BumbleBee: Dynamic KV-Cache Streaming Submodular Summarization for Infinite-Context Transformers

Lilly Kumari[◇ ‡] Shengjie Wang[◇ †] Anthony Rowe[§] Jeff Bilmes[◇]

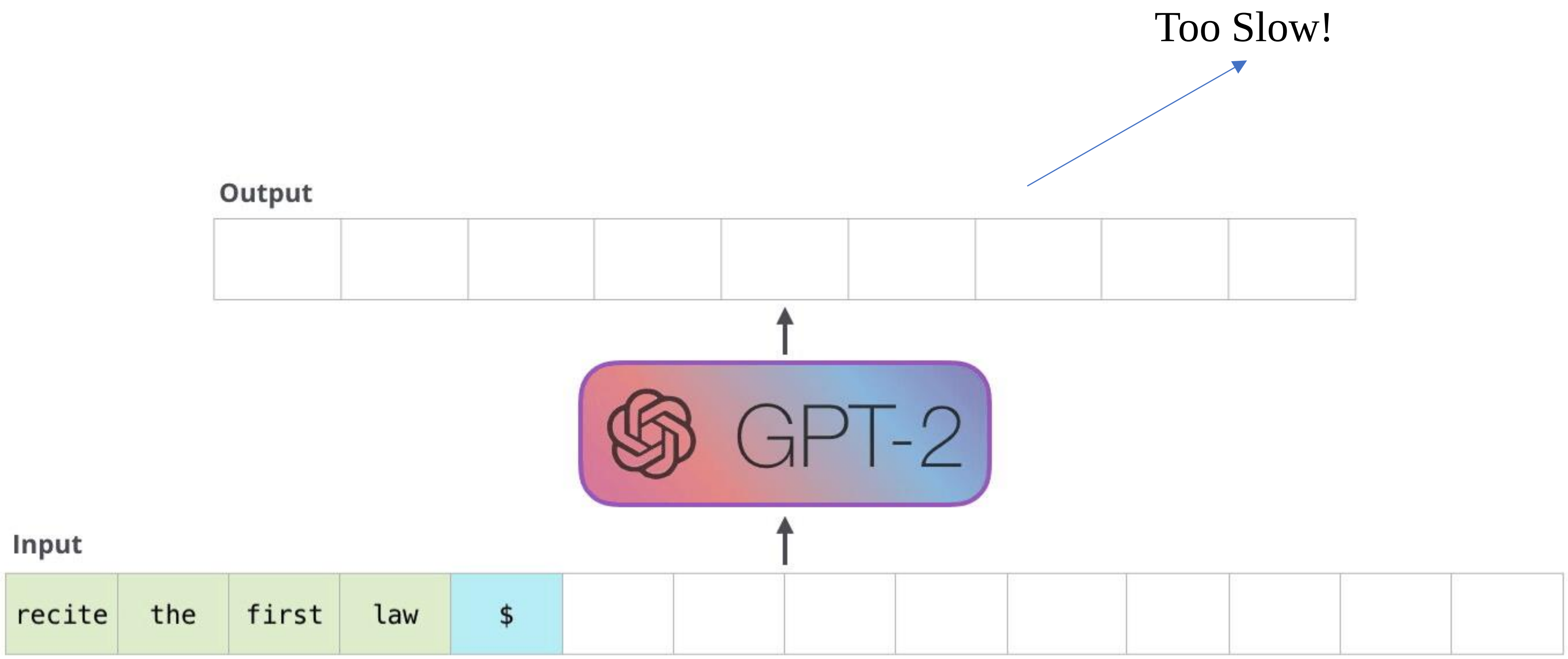
[◇]University of Washington, Seattle

[†]NYU Shanghai

[‡]Google

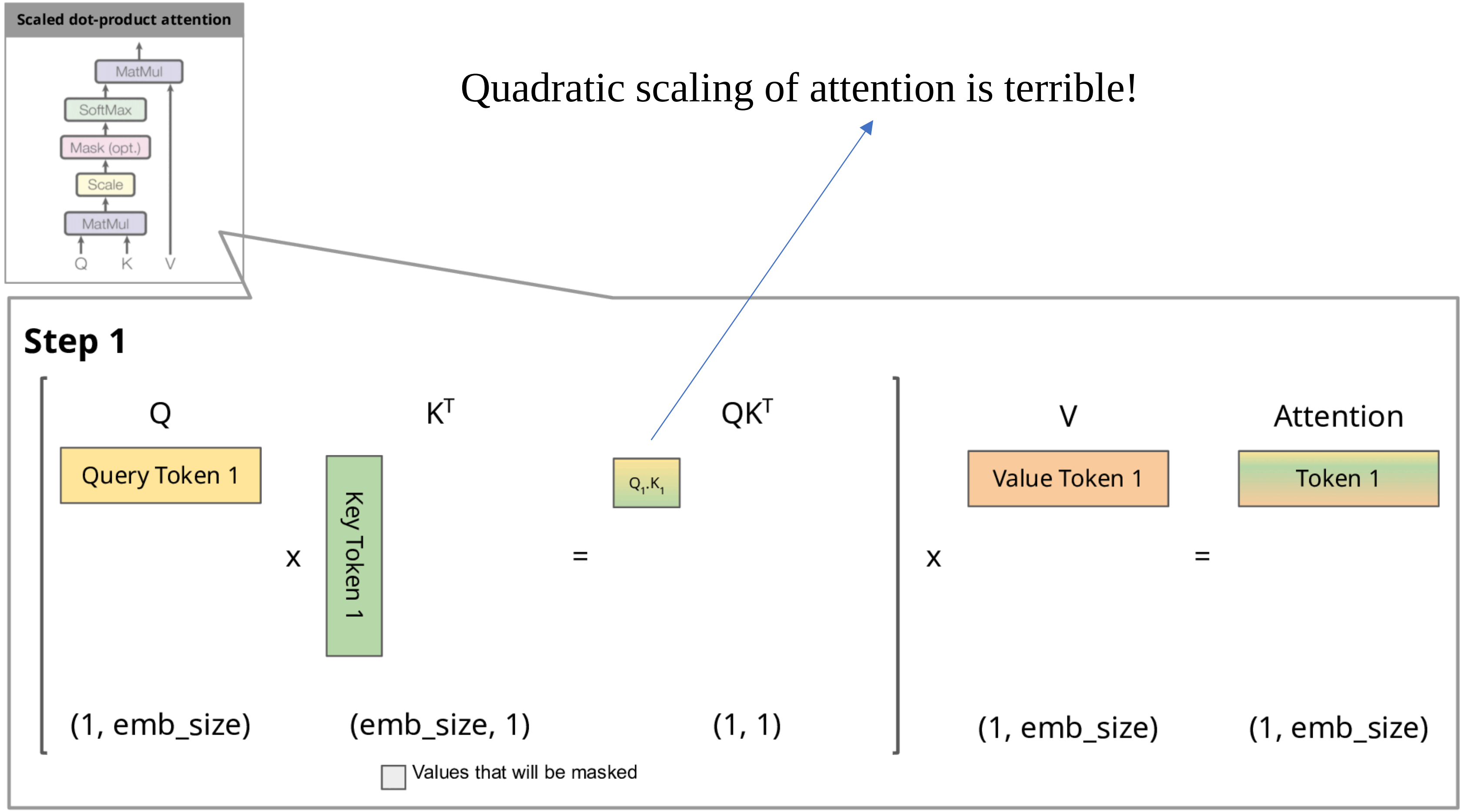
[§]Carnegie Mellon University

What’s the problem with deploying LLMs???



The auto-regressive generation of the decoder.

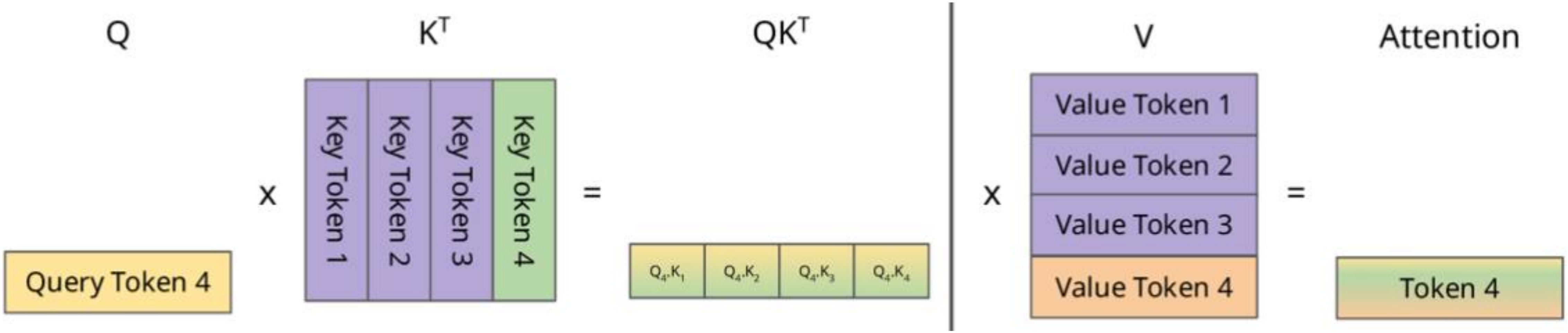
Quadratic scaling of attention is terrible!



Zoom-in! (simplified without Scale and Softmax)

Step-by-step visualization of the scaled dot-product attention in the decoder.

What is KV Cache???



Scaled dot-product attention with KV caching

On a Google Colab notebook, using a Tesla T4 GPU, these were the reported average and standard deviation times, for generating 1000 new tokens:

with KV caching: 11.885 +- 0.272 seconds
without KV caching: 56.197 +- 1.855 seconds

Problems for using KV Cache to store the entire sequence???

For Caching KV States (for a certain LLM), we need Memory:

2 * (hidden_size) * (num_layers) * (decoded_length) * size_of (type)

