ICLR

Draft-based Approximate Inference for LLMs

Kevin Galim*, Ethan Ewer*, Wonjun Kang, Minjae Lee,
Hyung Il Koo, Kangwook Lee

Motivation

- **Long-context LLMs are expensive**
  - Compute: quadratic in sequence length
  - Memory: grows with KV cache
- **Two common approximation methods**
  - KV cache dropping
  - Prompt compression
- **Core challenge:** How do we identify important tokens without hurting accuracy?

Prior Work and Gap

- **Past-heuristic methods**
  - Use past attention statistics
  - Often unreliable for future decoding
- **Lookahead methods**
  - Improve importance estimation
  - But still have limitations:
    - may require target-model memory
    - lack theoretical justification
- **Gap:** Can a small draft model reliably guide a large target model?

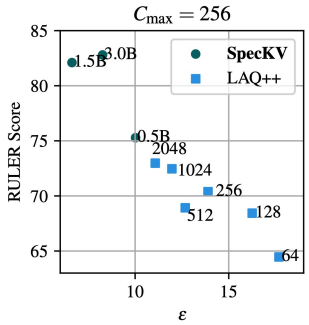
Our Contributions

- **A unified framework for draft-based approximation**
  - **SpecKV:** draft-based KV cache dropping
  - **SpecPC:** draft-based prompt compression
  - **SpecKV-PC:** cascaded pipeline combining both
- **Main contribution:** First theoretical justification for draft-based approximate inference

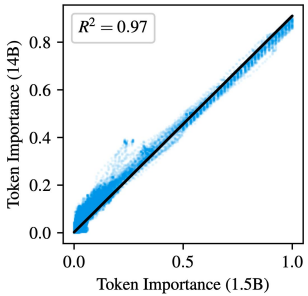
Why It Works

- **Key theoretical result**
  - If the draft model approximates the target well,
  - then it can also reliably estimate:
    - token importance
    - attention structure
- **Implication:** A small draft model can serve as a reliable guide for approximation

Downstream Accuracy vs Draft-Target Error ε

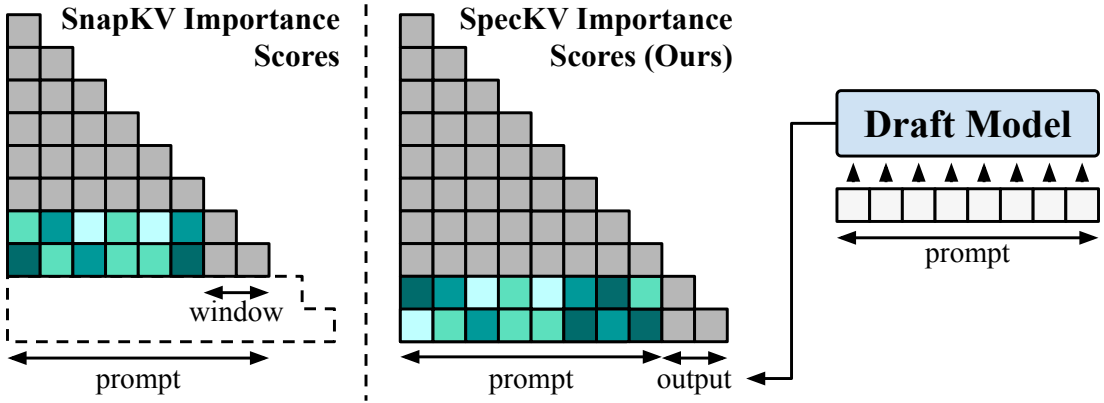


Token Importance Corr. (Qwen 1.5B vs 14B)



