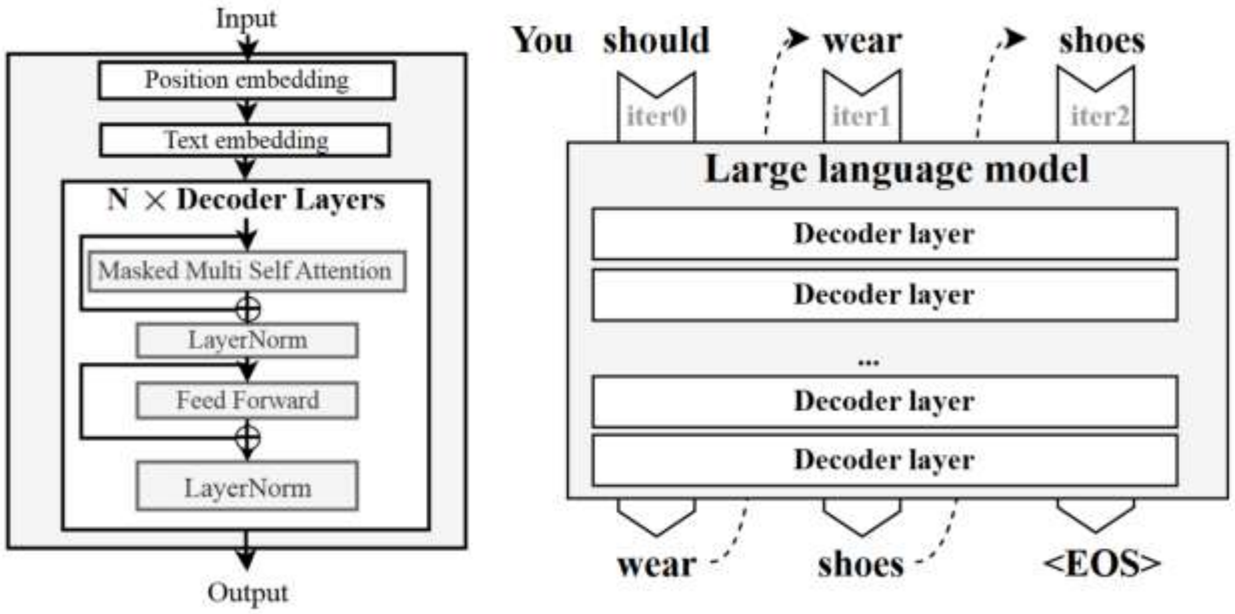
Speculative Decoding

14 Dec 2023

Background

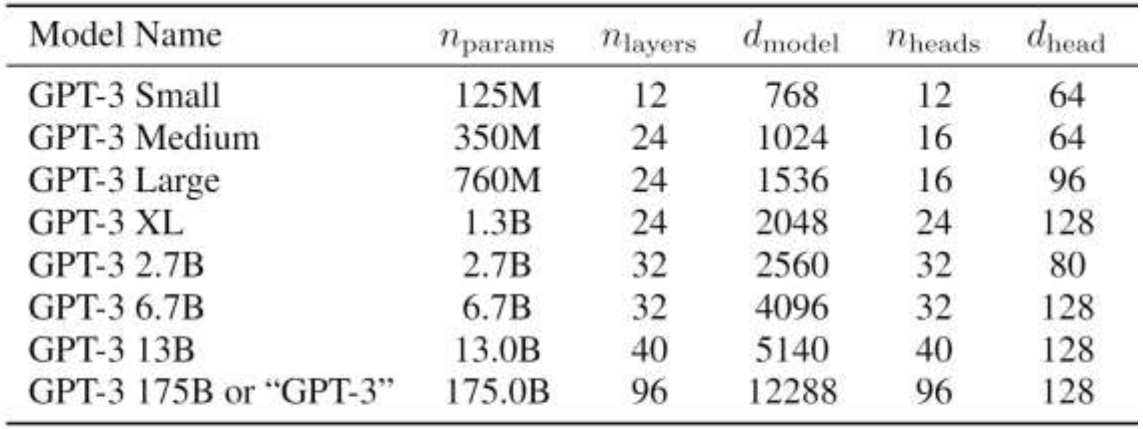
- The generation of one token depends on previous tokens in autoregressive inference
- Some steps are easy and the other steps are hard



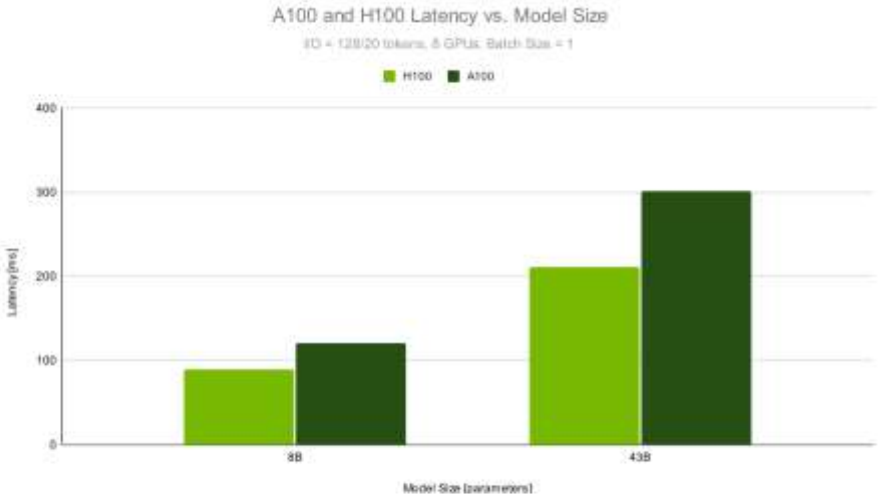
(a) GPT3 architecture (b) Autoregressive inference

Background

- A LLM is usually released with different size models
- The inference of a small model can be much faster than that of a large model

