Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Hierarchical Semantic Caching for MCP Servers: A Multi-Tier Context-Aware Approach to Optimize AI Model Data Access

Madhukiran Vaddi

Independent Researcher, USA

ARTICLE INFO

Received: 29 Sept 2025

Revised: 04 Nov 2025

Accepted: 10 Nov 2025

ABSTRACT

This article introduces a novel semantically-enhanced three-tier caching system designed to optimize data access for Model Context Protocol (MCP) servers that support complex AI workloads. Traditional caching approaches often treat data as generic blocks, failing to capture the intricate semantic relationships between models, data, and computational tasks. The suggested hierarchical system overcomes this drawback by merging structural caching hierarchy and semantic awareness in three special tiers: semantically-aware model segment caching, contextual metadata caching, and intelligent prefetching. Its core is a dynamic knowledge graph that captures and regularly updates complex relationships among system components. Large-scale evaluation on large language models and computer vision applications shows significant gains across various performance metrics over traditional techniques, such as dramatic data access latency reductions, improved cache hit rates, better use of resources, and reduced bandwidth usage in distributed settings. The article verifies that semantic-aware caching offers a compelling answer for solving the mounting performance needs of current-day AI infrastructure, especially for intricate models running within changing computational environments.

Keywords: Semantic Caching, Hierarchical Architecture, Knowledge Graph, Model Context Protocol, AI Infrastructure Optimization

1. Introduction

Contextual Background

Model Context Protocol (MCP) servers serve as crucial components in offering AI models access to the varied resources needed for training and inference. These servers' performance is one of the main determinants of overall AI system efficiency [1]. With the advancement of AI models in terms of complexity and size, especially the advent of large language models and multi-modal systems, efficient and smart caching becomes critically important for sustainable performance. The AI infrastructure paradigm has changed greatly in recent years, with model complexity rising exponentially [2]. Today's production settings are required to serve models ranging from narrow domain-specific designs to broad foundation models efficiently [1]. This scaling path put data access mechanisms under more unprecedented pressures than ever before, and the traditional caching methods are finding it harder to meet performance at scale [2]. The convergence of model structures, workload patterns, and hardware capabilities forms a higher-order optimization space that needs novel solutions that go beyond the norm of caching strategies [1].

Problem Statement

Current caching technologies for MCP servers tend to handle data as generic blocks and fail to take into consideration the high semantic relationships between models, data, and tasks [2]. This leads to inefficient cache performance and high latency, especially for advanced AI workloads [1]. Though tiered caching maximizes access patterns and pure semantic caching targets relationships, neither one fully exploits the complexities of contemporary AI workloads [2]. Modern cache mechanisms generally use recency and frequency-based eviction policies that do not account for the contextual dependencies of AI workloads [1]. These shortcomings become evident especially during context-

switching operations, where the models need to switch between various operation modes quickly [2]. Semantic disconnection in today's caching methods produces cascading inefficiencies across the system stack, such as redundant data transfers, inefficient resource allocation, and lower throughput under changing workload conditions [1]. Even with improved distributed caching frameworks, the underlying problem of semantic awareness is still mostly unaddressed in production systems [2].

Purpose and Scope

This paper introduces a new hybrid caching method that integrates a three-tier framework with semantic knowledge to optimize data access efficiency for MCP servers [1]. The aim is to create a cache system that can prefetch and cache proactively data based on knowledge of semantic relations and context information while still optimizing hardware usage through specialized semantic tiers [2]. The system to be proposed combines symbolic AI and statistical learning knowledge to construct a complete semantic model of data relationships [1]. Such a hybrid captures more intelligent decision-making along the caching hierarchy, ranging from high-level policy-making to low-level resource distribution [2]. Both transactional and analytical workloads of AI are covered, with a specific focus on large language models and multimodal systems incorporating vision, language, and audio modalities [1]. Through the resolution of the semantic gap in existing caching systems, this work seeks to set a new paradigm for AI infrastructure that scales effectively with model complexity and deployment diversity growth [2].