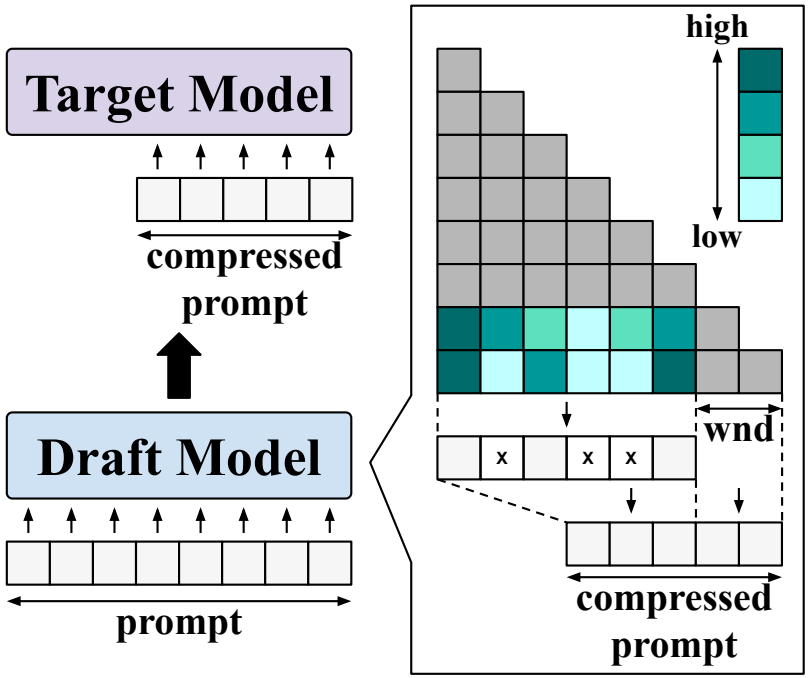
SpecKV: Speculative KV Cache Dropping

1. **Draft Generate:** A tiny, fast draft model predicts future tokens.
2. **Target Prefill:** The target model processes these "lookahead" tokens with the prompt.
3. **KV Dropping:** By peeking at the “future”, SpecKV accurately identifies which KV pairs are truly important.



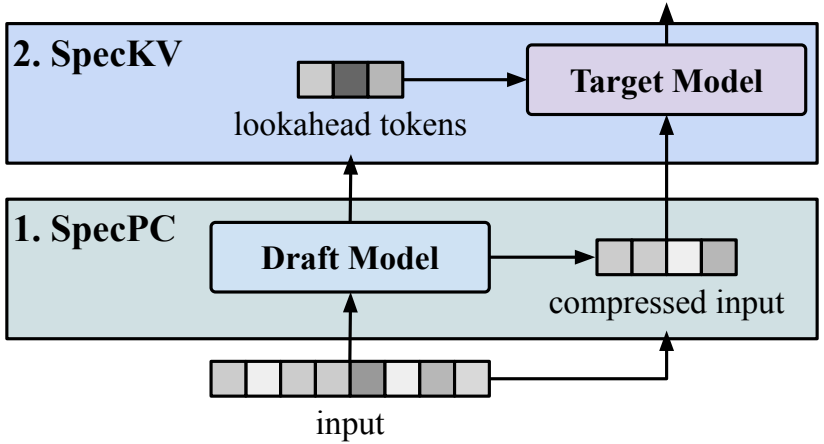
SpecPC: Speculative Prompt Compression

1. **Draft Process:** A small, fast draft model processes the full input prompt.
2. **Importance Scoring:** Draft attention scores are extracted to reliably estimate token importance.
3. **Prompt Compression:** Unimportant tokens are dropped before they ever reach the target model.



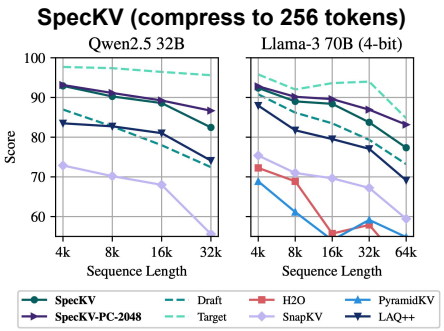
SpecKV-PC: The Cascaded Pipeline

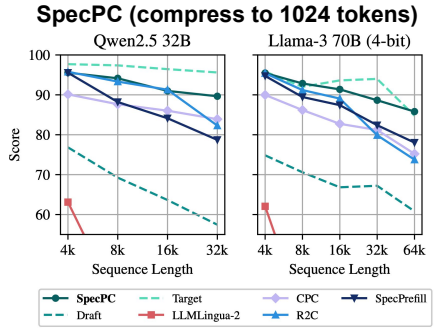
1. **Stage 1 (SpecPC):** Compresses the initial prompt to accelerate the target model's prefill phase.
2. **Stage 2 (SpecKV):** Optimizes the target's KV cache using lookahead tokens for efficient decoding.
3. Maximum Efficiency: The cascaded pipeline delivers optimal speed and minimizes peak memory usage.



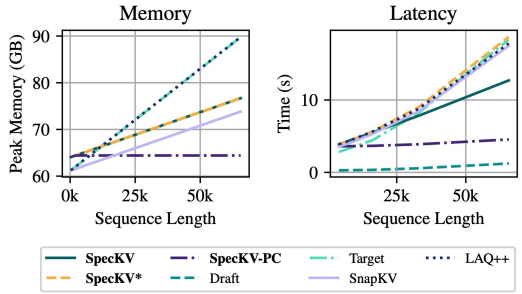
Experimental Results

- **Accuracy (RULER & LongBench):**
  - SpecKV, SpecPC, and SpecKV-PC consistently outperform H2O, SnapKV, and LLMLingua-2.
  - **Strongest gains at 32k+ context lengths.**





- **Efficiency:**
  - At 64k context, SpecKV-PC reduces **latency by 75%** and **saves 25 GB of peak memory** compared to LAQ++.



Conclusion

- **Summary:**
  - **Unified Framework:** Solved memory bottlenecks of prior lookahead methods.
  - **Theory:** Provided the **first theoretical proofs** for draft-based approximation.
  - **Performance:** SpecKV, SpecPC, and SpecKV-PC offers SOTA accuracy & efficiency.
- **Code:** github.com/furiosa-ai/draft-based-approx-llm

FURIOSA

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON