

Improving IoT Content Distribution with In-network Redundancy-Aware Semantic Caching

Marica Amadeo*, Giuseppe Ruggeri[†], Antonella Molinaro[†], Salvatore Serrano*

* Department of Engineering, University of Messina, Italy

[†] DIIES Department, University Mediterranea of Reggio Calabria, Italy

Emails: {marica.amadeo, salvatore.serrano}@unime.it, {giuseppe.ruggeri, antonella.molinaro}@unirc.it

Abstract—The widespread adoption of Internet of Things (IoT) applications calls for efficient mechanisms for dynamic content discovery and provisioning. Named Data Networking (NDN) is a natural fit for this setting, thanks to its name-based access to information and native in-network caching. However, NDN's reliance on exact name matching limits the effectiveness of caching when semantically equivalent content is published by heterogeneous IoT producers under different naming conventions. Extending cache lookup with semantic similarity addresses this limitation, but introduces a previously unexplored inefficiency: semantically equivalent contents named differently may occupy multiple cache entries, wasting capacity that a single representative could serve. This paper proposes a lightweight redundancy-aware semantic caching policy that admits new content only when it increases the semantic diversity of the cache. Simulation results, compared against legacy exact-match caching and a semantic baseline that disregards redundancy, show that the proposed policy consistently increases the cache hit ratio and reduces data retrieval latency.

Index Terms—Semantic Caching, Internet of Things, Named Data Networking, Semantic Forwarding

I. INTRODUCTION

The widespread adoption of Internet of Things (IoT) applications across smart environments is generating an ever-growing volume of dynamic contents: sensor readings, environmental measurements, and status updates that are continuously retrieved by a variety of consumer services [1]. IoT applications are inherently information-centric: they are interested in the content itself, e.g., the current air quality in a given area, rather than in the specific device or host that produced it. This makes Information Centric Networking (ICN) a natural architectural fit, with Named Data Networking (NDN) in particular widely explored as an enabling communication paradigm for IoT scenarios [2].

NDN replaces host-based communication with name-based content retrieval: hierarchical content names identify data directly at the network layer, decoupling consumer requests from the specific network address of a producer [3]. Consumers issue content request packets called Interests that are forwarded toward a producer, according to a longest-prefix match over the entries of the Forwarding Information Base (FIB). Intermediate nodes may satisfy a request directly from their Content Store (CS) if a matching Data packet was previously cached, while forwarded Interests are tracked in the Pending Interest Table (PIT), which aggregates duplicate requests for

the same content. This forwarding process, however, relies entirely on exact name matching.

In heterogeneous IoT deployments, semantically equivalent content can be published by different producers under different naming conventions, causing both forwarding failures, when a request cannot reach an alternative producer of the same information, and missed caching opportunities, when equivalent content already cached under a different name goes unrecognized [4]. To address this limitation, recent works have introduced semantic-aware packet processing into different components of the NDN forwarding plane [5] [6] [7].

In particular, introducing semantic similarity into cache lookup allows content published under different names, but conveying the same information, to be recognized and served interchangeably. This capability, however, exposes a previously unexplored inefficiency: multiple semantically equivalent contents may end up occupying distinct cache entries, even though a single representative would suffice to satisfy all related requests, wasting cache capacity that could otherwise serve additional, distinct content.

This paper addresses this open issue by proposing a lightweight semantic caching policy for NDN-based IoT scenarios, which detects semantic redundancy among cached contents. After characterizing semantic redundancy through the similarity between content names, we introduce a selective caching decision that admits incoming content only when it extends the semantic diversity of the CS. We evaluate the proposed policy against legacy NDN exact-match caching and a semantic baseline that disregards redundancy. Results consistently show that explicitly managing semantic redundancy in the CS increases the cache hit ratio and reduces data retrieval latency with respect to both benchmarks, confirming the benefit of coupling semantic caching with redundancy awareness.

The remainder of the paper is organized as follows. Section II reviews related work on semantic forwarding and caching for NDN. Section III details the proposed caching policy. Section IV presents the performance evaluation, while Section V concludes the paper.