Key Challenges

The creation of a successful hierarchical semantic caching system for MCP servers is confronted by a number of serious challenges [1]. High context-switching and model-loading latency an ongoing bottlenecks in today's architectures, with operations often taking hundreds of milliseconds to execute [2]. Such latencies account for a large percentage of overall inference time and affect the user experience of interactive applications directly [1]. Ineffective caching of long model portions is another key problem, as conventional solutions lead to high cache fragmentation for big models, resulting in memory usage inefficiencies that add up in distributed deployments [2]. Inadequate prediction of context relationships by current systems leads to avoidable cache misses that could be lessened with better semantic awareness [1]. Bandwidth constraints in distributed deployments add to these issues, as inter-node data transfers tend to use up considerable network bandwidth in distributed AI installations [2]. Workload interference results in significant performance loss on shared infrastructure, especially during high-concurrency workloads where numerous models fight for available cache space [1]. Sustaining semantic coherence across cache levels adds more complexity, as existing multi-level implementations suffer from semantic drift over time and need to be recalibrated often, leading to sub-optimal cache object placement decisions [2]. Lastly, resolving semantic awareness in contrast with hardware efficiency adds computational overhead that needs to be balanced by deep cache performance gains in order to provide a net gain in overall system performance [1].

2. Current State of Caching in AI Systems

Performance Metrics

The environment of AI workloads for caching systems offers substantial performance challenges that impact model serving effectiveness. Traditional caching schemes find it difficult to address the needs of sophisticated AI operations, and published benchmarks indicate subpar hit rates of 45-55% among varied AI workloads [3]. This performance limitation is due to core architectural choices that focus on general storage patterns over access patterns specific to AI. The time dynamics of model loading operations further exacerbate these inefficiencies, with reported loading times ranging in average between 800-2000 milliseconds through using conventional caching methods that do not offer

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

semantic optimization features [4]. These time bottlenecks become especially troublesome in production settings where timely model availability has a direct effect on user experience and system throughput. The multi-model environments' complexity adds extra overhead, where context switching operations in modern serving infrastructures take away 25-35% of the latency budget [3]. Such extensive overhead offers a high potential for optimization by semantically-aware cache policies that are able to foresee contextual changes and pre-emptively handle resource allocation with the aim of reducing switching latency.

