future internet

MDPI

Review

A Review on Blockchain Sharding for Improving Scalability

Mahran Morsidi [1,*], Sharul Tajuddin [1], S. H. Shah Newaz [1,*], Ravi Kumar Patchmuthu [1] and Gyu Myoung Lee [2]

[1] School of Computing and Informatics, Universiti Teknologi Brunei, Jalan Tungku Link, Gadong BE 1410, Brunei; sharul.tajuddin@utb.edu.bn (S.T.); ravi.patchmuthu@utb.edu.bn (R.K.P.)
[2] School of Computer Science and Mathematics, Liverpool John Moores University, Liverpool L3 3AF, UK; g.m.lee@ljmu.ac.uk
* Correspondence: mahran.morsidi@gmail.com (M.M.); shah.newaz@utb.edu.bn (S.H.S.N.)

Abstract

Blockchain technology, originally designed as a secure and immutable ledger, has expanded its applications across various domains. However, its scalability remains a fundamental bottleneck, limiting throughput, specifically Transactions Per Second (TPS) and increasing confirmation latency. Among the many proposed solutions, sharding has emerged as a promising Layer 1 approach by partitioning blockchain networks into smaller, parallelized components, significantly enhancing processing efficiency while maintaining decentralization and security. In this paper, we have conducted a systematic literature review, resulting in a comprehensive review of sharding. We provide a detailed comparative analysis of various sharding approaches and emerging AI-assisted sharding approaches, assessing their effectiveness in improving TPS and reducing latency. Notably, our review is the first to incorporate and examine the standardization efforts of the ITU-T and ETSI, with a particular focus on activities related to blockchain sharding. Integrating these standardization activities allows us to bridge the gap between academic research and practical standardization in blockchain sharding, thereby enhancing the relevance and applicability of our review. Additionally, we highlight the existing research gaps, discuss critical challenges such as security risks and inter-shard communication inefficiencies, and provide insightful future research directions. Our work serves as a foundational reference for researchers and practitioners aiming to optimize blockchain scalability through sharding, contributing to the development of more efficient, secure, and high-performance decentralized networks. Our comparative synthesis further highlights that while Bitcoin and Ethereum remain limited to 7–15 TPS with long confirmation delays, sharding-based systems such as Elastico and OmniLedger have reported significant throughput improvements, demonstrating sharding's clear advantage over traditional Layer 1 enhancements. In contrast to other state-of-the-art scalability techniques such as block size modification, consensus optimization, and DAG-based architectures, sharding consistently achieves higher transaction throughput and lower latency, indicating its position as one of the most effective Layer 1 solutions for improving blockchain scalability.

Keywords: blockchain scalability; blockchain sharding; Layer 1 sharding; scalability; sharding; artificial intelligence

Academic Editors: Klitos Christodoulou and Massimiliano Garda

Received: 13 August 2025
Revised: 12 October 2025
Accepted: 16 October 2025
Published: 21 October 2025

Citation: Morsidi, M.; Tajuddin, S.; Newaz, S.H.S.; Patchmuthu, R.K.; Lee, G.M. A Review on Blockchain Sharding for Improving Scalability. *Future Internet* **2025**, *17*, 481. https://doi.org/10.3390/fi17100481

1. Introduction

Blockchain is a series of blocks of information securely linked to each other that grows chronologically and has timestamps with dates or times. It can be divided into two types: public blockchain and consortium blockchain. A public blockchain is one where

the network is accessible to everyone, while a consortium blockchain is restricted to a group of organizations within the network. But this technology was mentioned in 1991 by Stuart Haber and W. Scott Stornetta, who introduced a digital document system that had a time-stamping feature; thus, it is not a new idea [1]. It gained popularity with the rise of Bitcoin as a cryptocurrency and has since expanded to incorporate smart contracts, which are self-executing agreements where code automatically enforces predefined conditions.

When smart contracts are incorporated, they help execute code and digitize contracts that may be applied to smart city services like voting [2], bills [3], identity management [4–7], and governance [8,9], enhancing security through immutability, integrity, and auditability [10].

Despite what blockchain may offer, blockchain faces scalability challenges, a critical issue highlighted in literature across domains like Smart City [9,11–13], Software-defined Networking [14–16], Identity Management [10,17,18], and Metaverse [19]. Scalability concerns arise with the increase in the number of nodes and transactions in a blockchain network, impacting performance, which is generally measured as Transactions Per Second (TPS) and latency (TPS measures the number of transactions a blockchain can process in a second, while latency refers to the delay before a transaction is confirmed and recorded). In the mainstream blockchain technology, Bitcoin processes 7 TPS with a 10 min confirmation time [20], whereas Ethereum improves on this with 15 TPS and a 19 s confirmation time [21]. However, these figures are low when compared to a network platform like VISA, which can perform up to 24,000 TPS with instant confirmation [20]. In order to make blockchain a widely adopted technology, the blockchain performance needs to be on par with VISA's by improving its scalability [22]. Compared with these baseline figures, early static sharding systems such as Elastico [23] demonstrated improvements beyond the performance of Bitcoin and Ethereum, and subsequent layered and dynamic sharding designs [24–26] continued this trend. These comparative results indicate that sharding achieves higher throughput and lower latency than consensus modifications or block size adjustments, reinforcing its position as a state-of-the-art Layer 1 scalability technique. Therefore, improving blockchain scalability has been an interesting research avenue for many researchers.

Addressing scalability involves examining blockchain's three-layer architecture, with each layer offering potential scalability enhancement opportunities [20,22,27,28]. When blockchain processes are separated into different layers as depicted in Figure 1, it is easier to understand how blockchain architecture or structure interacts with different elements. In blockchain, Layer 0 underpins data transportation [29], Layer 1 constitutes the blockchain architecture itself (on-chain) [27], and Layer 2 operates externally to relieve the on-chain system (off-chain) [27]. Among all the layers, the room for performance improvement is fundamental in Layer 1 [30–34].

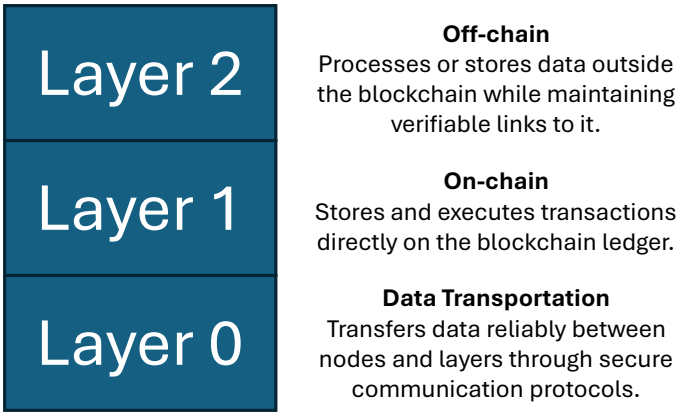

Figure 1. Blockchain layers.

Various approaches exist in Layer 1 to enhance the scalability performance, such as introducing new consensus mechanisms (e.g., Proof-of-Authority (PoA) [35], Proof-of-Burn [35] and Bitcoin-NG [33]), manipulation of block size (e.g., [30,36–38]), applying Directed Acyclic Graph (DAG) (e.g., [39,40]) and adopting sharding approaches (e.g., [41–44]). Consensus allows for transaction validation by nodes, which is related to scalability via adjustments in block size. However, altering consensus mechanisms and block sizes might lack long-term viability and adaptability in a blockchain network [29]. Therefore, implementing DAG and sharding provide more versatile alternatives for handling blockchain transactions. DAG necessitates a comprehensive redesign of blockchain architecture, enabling concurrent block generation by allowing multiple vertices to connect to a previous one, thus increasing transaction capacity [45,46]. In contrast, sharding segments transactions into "shards" for parallel processing, reducing individual workload. This makes sharding a compelling Layer 1 approach [22,47], potentially better addressing the blockchain trilemma of security, scalability, and decentralization than other Layer 1-based approaches for scalability improvement in blockchain. Alongside these advancements, recent studies have explored the incorporation of Artificial Intelligence (AI) to assist in blockchain sharding processes, such as shard management, node coordination, and consensus configuration (e.g., [48,49]). The timeline infographic presented in Figure 2 shows the evolution of sharding technology starting from 2016. Standardization activities have also advanced in parallel. For example, Recommendation ITU-T F.751.19 [50] defines a framework and technical requirements for sharded distributed ledger technology, specifying key components and requirements for scalability and security.

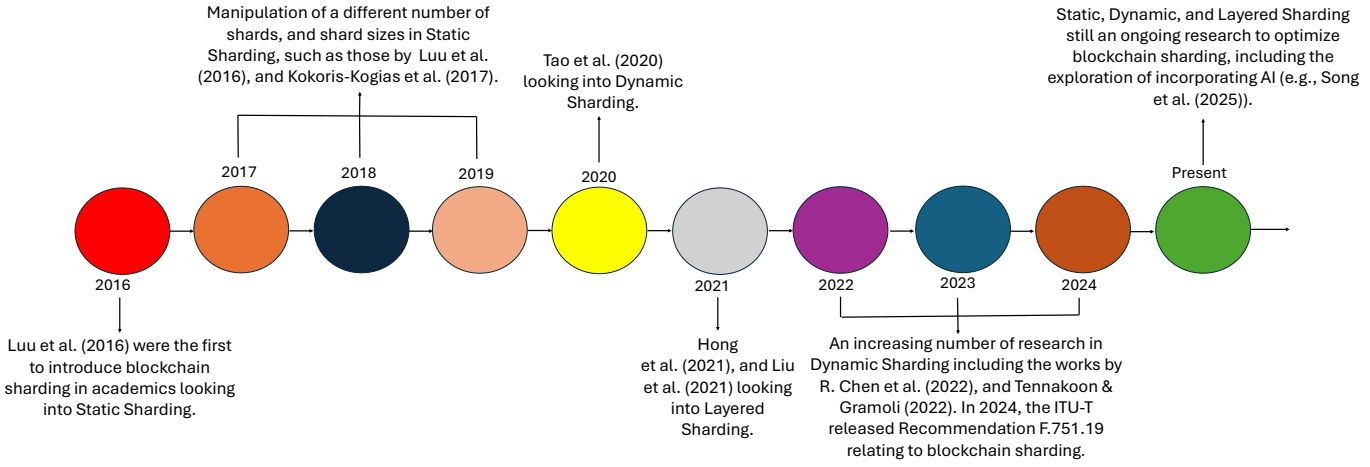

Figure 2. Timeline of blockchain sharding development from Static (2016) to Dynamic (2020) and Layered (2021) approaches, alongside ITU-T's 2024 standardization framework and recent AI integration. Sources: Luu et al. [23]; Hong et al. [24]; Tao et al. [25]; ITU-T Recommendation F.751.19 [50]; Kokoris-Kogias et al. [51]; R. Chen et al. [52]; Tennakoon and Gramoli [53]; Liu et al. [54]; Song et al. [55].

