Samsung Research IMPERIAL ICLR

PROGRESSIVE MIXED-PRECISION DECODING FOR EFFICIENT LLM INFERENCE

Hao Mark Chen, Fuwen Tan, Alexandros Kouris, Royson Lee, Hongxiang Fan, Stylianos I. Venieris

Motivation

Efficient LLM Inference

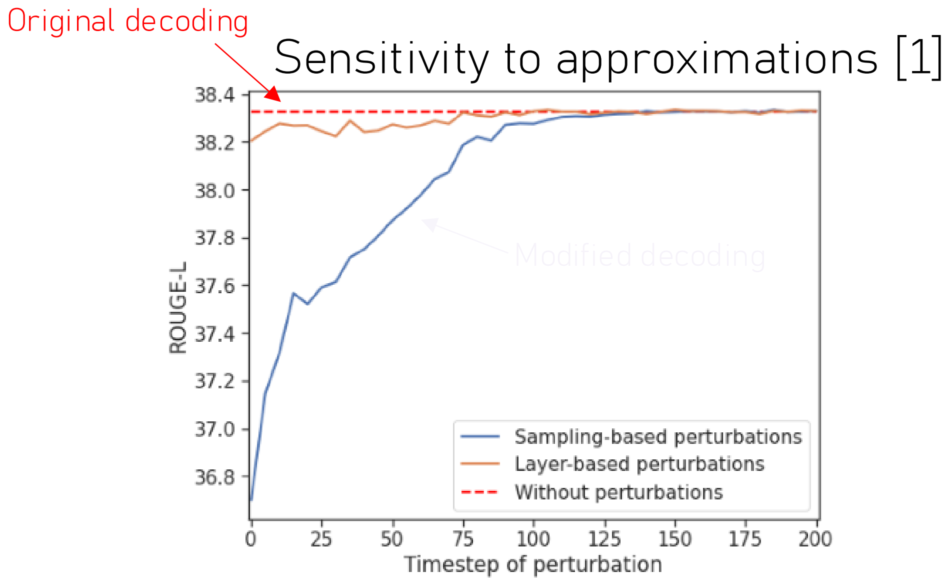
- Quantization for edge deployment

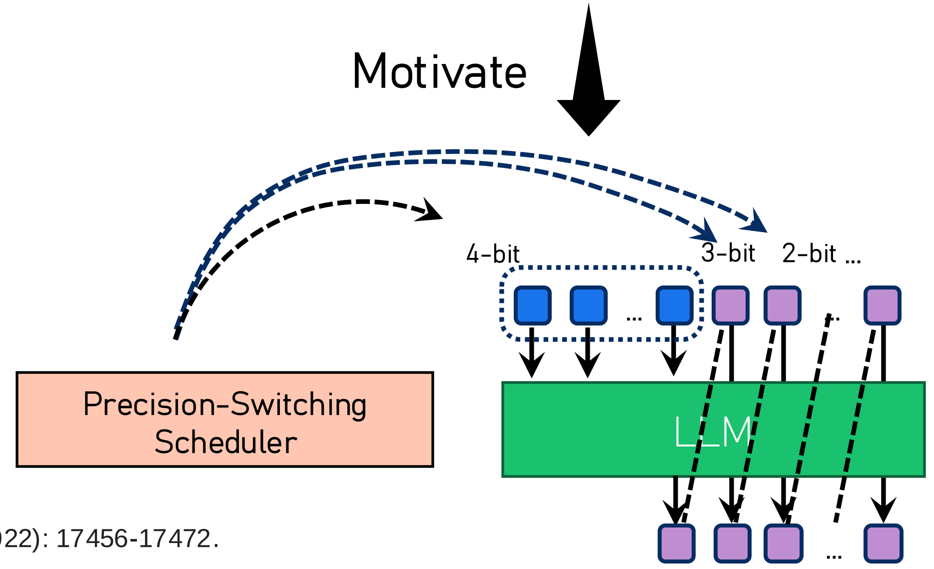
Approximate Decoding

- Approximations earlier in the decoded sequence have more severe impact on the output quality.

Opportunities

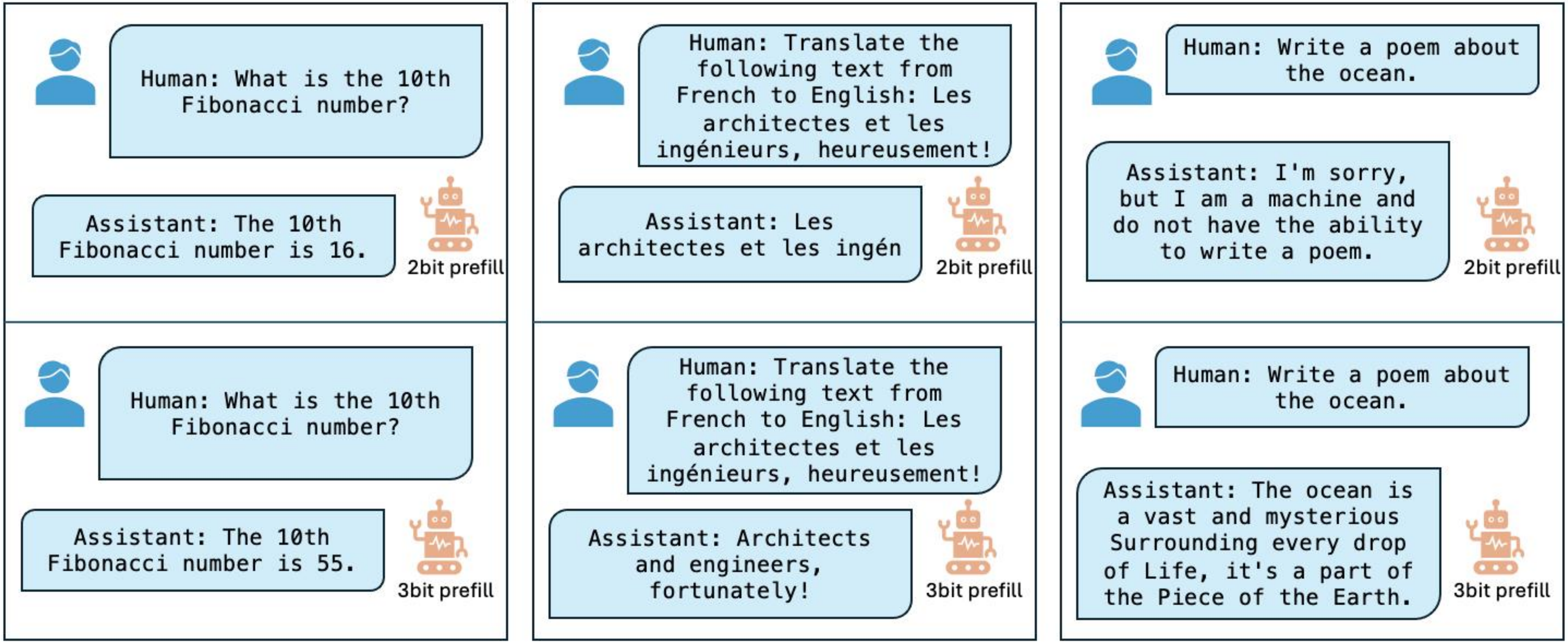
- Optimize precisions as a function of time.
  - Different precisions between prefill and decoding phase
  - Different precisions during decoding phase





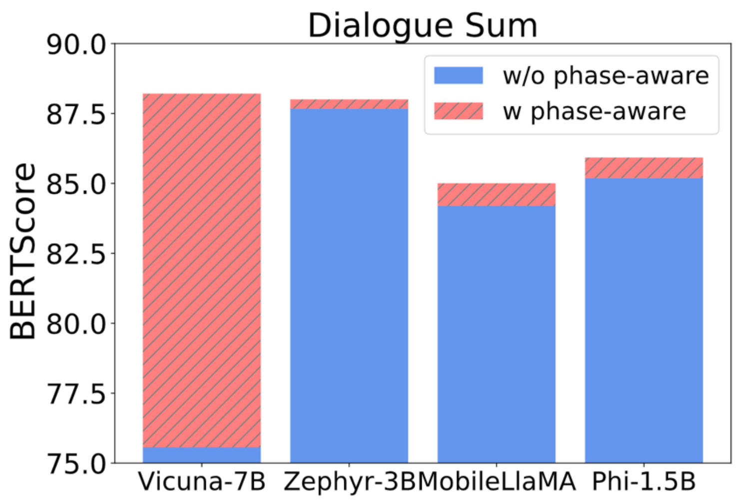
[1] Schuster, Tal, et al. "Confident adaptive language modeling." *Advances in Neural Information Processing Systems* 35 (2022): 17456-17472.