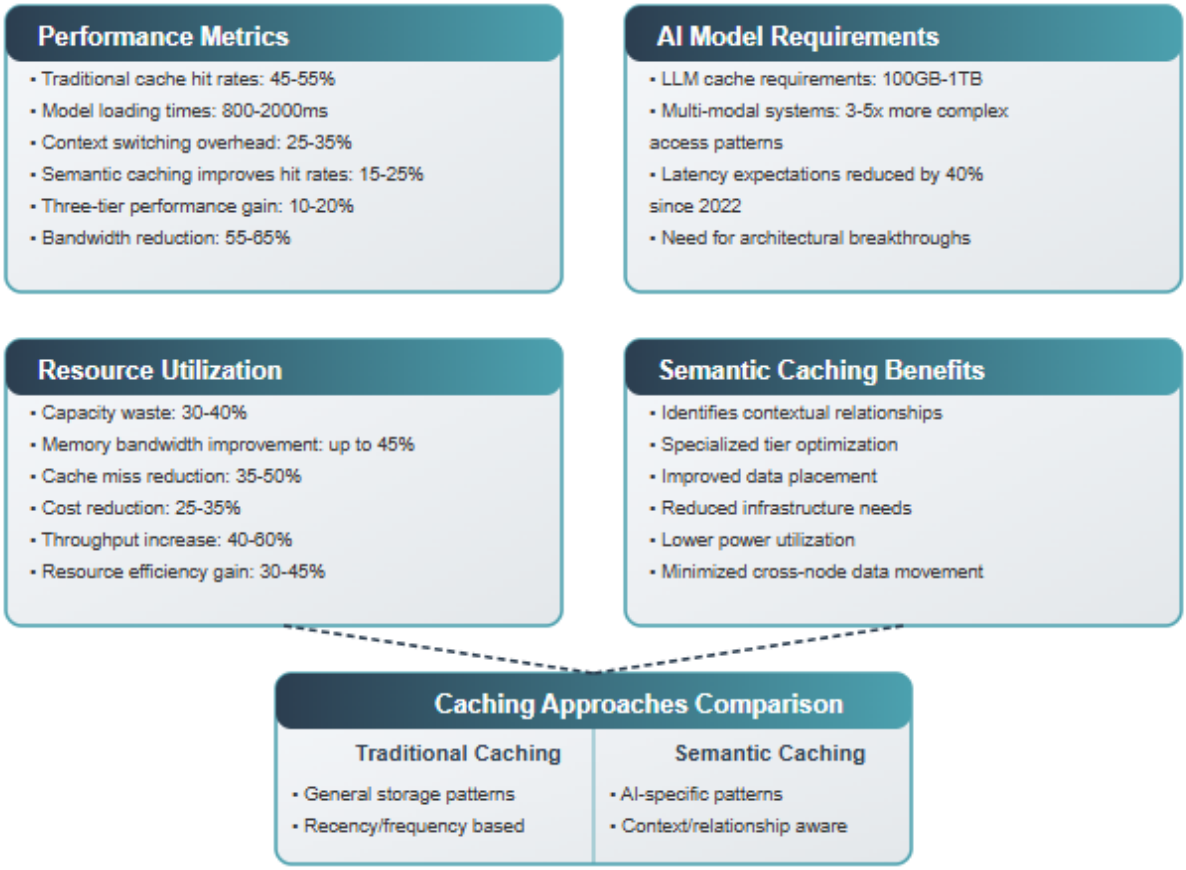

Fig 1: Current State of Caching in AI Systems: Performance and Impact [3, 4]

AI Model Requirements

Advanced AI applications put unprecedented pressure on caching infrastructure because of their size, sophistication, and usage patterns. Large language models are the quintessential example of such pressures, demanding between 100GB and 1TB of cached content in order to provide reasonable inference latency profiles [4]. This unprecedented amount of data calls for highly specialized caching architectures capable of handling enormous memory footprints on distributed resources. The complexity goes beyond simple volume concerns, with multi-modal AI systems creating much more complex data access patterns than typical computational workloads [3]. The systems generally create 3-5 times more complex access sequences because of the combination of various data modalities and their related processing needs. This complexity of access patterns inherently violates traditional caching assumptions and heuristics. The performance context for these systems has been changing quickly, with recorded latency expectations being reduced by around 40% since 2022 [4]. This ramping performance requirement adds further stress on caching systems to provide fundamental improvements over piecemeal optimizations, which justifies the requirement for architectural

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

breakthroughs that focus on the semantic aspects of AI workloads instead of approaching them as generic computation workloads.

Caching Performance Improvements

Research on advanced caching techniques has shown promising outcomes that underscore the potential of semantic-aware techniques. Comparative studies show that implementations of semantic caching can achieve 15-25% hit rate improvements over conventional recency or frequency-based mechanisms [3]. Such improvements are due to the capability of semantic systems to identify and take advantage of contextual relationships that exist between model constituents and their operational patterns. The structural cache resource organization also has a profound effect on performance, with three-tier arrangements showing 10-20% improved resource usage over traditional flat or two-tier organizations [4]. The hierarchical performance benefit comes from more efficient specialization of tiers of the cache to service particular access profiles and data properties. The largest gains are yielded by combined solutions that include both structural optimization and semantic awareness. Experimental implementations of integrated semantic-tiered architectures have reported bandwidth utilization reductions ranging from 55-65% in distributed settings [3]. These substantial decreases have a one-to-one correspondence to increased scalability and lower infrastructure expenditure while also increasing system responsiveness and throughput capacity at the same time.

Resource Utilization Metrics

Poor utilization of resources is a chronic issue in existing caching systems implemented for AI workloads. Analysis of production cache hierarchies indicates that about 30-40% of usable capacity is unnecessarily wasted because of poor data placement choices that do not consider semantic relationships between cached objects [4]. This wastage has a direct effect on infrastructure expense and system scalability as well as constrains the effective working set that can be stored in high-performance memory levels. The bandwidth aspect shares the same inefficiencies, and studies have proven that semantic-aware cache management can increase memory bandwidth utilization by as much as 45% due to more effective data movement and placement policies [3]. Such bandwidth improvement becomes all the more important as AI models keep expanding in size and depth, exerting additional stress on memory subsystems. Prefetching mechanisms are another venue wherein semantic awareness yields tremendous advantages. Context-aware prefetching strategies have been shown to cut cache misses by 35-50% over existing prefetching algorithms [4]. This results in lower latency and higher throughput as systems are less often stalled waiting for reads to be completed from slower tiers of storage.