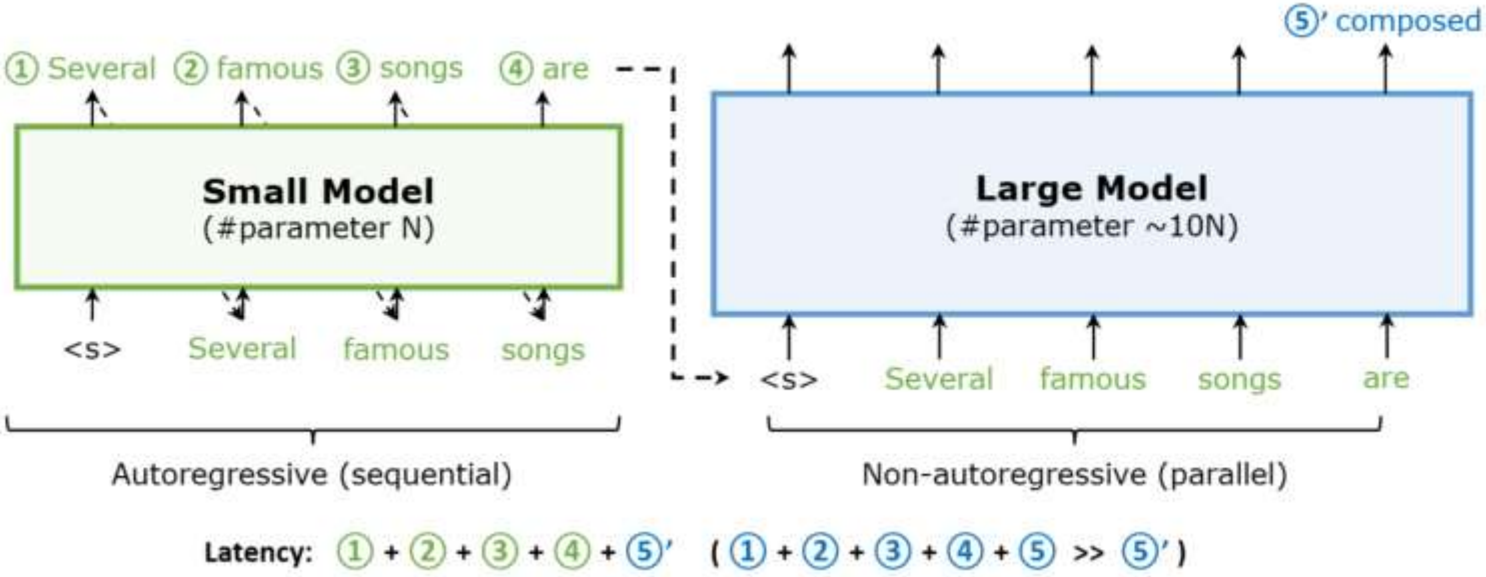


Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}
GPT-3 Small	125M	12	768	12	64
GPT-3 Medium	350M	24	1024	16	64
GPT-3 Large	760M	24	1536	16	96
GPT-3 XL	1.3B	24	2048	24	128
GPT-3 2.7B	2.7B	32	2560	32	80
GPT-3 6.7B	6.7B	32	4096	32	128
GPT-3 13B	13.0B	40	5140	40	128
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128



Speculative Decoding

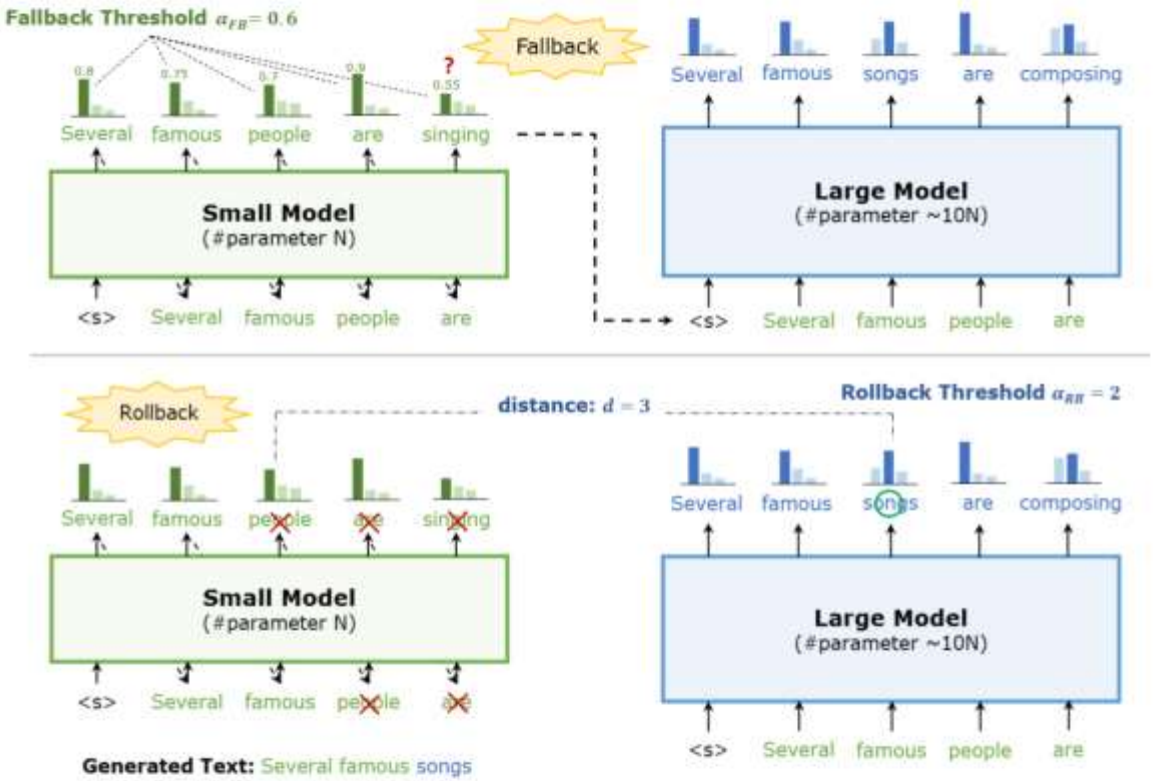
- Firstly, let a small model(draft model) sample tokens
- Then let the target model verify sampled tokens in parallel



① Several ② famous ③ songs ④ are
Small Model
(#parameter N)
<s> Several famous songs
Autoregressive (sequential)
⑤' composed
Large Model
(#parameter ~10N)
<s> Several famous songs are
Non-autoregressive (parallel)
Latency: ① + ② + ③ + ④ + ⑤' (① + ② + ③ + ④ + ⑤ >> ⑤')

