Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Advancements in Cache Management for Named Data Networking: A Survey

Sushil Kumar Bagi 1, Prof. (Dr.) Neeraj Kumar [2]

[1] Research Scholar, Suresh Gyan Vihar University, Jaipur,INDIA
2 Professor, Suresh Gyan Vihar university, Jaipur,INDIA

sushil.23183963@mygyanvihar.com, *neeraj.kumar1@mygyanvihar.com*

***Abstract*—** Named Data Networking (NDN) has emerged as a promising architecture to improve the efficiency and scalability of content retrieval by focusing on content names rather than host addresses. A core component of NDN is caching, which enables frequently accessed contents to be stored closer to consumers, thereby reducing latency and bandwidth consumption. This paper presents a comprehensive survey of caching strategies in NDN, covering both traditional approaches (e.g., FIFO, LRU, LFU, RR) as well as advanced methods such as probabilistic, machine learning-based, and PIT/FIB-aware caching. The strengths and limitations of each strategy are analysed, and their impact on performance metrics such as cache hit ratio, latency, and resource utilization is discussed. Finally, the paper highlights open challenges and research directions to guide the development of more intelligent and adaptive caching solutions in NDN.

***Keywords*—** Named Data Networking (NDN), Cache Management, Caching Strategies, Content Placement & Replacement

I. Introduction

The Named Data Networking (NDN) has developed as a viable alternative to the traditional TCP/IP-based networking paradigm by emphasising content dispatching rather than host-to-host communication. Unlike conventional networks, NDN integrates in-network caching, reducing data retrieval latency and improving bandwidth utilization. Still, optimising caching strategies remains a big challenge due to the dynamic nature of data content popularity and the scalability demands of modern usage. This survey examines recent studies on caching strategies in NDN, providing a classification of approaches and identifying research gaps to guide future work. Several research papers have been published regarding caching strategies in NDN. In the study [1], the authors proposed a Graph Neural Network (GNN)-based caching approach, which performed better than basic strategies such as LFU and LSTM-ED in terms of cache hit ratio and latency reduction. Similarly in the research paper [2] authors has discussed the importance of NDN-based caching for indoor positioning systems, showing how it reduces server load compared to traditional architectures as well as the authors found that after applying NDN-based indoor positioning and navigation system with previous algorithm the improvements achieved in the position of floor detection, localization and navigation by 77%, 33% and 99 % respectively. In [3], the authors introduced a social-aware caching method, where selecting influential consumers in Online Social Networks

(OSNs) slightly improved cache hit ratios, thereby reducing network traffic and operational cost in self-operated content delivery networks. Although these advancements exist, existing caching strategies are often based on static or historical data to determine content popularity, limiting their effectiveness in dynamic environments such as real-time streaming platforms and social media. Moreover, many researchers do not consider features from both the Pending Interest Table (PIT) and Forwarding Information Base (FIB) in cache decision-making, leading to suboptimal performance in high-traffic networks. Additionally, most AI/ML-based caching approaches rely on supervised learning, requiring labelled datasets and large-scale training, which makes them complex and impractical for real-time decision-making in NDN, even though routers maintain local tables for forwarding requests.

Despite the wide range of caching strategies proposed in NDN. Although few studies have explored the cache replacement technique based on Meta data of FIB table, the research that combined both the pending interest table and Forwarding Information Base remain scarce. This lack of PIT-FIB aware design leads to suboptimal performance in named data networking with highly dynamic and large scale

Correspondence to: Sushil Kumar Bagi, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: sushil.23183963@mygyanvihar.com

environment. Therefore, the central problem addressed in this study is the absence of an intelligent caching strategy that jointly leverages PIT and FIB information to make adaptive, real-time caching decisions. Besides the limited use of PIT-FIB information in caching strategies there is a lack use of model free machine learning to enhance the caching strategy.
The rest of this paper is organised as follows: Section 2 presents the research methodology used for selecting and analysing relevant studies. Section 3 provides a system overview. Section 4 reviews existing literature on caching in NDN. Section 5 compares various caching strategies through qualitative analysis. Section 6 discusses research gaps and limitations. Section 7 highlights current challenges and emerging trends. Section 8 outlines future research directions, and Section 9 concludes the paper by summarising key findings.

II. Research Methodology

This review primarily focuses on research articles published between 2020 and 2025, reflecting the most recent developments in caching strategies for Named Data Networking (NDN) under the broader Information-Centric Networking (ICN) paradigm. Foundational works published between 2010 and 2019 were also included to provide historical context and to illustrate the evolution of caching techniques. The search strategy involved querying multiple digital libraries, including IEEE Xplore, SpringerLink, ScienceDirect, ACM Digital Library, and Google Scholar. Search keywords included combinations of: “Named Data Networking”, “Information-Centric Networking”, “caching strategies”, “content store”, “machine learning”, “reinforcement learning”, “PIT”, and “FIB”. Boolean operators (AND/OR) were used to refine search results, and both title/abstract and full-text searches were performed.

A. *Purpose and Scope:*

The purpose of this review is to provide a broad overview of caching strategies in Named Data Networking (NDN) and their impact on network performance. The review describes the evolution of cache placement and cache replacement techniques, from basic static methods to adaptive and machine learning–based approaches. Its scope covers research works published between 2020 and 2025, while also referring to foundational studies (2010–2019) to give historical context on on-path caching and cache replacement strategies. The review highlights key strategies, simulation tools, and evaluation metrics used in the literature, offering new researchers a clear understanding of developments in NDN caching.

B. *Inclusion Criteria:*

a) Research published in peer-reviewed journals or conferences.
b) Papers focusing on caching strategies within NDN or ICN, including both AI/ML-based and non-AI approaches.
c) Studies providing simulation-based, analytical, or experimental performance evaluation.
d) Articles written in English.

C. *Exclusion Criteria:*

a) Works unrelated to NDN caching (e.g., general web caching not within ICN context).
b) Articles without performance evaluation or technical discussion.
c) Non-peer-reviewed content such as theses preprints without peer review, blogs, or presentations.

The initial search yielded 120 articles. After removing duplicates and applying the inclusion/exclusion criteria, 46 relevant research papers were selected for detailed analysis. These were then categorized by publication year to identify trends and assess the progression of NDN caching research.

III. System overview

A. *Comparison between NDN and Traditional IP Networks*

This NDN architecture provides more efficient data fetch and reduces the reliance on traditional IP-based routing methods; ultimately it improves overall network performance and user experience in terms of delay and throughput [4].Current network is based on TCP/IP protocol suit where network works on end to end connectivity of devises and It starts from application layer to physical layer and vice versa to deliver data through the concept of transmission, logical addressing, forwarding etc. the main objective is that the routers use in the traditional network is not for data caching, it use only for to provide the best route to data packets. So processing of all steps in IP network is time taking and if any data lost during transmission then the data must be retransmitted from original source where NDN is based on the concept of content centric whatever the rule and regulation needed are surrounded towards data. The tradition network and NDN could be distinguish with many parameters like DNS, Message types, packet formation, logical addressing, types of connection etc. [5].

Correspondence to: Sushil Kumar Bagi, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: sushil.23183963@mygyanvihar.com

B. *Working of NDN*

The main architecture of Named Data Networking is based on the concept of content-centric networking, where data is retrieved based on its content name rather than its location. As in Fig. 1, we can easily see that every node in the NDN just responds to the request if the node is not fulfilling the request, it will forward the request. Whenever the consumer requests to network the nearest router will respond if it has the contents otherwise it will send it to the next router to obtain the content If the next router also has no content, then send it to next node, this process will continue until it reaches to producer. At last producer will send the content to the same path from which the interest packet came. It has five components.
1. PIT (Pending Interest Table
2. FIB (Forwarding Information Base)
3. CS (Content Store)
4. Consumer
5. Producer

These components work together to manage data requests, cache content locally for faster access, and ensure that interests are forwarded efficiently throughout the network.
Now to understand the background of NDN we can refer to Fig. 2, where the consumer needs to access a specific piece of content; it sends a request as an interest packet which contains the name of the desired data into the network. The router which has CS (content Store) checks whether the requested content of consumer is available in the cache memory. If it is found, the router retrieves the content from the CS and sends it back to the consumer, significantly reducing latency. If the content is not available in the cache, the interest packet is forwarded through the FIB to locate the producer that holds the requested data then router based on the interest writes the pending interest in the PIT (Pending Interest Table), allowing it to track which interests are still unresolved. This mechanism ensures that once the data is retrieved from the producer, all waiting consumers can receive the content simultaneously, further optimising network efficiency and resource utilisation. After maintaining the PIT, the router forwards the interest based on FIB to the appropriate next hop, ensuring that the request reaches its destination promptly. This process not only enhances data retrieval speed but also minimises unnecessary traffic across the network, contributing to a more streamlined and responsive communication system also discussed [6].

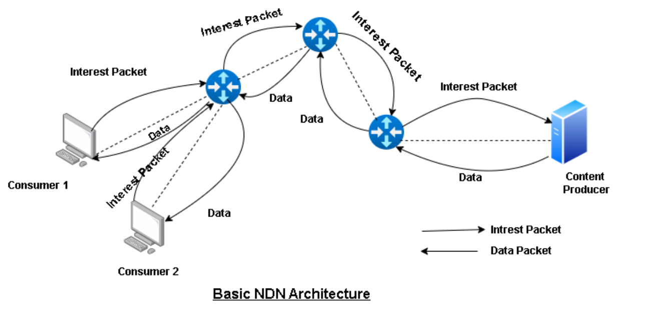

Fig. 1. Architecture of NDN Networking

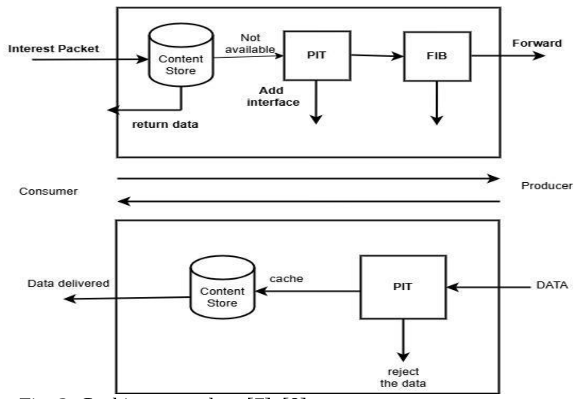

Fig. 2. Caching procedure [7], [8]

C. *NDN Communication Workflow Using PIT, FIB, and Caching*

The NDN communication is based on maintaining two tables i.e. PIT and FIB, the detailed mechanism which is explained in Fig (3) and Fig. (4) Where it shows how NDN initially works before and after caching. Figure 3 shows how to request first time from the consumer to the producer. When first time a consumer wants any data, it firstly generates an interest packet and sends it to the nearest router to process the request. In figure it is shown that there are two consumers 1 and 2 who want to access data content c1 and c2 respectively which are produced by producers 1 and 2. Consumer first and consumer second generate the interest 1 and interest 2 messages respectively both consumers send the request to the nearest router R1. There are totally three routers between the consumer and producer i.e. R1, R2 and R3. At router R1 when request packets i.e. Interest 1 and Interest 2 reach router R1, R1 checks its content store if requested data packets are available then it provides them to the consumer. In Figure (3) the content store is empty then router R1 sends to the next node after updating the PIT i.e. pending interest table where router1 checks whether any earlier request for the same is in pending in PIT if not then it writes the request in the PIT. Here R1 wrote interest 1 and interest 2 in PIT. And according

Correspondence to: Sushil Kumar Bagi, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: sushil.23183963@mygyanvihar.com

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

to the routing information of FIB, router 1 will forward the request to the next node. In FIB of R1, it is mentioned that for interest 1 and interest 2, they go to router 2 and the same course of action router 2 has done. When the request will reach router 3 it sends the appropriate producer for the fulfilment of interest1 and interest2. In fig. (4) Which shows how data packets of content c1 and c2 are delivered to consumer1 and consumer2 respectively. When content c1 and c2 found in producer 1 and producer 2 they send it to the same return path of interest. It is sent to first router R3 and R3 stores the content c1 and c2 in its content store and then it deletes the previous pending request from its own PIT table accordingly and updates the FIB for the content c1 and c2.Then it sent the contents to the next router R2 from which the interest received previously continuously repeat the same action at the router R2 i.e. copy the content in its own content store and delete the pending request from PIT table and update FIB table. In this way, the content c1 and c2 are delivered to the consumer 1 and consumer 2. Now the content is cached at the routers R1, R2 and R3.If the same data request is generated by consumers then it will be fulfilled by router R1.By this procedure we can achieve fast delivery of data packets for the same request. So we can say that caching is the way by which we store the contents of routers between the consumer and producer.