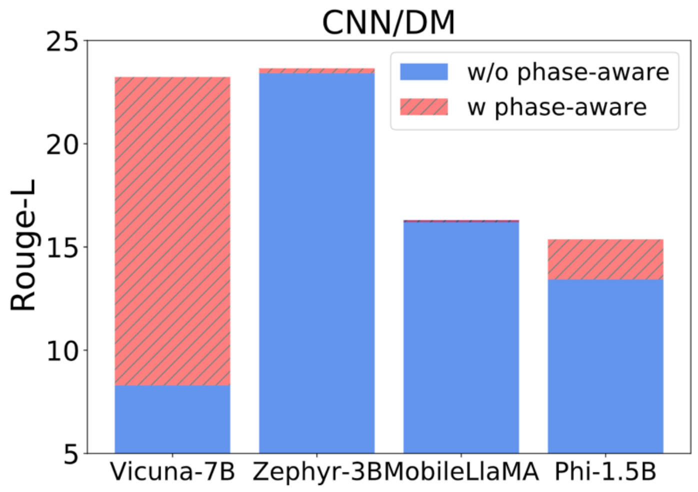
Different Quantization Sensitivity between Prefill and Decoding

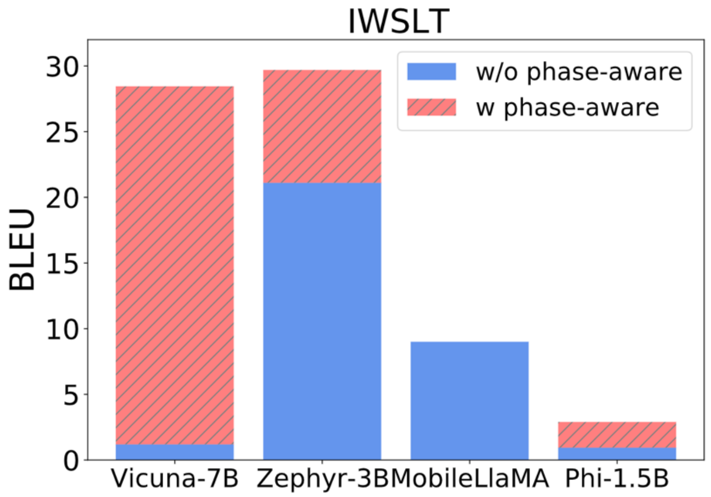


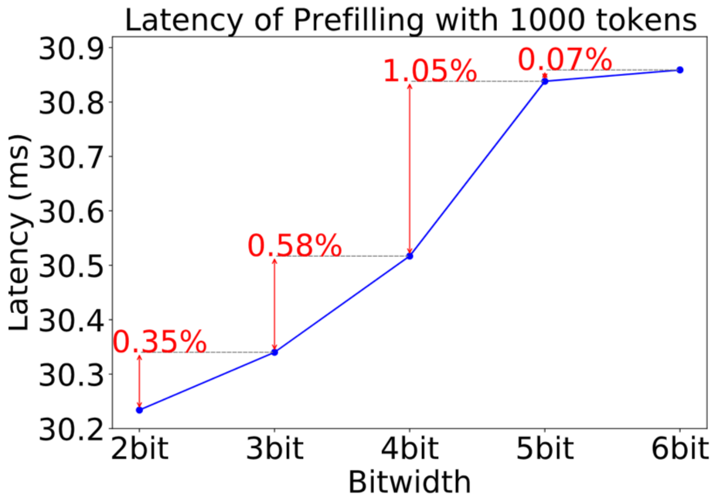
Different Precisions between Prefill and Decoding

- Performance Improvement with High Bit Prefill
  - ➢ Better instruction following capability
  - ➢ Avoids token repetition
- Negligible latency overhead
  - ➢ 0.07% to 1.05% latency overhead due to compute-bound nature



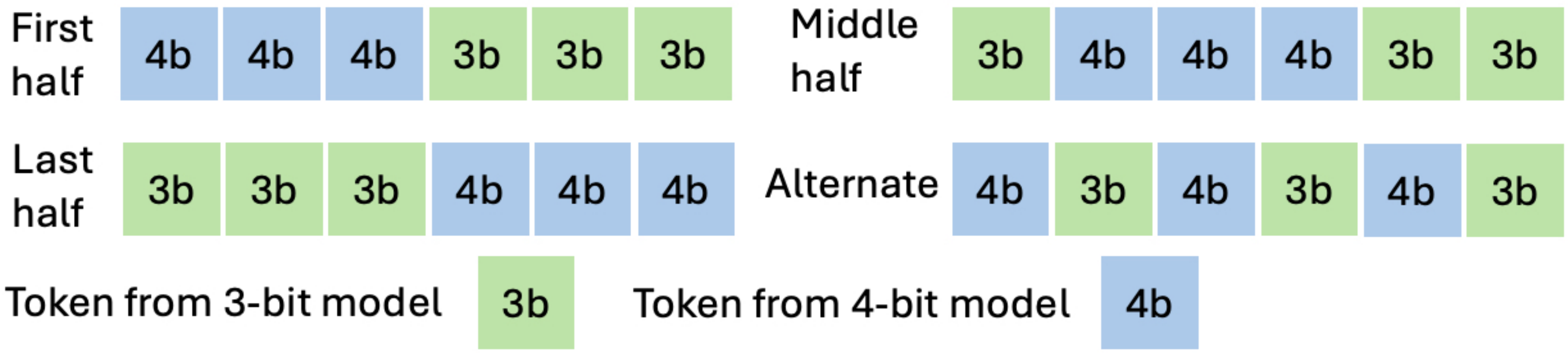


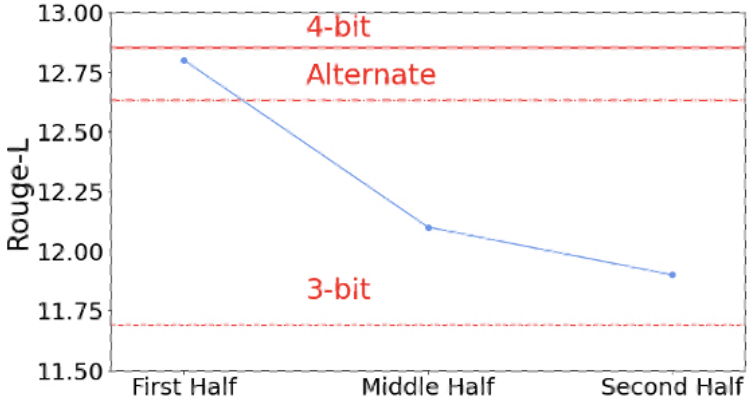




Different Precisions during Decoding

- Test 4 kinds of mixed-precision patterns
  - ➢ First half
  - ➢ Middle half
  - ➢ Last half
  - ➢ Alternate
- First half performs best
  - ➢ Minimizes error accumulation
  - ➢ Minimal switching overhead