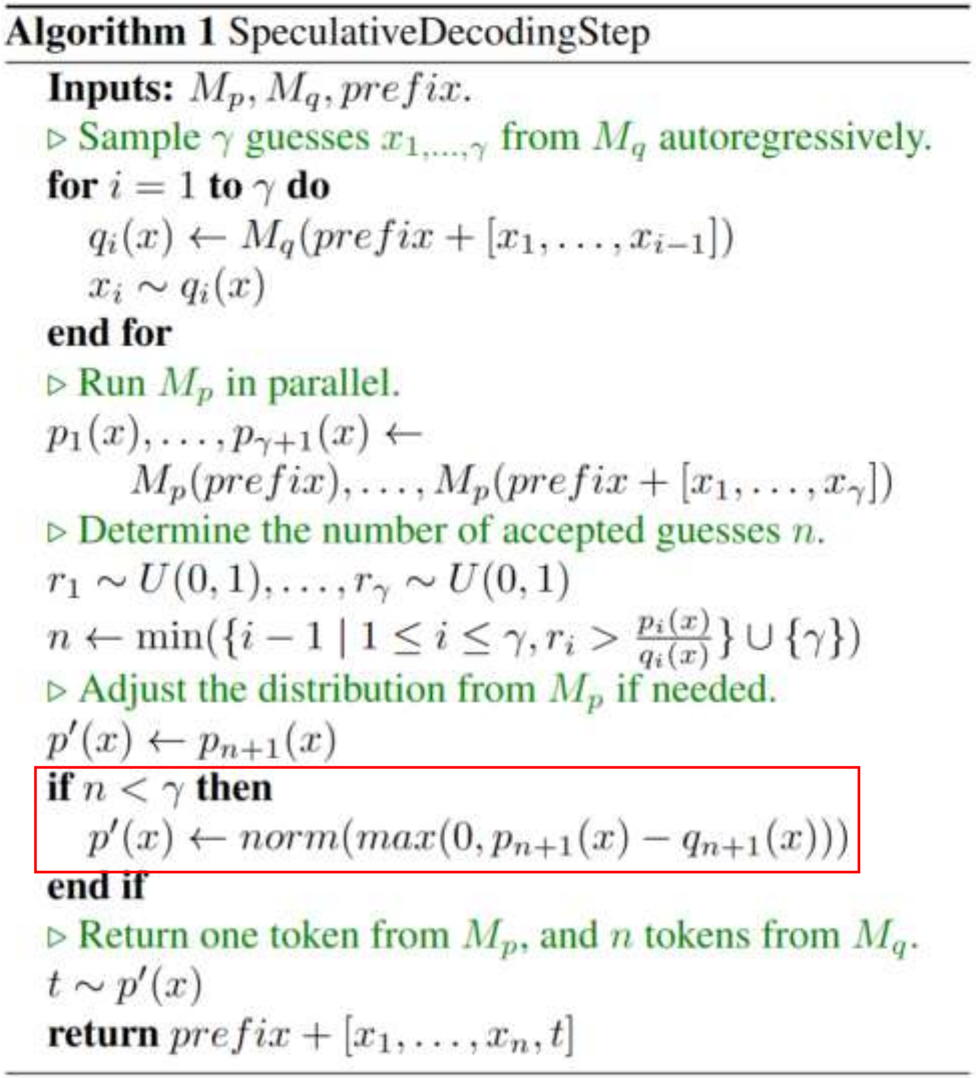
Speculative Decoding: Roll back

- If not all tokens are verified successfully, roll back to the last succeed token and continue to generate tokens.



Speculative Decoding: Algorithm

- γ: number of sampled tokens
- r_i is sampled from U(0,1)
- M_p:Target model, M_q: Draft model
- A token is accepted if r_i<max(1,$\frac{p(x)}{q(x)}$)
- In one Iteration, the draft model will generate γ tokens and the target model will generate 1 token.
- The final generated token has the same distribution as the target model

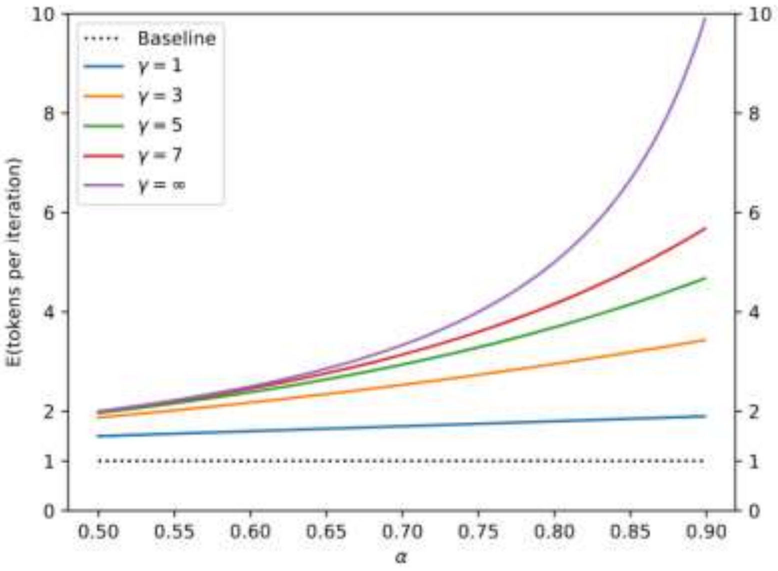


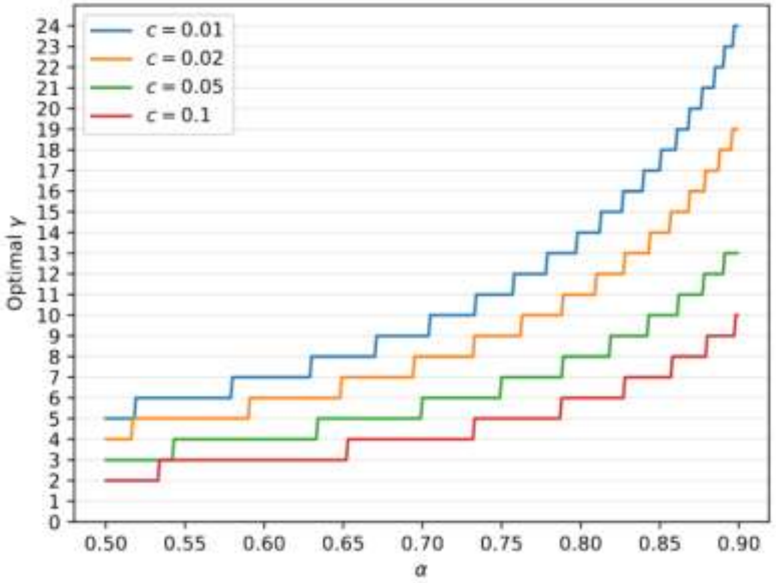
Algorithm 1 SpeculativeDecodingStep

```
Inputs: $M_p, M_q, prefix$.
▷ Sample $\gamma$ guesses $x_{1,\dots,\gamma}$ from $M_q$ autoregressively.
for $i = 1$ to $\gamma$ do
    $q_i(x) \leftarrow M_q(prefix + [x_1, \dots, x_{i-1}])$
    $x_i \sim q_i(x)$
end for
▷ Run $M_p$ in parallel.
$p_1(x), \dots, p_{\gamma+1}(x) \leftarrow$
        $M_p(prefix), \dots, M_p(prefix + [x_1, \dots, x_\gamma])$
▷ Determine the number of accepted guesses $n$.
$r_1 \sim U(0,1), \dots, r_\gamma \sim U(0,1)$
$n \leftarrow \min(\{i - 1 \mid 1 \le i \le \gamma, r_i > \frac{p_i(x)}{q_i(x)}\} \cup \{\gamma\})$
▷ Adjust the distribution from $M_p$ if needed.
$p'(x) \leftarrow p_{n+1}(x)$
if $n < \gamma$ then
    $p'(x) \leftarrow norm(max(0, p_{n+1}(x) - q_{n+1}(x)))$
end if
▷ Return one token from $M_p$, and $n$ tokens from $M_q$.
$t \sim p'(x)$
return $prefix + [x_1, \dots, x_n, t]$
```