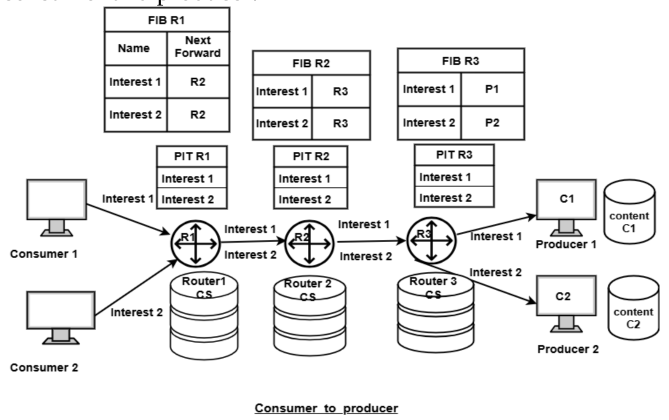

Fig. 3. Consumer to producer [9]

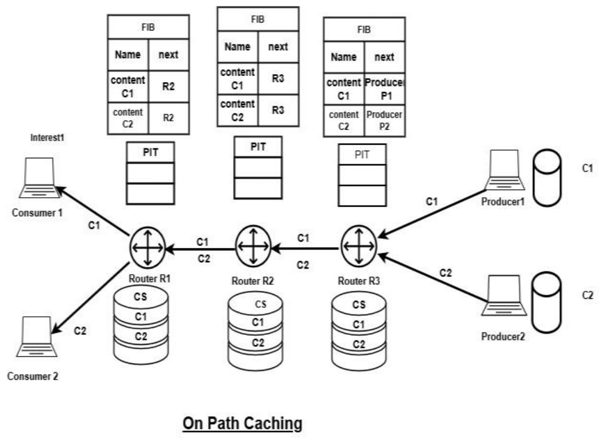

Fig. 4. Producer to consumer [9]

IV. Conclusions

There are various caching strategies which store the data between the consumer and producer. The caching is divided into two parts: cache placement and cache replacement. This paper provides an overview, classification, and research directions, offering a comprehensive literature survey focusing on caching strategies within Named Data Networking (NDN). Caching is generally categorized as on-path caching and off-path caching. Various caching strategies have been developed for NDN, emphasising the shift from traditional IP-based networking to a content-centric approach. In-network caching stores data at intermediate nodes rather than just at the endpoints, improving cache hit rates and minimising retrieval hops. Similarly, content replication enhances data availability by distributing multiple copies across caching nodes, reducing network traffic and improving overall performance. However, the majority of approaches rely on concepts such as node popularity, content popularity, content priority, content diversity, routing, producer mobility, energy consumption, information sharing, learning-based caching, indexing-based caching, and some on the integration of CS and PIT. There are many caching algorithm has been developed and simulated using ndnSIM [8], [10].

A. *Recent Research Trends*

The review paper [11] surveys caching strategies in Named Data Networking (NDN) from 2015 to 2021 and offers a structured map of the literature by publication venue, application domain, modern strategy families, and simulation environments. It contrasts foundational replacement baselines—FIFO, LFU, and LRU—with popularity-aware and learning-based methods, and distinguishes cache placement from replacement, which is helpful for clarifying the decision points in the content lifecycle. On the placement side, the paper summarizes algorithms such as BEACON (betweenness-aware popularity estimation), DLCPP and SAE (deep models for popularity prediction), CPC (cache capacity control), Bloom-filter–based designs, green caching/routing, EHCP (energy-aware hybrid placement), and MAED (Markov approximation for energy–delay trade-offs). The review also notes the emergence of Q-learning and other ML techniques, along with compound popularity models, signalling a shift toward adaptively. Crucially, the paper also touches on cache replacement and enumerates a range of strategies beyond the classical baselines. These include: ANFIS-based replacement (leveraging a false-locality parameter to counter locality-disruption attacks), Universal Caching (UC) (reported to outperform traditional policies across scenarios), Deep Cache (deep learning–based popularity prediction for eviction), LSTM encoder–decoder approaches (time-series modelling of content popularity), Atomic Caching (replacement guided by overlapping name prefixes), On-the-Fly Caching (restricting storage to cache-specific prefixes), BEP (Betweenness and Edge Popularity) (node-centric popularity for eviction), VNDN-oriented policies (frequency-driven replacement for vehicular settings), LFF—Least Fresh First (freshness-centric eviction for IoT), and FDC—Freshness-Driven Caching for vehicular NDN (incorporating content lifetime). By grouping these techniques, the review helps readers see the design space ranging from topology-aware to freshness- and ML-driven policies. That said, the analysis in [5] remains primarily descriptive. It does not provide discrete, metric-based comparisons across a standard testbed or common traces. In particular, it does not evaluate strategies using core NDN metrics such as cache hit ratio, latency, content diversity, hop count, content stretch, link load, content redundancy, inter-domain traffic, energy consumption, or server load. Nor does it present side-by-side benchmarking under identical topologies or workloads, which would enable statistically grounded conclusions about efficiency, scalability, or robustness to popularity shifts. Additionally, integration with PIT/FIB-aware signals in the replacement decision—an area with growing interest—receives limited empirical treatment. Implication and research gap. While this research paper is valuable as a landscape survey and taxonomy, its lack of quantitative, head-to-head evaluation limits prescriptive guidance for practitioners. This gap motivates the present work: a systematic, metrics-driven comparison of classical (FIFO, LFU, LRU) and advanced replacements (ANFIS, UC, Deep Cache, LSTM-based, Atomic, On-the-Fly, BEP, VNDN-specific, LFF, FDC), ideally under controlled topologies (e.g., 2 consumers, 2 producers, 5 routers) and dynamic workloads, with reporting on the full metric suite listed above. Extending this with PIT/FIB-aware decision signals and model-free RL (e.g., Q-learning) would further test adaptivity to non-stationary popularity, providing the empirical evidence missing from the current review.The study in [11] where authors have proposed new system model which maintains two tables IST and IRT and they discusses a cache placement technique based on compound popularity, which combines both content popularity and node popularity. In this approach, content is divided into global popularity and local popularity. By considering both content popularity and node popularity, the caching strategy can make more informed decisions about where to store content. Authors have also discussed modern caching strategies in NDN such as cache capacity-aware CCN, spanning tree heuristic and distributed algorithms, distributed and reconfigurable DL based on SDN, green caching, routing-efficient hybrid content placement, node popularity, content popularity, active edge caching algorithms, etc. In the paper [12] authors proposed the CAL i.e. Cache Aging with Learning method for IoT network which is eight step processes to manage and process data freshness with using specific formula. It uses prediction mechanism which is based on Nonlinear Autoregressive (NAR) neural network so that it can increase the cache hit ratio. This neural network model is based on past traffic pattern and does not account for real time network conditions. This limitation reduces its adaptability towards sudden changes in content popularity. In the paper [13], authors proposed a machine learning approach using Apriori algorithm to predicts and cache frequently accessed data so that cache hit ratio will be increased. Authors are used Apriori algorithm which is supervised learning algorithm to find the association rules and based on that it predicts the next requested data, means it uses prior knowledge of frequent item sets. In this paper authors shown the performance of recommended machine learning algorithm with LRU and prove that apriori based algorithm perform high cache hit ratio. Still this algorithm was based on real time data sets of traffic for analysis with single metrics i.e. cache hit rate. The work does not leverage based on more performance metrics like network traffic load, server hit rate with considering current data analysis i.e. real time NDN specific information. In the paper [14] the authors proposed an intelligent caching method for vehicular networks using deep transfer learning. The approach introduced a time-varying mechanism to predict content popularity, supported by a customized hybrid neural network model. The model was trained in three stages to identify and replace less popular content with more popular content in the cache. While the method demonstrated its capability to improve content delivery by predicting popularity trends, the evaluation was conducted entirely in MATLAB rather than in a network simulation environment such as ndnSIM. This limits the assessment of the method’s performance under realistic NDN conditions with real-time network dynamics.

The paper [15] introduces the CRPM (Cache Replacement Policy based on Multi-factors) for NDN, which calculates a "cache value" for each content item using a weighted combination of parameters such as popularity, acquisition cost, energy consumption, and freshness. Content with the lowest value is evicted. The authors validated CRPM on a simple linear topology, leaving a clear path for future research. To fully assess CRPM's potential, researchers should test it on diverse NDN topologies like fat-tree or mesh. Furthermore, a more comprehensive evaluation is needed using NDN-specific metrics like average interest satisfaction latency, number of hops saved, and the policy's impact on network congestion. This would provide a more robust understanding of CRPM's performance and applicability in real-world NDN environments. For instance, paper [16] introduced a probability-based eviction mechanism that considers content popularity distribution. The scheme assigns each item a probability of being retained or removed, balancing storage between popular and less popular content. Analytical models were developed to compute average miss probabilities across multi-level cache networks. The approach reduces redundancy compared to naive caching but relies on accurate and relatively stable popularity estimation. While theoretically sound, the lack of adaptive mechanisms and real-world validation limits its scalability in dynamic NDN environments. Where the paper [17] in this paper authors proposed a *Popularity and Gain Based Caching Scheme (PGBCS)* for ICNs that considers both the popularity of content chunks and their caching gain to guide caching decisions. The authors emphasised the dynamic nature of content value, suggesting that frequently requested data should be retained longer. Simulation results showed improvements in cache hit ratio, user access delay, and service quality compared to conventional strategies. However, the approach still requires validation in diverse real-world network settings. The paper [18] proposed a model for video content caching that prioritises items likely to be accessed again, improving efficiency. Beyond single-metric policies, multi-factor approaches incorporate additional parameters, such as content size, request frequency, and user distance. However, the approach mainly considers popularity and a few network parameters, limiting its adaptability to highly dynamic network conditions and diverse content types. The study [9] provides an extensive review of caching techniques in NDN. It first highlights the limitations of traditional location-based Internet architectures, such as network congestion and high latency, emphasising the need for efficient caching mechanisms in NDN to improve data retrieval. The study discusses various caching techniques, particularly popularity-based caching, which prioritises repeatedly requested data content to increase cache hit ratios and maximise network performance. A comparative analysis is conducted on multiple caching strategies, including Compound Popular Content Caching Strategy (CPCCS), Max-Gain In-network Caching (MAGIC), and Hop-based Probabilistic Caching (HPC), evaluating their efficiency in reducing content redundancy, improving cache hit ratios, and alleviating network congestion. Furthermore, the paper explores the role of caching strategies in new technologies such as IoT, Fog Computing, Edge Computing, and 5G networks, emphasising their adaptability in dynamic environments. This study gives a presentation on cache replacement techniques that are used to replace the content from the cache when there is no space in memory. The cache replacement algorithm is classified into eight different types of techniques. These are (i) static (ii) space security (iii)content update (iv) centralised (v) Energy efficient (vi) weighted (vii)adaptive (viii)Dynamic Popularity. Some of the very well-known cache replacement static techniques are RR, LRU, LFU, and FIFO. These techniques are simple but not so effective. The study in [19] emphasises dynamic content popularity-based caching, which caching of popular content enhances the consumer retrieval rate in NDN networks where popularity is based on request cycles and the cache threshold is maintained according to every node's cache. Popularity changes over time. There are various proposed algorithms which has been developed so far to enhance the popularity For example the paper [20] proposed MPC, in which the node caches content whose popularity has gone beyond the threshold value. In the paper [21] proposed a cache method which was based on content popularity and router level, in this approach those routers which are close to the consumer can cache high popularity content. Accordingly, the authors proposed two algorithms out of which one is for cache placement and another for cache replacement. In cache placement, the proposed policy is known as dynamic popularity cache-placement (DPCP) which calculates the data content popularity based on the number of content requests in the current and past cycles. In cache replacement the proposed policy is based on replacement value (RVCS), this value will be calculated with the help of last popularity, request time and transmission cost. Content will be replaced if and only if the replacement value is low. The researchers in [18] had applied Graph Neural Network (GNN) to node level classification problem. They found that GNN is efficient for node-level, edge-level and graph-level prediction. In this study authors shown that there are various techniques had been developed so far based on content popularity by using neural network like deep learning-based content popularity prediction (DLCPP) to perform cache decisions in the SDN based ICN same as stimulate neural network (SNN) is used to make caching decision. The GNN-based cache replacement policy in NDN was the first proposed method in NDN caching. GNN is used to learn spatial dependency to collect the traffic data. It demonstrates how a GNN-based caching method can significantly reduce access latency by caching popular information close to the consumer. But still, there is some limitation in GNN like the statistical model used to generate users' content preferences may rely on assumptions that do not hold true in all contexts. If user behaviour significantly deviates from these assumptions, the effectiveness of the caching strategy could be compromised; leading to lower cache hit ratios than expected. There are so many assumptions that were made during the research for example users' content preferences, static user behaviour, network condition, cache space limitation and user request history If any one of them deviates then performance will go down.