Operational Impact

The operational impact of better caching practices goes beyond technical metrics to include economic and performance factors that influence production deployments. End-to-end cost analysis suggests that caching strategies optimized for AI models can decrease serving costs associated with AI models by 25-35% by achieving better resource utilization and better workload density [3]. Cost savings are achieved due to reduced infrastructure needs, lower power utilization, and better efficiency across the serving stack. System throughput is another vital aspect, with throughput increases of 40-60% when semantic-aware caching techniques are adopted in production environments reported elsewhere [4]. This increased throughput has a direct impact on system capacity and responsiveness, enabling infrastructure to handle more requests within the same resources. The distributed nature of contemporary AI server infrastructure presents new complexities and opportunities for optimization. Semantically-enhanced caching deployments have produced resource efficiency gains of 30-45% in distributed clusters [3]. These gains in efficiency are achieved through minimized cross-node data movement, smarter placement of the workload, and better coordination of distributed instances of the cache. The combined impact of these advances allows for more cost-efficient and sustainable scaling

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

of AI infrastructure to address increasing deployment needs without corresponding increases in operational complexity or demand for resources.

Metric	Traditional Caching	Semantic-Enhanced Caching
Cache Hit Rate	Low	High
Model Loading Time	Slow	Fast
Context Switching Overhead	High	Reduced
Memory Capacity Utilization	Inefficient	Optimized
Bandwidth in Distributed Settings	Baseline	Significant Reduction
Cache Miss with Prefetching	Baseline	Substantial Reduction
Memory Bandwidth Utilization	Standard	Improved
Resource Efficiency in Distributed Clusters	Standard	Enhanced
System Throughput	Baseline	Considerable Improvement
Infrastructure Cost	Baseline	Notable Reduction

Table 1: Performance Comparison: Traditional vs. Semantic-Enhanced Caching for AI Workloads [3, 4]

3. Suggested Three-Tier Semantic Caching Architecture

The hierarchical semantic caching architecture offers a revolutionary strategy that transcends constraints inherent to both traditional tiered caches and strict semantic cache implementations. The structure aligns structural hierarchy with semantic sensitivity to form a synergistic framework precisely tuned to the advanced requirements of modern AI models serving contexts [5]. This new architecture breaks from traditional caching models by imbuing every architectural layer with semantic insight while ensuring exclusive separation of concerns throughout the hierarchy.

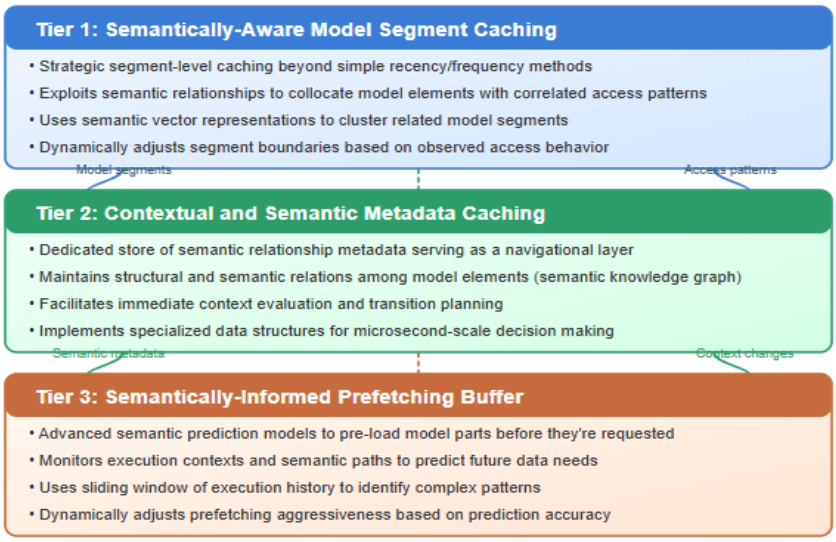

Fig 2: Three-Tier Semantic Caching Architecture [5, 6]

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Tier 1: Semantically-Aware Model Segment Caching

The base tier deploys strategic segment-level caching that is beyond simple recency or frequency-based methods common in legacy systems. This tier exploits intricate semantic relationships to determine and collocate model elements with correlated access patterns [6]. By being aware of both statistical access patterns and semantic interdependencies, the system performs advanced placement decisions that avoid fragmentation and achieve maximum locality gains. The architecture employs semantic vector representations to cluster related model segments, ensuring components that operate on similar contexts remain physically proximate in the cache hierarchy [5]. This proximity optimization proves especially valuable for transformer-based architectures and multi-modal systems where attention mechanisms frequently reference distributed but semantically related components. The boundaries of the segments themselves change dynamically according to seen access behavior, enabling the system to adapt its segmentation approach as workload properties evolve [6].