The structure of the paper, illustrated in Figure 3, offers a high-level overview of the organization of this review. Section 2 reviews prior studies on blockchain scalability, ranging from broad examinations of scalability challenges to surveys focused specifically on sharding, and identifies gaps that frame the rationale for our contribution. Section 3 outlines the specific contributions of this work. Section 4 describes in detail the systematic literature review methodology adopted in this study. Section 5 provides an overview of sharding and includes a detailed explanation of various sharding techniques. Section 6 highlights the existing standardization efforts exist in blockchain sharding introduced by major standardization bodies. Section 7 details the key features of these sharding techniques. Section 8 is dedicated to discussing these features further, while Section 9

explores the lessons learned, the open research issues and future directions. The paper concludes with Section 10, summarizing the work.

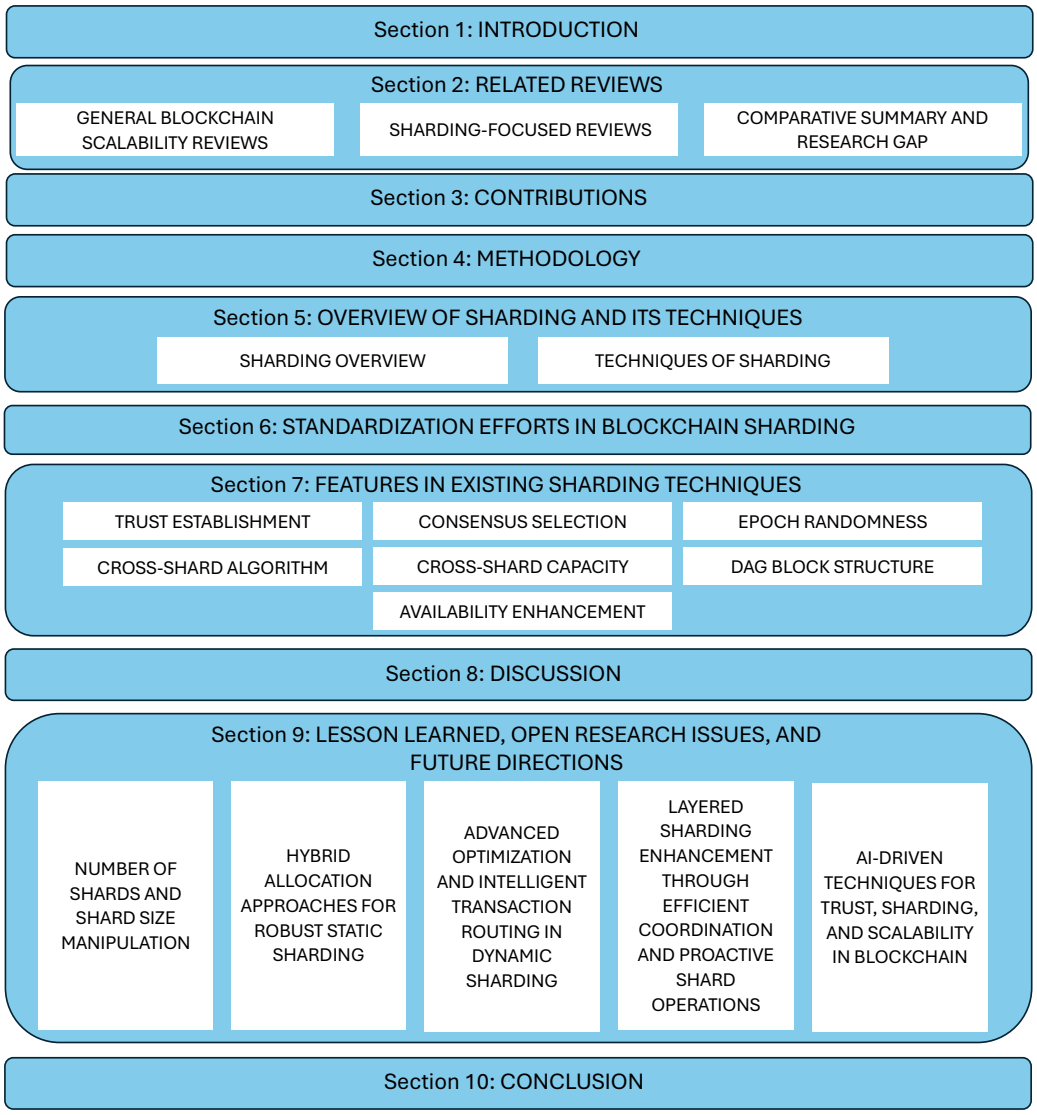

Figure 3. High-level structure of our review.

2. Related Reviews

The rapid growth of blockchain applications has motivated numerous review studies that examine scalability challenges and solutions across different layers of the blockchain stack. However, these studies vary in scope and depth, with some focusing broadly on general scalability mechanisms while others target specific techniques such as sharding. A structured review of these works is therefore essential to situate our contribution within the broader research landscape, identify what prior surveys have addressed, and highlight the research gaps that remain unresolved. In the following subsections, we first summarize general blockchain scalability reviews and then narrow the focus to sharding-specific surveys, before synthesizing the comparative gaps that provide the rationale for our contributions.

2.1. General Blockchain Scalability Reviews

Although numerous surveys or reviews have examined blockchain scalability across different architectural layers, none of these focus primarily on sharding techniques and their classifications. For example, the reviews on general blockchain scalability are de-

livered in [20,22,27–29]. Nasir et al. in [20] provide an extensive review of blockchain scalability, addressing enhancements across various layers and discussing issues like block size limitations and consensus mechanism constraints. Nevertheless, only few works on sharding are explored in [20].

The survey presented in [22] is oriented more on the classification of existing scalability solutions, evaluating the trade-offs between security and performance, particularly with a focus on sharding. Hafid et al. in [22] offer a comparative analysis of various protocols, emphasizing metrics such as TPS and latency. However, the discussion remains conceptual and lacks in-depth empirical validation in [22]. Zhou et al. in [27] focus on potential scalability solutions classified within the three blockchain layers: Layer 2 (off-chain approaches), Layer 1 (on-chain approaches including sharding), and Layer 0 (data propagation techniques). They offer a detailed comparison of these solutions, noting their effects on TPS and latency, while it does not reflect more recent developments.

The survey in [29] shifts focus to scalability challenges in time-sensitive applications, particularly examining Layer 1 approaches such as increasing block size and modifying consensus mechanisms. However, the work in [29] lacks the critical depth on technical and implementation challenges. Meanwhile, Sanka and Cheung in [28] introduce a five-layer conceptual model consisting of the platform, network, consensus, data, and application layers. Their survey in [28] provides a comprehensive analysis of the performance and scalability of existing Layer 1 blockchain approaches. Although one analysis provides a classification based on write, read, and storage performance, it lacks empirical validation for its claims. This reflects a broader gap in the literature, where surveys often address general blockchain scalability without offering a thorough classification or detailed discussion of specific sharding techniques.

2.2. Sharding-Focused Reviews

There exist few survey works focusing on sharding techniques introduced in the existing literature [47,56–59]. Li et al. in [47] review state-of-the-art sharding-based blockchains for both permissioned and permissionless networks. Their work covers Static, Dynamic, and Layered Sharding approaches, addressing key components, performance, and security aspects. It also outlines common attack surfaces and countermeasures. However, the analysis remains literature-based without experimental validation.

The surveys in [56,57] further explore sharding mechanisms and emphasize their importance for blockchain scalability. For instance, Yu et al. in [56] provide a systematic overview of major sharding mechanisms, highlighting both efficiency and security limitations. Their detailed comparison and evaluation offer valuable insights into the features, limitations, and theoretical TPS upper bounds of these mechanisms, serving as critical benchmarks for understanding the scalability potential of blockchain technology.

Huang et al. in [57] review various sharding approaches, identifying factors to enhance blockchain performance that provides a detailed review of sharding approaches, emphasizing their role in achieving horizontal scaling by dividing the blockchain into shards for increased TPS. They have thoroughly highlighted the specific challenges, features, and limitations of sharding through a comprehensive analysis. However, both surveys in [56,57] fall short in discussing practical implementation details and offer only minimal coverage of evolving trends.

Meanwhile, Hashim et al. in [59] review various consensus mechanisms for sharding, suggesting improvements in areas such as cross-shard communication and shard formation. The review meticulously examines the scalability challenges in blockchain technology and the approaches offered by database sharding to mitigate these issues. It includes an in-depth analysis of various sharding consensus mechanisms and provides

valuable insights into current strategies. Furthermore, they outline significant unresolved challenges, such as effective shard formation, node assignment to shards, any possibilities of shard takeovers, and determination of the optimal number of shards in the blockchain network, yet their review in [59] does not critically assess more recent methodologies or technological advancements.

Then, the survey in [58] breaks down sharding into functional components and presents a modular architecture for sharding approaches. While the survey offers valuable insights into the underlying models and components by detailing key functions such as Node Selection, Epoch Randomness, Node Assignment, Intra-shard Consensus, Cross-shard Transaction Processing, Shard Reconfiguration, and Motivation Mechanisms and by emphasizing their integration into a cohesive approach, it does not provide a comparative evaluation of practical implementations or real-world approaches.

While recent studies have concentrated on blockchain sharding technologies, their analyses offer specific insights. For instance, Xiao et al. in [60] introduce a categorization of sharding methods into network, transaction, and state sharding, and discusses intra-slice and inter-slice consensus mechanisms. However, their analysis lacks comprehensive comparative metrics, benchmarks, and detailed application insights. Similarly, Zhang et al. in [61] analyze classical sharding mechanisms by thoroughly examining key components such as shard formation, reshuffle processes, intra-shard consensus, and cross-shard communication. While highlighting the advantages and limitations of existing scalability approaches, this study does not offer detailed empirical performance evaluations and practical guidance. In contrast, Yang et al. in [62] provides a comprehensive overview, integrating theoretical perspectives with experimental evaluations. Although their analysis captures current advancements and future trends across diverse blockchain ecosystems, it lacks a structured comparative framework for deeper analysis.

The survey in [63] offers a more structured and comprehensive review by systematically classifying and evaluating existing sharding approaches for both permissionless and permissioned blockchains. It introduces a robust evaluation framework based on scalability (latency, TPS, communication overhead), applicability (e.g., Trusted Execution Environment dependence, smart contract compatibility, universality, privacy), and reliability (randomness and fault tolerance). The study reviews existing sharding approaches and presents its findings through detailed classification tables and critical thematic analysis. Unlike prior works, it provides a unified comparative structure, though it remains literature-based and does not aggregate or analyze quantitative performance data from simulations or testbeds.

2.3. Comparative Summary and Research Gap

To summarize, none of the surveys described above specifically focuses on a comprehensive classification of sharding techniques, as one can observe from the comparison between our work and previous surveys presented in Table 1. While Liu et al. in [58] introduce some features of sharding, it focuses on modular architecture rather than classifying sharding methods comprehensively. Our review fills this gap by providing a structured classification of sharding techniques into Static, Dynamic, and Layered approaches, along with a detailed analysis of features and implications for blockchain scalability. In addition, it is the first to incorporate and examine standardization efforts, such as those from ITU-T (e.g., [50]), and ETSI (e.g., [64]), offering a broader perspective that has not been considered in previous surveys and reviews. Furthermore, our review also identifies AI-assisted sharding approaches and maps them to relevant features such as Consensus Selection and Epoch Randomness. Most of these approaches are discussed in context but are not included within our sharding classifications, as they often span multiple functional aspects and remain in the early stages of development.

Table 1. Comparison of the contributions of our review with previous reviews (✓ represents a criterion is met). Here, BC and Stand. indicate Blockchain and Standardization, respectively.

Reference, Year	Review Scopes						
	BC Performance Aspects	Existing Works on BC Sharding Techniques			AI-Assisted Sharding Approaches	Stand. Activities on BC Sharding	Remarks
		Static	Dynamic	Layered			
Hafid et al. [22], (2020)	✓						Focuses on performance analysis with comparative metrics (e.g., TPS). However, the discussion remains largely conceptual and lacks in-depth empirical validation.
Zhou et al. [27], (2020)	✓						Provides a broad overview of BC performance and scalability solutions up to 2020, but the review is partially outdated and does not reflect the recent developments.
Yu et al. [56], (2020)	✓	✓					Presents a systematic analysis of performance and static sharding as of 2020, yet the review does not address subsequent advancements or practical implementation details.
Huang et al. [57], (2021)	✓	✓					Offers a detailed conceptual review of performance and static sharding, but is limited to prevailing techniques up to 2021, with minimal focus on evolving trends.
Khan et al. [29], (2021)	✓						Emphasizes performance analysis in time-sensitive blockchain applications. However, the evaluation lacks critical depth on technical and implementation challenges.
Sanka and Cheung [28], (2021)	✓						Analyzes performance using a conceptual multi-layer model, but does not critically examine limitations or support claims with empirical data.
Liu et al. [58], (2021)	✓	✓					Discusses performance and modular architectures for sharding, yet offers limited benchmarking or comparative evaluation of practical approaches.
Li et al. [47], (2023)	✓	✓	✓	✓			Includes discussion of Static, Dynamic, and Layered Sharding approaches along with performance and security considerations, yet the analysis is qualitative and not supported by experimental evaluation.
Nasir et al. [20], (2022)	✓						Surveys performance-related scalability challenges broadly, though the discussion remains generic and lacks recent advancements.
Hashim et al. [59], (2023)	✓	✓	✓				Provides analysis of performance and consensus mechanisms, identifying unresolved challenges,but the discussion is mostly conceptual and lacks evaluation of recent methodologies.
Xiao et al. [60], (2023)		✓					Describes Static Sharding methods. The discussion is introductory and does not offer comparative benchmarks or detailed application insights.
Liu et al. [63], (2023)	✓		✓	✓			Presents a structured literature-based evaluation of performance and sharding schemes. However, the review does not include quantitative results from real-world deployments or simulations.

Table 1. *Cont.*

Reference, Year	Review Scopes						
	BC Performance Aspects	Existing Works on BC Sharding Techniques			AI-Assisted Sharding Approaches	Stand. Activities on BC Sharding	Remarks
		Static	Dynamic	Layered			
Zhang et al. [61], (2024)	✓	✓		✓			Delivers an in-depth review of performance and key sharding components, but omits empirical performance evaluation and detailed practical guidance.
Yang et al. [62], (2025)	✓	✓	✓	✓			Combines theoretical and experimental perspectives on performance, yet lacks a structured comparative framework for deeper analysis.
Our review	✓	✓	✓	✓	✓	✓	Provides complete coverage, including performance, Static, Dynamic, and Layered Sharding classifications, and identifies AI-assisted sharding approaches, offering a holistic and up-to-date synthesis along with the inclusion of stand. activities.

While Table 1 summarizes the scope of prior surveys across performance, sharding types, AI-assisted approaches, and standardization activities, our review extends the comparison dimensions by integrating both technical features (e.g., Trust Establishment, Consensus Selection, Epoch Randomness, Cross-shard Algorithm, Cross-shard Capacity, DAG, and Availability Enhancement) and practical considerations such as standardization. This multidimensional approach allows for a more holistic evaluation framework compared to earlier works, capturing both the technical and practical dimensions of blockchain scalability and sharding techniques.

3. Contributions

The previous section reveals that, to date, there is no comprehensive review focusing specifically on sharding classifications within blockchain technologies. Thus, we offer a unique and thorough review of the research centered on advancements in sharding methods, an area that has not yet been reviewed in detail. Our review specifically addresses the gaps in existing studies, aiming to refine blockchain methodologies through an in-depth exploration of sharding. The main contributions of our review are outlined as follows:

- Our review introduces a structured classification that highlights three distinct sharding techniques, primarily concentrating on Layer 1. This classification provides a foundation for understanding how each sharding technique enhances blockchain scalability within the context of secure and decentralized shard-based blockchain systems.
- To provide readers with a broad overview of standardization efforts in blockchain sharding, we review the initiatives introduced by major standardization bodies (e.g., ITU-T, ISO, IEEE, and ETSI) and key industry consortia. We also highlight the operational guidelines and technical frameworks proposed by these organizations for sharding.
- We conduct a systematic literature review, through which we delve into various sharding techniques to assess their characteristics and limitations. This includes a thorough examination of key features such as Trust Establishment, Consensus Selection, Epoch Randomness, Cross-shard Algorithm, Cross-shard Capacity, DAG Block Structure, and Availability Enhancement, along with recent developments in AI-assisted sharding approaches, considered within these key feature areas to provide a comprehensive understanding of each technique and its underlying mechanisms.

- Through a comparative analysis, our review focuses specifically on the performance outcomes of these sharding techniques, with a particular emphasis on their ability to increase TPS and reduce transaction latency. This review aims to offer insights into the effectiveness and efficiency of the different sharding approaches in improving blockchain performance.
- Based on this review, we derive important insights, identify the major challenges and provide important future research directions.

4. Methodology

Our review addresses the following questions by concentrating on blockchain scalability concerns. As illustrated in the process in Figure 4, this focus allows for the identification of specific issues from past literature that form the foundation of this review.

To ensure rigor, reproducibility, and comprehensiveness, we adopted a systematic literature review methodology, following established guidelines in software engineering and information systems research [65]. A systematic approach was chosen because blockchain sharding is a relatively new research area (first introduced in 2016 with Elastico [23]), and studies are dispersed across multiple venues. This methodology allows us to minimize bias, justify inclusion/exclusion decisions, and provide a transparent, replicable process for identifying the most relevant works.

- What are the fundamental obstacles to blockchain scalability, and what strategies have researchers developed to overcome them?
- What are the methods or techniques used in blockchain sharding along with its advantages and disadvantages?
- How do different sharding methods adapt to the existing challenges?

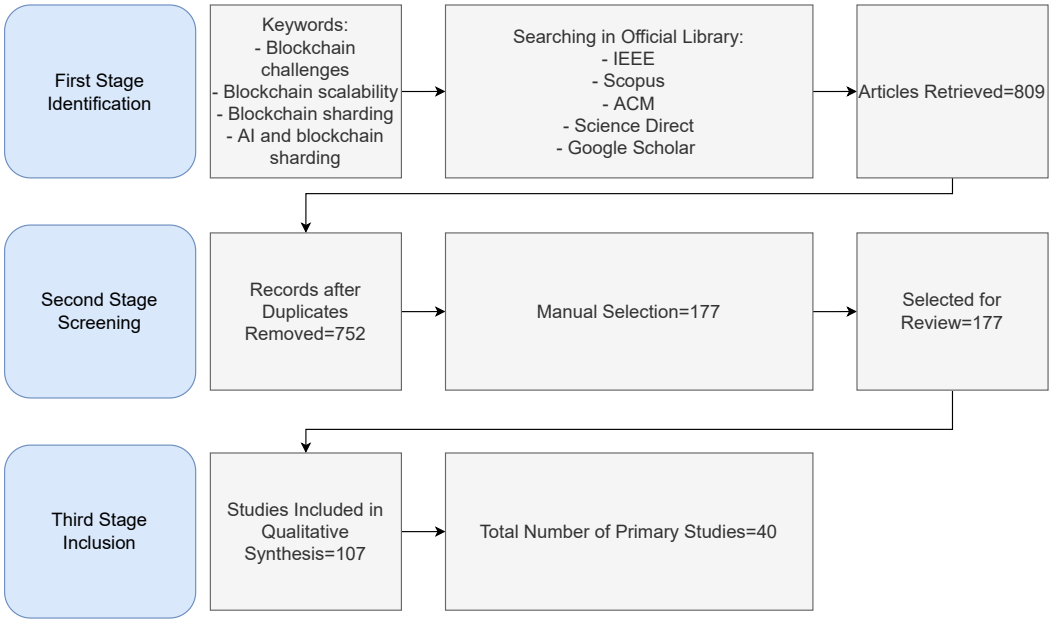

Figure 4. Workflow of the systematic mapping process for blockchain sharding review.

In the first phase of reviewing articles, sources were obtained from IEEE, Scopus, ACM, ScienceDirect, and Google Scholar. Keywords were used for the search to extract the relevant articles: “blockchain challenges”, “blockchain scalability”, “blockchain sharding” and “AI and blockchain sharding“. After conducting a consolidated search using all keywords, a total of 809 articles were identified. These include journal and conference papers. As blockchain sharding research only started in 2016, the number of related publications

remains relatively small. Moreover, to further filter the articles, these two criteria need to be taken into account which are:

- Exclusion: Papers published before 2016, uncited works, duplicates, abstracts only, and non-English publications.
- Inclusion: English language articles published after 2016, with citations, and directly addressing the research questions.

Publications were chosen for this review based on their relevance to the research questions discussed earlier. Most pertinent publications are categorized based on the title, authors, year of publication, and proposed ideas. Followed by the manual filtering process where each of the chosen publications underwent a detailed analysis.

In the second phase of our methodology, 752 publications remained after duplicate removal. From these, 177 distinct papers were selected for initial review. These publications focused on the defined keywords, including blockchain scalability, sharding, and challenges, including the application of AI in blockchain sharding, as well as broader topics like cryptocurrency and smart contracts.

The third phase involved a refined relevance assessment based on predefined inclusion criteria, which focused on methodological depth, technical contribution, applicability to sharding-based scalability, and novelty. This process resulted in the selection of 107 papers that were deemed thematically and methodologically relevant. These comprised surveys, review articles, conceptual frameworks, system models, and foundational theoretical studies.

From the 107 papers, a final set of 40 primary studies was chosen for in-depth analysis. The selection was guided by factors such as citation impact, methodological rigor, system implementation clarity, and representativeness of different sharding techniques. This final set consisted of 6 survey articles, 5 review papers, 2 standardization papers, 8 papers on AI-assisted sharding, and 19 papers describing sharding-based blockchain projects.

To strengthen the reliability and reproducibility of our synthesis, we applied a consistent set of evaluation dimensions across all selected studies, focusing primarily on throughput (TPS), latency, and cross-shard efficiency. Although the underlying experiments varied, for example AWS EC2 simulations in Kronos [66], large-scale node experiments in DL-Chain [67], and protocol-specific testbeds in DYNASHARD [68], our classification framework (Figure 3) ensured that results were interpreted within these shared benchmarks. This uniform lens enhances comparability across heterogeneous systems. Nevertheless, differences in simulation environments, parameter choices, and node configurations can introduce variability that may limit strict reproducibility across versions of framework components. Highlighting these divergences is essential to provide a transparent assessment of robustness and to caution against overgeneralization of performance claims.

5. Overview of Sharding and Its Techniques

In this section, we provide an overview of sharding in blockchain technology and outline its three primary techniques: Static, Dynamic, and Layered Sharding. Each technique offers distinct approaches to partitioning blockchain data, which is important for enhancing the scalability and efficiency of transaction processing across different blockchain network conditions.

5.1. Sharding Overview

The concept of sharding, originally derived from database management, involves dividing a larger database into numerous smaller datasets across various nodes [56]. In the context of blockchain, sharding was first introduced in [23], subsequently leading to the development of various sharding approaches (e.g., [24,51,52]).

A blockchain network is divided into multiple shards (say, n number of shards). Each shard consists of a number of nodes (say, m). This defines the shard size (i.e., the number of nodes within each shard determines the size of a shard). Then, the total number of nodes in the blockchain network is $n \times m$. Figure 5 illustrates the concept of sharding in a blockchain network. Each shard maintains its own independent blockchain, processing its own sequence of blocks (e.g., Block 1 to Block x of Shard 1 in Figure 5) and transactions without any interactions with other shards (i.e., transactions are processed in parallel within their respective shards unless a cross-shard communication mechanism is implemented).

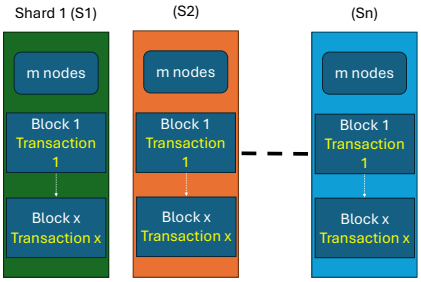

Figure 5. Sharding architecture illustrates a sharding approach with n shards (Sn specifically denotes the nth shard), each containing m nodes. It demonstrates parallel transaction processing, where each shard independently processes transactions within its own sequence of blocks — S1 Block 1 (Shard 1), S2 Block 1 (Shard 2), and Sn Block 1 (Shard n). Block x and Transaction x represent the respective block number and transaction number within each shard.

When a transaction involves multiple shards communicating with each other, a cross-shard transaction occurs. Figure 6 illustrates an example process of cross-shard transactions involving shard 1 and shard 2, where transaction 2 in shard 1 requires communication with transaction 4 in shard 2 to complete the transaction process. Cross-shard transactions are more complex because they cause higher latency due to communication between shards, add coordination overhead to manage transaction steps across shards, and create security challenges in maintaining consistent and secure data across multiple shards.

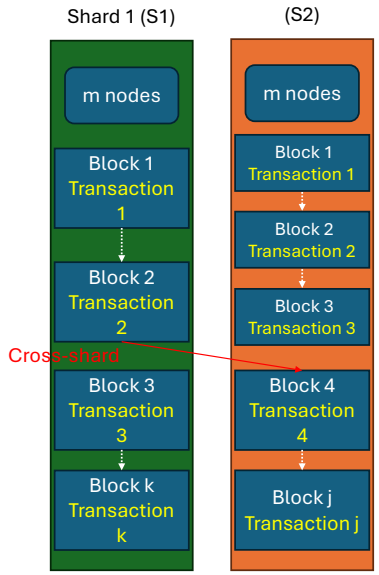

Figure 6. Cross-shard transaction involves interaction between different shards, such as a smart contract transaction from Node 1 in Shard 1 to Node 2 in Shard 2 as shown here. If both accounts are in the same shard, it is a normal sharding transaction. Else, Shard 1 sends transaction 2 to Shard 2, and Shard 2 verifies and records it as transaction 4 to complete.

The discussion on sharding raises questions about the number of shards, shard size, and their optimal values, which appear to be inconsistent across different usage contexts. This has led to the evolution of sharding techniques, including Static (e.g., Elastico [23]), Dynamic (e.g., User Distribution [69]), and Layered Sharding (e.g., Pyramid [24]).

5.2. Techniques of Sharding

This subsection provides a brief overview of each sharding technique.

5.2.1. Static Sharding

Static Sharding is a technique that determines the shard size and number of shards based on the number of nodes that will be allocated equally across different shards. Various predetermined numbers of shards and shard sizes have been tested to determine the optimal configuration that maximize transaction processing speed and minimize latency. However, conclusions about the optimal number of shards and shard sizes to enhance sharding effectiveness are still inconclusive. Static Sharding approaches in public blockchains include Elastico [23], OmniLedger [51], RapidChain [70], Monoxide [42], Chainspace [43], DL-Chain [67], and Kronos [66]. Notably, Chainspace [43] integrates smart contracts, while Meepo [71] represents a consortium blockchain within Static Sharding.

5.2.2. Dynamic Sharding

Compared to Static Sharding, Dynamic Sharding facilitates dynamic resource sharing to cope with demand. That may allocate the nodes across different shards and dynamically change the shard sizes and number of shards. Due to the weakness of cross-shard communication, such as the one discussed in [72], which could exhaust the network and reduce the scalability benefits of sharding, there is a high cross-shard communication overhead. An alternative Dynamic Sharding approach is investigated, as mentioned in the sharding overview section (see Section 5.1). Dynamic Sharding is one of the alternatives to increase the performance (TPS) of blockchain network that supports smart contracts as it may help reduce the overhead introduced by cross-shard communication overhead as discussed in [25]. There are eight different existing research efforts in Dynamic Sharding to the best of our knowledge, particularly in public blockchain: On Sharding Open Blockchains [25], User Distribution [69], Dynamic Blockchain Sharding [53], Effective Sharding Consensus Mechanism [52], DYNASHARD [68], AEROChain [55], SkyChain [73] and one consortium blockchain: Dynamic Sharding Protocol Design [72].

5.2.3. Layered Sharding

Since all the approaches in sharding employ complete sharding, this significantly adds additional overhead in ensuring cross-shard transaction atomicity and consistency, which in turn negatively affects the sharding performance [24]. To overcome this challenge, Layered Sharding is introduced. In this sharding technique, when a shard(s) is occupied with transactions, any subsequent transactions of the shard are assigned to a shard having sufficient processing capability, thereby avoiding the communications overhead that cross-sharding transaction imposes. Three notable works have investigated the Layered Sharding technique: Pyramid [24], OverlapShard [54] and SPRING [74]. Unlike complete sharding, in a Layered Sharding blockchain, the shards can validate and execute cross-shard transactions directly and efficiently. Though these sharding techniques leverage blockchain employing cross-shard transactions, there is a big room to further improve the blockchain performance by further improving cross-shard techniques.

6. Standardization Efforts in Blockchain Sharding

The evolution of blockchain technology has led to increased efforts by international and industry standards organizations to address scalability challenges, particularly through the standardization of sharding mechanisms. Organizations including ITU-T, ETSI, ISO, IEEE, and several national bodies have each addressed these challenges from distinct perspectives. While ISO, IEEE, and W3C primarily reference sharding in broad architectural frameworks or performance metrics, it is ITU-T and ETSI that have produced the most concrete and operational standards for sharding, with ETSI also pioneering guidelines on AI-assisted sharding.

ITU-T Recommendation F.751.19 [50] specifies a structured framework and technical requirements for sharded blockchains, delineating an "on-chain sharding" layer composed of five core functional components: shard management, sharding strategy management, intra-shard processing, cross-shard distribution, and cross-shard coordination. Each component is defined with mandatory capabilities, and for instance, shard management is required to oversee the lifecycle of shards and maintain data integrity during auto-scaling and migration, while sharding strategy management is responsible for mapping transactions to shards to facilitate near-linear scalability. Nevertheless, the recommendation in [50] primarily describes the objectives to be achieved and leaves substantial implementation challenges to system developers. These challenges include the design of flexible strategies and the management of distributed deadlocks. Furthermore, F.751.19 [50] mandates features to address critical risks, such as random validator allocation and dynamic reallocation, but does not prescribe the technical specifics necessary to ensure robust implementation. The voluntary and high-level nature of the recommendation in [50] may therefore result in inconsistent or suboptimal deployments, which highlights the ongoing need for more prescriptive standards and deeper technical analysis as sharding adoption matures.

In parallel, ETSI through its Industry Specification Group on Permissioned Distributed Ledger, has introduced explicit operational guidance on AI-driven sharding. ETSI GR PDL-032 V1.1.1 (2025-04) in [64], "Artificial Intelligence for PDL," recommends the use of predictive models such as Long Short-Term Memory neural networks to forecast transaction loads and dynamically adjust shard boundaries. The report in [64] documents reductions of up to 23% in cross-shard interactions compared to Static Sharding. The specification further details adaptive shard allocation, intelligent cross-shard transaction management, and AI-based optimization of key DLT functions including security, consensus, and resource allocation. While this group report marks a significant advance in standardizing AI integration with blockchain sharding, its guidelines remain advisory, highlighting the ongoing need for more prescriptive, implementation-oriented standards to support broader industry adoption.

Other standardization organizations, such as ISO, particularly ISO/TR 24332:2025 [75], the IEEE, which develops blockchain and distributed ledger technology standards through the IEEE Standards Association [76], and the W3C with its Decentralized Identifier Resolution (DID Resolution) v0.3 [77], primarily address blockchain scalability and interoperability through high-level reference documents. For example, ISO/TR 24332:2025 [75] mentions sharding within its frameworks but does not define specific operational models. IEEE provides a broad set of standards covering interoperability, the Internet of Things (IoT), and cryptocurrencies, but does not prescribe sharding-specific techniques. Meanwhile, W3C focuses on ensuring interoperability for decentralized identity solutions, leaving the technical implementation details to individual platform developers.

In summary, while the landscape of blockchain standardization is broad, only ITU-T in [50] and ETSI [64] have advanced detailed, operational frameworks and guidelines for sharding, with ETSI further pioneering AI-based sharding management. Within the ITU-T

framework, Recommendation F.75-1.19 in [50] provides a focused set of requirements for blockchain sharding, defining core functional components. While ITU-T's Recommendation F.751.19 [50] and ETSI GR PDL-032 V1.1.1 (2025-04) [64] are valuable, their voluntary status means they provide limited technical depth and lack prescriptive implementation guidance. This highlights the need for more rigorous technical specifications, comprehensive analysis of implementation challenges and trade-offs, and a balanced assessment of risks and benefits as these technologies mature and converge. Meanwhile, the broader standards community continues to provide essential but more generic frameworks that support ongoing innovation and convergence in this rapidly evolving field.

7. Features in Existing Sharding Techniques

There has been a large body of research devoted in introducing efficient sharding techniques to date. Each sharding technique has its respective features to enhance blockchain performance. Beyond classifying sharding into Static, Dynamic, and Layered, our review improves the comparison by linking these techniques to seven key features, giving a clearer and more comprehensive framework for evaluation. This section aims at providing an extensive review of the works contributing to improving the different features of the sharding techniques along with the use of AI in relevant features (e.g., [66]). Although features such as Trust Establishment and Consensus Selection are foundational to blockchain technology in general, they require substantial re-engineering in sharded architectures to address decentralized shard formation, shard-specific security, and cross-shard transactional consistency.

The relationship between these features and key functional phases in sharded blockchain systems is illustrated in Figure 7, further reinforcing their importance as critical design pillars. Specifically, Trust Establishment relates to the Initialization phase, ensuring secure and decentralized formation of nodes into shards. Consensus Selection and DAG Block Structures contribute to Consensus and Block Formation, addressing intra-shard and cross-shard transaction ordering. Epoch Randomness underpins Security Enhancement by introducing unpredictability to mitigate shard takeover attacks. Cross-shard Algorithms and Cross-shard Capacity are integral to Cross-Shard Coordination, facilitating efficient and scalable inter-shard transaction handling. Lastly, Availability Enhancement supports Resiliency, ensuring system robustness even under partial shard failures. Although different systems may prioritize certain features differently, these aspects together form the core foundations that support the scalability, security, and resilience of sharded blockchain ecosystems.

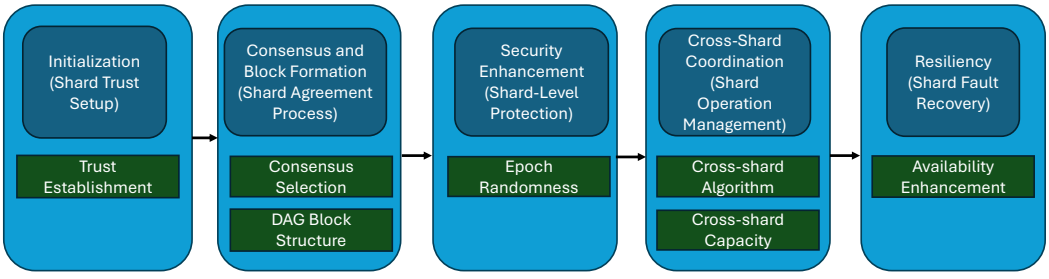

Figure 7. Features related to blockchain sharding architectures and their responsibilities.

7.1. Trust Establishment

Trust Establishment is the process of verifying a node's or user's identity to enable secure transactions within a blockchain. Within Static Sharding, the work in [23] introduces unique features, such as the Identity Establishment and Overlay Setup for Committees. Here, each processor autonomously generates an identity composed of a Proof-of-Work (PoW), IP address, and public key. During the Committee Formation phase, processors

verify each other's identities by solving a PoW problem with publicly verifiable solutions. Sharding typically employs a specialized method for identity verification, which can enhance overall security. However, the reliance on PoW introduces significant scalability challenges, as its computational demands become increasingly burdensome with network expansion.

Significant differences are observed in how various blockchain architectures implement Trust Establishment. For instance, Elastico [23] explores a two-stage Trust Establishment process, potentially enhancing security through multiple validations but complicating the consensus process and potentially slowing down transactions. Conversely, OmniLedger [51] and RapidChain [70] emphasize a single primary stage to enhance speed and simplicity, though this may make them more vulnerable to coordinated attacks. These differences underscore the crucial trade-off between security and efficiency in blockchain design, with each approach offering its own strengths and limitations.

Approaches such as Monoxide [42] and Chainspace [43] do not introduce novel methods for establishing trust, but instead integrate trust directly into their consensus mechanisms. Although this streamlined approach could enhance process efficiency, it also exposes to risks if the consensus mechanism itself is compromised. A notable drawback is that the absence of separate Trust Establishment feature could hinder these technologies' ability to adapt to emergent security threats.

An interesting case is Meepo [71], a consortium blockchain, where Trust Establishment is inherently assured due to the closed nature of its approach, eliminating the need for new Trust Establishment methods. This built-in trust simplifies management and keeps transactions private within the group, but it also restricts the network's openness and may not perform well in larger, more decentralized environments, which is a significant limitation in its design.

Liu et al. in [66] propose Kronos, which establishes trust through a secure shard configuration mechanism that may utilize PoW, Proof-of-Stake (PoS), or public randomness to assign nodes to shards and resist Sybil attacks at the entry point. This process ensures that nodes cannot cheaply fabricate multiple identities during initialization or periodic reconfiguration. Simulations across thousands of nodes on AWS EC2 validate the effectiveness of Kronos under different network models (synchronous, partially synchronous, and asynchronous). However, trust is statically assigned: once a node is verified, no ongoing trust reassessment is performed. Thus, if a node initially behaves correctly but later colludes or becomes compromised, the work in [66] has no built-in mechanisms to detect or expel it. The security model also critically depends on the honest majority assumption within shards, which may not hold under economically incentivized attacks. No behavioral monitoring or epochal re-verification is embedded, posing long-term scalability risks for highly dynamic public deployments.

Lin et al. in [67] introduce DL-Chain, a sharding system that establishes trust and node assignment at the start of each epoch using epoch randomness generated by Verifiable Random Functions (VRFs) combined with Verifiable Delay Functions (VDFs). Each node is randomly assigned to a Proposer Shard and a Finalizer Committee based on this process, ensuring fair and unpredictable allocation. This approach prevents adversaries from concentrating malicious nodes within a single shard, thereby enhancing security while avoiding the computational overhead of PoW-based approaches. Experimental results with up to 2550 nodes demonstrate that the work in [67] random assignment strategy maintains negligible failure probability and resists targeted shard capture. However, DL-Chain treats Trust Establishment as a static process within each epoch. There are no mechanisms for dynamic reassignment or behavior-based penalties during an epoch. As a result, malicious nodes admitted at epoch formation persist throughout the epoch without recertification

or removal. While this work assumes the underlying randomness process is bias-resistant and publicly verifiable, it does not explicitly discuss the risks of potential collusion in VRF/VDF generation.

In Dynamic Sharding, Trust Establishment remains an essential process. However, the works in [52,53] do not introduce novel approaches. For example, the work in [69] employs the Louvain Algorithm to facilitate Trust Establishment, yet it uses a well-established method rather than presenting an innovative approach. However, this reliance on conventional methods may limit the potential for significant improvements in security and system flexibility.

Liu et al. [68] propose DYNASHARD, which utilizes a secure random process for managing committee selection to promote fairness and reduce the risk of collusion. While the protocol ensures committees are selected via internal randomness, it does not explicitly specify periodic or epoch-based reshuffling of committees, nor does it present simulation results quantifying committee diversity or capture rates over time. The Trust Establishment mechanism in DYNASHARD is solely randomness-driven, without incorporating historical node behavior or penalization for misbehavior in the committee selection process. Furthermore, the protocol does not describe any external or decentralized randomness beacon for committee selection, nor does it detail how the randomness source is made publicly verifiable. This lack of external verifiability could limit transparency and potentially undermine trust in adversarial scenarios.

AEROChain [55] establishes trust implicitly through its dual-shard architecture, where each node belongs to a physical shard for transaction validation and to a logical shard for account migration coordination. Both layers use Practical Byzantine Fault Tolerance (PBFT) for consensus, embedding trust into repeated voting among nodes. However, the approach lacks mechanisms for behavioral scoring or identity revalidation across epochs. Trust is static once nodes are assigned, which makes the work in [55] vulnerable to long-term adversarial drift. The scope in [55] focuses on scalable and balanced account migration, not direct trust modeling. No trust-specific simulations are presented, and findings center on the Deep Reinforcement Learning (DRL) based migration model.

SkyChain [73] proposes periodic re-sharding to dynamically reassign nodes across committees, aiming to balance performance and security in a dynamic blockchain environment. Their method leverages an adaptive ledger protocol and a DRL-based sharding mechanism to adjust shard configurations based on observed network state. However, identity establishment is handled through Sybil-resistant PoW puzzles, with no additional cryptographic attestation or behavioral scoring mechanisms in place. The re-sharding process is explicitly adaptive, driven by the DRL policy rather than fixed or deterministic intervals. The primary focus of SkyChain is on scalable dynamic re-sharding, with contributions centered around optimizing performance and security trade-offs. Findings are reported from simulation-based evaluations, focusing on TPS, latency, and safety metrics, without any explicit trust or reputation evaluation.

Similarly, in Layered Sharding, while trust is inherently addressed in every approach, one work in [24] adopts an Assignment System for Trust Establishment, whereas another work in [54] does not propose any specific new method. But inconsistency in addressing trust challenges underscores the need for more innovative solutions that can better meet the evolving security demands of blockchain networks.

SPRING [74] implements a Trust Establishment mechanism where nodes undergo a one-time PoW process at registration. The last bits of the PoW solution directly determine the initial shard assignment. The protocol employs a reconfiguration phase in which consensus nodes are regularly shuffled among shards for security, using VRF to generate unpredictable, bias-resistant randomness for node redistribution and leader selection. This

ongoing reconfiguration mechanism is explicitly designed to prevent persistent collusion or the formation of static shard compositions. While the initial entry barrier is minimal compared to protocols with ongoing reputation or slashing, the security model relies on periodic node rotation and PBFT consensus within shards, not on dynamic behavioral trust assessment. Therefore, the risk of persistent malicious behavior is mitigated by the enforced protocol-level reshuffling of nodes, maintaining the integrity and unpredictability of shard compositions over time.

TBDD [78] introduces a promising response to these limitations through a trust-based and DRL-driven framework. It integrates multi-layered trust evaluation using historical, direct, and indirect feedback to generate local and global trust scores. These scores are used by a decentralized TBDD Committee (TC) to guide re-sharding through DRL, classifying nodes into risk levels and reallocating them to enhance security and reduce cross-shard transactions. Empirical results demonstrate up to 13% improvements in TPS and significantly better risk distribution under adversarial loads. However, maintaining accurate trust scores requires frequent updates and cross-node communication, posing challenges in high-churn environments like IoT. Additionally, aggregated trust metrics may lag behind behavioral shifts, and the committee structure introduces a potential point of failure if compromised. Bootstrapping new or low-activity nodes remains a vulnerability, and the centralized coordination role of the TC raises concerns about long-term decentralization.

Trust Establishment is a fundamental component of Static, Dynamic, and Layered Sharding. Static Sharding, leveraging innovative Trust Establishment methods, often achieves demonstrably superior security. In contrast, Dynamic and Layered Sharding, reliant on conventional techniques, face inherent limitations in security enhancement and adaptive system reconfiguration. This reliance not only impedes immediate progress but also creates a potential bottleneck for future advancements in resilience and agile scaling. While recent AI-driven frameworks such as TBDD introduce layered trust scoring and DRL-based re-sharding, they depend on centralized committee coordination and continuous cross-node communication, which may limit scalability in high-churn environments. Several approaches, including SPRING [74] and DL-Chain [67], treat trust as a static entry event without behavioral reassessment or epochal refresh, which increases vulnerability to long-term collusion. Furthermore, trust metrics in many models are not verifiably auditable or resistant to subtle manipulation, especially under adversarial conditions.

7.2. Consensus Selection

The selection of consensus mechanisms is critical for ensuring the integrity and efficiency of the networks. Various approaches have been developed to enhance the consensus process [79,80] across different blockchain architectures, each with its unique advantages and limitations.

Within Static Sharding, Elastico [23] demonstrates a model that leverages PoW alongside PBFT to establish secure consensus within committees. This hybrid consensus mechanism allows for a Final Consensus Broadcast after achieving agreement within a committee using a conventional Byzantine Agreement protocol. The final committee aggregates the results from all committees and uses a Byzantine consensus process to finalize the outcome then broadcast it to the entire network. The merging of PoW and PBFT is unique, but handling many committees and combining outcomes may slow down the process and increase overhead, especially as the network scales.

OmniLedger [51] employs ByzcoinX for its consensus mechanism. ByzcoinX extends the classical Byzantine consensus by forming a communication tree, where validators are organized hierarchically to reduce messaging overhead. For example, instead of every

node broadcasting to all others, leaders in each subgroup aggregate votes and forward them upward, improving efficiency and reducing latency under high transaction loads. In OmniLedger [51], this enhances the traditional roles of PoW and PoS, which in this context do not directly contribute to transaction validation but rather to the representation of validators in the Identity Block Creation process. ByzcoinX addresses the need for more resilient communication within shards to manage transaction dependencies and improve block parallelization, even in scenarios where some validators fail. ByzcoinX enhances shard communication, but its reliance on validators for identifying block production could lead to inefficiencies or vulnerabilities if coordination fails.

RapidChain [70] offers an Intra-committee Consensus (PBFT) that leverages a unique gossiping protocol suitable for handling large blocks and achieving high TPS through block pipelining. This setup includes a two-tier validation process where a smaller group of validators processes transactions quickly, which are then re-verified by a larger, slower tier for enhanced security. Even though speed and security are increased by this two-tier method, delays may be introduced by the second validation phase.

Monoxide [42] proposes Asynchronous Consensus Zones to scale the block-chain network linearly without compromising on security or decentralization. Its consensus mechanism, Chu-Ko-Nu mining, also functions as a Trust Establishment tool by ensuring uniform mining power across zones and introducing Epoch Randomness. Chu-ko-nu mining, a novel proof-of-work scheme, is employed in Asynchronous Consensus Zones to enhance security. This allows miners to use a single PoW solution to create multiple blocks simultaneously, one per zone, ensuring that mining power is evenly distributed across all zones. As a result, attacking a single zone is as difficult as attacking the entire network. Asynchronous Consensus Zones of Monoxide [42] provide linear scalability, but the challenge of synchronizing randomization and coordinating consensus across multiple zones may lead to increased overhead and latency, particularly in larger networks.

Chainspace [43] utilizes the Sharded Byzantine Atomic Commit (S-BAC) protocol, a combination of Atomic Commit and Byzantine Agreement protocols, to ensure consistency across transactions that involve multiple shards. This method guarantees that a transaction must be unanimously approved by all shards it touches before it can be committed, thus maintaining transaction integrity. Although the S-BAC protocol maintains high integrity in cross-shard transactions, the need for unanimous clearance may introduce inefficiencies or delays, especially as the number of shards increases. In practice, S-BAC works like a two-phase commit extended to sharded environments. For instance, if a transaction spans Shard A and Shard B, both shards first enter a 'prepare' phase and lock the resources. Only if both confirm in the 'commit' phase is the transaction finalized; otherwise, both shards roll back. This ensures atomicity across shards, though at the cost of higher communication overhead.

Meepo [71], on the other hand, does not introduce a new consensus mechanism but rather employs a non-modified PoA, relying on the existing trust model inherent in its consortium blockchain framework. The blockchain's flexibility and openness are restricted by Meepo's use of PoA, which makes it less suitable for decentralized or permissionless networks, even though it simplifies the consensus process in a regulated setting.

Kronos [66] distinguishes between "happy" and "unhappy" paths in cross-shard transaction processing. In normal operations, transactions are processed or rejected with minimal overhead using standard intra-shard Byzantine Fault Tolerance (BFT), while the unhappy path invokes additional rollback mechanisms to ensure atomicity and shard consistency. AWS experiments demonstrate that Kronos achieves high TPS and low latency under various network synchrony and Byzantine fault conditions. The protocol's transaction integrity and atomicity guarantees are established through formal security

analysis presented in this work, rather than through empirical experiments. However, the rollback mechanism introduces significant communication overhead and increases finalization latency under persistent fault conditions. Moreover, the protocol assumes low rollback frequency, and frequent rollbacks in highly adversarial environments could lead to severe TPS degradation. Kronos also does not dynamically adapt committee memberships or quorum thresholds based on real-time fault rates, limiting flexibility.

DL-Chain [67] modularizes consensus by dividing transaction proposal and finalization into two distinct layers. Proposer shards are responsible for assembling and processing transactions, while finalizer committees independently validate and finalize these transactions. This architectural separation enhances parallelism and fault isolation, resulting in significant TPS improvements as demonstrated in experimental results. Node assignment to proposer shards and finalizer committees is performed at the beginning of each epoch using a randomness process, and these assignments remain fixed throughout the epoch. Consequently, DL-Chain does not support dynamic reassignment or migration of workloads during an epoch. In scenarios with highly variable or uneven transaction loads, some proposer shards may become bottlenecks due to the absence of intra-epoch load balancing mechanisms. If a finalizer committee leader becomes non-responsive due to validator churn or failure, DL-Chain employs a view change protocol within the Fast Byzantine Fault Tolerance (FBFT) consensus algorithm to replace the faulty leader and restore liveness within the committee.

In Dynamic sharding, DYNASHARD [68] adopts a hybrid consensus architecture, utilizing BFT for intra-shard transaction validation and a combination of Multiparty Computation (MPC) and threshold signature schemes for global coordination. This design ensures that transaction commitments, both within and across shards, are achieved with strong atomicity and security guarantees. The protocol's evaluation demonstrates resilience to adversarial behaviors, including collusion and double-spending, through comprehensive security analysis and simulation-based validation. Nonetheless, the use of threshold signature aggregation and MPC introduces non-trivial computational and bandwidth overheads. These cryptographic protocols require each participant to compute and exchange partial signatures or intermediate values in multiple rounds, significantly increasing the computational workload and network traffic compared to traditional consensus mechanisms. As transaction volume and committee size scale, these overheads may impact TPS and latency, making DYNASHARD less suitable for high-frequency or latency-sensitive applications (e.g., [51]). Additionally, DYNASHARD does not incorporate early commitment or fast-finality optimizations, leaving it potentially susceptible to synchronization delays during periods of peak system concurrency.

AEROChain [55] uses PBFT at two levels: physical shards (validate local transactions) while the logical shard (handles migration proposals during the reconfiguration phase). The contribution lies in separating state migration from transaction consensus. This layered PBFT structure provides modularity but introduces coordination overhead and potential bottlenecks during high migration volumes. The approach was validated as part of the full AEROChain simulation, but no consensus-specific benchmarks were isolated. The absence of fast-path execution or rollback handling limits resilience to stalled consensus rounds.

SkyChain [73] uses standard BFT protocols for intra-shard consensus without modifications or enhancements. Its novelty lies in DRL-powered re-sharding, not consensus innovation. Their method assumes fewer than one-third faulty nodes but lacks rollback, fast-track, or speculative commit techniques. No simulations were presented to test consensus scalability. Thus, while structurally sound, its consensus mechanism is basic, and no fallback mechanisms are discussed.

While in Layered Sharding, SPRING [74] employs PBFT as the intra-shard consensus protocol for both A-Shard and T-Shard. The protocol operates under the partial synchrony assumption common to BFT systems. SPRING's DRL agent dynamically assigns new addresses to shards in order to balance transaction load and minimize cross-shard transactions. In parallel, the protocol includes a periodic reconfiguration phase in which consensus nodes are reshuffled across shards to maintain security and prevent persistent collusion. While workload imbalance among shards may still arise due to the power-law distribution of transaction activity, SPRING's design seeks to balance TPS and fairness without requiring dynamic committee resizing or more advanced consensus adaptations. Experimental results show that SPRING reduces cross-shard transaction ratio and improves TPS, but all evaluated consensus groups are periodically rotated and operate under standard PBFT assumptions.

In Dynamic (e.g., [25,52,53]) and Layered Sharding (e.g., [24,54]), a key challenge is designing consensus protocols that efficiently coordinate how blocks are generated and verified across shards. These approaches typically rely on existing, non-modified consensus mechanisms such as PoW, PBFT, or byzantine-based techniques, which may not introduce new consensus mechanisms but are essential for the proper functioning of these sharded techniques. As the demand for more effective and scalable approaches grows, reliance on pre-existing consensus processes may limit the flexibility and scalability of Dynamic and Layered Sharding systems.

Recent efforts have explored the integration of AI into sharding and consensus mechanisms to enhance blockchain scalability, fairness, and energy efficiency. El Mezouari and Omary in [81] present a hybrid consensus framework that combines PoS for block creation with AI-enhanced sharding for transaction validation. In their approach, decision tree algorithms dynamically allocate tasks to shards based on network load and historical node behavior. Entropy measures and Haversine distance metrics are used to optimize shard load distribution and minimize cross-shard communication overhead. El Mezouari and Omary in [81] argue that this model mitigates the centralization risks of pure PoS while improving decentralization, TPS, and energy efficiency compared to traditional PoW models. However, incorporating AI-driven shard management introduces challenges related to algorithmic transparency, susceptibility to model drift, and dependency on high-quality, unbiased training datasets. Additionally, the operational complexity of coordinating between PoS and intelligent sharding logic could pose risks to system stability if not rigorously optimized.

Similarly, Chen et al. in [82] propose Proof-of-Artificial Intelligence (PoAI) as an alternative to traditional PoW and PoS consensus mechanisms. In PoAI, nodes are classified into "super nodes" and "random nodes" using Convolutional Neural Networks (CNNs) trained on metrics such as transaction volume, network reliability, and security posture. Validators are dynamically selected based on capability scores rather than hash power or stake, aiming to reduce resource consumption and promote fairer node rotation. While PoAI offers improvements in efficiency and energy conservation, it introduces concerns about model explainability, fairness, and vulnerability to adversarial attacks. The criteria defining "super nodes" could inadvertently concentrate power among high-resource participants, undermining decentralization goals, particularly if CNN biases are not properly mitigated.

To address the limitations observed in conventional sharding and consensus mechanism designs, emerging AI-augmented mechanisms offer promising alternatives. Consensus mechanisms like PoAI [82] leverage machine learning models to intelligently assign validator roles, thereby improving transaction confirmation speed and reducing energy consumption. Similarly, hybrid approaches that integrate PoS with AI-based shard reconfiguration [81] provide dynamic adaptability to evolving network conditions. However,

despite demonstrating measurable performance improvements, these AI-driven designs introduce new challenges related to transparency, fairness, and adversarial resilience. Future work must critically address these issues, including enhancing model interpretability, safeguarding against manipulative behaviors, and developing lightweight validation protocols, before widespread deployment in permissionless blockchain environments can be realized.

Li et al. in [83] focuses on improving TPS in sharded blockchain systems through optimization of consensus-layer parameters, such as block size, shard count, and time interval. It introduces Model-Based Policy Optimization for Blockchain Sharding (MBPOBS), a Reinforcement Learning (RL) framework that uses Gaussian Process Regression to model blockchain performance and guides parameter optimization via the Cross-Entropy Method. The DRL component is used to predict performance outcomes and select optimal configuration policies in a sample-efficient manner. Simulation results show that MBPOBS yields substantial TPS improvements (1.1× to 1.26×) compared to model-free RL baselines (Batch Deep Q-learning and Deep Q-Network with Successor Representation). The primary strength of this work lies in its statistically grounded, sample-efficient method for optimizing consensus-related parameters. However, the study is limited to consensus performance and does not incorporate aspects such as trust, shard reliability, or cross-shard fault tolerance. Additionally, all evaluation is performed in a simulated environment, though the model's robustness is tested under varying rates of malicious nodes (adversarial settings).

In conclusion, while these various consensus mechanisms provide effective approaches for managing blockchain transactions across different systems, they also present challenges related to scalability, reliability, and complex implementation. A thorough evaluation of each approach is essential to determine the best strategy for maintaining system performance and security. The scalability and adaptability of blockchain networks, especially in large or dynamic environments, are at risk due to bottlenecks or inefficiencies, whether from traditional consensus mechanisms or newer, more complex methods that could cause new complications. Although AI-based models such as PoAI, hybrid PoS-AI designs, and model-based optimization frameworks have shown measurable improvements in simulation settings, most have not been validated under real-world or adversarial conditions. In addition, many of these approaches do not address Trust Establishment, cross-shard fault tolerance, or model explainability, limiting their practical assessment for deployment in decentralized environments.

7.3. Epoch Randomness

The integration of Epoch Randomness within various sharding techniques significantly enhances the security and fairness by introducing unpredictability in node and shard assignments. This feature is crucial for preventing manipulation and ensuring equitable distribution of network load and responsibilities. Epoch Randomness enhances security, but its application across different sharding techniques presents challenges in balancing operational efficiency with increased complexity.

In Static Sharding, Epoch Randomness is implemented through distinct methods in several approaches. Elastico [23] employs a Distributed Commit-and-XOR method, which creates a biased yet constrained set of random values that directly influence the PoW challenges in the subsequent epoch. This method ensures that randomness plays a role in the mining process, which helps strengthen security by making it harder for attackers to predict or manipulate the mining outcomes. Although the Commit-and-XOR method improves security, its complexity could result in excessive overhead, which may impact network efficiency as the network scales. OmniLedger [51] uses a combination of VRF [84,85] and RandHound [26], which ensures the randomness is both unbiased and unpredictable. RandHound's approach, which involves dividing servers into smaller groups and using a

commit-then-reveal protocol, ensures that the randomness includes contributions from at least one honest participant, thus maintaining integrity. The reliance on multiple server groups and protocols may slow down the process, especially in larger networks, potentially impacting overall performance. RapidChain [70] opts for a Distributed Random Generation protocol optimized by a brief reconfiguration protocol based on the Cuckoo Rule [86], allowing for rapid and unbiased randomness generation essential for its operational efficiency. Although RapidChain's method speeds up transaction processes, it may not be able to expand when more complex random choices are required. As the need for sophisticated approaches grows, scalability issues may arise within blockchain networks.

Kronos [66] generates epoch randomness for shard assignment using public randomness, which can be derived from PoW, PoS, or other secure sources. This process provides non-predictable, deterministic validator assignment. The protocol is designed to ensure consistent shard diversity and resilience against validator collusion through its random assignment process, assuming the underlying randomness is secure. However, if the randomness generation relies on PoW or PoS outputs rather than a publicly verifiable randomness beacon, it may be vulnerable if those outputs become skewed or dominated by a colluding miner or staker group.

DL-Chain [67] generates randomness at each epoch using outputs VRFs, which provide unpredictability and allow local proof verification. The work in [67] claims that the assignment process based on this randomness is bias-resistant, attributing this property to the use of VRF and VDF technologies as established in prior work. However, DL-Chain does not include simulation studies or experiments specifically evaluating the bias-resistance or security of its own randomness mechanism. Additionally, the protocol does not incorporate decentralized randomness aggregation or zero-knowledge proofs for randomness generation. This absence could, in principle, allow adversaries who control VRF private keys to subtly bias role allocations, a limitation that this work does not explicitly address.

Dynamic Sharding also explores Epoch Randomness but with varying emphases and integration depths. For example, the work in [52] significantly focuses on incorporating Epoch Randomness within its cross-shard transaction process to enhance security. Other Dynamic Sharding approaches, such as those introduced in [25,53], recognize the importance of randomness but do not delve as deeply into its systematic integration as seen in Static Sharding. The integration of certain mechanism in Dynamic Sharding, such as those proposed in [69], lacks proper organization, making them potentially vulnerable to manipulation or attacks. This risk is heightened in complex networks with high transaction volumes, and as the system scales, the threat becomes more evident.

DYNASHARD [68] selects committees through an internal secure random process (in this context, committee selection refers to the random assignment of validators to serve as consensus groups for individual shards, responsible for transaction validation and consensus within the protocol). However, the protocol does not describe the frequency of reseeding or provide simulation evidence of committee diversity across epochs. Moreover, the randomness source is not publicly auditable, and in a permissionless adversarial setting, compromised entropy could bias committee selection without detection. This risk could be mitigated by adopting decentralized or externally verifiable randomness commitment protocols.

AEROChain [55] introduces a single shared random seed per epoch, which governs both node reassignment and migration transaction determinism. This ensures synchronization without external coordination overhead. However, this randomness is not generated through a verifiable process such as VRFs or public randomness beacons. Its centralization could lead to vulnerabilities if the seed is manipulated. The scope is to enable determin-

istic AERO policy execution, validated indirectly through simulation-based performance improvements but not through cryptographic robustness tests.

