Enhancing IoT Service Discovery Through Semantic Name-Based Forwarding

Marica Amadeo, *Member, IEEE*, Salvatore Serrano, Antonella Molinaro, *Senior Member, IEEE*, Michele Nitti, *Senior Member, IEEE*, and Giuseppe Ruggeri, *Member, IEEE*

Abstract

Named Data Networking (NDN) has emerged as a promising communication paradigm for the Internet of Things (IoT). In NDN, application-level service names, expressed as Uniform Resource Identifiers (URIs), are used directly at the network layer to facilitate service discovery and retrieval. Forwarding decisions at each NDN router rely on the longest-prefix match between the name carried in the incoming service request and the entries in its Forwarding Information Base (FIB). However, in highly dynamic and heterogeneous IoT environments, client applications, interested in accessing a given IoT service, may not always know the exact service names in advance. Additionally, different service providers, each using distinct namespaces but offering functionally equivalent services, may be capable of fulfilling the same request. This paper explores an alternative name-based forwarding approach that leverages semantic similarity to improve the chances of successfully resolving incoming IoT service requests. When an NDN node cannot find an exact name match, it applies a semantic similarity-aware mechanism to identify the most relevant forwarding path in the FIB toward a potential IoT service provider. This approach is implemented using deep learning models for sentence embedding, seamlessly integrated into the NDN forwarding fabric. Evaluation results demonstrate that semantic-aware forwarding significantly enhances service discovery in scenarios where exact name matching fails, while maintaining low processing times.

I. Introduction

The Internet of Things (IoT) has fundamentally reshaped the way information and services are generated, shared, and consumed, driving significant advancements across various application domains such as smart cities and healthcare [1]. This transformation is driven by the vast and diverse ecosystem of IoT devices, ranging from low-power environmental sensors to autonomous vehicles and cyber-physical systems, acting as service providers in dynamic and often intermittent network conditions. In this context, IoT applications require efficient mechanisms for service discovery and provisioning that operate independently of specific providers. For example, in a smart city, a user might request a service of traffic updates or crowd monitoring insights without needing to know the exact source of such services, yet expects accurate, reliable, and timely results [2]. To meet these expectations, Information-Centric Networking (ICN) has emerged as a promising solution for IoT applications [3]. Unlike traditional host-centric IP networking, which focuses on addressing specific devices, ICN prioritizes contents and services by leveraging application-level names directly at the network layer. Additionally, ICN natively supports in-network caching, improving both service availability and scalability.

The most widely used ICN architecture in IoT is Named Data Networking (NDN), which leverages hierarchical names in the form of Uniform Resource Identifiers (URIs) [4]. For example, an IoT service name *smartcity/district5/parking/availability* can be used to request real-time parking availability information in District 5 of a smart city.

In NDN, forwarding named requests, referred to as Interest packets, is based on an *exact* longest-prefix match lookup between the name carried in the request and the names stored in the local Forwarding Information Base (FIB). Several naming schemes for IoT contents and services have been proposed over the years [5], typically assuming that clients and service providers adopt a shared and consistent naming convention. However, in real-world IoT scenarios, this assumption cannot always hold, due to the high heterogeneity of IoT devices, applications, and administrative domains. Providers offering similar services may adopt different namespace structures, leading to inconsistencies in name representation. As a result, semantically equivalent service requests may fail to be resolved if expressed with syntactically different names. For instance, two pollution-monitoring providers might use distinct naming schemes despite offering the same type of service, e.g., */smartcity/environment/pollutionLevel*, */urban/environmentStatus/pollutionDegree*. Clients may only have partial

This work was supported in part by the National Research project PRIN 2022 "TOGETHER: Models and algoriThms fOr 6G Era digital Twin orcHEstratoR" through the European Union–Next Generation EU, Mission 4 Component 1 CUP C53D23000450006.

Digital Object Identifier: 10.1109/MIOT.2025.3587052
Date of Current Version: 24 February 2026
Date of Publication: 10 July 2025

Marica Amadeo (corresponding author) and Salvatore Serrano are with the Department of Engineering, University of Messina, 98166 Messina, Italy, and also with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 43124 Parma, Italy; Antonella Molinaro is with DIIES Department, University Mediterranea of Reggio Calabria, 89122 Reggio Calabria, Italy, also with Paris-Saclay University, 91190 Gif-sur-Yvette, France, and also with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 43124 Parma, Italy; Michele Nitti is with DIEE Department, University of Cagliari, 09123 Cagliari, Italy, and also with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 43124 Parma, Italy; Giuseppe Ruggeri is with DIIES Department, University Mediterranea of Reggio Calabria, 89122 Reggio Calabria, Italy, and also with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 43124 Parma, Italy.

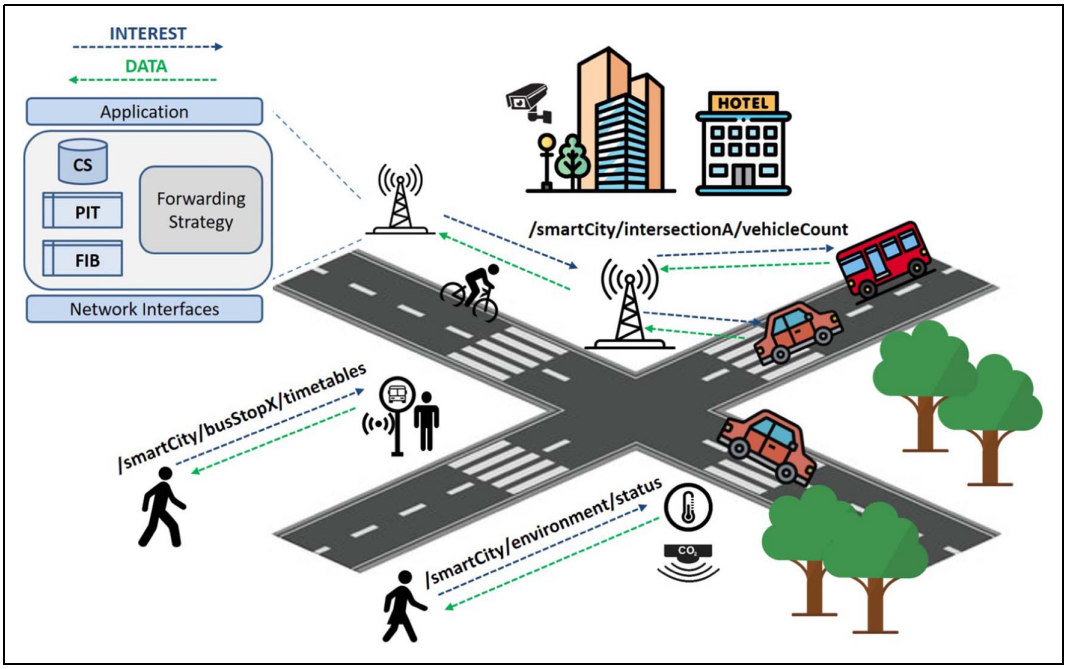

FIG. 1. NDN node architecture and packet exchange for IoT service access in a smart city scenario.

knowledge of these naming conventions, making exact name matching impractical in some cases.

The study in [6] introduced a fuzzy forwarding mechanism leveraging semantic similarity between the names in the Interests and those in the FIB entries, when exact name matching fails. This preliminary approach focused on single-component name matching, using Word2Vec to generate vector-based representations of name components and applying cosine similarity to measure their semantic closeness.

In this paper, we propose an advanced scheme for name-based forwarding with semantic similarity awareness, designed to enhance IoT service discovery. To compute semantic similarity, we utilize deep learning models, specifically Sentence Transformers [7], which effectively generate vector embeddings of NDN names. These numerical representations capture the semantic meaning of sentences, instead of a single name component, enabling comparisons even when the names in Interest packets differ from those stored in the FIB. To ensure a fast discovery process and limit computational overhead, we introduce a caching mechanism for name embeddings and their corresponding discovered routes.

We conduct an extensive simulation campaign using different Sentence Transformer models across varying IoT service request rates. Results demonstrate the feasibility and efficiency of our approach in successfully handling requests when exact name matching fails, highlighting the critical role of embedding caching in significantly reducing computation time.

Our findings underscore the importance of integrating advanced semantic similarity techniques to enhance the flexibility and efficiency of NDN-based IoT networks, paving the way for more adaptive and resilient service provisioning.

The remainder of the paper is organized as follows. Section II presents the NDN architecture and related work. Section III illustrates the proposed strategy, while Section IV details the performance evaluation. Finally, Section V concludes the paper.

II. Background and Motivations

A. NDN Basics

NDN communication is receiver-driven: clients transmit Interest packets carrying the name of the requested information, or service, and the network forwards these requests towards a provider using an anycast approach. The provider may either send the requested information directly, or execute the service and then return the result [8]. In both cases, Data packets are transmitted, carrying the same name as the corresponding Interest.

Fig. 1 shows the NDN node architecture and packet exchange for IoT service access in a smart city scenario.

Each NDN node maintains three tables at the forwarding plane: a Content Store (CS) for caching incoming Data packets, a Pending Interest Table (PIT) to track unsatisfied Interests, and a FIB that stores pairs of name prefixes and their corresponding outgoing interfaces. When an Interest packet is received, the node first checks the CS for a matching Data packet. In case of failure, the node examines the PIT. A match here indicates that the same request has already been forwarded, so the node simply aggregates the Interest without retransmitting it. If no match is found, the node looks in the FIB to determine the appropriate outgoing interface for forwarding the Interest toward the provider(s). Forwarding relies on finding an exact longest-prefix match between the name in the Interest and an entry in the FIB. A name-based routing protocol is used by providers to advertise service availability and disseminate name prefixes across the network. Data packets, instead, just follow the chain of PIT entries back to the requesting clients.

When no matching entry is found in the FIB, the node discards the request. Optionally, it can transmit back a Negative Acknowledgement (NACK) to inform previous nodes that no route is available.

Specific naming conventions and namespaces can be designed to effectively support IoT name lookups [5], [9]. Typically, names reflect the IoT application domain, the specific service being offered and its scope. For example, an IoT service name like */smartcity/intersectionA/vehicleCount* can be used to request a smart city service, giving information on the average number of vehicles passing through a specific intersection (labeled *A*) in a smart city. However, the exact-match dependency inherent in such naming conventions presents significant challenges in the IoT context, where heterogeneous (often mobile) clients may not always have precise knowledge of the naming structure or the exact prefix of the service they need. For example, the same service might be referred to by different terms or languages, or naming conventions may be inconsistently applied. The dynamic nature of IoT providers further complicates matters and makes static name matching largely ineffective.

B. Semantic-Aware Forwarding

A solution to improve IoT service discovery involves extending the NDN forwarding fabric to incorporate semantic understanding. While some studies have explored this approach [6], [10], [11], they are primarily limited to basic data retrieval scenarios and offer preliminary evaluations.

The authors of [6] introduced the Fuzzy Interest Forwarding (FIF) mechanism, which utilizes vector-based models, specifically Word2Vec, to generate vector representations of name components, allowing the system to identify and forward Interests based on semantic proximity. Cosine similarity is applied to determine how closely aligned two names are in the vector space, with a higher cosine similarity indicating stronger semantic connections. The evaluation, which focused on the semantic similarity of individual name components under limited Interest traffic (i.e., 10 Interests/s processed by an NDN router), showed improved data retrieval rates compared to exact-match forwarding. The study was extended in [10], where the Interest rate was increased to 20 Interests/s.

The authors in [11] introduced a hybrid naming structure combining hierarchical and flat name components, to better manage the large variety of IoT devices and services. The forwarding process leverages contextual metadata to enhance the matching between Interest and Data. However, the computation of semantic similarity depends on predefined naming conventions and metadata annotations rather than advanced machine learning models. This limits the system's adaptability in more complex and dynamic IoT scenarios.

Similar to [6], our work also explores semantic similarity through vector-based models. However, our approach leverages deep learning techniques, specifically designed for sentence-level understanding, providing enhanced contextual awareness for name-based forwarding. By applying Sentence Transformers models [7] to the entire service name, our approach captures richer contextual relationships, enabling more accurate semantic matching that goes beyond individual name components. Moreover, to reduce computation time, we introduce a caching mechanism that stores the Interest name along with its embedding and the corresponding forwarding decision, eliminating the need for recomputation.

III. Semantic-Driven Service Discovery

This section presents the Semantic-Aware Forwarding (SAF) strategy, designed to enhance IoT service discovery when exact name matching fails.

A. Understanding Semantic Similarity

Semantic similarity measures how closely two names or phrases are related in terms of their meaning. It determines whether names such as *smartCity/transport/trafficMonitoring* and *smartTown/urbanMobility/trafficStatus* represent similar concepts to a certain extent. This challenge is central to Natural Language Processing (NLP), a field that enables machines to effectively understand and process human language.

Transformer-based models have greatly improved the ability to capture contextual relationships, enabling advanced semantic similarity approaches [12]. Notably, the Minimal Language Model (MiniLM) [13] and Masked and Permuted Pre-training for Language Understanding (Mpnet) [14] are two examples of compact yet powerful models balancing efficiency and accuracy. MiniLM is a lightweight model that retains the key features of larger language models while being optimized for low-resource environments. In contrast, Mpnet combines different learning techniques to better capture word relationships, offering higher accuracy but with increased computational complexity.

Although MiniLM and Mpnet perform well in general NLP tasks, they are not optimized for capturing semantic similarity between full sentences. The Sentence Transformers framework [7] adapts these models using Siamese or triplet architectures to generate fixed-size sentence embeddings, which can be efficiently compared using cosine similarity. These models are also fine-tuned on dedicated datasets to improve performance in semantic matching tasks.

