Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

MATRIX AND VECTOR METHODS FOR LANGUAGE-MODEL EFFICIENCY: DISTINGUISHING EMBEDDING COMPRESSION, TOKEN REDUCTION, AND RAG

Pedro Emílio Amador Salomão
peas8810@gmail.com
UFVJM-MG

Lívio Almeida Amador
livioamador@hotmail.com
University of Gama Filho - RJ

ABSTRACT

Language-model efficiency is often discussed through concepts that are related but not equivalent: token count, embedding dimensionality, parameter count, key-value cache size, and retrieved-context length. This article presents a structured critical review, supported by document-level traceability, of matrix and vector methods used to reduce storage and inference costs in language models and retrieval-augmented generation (RAG) systems. The analysis shows that matrix factorization, adaptive embeddings, and Matryoshka representations compress parameters or vectors but do not automatically decrease the number of input tokens. Direct token reduction requires sequence-level interventions, including appropriate tokenization, selection, prompt compression, gist tokens, and retrieved-context control. KV-cache compression reduces memory and latency without necessarily changing the tokens submitted to the model. We propose a mathematical framework that separates V (vocabulary size), d (embedding dimension), n (sequence length), k (number of retrieved documents or passages), and nKV (cache positions retained). Efficient optimization therefore requires matching each technique to its intended operational metric: token savings, parameter reduction, vector-storage reduction, lower FLOPs, or decoding-memory compression.

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

Keywords: language models; embeddings; prompt compression; token reduction; retrieval-augmented generation; linear algebra.

1 INTRODUCTION

Transformer-based language models process discrete sequences that are converted into dense vectors and subsequently transformed through matrix operations for projection, attention, and linear combination. This processing chain creates distinct cost centers: text is segmented into tokens; each token indexes a row of an embedding matrix; the representations are projected into queries, keys, and values; and decoding retains intermediate states in the KV cache. Although these stages are interdependent, they cannot be treated as a single variable of "consumption" (Vaswani et al., 2017).

Token count is determined by the tokenizer and the textual sequence. Subword methods, such as byte-pair encoding and SentencePiece, control the granularity with which words, morphemes, punctuation marks, and spaces are converted into discrete units (Sennrich, Haddow, and Birch, 2016; Kudo and Richardson, 2018). After tokenization, each identifier is mapped to a vector with d components. Thus, an input containing n tokens produces a representation matrix with n rows and d columns. Increasing d raises the cost per token; by itself, it does not increase n.

This distinction is essential for interpreting the literature. Low-rank factorization, adaptive embeddings, and parameter sharing can reduce model size or the memory used by lexical representations (Baevski and Auli, 2019; Lan et al., 2020). Nested representations can reduce the dimensionality used in vector search (Kusupati et al., 2022). By contrast, prompt compression and context selection operate on the sequence itself and can decrease the number of processed tokens (Jiang et al., 2023; Jiang et al., 2024). KV-cache compression operates on states stored during inference, retaining only positions considered relevant (Zhang et al., 2023; Li et al., 2024).

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

In RAG systems, the problem encompasses both dimensions. Embeddings are used to retrieve documents or passages that are semantically close to the query, whereas the retrieved content is typically converted back into text and concatenated with the generator prompt (Lewis et al., 2020). Efficiency therefore depends on the dimensionality and quantization of the vector index, the number of retrieved candidates, passage length, the reranking stage, and any subsequent context compression. Approaches such as xRAG explore a more direct integration between retrieved embeddings and the model's representational space, replacing long textual contexts with compact representations (Cheng et al., 2024).

Given these distinctions, this article addresses the following question: which matrix and vector methods effectively reduce token count, and which reduce other components of computational cost without changing sequence length? The objective is to construct an operational taxonomy that supports the selection of techniques consistent with the intended metric and prevents unwarranted causal inferences between embedding compression and token savings.

2 MATHEMATICAL FOUNDATIONS AND OPERATIONAL DEFINITIONS

2.1 Tokenization, embedding matrix, and sequence length

Let s denote a text and T a tokenizer. The resulting discrete sequence is x = T(s) = (x1, x2, ..., xn), where n is the number of tokens. Let V denote the vocabulary size and E an embedding matrix with V rows and d columns. Indexing the rows associated with the tokens produces X, the model's input matrix:

$$x = T(s), \quad E \in \mathbb{R}^{V \times d}, \quad X = E[x] \in \mathbb{R}^{n \times d}.$$

In this formulation, n and d are independent variables. More compact tokenization can reduce n, although it may require a larger vocabulary or impair generalization to rare terms. A representation with a smaller d reduces stored values

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

and multiplications per position but retains the same number of discrete identifiers. Therefore, a "smaller embedding" is not synonymous with "fewer tokens."