Fixed Fixed Increasing Fixed

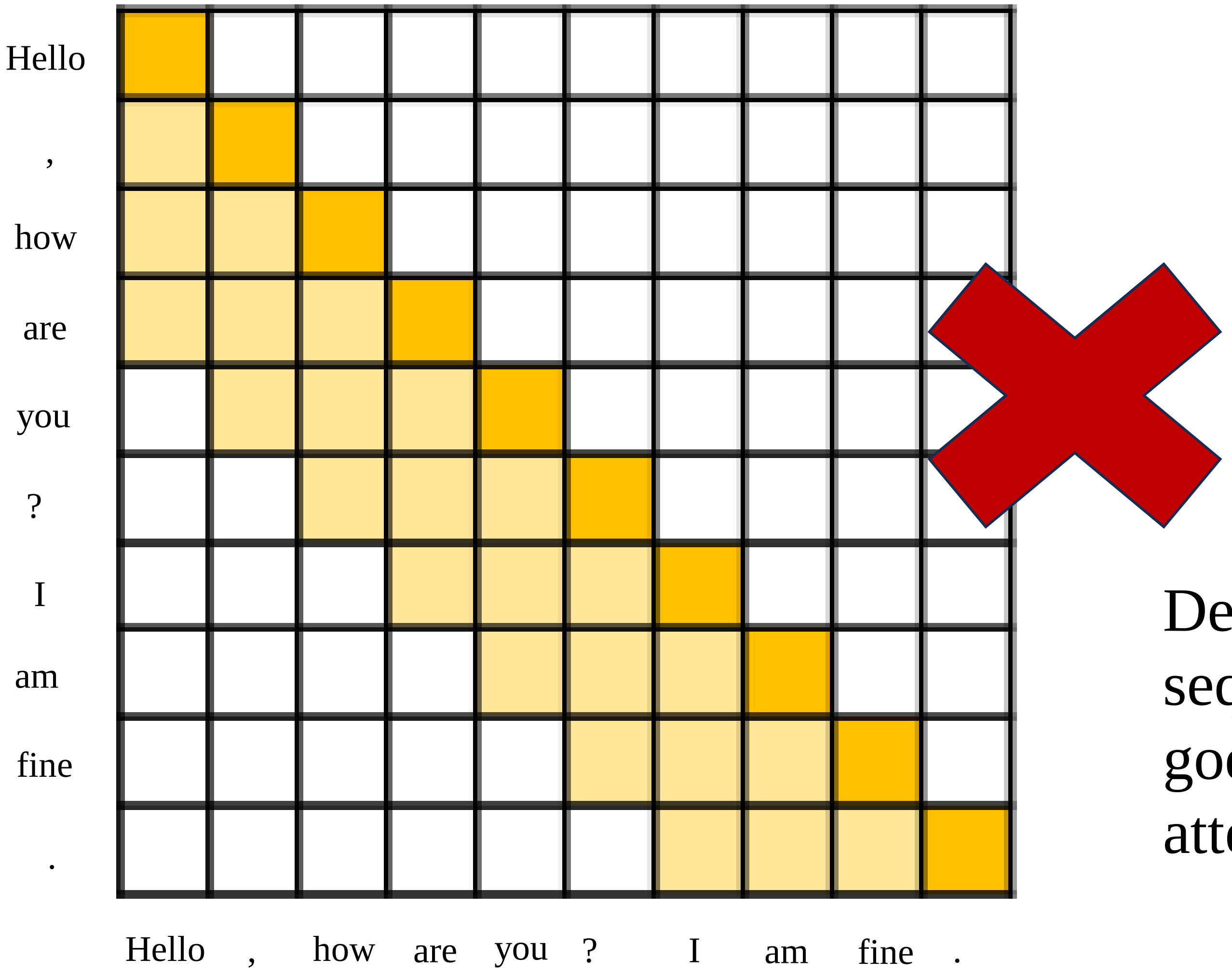
In a long Context situation, decode length is very large!

“a 30 billion-parameter model with an input batch size of 128 and a sequence length of 1024 results in 180GB of KV cache”

As decode goes by, decoded_length will keep accumulating util out of memory!

Some ideas to solve this problem.

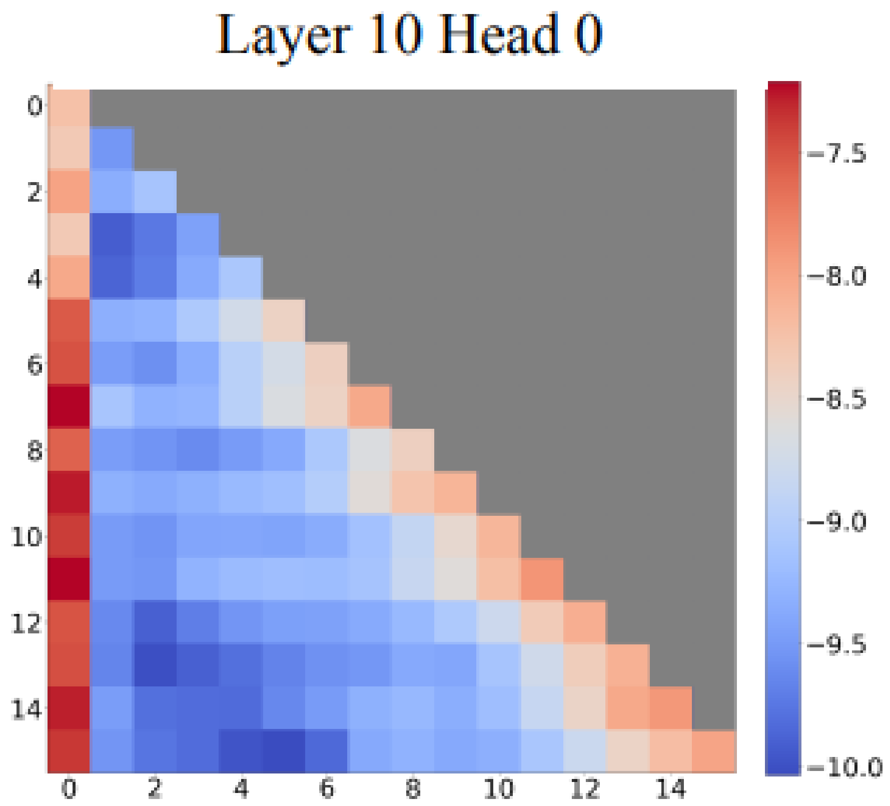
What if we use few KV State to present the entire KV States?

