v/s latency on ARC-E



Why does accuracy degrade?

- The model is not trained to consider new incoming information after it starts reasoning!



We use DPO to teach *adaptive* reasoning

Preference pairs are created using rejection sampling

$$\mathscr{L}_{\mathrm{DPO}}(\pi_\theta) = -\,\mathbb{E}_{(x,y^+,y^-)\sim\mathscr{D}}\left[\log\sigma\left(\beta\left[\log\frac{\pi_\theta(y^+\mid x)}{\pi_{\mathrm{ref}}(y^+\mid x)} - \log\frac{\pi_\theta(y^-\mid x)}{\pi_{\mathrm{ref}}(y^-\mid x)}\right]\right)\right]$$

1. Train SFT model π_{ref} with QC = 75%.
2. Generate 10 outputs per prompt (x), where we force the model to decode `<start_cot>` at QC = 75%.
3. Select a response with correct answer as y^+, and one with wrong answer as y^-.

Accuracy improves significantly after DPO