Speculative Decoding: Analysis

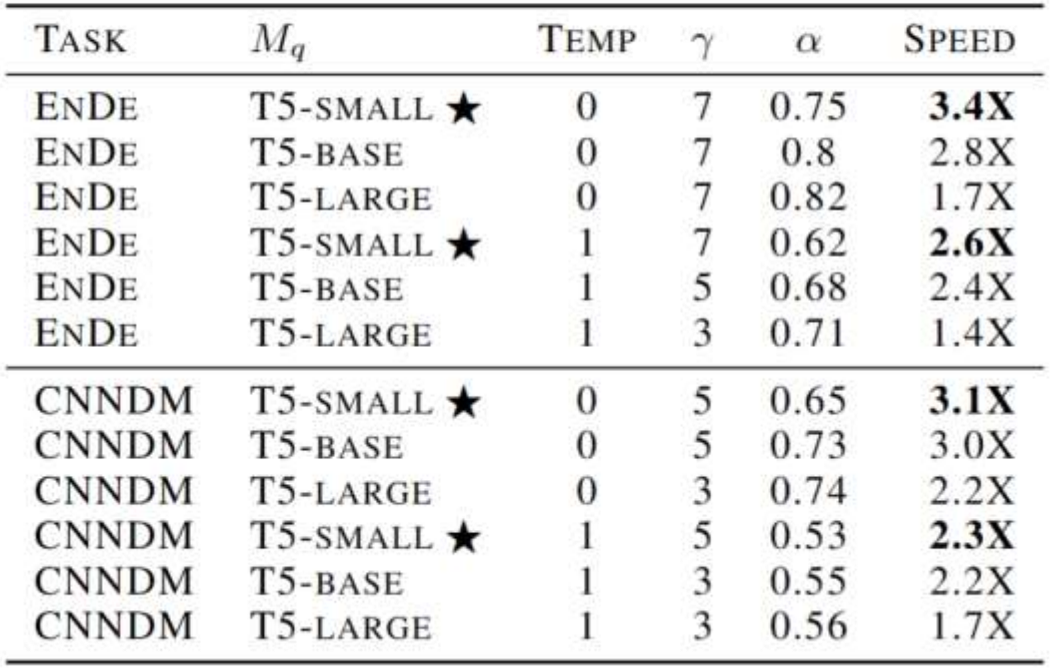
- α is the average acceptance rate
- c is the ratio between the time for a single run of the draft model and the time for a single run of target model





Speculative Decoding

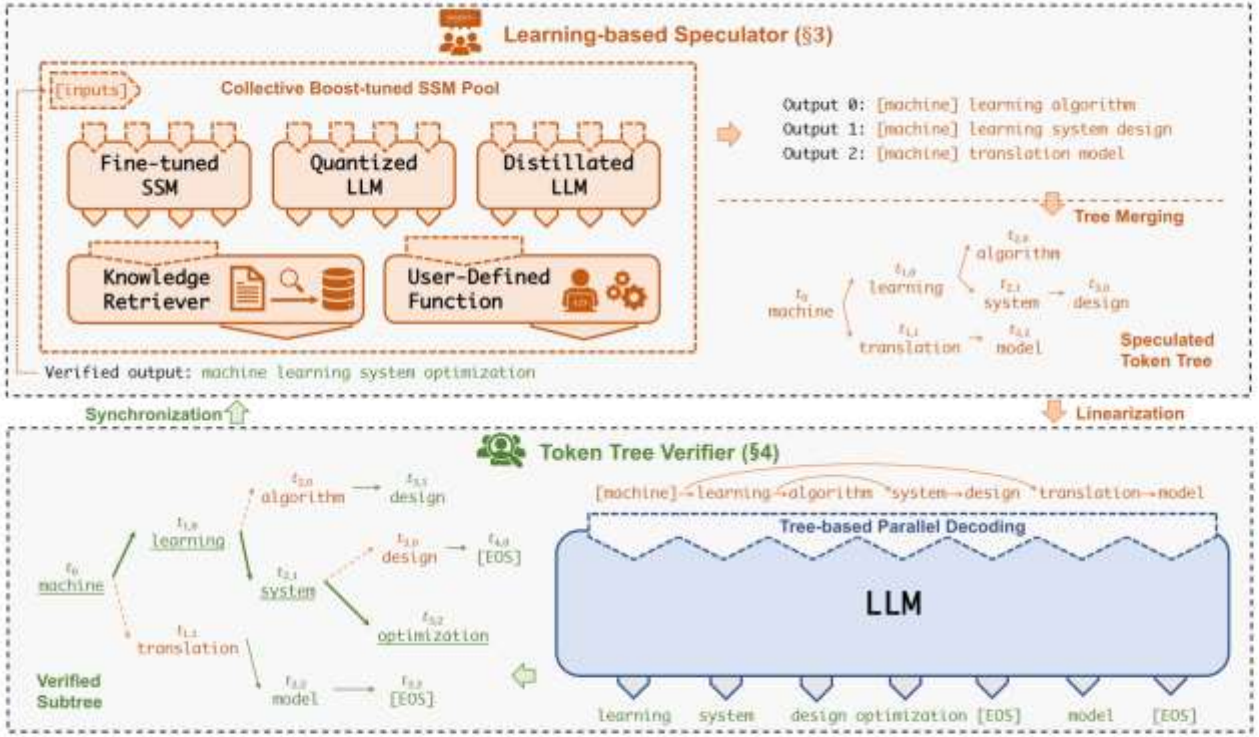
- Evaluation
- Target model:T5-XXL 11B



TASK	M_q	TEMP	γ	α	SPEED
ENDE	T5-SMALL ★	0	7	0.75	**3.4X**
ENDE	T5-BASE	0	7	0.8	2.8X
ENDE	T5-LARGE	0	7	0.82	1.7X
ENDE	T5-SMALL ★	1	7	0.62	**2.6X**
ENDE	T5-BASE	1	5	0.68	2.4X
ENDE	T5-LARGE	1	3	0.71	1.4X
CNNDM	T5-SMALL ★	0	5	0.65	**3.1X**
CNNDM	T5-BASE	0	5	0.73	3.0X
CNNDM	T5-LARGE	0	3	0.74	2.2X
CNNDM	T5-SMALL ★	1	5	0.53	**2.3X**
CNNDM	T5-BASE	1	3	0.55	2.2X
CNNDM	T5-LARGE	1	3	0.56	1.7X

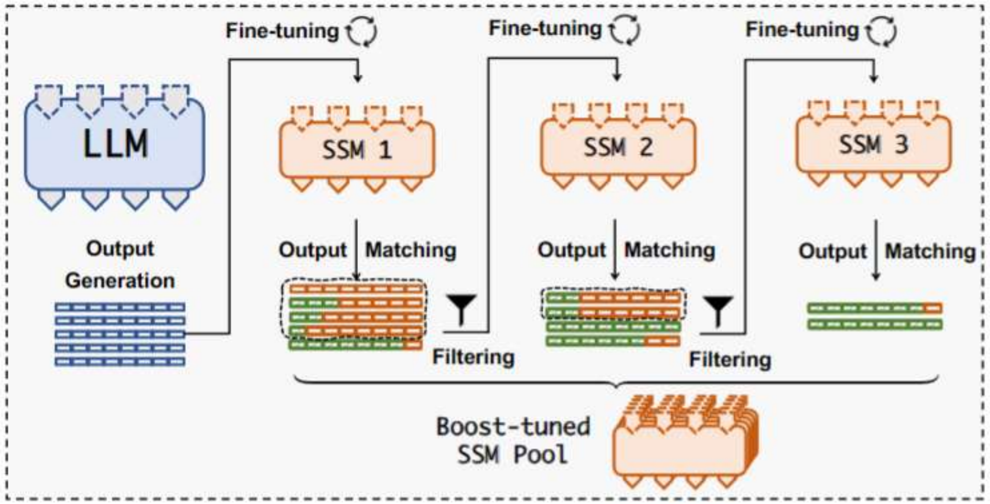
SpecInfer

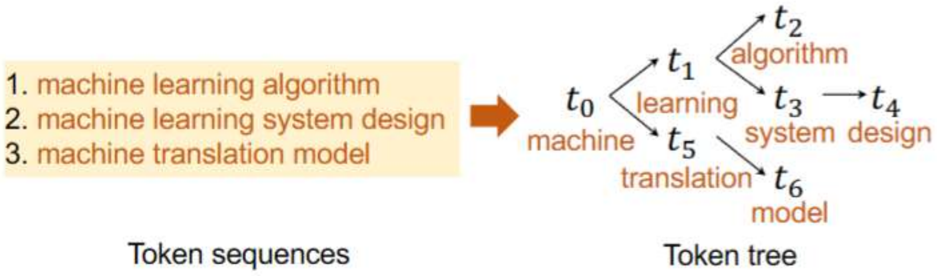
- Combine various collectively boost-tuned small draft models to jointly predict the LLM’s output
- Use the target model to verify all predictions