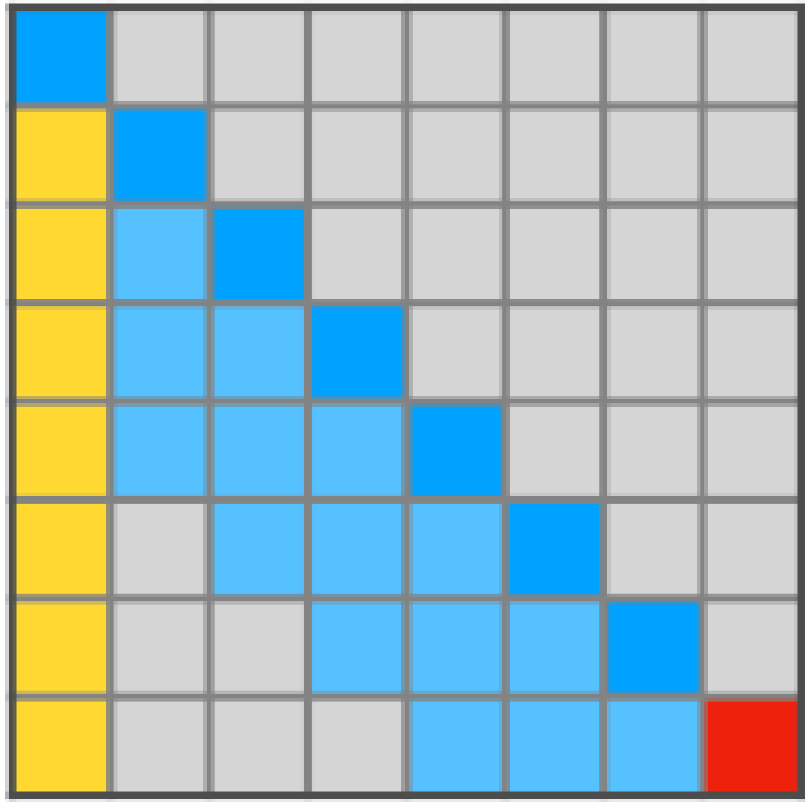


Degrade when the sequence length goes beyond the attention window!

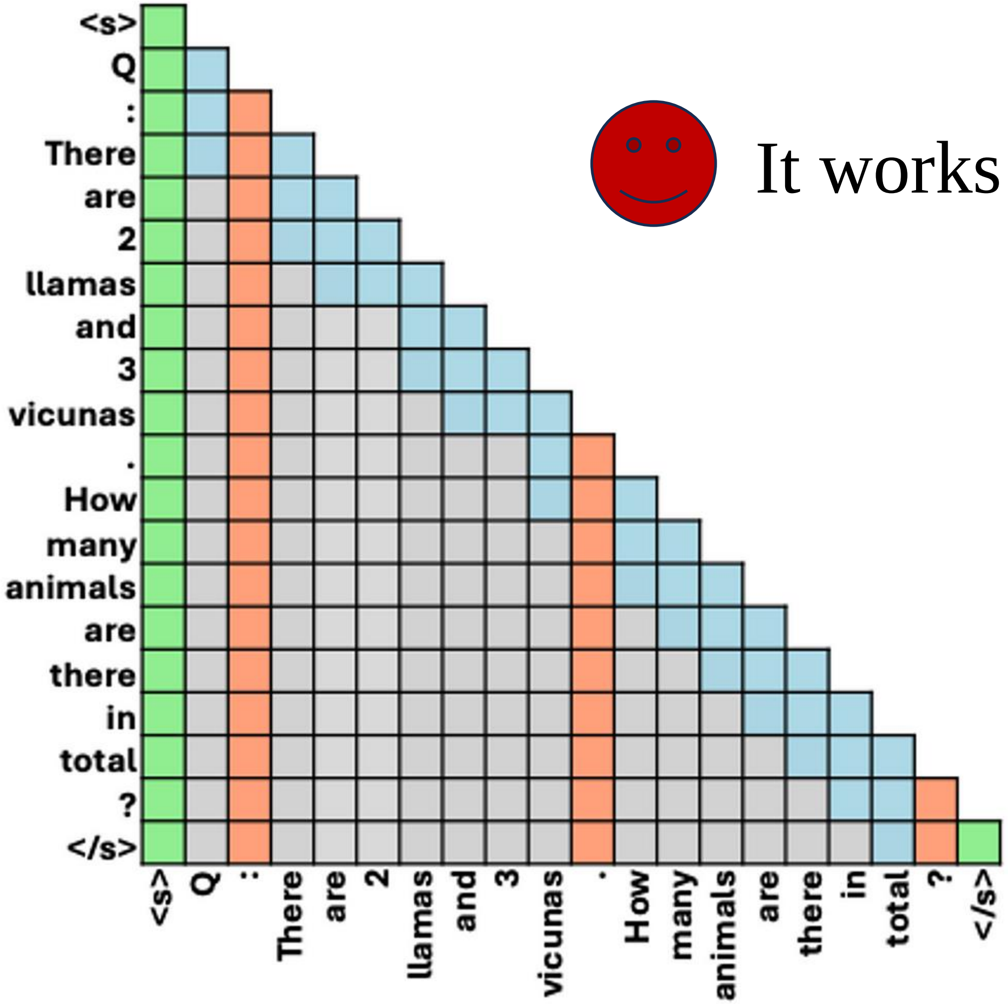
Longformer : Window Attention

It works but not good enough





StreamingLLM : A lot of attention allocated
to the first token & last neighboring tokens



It works but not good enough

FastGen : Special tokens (green) + Punctuation tokens (orange) + Local attention (blue)

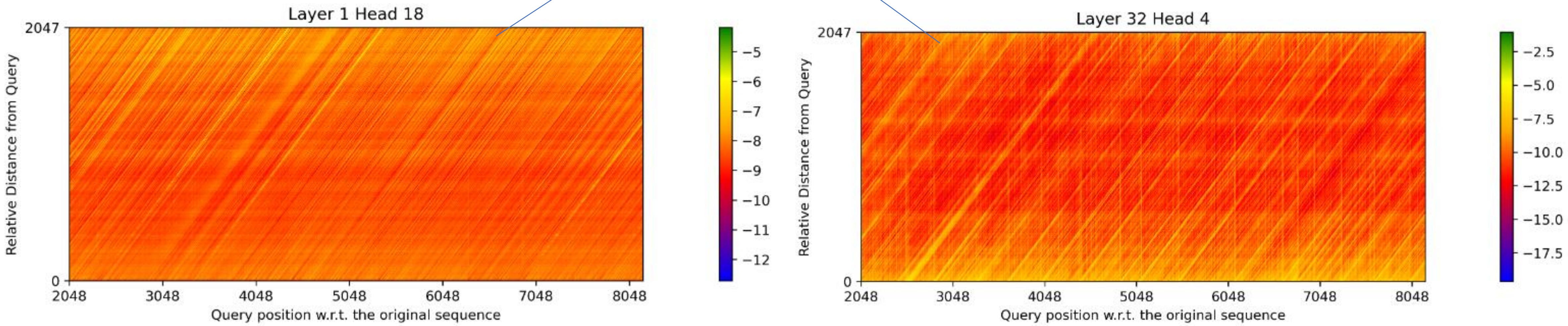
A Hypothesis

"self-attention is selective,

and keeping only a set of both diverse and important tokens is sufficient to maintain performance."

Motivation

The anti-diagonal pattern shows that there is a small subset of tokens that are strongly attended



use a LLaMA-7B model for a next-token prediction task on randomly sampled articles from wikitext-103

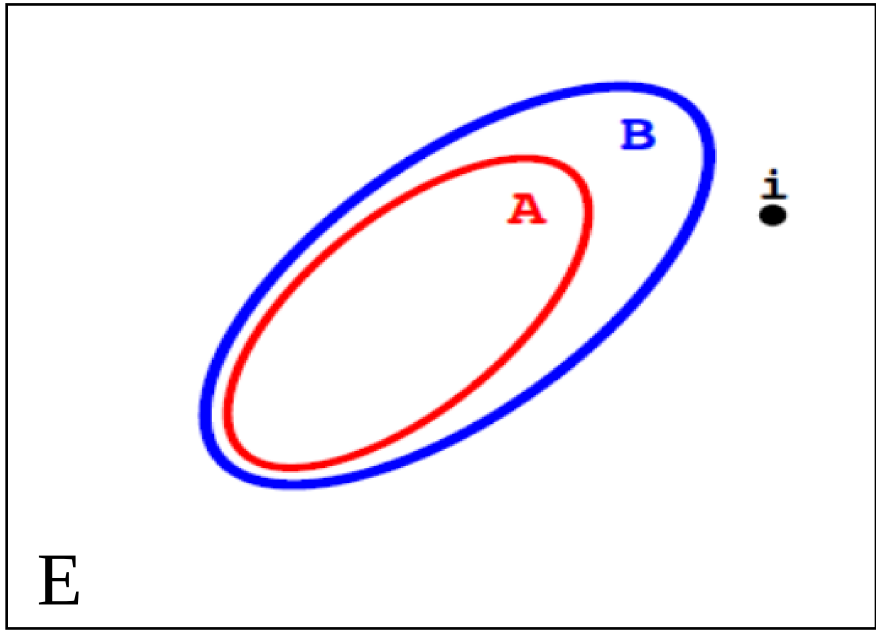
How to choose a set of both diverse and important tokens???

Submodular function

A function f is submodular if for all $A \subseteq B \subseteq E$,

we have $f(B \cup \{i\}) - f(B) \leq f(A \cup \{i\}) - f(A)$

for all $i \in E \setminus B$.



Two submodular function used as one

$$f_{\mathrm{FL}}(A) = \sum_{v \in \mathsf{V}} \max_{v' \in A} \mathrm{sim}(v, v').$$

Emphasize diversity

$$c(A) = \sum_{u \in U} \phi_u\Big(\sum_{v \in A} m_u(v)\Big).$$

Emphasize relevance/importance

$$g(A) = \lambda\, f(A) + (1 - \lambda)c(A)$$

We can use Submodular function to select tokens

B can be seen as KV Cache (with fixed max size N)

E can be seen as Past Tokens

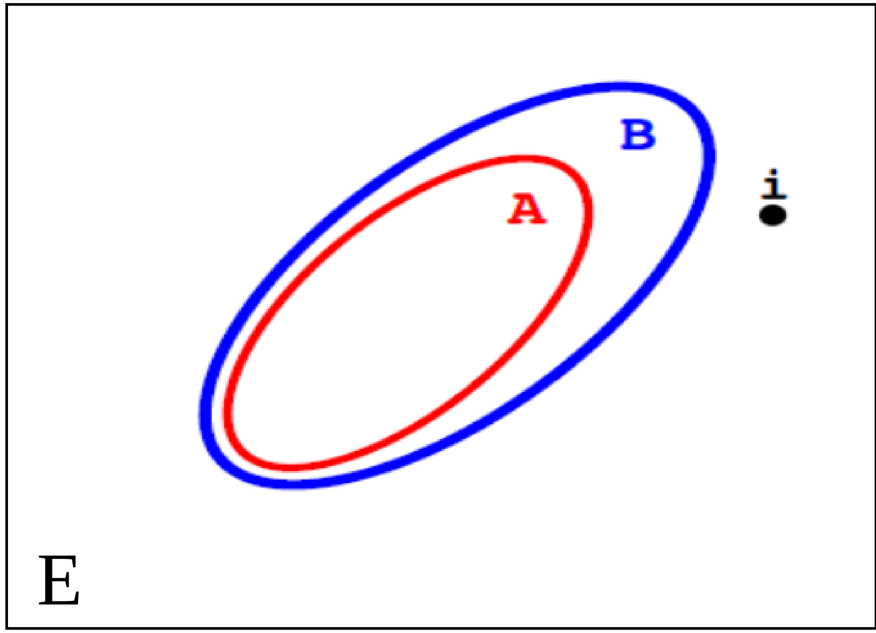
Every iteration we need to choose a token i and put it into KV Cache and discard that:

argmax $g(B \cup i) - g(B)$

If $|B| = N+1$

Choose a token i from B which

argmin $g(i|B/i)$



Algorithm 1 Offline Submodular KV cache Summarization during Prefill/Encoding Phase

Input: Submodular functions capturing diversity f_{FL} in the key embeddings space and importance c via attention frequency for layer l and attention head h; mixture function $g_\lambda(\cdot) = \lambda f_{\text{FL}}(\cdot) + (1-\lambda)c(\cdot)$; a set of n KV attention states $K_n = \{(k_i)\}_{i=1}^n, V_n = \{(v_i)\}_{i=1}^n$ corresponding to the n prompt tokens; budget τ_s.

Output: A final summary S_n such that $S_n \subseteq \{(k_i, v_i)\}_{i=1}^n$ and $|S_n| \leq \tau_s$.

Initialize: $S_n = \emptyset$; compute accumulated attention score vectors a_n for each key $k \in \{k_i\}_{i=1}^n$. a_n^i denotes accumulated attention scores attributed to key k_i across all n query tokens.

for $j = 1$ to τ_s **do**

  $k_{\text{imp}} \leftarrow \text{argmax}_{e \in K_n \setminus S_n} g_\lambda(S_n \cup e) - g(S_n)$

  $S_n \leftarrow S_n \cup \{(k_{\text{imp}}, v_{\text{imp}})\}$ where v_{imp} is the value embedding associated with k_{imp}.

end for

Algorithm 2 *BumbleBee*: Streaming Submodular KV cache Summarization for Transformers

Input: Submodular functions for diversity f_{FL} in the key embeddings space and importance c w.r.t. attention frequency resp. for layer l and attention head h; mixture function $g_\lambda(\cdot) = \lambda f_{\text{FL}}(\cdot) + (1-\lambda)c(\cdot)$; stream of QKV attention states $\{(q_i, k_i, v_i)\}_{i=1}^{n}$; budget τ_s.

Output: A running summary S_t of for every time step t such that $S_t \subseteq \{(k_i, v_i)\}_{i=1}^{t}$.

Initialize: $S_0 = \emptyset$, $a_0 = \emptyset$ where $a_t \in \mathbf{R}^{|S_t|}$ denotes the accumulated attention scores corresponding to keys present in S_t across t time steps.

for $t = 1, \ldots, n$ **do**
  Update a_t for each $k \in S_{t-1}$ by adding $a(q_t, k, S_{t-1} \cup k_t)$
  if $t < \tau_s$ **then**
    $S_t \leftarrow S_{t-1} \cup \{(k_t, v_t)\}$
    Append $a(q_t, k_t, S_t)$ to a_t s.t. $|a_t| = |S_t|$
  else
    Let $S'_t = S_{t-1} \cup \{(k_t, v_t)\}$; $k_{\text{discard}} \leftarrow \operatorname{argmin}_{k_i \in S'_t} g_\lambda(k_i | S'_t \setminus k_i)$
    $S_t \leftarrow S'_t \setminus \{(k_{\text{discard}}, v_{\text{discard}})\}$
    if $k_{\text{discard}} \neq k_t$ **then**
      Evict a_t^j (the accumulated attention score for the discarded key k_{discard}) from a_t.
      Append $a(q_t, k_t, S_t)$ to a_t
    end if
  end if
end for

Experiments

Datasets & Tasks

1. lm-eval-harness (six few-shot datasets):

(1) OpenbookQA (Mihaylov et al., 2018)	Commonsense Reasoning and Science Knowledge QA.
(2) COPA (Roemmele et al., 2011)	Causal and Teleological Reasoning.
(3) RTE (Wang et al., 2018)	Recognizing Textual Entailment.
(4) MathQA (Amini et al., 2019)	Mathematical Problem Solving.
(5) PiQA (Bisk et al., 2020)	Physical Commonsense Reasoning.
(6) Winogrande (Sakaguchi et al., 2021)	Pronoun Resolution and Complex Reasoning.

2. LongBench (long-context understanding):

(1) Qasper (Dasigi et al., 2021) and MultiFieldQA — Single document question answering

(2) HotpotQA (Yang et al., 2018) and 2WikiMultihopQA (Ho et al., 2020) — Multi-document question answering

(3) QMSum (Zhong et al., 2021) — Summarization

(4) TREC (Li & Roth, 2002) — Few-shot learning

3. HELM (document summarization):

XSUM (Narayan et al., 2018)

1. lm-eval-harness (six few-shot datasets):

(♥) log-based: $\varphi(x) = \log(1 + x)$
(♦) power-based: $\varphi(x) = g-1(x)$ where $g(y) = \alpha y1/\alpha + \beta y$

Model	Methods	OpenBookQA	COPA	RTE	MathQA	PiQA	Winogrande
LLaMA-13B	**All**	47.4	85	73.28	31.86	80.36	75.69
	Local	28.4	64	53.43	23.25	58.32	49.88
	Random + Local	27.6	58	54.63	21.76	54.13	50.64
	Attn Sinks + Local	44.4	80	67.51	29.78	79.22	70.48
	H2 + Local	44.2	83	64.98	29.71	**79.49**	70.32
	BumbleBee ♥	**47.6**	**85**	**71.48**	**31.02**	79.38	71.98
	BumbleBee ♦	46.6	83	67.15	30.82	**79.49**	**73.01**
LLaMA-7B	**All**	44.6	81	68.95	29.85	80.03	71.51
	Local	28.4	56	50.90	23.02	58.27	51.38
	Random + Local	28.0	63	51.26	21.76	53.94	49.30
	Attn Sinks + Local	41.6	**82**	58.12	27.40	78.07	67.80
	H2 + Local	41.4	78	63.54	27.50	77.31	65.82
	BumbleBee ♥	**43.2**	79	**68.95**	27.74	78.24	**68.75**
	BumbleBee ♦	**43.2**	79	63.90	**28.51**	**78.56**	68.19

KV cache summarization budget is 0.1× the input sequence length

Matrix : accuracy

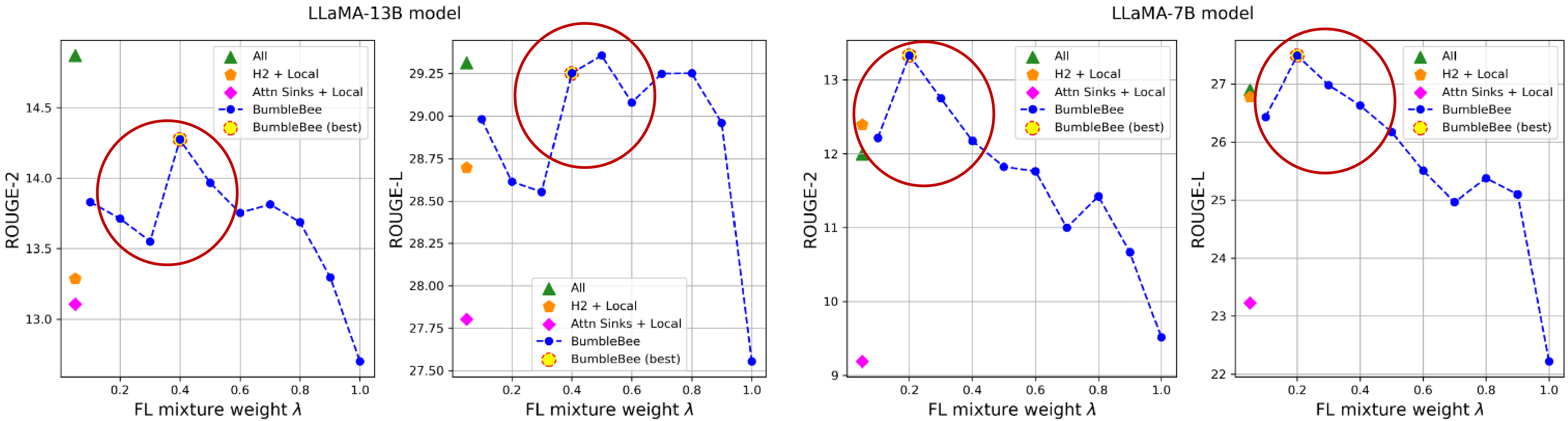
2. LongBench:

Model	Method	Qasper	MultiFieldQA-en	HotpotQA	2WikiMQA	QMSum	TREC
LLaMA-7B-chat 4k	**All***	19.20	36.80	25.40	32.80	20.80	61.5
	All (self)	21.60	36.76	27.55	31.58	20.78	64.0
	Attn Sinks + Local	14.74	22.93	22.08	29.73	19.25	56.0
	H2 (20%)	**19.82**	26.60	26.28	25.69	**21.45**	60.0
	BumbleBee (20%) ♥	19.37	27.73	26.14	27.67	20.68	**61.5**
	BumbleBee (20%) ♦	19.59	**28.60**	**28.99**	**30.19**	21.05	59.0
LongChat-7B 32k	**H2** (SW, 20%)	21.64	30.72	14.07	15.10	18.11	40.5
	BumbleBee (SW, 20%) ♦	**23.27**	**33.16**	**22.52**	**17.58**	**20.27**	**44.5**

KV cache summarization budget is 0.2× the input sequence length

Matrix : F1 score for Qasper, MutiFieldQA-en, HotpotQA, 2WikiMQA
Rouge-L and accuracy for QMSum and TREC

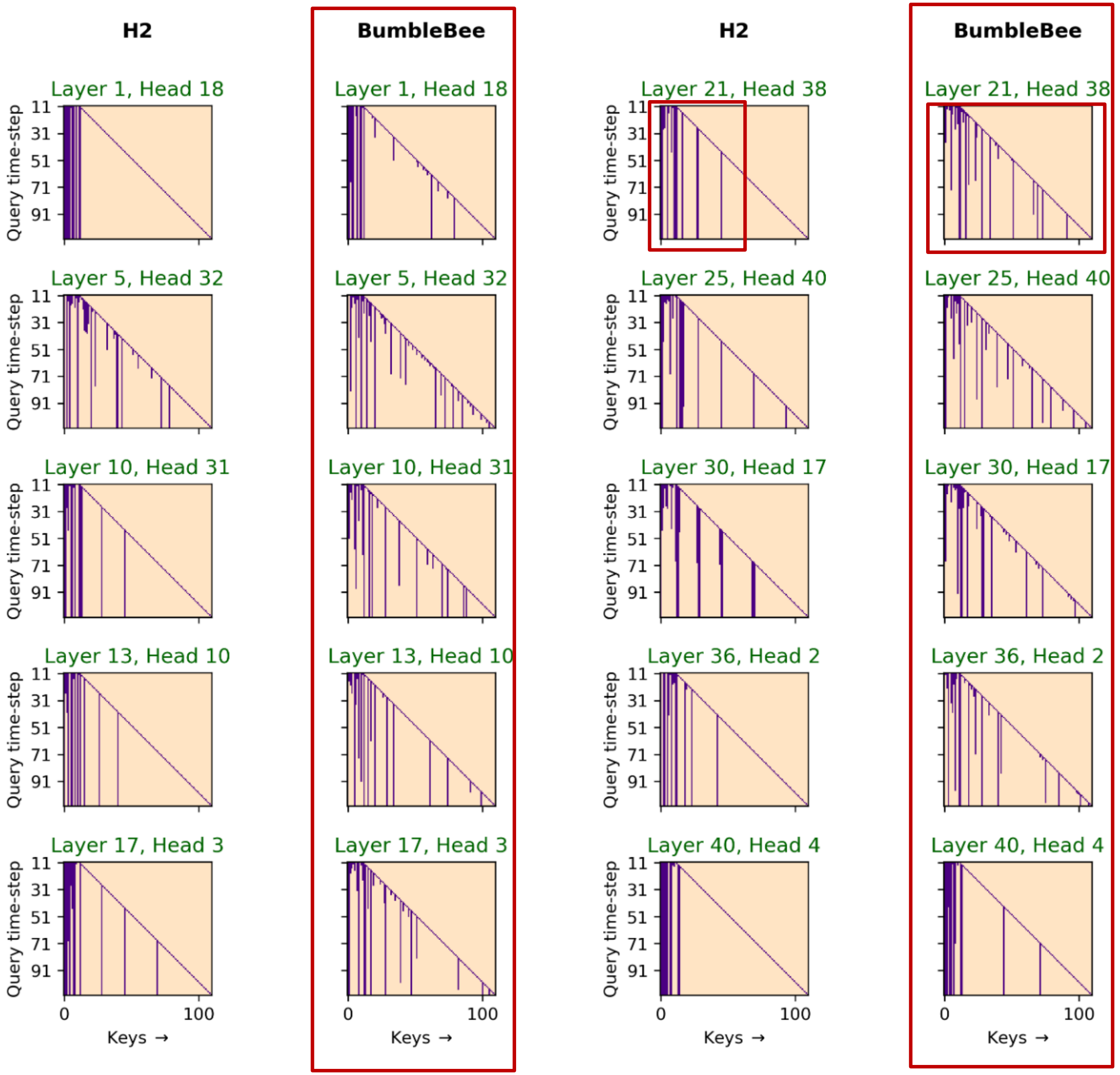
3. XSUM Summarization:



KV cache summarization budget is 0.2× the input sequence length

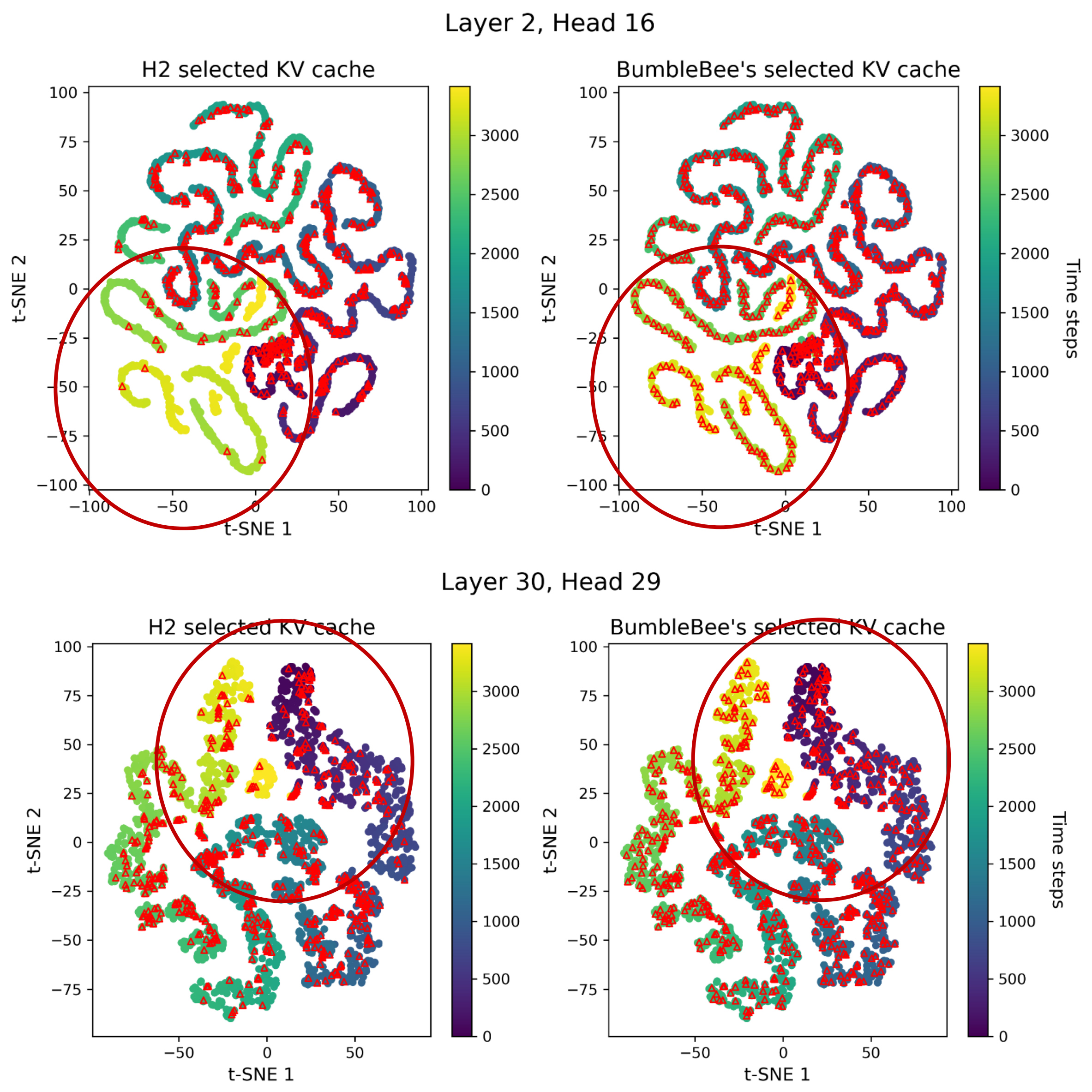
$$g(A) = \lambda\, f(A) + (1 - \lambda)c(A)$$

4. Visualization of chosen KV pairs



Sample selected from the test set of COPA (Roemmele et al., 2011)

4. Visualization of chosen KV pairs via t-SNE



Samples chosen from the 2WikiMultihopQA (Ho et al., 2020) dataset from the LongBench task with LLaMA-7B-chat-4k model

5. decoding speed

Context reduction ratio	Original Context Length	
	16k	**100k**
1:1	59.30 ± 0.39	OOM
5:1	47.49 ± 4.16	71.50 ± 0.10
10:1	39.74 ± 1.31	48.16 ± 0.09

Decoding speed (in ms/token). All experiments are performed on an A100 80GB GPU using the LongChat-7B-32k with a batch size of 1.

Conclusion

1. The key idea of this paper : utilize Submodular Function to select the most diverse and important N KV pairs.

2. The computation of g(·) seems fine, with O(s^2*d) time.

3. The lm-eval-harness datasets evaluation is based on the Llama-1 7B/13B models, and it is unclear whether the results would generalize to newer, stronger models.

4. Experiment 3 suggests that performance can be fairly sensitive to λ and model size, generalization to different tasks and settings is unknow.

Some ideas

What situation can Submodular Function be utilized?

1. Recommend System

2. Select diverse and meaningful data for model training

Thank you for your listening!