II. BACKGROUND

Recent efforts to introduce semantic awareness into NDN can be broadly classified into *semantic forwarding* and *semantic caching* solutions. While the term *semantic caching* has

recently gained significant attention in Large Language Model (LLM) based systems, where semantically similar prompts reuse previously generated responses through embedding similarity (e.g., GPTCache [8]), its implications for *in-network caching* in NDN remain largely unexplored. In this paper, we use the term semantic caching with focus on the latter domain.

Semantic forwarding. A few works exploit Natural Language Processing (NLP) techniques to overcome the limitations of exact name matching during NDN Interest forwarding. The pioneering work in [4] introduces fuzzy name matching using Word2Vec embeddings. In [9], Raza et al. enrich content names with semantic tags and use dedicated resource-rich *supernodes* to support semantic forwarding. Our previous work [5] leverages Sentence Transformer embeddings [10] to identify semantically related FIB entries whenever exact longest-prefix matching fails. An *Embeddings Store* caches name embeddings together with the associated forwarding decisions to reuse discovered semantic matching and keep processing overhead low.

Semantic caching. More recently, semantic matching has been extended to cache operations. The work in [7] focuses on service discovery among Digital Twins hosted at the network edge. If the forwarding fabric based on exact name matching fails, it introduces a semantic lookup procedure in the CS to identify cached service results able to satisfy semantically equivalent requests under different names. However, content placement follows the standard Cache Everything policy, without considering the semantic redundancy that may emerge among cached entries. The work in [6] couples Cuckoo hashing with a reinforcement learning controller driven by a content-*importance* score derived from criticality, source trust and freshness; there, “semantic” denotes content priority rather than name meaning, and cache lookups remain exact, so requests carrying heterogeneous names are not resolved from the cache. Compared to existing works, this paper introduces a caching policy that identifies incoming semantically redundant contents and prevents them from occupying cache space, maximizing the semantic diversity of the cached items.

III. THE PROPOSAL

We consider an edge NDN domain where IoT devices act as producers of transient contents and consumer applications request them by name. Each IoT data is identified by a hierarchical name, e.g., /smartCity/districtA/airQuality. Edge nodes are equipped with a semantic CS, enabling the reuse of cached contents across heterogeneous naming conventions. Fig. 1 represents the scenario in a smart city context.

A. Workflow

Following our previous work [7], a sentence embedding model is used to map the name s of cached data into a dense vector representation $e(s)$ stored alongside the corresponding cache entry. The semantic similarity between two names s_1 and s_2 is measured by their cosine similarity:

$$\sigma(s_1, s_2) = \frac{e(s_1) \cdot e(s_2)}{\|e(s_1)\| \|e(s_2)\|} \in [0, 1], \quad (1)$$

where $\sigma = 1$ denotes identical semantic content and $\sigma = 0$ unrelated meanings. Two names are deemed a match when $\sigma \geq Th$, where $Th \in (0, 1)$ is a predefined similarity threshold.

Upon receiving an Interest carrying name q , the node first performs the legacy exact-match lookup in the CS. On failure, it performs a *semantic lookup*, selecting the cached entry n^* with the highest similarity. If $\sigma(q, n^*) \geq Th$, a cache hit is considered and the content is sent back to the consumer; otherwise, the Interest proceeds through the standard PIT and FIB lookup stages. If the FIB exact match also fails, the node falls back to semantic forwarding toward an alternative producer. Finally, if semantic forwarding fails as well, the Interest is either discarded or forwarded to a centralized resolver, e.g., a cloud data center maintaining an up-to-date replica of producer-generated content.

B. Semantic Redundancy

The introduction of semantic lookup changes the effective utility of cached contents. In a traditional CS, each cached object contributes independently to future cache hits. In contrast, under semantic retrieval, two distinct cache entries may provide overlapping ability to satisfy future requests, even if they are published under different names. Therefore, multiple providers exposing equivalent information under different naming conventions no longer need to be represented individually. To formalize this concept, we introduce the notion of *semantic class*, which denotes a group of names regarded as semantically equivalent according to the similarity threshold Th . For example, /Messina/districtA/airQualityIndex, /airMonitoring/Messina/zoneA/quality, and /Messina/zoneA/airMonitoring/qualityIndex belong to the same air-quality class and reflect the same measure in a certain geographical area, whereas names including components like parking availability, noise monitoring, and traffic monitoring define different classes.

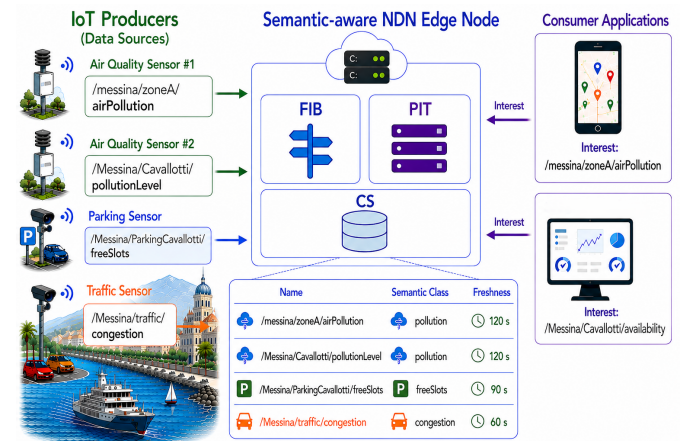


Fig. 1: Reference scenario: IoT devices produce named contents for consumer applications in a smart city.

In semantic caching, a request with name q generates a *semantic hit* whenever the CS contains at least one entry n_i such that $\sigma(q, n_i) \geq Th$. Therefore, the set of requests that can be satisfied locally depends on the semantic classes represented in the cache, rather than on the specific names stored in it. This observation reveals a previously overlooked form of *semantic redundancy*. Two cache entries n_i and n_j are considered semantically redundant whenever: $\sigma(n_i, n_j) \geq Th$. In this case, both entries belong to the same semantic class and provide overlapping coverage, since either one can satisfy the same neighborhood of semantically related requests. As a consequence, the effectiveness of a semantic CS is determined by its *semantic diversity*, i.e., the number of distinct semantic classes represented in the cache. The caching algorithm introduced next is designed to preserve this diversity as new content arrives, deciding whether it enriches the set of represented classes or merely reproduces one already present.

C. The Caching Algorithm

The proposed algorithm includes a *selective caching decision*, applied to every incoming Data packet. If the new packet is admitted and the CS is full, a Least Recently Used (LRU) replacement of an existing entry is applied. LRU is adopted here as a widely used replacement policy in NDN, combining proven effectiveness with low implementation complexity [2]. Moreover, by leveraging the `FreshnessPeriod` field natively supported by Data packets, the CS is also freshness-aware, allowing stale entries to be identified and evicted.

Let the CS hold N cached contents $\mathcal{C} = \{n_1, n_2, \dots, n_N\}$, each represented by its hierarchical name and pre-computed embedding. When an incoming Data packet c with name n_c arrives at the edge node, it is evaluated against the entries currently in $\mathcal{C}$. Therefore, the node first identifies the semantically closest cached entry,

$$n_r = \arg \max_{n_i \in \mathcal{C}} \sigma(n_c, n_i), \quad (2)$$

and applies the following criterion.

- If $\sigma(n_c, n_r) < Th$, no cached entry currently represents the semantic class of c ; the content is *admitted* for caching, as it increases the semantic diversity of the CS.
- If $\sigma(n_c, n_r) \geq Th$, c belongs to the semantic class already represented by n_r and therefore does not increase the semantic diversity of the CS. In this case, c replaces n_r only if it is more recent, keeping the class representative up to date; otherwise, c is discarded, as it carries no additional information.

As a result, the caching policy keeps the CS free of semantic redundancy, with each represented class always mapped to its most recent content.

In terms of computational cost, the proposed algorithm imposes a modest and well-bounded overhead, governed by the number of entries N currently held in the CS. Evaluating an incoming Data packet against the set of cached contents $\mathcal{C}$ to locate its closest entry n_r costs $O(N)$, and this cost is incurred once per packet, regardless of whether the outcome is admission, refresh, or discard. Storage overhead follows a

similar logic: each cached entry carries a name embedding of fixed dimensionality d (e.g., $d = 384$ for Sentence Transformers embeddings obtained with the all-MiniLM-L6-v2 model [10]), so the CS as a whole requires $O(N \cdot d)$ memory. Since edge nodes typically operate with small, bounded values of N , both the computational and memory footprint of the proposed policy remain well within practical limits.

IV. PERFORMANCE EVALUATION

Scenario. We evaluate the proposal through a custom Python-based simulator that reproduces an edge NDN domain where consumer applications request IoT contents generated by heterogeneous producers. The naming catalog reflects distinct semantic classes, each representing a distinct smart-city IoT content (e.g., air quality in a specific geographical area). For every class, multiple producers publish equivalent information under heterogeneous naming conventions and the corresponding names are registered in the FIB of edge nodes. Consumers can use exact names or alternative, paraphrase-like names to request the same information. In the latter case, queries can be resolved through semantic matching, either at the CS or, upon a CS miss, at the FIB. The request pattern follows a Zipf popularity distribution, with skewness parameter set to 0.8 [11]. Popularity is assigned per name rather than per semantic class: this lets multiple producer names belonging to the same class be requested independently, thus reproducing a concurrent, multi-provider naming scenario. Content names are mapped to embeddings using Sentence Transformers [10]. Among the available pre-trained models, we adopt all-MiniLM-L6-v2, following our previous work [5], where it was shown to achieve accuracy comparable to larger models at a substantially lower computational cost.

Benchmarks. We compare our solution against two benchmark schemes. *Legacy NDN* leverages the traditional exact-match pipeline, with every retrieved Data packet unconditionally cached and LRU replacement, if the CS is full [3]. This solution provides the reference lower bound, quantifying the benefit of introducing any form of semantic awareness. The *semantic baseline* reproduces the semantic-aware CS introduced in our prior work [7], where cache admission still follows a Cache Everything policy irrespective of semantic redundancy, with LRU replacement. The Interest processing pipeline grants absolute priority to exact matching, both at the CS and at the FIB, before falling back to semantic matching. This baseline isolates the contribution of semantic retrieval alone, without any redundancy-aware admission control, and represents the closest existing solution to the proposal. After a tuning campaign, not reported for brevity, the similarity threshold Th for both semantic approaches is set to 0.75, a value that identifies the operating region with the best trade-off between precision and recall.

Metrics. We compute the *cache hit ratio*, capturing the fraction of Interests satisfied directly by the CS, either through an exact-name match or through a semantic match. To directly capture the semantic redundancy introduced by each cache admission scheme, we compute the *wasted cache capacity*,

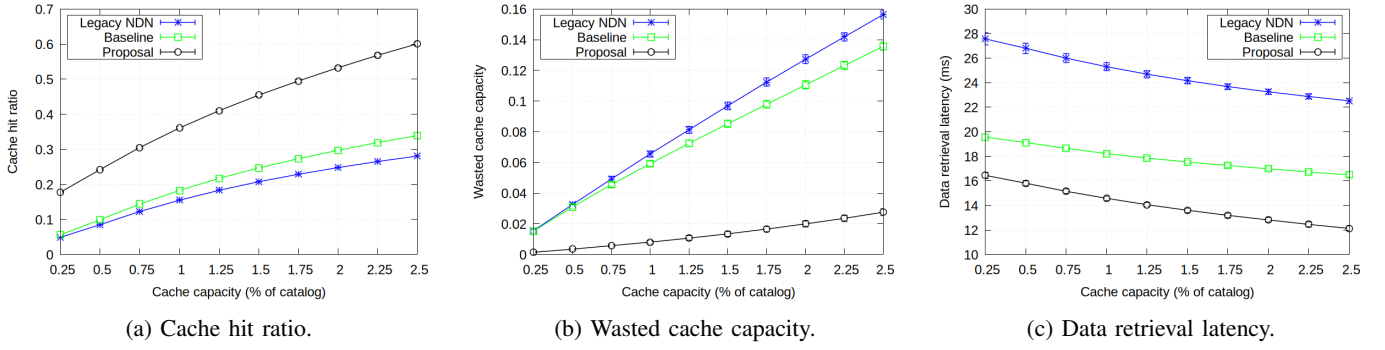


Fig. 2: Performance evaluation results when varying the cache capacity (40 Interests/s).

i.e., the fraction of cached entries that do not correspond to a distinct semantic class. Finally, we compute the *data retrieval latency*, i.e., the time between Interest transmission and the corresponding Data packet delivery. Similarly to [5], and in alignment with empirical measurements in edge networks, this comprises an access latency between consumer and edge node, uniformly drawn at random in the 2–10 ms range, plus a resolution component that depends on where the request is satisfied: a fixed 1 ms for a CS hit, or, otherwise, a latency uniformly drawn at random in the 10–20 ms range for a producer fetch via the FIB, or in the 30–60 ms range for a round trip to the centralized resolver. To account for possible false positives introduced by semantic matching, each semantic resolution is evaluated against the ground-truth class available in the simulator. Whenever the selected cached content belongs to a different semantic class than the requested content, the semantic hit is considered unsuccessful and the Interest is assumed to be retransmitted toward the centralized resolver, adding the corresponding round-trip delay.

Results are averaged over 15 independent runs and reported with 95% confidence intervals.

Results. Fig. 2a reports the average cache hit ratio at the edge access node as the cache capacity increases. As expected, the hit ratio grows for all schemes as more capacity becomes available, with the proposal consistently achieving the highest hit ratio and the gap over both benchmarks widening as the cache grows. This follows directly from the selective admission criterion: rather than storing fewer entries, the proposal uses the same cache capacity to represent more distinct semantic classes, each able to satisfy multiple semantically equivalent requests.

Fig. 2b reports the wasted cache capacity as the cache capacity increases. The legacy NDN and semantic baseline approaches waste a substantial and growing share of the CS on semantically redundant entries, since cache-everything admission keeps accumulating multiple producer variants of the same class regardless of whether it already has a representative cached. The proposal instead keeps redundancy consistently low and increases the diversity of cached contents. Consequently, more Interests are satisfied directly at the edge node, reducing accesses to remote producers or the centralized

resolver. This translates into the lowest data retrieval latency among all schemes, as shown in Fig. 2c.

V. CONCLUSIONS

This paper introduced a lightweight semantic caching policy for NDN-based IoT scenarios, admitting new content only when it extends the semantic diversity of the cache. Results show that the proposal improves the cache hit ratio, reduces semantic redundancy, and consequently lowers retrieval latency with respect to both legacy exact-match caching and a semantic caching baseline that disregards redundancy. As future work, we plan to investigate replacement policies that jointly account for semantic diversity and access patterns.

ACKNOWLEDGMENT

This work was partially supported by the FFABR Unime Giovani Ricercatori 2025 grant, funded by the University of Messina.

REFERENCES

- [1] Y. Hajjaji *et al.*, “Big data and IoT-based applications in smart environments: A systematic review,” *Computer Science Review*, vol. 39, p. 100318, 2021.
- [2] Z. Zhang *et al.*, “In-network caching for ICN-based IoT (ICN-IoT): A comprehensive survey,” *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14595–14620, 2023.
- [3] A. Afanasyev *et al.*, “A brief introduction to Named Data Networking,” in *IEEE MILCOM*, pp. 1–6, 2018.
- [4] K. Chan *et al.*, “Fuzzy Interest forwarding,” in *Asian Internet Engineering Conference*, pp. 31–37, 2017.
- [5] M. Amadeo, S. Serrano, A. Molinaro, M. Nitti, and G. Ruggeri, “Enhancing IoT service discovery through semantic name-based forwarding,” *IEEE Internet of Things Magazine*, 2026.
- [6] Z. Awais *et al.*, “ISCC: Intelligent Semantic Caching and Control for NDN-Enabled Industrial IoT Networks,” *IEEE Access*, 2025.
- [7] M. Amadeo *et al.*, “Service provisioning in Digital Twin Networks with semantic-aware Information Centric Networking,” in *IEEE CCNC*, 2026.
- [8] F. Bang, “GTPache: An open-source semantic cache for LLM applications enabling faster answers and cost savings,” in *3rd Workshop for Natural Language Processing Open Source Software*, 2023.
- [9] G. M. Raza *et al.*, “Inf-NDN IoT: An intelligent naming and forwarding in Name Data Networking for Internet of Things,” *IEEE Access*, vol. 12, pp. 114319–114337, 2024.
- [10] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” in *Conf. on Empirical Methods in Natural Language Processing*, 11 2019.
- [11] B. Chen, L. Liu, M. Sun, and H. Ma, “IoTCache: Toward data-driven network caching for Internet of Things,” *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10064–10076, 2019.