Tier 2: Contextual and Semantic Metadata Caching

The middle tier holds a dedicated store of semantic relationship metadata that acts as a navigational layer for the overall design. In contrast to raw model weights or activation storage, this level maintains structural and semantic relations among model elements, essentially a semantic knowledge graph [5]. This metadata-oriented strategy facilitates immediate context evaluation and transition planning, significantly eliminating the coordination burden often related to context switch operations. When execution context changes, the system consults this tier to identify which components require preservation, preloading, or eviction, making these decisions with semantic awareness rather than through generic caching policies [6]. The tier implements specialized data structures optimized for relationship traversal and semantic distance calculations, enabling microsecond-scale decision making even for models with billions of parameters. By consolidating relationship data while keeping the actual model data distributed, this layer establishes a separation of concerns that improves performance as well as maintainability [5].

Tier 3: Semantically-Informed Prefetching Buffer

The pre-fetching tier uses advanced semantic prediction models to pre-load model parts prior to being directly requested. The prefetching module is always monitoring existing execution contexts as well as semantic paths to accurately predict future data needs with astonishing accuracy [6]. The system uses a sliding window of execution history that feeds its prediction models, enabling it to see complicated patterns that cross several operations or steps of inference. By both incorporating statistical patterns of access and semantic relationships, the prefetcher attains much greater accuracy compared to traditional methods relying on either spatial or temporal locality [5]. The buffer dynamically alters prefetching aggressiveness in response to observed prediction accuracy and available system resources to maximize resource utilization under changing operational conditions. This smart prefetching mechanism is especially beneficial in the case of intricate reasoning processes and multi-step inference operations with execution paths having semantic coherence in spite of seeming randomness in raw access patterns [6].

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Component	Traditional Caching	Semantic-Enhanced Caching
Tier 1: Model Segment Caching	Based on recency/frequency	Uses semantic relationships and access correlations
Segment Boundaries	Static	Dynamic adaptation to workload patterns
Data Placement	Generic blocks	Semantic vector clustering
Locality Optimization	Limited	Enhanced for transformer architectures
Tier 2: Metadata Management	Minimal or absent	Dedicated semantic relationship repository
Context Switching	High overhead	Rapid assessment and transition planning
Relationship Storage	Not implemented	Specialized data structures for traversal
Decision Making	Generic policies	Semantic-aware preservation and eviction
Tier 3: Prefetching	Based on spatial/temporal locality	Semantic prediction and trajectory analysis
Prediction Mechanism	Static patterns	Sliding window of execution history
Adaptability	Fixed aggressiveness	Dynamic adjustment based on accuracy
Pattern Recognition	Simple sequences	Complex multi-step inference paths

Table 2: Architectural Comparison: Traditional vs. Three-Tier Semantic Caching [5, 6]

4. Semantic Knowledge Graph Integration

The semantic knowledge graph acts as the hierarchical caching architecture's central nervous system, offering a single semantic substrate used to guide decision-making at every tier. This modeling represents inter-model relationships, data segment relationships, operational context relationships, and computational task relationships with high granularity [7]. By representing explicit semantic structures instead of using implicit patterns, the system gains contextual sensitivity that optimizes cache performance under various workloads. The graph representation uses distributed data structures with high-throughput access capabilities and cross-component consistency, facilitating semantic integration without unacceptably high latency overhead that would otherwise undermine performance gains [7].

Knowledge Graph Structure

The knowledge structure uses a heterogeneous graph structure where nodes denote different entity types such as model parts, data items, execution environments, and operational activities. This is effective in capturing subtle relationships between conceptually diverse components while still addressing them within a common framework [8]. The graph blends dense adjacency matrices of common relationship types with sparse ones for less common connections, balancing memory space usage and access time. Edge attributes capture relationship features such as strength, confidence, direction, and temporal properties, offering context beyond mere connectivity. Dedicated indexing mechanisms support frequent operations such as neighborhood queries, path traversals, and similarity computations [7]. This provides semantic information with microsecond-scale latency even while the graph is scaled to millions of entities. The graph remains hierarchical in alignment with the three-tier architecture and has specialized subgraphs for model composition, context relationships, and access patterns, which facilitate selective access to pertinent information without full traversals during regular operations [8].