SpecInfer : Decoding

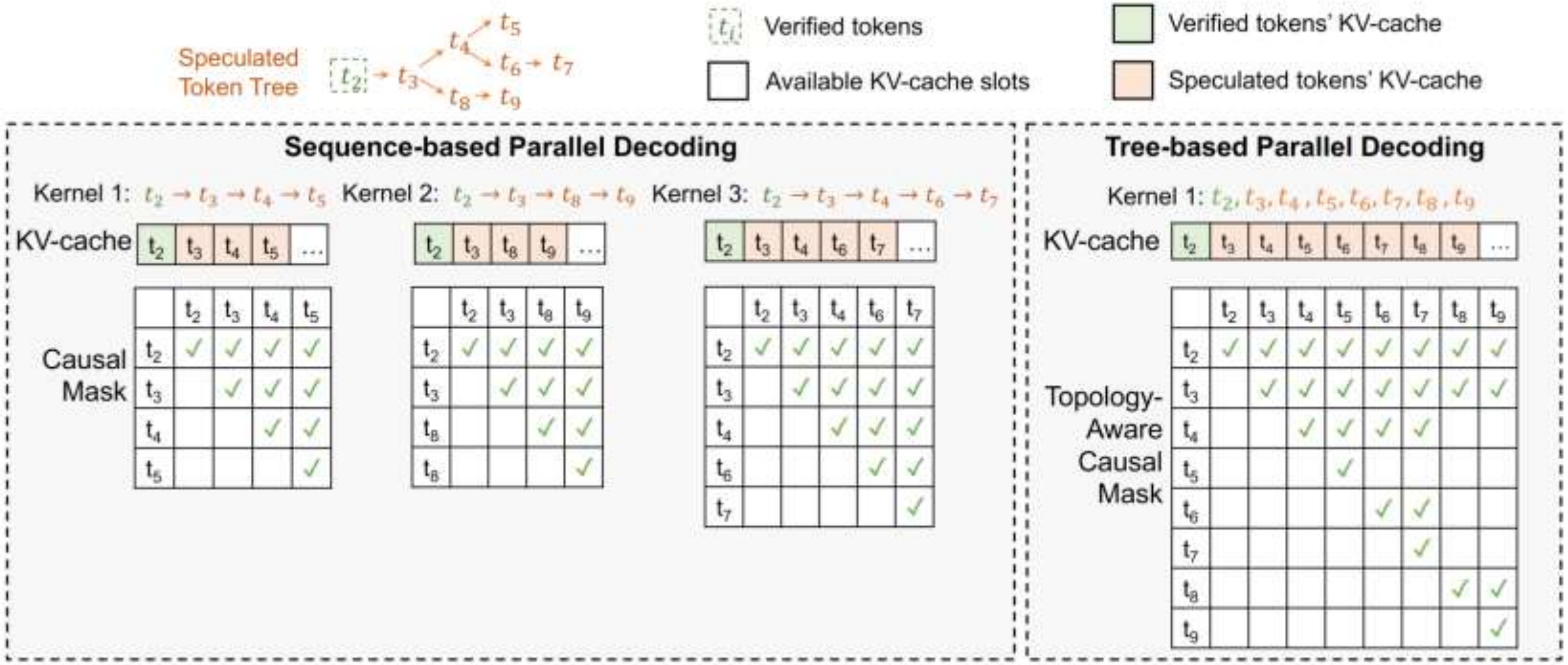
- Collectively boost-tune the draft models
- When decoding, every draft model generate a sequence
- The sequences will be merged into a tree




1. machine learning algorithm
2. machine learning system design
3. machine translation model

SpecInfer: Verification

- A topology-aware casual mask based on the token tree is used in self-attention.
- Target model will take all speculated tokens as input


Speculated Token Tree
$t_2 \rightarrow t_3 \rightarrow t_4 \rightarrow t_5$; $t_4 \rightarrow t_6 \rightarrow t_7$; $t_3 \rightarrow t_8 \rightarrow t_9$
t_i Verified tokens
Available KV-cache slots
Verified tokens' KV-cache
Speculated tokens' KV-cache
Sequence-based Parallel Decoding
Kernel 1: $t_2 \rightarrow t_3 \rightarrow t_4 \rightarrow t_5$
Kernel 2: $t_2 \rightarrow t_3 \rightarrow t_8 \rightarrow t_9$
Kernel 3: $t_2 \rightarrow t_3 \rightarrow t_4 \rightarrow t_6 \rightarrow t_7$
KV-cache
t_2 t_3 t_4 t_5 ...
t_2 t_3 t_8 t_9 ...
t_2 t_3 t_4 t_6 t_7 ...
Causal Mask
Tree-based Parallel Decoding
Kernel 1: $t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9$
KV-cache
t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 ...
Topology-Aware Causal Mask