2.2 Attention and length-dependent cost

In an attention layer, X is multiplied by projection matrices to form queries, keys, and values. For notational simplicity, consider a single attention head:

$$Q = XWQ, \quad K = XWK, \quad V = XWV, \quad A = softmax(QK^T/\sqrt{dh})V.$$

The product QK^T produces an n-by-n matrix. In dense attention, this term introduces quadratic cost with respect to sequence length, in addition to the linear costs associated with the projections. Consequently, reducing n can strongly affect runtime and memory; reducing d also decreases operations, but through a different mechanism (Vaswani et al., 2017).

2.3 Five metrics that should not be conflated

Table 1 - Operational efficiency metrics

Metric	Interpretation
Input and output tokens	Discrete units generated by the tokenizer and decoder. They affect the context window, latency, and, in commercial services, may affect billing.
Model parameters	Trainable or frozen weights, including the embedding matrix. They affect model storage and training.
Vector dimensionality	Number of components in each embedding. It affects index storage, bandwidth, and the cost of per-vector operations.
FLOPs and latency	Number of operations and observed execution time. They depend on n, d, architecture, hardware, batch size, and implementation.
Inference memory	Includes activations and the KV cache. It can grow linearly with the number of retained positions and the number of layers.

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

3 METHOD

A structured critical review of the literature was conducted, with search and document verification performed in July 2026. The objective was not to estimate a pooled effect but to compare mathematical mechanisms that operate on different components of language-model cost. The study was classified as a critical review rather than a PRISMA systematic review because of the heterogeneity of the problems, architectures, metrics, and datasets examined.

Primary repositories and editorial pages of computer-science conferences and journals were consulted, with priority given to ACL Anthology, the Proceedings of NeurIPS, OpenReview/ICLR, and arXiv when the conference version did not provide sufficient detail. Search combinations included the terms "tokenization," "embedding factorization," "adaptive input embeddings," "prompt compression," "soft prompt," "gist tokens," "retrieval-augmented generation," "context compression," "KV cache compression," and "Matryoshka representation."

Studies were included when they presented: (a) an explicit formulation involving matrices, vectors, or position selection; (b) an application to language models, semantic retrieval, or retrieval-augmented generation; and (c) an identifiable operational metric, such as parameters, dimensionality, tokens, FLOPs, latency, memory, or quality. Studies were excluded when "embedding" referred to watermark insertion, steganography, the incorporation of operations into a programming language, temporal reconstruction of neural spike trains, or circuit de-embedding, because these uses are not equivalent to linguistic embeddings.

The synthesis was organized into five classes: tokenization; compression of the embedding matrix or embedding dimensionality; prompt and context compression; vector retrieval and RAG; and KV-cache compression. For each class, the variable directly modified by the technique was identified, and the analysis determined whether

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

the method reduced discrete tokens or another component of cost. Traceability was maintained by linking each technical claim to its corresponding primary source.

4 RESULTS

4.1 Factorization and adaptive embeddings: fewer parameters, not fewer tokens

The lexical matrix E may account for a substantial share of model parameters when V and d are large. A low-rank approximation replaces the full matrix with two smaller matrices:

$$E \approx AB, \quad A \in \mathbb{R}^{V \times r}, \quad B \in \mathbb{R}^{r \times d}, \quad r \ll \min(V,d).$$

The parameter count decreases from Vd to Vr + rd. ALBERT uses factorized parameterization to decouple embedding size from the Transformer's hidden dimension, reducing parameter memory and facilitating scalability (Lan et al., 2020). Adaptive embeddings allocate different capacities according to the frequency of vocabulary items, reducing parameters for rare units and improving training efficiency in the reported experiments (Baevski and Auli, 2019).

These mechanisms alter the representation of each position but do not modify the sequence x = T(s). Therefore, a text that produces n tokens continues to contain n tokens. Savings occur in model storage, memory bandwidth, and some per-token multiplications. This distinction should be stated explicitly in any study proposing "token reduction through embedding optimization."

Matryoshka representations extend this principle to downstream embeddings: the vector is trained so that lower-dimensional prefixes retain useful information. This makes it possible to adapt representation size to search or classification constraints and can accelerate vector retrieval without training a separate model for every dimensionality (Kusupati et al., 2022). Again, this is vector compression; the document's token count changes only when an additional stage modifies the text or the context submitted to the generator.

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

4.2 Tokenization and hard prompt compression: direct reduction of n

Direct token reduction requires modifying T(s) or selecting a subset of the sequence. Subword methods balance vocabulary size and sequence length: larger units tend to produce shorter sequences for frequent patterns, whereas smaller units improve robustness to rare words and linguistic diversity (Sennrich, Haddow, and Birch, 2016; Kudo and Richardson, 2018). There is, however, no universally optimal tokenizer; efficiency depends on the domain, language, and text distribution.

In hard compression, a binary mask m selects prompt positions. The new length is n' = Σ mi, with n' < n. The challenge is to reduce n while preserving the information required for the task:

$$m \in \{0,1\}^n, \quad x' = \{xi \mid mi = 1\}, \quad n' = \Sigma_i\, mi.$$

LLMLingua combines budget control, iterative token-level compression, and alignment between models to produce shorter prompts. The study reports compression ratios of up to 20× with limited performance loss on specific datasets, although the results depend on the task type, compression model, and target model (Jiang et al., 2023). LongLLMLingua adapts selection to long contexts and to the position of relevant information; on NaturalQuestions, the authors report approximately fourfold token reduction and performance gains in the evaluated setting, as well as lower latency for prompts on the order of ten thousand tokens (Jiang et al., 2024).

These techniques have a direct relationship with token savings because the textual sequence actually submitted to the model is shortened. However, compression introduces its own cost, may remove necessary evidence, may alter instructions, and may behave differently with closed models. Compression ratio should therefore be assessed alongside fidelity, accuracy, end-to-end latency, and compressor cost.

4.3 Continuous prompts and gist tokens

Continuous prompts replace part of a textual instruction with learned vectors. In prefix-tuning, model parameters remain frozen while a small set of task-specific vectors

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

is optimized to function as "virtual tokens" processed by the model's layers. The method reduces the number of parameters adjusted for each task, does not guarantee a shorter factual sequence, and requires access to the model so that continuous representations can be inserted (Li and Liang, 2021).

The gist-token method trains a model to condense a prompt into a small number of reusable positions. Across the models and tasks evaluated, Mu, Li, and Goodman (2023) report compression of up to 26×, reduced FLOPs, and storage savings with limited quality loss. Unlike simple word deletion, information is represented in learned latent states. A practical limitation is that these states are generally neither interchangeable across models nor accessible through APIs that accept text only.

Continuous prompts can therefore reduce repeated discrete text and computation in controlled architectures, but "billed-token savings" depend on the deployment interface. A rigorous evaluation must report whether the vectors count as internal positions, whether they are precomputed, and whether training cost has been amortized.

4.4 Vector retrieval, RAG, and context compression

RAG combines a parametric model with non-parametric memory. The query is encoded as a vector q; each document passage has a vector zj; and retrieval selects the items with the highest similarity, for example using the inner product or cosine similarity:

$$score(q,z_j) = q^{T}z_j / (||q||\cdot||z_j||), \quad C = concat(top\text{-}k(score)).$$

In the conventional design, retrieved passages are converted into text and inserted into the generator's context (Lewis et al., 2020). Compact embeddings can therefore reduce index size and accelerate search, whereas k, passage length, and reranking determine how many tokens enter the prompt. Retrieving too many documents can increase cost and introduce distraction; retrieving too few can reduce evidence coverage.

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

RAG reduces tokens only when the pipeline controls the final context. Practical strategies include chunking that reflects document structure, two-stage retrieval, deduplication, reranking, extraction of relevant sentences, and compression of retrieved material. Embedding dimensionality and the amount of returned text should be optimized separately.

xRAG explores a modality bridge that projects a retrieved document embedding into the model's representational space, eliminating the need to provide the corresponding full text. The study reports integration of retrieved context into an extremely compact representation, lower FLOPs, and competitive performance on the evaluated tasks (Cheng et al., 2024). The method demonstrates that embeddings can replace textual tokens under specific training and architectural conditions; this conclusion cannot be generalized to conventional RAG systems that merely concatenate documents with the prompt.