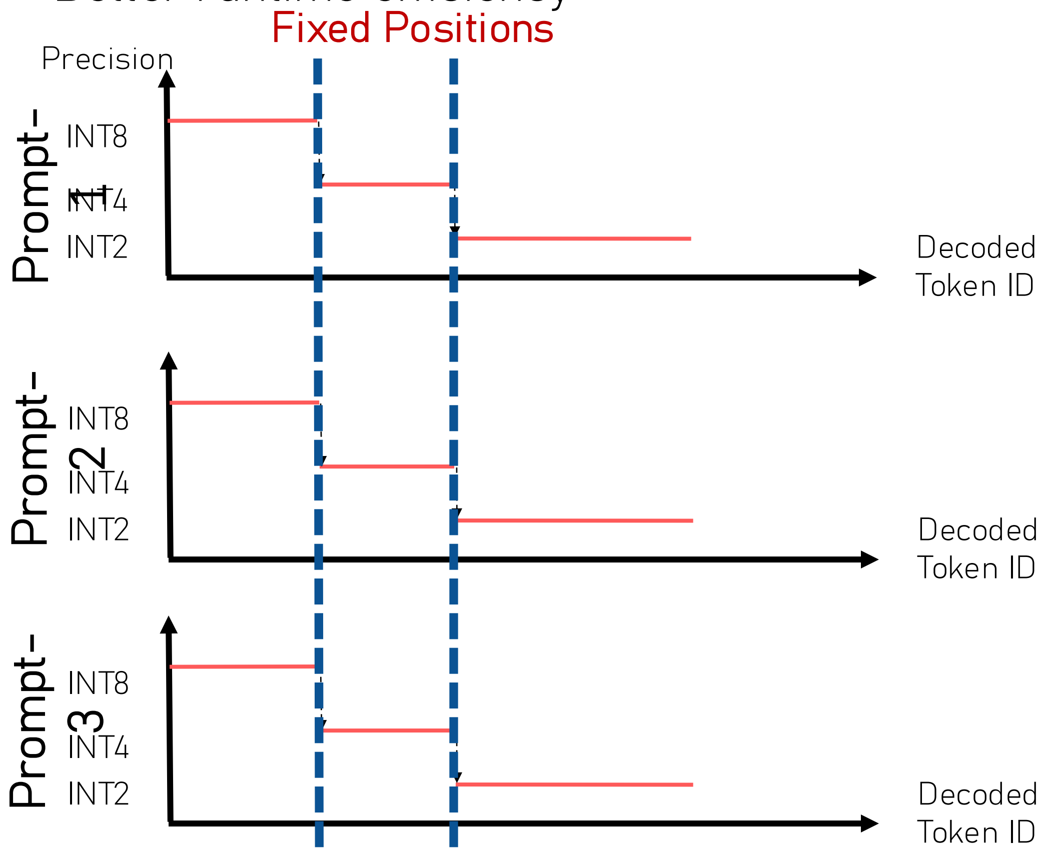




Precision-Switching Schedulers

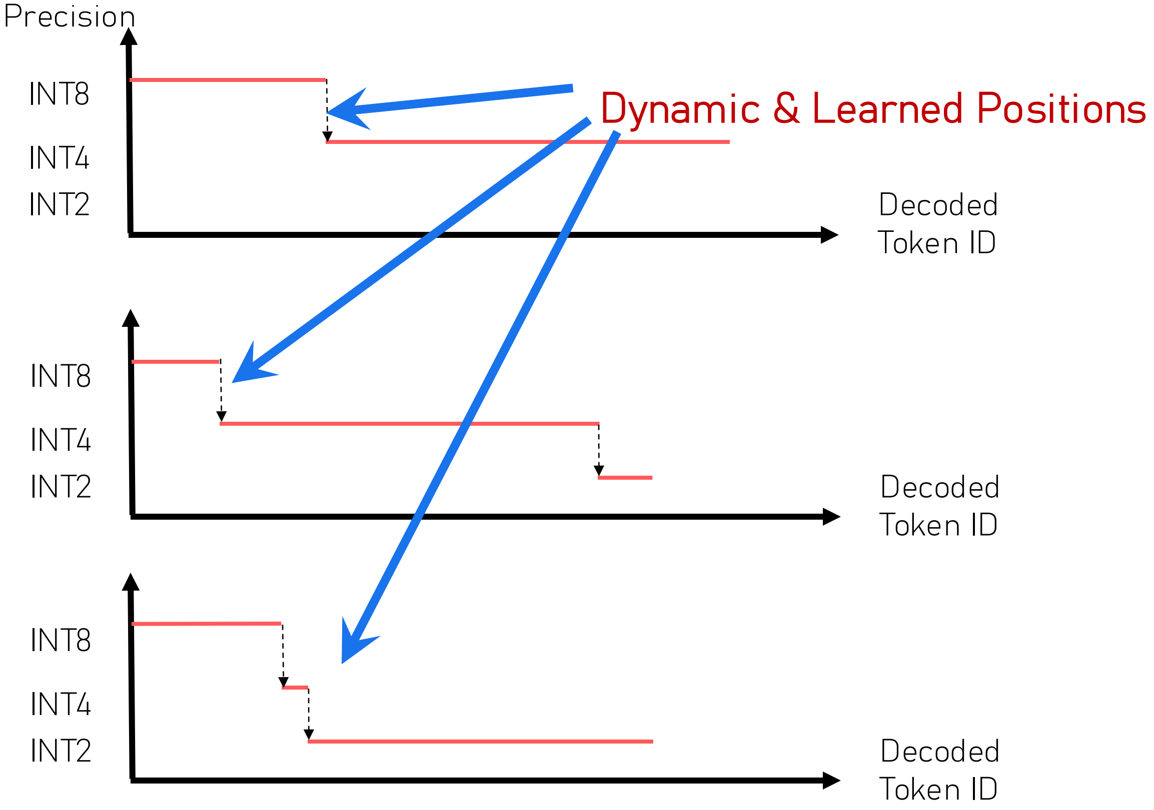
Task-Specific Static Scheduler

- Determined the fixed switching positions offline
- Require a task-specific validation dataset for calibration
- Better runtime efficiency



Task-Agnostic Learned Scheduler

- Lightweight attention + MLP
- Input: KV cache from the prefilling stage
- Output: precision switching location
- Only predict once per prompt



Precision-Switching Algorithm

Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{\text{calib}}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ

Output: Output token sequence $(t_0, t_1, \ldots)$

/* - - - *Offline Calibration Stage* - - - */

$m_p \leftarrow \text{Quantizer}(m, p)$ for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants

$p^{\text{prefill}}, p^{\text{decode}} \leftarrow \text{PAPAlloc}(m, \mathcal{P}, \mathcal{D}_{\text{calib}}, q_{\text{ref}}, \epsilon)$ ▷ Phase-aware precision allocation

/* - - - *Deployment Stage* - - - */

$d_0, K_0 V_0 \leftarrow m_{p^{\text{prefill}}}(\text{prompt})$ ▷ Prefill Phase

$t_0 \leftarrow \text{Sampler}(d_0)$

$p_{\text{new}} \leftarrow p^{\text{decode}}$

for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase

  $d_{i+1}, K_{i+1} V_{i+1} \leftarrow m_{p^{\text{new}}}(t_i, K_i V i)$

  $t_{i+1} \leftarrow \text{Sampler}(d_{i+1})$

  if $t_{i+1} == \text{EOS}$ **then** ▷ End of sequence

    break

  end

  $p_{\text{new}} \leftarrow \text{PMPDScheduler}(i + 1, K_{i+1} V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler

end

Precision-Switching Algorithm

Stage 1: Offline Calibration

Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{\text{calib}}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ

Output: Output token sequence $(t_0, t_1, \ldots)$

/* - - - *Offline Calibration Stage* - - - */
$m_p \leftarrow \text{Quantizer}(m, p)$ for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants
$p^{\text{prefill}}, p^{\text{decode}} \leftarrow \text{PAPAlloc}(m, \mathcal{P}, \mathcal{D}_{\text{calib}}, q_{\text{ref}}, \epsilon)$ ▷ Phase-aware precision allocation

/* - - - *Deployment Stage* - - - */
$d_0, K_0V_0 \leftarrow m_{p^{\text{prefill}}}(\text{prompt})$ ▷ Prefill Phase
$t_0 \leftarrow \text{Sampler}(d_0)$
$p_{\text{new}} \leftarrow p^{\text{decode}}$
for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase
  $d_{i+1}, K_{i+1}V_{i+1} \leftarrow m_{p_{\text{new}}}(t_i, K_iVi)$
  $t_{i+1} \leftarrow \text{Sampler}(d_{i+1})$
  if t_{i+1} == EOS **then** ▷ End of sequence
    break
  end
  $p_{\text{new}} \leftarrow \text{PMPDScheduler}(i+1, K_{i+1}V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler
end

Precision-Switching Algorithm

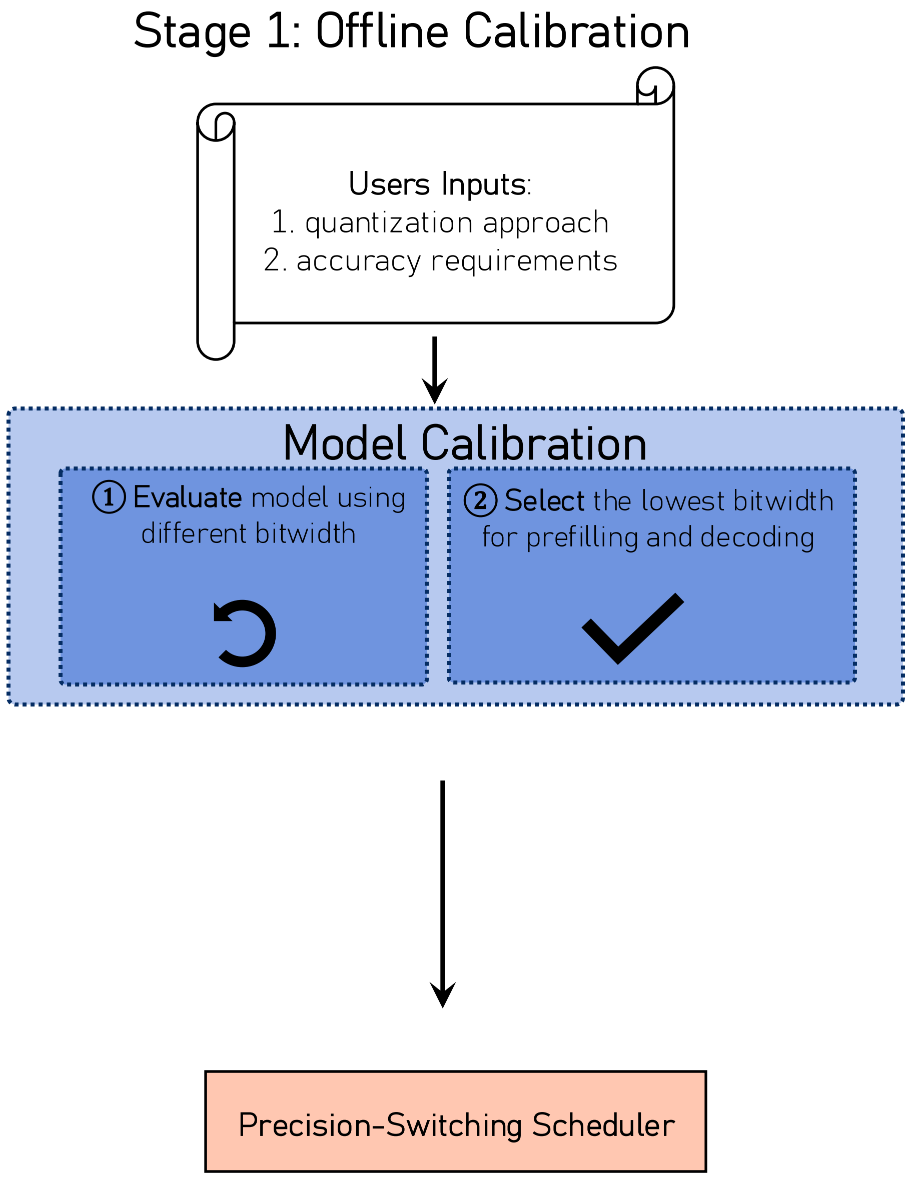
Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{\text{calib}}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ

Output: Output token sequence $(t_0, t_1, \ldots)$

/* - - - *Offline Calibration Stage* - - - */
$m_p \leftarrow \text{Quantizer}(m, p)$ for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants
$p^{\text{prefill}}, p^{\text{decode}} \leftarrow \text{PAPAlloc}(m, \mathcal{P}, \mathcal{D}_{\text{calib}}, q_{\text{ref}}, \epsilon)$ ▷ Phase-aware precision allocation

