Maciej Besta, Lorenzo Paleari

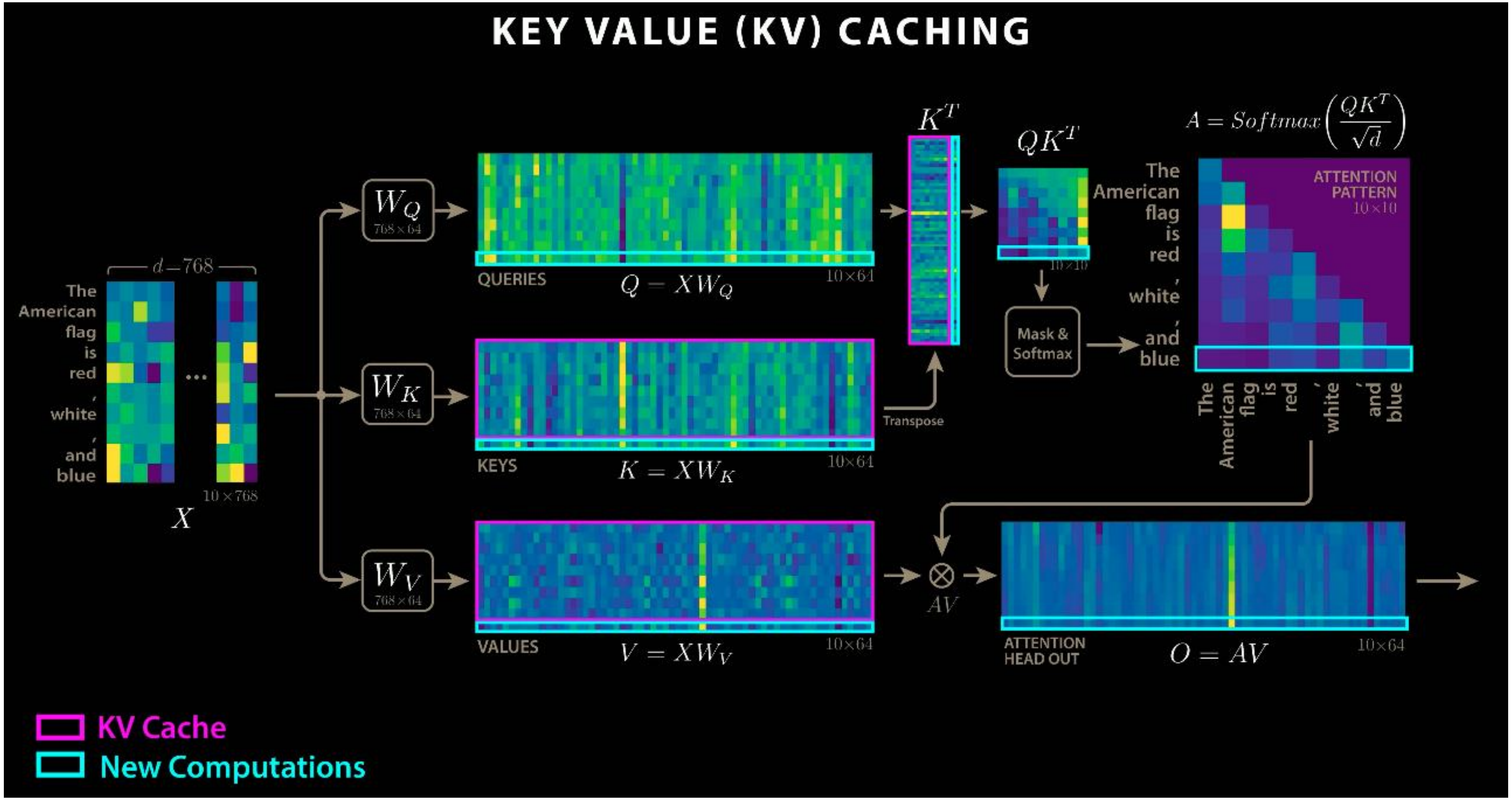
15.10.2025

Graphs & LLMs: Synergy

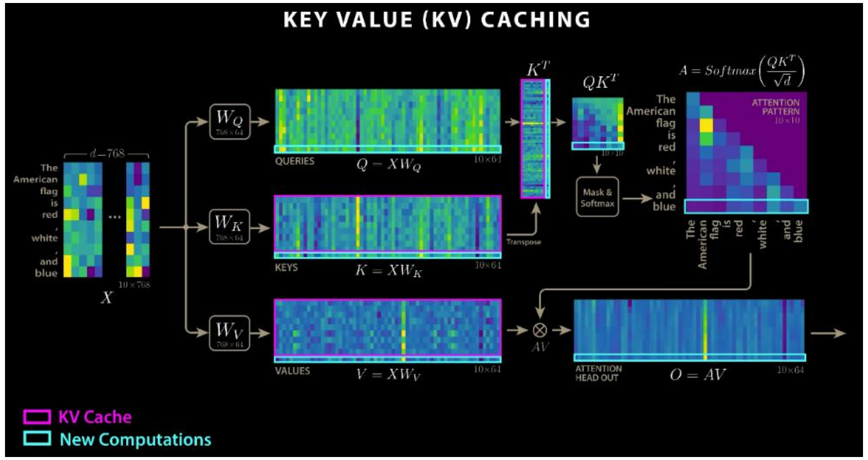
KV Cache – Exploration Summary

- Motivation
- System prompt compression
- Positional embedding
- KV cache distillation
- Benchmark Analysis
- Next Steps

KV Cache – Motivation



KV Cache – Motivation



Objective
- Faster Inference

How we obtain this
- Caching Key and Values Matrices
- Save computation

KV-Cache Downside
- Huge amount of memory consumption
  - $2\ x\ N_{head}\ x\ d_{head}\ x\ N_{layers}\ x\ C_{length}\ x\ 2\ (16\ bit)$
  - Memory needed
  - Data movement needed
- What happens when it does not fit in a GPU?
  - Ring Attention
    - Many GPUs used → Reduce number of GPUs
    - Reduce Communication Overhead
- **How do we reduce KV-Cache, while still maintaining same performance/expressiveness?**

How much computation we save in an ideal setting – B=1
- $X \times W_K,\ X \times W_V \rightarrow 2 \times (C_{length}, d_{model}) \times (d_{model}, d_{head}) = C_s \rightarrow O(C_{length} \times d_{head} \times d_{model}))$
- Per each Head and Layer → $C_s \times L \times H$
- Per each Forward Pass
- What is left to be computed for each head are 6 VMM→$O(C_{length} \times d_{head})$