By leveraging Sentence Transformers' ability to understand the contextual relationships of NDN hierarchical names, our approach enhances forwarding decisions, making it well-suited for adapting to the dynamic nature of IoT naming conventions.

B. SAF

The proposed SAF strategy can be selectively deployed at edge access NDN routers, where naming heterogeneity can be detected, and does not require changes to the broader network infrastructure. Its integration with the legacy NDN forwarding fabric can be facilitated through network softwarization and virtualization technologies.

Basics. Like in standard NDN deployments, we assume that network routing protocols populate the FIBs of NDN nodes. Once the FIB entries are established, their names are processed to generate the corresponding embeddings. New FIB entries can be dynamically created at runtime as nodes operate, requiring only the generation of the embeddings for the newly added entries. Similarly, FIB entries can be invalidated and removed if the corresponding routes become unavailable.

Upon receiving an Interest packet, an NDN node *n* first performs the standard lookup process. If no exact match is found in the CS, PIT, or FIB, the node activates semantic understanding to identify the most relevant route for the request, if available. Therefore, node *n* extracts the service name from the Interest and generates its embedding using a Sentence Transformer model. The resulting embedding is then

compared to the precomputed embeddings of FIB entry names using the cosine similarity metric. The resulting *similarity score* (S_{score}) ranges between zero and one: values close to one indicate a strong semantic relationship between the Interest and a FIB entry; values close to zero suggest no meaningful similarity. If a sufficiently similar FIB entry is found, the node forwards the Interest toward the potential provider; otherwise, the Interest is discarded.

Threshold-based decision. To implement the forwarding decision, node *n* leverages a threshold-based mechanism. For each FIB entry, the node calculates the similarity score with the Interest name. If S_{score} meets or exceeds a predefined threshold *Th*, the FIB entry is deemed sufficiently relevant, and the corresponding outgoing network interface may be selected for forwarding.

In the current design, node *n* forwards the Interest using the FIB entry that has the highest S_{score}, among those that satisfy the condition $S_{score} \geq Th$. In case two or more entries share the same S_{score}, node *n* selects the one with the lowest Round Trip Time (RTT). To facilitate downstream forwarding, node *n* prepends the selected FIB name to the Interest name, using the delimiter |. This structured format allows subsequent nodes to recognize the discovered name and forward the request toward the provider, without needing to recompute the S_{score}, since their FIB entries would have already been established by the routing protocol. Before forwarding, node *n* records the Interest in the PIT to enable the transmission of the expected Data back to the requester. If no FIB entry satisfies $S_{score} \geq Th$, the Interest is discarded and a NACK is sent back to the client to indicate no suitable match.

Caching of embeddings. Despite being optimized for semantic tasks, Sentence Transformers models may still introduce computational overhead. To mitigate this, we implement a caching mechanism for the most frequently used embeddings, reducing redundant computations and improving response times. While embeddings of FIB names are precomputed and stored within an additional field of each FIB entry, embeddings of Interest names are computed on demand and stored in a dedicated cache, referred to as the *Embeddings Store* (ES). Each entry in the ES includes: the Interest name, the corresponding embedding, the discovered name from the FIB with the outgoing interface and the corresponding S_{score}. Before computing the embedding of an incoming Interest, node *n* first checks the ES to determine if the same request has already been processed. If a match is found, the information regarding the outgoing interface is reused. Only if no match exists in the ES, node *n* computes the new embedding and adds it to the ES.

To efficiently manage storage resources, we adopt a Least Recently Used (LRU) cache replacement policy, which balances simplicity and effectiveness by ensuring that frequently accessed embeddings remain available while less relevant entries are evicted.

When the FIB table is updated, the ES is checked to maintain consistency. If a FIB entry is removed (e.g., due to a provider becoming unavailable), any ES entries referencing that route are invalidated. If an alternative valid route exists, the ES is updated accordingly, ensuring no disruption. Similarly, when a new FIB entry is added, the ES is checked to identify a potential new best route. In both cases, since the embeddings have already been computed and stored, the computational overhead remains minimal.

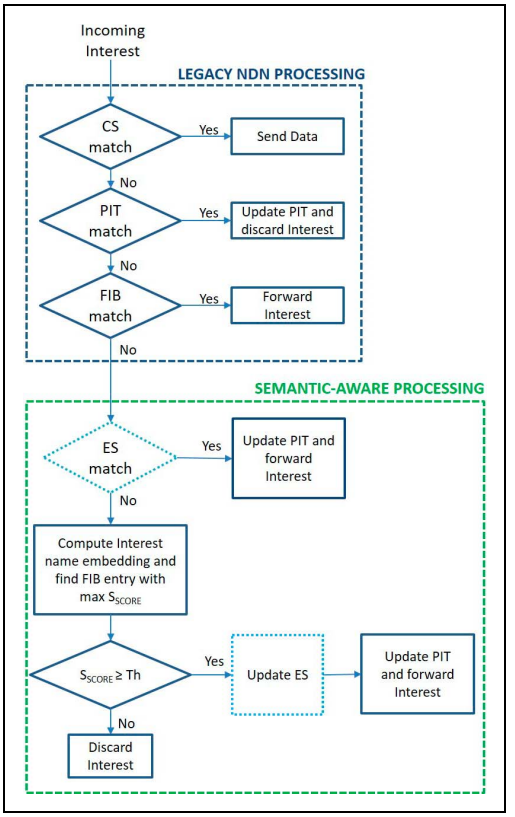

FIG. 2. Interest packet processing with SAF. The additional processing steps introduced by the semantic-aware forwarding mechanism are enclosed in the green dashed box, while operations involving the Embedding Store (ES) are indicated by dashed cyan lines.

The flowchart in Fig. 2 summarizes the Interest packet processing with SAF+ES.

IV. Performance Evaluation

To evaluate the performance of the proposed strategy, we conduct a realistic simulation campaign that replicates a typical NDN-based IoT service discovery scenario. Our goal is twofold: *(i)* to assess feasibility and computational overhead of semantic-aware forwarding at the NDN router level, and *(ii)* to evaluate the effectiveness of semantic recognition in service discovery. To this end, we developed a custom Python-based simulator that models the NDN architecture while enabling tight integration with pretrained NLP models. This environment allows precise measurement of computation delays, as well as fine-grained control over Interest generation and forwarding behavior.

A. Simulation Settings

1) Scenario: We simulate a scenario where both clients and providers are located within an edge network domain, such as a smart city environment. An NDN router, functioning as an access-edge node, processes a continuous stream of Interest packets, with client request rates ranging from 30 to 300 Interests/s. Each Interest represents an IoT service request from a client, and the router processes each request individually, determining the appropriate

forwarding path to the corresponding IoT provider. Each FIB entry at the NDN router corresponds to a distinct provider offering an IoT service and is associated with a specific outgoing interface. For instance, a FIB size of 50 entries reflects 50 available IoT providers. The one-way link latency between router and clients is set to 10*ms*, while the latency between router and IoT providers is 20*ms*, in alignment with empirical measurements in edge networks.

We evaluate the proposed SAF+ES strategy alongside two benchmarks: *(i)* a vanilla SAF scheme without ES, representative of [6], and *(ii)* a standard NDN approach relying on exact name matching. Since our primary focus is on evaluating forwarding performance without the influence of in-network caching dynamics, we set the CS size to zero in both configurations. The PIT size is instead unbounded in our experiments to prevent packet drops due to table overflows, ensuring that all transmitted Interests can be tracked.

In the standard NDN approach, if the FIB matching fails, the router transmits a NACK back to the client. Conversely, in the SAF approach, the router leverages semantic-similarity to identify the most relevant path for the request. If the similarity score is greater than or equal to $Th = 0.7$, the Interest is forwarded towards the provider. Otherwise, a NACK is sent back to the client. The threshold value was selected through a tuning campaign, evaluating the frequency of correct and incorrect forwarding of Interests, to ensure a sufficient degree of semantic closeness between an Interest name and a FIB entry, while preventing semantically unrelated matches. Specifically, we measured: true positives (correctly forwarded Interests), false positives (incorrectly forwarded Interests), true negatives (correctly rejected Interests), and false negatives (missed forwarding opportunities). We observed that lower thresholds (e.g., $Th \leq 0.6$) increase forwarding coverage but lead to more false positives due to overly permissive matching, while higher thresholds (e.g., $Th \geq 0.8$) reduce false positives but increase false negatives, thus missing valid forwarding opportunities. A threshold of $Th = 0.7$ provides the best trade-off: it guarantees high forwarding accuracy with minimal incorrect matches across different Interest catalog configurations. Experiments were conducted on a machine equipped with an Intel® Core™ Ultra 7 155H processor (up to 4.8 GHz) and 15 GB of DDR4 RAM. The simulator runs as a single-threaded Python application.

2) Request Pattern: Unless otherwise specified, we configure the FIB with 50 entries, each one associated to an IoT smart city service and generate a catalog of 350 Interest names to simulate different request patterns. Each of the 50 names in the FIB is associated with a cluster of semantically related Interest names, reflecting real-world variations in client queries. For each service name, we consider one exact match Interest name (the canonical service name) and six synonymous variations, resulting in a total of 350 distinct Interest names. For instance, the service */smartCity/mobility/bus/timetables* may be requested using alternative names such as: */smartMobility/bus/timetables* or */urbanArea/transportation/bus/schedules*.

To model service popularity, we consider two distributions: *(i)* uniform distribution, where all services are requested with equal probability; *(ii)* Zipf distribution, with a skewness parameter of 1.2, which models real-world request patterns where some services are significantly more popular than others.

3) SAF Models: We incorporate Sentence Transformers [7] into the NDN forwarding fabric in order to compute the semantic similarity between Interest names and FIB entries' names. We select three Sentence Transformers models, namely *all-mpnet-base-v2*, *all-MiniLM-L12-v2*, and *all-MiniLM-L6-v2*.

The *all-mpnet-base-v2* model (labeled as Mpnet in the plots) offers the highest embedding quality at the cost of increased computational complexity. With a model size of 420MB, it maps sentences into a 768-dimensional dense vector space. In contrast, the *all-MiniLM-L6-v2* model (labeled as MiniLM-L6 in the plots) is faster while maintaining competitive performance. With a compact size of 80MB, it maps sentences into a 384-dimensional dense vector space, offering an efficient alternative for real-time applications. The *all-MiniLM-L12-v2* model, an extension of the L6 variant (labeled as MiniLM-L12 in the plots), includes 12 transformer layers. Despite sharing the same 384-dimensional embedding space as *all-MiniLM-L6-v2*, its increased depth results in improved semantic representation quality. The models are pretrained offline by their developers on large public datasets for semantic tasks and are integrated directly into the NDN forwarding logic. Since forwarding must operate under real-time constraints, only inference is performed at runtime. Embeddings of Interest names are therefore computed upon Interest reception, unless already available in the ES. Although we do not apply model fine-tuning in our evaluation, this step could be considered in future work to better align the models with NDN-specific naming conventions.

B. Efficiency Analysis

We evaluate the efficiency of the forwarding strategy in terms of *service discovery time*, defined as the total delay incurred in discovering a provider. For SAF, this includes: *(i) client-to-router communication time*, referring to the time required for an Interest packet to travel from the client to the NDN router; *(ii) queuing time*, representing the duration an Interest spends in the processing queue before being handled by the semantic-aware mechanism, occurring when the router is already busy processing previous requests; *(iii) processing time*, referring to the actual time required to apply semantic recognition and compute the forwarding decision; *(iv) router-to-provider communication time*, referring to the time taken for the Interest packet to reach the designated provider. The actual computation time at an NDN router is therefore the sum of queuing time and processing time. When the Interest arrival rate exceeds the router's processing capacity, Interests are queued, leading to increased latency. Conversely, when the arrival rate is lower, queuing time becomes negligible.

In the standard NDN approach, instead, the service discovery time does not include the time required for semantic recognition. When no match is found in the FIB, it accounts for the additional communication delays due to NACK transmissions and client-side retransmissions of Interests with a modified service name.

We first assess the service discovery time of the vanilla SAF scheme and the NDN baseline. Fig. 3 shows the service discovery time when varying the Sentence Transformers models and the Interest rate, assuming a uniform popularity distribution. As

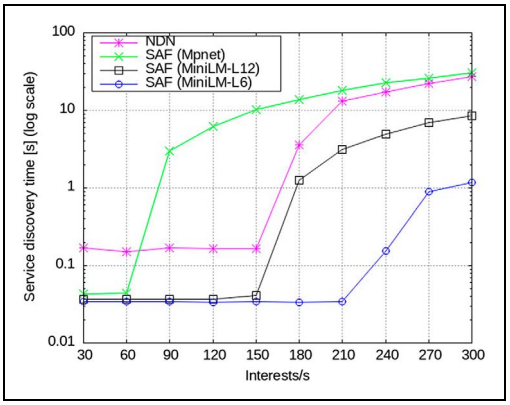

FIG. 3. Service discovery time under the vanilla SAF scheme (without ES) and the standard NDN approach, evaluated across different interest rates and Sentence Transformers models, assuming a uniform popularity distribution.

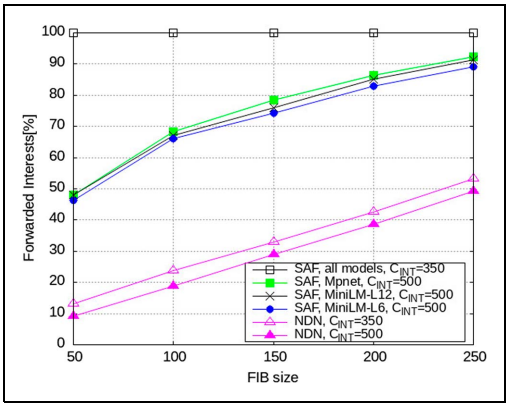

FIG. 4. Percentage of forwarded Interests when varying the number of FIB entries and assuming a uniform popularity distribution.

expected, the Mpnet model incurs the highest latency, due to its greater computational complexity. Conversely, MiniLM-L6 remains the most efficient across all Interest rates, due to its lighter structure. Beyond 210 Interests/s, the service discovery time for MiniLM-L6 increases, indicating a bottleneck effect where the router struggles to keep up with the growing arrival rate. However, its latency remains consistently lower than that of standard NDN, despite performing similarity-based matching. The MiniLM-L12 model follows an intermediate trend, outperforming Mpnet but exhibiting a rise in computation time beyond 150 Interests/s. This suggests that deeper transformer models may introduce scalability issues in high-load conditions. The NDN baseline incurs higher delays than the MiniLM models, as clients struggle to find an exact name match and are forced to retransmit Interests multiple times.

C. Effectiveness Analysis

To evaluate forwarding effectiveness, Fig. 4 shows the percentage of Interests successfully forwarded *on the first attempt*, comparing SAF with standard NDN. In addition to the 350-Interest catalog used in the previous analysis (labeled as $C_{INT} = 350$), in which all Interest names correspond to available services (i.e., the ground truth contains only true positives), we considered a larger catalog of 500 Interests (labeled as $C_{INT} = 500$). In this case, we introduced a broader set of Interest names: only 50% correspond to actual available services (i.e., are semantically linked to FIB entries), and the other 50% are semantically unrelated. In both cases, a similarity threshold of $Th = 0.7$ is applied, with the request rate fixed at 300 Interests/s and a uniform popularity distribution.

It can be observed that larger FIB sizes improve Interest name coverage, leading to higher forwarding rates due to the increased likelihood of finding either an exact match or a semantically similar name. SAF significantly enhances service discovery capability compared to standard NDN. It achieves 100% correct forwarding when using the 350-Interest catalog across all three models, while successfully routing over 47% of incoming requests with the larger catalog, which aligns closely with the 50% ground truth availability. Conversely, since standard NDN forwarding relies on exact name matching, any deviation in Interest names from those stored in the FIB results in a forwarding failure and the generation of a NACK, leading to significantly lower performance in name resolution.

These results highlight the clear advantage of semantic matching, which enables flexible name interpretation and supports service discovery even when exact matches are unavailable. The choice between Mpnet, MiniLM-L6 and MiniLM-L12 has minimal impact, as they exhibit nearly identical performance in semantic recognition under the given scenario. However, due to Mpnet's higher computational overhead, we focus on the MiniLM variants in the following analysis.

D. SAF+ES Analysis

Latency evaluation. Fig. 5(a) shows the service discovery time when implementing SAF+ES and varying the Interest rate, the model and the popularity distribution. It can be observed that caching discovered routes significantly reduces the latency compared to the vanilla SAF shown in Fig. 3. Indeed, a matching in the ES avoids the computationally expensive step of generating a new embedding and computing multiple similarity scores. The latency reduces when the request pattern follows a Zipf distribution, since requests concentrate on a few popular services and there is a higher chance to find the route available in the ES. MiniLM-L6 achieves the fastest and most stable performance, enhancing scalability under both popularity distribution. Conversely, MiniLM-L12 exhibits an increase in service discovery time at higher Interest rates, due to its higher computational overhead when processing a larger number of requests.

To better understand the impact of ES on the SAF's computational latency, Fig. 5(b) shows the processing time. As expected, MiniLM-L6 exhibits lower processing times compared to MiniLM-L12, owing to its reduced computational complexity. Interestingly, processing time decreases as the Interest rate increases. This trend is due to the ES caching mechanism: when an Interest arrives, SAF checks if its name has been previously processed. If so, the cached embedding and forwarding decision are reused, avoiding recomputation. At higher Interest rates, especially under Zipf-like distributions where a few service names dominate, the likelihood of repeated names increases, boosting the cache hit rate and reducing processing overhead.

Discovery effectiveness is not altered by the ES implementation, as caching impacts service provisioning time rather than semantic recognition. Given that

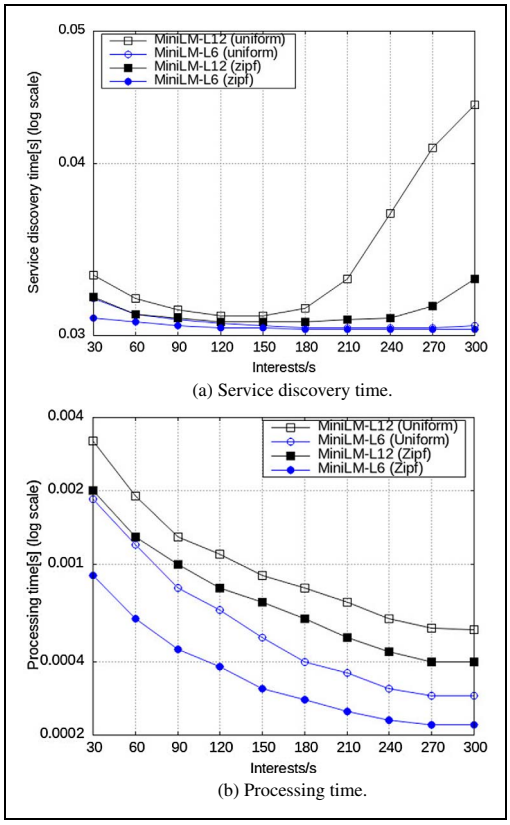

FIG. 5. Performance metrics when implementing SAF+ES and varying the incoming Interest rate, the Sentence Transformers model and the popularity distribution.

MiniLM-L6 achieves comparable recognition performance to MiniLM-L12, while requiring fewer computational resources, it represents a more efficient choice for practical deployment.

Memory analysis. In our experimental setup, loading the MiniLM-L6-v2 model, during initialization, incurs a one-time cost of approximately 2.8 seconds wall-clock time and 2.0 seconds CPU time, with a resident memory size (RSS) increase of 141MB. Caching embeddings introduces additional memory usage at the edge router: each embedding, represented as a 384-dimensional vector of 32-bit floating-point values, requires approximately 1.5KB per FIB entry. Even with 50,000 stored Interest embeddings, the total memory footprint remains under 75MB. When combined with the 141MB required to load the model, the overall memory usage remains well within the capabilities of modern edge devices.

V. Conclusion

This paper presented a semantic-aware forwarding strategy to enhance IoT service discovery in NDN networks. Results demonstrate both its effectiveness in semantic matching and its efficiency, with low computation times. Future work will focus on extending the simulation campaign to a full edge network topology and further optimizing the discovery process.

References

[1] Y. Hajjaji, W. Boulila, I. R. Farah, I. Romdhani, and A. Hussain, "Big data and IoT-based applications in smart environments: A systematic review," *Comput. Sci. Rev.*, vol. 39, 2021, Art. no. 100318.

[2] B. Nour et al., "A survey of internet of things communication using ICN: A use case perspective," *Comput. Commun.*, vol. 142, pp. 95–123, Jun. 2019.

[3] A. Djama, B. Djamaa, and M. R. Senouci, "Information-centric networking solutions for the internet of things: A systematic mapping review," *Comput. Commun.*, vol. 159, pp. 37–59, Jun. 2020.

[4] W. Shang et al., "Named data networking of things," in *Proc. IEEE Int. Conf. Int. Things Des. Implementation (IoTDI)*, 2016, pp. 117–128.

[5] B. Nour, K. Sharif, F. Li, H. Moungla, and Y. Liu, "A unified hybrid information-centric naming scheme for IoT applications," *Comput. Commun.*, vol. 150, pp. 103–114, Jan. 2020.

[6] K. Chan, B. Ko, S. Mastorakis, A. Afanasyev, and L. Zhang, "Fuzzy interest forwarding," in *Proc. 13th Asian Int. Eng. Conf.*, 2017, pp. 31–37.

[7] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2019, p. 11.

[8] M. Amadeo, G. Ruggeri, C. Campolo, and A. Molinaro, "IoT services allocation at the edge via named data networking: From optimal bounds to practical design," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 661–674, Jun. 2019.

[9] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Information centric networking in IoT scenarios: The case of a smart home," in *Proc. IEEE ICC*, 2015, pp. 648–653.

[10] S. Mastorakis, K. Chan, B. Ko, A. Afanasyev, and L. Zhang, "Experimentation with fuzzy interest forwarding in named data networking," 2018, *arXiv:1802.03072*.

[11] 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.

[12] K. S. Kalyan, A. Rajasekharan, and S. Sangeetha, "AMMUS: A survey of transformer-based pretrained models in natural language processing," 2021, *arXiv:2108.05542*.

[13] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 5776–5788.

[14] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "MPNET: Masked and permuted pre-training for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16857–16867.