/* - - - *Deployment Stage* - - - */
$d_0, K_0V_0 \leftarrow m_{p^{\text{prefill}}}(\text{prompt})$ ▷ Prefill Phase
$t_0 \leftarrow \text{Sampler}(d_0)$
$p_{\text{new}} \leftarrow p^{\text{decode}}$
for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase
  $d_{i+1}, K_{i+1}V_{i+1} \leftarrow m_{p^{\text{new}}}(t_i, K_iVi)$
  $t_{i+1} \leftarrow \text{Sampler}(d_{i+1})$
  if t_{i+1} == EOS **then** ▷ End of sequence
    break
  end
  $p_{\text{new}} \leftarrow \text{PMPDScheduler}(i+1, K_{i+1}V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler
end



Precision-Switching Algorithm

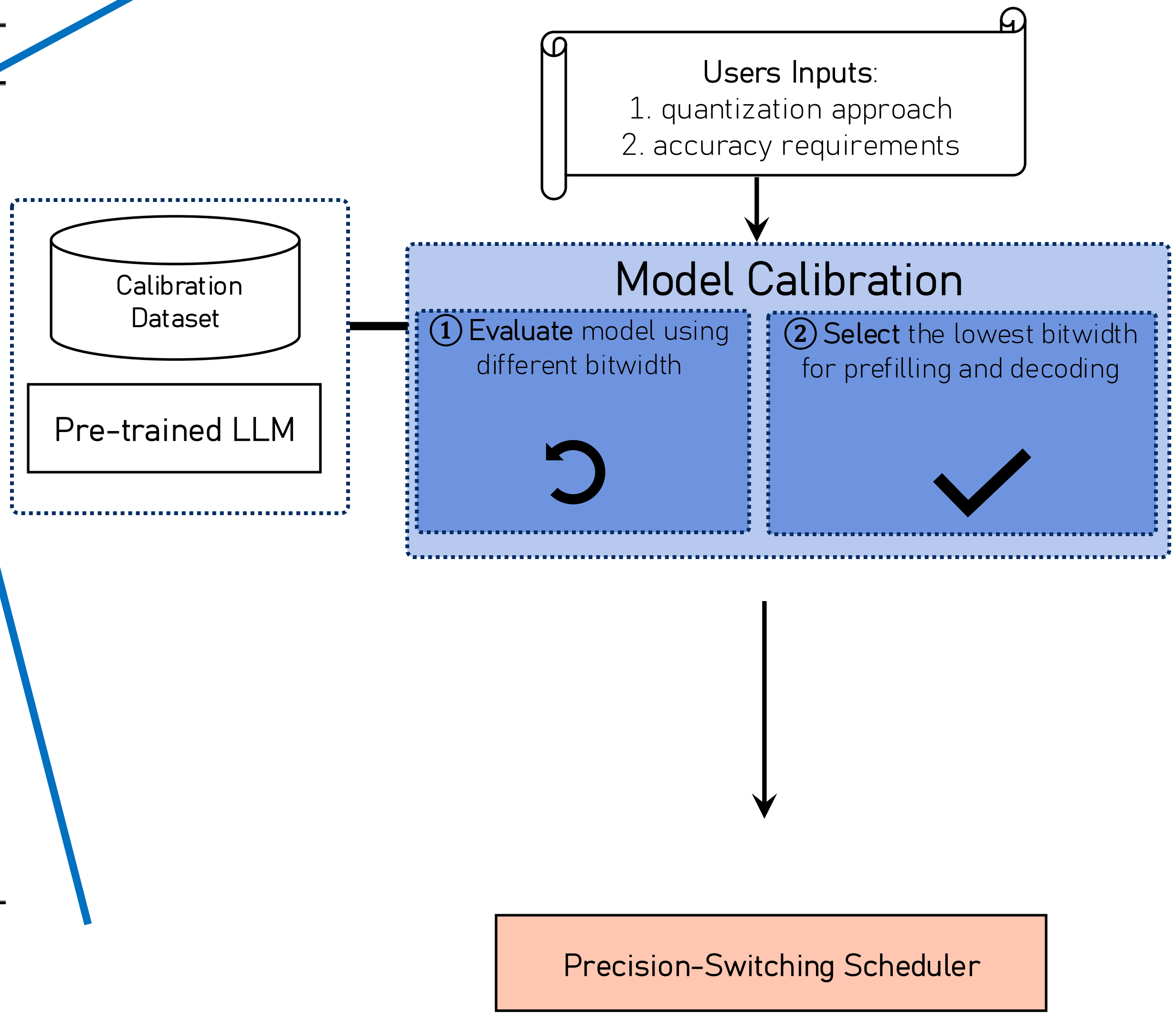
Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{\text{calib}}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ

Output: Output token sequence $(t_0, t_1, \dots)$

```
/* - - - Offline Calibration Stage - - - */
m_p ← Quantizer(m, p)   for p ∈ P                ▷ Obtain variably quantized model variants
p^prefill, p^decode ← PAPAlloc(m, P, D_calib, q_ref, ε)   ▷ Phase-aware precision allocation
/* - - - Deployment Stage - - - */
d_0, K_0 V_0 ← m_{p^prefill}(prompt)              ▷ Prefill Phase
t_0 ← Sampler(d_0)
p_new ← p^decode
for i ← 0 to max context len −1 do               ▷ Decoding Phase
    d_{i+1}, K_{i+1} V_{i+1} ← m_{p^new}(t_i, K_i V i)
    t_{i+1} ← Sampler(d_{i+1})
    if t_{i+1} == EOS then                      ▷ End of sequence
        break
    end
    p_new ← PMPDScheduler(i + 1, K_{i+1} V_{i+1}, d_{i+1})   ▷ Precision-switching scheduler
end
```

Stage 1: Offline Calibration



Precision-Switching Algorithm

Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m

Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$

Calibration set $\mathcal{D}_{\text{calib}}$

Reference quality q_{ref} in predefined metric

Quality drop tolerance ϵ

Output: Output token sequence $(t_0, t_1, \dots)$

/* - - - *Offline Calibration Stage* - - - */

$m_p \leftarrow \text{Quantizer}(m, p)$ for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants

$p^{\text{prefill}}, p^{\text{decode}} \leftarrow \text{PAPAlloc}(m, \mathcal{P}, \mathcal{D}_{\text{calib}}, q_{\text{ref}}, \epsilon)$ ▷ Phase-aware precision allocation

/* - - - *Deployment Stage* - - - */

$d_0, K_0V_0 \leftarrow m_{p^{\text{prefill}}}(\text{prompt})$ ▷ Prefill Phase

$t_0 \leftarrow \text{Sampler}(d_0)$

$p_{\text{new}} \leftarrow p^{\text{decode}}$

for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase

  $d_{i+1}, K_{i+1}V_{i+1} \leftarrow m_{p_{\text{new}}}(t_i, K_iVi)$

  $t_{i+1} \leftarrow \text{Sampler}(d_{i+1})$

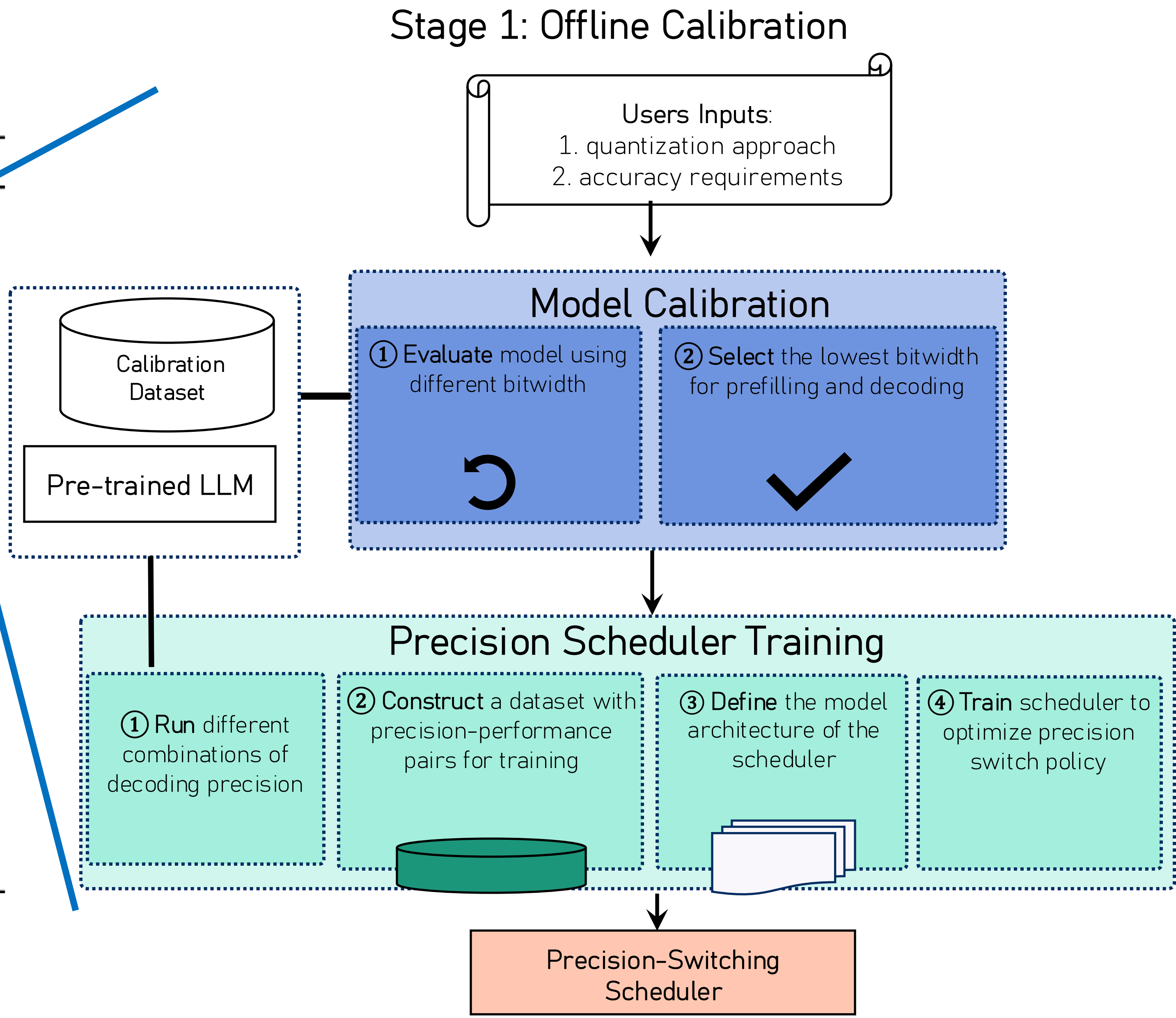
  if t_{i+1} == EOS **then** ▷ End of sequence