KV Cache – Motivation

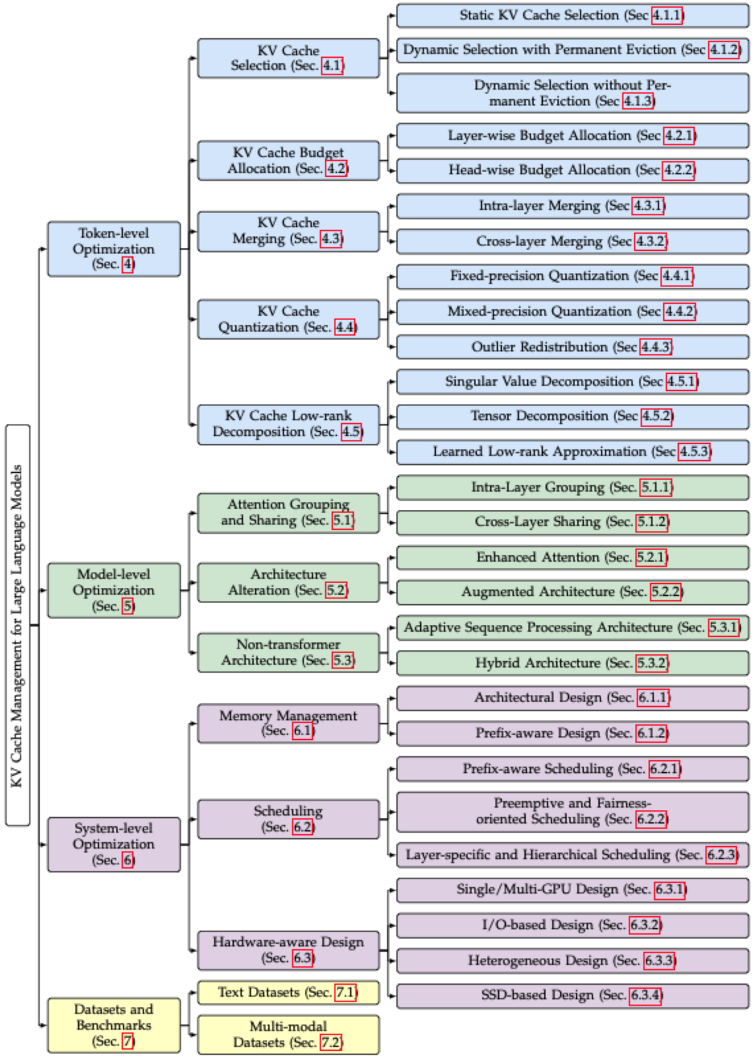


Fig. 2. Taxonomy of KV Cache Management for Large Language Models.

- Token-Level Optimizations
- Model-Level Optimizations
- System-Level Optimizations

- Token-Level
  - Selection / Merging / Quantization / Low-Rank Reduction
  - Fine-Grained focus
  - No-architectural changes
  - **Always Applicable**
- **Model-Level**
  - Grouping / Sharing / Architectural Changes
  - Model-Structure Changes – Transformer / Attention
  - **NOT Always Applicable**
- System-Level
  - Memory Management / Scheduling / Multi-GPU / I-O improvements
  - vLLM – Paged Attention
  - Block-Wise Attention / Ring Attention / Flash Attention

KV Cache – Motivation

Some quick examples of prominent method and where they fail ?

KV Cache – Overview

Quick overview over all the topics discussed In later slides.

All work directions we explored
- System Prompt
- Positional Embedding
- KV Cache distillation
- Benchmark analysis

KV Cache – System Prompt Compression

- System prompts can be huge (10k+ tokens for Claude)
- **We may want to add ’User Memories’ like OpenAI**
  - Addition of many tools usually sits in the system prompt
  - Valid point also for Open-Source models --> No System Prompt

- Would really be useful to have a System Prompt compression mechanism, but is this novel?
  - Rope issues

KV Cache – System Prompt Compression

- We have a long ongoing chat with an Agent/LLM.
  - The longer the chat becomes (if it remains in context), the larger the KV cache. We can compress older segments asynchronously while the conversation continues.
  - We can use the processing power of GPUs. While they are used for forward passes (Memory Bound).

KV Cache – Positional Embedding

- RoPE
  - SOTA for LLMs now. RoPE encodes absolute positions via rotation matrices and embeds relative positions into the attention mechanism.
  - If we modify User Memories in the middle of the prompt --> Delta re-computation of all tokens after the change must be done. Not much expensive, but we can do better.
- **Hierarchical Positional Embedding**
- Multi Layers RoPE or Hybrid approach. Based on the System Prompt mainly.
    - *Instructions*
    - *User Memories*
    - *Tools*
    - *Other Instructions*
    - …
  - Each chunk has a Learned/RoPE positional embedding scaled by some factor to make it heavier. Tokens inside the same chunk follows a “standard” RoPE embedding.
  - Retraining if you change Chunk positioning? If learned yes, but at Fine-Tuning.
    - *We can use a different rotation plane to embed chunk positions inside rotations*

KV Cache – Distillation

- How to make the model learn at this point?
  - Training – We need data that we are missing, and it is not available
  - LoRA – Can be used, but needs a little data to work
  - The problem can be directly solved → mimic GPTQ paper

- GPTQ paper
  - They solve the problem of efficient quantization by solving the underlying equation problem.
  - This allow very fast and efficient computation
  - We can follow same step and expand the math to our problem

$$\operatorname{argmin}_{\widehat{\mathbf{W}}} ||\mathbf{W}\mathbf{X} - \widehat{\mathbf{W}}\mathbf{X}||_2^2.$$

$$w_q = \operatorname{argmin}_{w_q} \frac{(\operatorname{quant}(w_q) - w_q)^2}{[\mathbf{H}_F^{-1}]_{qq}}, \quad \boldsymbol{\delta}_F = -\frac{w_q - \operatorname{quant}(w_q)}{[\mathbf{H}_F^{-1}]_{qq}} \cdot (\mathbf{H}_F^{-1})_{:,q}.$$

$$\mathbf{H}_{-q}^{-1} = \left(\mathbf{H}^{-1} - \frac{1}{[\mathbf{H}^{-1}]_{qq}} \mathbf{H}_{:,q}^{-1} \mathbf{H}_{q,:}^{-1}\right)_{-p}.$$

KV Cache – Benchmarks

Benchmarks

Name	Max. Length	Solved	Type	Comments	Link	How is generated	How question are created / what they aim to
NIHS	1M tested - 10M mentioned by Google Looping 148k token max 110k words	99.7% - Video 10h - Audio 5 days 99.2% on 10Million recall	Search information in Long Context. - Text - Audio - Video Position information at different depth.	Possibility of recalling Multiple NIHS with changes. 100 needle tested with recall of 70% top (128k) No Reasoning at all. Precise wording necessary	Google Cloud The Needle in the Haystack Test and How Gemini Pro Solves It... https://github.com/gkamradt/LLMTest_NeedleInAHaystack arxiv.org	Constructed by taking long essays and inserting a needle statement such as "The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day"	locate or repeat the inserted statement Pure retrieval
Multimodal NIHS	40k images 560k Captions 280k Needles Max number of images is: max_upload x grid_dim	97% 10 2x2 27% 10 4x4	Send some images with sub-images inside. Test wether the model can find the correct image/s given a caption	Constrained by model maximum image upload. Expanded by stitching together images. Stitching has great impact in model result —— Image downscaling? Image tokens?	arxiv.org	Images 1-10 Every image can be created stitching together a varying amount of smaller images	The dataset contains the captions to all the images used
MMMU MMMU-Pro MME MMBench MMT Vibe-Eval	Images Hard Questions Short length most of them. (MMT little longer)	80%+ — 80 80 63 60	Hard question both textual and cross modality.	Mentioned in Multimodal NIHS MMT pretty hard and complete on the testing modalities, going with temporal, reasoning, 3d ecc...	arxiv.org arxiv.org arxiv.org arxiv.org arxiv.org	From college exams, quizzes and text books. 6 Disciplines considered Embed questions in images and remove only textual ones	4 options QA, hard questions requiring expertise Vision is required and 10 options
LongBench LongBenchv2	Pure Text Until 10k token from 8k to 2M	— 63%	Simple multiple choice questions Summarization Code Long Structured data understanding	v2 is more focused also on reasoning, even if always related to QA	arxiv.org arxiv.org	21 databses taken into consideration: research books books documents Code repos 100 annotators where put to the task of finding long context and annotating them rewarding bonuses for longer contextes	QA, multi-doc QA Summary Code completion Dialogues added, in general required to read in long context
InfiniBench	Average of 200k Tokens	50%	retrieval, code debugging, math problems, novel summarization/QA and dialogue		arxiv.org	novels with key-entity replacement, long movie scripts (dialogues), code repositories with inserted bugs, math problems and synthetic retrieval sequences	QA Summary Open questions Multi-hop tracking QA Embedded in long distractors the answers
Ruler	From 4k to 128k more configurablke	96% Gemini	retrieval Multi-hop tracing Aggregation QA		arxiv.org	Context is padded with distracting tokens	QA multi-hop Aggregation Retrieval

Next Steps

- KV Cache Distillation
  - Where we are currently
  - (Maybe in 2/3 weeks we are going to have a couple of first plot of results)
  - Depending on where we are we are going to discuss next steps
    - *For sure we are including Benchmarks we will use*
    - *And the comparison baseline we want to beat*