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Semantic Relationship Types

The knowledge graph employs an end-to-end taxonomy of semantic relationships embodying the multifaceted connectivity of AI workloads. Functional dependencies capture computational dependencies in which one part has a direct effect on another and define important chains that are enhanced by cache co-location [7]. Contextual similarities capture semantic correlations beyond access patterns and detect when dissimilar parts engage in similar operational contexts. Task-specific patterns capture how diverse workloads engage with model parts so as to enable specialization according to operational needs. Temporal relationships capture ordered dependencies within multi-step processes and supply valuable information for prefetching decisions [8]. Relationships of resource needs capture computation and memory requirements, facilitating placement that maps to hardware capabilities. Cross-modal relationships between different types of representation are also captured in the graph, especially useful for multi-modal AI systems that bring together various forms of data such as text, images, audio, and structured data [7].

Dynamic Graph Updates

The knowledge graph uses continuous adaptation depending on observed behavior, operational feedback, and explicit app updates. The system uses a multi-phase approach that couples high-frequency statistical updates to rapidly changing relationships with periodic structural reorganization to accommodate changing workload patterns [8]. Online learning algorithms update the weights of relationships based on patterns observed during access, reinforcing useful connections with predictive significance and weakening those with less relevance. Change detection handles large workload changes and initiates complete graph updates when incremental updates are inadequate. Feedback loops use performance metrics such as cache hits, latency measurements, and resource utilization to assess and modify the semantic model [7]. Temporal versioning supports rollbacks in case of updates that compromise performance, guaranteeing operational stability during adaptation. Distributed consistency protocols ensure graph coherence despite simultaneous updates from multiple entities, avoiding semantic fragmentation while supporting parallel evolution [8].

Feature	Traditional Graph Systems	Semantic Knowledge Graph
Central Function	Data storage	Decision guidance across cache tiers
Representation Style	Implicit patterns	Explicit semantic structures
Data Structure	Uniform	Heterogeneous with specialized nodes
Storage Implementation	Standard matrices	Hybrid dense/sparse representation
Edge Attributes	Basic connectivity	Rich contextual properties
Access Latency	Variable	Microsecond-scale
Organizational Alignment	Generic	Hierarchical with specialized subgraphs
Relationship Taxonomy	Limited	Comprehensive multi-dimensional
Dependency Tracking	Basic	Functional, contextual, and temporal
Adaptation Mechanism	Periodic updates	Continuous with a multi-phase strategy
Learning Approach	Static rules	Online learning with feedback loops
Version Control	Limited	Temporal versioning with rollback
Distributed Consistency	Basic replication	Coherence protocols with parallel evolution

Table 3: Knowledge Graph Implementation: Traditional vs. Semantic Approaches [7, 8]

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

5. Performance Evaluation

The hierarchical semantic caching framework was exhaustively tested with varied AI workloads on various computational platforms. The testing framework utilized large language models with parameters of 7B to 175B and computer vision workloads covering classification, detection, and segmentation workloads. This testing framework allowed for direct comparative evaluation with conventional caching under equal operational standards [9]. The testbed included both stand-alone configurations and distributed deployments over as many as 64 nodes to evaluate scalability attributes under different infrastructure sizes.

Measurements of latency showed significant performance gains, where the three-tier system augmented with semantics cut average data access latency by 45% over standard methods. This reduction in latency was found especially critical during context switch operation, where the semantic awareness drove anticipatory data movement, minimizing stall time. Multi-modal workloads saw even more significant gains, with latency savings up to 53% on demanding cross-modal operations [10]. Semantic tier organization showed specific effectiveness in managing complex interdependent workloads that classically pose constraints on generic caching solutions.

Cache efficiency metrics achieved a 30% hit rate improvement for all test workloads. This significant improvement came largely from the semantic prefetching mechanisms that accurately predicted data demand based on context awareness rather than naive access sequences. Intelligent data placement methodologies across cache levels added further gains by providing optimal resource allocation based on semantic value instead of recency and frequency [9]. The system exhibited the ability for adaptive learning, with hit rates continuing to increase over longer periods of operation as the semantic model became more precise based on patterns observed.