4.5 KV-cache compression: lower decoding memory

During autoregressive generation, each layer retains the keys and values of previous positions to avoid recomputation. In a simplified approximation, cache memory grows with the number of layers L, the number of retained positions nKV, hidden dimension d, and the number of bytes b per element:

$$MKV \approx 2 \cdot L \cdot nKV \cdot d \cdot b.$$

H2O formulates position retention as a policy that preserves recent tokens and heavy hitters, that is, positions that make substantial contributions to attention. The method reduces cache size and improves throughput and latency in the evaluated systems (Zhang et al., 2023). SnapKV identifies attention patterns from an observation window and selects positions clustered by head; for inputs containing 16,000 tokens, the authors report faster generation and improved memory efficiency with comparable performance on their benchmarks (Li et al., 2024).

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

These approaches reduce nKV, not necessarily n. A model may receive the same prompt and a service may count the same number of tokens while retaining fewer states in GPU memory during decoding. This is therefore an inference-memory optimization, not textual compression.

5 SYNTHESIS: WHICH VARIABLE DOES EACH TECHNIQUE CHANGE?

Table 2 - Relationship among technique, mathematical variable, and operational effect

Technique class	Primary variable	Reduces discrete tokens?	Main benefit	Risk or condition
Tokenization/subword units	n and V	May reduce n	More compact sequence and lexical coverage	Trade-off among vocabulary size, language, and rare terms
Embedding-matrix factorization	Parameters $Vd \rightarrow Vr + rd$	No	Smaller model and lower cost per representation	Low-rank constraint may limit capacity
Adaptive/Matryoshka embeddings	d per item or dimensionality used	No	Smaller index and lower search and storage costs	Quality varies by task and dimensionality
Hard prompt compression	$n \rightarrow n'$	Yes	Less context processed and potentially lower cost	Semantic loss and compressor overhead
Soft prompts/gist tokens	m latent positions	Conditional	Instruction reuse and lower computation in a controlled model	Requires internal access/training; billing depends on the interface

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

Technique class	Primary variable	Reduces discrete tokens?	Main benefit	Risk or condition
RAG with top-k and reranking	k and passage tokens	Yes, if the final context becomes shorter	Less irrelevant text and greater evidence density	Retrieval error and loss of coverage
xRAG/modality bridge	Document representation	Yes, in the specific design	Replacement of retrieved text with a projected vector	Dependence on training and architecture
KV-cache compression	nKV	No	Lower memory use and faster decoding	Potential removal of important positions

6 DISCUSSION

The principal contribution of this review is to replace the generic relationship "optimizing embeddings reduces tokens" with a multivariable causal model. Embeddings belong to the continuous domain, whereas tokens belong to the discrete domain. Matrix E transforms identifiers into vectors, but the number of identifiers has already been determined by T. Only techniques that modify T, select positions, summarize context, or replace text with latent states directly alter n.

This distinction also explains why results from recommender systems or computer vision cannot be transferred automatically to LLMs. Optimizing distributed embedding tables, for example, may reduce communication across GPUs without producing any token savings. Likewise, the term embedding in watermarking and steganography refers to inserting information into a medium rather than to a semantic

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

vector representation. Reviews based solely on lexical matches in titles risk conflating distinct concepts.

For RAG applications, an efficient architecture should be designed in layers. First, documents are segmented into semantically coherent units. Second, embeddings are generated and stored with dimensionality and numerical precision appropriate to the trade-off between memory and retrieval quality. Third, broad retrieval produces candidates, followed by more selective reranking. Fourth, retrieved content is deduplicated, extracted, or compressed before reaching the generator. Finally, during decoding, cache policies can reduce memory without changing content that has already been submitted. Each layer requires its own metric.

At a minimum, an experimental evaluation should report: token counts before and after compression; embedding dimensionality and numerical format; index size; retrieval recall; response quality; latency of the compressor, retriever, and generator; FLOPs or an equivalent proxy; peak memory use; and total end-to-end cost. Reporting compression ratio alone can conceal quality degradation or shift cost to an auxiliary stage.

Language and domain must also be considered. Tokenizers trained predominantly on one distribution may segment technical terms, proper names, or Portuguese-language texts differently. Perplexity-based compressors may remove elements that appear redundant but are legally, clinically, or scientifically essential. In high-risk contexts, evidence preservation and auditability should take precedence over aggressive compression.

Future research could jointly optimize n, d, k, and nKV under explicit quality constraints. A multi-objective function could minimize total cost and performance loss rather than optimize a single quantity. Methods integrating nested representations, context selection, and modality bridges may bring retrieval and generation closer

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

together, but they require evaluation for robustness, cross-model generalization, and transparency of the retrieved evidence.