	t_2	t_3	t_4	t_5
t_2	✓	✓	✓	✓
t_3		✓	✓	✓
t_4			✓	✓
t_5				✓

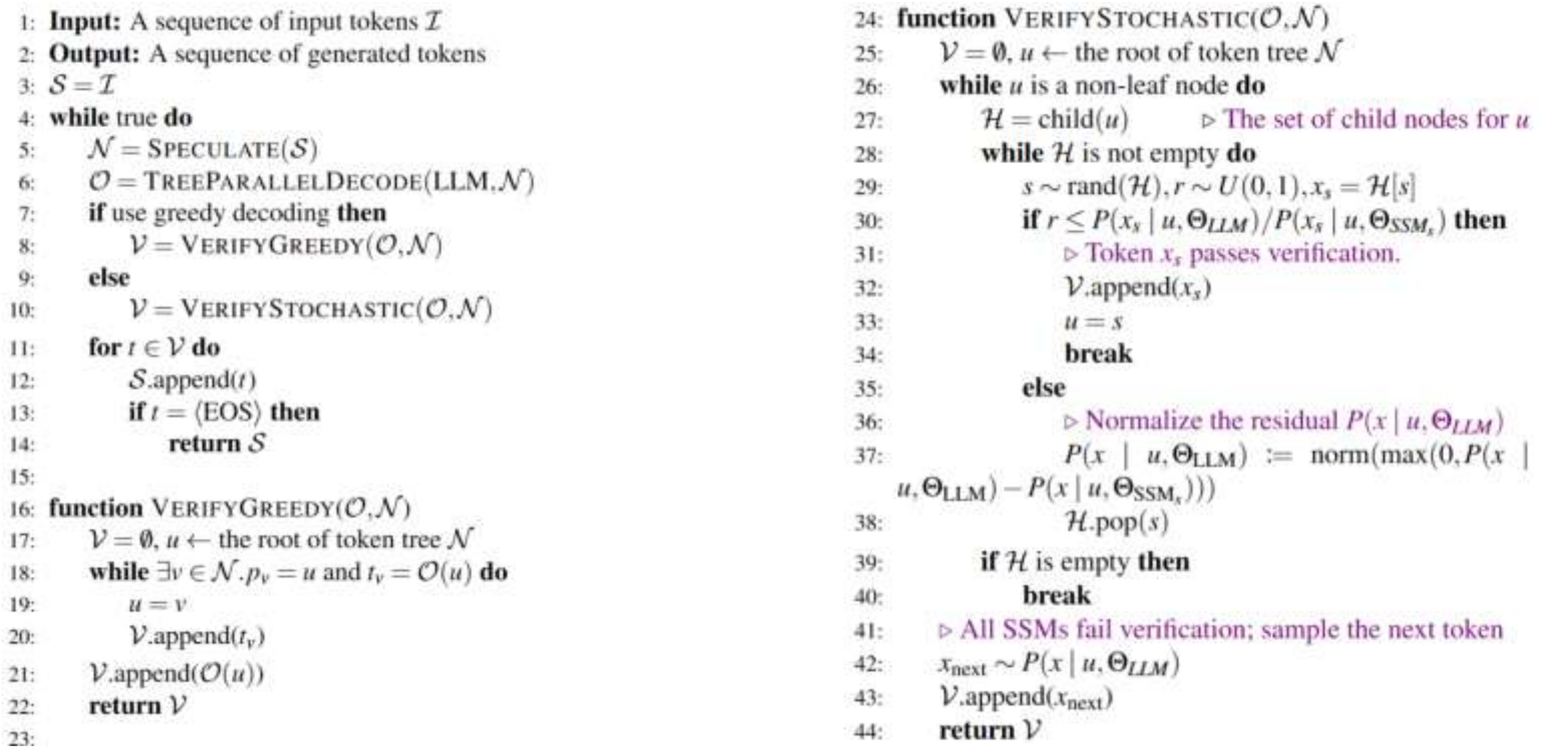
	t_2	t_3	t_8	t_9
t_2	✓	✓	✓	✓
t_3		✓	✓	✓
t_8			✓	✓
t_8				✓

	t_2	t_3	t_4	t_6	t_7
t_2	✓	✓	✓	✓	✓
t_3		✓	✓	✓	✓
t_4			✓	✓	✓
t_6				✓	✓
t_7					✓

	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
t_2	✓	✓	✓	✓	✓	✓	✓	✓
t_3		✓	✓	✓	✓	✓	✓	✓
t_4			✓	✓	✓	✓		
t_5				✓				
t_6					✓	✓		
t_7						✓		
t_8							✓	✓
t_9								✓

SpecInfer: Algorithm

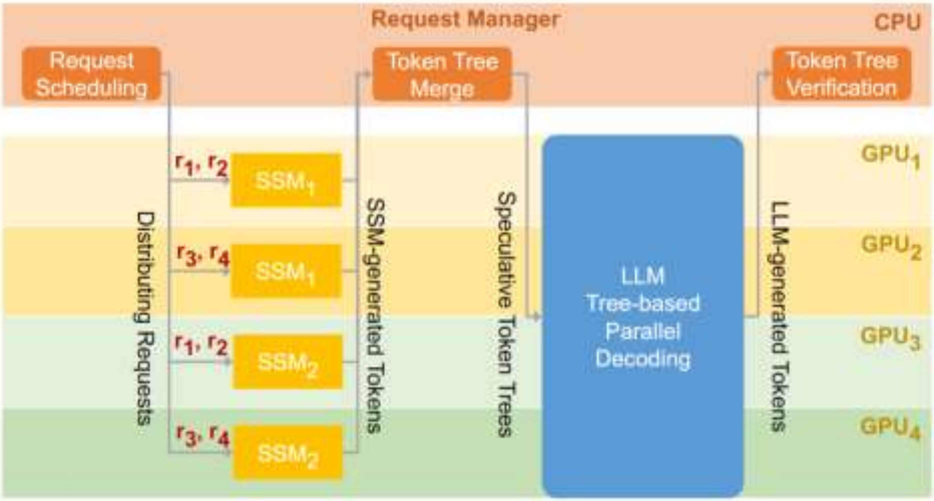
- VerifyGreedy: Always choose the token with maximum probability
- VerifyStochastic: Sample tokens from a probability distribution