V. Qualitative Comparative Analysis of Existing Caching Strategies

Caching in NDN is a pivotal mechanism that enhances data retrieval efficiency by storing content at various network nodes. This approach not only reduces latency but also improves content availability, making it a promising alternative to traditional IP-based networks. Several innovative caching methodologies have been proposed to optimise performance in NDN environments. Essentially, there are two kinds of caching named is (i) On path caching and (ii) off-path caching as fig (5) shown. But some data caching techniques comes under both categories depending on its implementation. In this paper, we are only concerned with on-path caching. In on-path caching in Named Data Networking involves storing content at routers along the data path from producer to consumer, enhancing response times. However, it lacks the cooperative advantages of neighbourhood caching techniques, which improve performance through collaboration among neighbouring routers in content retrieval and caching decisions.

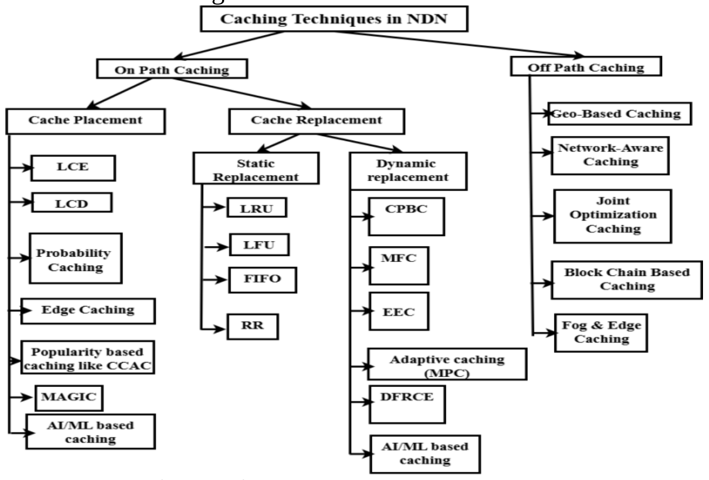

Fig. 5. Classification of caching

A. *Cache Placement and Replacement Strategies in NDN*

Caching is a fundamental component of the Named Data Networking (NDN) architecture, as it directly influences how and where data is stored within the network to enhance efficiency and reduce latency. Proper cache placement ensures that frequently accessed content is readily available, thereby minimizing redundant requests to remote servers and improving overall network performance. Cache replacement strategies further determine which data should be retained or evicted from the content store. As illustrated in Figure 5, static replacement strategies rely on pre-computed decisions; however, due to their inability to adapt to varying network dynamics, their effectiveness is limited in highly dynamic environments. Conversely, dynamic replacement strategies are designed to adapt to changing traffic patterns and content popularity, enabling more intelligent eviction decisions and enhancing data retrieval efficiency. To provide a comparative understanding, Table 1 summarizes A. cache placement techniques, B. static replacement policies, and C. dynamic replacement policies, along with their respective advantages and disadvantages.

Table 1. Cache Strategies

A	**Cache Placement**
1.	Compound popularity based caching [11][22]
Mechanism	Based on compound popularity, which combines both content popularity and node popularity
Parameters	Metrics: CHR, Latency, Link Load Topology:Tiscali-3257, Tree topology simulator: Icarus
Advantages	(i) A proposed technique outperforms the four base line caching strategies across both topologies and all values of S. (ii) It achieves 41.2% CHR at S = 0.25 in Tiscali-3257 topology, which is 5.1% higher than the second-best (LCD at 36.1%). (iii) It provides 2.4% improvement over LCD when S = 0.05. (iv) In tree topology, the proposed scheme achieves an average CHR improvement of up to 3.1% compared to LCD and CL4M. (v) In terms of latency the proposed technique achieves the lowest latency in both Tiscali-3257 (64.65 ms) and tree topology (66.2 ms). (vi) In Tiscali-3257, latency is 4.8% lower than LCD, 9.9% lower than CL4M, 13.1% lower than ProbCache, and 12.1% lower than LCE. (vii) In tree topology, latency is 3.3% lower than LCD, 3.3% lower than CL4M, 8.3% lower than ProbCache, and 12.2% lower than LCE. (viii) It effectively utilizes edge caching (LPC and GPC), reducing consumer response time. (ix) Proposed scheme effectively reduces link load when cache size is large.
Disadvantages	(i) Marginal Gains in Tree Topology for example CL4M performs nearly as well as LCD, with CHR values close to 36–37%, which narrows the relative improvement margin of the proposed scheme. (ii) Consistently achieves higher cache hit ratio than other strategies under the same cache sizes. (iii) In terms of latency CL4M and LCD have nearly equal latency in tree topology, narrowing the proposed scheme’s relative advantage. (iv) Performance gap is smaller in tree topology compared to Tiscali-3257. (v) All strategies benefit from larger cache sizes, reducing the relative improvement margin.
Advantages	(vi) Latency performance is nearly the same in both topologies under α = 0.8, S = 0.15, limiting topology-specific insights. (vii) In tree topology, link load = 348.26 bytes, which is slightly worse (0.06% higher) than LCD (346.32 bytes). (viii) Overall improvement is marginal with small cache sizes.
2	Hop-Based Probabilistic Caching (HPC)[9], [10], [23]
Mechanism	Utilizes two key factors—CacheWeighty and CacheWeightMRT—to optimize caching. CacheWeighty determines the probability of caching content based on the number of hops, while CacheWeightMRT sets the caching duration using the mean residence time (MRT) of the content. This probabilistic approach aims to reduce redundancy and manage caching along the consumer–producer path.
Metrics:	Content Diversity, CHR, Content Redundancy, Stretch Ratio Topology: Linear Path Topology simulator**:** custom-built simulator[10], SocialCCNSim[9],[23]
Advantage	(i) Probability-based caching technique. (ii) achieves a low stretch ratio. (iii) Provides a better cache hit ratio. (iv) Performs better than many traditional caching strategies.

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Disadvantages	(i) Does not prioritize less popular content. (ii) May lead to high redundancy. (iii) Increased memory consumption (iv) Low content diversity.
3)	WAVE Popularity-based caching strategy [9], [24]
Mechanism	A popularity-based caching technique that distributes data chunks across the network according to their content popularity
Parameters	Metrics: Average Hop Count, Link Stress, Inter-ISP Traffic Reduction, Cache Hit Ratio, Cache Replacement Count, Caching Efficiency, Relative Hop Count and Number of Chunks. Topology: GT-ITM–generated hierarchical topology simulator: discrete event-driven simulator[24]
Advantages	(i) WAVE achieves the shortest average hop count, placing popular content closer to users and reducing retrieval delay. (ii) It reduces cache diversity because it only consider on popular content. WAVE caches early (low-index) chunks 8–15% closer to users. (iii) It reduces link stress by distributing content across multiple C-routers. (iv) WAVE caches more popular files locally, reducing inter-ISP traffic and lowering external server dependence. (v) Increasing file chunk count reduces unnecessary caching overhead. (vi) Reduces average cache management cost. (vii) It achieves a high cache hit ratio because each cached chunk is reused 23.5 times on average, at least 16× higher than competing schemes. (viii) lowers delay, shortens stretch ratio, and introduces fewer repetitions of similar content (less redundancy compared to HPC and CACC).
Disadvantages	(i) Popularity-blind caching which achieves extremely poor efficiency (0.09 times per cached chunk). (ii) In-network caching like WAVE still requires at least one inter-ISP fetch for new content. (iii) It Consumes high memory and bandwidth, and may result in a lower cache hit ratio under certain conditions.
4	Most Popular Cache(MPC)[9], [20]
Mechanis	Each node maintains a popularity table that tracks content names, popularity counts, and a threshold value. When the popularity count meets the threshold, the content is labeled as popular, prompting the router to recommend caching it to neighboring routers.
Parameters	Metrics **:** CHR, Stretch, Ratio of Cached Elements, Diversity Topology**:** Tree, Abilene, Tiger2, Geant, DTelekom, Level3 Simulator**:** ccnSim, a chunk-level CCN simulator, developed in C++ over the Omnet++ framework[20]
Advantages	(i) Increases cache hit ratio by outperforming the default strategy. It achieves >85% hit ratio. (ii) Reduced Storage Overhead. (iii) It ensures efficient resource utilization. (iv) reduces network traffic. performs better than WAVE and CACC. (v) It improves overall network resource consumption.
Disadvantages	(i) Lower Diversity; the MPC diversity: 3%–18%, much lower than CCN (28%–35%). (ii)Introduces communication overhead because it requires Popularity Table, Popularity Threshold, and Reset Value tuning, increasing complexity. (iii) Performance heavily depends on threshold selection. (iv) caching is biased toward nodes neighboring producers. (v) Topology Sensitivity.
5	Leave Copy Every Where (LCE) [25]
Mechanism	This is the simplest cache placement policy in Named Data Networking (NDN), where a data packet is cached at every node along the forwarding path between the consumer and the producer.
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM) Topology: Binary Tree Network Topology Simulator: Icarus

Advantages	(i) High Availability: Ensures content is highly available since it is cached at all nodes along the path, reducing latency for popular content. (ii) Reduced Traffic Load: By caching at multiple routers, requests can be served from the nearest cached node, lowering traffic overhead [15]. (iii) Performance Evidence: LCE reduced RTT by 82% and outperformed LCD in comparative evaluations [26].
Disadvantages	(i) Redundancy: Excessive duplication wastes cache space and bandwidth. (ii) Cache Management Complexity: Maintaining validity of many cached copies is challenging [15]. (iii) Low Content Diversity: Redundant caching reduces diversity and leads to inefficient cache use [11]
6	Leave Copy Down (LCD) [25]
Mechanis	After a cache hit, the requested data is cached only on the downstream router (one hop closer to the consumer) rather than on every node along the path.
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM) Topology: Binary Tree Netwotk Topology Simulator: Icarus
Advantages	(i) Minimum Redundancy: LCD avoids excessive duplication by placing a copy only at one hop downstream, making cache use more efficient. (ii) Lower Replacement Errors: Conservative caching reduces the rate of unnecessary replacements. (iii) Performance Evidence: LCD achieved up to 59.15% reduction in RTT in experimental analysis.
Disadvantages	(i) Slow Content Diffusion: Popular content requires multiple requests to propagate toward edge routers, delaying availability. (ii) Path Redundancy: While minimizing redundancy per node, it can still cause duplication along the consumer–producer path . (iii) Implementation Complexity: Requires careful tracking of requests and cache states, increasing overhead.
7	Bernoulli random caching [23],[26],[27]
Mech	Content is cached randomly at each node with a given traversal probability (p), meaning each node independently decides whether to store the content.
Parameters	Metrics: CHR, Latency / Response Time, Bandwidth Savings, Cache Utilization, Computational Overhead, Fairness / Adaptability Simulator: Icarus/own custom simulator based on c++
Advantages	(i) Prevents caching rarely requested content, which can increase CHR compared to naïve "always cache.". (ii) Simple and lightweight — O(1) per decision (just generate a Bernoulli random variable). (iii) Works well as an admission policy combined with LRU/LFU for eviction, (iv) Load Distribution: Reduces server load by spreading content across multiple cache nodes, lowering repeated retrievals from the producer. (v) Adaptive Hit Ratio: Improves cache hit ratio by introducing a probabilistic storage approach that adapts to varying request patterns.
Disadvantages	(i) Randomized behavior, Suboptimal Decisions: If probability (p) is not well-calibrated, unpopular content may be cached unnecessarily. (ii) Lower Predictability: Randomness can cause inconsistent caching performance compared to popularity-aware strategies. (iii) Not adaptive to changing content popularity (static probability). (iv) Lower hit ratio compared to smarter policies (like LFU, ARC, or ML-based strategies) if not tuned. (v) Parameter sensitivity : performance highly depends on choosing the right p.
8	Random choice caching[25], [28]
Mechanism	Caches the content item at only one randomly selected node.
Parameters	Metrics: Energy Savings, Caching Benefit , Delivery Success Rate Topology: Random/uniform placement in a 2D area Simulator: Icarus

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Advantages	(i) Simple and easy to implement: Simple to implement and moderately less affected by node energy compared to LCD. Or we can say that Random Choice caching offers slightly better energy-saving stability than LCD, but as both Pc and the number of users increase, its energy-saving effect approaches zero. (ii) Random choice caching has high caching benefit than LCD.
Disadvantages	(i) Inconsistent performance; does not guarantee a high cache hit ratio since decisions are purely random. (ii) Random Choice caching suffers from poor energy-saving efficiency, as its effect drops to nearly zero when both Pc and the number of users increase.
9	cache placement strategy based on energy consumption optimization (ESCPS)[28]
Mechanism	It first calculating the energy consumption of content delivery from different nodes to the user, then formulating a reward function that balances delivery energy and cache-switching costs. Using optimal stopping theory, the algorithm selects the best cache node to maximize expected energy savings
Parameters	Metrics:Energy savings,Caching benefit,Delivery success rate Topology:simplified, abstract model simulator: MATLAB
Advantages	(i) Achieves the highest energy savings across all scenarios (users, content size, node distribution). (ii) Considers user perspective when selecting cache nodes, leading to better placement decisions. (iii) Energy savings increase almost linearly with content size due to optimized income function. (iv) Stable performance even when node replacement cost (Pc) is high or number of users increases. (v) Provides the best caching benefit (energy saved per unit content) incomparision to LCD and Random. (vi) Strong adaptability to dynamic network environments and hotspot traffic. (vii) High delivery success rate: probability of finding optimal cache placement remains high even with increasing node range
Disadvantages	(i) Higher computational complexity compared to Random or LCD. (ii) Requires calculating reward/income functions for optimization. (iii) More difficult to implement in large-scale or real-time systems due to overhead. (iv) May need additional resources (processing and storage) to maintain efficiency. (v) Delivery success rate slightly decreases as Pc increases (extra energy cost per observation).
10	Centrality-based caching/ Centrality-Measures Based Algorithm[25], [29]
Mechanism	Data contents are cached only once along the path, based on the value of betweenness centrality.
Parameters	Metrics: Server Hit Reduction Ratio, Hop Reduction Ratio, Topology: random network topology, BRITE network topology using 500 nodes Simulator: ndnSIM [29]
Advantages	(i) CMBA outperforms Random, UC, and Betweenness because it intelligently selects cache routers using multiple centrality measures and cache capacity, while Random causes more cache misses, and UC/Betweenness often tie due to equal centrality values. (ii) CMBA maintains high performance in terms of Hop Reduction Ratio (HRR) and Server Hit Reduction Ratio (SHRR) compared to Random, UC, and Betweenness schemes. (iii) Suitable for highly dynamic networks, making it well aligned with NDN.
Disadvantages	(i) Complex, as it requires estimating or learning the node betweenness centrality value, which is challenging. In another word we say that Computational Complexity – Requires calculating multiple centrality measures (degree, closeness, reachability, betweenness), which is computationally expensive for large dynamic networks. (ii) Overhead in Dynamic Topologies – In dynamic scenarios, centrality values need to be recomputed frequently, adding processing and communication overhead.
11	Hash-routing [20] [30]

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Mechanis	Uses a hash function to map the content identifier to a specific caching node, and forwards the request to that node.
Parameters	Metrics: CacheHitRatio, Content popularity skewness, Average Link Load, Topology: GEANT, GARR, WIDE, Tiscali Simulator:Icarus2
Advantages	(i) Off-path hash-routing schemes consistently outperform on-path caching strategies, achieving better utilization of distributed caches. (ii) Among off-path approaches, symmetric hash-routing achieves the highest hit ratio across all values of Zipf α, (iii) Robust Across Cache Sizes. (iv) Eliminates data cache redundancy. (v) Hash routing mechanism increase the cache hit by 31 % in comparison to on-path caching.
Disadvantages	(i) Increased Link Load – Many off-path HR schemes (e.g., symmetric HR, HR multicast, HR hybrid SM) cause significant additional link traffic compared to on-path caching. (ii) Potential Latency Overhead – Because content may be cached at an off-path node, retrieval can require detours, increasing response time compared to simple on-path caching. (iii) Topology Sensitivity – Effectiveness depends on network topology (e.g., GEANT vs. BRITE), so performance gains are not uniform across all networks. (iv) Diminishing Advantage with Large Caches – As the cache-to-population ratio increases, the performance gap between on-path and off-path caching (including HR) shrinks significantly. Uneven Traffic Distribution (v) HR Asymmetric can suffer from imbalanced traffic, since some cache nodes may see limited traffic, reducing their usefulness and lowering overall cache efficiency.
12	Probabilistic cache/ProbeCache [25][10]
Mechanism	This modern strategy caches content based on both the availability of cache capacity at each node and its proximity to the consumer. Nodes closer to consumers have a higher probability of caching the content, thereby optimizing cache utilization and reducing redundancy.
Parameters	Metrics: Server Hits,Hop Reduction Topology: 6-level binary tree topology, heterogeneous cache deployments, scale-free topologies Simulator: custom-built simulator
Advantages	(i) Optimized Utilization: Balances cache capacity and demand, leading to efficient resource use. (ii) Reduced Server Hits & Hop Count: Improves hop reduction ratio and decreases reliance on producers. (iii) Adaptability: Works effectively in both homogeneous and heterogeneous cache size environments. (iv) Efficient Cache Resource Utilization. (v) Lower Cache Evictions. (vi) Scales to Heterogeneous Caches.
Disadvantages	(i) Implementation Complexity: Requires careful calibration to avoid uneven cache distribution. (ii) Dependency on Parameters: Relies on Time Since Inception (TSI) and Time Since Birth (TSB) values, which may increase overhead and complexity. (iii)Hop reduction improvement is relatively small compared to CE2 and LCD. (iv)Topology Dependency. Sensitivity to Cache Size and Content Popularity. (v) Not Always Optimal in Core Caching.
13	Mobility aware data caching (MAEDC)[31], [32]
Mechanism	Focuses on optimizing energy consumption while considering end-to-end delay in caching decisions. It aims to achieve near-optimal cache allocation performance across different network topologies.
Parameters	Metrics: Energy Consumption, Caching Hardware Technology,The Mean Hop Count Topology: TREE Topology, NSF Topology, EON Topology Simulator: MATLAB

Advantages	(i) Optimizes energy consumption and considers end-to-end delay, achieving near-optimal cache allocation performance. (ii) Reduces energy consumption by 19.65% (without delay) and 12.92% (with delay) compared to other strategies. (iii) Reduces mean hop-counts by 4.46% (without delay) and 0.96% (with delay), supporting delay-aware caching. (iv) Achieves near-optimal performance close to ILP solution (minor energy increase 7.16–11.42% (v) Popularity-aware caching using Zipf-distributed content requests. (vi) Supports distributed and parallel processing for faster convergence. Considers multiple energy components in CCN (interest packet transmission, security overhead).
Disadvantages	(i) The emphasis on energy efficiency may lead to trade-offs in cache hit ratio or content availability under certain conditions. (ii) Performance sensitive to caching hardware; high-power hardware like TCAM increases energy drastically. (iii) Computationally intensive due to ILP formulation and heuristic optimization, especially for large networks. (iv) Assumes single content provider; may not generalize well to multi-source networks. (v) Needs accurate knowledge of content popularity and network metrics for effective caching. (vi) Maximum acceptable delays are predefined; less adaptable to highly dynamic delay requirements. (vii) Energy-delay trade-off: strategies like CEE achieve minimal delay but at higher energy cost.
14	Randomly Copy One (RCO) [31][33]
Mechanis	Caches content randomly at one of the routers along the forwarding path between consumer and producer, instead of every node.
Parameters	Metrics: Coverage Relative Delay Latency Analysis, Relative Number of Replacement Topology:Power-law topology Simulator: Built own custom event-driven simulator
Advantages	(i) Reduced Redundancy: Compared to LCE, RCO minimizes duplication, leading to more efficient cache resource utilization. (ii) Balanced Availability: While not as aggressive as LCE, RCO still ensures that cached copies are distributed along the path, providing reasonable availability. (iii) Reduces redundancy (iv) fewer replacements (v) better latency under cache saturation (vi) efficient resource use (<1% nodes needed for near-optimal performance).
Disadvantages	(i) Suboptimal Placement Risk: Random selection can result in caching at less effective nodes, potentially increasing retrieval latency. (ii) Unpredictable Performance: The lack of deterministic placement may reduce performance consistency compared to popularity- or policy-based strategies. (iii) Higher latency at early stages. (iv) Randomness may place cache sub-optimally; lower hit probability in some cases (v) depends on popularity and cache size.
15	Cache Capacity Aware Cache (CCAC)[9], [31]
Mechanism	Cache capacity-aware caching is a strategy that optimizes the use of available cache space in a network by considering the capacity of the cache when making decisions about what content to store. It work on three stages to find the value of threshold i.e. CCVth value which represent threshold value by this value we can decide whether the content is high popular or low popular
Parameters	Metrics: Content Diversity, CHR, Content Redundancy, Stretch Ratio Topology: Linear Path Topology Simulator: SocialCCNSim [9]
Advantages	(i) Less stretch ratio i.e. less distance between consumer and cached data. (ii) It has higher diversity ratio than HPC. (iii) CCAC has high cache hit ratio than WAVE. (iv) CCAC has low Stretch Ratio than WAVE, MAGIC
Disadvantages	(i) No recognition for low popular content, Highly redundancy i.e. CCAC has high content redundancy than WAVE, MPC, DFGPC, MAGIC and CPCCS. (ii) High memory consumption (iii) Less diversity ratio. (iv) CCAC has high Stretch Ratio than HPC,DFGPC and CPCCS

16	Content Caching Strategy for NDN with Skip (CCndnS)[34]
Mechanism	Breaks cache dependencies by distributing file segments along the subscriber–producer path. Each cache independently handles its own queries, reducing unnecessary cache checks and latency.
Parameters	Metrics: network hit probability, Router hit Probability (edge router), Average hop distance, CS hit probability for an edge router, Edge Router Cache Size, CS Miss Probability, Skip Error Probability Topology: Abilene topology Simulator: purpose-built simulator
Advantages	(i) It has higher Router hit Probability when s=3 or 5. (ii) Network probability quickly exceeds when CS size increases. (iii) For specific zipf (α=2.5) distribution the average hop distance is minimum. (iv) It specify that Targets specific routers likely to hold a chunk, reducing router miss probability (v) Balances traffic between edge and core routers, avoiding edge-dominated caching (vi) Reduces memory latency and postpones router saturation (vii) Eliminates the filtering effect so that core caches remain effective even when edge caches grow (viii) Reduces redundancy without requiring extra coordination or control messages (ix) Provides a simple and accurate analytical model for predicting hit ratios and hop count (x) Allows use of advanced eviction policies like SLRU to reduce cache pollution
Disadvantages	(i) Overall network hit probability remains unaffected despite reduced router misses (ii) Skip errors can occur when Interests bypass caches that actually hold the data (iii) Requires careful tuning of parameters such as segment count (S) and hop bound (H); larger S reduces skip errors but increases hop distance (iv) More complex than default en-route caching (v) Cannot fully eliminate redundancy, which is partly determined by routing (vi) Performance depends on correct parameter estimation and may require dynamic adjustment
17	pre-caching strategy based on the relevance of requested content (PCSRC)[35]
Mechanis	Adopts a sliding window mechanism, using a router ID list in the Interest packet and LACC in the Data packet. After caching the content, it assigns a sojourn time.
Parameters	Metrics: CHR, Average Request Hop (ARH), Server Traffic Ratio (STR), Average Request Delay (ARD) Topology: 31 routing nodes • Clients at leaf nodes only • 10,000 contents, each divided into 10 chunks (10 MB each) • Cache capacity: 1000 MB per node Simulator: ndnSIM
Advantages	(i) Higher Cache Hit Ratio (CHR) than CEE and ProbCache (ii) Lower Average Request Hop (ARH) Reduced Server Traffic Ratio (STR) (iii) Lower Average Request Delay (ARD) (iv) Adaptive to request popularity using weight parameter γ
Disadvantage	(i) Performance sensitive to parameter tuning (γ, α). Higher cache management complexity. (ii) Requires sufficient cache space for pre-caching (iii) Assumes sufficient available bandwidth; results were demonstrated only on limited network topologies.
18	TOPSIS(Technique for Order Preference by Similarity to Ideal Solution) and EW(Entropy Weighting)-based caching.[36]

Mechanis	Combines the Entropy Weighting (EW) method, which assigns weights to each index based on information uncertainty, with TOPSIS, a widely used comprehensive evaluation method.
Parameters	Metrics: CHR, Latency, Link Load Topology: Tiscali-3257 topology Simulator: Icarus
Advantages	(i) CHR increases with cache capacity (S). (ii) Improves cache hit rate by considering content popularity at each node. (iii) Reduces average response hops, improving network efficiency. (iv) TOPSIS is highly flexible, as it imposes no strict restrictions on data distribution, sample size, or the number of indexes. The combination of EW and TOPSIS enables effective resource utilization and optimal node selection. (v) Reduces average latency compared to CBCP, LCD, CL4M, LCE, and ProbCache. Higher cache capacity (S) reduces latency as more content objects are cached. (vi) Lower latency is maintained across varying Zipf parameter α, adapting to content popularity (vii) Reduces average link load compared to LCD, LCE, CL4M, and ProbCache, while remaining comparable to CBCP.
Disadvantages	(i) Involves significant computational overhead and added complexity in implementation. (ii) Requires additional computational overhead to track per-node content popularity. (iii) Increases memory usage compared to simpler caching schemes. Performance depends on Zipf distribution; low α reduces benefits. (iv) Higher complexity than simple schemes like LCE or ProbCache, affecting scalability. Requires computation of HOP index for cache node selection, adding processing overhead. (v) CHR Performance improvement depends on cache size; smaller caches reduce benefits. (vi) Latency performance gain is smaller when Zipf parameter α is low, as content popularity is more uniform. (vii) Link load is slightly higher than CBCP in some scenarios due to lack of consideration of global content popularity. (viii) Link load performance is not optimal when cache capacity is increased from 0.1 to 0.25.
19	Content Caching Strategy for NDN with Skip (CCndnS)[34]
Mechanis	Breaks cache dependencies by distributing file segments along the subscriber–producer path. Each cache independently handles its own queries, reducing unnecessary cache checks and latency.
Parameters	Metrics: network hit probability,Router hit Probability (edge router), Average hop distance,CS hit probability for an edge router,Edge Router Cache Size, CS Miss Probability,Skip Error Probability Topology:Abilene topology Simulator: purpose-built simulator

Advantages	(i) It has higher Router hit Probability when s=3 or 5. (ii) Network probability quickly exceeds when CS size increases. (iii) For specific zipf (α=2.5) distribution the average hop distance is minimum. (iv) It specify that Targets specific routers likely to hold a chunk, reducing router miss probability (v) Balances traffic between edge and core routers, avoiding edge-dominated caching (vi) Reduces memory latency and postpones router saturation (vii) Eliminates the filtering effect so that core caches remain effective even when edge caches grow (viii) Reduces redundancy without requiring extra coordination or control messages (ix) Provides a simple and accurate analytical model for predicting hit ratios and hop count (x) Allows use of advanced eviction policies like SLRU to reduce cache pollution
Disadvantages	(i) Overall network hit probability remains unaffected despite reduced router misses (ii) Skip errors can occur when Interests bypass caches that actually hold the data (iii) Requires careful tuning of parameters such as segment count (S) and hop bound (H); larger S reduces skip errors but increases hop distance (iv) More complex than default en-route caching (v) Cannot fully eliminate redundancy, which is partly determined by routing (vi) Performance depends on correct parameter estimation and may require dynamic adjustment
B.	**Static Cache Replacement**
20)	RR(Random Replacement)[25], [37]
Mechanis	Randomly selects a cached content for eviction when the cache is full.
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM), Server Load , Round-Trip Hop Distance, Cache Hit Rate, Instantaneous Behavior Topology: Binary Tree Netwotk Topology simulator:Icarus [25]

Advantages	Always + RR Advantages: (i) Balanced content distribution across network (ii) Low server load in many cases (iii) Simple implementation (random replacement) (iv) Fast convergence to stable state
Advantage	Prob(p) + RR Advantages: (i) Similar to Always+RR (no negative impact) (ii) Fast convergence & simple (iii) Stable steady state
Disadvanta	Always + RR Disadvantages: (i) Hit rate generally lower than LFU (ii) Performs worse than Prob(p)+LRU in some scenarios (iii) Randomness may evict useful content
Disadvantage	Prob(p) + RR Disadvantages: (i) No real benefit from probabilistic insertion (results nearly identical to Always+RR) (ii) Still weaker hit rate than LFU or Prob(p)+LRU
21)	Least Recently Used (LRU)[25], [37]
Mechanism	Evicts the least recently accessed content from the cache.
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM) Topology: Binary Tree Netwotk Topology Simulator: ndnSIM[37]
Advantages	Simple LRU Advantages: (i) Simple and easy to implement. (ii) Adaptive to recent requests, good for dynamic workloads.
	Always + LRU Advantages: (i) Fast initial response (caches everything immediately) (ii) Simple to implement

	Prob(p) + LRU Advantages (i) Reduces duplicates (ii) better server load reduction (iii) Higher cache hit rate across node levels (iv) Better round-trip hop distance than Always+LRU (v) Lower memory waste
Disadvantages	Simple LRU Disadvantages: (i) LRU Suffers from performance degradation when cache overflows (blind replacement). (ii) Does not consider content popularity. (iii) may retain less relevant content in dynamic environments. (iv) can evict items that later regain popularity, leading to suboptimal cache performance. (v) It may produce higher miss rates compared to policies like LRFU, especially under high request loads. (vi) Results in higher stretch ratio than ARC performance degrades under the Independent Reference Model (IRM).
Disadvantages	Always + LRU Disadvantages: (i) High server load due to redundant replicas (ii) Poor cache hit rate beyond first-hop routers (iii) Worst round-trip hop distance in cascading topology (iv) Converges fast but to a low-performance steady state
Disadvanta	Prob(p) + LRU Disadvantages (i) Slower convergence (long initial warm-up) (ii) Sensitive to p value (too small p delays caching)
22)	Least Frequently Used (LFU)[25], [37]

Mechanis	Evicts the least frequently item from memory
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM) Topology: Binary Tree Netwotk Topology Simulator: Icarus/ ndnSIM
Advantages	Simple LFU Advantages: (i) LFU has high wall clock time w.r.t. content Catalogue size. (ii) Simple and easy to implement effective for stable and predictable access patterns. (iii) retains frequently accessed items, ensuring popular content remains available. (iv) minimizes cache misses for popular items; resource-efficient since it does not require extensive recency tracking, resulting in lower overhead (v) performs optimally under IRM compared to LRU.
Advantages	Always + LFU (i) Best cache hit rate overall (ii) Strong server load reduction (iii) Shorter round-trip hop distance (iv) Stable steady state performance

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Advantages	Prob(p) + LFU (i) Can reduce unnecessary caching under stable patterns
Disadvantages	Simple LFU Disadvantages: (i) Struggles to adapt to changing content popularity. (ii) prone to cache pollution when historically popular but currently irrelevant items occupy space. (iii) relatively complex to implement. (iv) exhibits a higher stretch ratio than ARC; computationally expensive. (v) unsuitable for large caches due to increased complexity with cache size. (vi) LFU policy incurs O(n) replacement cost, leading to performance degradation at large cache sizes compared to LRU, FIFO, and RAND.
Disadvantages	Always + LFU (i) Computationally expensive (O(n) updates) (ii) Susceptible to stale content pollution if access patterns shift
Disadvantages	Prob(p) + LFU (i) Higher server load than Always+ LFU (ii) Cache polluted by stale content with high past frequency (iii)Poor hit rate compared to Always+ LFU (iv) Converges to a weak steady state, worse with small p
23)	FIFO (First in First Out)[25]
Mechanism	Evicts the oldest data item in memory, meaning that the item loaded first is replaced by new incoming data.
Parameters	Metrics: Execution Time (Wall Clock Time), Memory Utilization (RAM) Topology: Binary Tree Netwotk Topology Simulator: Icarus
Advantages	(i) Simple to implement, with low computational overhead. (ii) provides predictable behavior by always removing the oldest item. (iii) efficient for streaming data where older items are less likely to be reused. (iv) requires no tracking of access frequency or recency. (v) Memory utilization does not vary much across policies (NULL ≈ LRU ≈ FIFO ≈ RAND).
Disadvantages	(i) Does not consider content popularity, so frequently accessed items may be evicted prematurely. (ii) performs poorly in non-sequential or irregular access patterns. (iii) lacks of adaptability to changing popularity trends. (iv) prone to thrashing when the working set exceeds cache size. (v) results in the highest stretch ratio among common policies.
24	Least Recently Frequently Used (LRFU) [38]
Mechanis	Uses Combined Recency Frequency (CRF) values to prioritize cached items, considering both how often and how recently they were accessed. LRFU merges the principles of LFU and LRU.

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

Parameters	Metrics: Hit Rate, Miss Rate Topology: No specific multi-node layout, links, or paths are modeled—the emphasis is on cache behavior per node. Simulator: Custom C++ implementation using GCC compiler (no standard simulator like ndnSIM was used)
Advantages	(i) Makes more informed decisions by balancing recency and frequency, improving cache performance. (ii) Produces a lower miss rate compared to LRU, especially as the number of generated interests increases. (iii) Performs better with larger cache sizes, making it suitable for systems with high storage capacity. (iv) Dynamically adapts to changing access patterns through CRF values. (v) Enhances user satisfaction by retaining relevant content and reducing retrieval latency.
Disadvantages	(i) More complex to implement compared to LRU or LFU due to CRF maintenance. (ii) Continuous CRF updates introduce computational overhead, especially in high-frequency access environments. (iii) May still underperform in bursty or highly unpredictable access patterns. (iv) Can evict still-relevant content if it is not recently accessed. (v) Requires higher memory usage (maintains both heap list and linked list), which may be problematic in resource-constrained systems.
C.	**Dynamic Cache Replacement Techniques**
25)	Popularity-based Network Coding (PopNetCod) [39]
Mechanism	In this method Each router measures the local popularity of content objects dynamically by analysing incoming requests. The replacement of data from content store is based on the popularity information collected.
Parameters	Metrics: Cache-hit rate, Client Goodput, Video Quality Distribution, Source Load Reduction Topology: Layered Topology of 1 source ,123 clients and 45 routers Simulator: Custom NetCodNDN-based simulator (event-driven)
Advantages	(i) Higher cache-hit rate than LCE+LRU (ii) Increased goodput at clients (iii) More high-quality video segments delivered (1080p) (iv) Reduced source load (~10% less vs LCE+LRU) (v) Better cache utilization by storing popular content at the edge. (vi) Improve user experience.
Disadvantages	(i) More complex (popularity estimation + network coding) (ii) Limited cache size (0.9%–2.3% of source data) (iii) Dependent on adaptation logic (dash.js) (iv) Higher initial delay vs LCE+NoLimit (v) Implementation harder than simple LRU/LCE Potential problem of cache pollution.
26)	Content Popularity Based Caching(CPBC)[40]
Mechanis	CPBC operates by maintaining a data structure that tracks the popularity of content within a cache.
Parameters	Metrics: cache hit ratio (CHR), Average number of hops Topology: Barabasi-Albert (BA) model network
Advantages	(i) High cache hit ratio due to popularity-aware content assignment. (ii)Reduced average number of hops → faster content delivery. (iii) Load balancing across management nodes. (iv) Efficient utilization of network resources and management nodes. (v)Reduced latency. (vi) Efficient bandwidth usage. (vii) Adaptability.

Disadvantages	(i) CPBC may lead to resource underutilization if the popularity of cached content changes rapidly. (ii) Performance Under Fluctuating Demand. (iii) Requires knowledge of content popularity in advance. (iv) ILP(Integer Linear Programming) optimization can be computationally expensive for large networks. (v) Does not account for propagation delay, processing time, or bandwidth limitations. (vi) Performance may degrade if popularity distribution changes dynamically.
27	Bloom-filter-based request node collaboration caching(BRCC)[41]
Mechanism	BRCC dynamically adjust data deployment based on request frequency and proximity to subscriber.
Parameters	Metrics: cache hit ratio (CHR), first hop hit ratio(FHHR), average route hop (ARH), average request delay (ARD) Topology: NDN Network Topology Simulator: MATLAB + C++ integrated simulator
Advantages	(i) Higher CHR & FHHR than LCE, ProbCache, WAVE (ii) Caches near subscribers to reduce delay (iii) Collaborative push of evicted data toward core (iv) Reduces redundancy and improves diversity (v) Efficient storage space utilization (vi) Lower ARH and ARD across varying cache sizes
Disadvantages	(i) Higher computational complexity (Bloom filter + collaborative decision logic) (ii) Performance depends on accurate popularity & request patterns (iii) Overhead in maintaining cache collaboration (iv) May be less efficient under sudden shifts in content demand
28	Adaptive Replacement Cache (ARC)[42]
Mechanism	ARC took advantages of two known caching strategies LRU and LFU. It manages the Cache entries based on both recency and frequency.
Parameters	Metrics: Cache Hit Count , Cache Miss Count, Cache Hit Ratio Topology: (3x3),(4x4), (5x5) Grid Topology Simulator: ndnSIM
Advantages	(i) Achieves higher hit rate than LRU (up to 4–5% improvement). (ii) Requires smaller Content Store size to achieve the same hit rate as LRU (e.g., 60 vs 100). (iii) Adapts effectively between recently used and frequently used content. (iv) No manual tuning parameters required. (v) Better cache utilization and storage efficiency.
Disadvantages	(i) Performance gain decreases as interest rate increases; converges with LRU at high rates. (ii) Slightly more complex than LRU due to dual-list management (T1 and T2). Hit rate decreases as grid size increases. (iii) Slightly higher computational overhead compared to LRU.
29	EEC(Energy Efficient in Caching) [43]
Mechanis	EEC employs an energy efficient cache placement (EECP) strategy that determines caching decisions based on energy consumption. It optimize the energy efficiency of content caching and transmission.

Parameters	Metrics: Cache hit rate (CHR), Average response hops (ARH), Energy saving rate (ESR) Topology: Physical Network Topology Simulator: ndnSIM
Advantages	(i) Highest cache hit rate (up to 75.4% for $\alpha=1$) (ii) Lowest average response hops (5.04 for $\alpha=1$) (iii) Highest energy saving rate (64.3% for $\alpha=1$) (iv) Effectively uses neighbor cache space (v) Adapts caching based on content popularity and node centrality (vii) Reduces latency and network energy consumption
Disadvantages	(i) Performance depends on cache size, content number, and α (ii) More complex cache management due to neighbor cooperation (iii) Requires accurate estimation of content popularity (iv) Limited improvement if cache space is insufficient
30	Name Popularity Algorithm (NPA)[44]
Mechanism	NPU operates by maintaining a history of content popularity, allowing it to make informed decisions about which items to keep in the cache.
Parameters	Metrics: CHR, No. of Hops, End-to-End Delay Topology: Telstra Topology, AT&T Topology, Tiscali Topology Simulator:ndnSIM

Advantages	(i) Maintains content popularity info via History Table (HT), even after content is evicted. Outperforms LRU, LFU, LFUDA, RANDOM across all real topologies. (ii) ~10–19% higher cache hit ratio. (iii) Lower hop count (up to 3.5% reduction). (iv) Lower end-to-end delay (up to ~13.8% reduction). (v) Reduces congestion & retransmissions by lowering hops and increasing hits. (vi) Performs best at Zipf $\alpha = 0.7$, widely considered realistic. Higher cache hit ratio, Adaptability,
Disadvantages	(i) Extra memory overhead due to History Table. (ii) Performance depends on proper HT size tuning (3% of CS size found best). (iii) Does not handle time-varying popularity (risk of cache pollution). (iv) Tested only with one producer – scalability with multiple producers not evaluated. (v) Results based on synthetic Zipf traffic, not real traces.
31	Content Popularity ranking (CPR)[45]
Mechanism	CPR assigns a ranking to content based on how frequently it is requested. Only content that exceeds a certain popularity threshold gets cached. When cache space is full the least content is evicted first.

Parameters	Metrics: Average Content Delivery Time Topology: Testbed architecture i.e. Real-world ICN-based testbed with 5 geographically distributed servers (3 remote + 2 cloud) and 10 client devices connected via Ethernet/Wi-Fi/4G across two cellular base stations; topology configurable and scalable Simulator: they built a real-world ICN testbed
Advantages	(i) Popularity-Aware Caching – Considers request frequency, file type, and content age to rank popularity, ensuring that frequently used and relevant content stays in cache. (ii) Efficient Cache Utilization – Prevents duplication across all servers by storing popular content closer to clients, improving overall cache hit ratio and reducing redundancy. (iii) Reduced Content Delivery Time – By caching popular content near consumers, response time is minimized compared to basic CCN and the current Internet. (iv) Mobility Support – Works with the developed mobility function to ensure seamless content delivery even when clients move between base stations. (v) Real-World Validation – Unlike many simulation-based studies, CPR was implemented and tested on a real ICN testbed, demonstrating practical feasibility. (vi) Dynamic Adaptation – PopularityThreshold allows caching decisions to adjust automatically to changing content popularity.
Disadvantages	(i) Limited Testbed Scale – Evaluated on only 5 servers and 10 clients; scalability to larger networks was not studied. (ii) Incomplete Formalization – Algorithm details were described conceptually, with promise of full proposal in future work; mathematical modeling is missing. (iii) Focus on Single Metric – Primarily optimized for content delivery time, while other key caching metrics (e.g., bandwidth savings, server hit reduction ratio) were not fully analyzed. (iv) Producer Scalability Not Addressed – Paper mentions scalability issues with multiple producers as a limitation for future work. (v) Overhead of Popularity Calculation – Continuous tracking of request counts, file types, and publication times may introduce computational overhead in large-scale deployments. (vi) Mobility Dependency – Performance benefits partly depend on the additional mobility support function, making it less isolated as a pure caching strategy.
32	MultiCache[46]
Mechanism	It is an overlay network architecture designed to enhance control for network operators by utilizing a distributed caching scheme. It aims to improve traffic localization and resource utilization, addressing inefficiencies in the current Internet architecture. The study also explores the feasibility of deploying this functionality within existing networks.
Parameters	Metrics: cache hit ratio (CHR) and intra-domain cache hit ratio (CHR-Intra) Topology: Generated using GT-ITM (Georgia Tech Internet Topology Model) Simulator: OMNeT++ and OverSim Framework

Advantages	(i) Achieves very high cache hit ratio (up to 98.5%) (ii) Reduces overlay multicast traffic through traffic localization. (iii) Overlay-aware eviction prevents premature removal of active data. (iv) No extra control signaling overhead for cache discovery. (v) Supports flexible replacement policies (LRU, MRU, MFU). (vi) Effective performance with only 25–50% deployment density
Disadvantages	(i) Requires additional OAR infrastructure, increasing cost and complexity. (ii) Cache replacement policies (LRU, MRU, MFU) show similar performance, no clear superior one, (iii) MFU chosen mainly for simplicity, not better performance. (iv) Fragment-level caching adds management complexity. (v) High localizability factor may overload caches and increase eviction. (vi) Dense deployments reduce request aggregation and local cache hit rates
33	The QoS-aware Cache Replacement (QCR) [47]
Mechanism	QCR policy categorizes the cache store into multiple sub-cache stores based on different traffic. Each sub-cache has a varying storage capacity tailored to the network's needs. The policy evaluates content using a popularity-density value, which balances the content's popularity with its size, ensuring that the most valuable content is retained while less valuable content is evicted.
Parameters	Metrics: No. of cached data, No. of replacement operation, No. of Ignored Data, cache usage behavior across Class A, B, C (traffic categories) Topology : Multi-hop Vehicular Network Topology simulator: Intel Core 5 Duo CPU
Advantages	(i) Popularity & Size-Aware means it uses popularity-density value (σc = popularity/size) → ensures that only useful and popular content is cached. (ii) When the popularity-density value increases, QCR is working exponentially. (iii) It considers the data packet's QoS. (iv) It considers data chunk's popularity. (v) Consider the size of data packet.
Disadvantages	(i) High implementation complexity, Adding extra bits like c-tag and p-tag to data chunks and cache store table. (ii) Splitting cache into sub-caches may lead to under-utilization if one class is under-loaded while another is overloaded. (iii) Caching happens during content forwarding using PIT info, not immediately. (iv) QCR ignores Some Useful Data.
34	Dynamic fine-grained popularity-based cache replacement (FGPC) & Dynamic Fine-Grained Popularity-based Caching (DFGPC)[48]
Types	Dynamic Cache Replacement [FGPC = static threshold, less adaptive & DFGPC= dynamic threshold, fully adaptive]
Mechanism	It maintains a popularity table that records the content name, access count, and timestamps to assess the popularity of each content item. This allows the system to make informed caching decisions based on real-time data about content usage patterns
Parameters	Metrics: Cache Hit Ratio (Hitting Rate), Effect of Cache Size, Effect of Simulation Time, Impact of File Size (α factor) Topology: three-layer hierarchical Internet-like topology Simulator: OPNET Modeler 16.0

Advantages	FGPC Advantages: (i) High cache hit ratio. (ii) Low latency. (iii) Short stretch ratio. (iv) Efficient utilization of cache space. (v) Selectivity: Keeps only the most popular content when cache is full. (vi) Improved performance over MPC. (vii) Handles file size variations better;
Advantage	DFGPC Advantages: (i) Dynamic adaptability (ii) Consistently higher hit rate. (iii) Cache size independence' File size resilience
Disadvantages	FGPC Dis advantages: (i) High redundancy ratio (ii) High memory consumption (iii) High bandwidth, minimum diversity ratio. (iv) Threshold sensitivity; Temporary performance drops. (v) Not fully adaptive: Thresholds are fixed, so it cannot adjust dynamically to changing traffic patterns
Disadvantages	DFGPC Disadvantages: (i) Complexity (ii) Overhead
35	Discard of Fast Retrievable Content (DFRC) [7]
Mechanis	Uses FIB table information to calculate content retrieval time based on two parameters: Grade of Retrieval (GOR) and Stale Parameter (SP). Content with shorter retrieval time is assigned a higher discard priority.
Parameters	Metrics: Average hit rate, Average round trip time, Redundancy of the data chunks Topology: WIDE (Japanese Internet backbone)., GARR (Italian university/research network), GEANT (European backbone) . Simulator:Icarus
Advantages	(i) Achieves the highest cache hit rate among all compared strategies (LRU, SRTT, OCRICN, EPPC). (ii) Dynamically adapts to content popularity changes over time using SP, discarding old popular content and storing new popular content. (iii) Reduces average RTT and improves user response time by prioritizing storage of content with longer retrieval times. (iv) Reduces redundant storage of data chunks compared to LRU and EPPC, leading to more efficient cache utilization. (v) Performs well across different network topologies (GEANT, GARR, WIDE) and varying cache sizes. (vi) Consistently outperforms strategies that assume fixed content popularity (OCRICN, SRTT). (vii) Improves network efficiency by reducing unnecessary duplication and balancing cache usage.
Disadvantages	(i) Slightly higher redundancy than OCRICN due to the SP mechanism. (ii) Requires continuous monitoring and updating of content popularity, which may introduce computational overhead. (iii) Complexity is higher compared to simpler strategies like LRU. (iv) Performance depends on correct tuning of the SP parameter and reinforcement learning U parameter. (v) May require additional simulation or real-time computation for optimal parameter selection. (vi) Assumes Zipf-law content popularity; performance may vary with non-Zipf distributions. (vii) Implementation may be more complex due to reinforcement learning and dynamic content tracking.
36	PFR (Popularity, Freshness, and Recency) based cache content eviction policy.[49]
Mechanism	PFR evaluates each cached content based on Popularity (request frequency), Freshness (remaining lifetime), and Recency (last access time). It computes a combined EvictionValue and removes the content with the highest value when cache space is needed. This process is dynamic, continuously adapting to new requests and content aging, ensuring that the most essential and relevant data remains in the cache.

Parameters	Metrics: CHR, Server hit reduction ratio, Average response delay, Average energy consumption Topology: Randomly generated Simulator: ndnSIM
Advantages	(i) Retains highly popular, fresh, and recently used content, improving cache hit ratio. (ii) Reduces the number of requests reaching the IoT publisher, lowering server load. (iii) Lowers average response delay, providing faster data access to subscribers. (iv) Decreases average energy consumption due to fewer retransmissions. Adapts to dynamic content characteristics in NDN-IoT networks.
Disadvantages	(i) Requires calculation of multiple attributes (popularity, freshness, recency), adding computational overhead. (ii) Performance depends on proper weight selection for attributes; suboptimal weights can reduce efficiency. (iii) Slightly more complex to implement than simpler policies like FIFO or LRU. (iv) May evict less popular content even if it could be requested later, potentially leading to occasional misses. (v) Needs continuous monitoring of content attributes, which can be challenging in resource-constrained IoT devices.
37	AI/ML based caching[49], [50]
Mechanism	AI/ML-based caching manages content selection, placement, and replacement by predicting content popularity and user demand. It uses historical data and real-time analytics to ensure that the most requested contents are available to users. many machine learning algorithm used in this case to optimize caching decision. Like DQL (Deep Q-Learning),DRL (Deep Reinforcement Learning),NMF (Non-negative Matrix Factorization), ANN (Artificial Neural Network), Deep RNN (Recurrent Neural Network), RL (Reinforcement Learning), RNN (Recurrent Neural Network), SNN (Spiking Neural Network),CNN (Convolutional Neural Network), MLP (Multi-Layer Perceptron), ILP (Integer Linear Programming), Q-Learning, Hyper Deep Q-Networks (DQNs)
Parameters	Metrics: Network scenario: ICN,NDN, Edge Computing etc. simulator : Any simulator can use for the performance analysis. like ndnSIM, Icarus, miniNDN, ccnSIM
Advantages	(i) Improved Prediction Accuracy: AI/ML algorithms analyze historical data to predict content popularity more accurately, leading to higher cache hit ratios, reduced latency, and enhanced user experience. (ii) Dynamic Adaptability: Can adapt in real time to changes in user behavior and network conditions, maintaining relevance of cached content. (iii) Enhanced Resource Utilization: Improves use of limited cache storage, reduces redundancy, and enhances overall network performance. (iv) Overcoming Traditional Limitations: Traditional caching techniques often cannot handle the dynamic nature of content popularity and network topology. ML/DL provides a way to analyze data, generate insights, and predict user needs, leading to more efficient caching systems
Disadvantages	(i) Complexity and Overhead: AI/ML models introduce computational overhead for training and inference, requiring more resources, which may be infeasible in some environments. (ii) Data Dependency: Performance relies heavily on the availability of high-quality historical data. Poor or insufficient data can lead to inaccurate predictions and degraded caching performance.