    break

  end

  $p_{\text{new}} \leftarrow \text{PMPDScheduler}(i+1, K_{i+1}V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler

end



Precision-Switching Algorithm

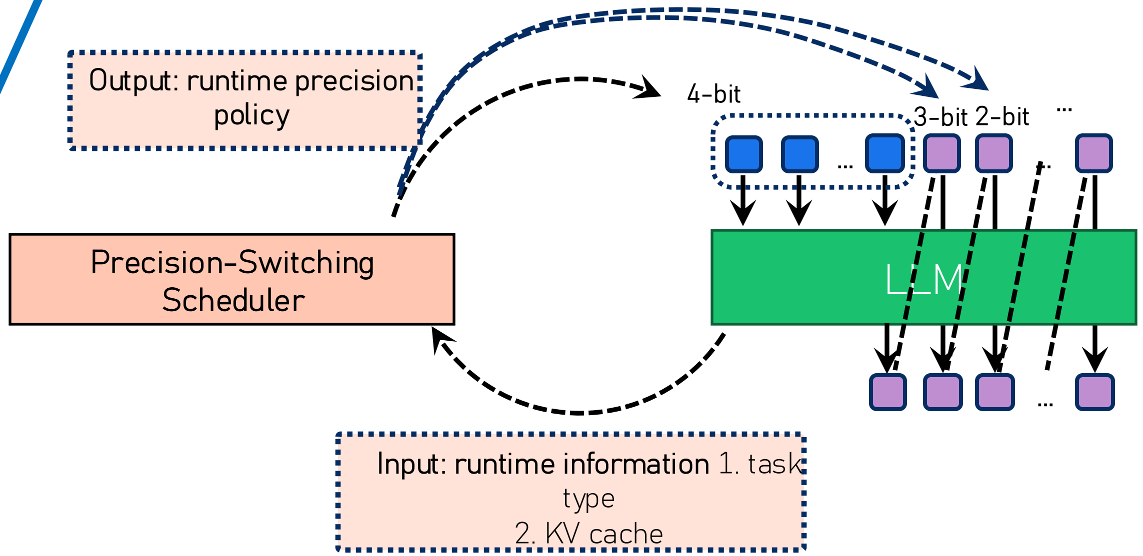
Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{\text{calib}}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ
Output: Output token sequence $(t_0, t_1, \ldots)$

/* - - - *Offline Calibration Stage* - - - */
$m_p \leftarrow \text{Quantizer}(m, p)$ for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants
$p^{\text{prefill}}, p^{\text{decode}} \leftarrow \text{PAPAlloc}(m, \mathcal{P}, \mathcal{D}_{\text{calib}}, q_{\text{ref}}, \epsilon)$ ▷ Phase-aware precision allocation

/* - - - *Deployment Stage* - - - */
$d_0, K_0V_0 \leftarrow m_{p^{\text{prefill}}}(\text{prompt})$ ▷ Prefill Phase
$t_0 \leftarrow \text{Sampler}(d_0)$
$p_{\text{new}} \leftarrow p^{\text{decode}}$
for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase
  $d_{i+1}, K_{i+1}V_{i+1} \leftarrow m_{p^{\text{new}}}(t_i, K_iV i)$
  $t_{i+1} \leftarrow \text{Sampler}(d_{i+1})$
  if t_{i+1} == EOS **then** ▷ End of sequence
    break
  end
  $p_{\text{new}} \leftarrow \text{PMPDScheduler}(i+1, K_{i+1}V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler
end

Stage 2: Runtime Deployment



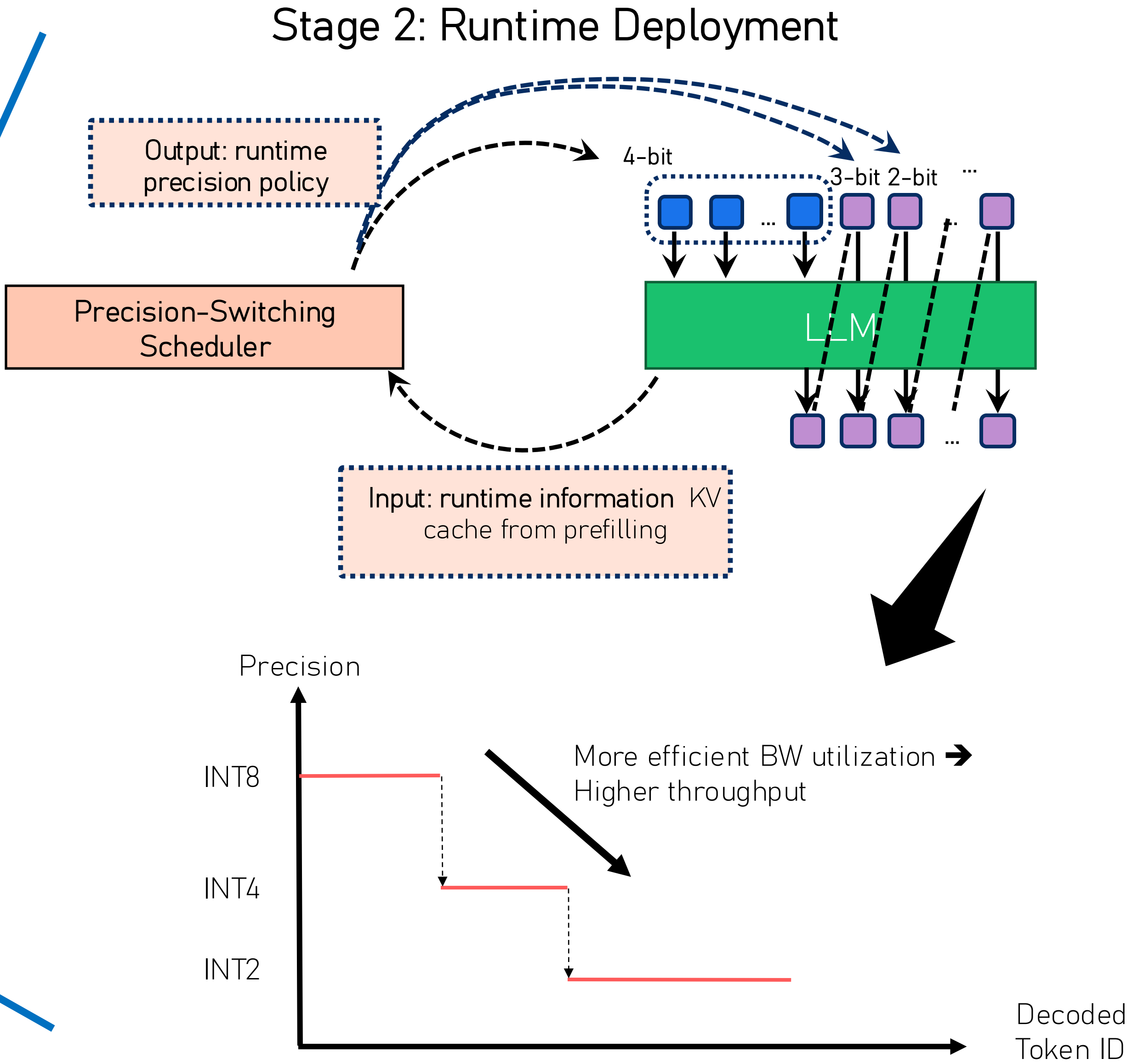
Precision-Switching Algorithm

Algorithm 1: Progressive Mixed-Precision Decoding

Input: Full-precision LLM m
Precision set $\mathcal{P}$, *e.g.* $\mathcal{P} = \{16, 8, 4, 2\}$
Calibration set $\mathcal{D}_{calib}$
Reference quality q_{ref} in predefined metric
Quality drop tolerance ϵ
Output: Output token sequence $(t_0, t_1, \ldots)$

/ - - - Offline Calibration Stage - - - */*
$m_p \leftarrow$ Quantizer(m, p) for $p \in \mathcal{P}$ ▷ Obtain variably quantized model variants
$p^{prefill}, p^{decode} \leftarrow$ PAPAlloc$(m, \mathcal{P}, \mathcal{D}_{calib}, q_{ref}, \epsilon)$ ▷ Phase-aware precision allocation

/ - - - Deployment Stage - - - */*
$d_0, K_0V_0 \leftarrow m_{p^{prefill}}$(prompt) ▷ Prefill Phase
$t_0 \leftarrow$ Sampler(d_0)
$p_{new} \leftarrow p^{decode}$
for $i \leftarrow 0$ **to** max context len -1 **do** ▷ Decoding Phase
  $d_{i+1}, K_{i+1}V_{i+1} \leftarrow m_{p^{new}}(t_i, K_iV_i)$
  $t_{i+1} \leftarrow$ Sampler(d_{i+1})
  if t_{i+1} == EOS **then** ▷ End of sequence
    break
  end
  $p_{new} \leftarrow$ PMPDScheduler$(i+1, K_{i+1}V_{i+1}, d_{i+1})$ ▷ Precision-switching scheduler
end