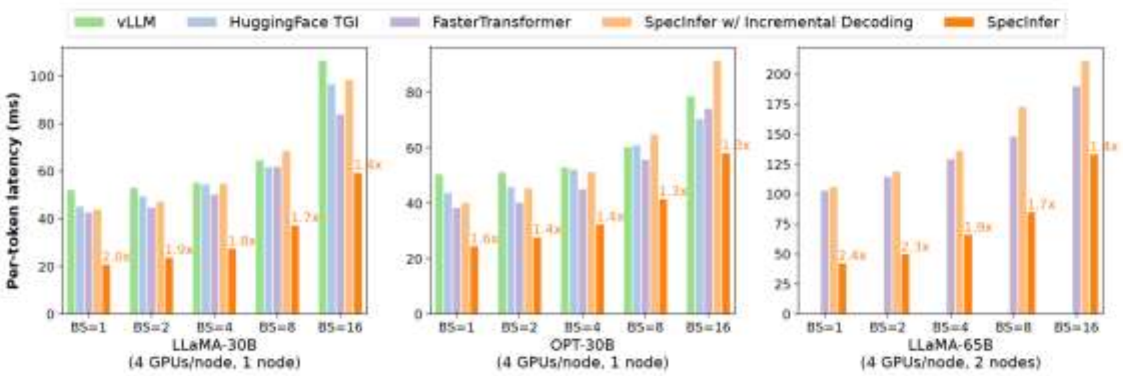


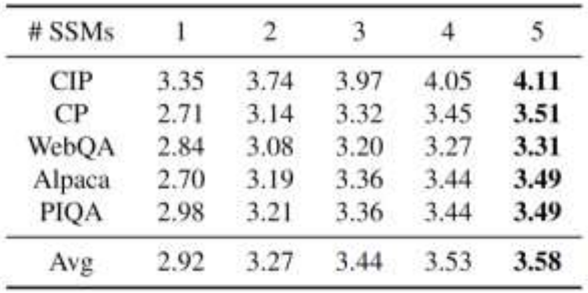
```
Input: A sequence of input tokens $\mathcal{I}$
Output: A sequence of generated tokens
$\mathcal{S} = \mathcal{I}$
while true do
    $\mathcal{N}$ = SPECULATE($\mathcal{S}$)
    $\mathcal{O}$ = TREEPARALLELDECODE(LLM, $\mathcal{N}$)
    if use greedy decoding then
        $\mathcal{V}$ = VERIFYGREEDY($\mathcal{O}, \mathcal{N}$)
    else
        $\mathcal{V}$ = VERIFYSTOCHASTIC($\mathcal{O}, \mathcal{N}$)
    for $t \in \mathcal{V}$ do
        $\mathcal{S}$.append($t$)
        if $t = \langle EOS \rangle$ then
            return $\mathcal{S}$

function VERIFYGREEDY($\mathcal{O}, \mathcal{N}$)
    $\mathcal{V} = \emptyset$, $u \leftarrow$ the root of token tree $\mathcal{N}$
    while $\exists v \in \mathcal{N}. p_v = u$ and $t_v = \mathcal{O}(u)$ do
        $u = v$
        $\mathcal{V}$.append($t_v$)
    $\mathcal{V}$.append($\mathcal{O}(u)$)
    return $\mathcal{V}$

function VERIFYSTOCHASTIC($\mathcal{O}, \mathcal{N}$)
    $\mathcal{V} = \emptyset$, $u \leftarrow$ the root of token tree $\mathcal{N}$
    while $u$ is a non-leaf node do
        $\mathcal{H}$ = child($u$)          ▷ The set of child nodes for $u$
        while $\mathcal{H}$ is not empty do
            $s \sim \text{rand}(\mathcal{H}), r \sim U(0,1), x_s = \mathcal{H}[s]$
            if $r \le P(x_s \mid u, \Theta_{LLM}) / P(x_s \mid u, \Theta_{SSM_s})$ then
                ▷ Token $x_s$ passes verification.
                $\mathcal{V}$.append($x_s$)
                $u = s$
                break
            else
                ▷ Normalize the residual $P(x \mid u, \Theta_{LLM})$
                $P(x \mid u, \Theta_{\text{LLM}}) := \text{norm}(\max(0, P(x \mid u, \Theta_{\text{LLM}}) - P(x \mid u, \Theta_{\text{SSM}_s})))$
                $\mathcal{H}$.pop($s$)
        if $\mathcal{H}$ is empty then
            break
    ▷ All SSMs fail verification; sample the next token
    $x_{\text{next}} \sim P(x \mid u, \Theta_{LLM})$
    $\mathcal{V}$.append($x_{\text{next}}$)
    return $\mathcal{V}$
```

SpecInfer: Evaluation

- 2 draft models, speculation length=16
- Verify mode is not mentioned



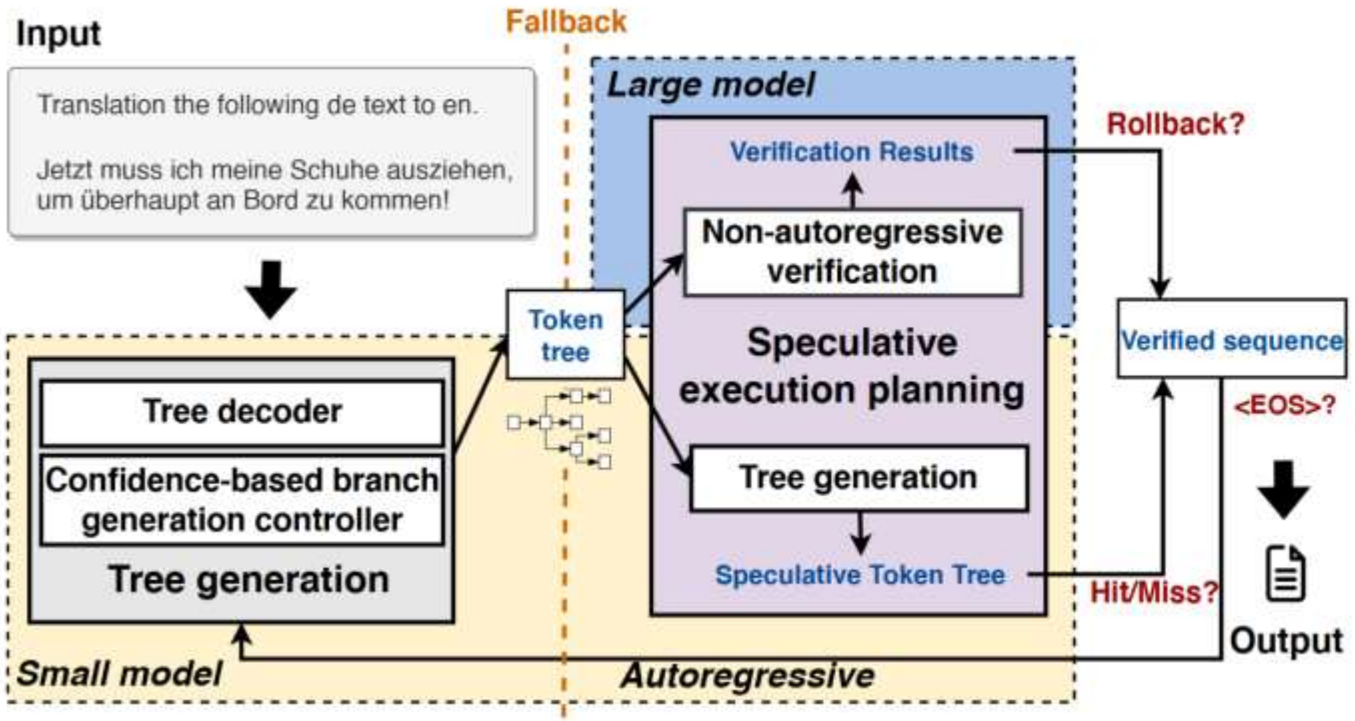




# SSMs	1	2	3	4	5
CIP	3.35	3.74	3.97	4.05	**4.11**
CP	2.71	3.14	3.32	3.45	**3.51**
WebQA	2.84	3.08	3.20	3.27	**3.31**
Alpaca	2.70	3.19	3.36	3.44	**3.49**
PIQA	2.98	3.21	3.36	3.44	**3.49**
Avg	2.92	3.27	3.44	3.53	**3.58**

LLMCad

- On-device LLM inference
- Use one draft model to generate token tree

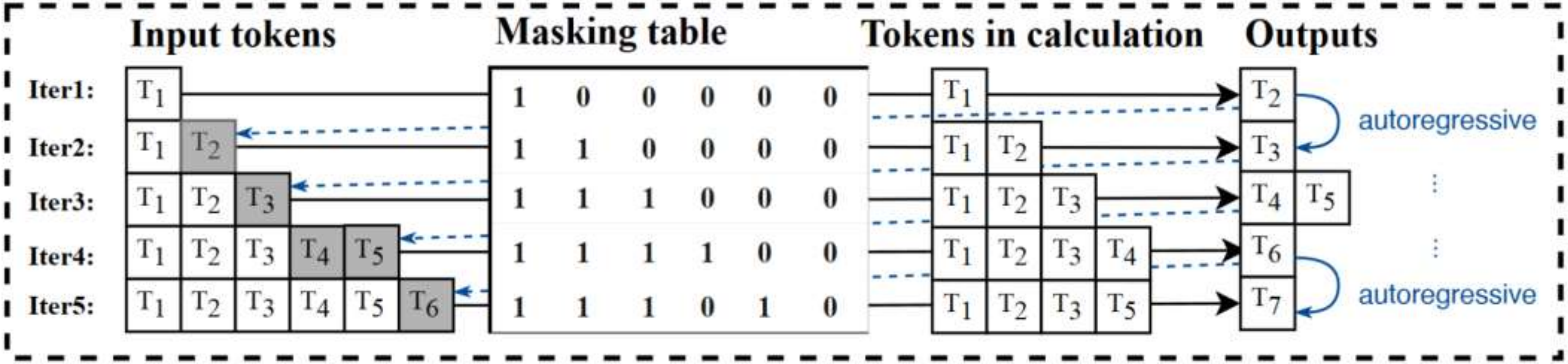


Translation the following de text to en.