7 LIMITATIONS

This is a structured critical review, not an exhaustive systematic review or meta-analysis. The compared studies use different architectures, hardware, datasets, and metrics; compression or acceleration values should not be compared as though they had been obtained under the same protocol. Some techniques require access to internal states or additional training and may therefore be inapplicable to models accessed only through an API. Token-billing outcomes depend on provider policy and cannot be inferred solely from reductions in FLOPs or cache size. Finally, the literature is evolving rapidly, requiring periodic updates to the taxonomy and independent replication.

8 CONCLUSION

Matrix and vector methods are central to language-model efficiency, but they operate on distinct components. Embedding factorization, adaptive capacity, and nested representations reduce parameters, dimensionality, or storage. Prompt compression, token selection, and control of RAG context reduce the discrete length of the input. Gist tokens and modality bridges can replace text with latent states in architectures trained for that purpose. KV-cache compression reduces decoding memory without necessarily decreasing the number of tokens processed or billed.

The correct question, therefore, is not whether embedding optimization generally reduces tokens, but which variable each technique changes and which operational metric is intended to be minimized. Future studies should distinguish n, d, V, k, and nKV, measure end-to-end cost, and preserve quality and traceability. This conceptual discipline enables the construction of more efficient RAG systems and LLMs without

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

attributing to vector compression benefits that actually arise from sequence or memory compression.

REFERENCES

BAEVSKI, A.; AULI, M. Adaptive Input Representations for Neural Language Modeling. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS, 2019. OpenReview: ByxZX20qFQ.

CHENG, X.; WANG, X.; ZHANG, X.; et al. xRAG: Extreme Context Compression for Retrieval-Augmented Generation with One Token. Advances in Neural Information Processing Systems, vol. 37, 2024. DOI: 10.52202/079017-3476.

JIANG, H.; WU, Q.; LIN, C.-Y.; YANG, Y.; QIU, L. LLMLingua: Compressing Prompts for Accelerated Inference of Large Language Models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Singapore: Association for Computational Linguistics, 2023. pp. 13358-13376. DOI: 10.18653/v1/2023.emnlp-main.825.

JIANG, H.; WU, Q.; LUO, X.; et al. LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Bangkok: Association for Computational Linguistics, 2024. pp. 1658-1677. DOI: 10.18653/v1/2024.acl-long.91.

KUDO, T.; RICHARDSON, J. SentencePiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Brussels: Association for Computational Linguistics, 2018. pp. 66-71. DOI: 10.18653/v1/D18-2012.

KUSUPATI, A.; BHATT, G.; REGE, A.; et al. Matryoshka Representation Learning. Advances in Neural Information Processing Systems, vol. 35, 2022. DOI: 10.52202/068431-2192.

LAN, Z.; CHEN, M.; GOODMAN, S.; et al. ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS, 2020. OpenReview: H1eA7AEtvS.

LEWIS, P.; PEREZ, E.; PIKTUS, A.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020.

LI, X. L.; LIANG, P. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In: Proceedings of the 59th Annual Meeting of the Association for Computational

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/06/2026 - **Accepted:** 27/07/2026
Vol: 01.02
DOI: 10.66104/5vd2bb95
Pages: 1-15

Linguistics and the 11th International Joint Conference on Natural Language Processing. 2021. pp. 4582-4597. DOI: 10.18653/v1/2021.acl-long.353.

LI, Y.; HUANG, Y.; YANG, B.; et al. SnapKV: LLM Knows What You Are Looking for Before Generation. Advances in Neural Information Processing Systems, vol. 37, 2024. DOI: 10.52202/079017-0722.

MU, J.; LI, X.; GOODMAN, N. Learning to Compress Prompts with Gist Tokens. Advances in Neural Information Processing Systems, vol. 36, 2023. DOI: 10.52202/075280-0848.

SENNRICH, R.; HADDOW, B.; BIRCH, A. Neural Machine Translation of Rare Words with Subword Units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Berlin: Association for Computational Linguistics, 2016. pp. 1715-1725. DOI: 10.18653/v1/P16-1162.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; et al. Attention Is All You Need. Advances in Neural Information Processing Systems, vol. 30, 2017.

ZHANG, Z.; SHENG, Y.; ZHOU, T.; et al. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. Advances in Neural Information Processing Systems, vol. 36, 2023. arXiv: 2306.14048.

Salomão, P. E. A. (2026). Tokens as Computational Units in Data Science and Machine Learning: Mathematical Foundations, Transformer Architecture, Inference Economy, and Caching Systems in Foundational Models. *Journal International Review of Research Studies*, *1*(02), 1-27. https://doi.org/10.66104/kxf7hk05