38	Max-Gain In Network Caching (MAGIC) [15]
Mechanism	Functions as both a cache placement and cache replacement algorithm. It updates content popularity based on local cache gain and max-gain values, aiming to reduce bandwidth consumption.
Parameters	Metrics : Bandwidth consumption, Server hit ratio, Caching operations. Topology: wireless network topology simulator: SocialCCNSim
Advantages	(i) Reduces bandwidth by 34.5%. (ii) Lowers server hit ratio by 17.91% . (iii) Reduces caching ops by 38.84%.[52] (iv) Provides high content diversity and achieves a high cache hit ratio under certain conditions. (v) Bandwidth efficiency. (vi)Popularity-aware caching. (vii) Fairer caching placement
Disadvantages	(i) Overhead of Max-gain calculation. (ii) Extra fields in Interest/Data packets. (iii) Biased caching toward heavy-request consumers. Limited scalability. (iv) Replacement cost not fully optimized. (v) Requires more resources and incurs higher costs due to its complex mechanism. May increase retrieval times and lead to lower cache hit ratios compared to CPCCS. Also results in higher memory consumption.

B. *Insight from Comparative analysis*

As summarised table 1, the cache placement strategies such as LCE, LCD, Magic etc. do not leverage with the PIT or FIB meta data which make reduction in the feature of network dynamicity, where the static cache replacement techniques like FIFO, LRU, LFU, RR and LRFU do not take the decisions based on real time data, hence these algorithms are static in nature which base on specific rule consecutively lead to degrade the named data networking performance. In contrast, by making caching decisions based on PIT-FIB, it will become adaptive and dynamically respond to network conditions and improve overall performance. Also, in dynamic cache replacement strategies, mostly strategies based on probability and historical data, they did not take real-time network data for replacement. Apart from this, most of the machines learning algorithms are either model-based or based on static data-driven, so there is a gap regarding the use of model-free machine learning in NDN for adaptive, real-time caching.

VI. Research Gaps and Limitations

Based on our literature review, two key research gaps remain in named data networking (NDN).
1. The absence of caching strategies that jointly use the PIT and FIB information for adaptive and real-time decisions.
2. The limited use of model-free machine learning to enhance the performance of NDN cache replacement.
Despite significant progress, many NDN caching strategies still face challenges related to content retrieval time, cache replacement policies. The integration of AI and machine learning offers promising opportunities to address these issues. We have gone through the various research papers from 2020 to 2025 related to caching and drew a pie chart and bar graph as fig. (6), fig. (7) based on two parameters, the research paper which was based on caching strategies with AI/ML technology vs. non-AI/ML, where Figure 6 clearly indicates that research in caching based on AI/ML is less than caching without any new technology of AI/ML and Figure 7 shows the research gap regarding the lesser use of the PIT and FIB tables.

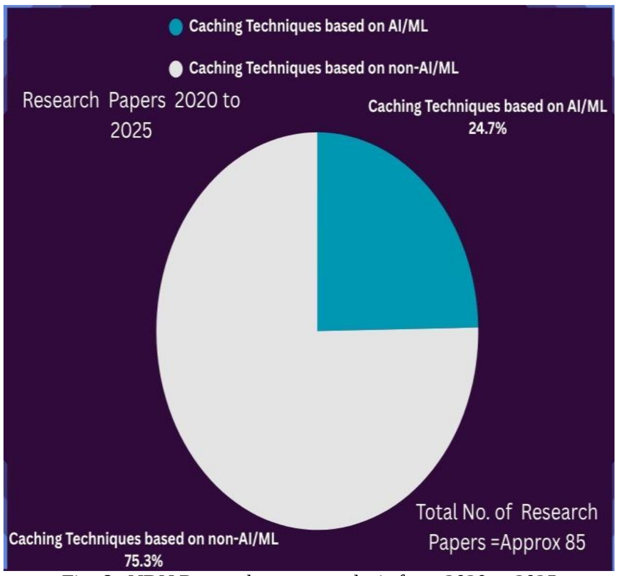

Fig. 6. NDN Research papers analysis from 2020 to 2025

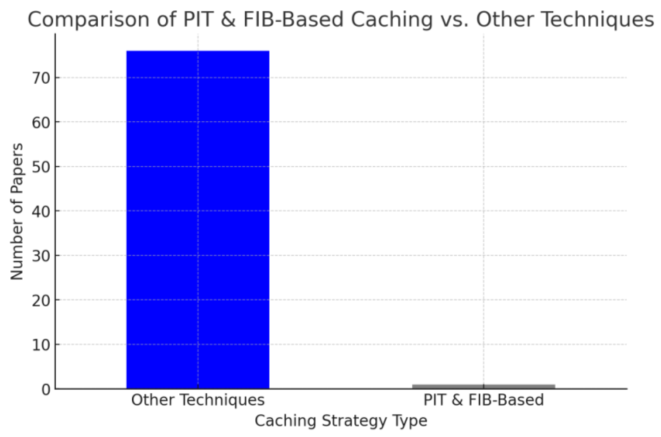

Fig .7. Numbers of papers based on caching with PIT and FIB.

VII. CURRENT CHALLENGES AND EMERGING TRENDS

Most existing caching strategies in Named Data Networking (NDN) rely primarily on historical content popularity rather than real-time request information obtained from the Pending Interest Table (PIT). Leveraging PIT data, such as the number of incoming faces, outgoing faces, request arrival rates, and content lifetime, along with relevant Forwarding Information Base (FIB) parameters, could significantly enhance caching decisions and overall network performance. However, without PIT-aware caching, redundant requests and suboptimal cache utilisation continue to limit efficiency, particularly in high-traffic environments. Another challenge is the increasing content demand in large-scale NDN deployments, which puts pressure on cache capacity and retrieval latency. While advanced machine learning algorithms can potentially optimise caching policies, highly complex models may introduce scalability issues, high computational overhead, and long convergence times. To address these limitations, model-free reinforcement learning approaches—such as Q-learning, multi-armed bandits, and deep reinforcement learning—are gaining attention in recent studies due to their ability to make adaptive caching decisions without the need for labelled datasets or extensive offline training. Integrating these approaches with PIT- and FIB-aware caching policies presents a promising research direction for improving adaptability, scalability, and efficiency in future NDN networks

VIII. FUTURE DIRECTION

There are various research areas where we can explore the solution to the problem in terms of caching-related issues. There are various caching attacks for e.g., time analysis, bogus announcements, cache pollution, and cache spoofing, in which we can work to maintain the integrity of caching [53]. The integration of on-path caching with emerging technologies such as IoT, edge computing, and 5G networks is also one of the major challenges. By caching content during transmission, this approach can enhance network efficiency, reduce latency, and improve overall scalability. With this understanding, there is a need to develop advanced caching strategies that incorporate intelligent criteria for content placement and efficient decision-making for content replacement. Applying machine learning techniques—including Markov Decision Processes (MDP), Q-learning, and dynamic programming—represents a promising avenue. These machine learning algorithm is based on feedback from the environment, so if we apply them in the named data networking, then definitely performance will be increased, and also we know that routers maintain both tables i.e. PIT and FIB which is based on real time data if we use these data for cache replacement then the cache replacement will be based on current networking condition. So future research should focus on designing adaptive cache replacement, which should be based on both tables PIT and FIB because details like user behaviour and environmental changes, interest packet and data packet are the main part of communication for sending requests and obtaining the reply in the NDN, and all these details are maintained by the same. Leveraging these parameters can improve cache efficiency, reduce redundancy, and provide smarter, adaptive content distribution across the network.

IX. CONCLUSION

This research paper highlighted the importance of caching strategies in Named Data Networking (NDN) and their impact on network performance. This manuscript reviewed and categorized existing approaches into two major groups: cache placement and cache replacement, where the basic e.g. for cache placements and replacements are LCD, LCE and FIFO, LRU, LFU, and RR, respectively. While these foundational strategies are simple and widely used but they are static in nature and cannot adapt to dynamic network conditions, such as fluctuations in content popularity or variations in PIT in-records and out-records. The advanced methods of caching, such as MAGIC, WAVE, and CPBC, achieve higher cache hit ratios and lower latency, but often introduce computational complexity. No single caching strategy is universally optimal; performance depends on network conditions, topology, and

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

workload. The paper also reveals that simulation methodologies in NDN caching research remain highly diverse in terms of metrics, network topologies, and simulator tools. It is noted that while algorithmic evaluations can be carried out on various simulators, ndnSIM remains the most reliable platform for actual performance measurement in NDN. Cache Hit Ratio (CHR) emerges as the most widely adopted evaluation metric, often complemented with measures such as latency, hop count, and bandwidth savings. However, advanced metrics like energy efficiency, fairness, and computational overhead are less frequently explored, limiting insights into caching performance under real-world constraints. Similarly, topology choices vary from simplified tree structures to realistic ISP-level graphs and IoT or vehicular scenarios, yet there is no standardized benchmark that enables fair cross-comparison. The contribution of this study lies in presenting a clear comparative analysis of cache placement and replacement strategies, identifying their strengths and weaknesses, and underscoring the research gap in exploiting PIT and FIB information for caching decisions. Furthermore, it emphasized the potential of machine learning to design intelligent caching strategies with the help of graph neural network and Apriori algorithm that adapt to dynamic network environments to enhance the caching decision at the same time also tells that too much use of machine learning in NDN caching leads to data dependency, complexity, memory consumption and communication overhead which will effect to slow down the performance under certain condition. This script also specifies that future work should focus on developing hybrid caching models that balance computational efficiency with adaptability, leveraging PIT/FIB data and lightweight ML techniques to achieve scalable and high-performance caching in next-generation NDN architectures.

References

[1] J. Hou, H. Xia, H. Lu, and A. Nayak, "A Graph Neural Network Approach for Caching Performance Optimization in NDN Networks," IEEE Access, vol. 10, pp. 112657–112668, 2022, doi: 10.1109/ACCESS.2022.3217236.

[2] S. U. Taki, A. Chakrabarty, Md. J. Piran, Q.-V. Pham, and D. Y. Suh, "An Indoor Positioning and Navigation System Using Named Data Networking," IEEE Access, vol. 8, pp. 196408–196424, 2020, doi: 10.1109/access.2020.3034114.

[3] T. Guo, Y. Ma, M. Zhou, X. Wang, J. Wu, and Y. Chen, "SocialCache: A Pervasive Social-Aware Caching Strategy for Self-Operated Content Delivery Networks of Online Social Networks," in ICC 2023 - IEEE International Conference on Communications, Rome, Italy: IEEE, May 2023, pp. 4931–4936. doi: 10.1109/icc45041.2023.10279588.

[4] M. N. D. Satria, F. H. Ilma, and N. R. Syambas, "Performance comparison of named data networking and IP-based networking in palapa ring network," in 2017 3rd International Conference on Wireless and Telematics (ICWT), Palembang: IEEE, July 2017, pp. 43–48. doi: 10.1109/ICWT.2017.8284136.

[5] Z. Sabir and A. Amine, "NDN vs TCP/IP: Which One Is the Best Suitable for Connected Vehicles?," in Recent Advances in Mathematics and Technology, S. Dos Santos, M. Maslouhi, and K. A. Okoudjou, Eds., in Applied and Numerical Harmonic Analysis. , Cham: Springer International Publishing, 2020, pp. 151–159. doi: 10.1007/978-3-030-35202-8_9.

[6] K. T. Ko, H. H. Hlaing, and M. Mambo, "A PEKS-Based NDN Strategy for Name Privacy," Future Internet, vol. 12, no. 8, p. 130, July 2020, doi: 10.3390/fi12080130.

[7] M. Hosseinzadeh, N. Moghim, S. Taheri, and N. Gholami, "A new cache replacement policy in named data network based on FIB table information," Telecommun Syst, vol. 86, no. 3, pp. 585–596, July 2024, doi: 10.1007/s11235-024-01140-7.

[8] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the Evolution of ndnSIM: an Open-Source Simulator for NDN Experimentation," SIGCOMM Comput. Commun. Rev., vol. 47, no. 3, pp. 19–33, Sept. 2017, doi: 10.1145/3138808.3138812.

[9] M. A. Naeem, M. A. U. Rehman, R. Ullah, and B.-S. Kim, "A Comparative Performance Analysis of Popularity-Based Caching Strategies in Named Data Networking," IEEE Access, vol. 8, pp. 50057–50077, 2020, doi: 10.1109/ACCESS.2020.2980385.

[10] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in Proceedings of the second edition of the ICN workshop on Information-centric networking, Helsinki Finland: ACM, Aug. 2012, pp. 55–60. doi: 10.1145/2342488.2342501.