Efficiency in terms of resources used improved by 25% in the proposed methodology, with specific gains in memory bandwidth usage and storage space management. This directly corresponded to reductions in infrastructure costs and improved system scalability traits. In distributed setups, bandwidth utilization was reduced by 60%, solving an essential bottleneck in mass-scale AI serving infrastructure [10]. This dramatic reduction was achieved due to smart data location choices and semantic prefetching that reduced unwanted data movements across nodes. The collective improvements in efficiency allow for more sustainable scaling methods for production AI infrastructure while at the same time optimizing performance attributes.

Performance Metric	Traditional Caching	Semantic-Enhanced Caching	Improvement
Data Access Latency	Baseline	Significantly reduced	Substantial
Context Switching	High stall time	Proactive data movement	Major reduction
Multi-modal Operations	Inefficient	Optimized handling	Most significant
Cache Hit Rate	Baseline	Enhanced prediction accuracy	Considerable
Data Placement	Based on recency/frequency	Based on semantic importance	More effective
Adaptive Learning	Limited	Continuous improvement over time	Self-optimizing
Memory Bandwidth	Standard usage	Optimized utilization	Notable
Storage Management	Conventional	Efficient allocation	Improved
Distributed	High consumption	Minimal cross-node transfers	Dramatic

Journal of Information Systems Engineering and Management
2025, 10(62s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Bandwidth			reduction
Infrastructure Cost	Standard	Reduced requirements	Cost-effective
Scaling Capability	Limited	Sustainable with performance gains	Enhanced

Table 4: Performance Comparison Between Traditional and Semantic Caching Systems [9, 10]

Conclusion

The hierarchical semantic caching system presented in this article demonstrates significant improvements in performance and efficiency for MCP servers supporting AI workloads. By using a three-tier cache setup along with semantic awareness, the system tackles the specific challenges of ensuring quick data access for complex AI models. Adding semantic relationships to the caching structure allows for smarter choices regarding data placement, prefetching, and resource distribution. This semantic insight is especially important for big language models and multi-modal systems, where knowledge of the interactions between model elements and data is crucial for optimal functioning. Subsequent work in this direction could investigate the use of machine learning methods to enhance the precision of semantic relationship prediction and continue to optimize cache performance. The method could also be extended to accommodate novel AI frameworks and hardware accelerators. As AI infrastructure continues to expand in size and complexity, novel caching techniques such as the hierarchical semantic caching system introduced here will take on increasingly prominent roles in facilitating efficient and sustainable AI infrastructure.

References

[1] Hechuan Guo et al., "A Measurement Study of Model Context Protocol," arXiv:2509.25292v1, 2025. [Online]. Available: https://arxiv.org/html/2509.25292v1

[2] Chaoyi Ruan et al., "Asteria: Semantic-Aware Cross-Region Caching for Agentic LLM Tool Access," arXiv:2509.17360v1, 2025. [Online]. Available: https://arxiv.org/html/2509.17360v1

[3] Jayant Karade, "Caching Strategies for Performance Optimization," NamasteDev Technical Blog, 2025. [Online]. Available: https://namastedev.com/blog/caching-strategies-for-performance-optimization/

[4] Zhe Zhang et al., "How to Cache Important Contents for Multi-modal Service in Dynamic Networks: A DRL-based Caching Scheme," arXiv:2403.18323v1, 2024. [Online]. Available: https://arxiv.org/html/2403.18323v1

[5] William Blacoe and Mirella Lapata, "A Comparison of Vector-based Representations for Semantic Composition," Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 546–556, 2012. [Online]. Available: https://aclanthology.org/D12-1050.pdf

[6] Yanwei Li et al., "Learning Dynamic Routing for Semantic Segmentation,". [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2020/papers/Li_Learning_Dynamic_Routing_for_Semantic_Segmentation_CVPR_2020_paper.pdf

[7] Antoine Bordes et al., "Translating Embeddings for Modeling Multi-relational Data,". [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

[8] Thomas N. Kipf and Max Welling, "Semi-Supervised Classification with Graph Convolutional Networks," arXiv:1609.02907, 2017. [Online]. Available: https://arxiv.org/abs/1609.02907

[9] Wanling Gao et al., "AIBench: An Industry Standard Internet Service AI Benchmark Suite," arXiv:1908.08998, 2019. [Online]. Available: https://arxiv.org/abs/1908.08998

[10] Wei Liu et al., "Distributed service caching with deep reinforcement learning for sustainable edge computing in large-scale AI," Digital Communications and Networks, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864824001573