Jetzt muss ich meine Schuhe ausziehen, um überhaupt an Bord zu kommen!

LLMCad: Decoding

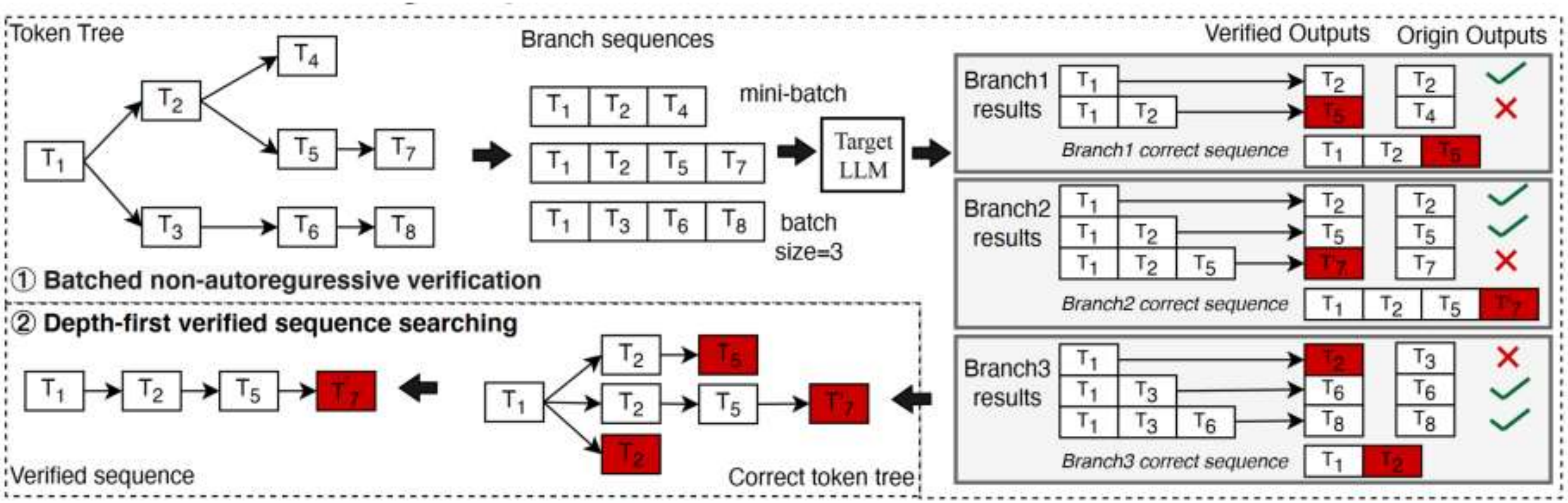
- Only one draft model to sample tokens
- Decide which branch to continue by $f(x) = M * \frac{C_x}{\sum_{i=0}^{N} C_i} - T_x^B$
- Get the branch’s context by a mask



1	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0
1	1	1	1	0	0
1	1	1	0	1	0

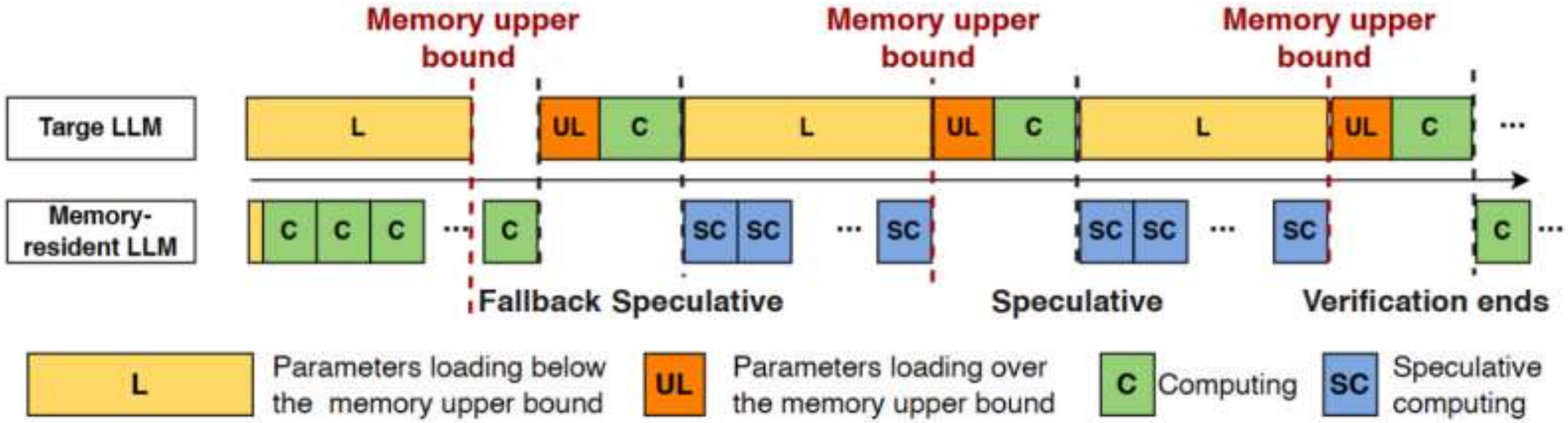
LLMCad: Verification

- self-adaptive threshold $\alpha_{i+1} = \begin{cases} \alpha_i * 0.5 & if\ N_{correct} == N_{all} \\ \alpha_i / T_c^{\frac{N_{all} - N_{correct}}{N_{all}}} & if\ N_{correct} < N_{all} \end{cases}$
- Verify when $T_c = max_{i=1}^{N_c} C_i < \alpha$
- Not tree mask but mini-batch



LLMCad: Speculative Generation Pipeline

- Allow draft model to decode in the verification process
- To avoid two LLMs memory contention, parameters loading will stop before the memory upper bound is exceeded
- Speculative execution only when below the memory budget



Other work

- Multi-stage draft models
- Medusa: Blockwise decoding + Token-tree
- Lookahead Decoding: Jacobi decoding + n-grams
- …

Conclusion

- Most of current work in this direction is still focusing on algorithm
- The speculative decoding algorithm itself is not yet fixed
- Need to do some profile to find some points of improvement

Reference

- Fast Inference from Transformers via Speculative Decoding
- SpecInfer: Accelerating Generative Large Language Model Serving with Speculative Inference and Token Tree Verification
- LLMCad: Fast and Scalable On-device Large Language Model Inference