[11] Y. Gui and Y. Chen, "A Cache Placement Strategy Based on Compound Popularity in Named Data Networking," IEEE Access, vol. 8, pp. 196002–196012, 2020, doi: 10.1109/ACCESS.2020.3034329.

[12] N. Hazrati, S. Pirahesh, B. Arasteh, S. S. Sefati, O. Fratu, and S. Halunga, "Cache Aging with Learning (CAL): A Freshness-Based Data Caching Method for Information-Centric Networking on the Internet of Things (IoT)," Future Internet, vol. 17, no. 1, p. 11, Jan. 2025, doi: 10.3390/fi17010011.

[13] C. A. Kerrache, G. Rathee, Y. Guellouma, H. Gasmi, and B. Ziani, "Towards an Efficient and Secure Cache Management Using Apriori-Based Interests Prediction in Named Data Networking," in 2023 11th International Conference on Intelligent Systems and Embedded Design (ISED), Dehradun, India: IEEE, Dec. 2023, pp. 1–6. doi: 10.1109/ised59382.2023.10444543.

[14] M. Wasim Abbas Ashraf, A. Raza, A. R. Singh, R. Singh Rathore, I. W. Damaj, and H. Herbert Song, "Intelligent Caching Based on Popular Content in Vehicular Networks: A Deep Transfer Learning Approach," IEEE Trans. Intell. Transport. Syst., vol. 25, no. 12, pp. 20643–20656, Dec. 2024, doi: 10.1109/tits.2024.3445640.

[15] M. Yu, R. Li, and Y.-W. Chen, "A Cache Replacement Policy Based on Multi-factors for Named Data Networking," Computers, Materials & Continua, vol. 65, no. 1, pp. 321–336, 2020, doi: 10.32604/cmc.2020.010831.

[16] Y. Zhu, Z. Mi, and W. Wang, "A Cache Probability Replacement Policy Based on Content Popularity in Content Centric Networks: A Cache Probability Replacement Policy Based on Content Popularity in Content Centric Networks," Journal of Electronics & Information Technology, vol. 35, no. 6, pp. 1305–1310, Feb. 2014, doi: 10.3724/SP.J.1146.2012.01143.

[17] Z. Fan, Q. Wu, M. Zhang, and R. Zheng, "Popularity and gain based caching scheme for information-centric networks," IJACR, vol. 7, no. 30, pp. 71–80, Apr. 2017, doi: 10.19101/IJACR.2017.730015.

[18] Haipeng Li, H. Nakazato, A. Detti, and N. B. Melazzi, "Popularity Proportional Cache Size Allocation policy for video delivery on CCN," in 2015 European Conference on Networks and Communications (EuCNC), Paris, France: IEEE, June 2015, pp. 434–438. doi: 10.1109/EuCNC.2015.7194113.

Correspondence to: Sushil Kumar Bagi, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: sushil.23183963@mygyanvihar.com

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

[19] Y. Zha, P. Cui, Y. Hu, L. Xue, J. Lan, and Y. Wang, "An NDN Cache-Optimization Strategy Based on Dynamic Popularity and Replacement Value," Electronics, vol. 11, no. 19, p. 3014, Sept. 2022, doi: 10.3390/electronics11193014.

[20] C. Bernardini, T. Silverston, and O. Festor, "MPC: Popularity-based caching strategy for content centric networks," in 2013 IEEE International Conference on Communications (ICC), Budapest, Hungary: IEEE, June 2013, pp. 3619–3623. doi: 10.1109/ICC.2013.6655114.

[21] M. Yu, R. Li, Y. Liu, and Y. Li, "A caching strategy based on content popularity and router level for NDN," in 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), Macau, China: IEEE, July 2017, pp. 195–198. doi: 10.1109/ICEIEC.2017.8076542.

[22] "W.-X. Liu, J. Zhang, Z.-W. Liang, L.-X. Peng, and J. Cai, '"Content popularity prediction and caching for ICN: A deep learning approach with SDN,"' IEEE Access, vol. 6, pp. 5075–5089, 2017.".

[23] Y. Meng, M. A. Naeem, R. Ali, and B.-S. Kim, "EHCP: An Efficient Hybrid Content Placement Strategy in Named Data Network Caching," IEEE Access, vol. 7, pp. 155601–155611, 2019, doi: 10.1109/ACCESS.2019.2946184.

[24] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and Sangheon Pack, "WAVE: Popularity-based and collaborative in-network caching for content-oriented networks," in 2012 Proceedings IEEE INFOCOM Workshops, Orlando, FL, USA: IEEE, Mar. 2012, pp. 316–321. doi: 10.1109/INFCOMW.2012.6193512.

[25] L. Saino, I. Psaras, and G. Pavlou, "Icarus: a Caching Simulator for Information Centric Networking (ICN)," in Proceedings of the Seventh International Conference on Simulation Tools and Techniques, Lisbon, Portugal: ICST, 2014. doi: 10.4108/icst.simutools.2014.254630.

[26] R. M. Negara, N. Rachmana Syambas, and E. Mulyana, "Integration Of Content Size Adjustment And Caching Policy to Improve Named Data Networking Performance," in 2022 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), Bandung, Indonesia: IEEE, Dec. 2022, pp. 1–5. doi: 10.1109/apwimob56856.2022.10014284.

[27] Han, Yuqi & Wang, Rui & Wu, Jun. (2019). Random Caching Optimization in Large-Scale Cache-Enabled Internet of Things Networks. IEEE Transactions on Network Science and Engineering. PP. 1-1. 10.1109/TNSE.2019.2894033.

[28] X. Zheng, G. Wang, and Q. Zhao, "A Cache Placement Strategy with Energy Consumption Optimization in Information-Centric Networking," Future Internet, vol. 11, no. 3, p. 64, Mar. 2019, doi: 10.3390/fi11030064.

[29] K. N. Lal and A. Kumar, "A Centrality-measures based Caching Scheme for Content-centric Networking (CCN)," Multimed Tools Appl, vol. 77, no. 14, pp. 17625–17642, July 2018, doi: 10.1007/s11042-017-5183-y.

[30] L. Saino, I. Psaras, and G. Pavlou, "Hash-routing schemes for information centric networking," in Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking, Hong Kong China: ACM, Aug. 2013, pp. 27–32. doi: 10.1145/2491224.2491232.

[31] R. Alubady, M. Salman, and A. S. Mohamed, "A review of modern caching strategies in named data network: overview, classification, and research directions," Telecommun Syst, vol. 84, no. 4, pp. 581–626, Dec. 2023, doi: 10.1007/s11235-023-01015-3.

[32] F. Dehghani and N. Movahhedinia, "Energy-delay-aware caching strategy in green CCN using markov approximation," Int J Communication, vol. 32, no. 15, p. e4109, Oct. 2019, doi: 10.1002/dac.4109.

[33] S. Eum, K. Nakauchi, M. Murata, Y. Shoji, and N. Nishinaga, "CATT: potential based routing with content caching for ICN," in Proceedings of the second edition of the ICN workshop on Information-centric networking, Helsinki Finland: ACM, Aug. 2012, pp. 49–54. doi: 10.1145/2342488.2342500.

[34] M. Rezazad and Y. C. Tay, "Decoupling NDN caches via CCndnS: Design, analysis, and application," Computer Communications, vol. 151, pp. 338–354, Feb. 2020, doi: 10.1016/j.comcom.2019.12.053.

[35] M. Zhang, B. Hao, R. Wang, and Y. Wang, "A Pre-Caching Strategy Based on the Content Relevance of Smart Device's Request in Information-Centric IoT," IEEE Access, vol. 8, pp. 75761–75771, 2020, doi: 10.1109/ACCESS.2020.2988926.

[36] Y. Gui and Y. Chen, "A Cache Placement Strategy Based on Entropy Weighting Method and TOPSIS in Named Data Networking," IEEE Access, vol. 9, pp. 56240–56252, 2021, doi: 10.1109/ACCESS.2021.3071427.

[37] S. Tarnoi, K. Suksomboon, W. Kumwilaisak, and Yusheng Ji, "Performance of probabilistic caching and cache replacement policies for Content-Centric Networks," in 39th Annual IEEE Conference on Local Computer Networks, Edmonton, AB: IEEE, Sept. 2014, pp. 99–106. doi: 10.1109/LCN.2014.6925761.

[38] M. A. P. Putra, H. Situmorang, and N. R. Syambas, "Least Recently Frequently Used Replacement Policy Named Data Networking Approach," in 2019 International Conference on Electrical Engineering and Informatics (ICEEI), Bandung, Indonesia: IEEE, July 2019, pp. 423–427. doi: 10.1109/ICEEI47359.2019.8988828.

[39] J. Saltarin, T. Braun, E. Bourtsoulatze, and N. Thomos, "PopNetCod: A Popularity-based Caching Policy for Network Coding enabled Named Data Networking," in 2018 IFIP Networking Conference (IFIP Networking) and Workshops, Zurich, Switzerland: IEEE, May 2018, pp. 271–279. doi: 10.23919/IFIPNetworking.2018.8696704.

[40] T. Fukushima, M. Iio, K. Hirata, and M. Yaoamoto, "Popularity-Based Content Cache Management for in-Network Caching," in 2019 International Conference on Information Networking (ICOIN), Kuala Lumpur, Malaysia: IEEE, Jan. 2019, pp. 411–413. doi: 10.1109/ICOIN.2019.8718180.

[41] R. Hou, L. Zhang, T. Wu, T. Mao, and J. Luo, "Bloom-filter-based request node collaboration caching for named data networking," Cluster Comput, vol. 22, no. S3, pp. 6681–6692, May 2019, doi: 10.1007/s10586-018-2403-9.

[42] P. Singh, R. Kumar, S. Kannaujia, and N. Sarma, "Adaptive Replacement Cache Policy in Named Data Networking," in 2021 International Conference on Intelligent Technologies (CONIT), Hubli, India: IEEE, June 2021, pp. 1–5. doi: 10.1109/CONIT51480.2021.9498489

[43] Y. An and X. Luo, "An In-Network Caching Scheme Based on Energy Efficiency for Content-Centric Networks," IEEE Access, vol. 6, pp. 20184–20194, 2018, doi: 10.1109/ACCESS.2018.2823722.

[44] A. Silva, I. Araujo, N. Linder, and A. Klautau, "Name Popularity Algorithm: A Cache Replacement Strategy for NDN Networks," JCIS, vol. 34, no. 2, pp. 206–214, 2019, doi: 10.14209/jcis.2019.22.

[45] N. A. Nasir and S.-H. Jeong, "Fast Content Delivery Using a Testbed-Based Information-Centric Network," IEEE Access, vol. 9, pp. 101600–101613, 2021, doi: 10.1109/ACCESS.2021.3096042.

[46] K. Katsaros, G. Xylomenos, and G. C. Polyzos, "A Hybrid Overlay Multicast and Caching Scheme for Information-Centric Networking," in 2010 INFOCOM IEEE Conference on Computer Communications Workshops, San Diego, CA, USA: IEEE, Mar. 2010, pp. 1–6. doi: 10.1109/INFCOMW.2010.5466659.

[47] H. Khelifi, S. Luo, B. Nour, and H. Moungla, "A QoS-Aware Cache Replacement Policy for Vehicular Named Data Networks," in 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA: IEEE, Dec. 2019, pp. 1–6. doi: 10.1109/GLOBECOM38437.2019.9013461.

[48] M. D. Ong, M. Chen, T. Taleb, X. Wang, and V. C. M. Leung, "FGPC: fine-grained popularity-based caching design for content centric

Correspondence to: Sushil Kumar Bagi, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: sushil.23183963@mygyanvihar.com

Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.12 Issue 1 Page No 65-94

networking," in Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems, Montreal QC Canada: ACM, Sept. 2014, pp. 295–302. doi: 10.1145/2641798.2641837.

[49] S. Mishra, V. K. Jain, K. Gyoda, and S. Jain, "A novel content eviction strategy to retain vital contents in NDN-IoT networks," Wireless Netw, vol. 31, no. 3, pp. 2327–2349, Mar. 2025, doi: 10.1007/s11276-024-03887-y.

[50] M. Chand, "A Comparative Survey On Different Caching Mechanisms In Named Data Networking (NDN) Architecture," p. 824601 Bytes, 2019, doi: 10.6084/M9.FIGSHARE.7993490.V1.

[51] R. M. Negara and N. Rachmana Syambas, "Caching and Machine Learning Integration Methods on Named Data Network: a Survey," in 2020 14th International Conference on Telecommunication Systems, Services, and Applications (TSSA, Bandung, Indonesia: IEEE, Nov. 2020, pp. 1–6. doi: 10.1109/TSSA51342.2020.9310811.

[52] J. Ren et al., "MAGIC: A distributed MAx-Gain In-network Caching strategy in information-centric networks," in 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada: IEEE, Apr. 2014, pp. 470–475. doi: 10.1109/INFCOMW.2014.6849277.