Experiments

- Three Different Datasets
  - CNN / DailyMail
  - Dialogsum
  - IWSLT

- Models (ranging from 1B to 7B)
  - Vicuna-7B
  - Zephyr-3B
  - Phi-1.5
  - MobileLlaMA,

- Evaluation Metrics
  - Rouge-L
  - BERTScore
  - BLEU
  - SacreBLEU

- Baselines
  - Baseline-L (single low precision)
  - Baseline-H (single high precision)
  - Dense-and-Sparse decomposition (DNS), SOTA low-precision quantization

Algorithm Performance

- PMPD-Static: Static Scheduler
  - Comparison with baseline-h: negligible performance loss with **up to 33%** reduction in bitwidth

Method	CNN/DM Model (↓): Vicuna-7B MobileLlaMA , Phi-1.5		Dialogsum Model (↓)): Vicuna-7B MobileLlaMA, Phi-1.5		IWSLT Model (↓)): Vicuna-7B MobileLlaMA, Zephyr-3B	
	Bit	**Rouge-L/ BERTScore**	**Bit**	**Rouge-L/ BERTScore**	**Bit**	**BLEU/ SacreBLEU**
Baseline-l	2	8.30 / 78.4	2	10.2 / 75.5	2	1.2 / 1.2
Baseline-h	3	24.2 / 86.9	3	24.4 / **88.2**	**3**	**31.6 / 31.6**
DNS	2.39	24.2 / 86.8	2.0	-	2.68	27.6 / 27.6
PMPD-Static	**2.39**	**24.3 / 87.0**	**2.0**	**25.0 / 88.2**	2.68	31.0 / 31.1
PMPD-Learned	2.43	24.0 / 86.7	2.74	24.5 / **88.2**	**2.37**	29.9 / 29.9
Baseline-l	3	16.3 / 83.3	3	15.8 / 84.1	3	9.8 / 9.83
Baseline-h	4	17.2 / 83.5	4	16.8 / 84.9	4	**12.7 / 12.7**
DNS	3.37	17.4 / 83.5	3.21	14.7 / 84.4	3.65	12.0 / 12.0
PMPD-Static	3.37	**17.6 / 83.7**	**3.0**	17.0 / **85.0**	3.65	12.6 / 12.6
PMPD-Learned	**3.19**	16.6 / 83.2	3.21	**17.1 / 85.0**	**3.48**	11.8 / 11.8
Baseline-l	3	13.4 / 82.4	3	15.3 / 85.1	3	21.1 / 21.1
Baseline-h	4	**16.2 / 84.0**	4	18.0 / 86.1	4	**30.4 / 30.4**
DNS	3.71	12.4 / 81.8	3.30	16.1 / 85.7	3.34	28.2 / 28.2
PMPD-Static	3.71	**16.2 / 84.0**	**3.30**	**18.1 / 86.2**	**3.0**	29.7 / 29.7
PMPD-Learned	**3.09**	15.5 / 83.4	3.52	17.9 / 86.1	3.34	29.8 / 29.8

Algorithm Performance

- PMPD-Static: Static Scheduler
  - Comparison with baseline-h: negligible performance loss with **up to 33%** reduction in bitwidth
  - Comparison with SOTA: under the same bitwidth budget, **outperform** DNS methods in all the cases

Method	CNN/DM Model (↓): Vicuna-7B MobileLlaMA , Phi-1.5		Dialogsum Model (↓)): Vicuna-7B MobileLlaMA, Phi-1.5		IWSLT Model (↓)): Vicuna-7B MobileLlaMA, Zephyr-3B	
	Bit	**Rouge-L/ BERTScore**	**Bit**	**Rouge-L/ BERTScore**	**Bit**	**BLEU/ SacreBLEU**
Baseline-l	2	8.30 / 78.4	2	10.2 / 75.5	2	1.2 / 1.2
Baseline-h	3	24.2 / 86.9	3	24.4 / **88.2**	**3**	**31.6 / 31.6**
DNS	2.39	24.2 / 86.8	2.0	-	2.68	27.6 / 27.6
PMPD-Static	**2.39**	**24.3 / 87.0**	**2.0**	**25.0 / 88.2**	2.68	31.0 / 31.1
PMPD-Learned	2.43	24.0 / 86.7	2.74	24.5 / **88.2**	**2.37**	29.9 / 29.9
Baseline-l	3	16.3 / 83.3	3	15.8 / 84.1	3	9.8 / 9.83
Baseline-h	4	17.2 / 83.5	4	16.8 / 84.9	4	**12.7 / 12.7**
DNS	3.37	17.4 / 83.5	3.21	14.7 / 84.4	3.65	12.0 / 12.0
PMPD-Static	3.37	**17.6 / 83.7**	**3.0**	17.0 / **85.0**	3.65	12.6 / 12.6
PMPD-Learned	**3.19**	16.6 / 83.2	3.21	**17.1 / 85.0**	**3.48**	11.8 / 11.8
Baseline-l	3	13.4 / 82.4	3	15.3 / 85.1	3	21.1 / 21.1
Baseline-h	4	**16.2 / 84.0**	4	18.0 / 86.1	4	**30.4 / 30.4**
DNS	3.71	12.4 / 81.8	3.30	16.1 / 85.7	3.34	28.2 / 28.2
PMPD-Static	3.71	**16.2 / 84.0**	**3.30**	**18.1 / 86.2**	**3.0**	29.7 / 29.7
PMPD-Learned	**3.09**	15.5 / 83.4	3.52	17.9 / 86.1	3.34	29.8 / 29.8

Algorithm Performance

- PMPD-Static: Static Scheduler
  - ➢ Comparison with baseline-h: negligible performance loss with **up to 33%** reduction in bitwidth
  - ➢ Comparison with SOTA: under the same bitwidth budget, **outperform** DNS methods in all the cases
- PMPD-Learned: Learned Scheduler
  - ➢ Better performance than baselines and SOTA approaches

Method	CNN/DM Model (↓): Vicuna-7B MobileLlaMA , Phi-1.5		Dialogsum Model (↓)): Vicuna-7B MobileLlaMA, Phi-1.5		IWSLT Model (↓)): Vicuna-7B MobileLlaMA, Zephyr-3B	
	Bit	**Rouge-L/ BERTScore**	**Bit**	**Rouge-L/ BERTScore**	**Bit**	**BLEU/ SacreBLEU**
Baseline-l	2	8.30 / 78.4	2	10.2 / 75.5	2	1.2 / 1.2
Baseline-h	3	24.2 / 86.9	3	24.4 / **88.2**	**3**	**31.6 / 31.6**
DNS	2.39	24.2 / 86.8	2.0	-	2.68	27.6 / 27.6
PMPD-Static	**2.39**	**24.3 / 87.0**	**2.0**	**25.0 / 88.2**	2.68	31.0 / 31.1
PMPD-Learned	2.43	24.0 / 86.7	2.74	24.5 / **88.2**	**2.37**	29.9 / 29.9
Baseline-l	3	16.3 / 83.3	3	15.8 / 84.1	3	9.8 / 9.83
Baseline-h	4	17.2 / 83.5	4	16.8 / 84.9	4	**12.7 / 12.7**
DNS	3.37	17.4 / 83.5	3.21	14.7 / 84.4	3.65	12.0 / 12.0
PMPD-Static	3.37	**17.6 / 83.7**	**3.0**	17.0 / **85.0**	3.65	12.6 / 12.6
PMPD-Learned	**3.19**	16.6 / 83.2	3.21	**17.1 / 85.0**	**3.48**	11.8 / 11.8
Baseline-l	3	13.4 / 82.4	3	15.3 / 85.1	3	21.1 / 21.1
Baseline-h	4	**16.2 / 84.0**	4	18.0 / 86.1	4	**30.4 / 30.4**
DNS	3.71	12.4 / 81.8	3.30	16.1 / 85.7	3.34	28.2 / 28.2
PMPD-Static	3.71	**16.2 / 84.0**	**3.30**	**18.1 / 86.2**	**3.0**	29.7 / 29.7
PMPD-Learned	**3.09**	15.5 / 83.4	3.52	17.9 / 86.1	3.34	29.8 / 29.8

Algorithm Performance

- PMPD-Static: Static Scheduler
  - Comparison with baseline-h: negligible performance loss with **up to 33%** reduction in bitwidth
  - Comparison with SOTA: under the same bitwidth budget, **outperform** DNS methods in all the cases
- PMPD-Learned: Learned Scheduler
  - Better performance than baselines and SOTA approaches
  - Slightly worse than PMPD-Static, but has higher generality since PMPD-Learned does not require validation datasets for calibration

Method	CNN/DM Model (↓): Vicuna-7B MobileLlaMA , Phi-1.5		Dialogsum Model (↓)): Vicuna-7B MobileLlaMA, Phi-1.5		IWSLT Model (↓)): Vicuna-7B MobileLlaMA, Zephyr-3B	
	Bit	Rouge-L/ BERTScore	Bit	Rouge-L/ BERTScore	Bit	BLEU/ SacreBLEU
Baseline-l	2	8.30 / 78.4	2	10.2 / 75.5	2	1.2 / 1.2
Baseline-h	3	24.2 / 86.9	3	24.4 / **88.2**	**3**	**31.6 / 31.6**
DNS	2.39	24.2 / 86.8	2.0	-	2.68	27.6 / 27.6
PMPD-Static	**2.39**	**24.3 / 87.0**	**2.0**	**25.0 / 88.2**	2.68	31.0 / 31.1
PMPD-Learned	2.43	24.0 / 86.7	2.74	24.5 / **88.2**	**2.37**	29.9 / 29.9
Baseline-l	3	16.3 / 83.3	3	15.8 / 84.1	3	9.8 / 9.83
Baseline-h	4	17.2 / 83.5	4	16.8 / 84.9	4	**12.7 / 12.7**
DNS	3.37	17.4 / 83.5	3.21	14.7 / 84.4	3.65	12.0 / 12.0
PMPD-Static	3.37	**17.6 / 83.7**	**3.0**	17.0 / **85.0**	3.65	12.6 / 12.6
PMPD-Learned	**3.19**	16.6 / 83.2	3.21	**17.1 / 85.0**	**3.48**	11.8 / 11.8
Baseline-l	3	13.4 / 82.4	3	15.3 / 85.1	3	21.1 / 21.1
Baseline-h	4	**16.2 / 84.0**	4	18.0 / 86.1	4	**30.4 / 30.4**
DNS	3.71	12.4 / 81.8	3.30	16.1 / 85.7	3.34	28.2 / 28.2
PMPD-Static	3.71	**16.2 / 84.0**	**3.30**	**18.1 / 86.2**	**3.0**	29.7 / 29.7
PMPD-Learned	**3.09**	15.5 / 83.4	3.52	17.9 / 86.1	3.34	29.8 / 29.8

Hardware Performance

- GPU Speedup Comparison
  - Evaluation platforms: Nvidia RTX 4090 & A40
  - Baseline-h versus. PMPD
  - Operations: Attn Proj. (attention projection) and MLP Proj. (MLP projection)
  - PMPD shows consistent speedup over Baseline-h on both GPU platforms

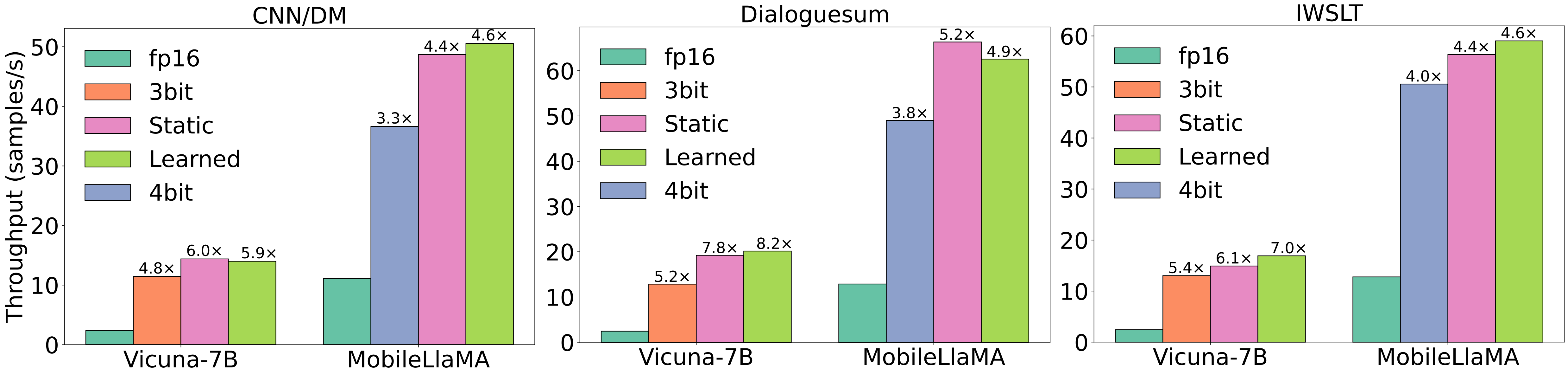
		Vicuna-7B		MobileLlama		Phi-1.5		Zephyr-3B	
		Attn Proj.	MLP Proj.	Attn Proj.	MLP Proj.	Attn Proj.	MLP Proj.	Attn Proj.	MLP Proj.
RTX 4090	Baseline-h	6.25×	11.32×	1.25×	2.50×	1.32×	1.70×	3.33×	2.54×
	PMPD	6.25×	12.20×	1.40×	2.81×	1.51×	1.84×	2.73×	2.82×
A40	Baseline-h	5.81×	3.77×	3.00×	3.96×	2.57×	2.83×	3.50×	3.02×
	PMPD	6.58×	4.60×	3.37×	4.74×	2.77×	3.51×	3.81×	4.27×

Hardware Performance

- Dataflow: Simulate PMPD support (weight transfer for centroids)

- Speedup of PMPD on Dataflow
  - PMPD introduce 4 ~ 8x speedup on Dataflow architecture
  - Higher speedup in 7B models: more memory-bound

- Static Scheduler vs Learned Scheduler
  - Similar speedup across different models & datasets



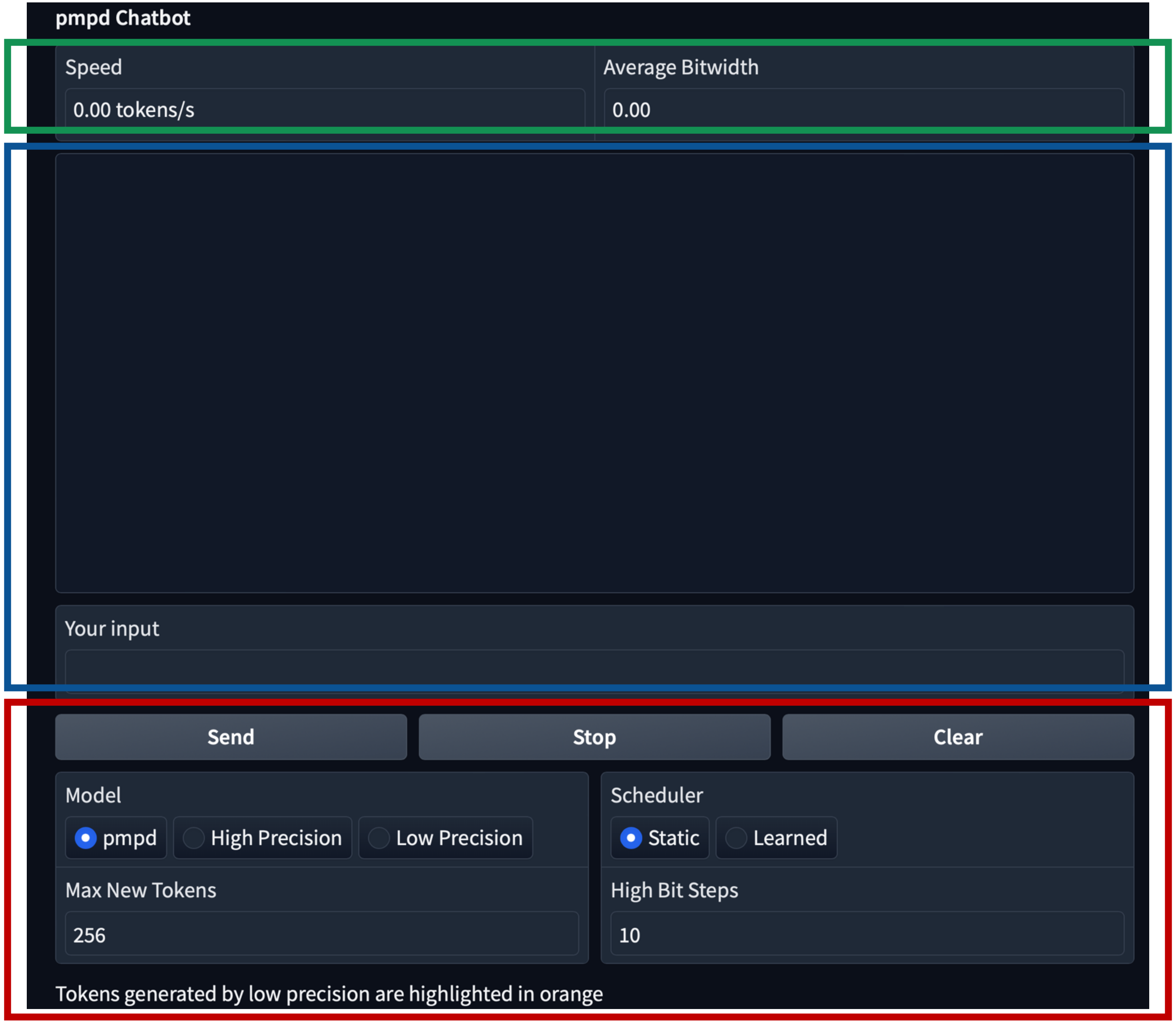
Demo

- **Demo GUI**

- Speed Measurement
  - Decoding Speed
  - Average Bitwidth

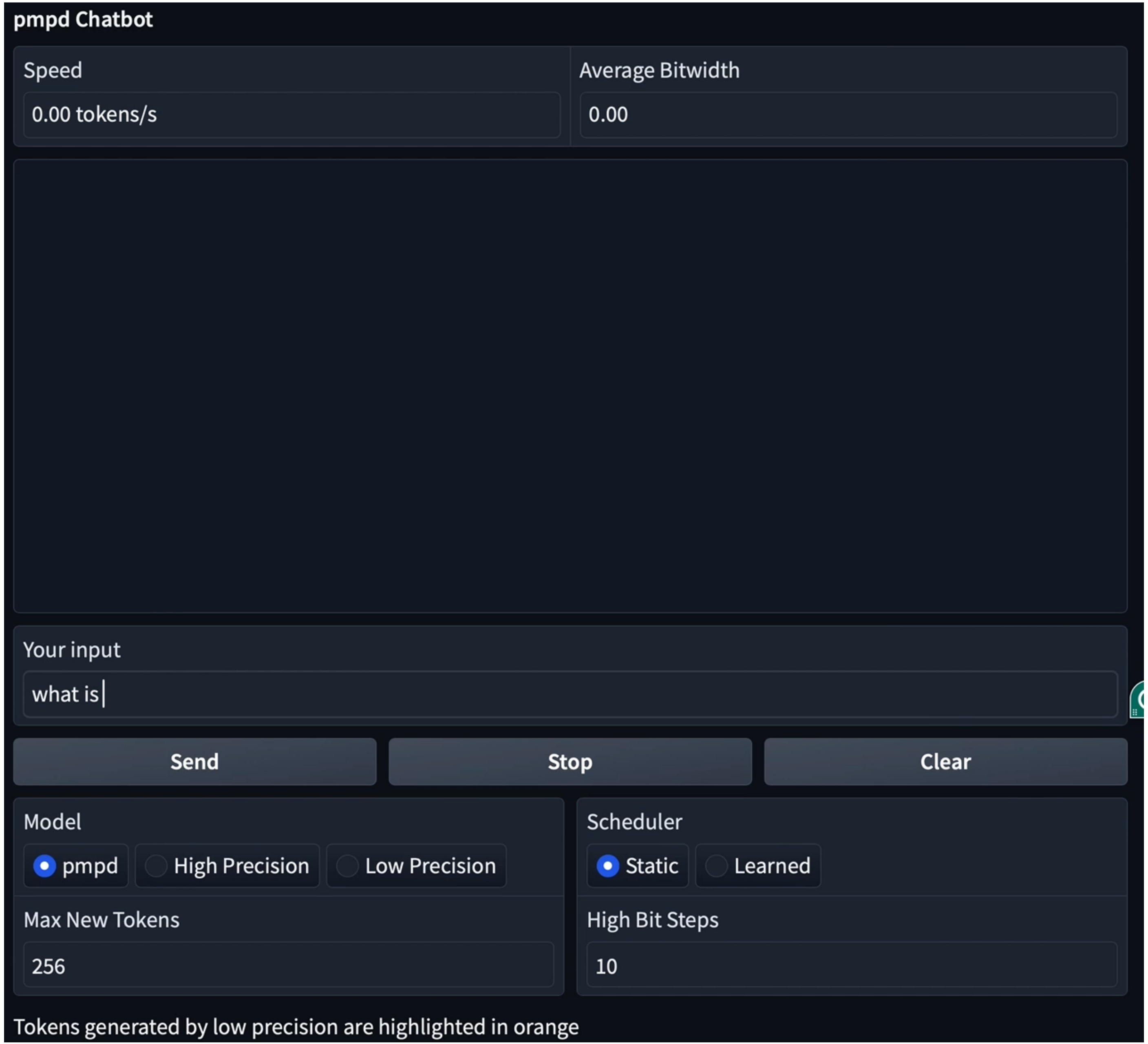
- Chatbot with custom input prompts
  - Input box for any prompts
  - Output box for generated results

- Configuration Panel
  - Model Spec (PMPD/Low/High)
  - Config of Max New Tokens
  - Scheduler Choice (Static/Learned)



Demo

- **Demo1: Inference with lower average bitwidth**
  - High precision at the beginning
  - Low precision (orange) in the later stage of the decoding

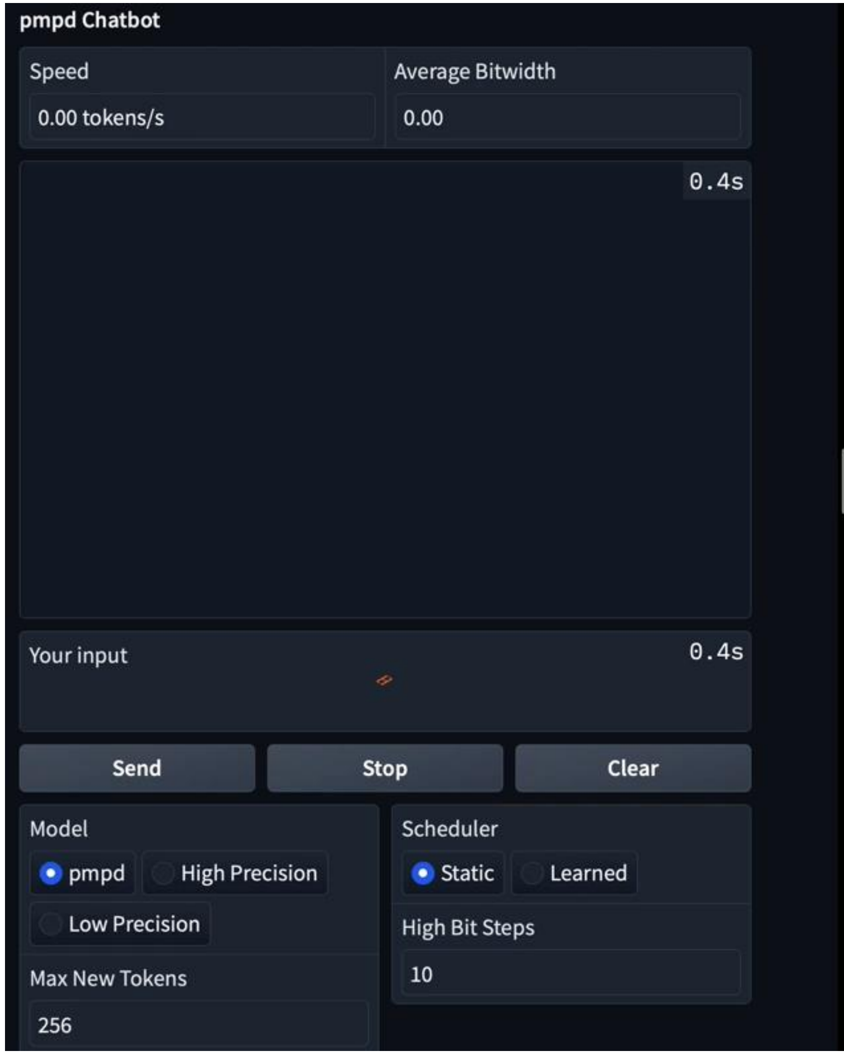


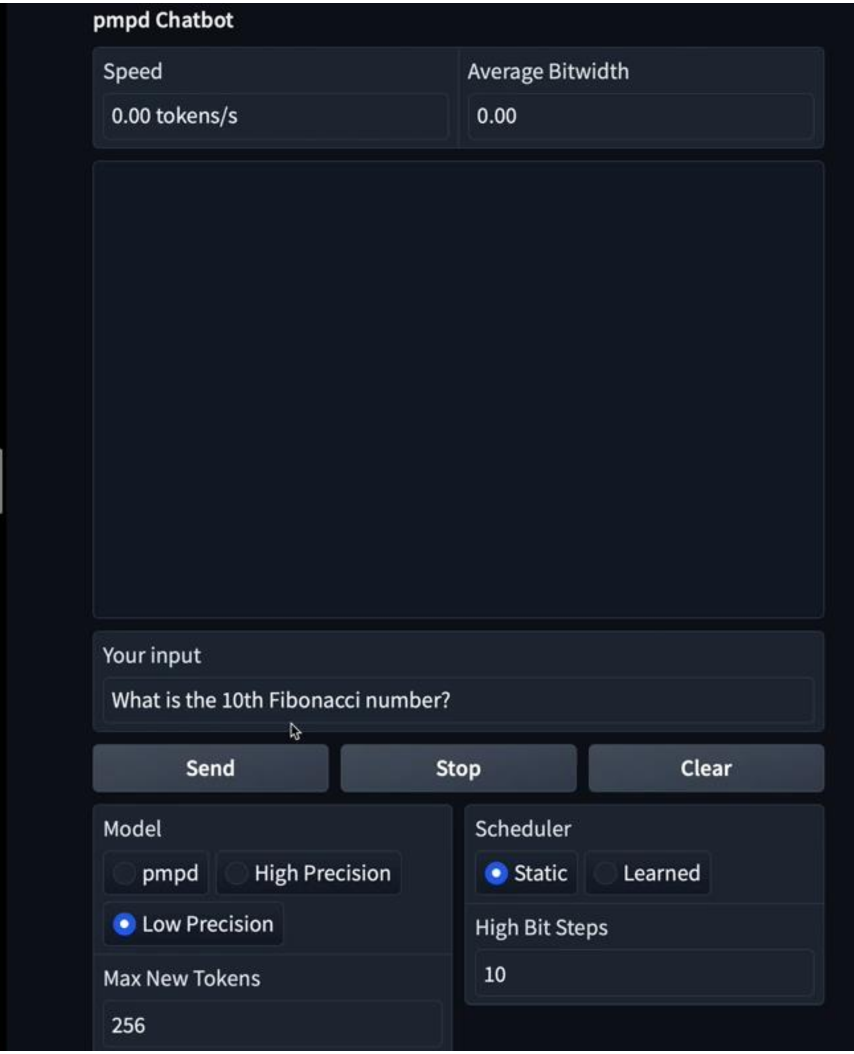
Demo

- Demo2: Generation Quality
  - PMPD: generates the correct result
  - Baseline-L: predicts the wrong answer

PMPD: Correct results (55)

Baseline-L: Wrong results (10)





Thank you!