SkyChain [73] supports epoch-based reconfiguration, but the source and security of its randomness are unspecified. While re-sharding intervals and block size are adjusted based on DRL policies, the randomness mechanism remains opaque. As a result, it lacks public verifiability or resistance to seed manipulation. Its randomness approach is implicit and not evaluated independently.

Layered Sharding (e.g., Pyramid [24], and OverlapShard [54]), meanwhile, incorporates Epoch Randomness into the Cross-shard Algorithm process, ensuring that transactions across different layers of shards maintain unpredictability and security. This method is important for preventing targeted attacks and ensuring a fair distribution of transaction loads across the network. The multiple layers in this approach likely add to its complexity, which could lead to inefficiencies and affect overall performance. These risks are more likely to arise in large-scale implementations.

On the other hand, SPRING [74] assigns nodes to shards at registration based on the last bits of the PoW solution string. The protocol incorporates a periodic reconfiguration phase in which consensus nodes are regularly shuffled among shards using VRF-generated randomness, ensuring that shard compositions remain unpredictable and resistant to long-term adversarial planning or validator collusion.

Traditional approaches relying on static randomness in sharding risk inefficiency as workloads become increasingly dynamic and predictable over time. AI-Shard introduced in [49] addresses this by using a Graph Convolutional Network–Generative Adversarial Network (GCN-GAN) model to generate predictive node interaction matrices, enabling time-sensitive reshuffling that optimizes shard configurations based on anticipated workloads. This method demonstrably reduces cross-shard transactions and improves throughput in dynamic IoT environments. Wang et al. [49] present prediction-based sharding as superior to static randomization, it inherently trades pure randomness for workload-driven optimization. From a security perspective, reliance on historical data and model predictions could potentially introduce patterns susceptible to adversarial exploitation if model errors or biases occur—though such risks are not discussed by the authors. Simulation results confirm AI-Shard's performance advantages, but the ultimate security and adaptability of the framework would depend on the ongoing accuracy and robustness of its predictive models. To further enhance adaptability, Wang et al. [49] introduce a dual-layer architecture with DRL-based parameter control (via Double Deep Q-Network), allowing for continuous reconfiguration in response to environmental changes. While this work highlights significant computational cost and challenges in real-world deployment, it lacks additional considerations such as robustness and model explainability.

Epoch Randomness holds an essential place in keeping blockchain networks safe and fair. Different sharding techniques use randomness in unique ways. Static Sharding uses solid and direct techniques. On the other hand, Dynamic and Layered Sharding are still evolving to improve randomness methods. Current research highlights the importance of randomness for the integrity and efficiency of blockchain tasks. It also identifies areas where these methods may not yet be fully effective. However, several approaches lack cryptographic verifiability, such as the use of non-transparent seed generation in AEROChain [55] and SkyChain [73], which do not provide public randomness proofs or resistance to manipulation. In AI-based approaches like AI-Shard [49], shard assignments are decided by model predictions rather than by secure random numbers. This means that if the model makes mistakes or is biased, attackers might find and use patterns in how nodes are assigned. Moreover, approaches such as SPRING [74] and DYNASHARD [68]

do not incorporate decentralized or auditable randomness sources, raising concerns about long-term entropy integrity in adversarial environments.

7.4. Cross-Shard Algorithm

Cross-shard Algorithm helps optimize transaction processing across different shards, which plays an important role in improving the overall resilience of a blockchain network.

Within Static Sharding technique, the Two-Phase Commit, as utilized in OmniLedger [51], represents a fundamental approach where transactions affecting multiple shards are handled atomically. This method employs a bias-resistant public-randomness approach to select large, statistically representative shards, ensuring fair and efficient transaction execution. The cross-shard Atomix in [51] extends this concept by ensuring that transactions are either fully completed or entirely aborted, maintaining consistency across shards in a Byzantine environment. The Two-Phase Commit process can cause delays, especially in busy networks, leading to slower transaction speeds and a noticeable impact on overall performance.

Al-Bassam et al. in [43] implemented S-BAC further contributes to these robust cross-shard mechanisms by detailing a five-phase process starting from the Initial Broadcast to the Final Process Accept. This structure helps in mitigating issues such as rogue BFT-Initiators by implementing a two-phase procedure that waits for a timeout before taking action, thus safeguarding the integrity of transaction processing. Although S-BAC enhances security, its five-phase procedure is likely to cause unnecessary delays, particularly in time-sensitive situations, making transaction execution more difficult.

In contrast, RapidChain [70] adopts a faster approach with its Cross-shard Verification, using a routing method inspired by Kademlia that minimizes latency and storage requirements for each node (meaning nodes only store their own shard's data, not the entire blockchain) which enables quick identification and verification of transactions across shards, streamlining the validation process. RapidChain's approach lowers latency, but if not properly monitored, its reliance on quick routing (where transactions are directed to the correct shard through a small number of efficient hops utilizing a routing table) may pose security threats, especially in a network with a large number of nodes.

Wang and Wang in [42] introduce a novel concept of Eventual Atomicity where transactions are efficiently completed without relying on the traditional Two-Phase Commit mechanism. This method allows for asynchronous, lock-free interleaving of transactions across zones, enhancing the overall TPS of the blockchain network and reducing the confirmation latency for cross-zone transactions. To guarantee that all nodes come to a final consensus regarding the transaction status, the absence of a conventional commit procedure may cause issues with dependability and consistency.

Meepo [71] presents a comprehensive investigation into sharded consortium blockchains by enhancing cross-shard efficiency, cross-contract flexibility, and shard availability through Cross-epoch, Cross-call mechanisms, Partial Cross-call Merging Strategy and maintaining rigorous transaction atomicity through a Replay-epoch. Although Meepo provides a thorough approach to meeting these demanding requirements, its reliance on mechanisms such as Cross-epoch, Cross-call mechanisms, Partial Cross-call Merging Strategy, and Replay-epoch may introduce additional complexity and overhead, particularly in large-scale deployments or environments with high transaction volumes.

Kronos [66] introduces batch certification for cross-shard transactions using either vector commitments or Merkle trees, enabling atomic certification of multiple transactions in a single protocol instance. This "batch-proof-after-BFT" approach significantly reduces cross-shard messaging overhead, as a single batch proof can replace the need for separate proofs for each transaction, thus improving efficiency when processing large numbers of

cross-shard transactions. The protocol provides strong atomicity guarantees by design, as proven in its formal analysis, and experiments confirm high TPS and low latency under various workloads. Within each batch, transactions are validated individually, and if a transaction fails validation, it is rejected without impacting other transactions in the batch.

DL-Chain [67] applies a relay-forwarding model for cross-shard transaction handling. While efficient under normal conditions, relay operations lack a rollback or compensation mechanism. If a relay node fails or a destination shard becomes unavailable, transactions may be left in an incomplete or pending state without automatic recovery. This work does not present experiments specifically on this failure scenario, but the absence of a rollback protocol means such risks are not fully addressed.

Additionally, the work in [52] explores Dynamic Sharding by employing a Two-Phase Commit protocol and transaction splitting to manage cross-shard transactions effectively. It uses Anchorhash in conjunction with the Jump Consistent Hash Algorithm to minimize disruptions in node assignment mapping caused by sharding changes. This method is recognized for enhancing scalability through more efficient transaction processing and robust auditing capabilities. Even though dynamic reconfiguration during transaction processing offers flexibility, it may introduce performance bottlenecks in rapidly growing networks. Notably, the work in [52] is the only work we identified within the Dynamic Sharding paradigm that incorporates a Cross-shard Algorithm, suggesting that this strategy is uncommon in Dynamic Sharding.

DYNASHARD [68] validates cross-shard transactions using a combination of MPC and threshold signature schemes. This ensures atomic validation even under adversarial conditions, providing strong security guarantees against coordinated attacks. However, the reliance on synchronous MPC introduces significant computational overhead, and the protocol does not optimize for early commitment under low-fault scenarios, resulting in persistent high latency even when failures are rare.

Building on this foundation, DYNASHARD [68] proposes a hybrid consensus mechanism that integrates threshold-based signatures with decentralized validation to enhance cross-shard transaction efficiency. Unlike traditional commit-based protocols, it employs Merkle-based synchronization and real-time shard boundary adjustments, allowing faster responsiveness under varying workloads. This model exemplifies a broader shift toward dynamic, lightweight cross-shard coordination mechanisms tailored for high-throughput, high-variability blockchain environments.

AEROChain [55] features a Cross-shard Transaction Module (CSTM) inspired by Monoxide's relay-based design. It processes intra-shard and cross-shard transactions during consensus, while migration-specific cross-shard transactions are executed during reconfiguration. Grouping by prefix-based account abstraction reduces overhead. However, AEROChain lacks atomic commit, rollback, or escrow mechanisms for cross-shard failures. The focus is on reducing cross-shard transaction frequency through migration rather than ensuring atomicity. The module is embedded in the DRL-evaluated framework but not tested separately for failure tolerance.

SkyChain [73] claims cross-shard support, but its implementation lacks clarity. There is no specification of consistency guarantees or error recovery protocols in partial transaction failure scenarios. Its cross-shard mechanism is undeveloped and not supported by simulation results or architecture diagrams, making this a major limitation in transactional robustness.

Finally, the Layered Sharding architecture in Pyramid [24] offers a sophisticated framework for managing cross-shard transactions. It employs a combination of internal (i-shard) and bridging (b-shard) shards, with b-shard nodes tasked with verifying and proposing blocks that span multiple shards. In contrast, OverlapShard [54] primarily relies

on a structure comprising both Actual and Virtual shards. The work in [24] supports a unique block preparation process using Co-si for scalability, which needs further refinement to enhance its effectiveness. While the Layered Sharding approaches introduce innovation, it could complicate block preparation, potentially reducing TPS and increasing latency, particularly in complex, highly layered architectures.

SPRING [74] minimizes cross-shard interactions by proactively assigning new addresses to shards using a DRL agent, significantly reducing the frequency of cross-shard transactions. For unavoidable cross-shard transactions, SPRING adopts a relay-based processing model, in which the transaction is processed on the source shard and then relayed to the target shard for finalization. However, the protocol does not specify any formal rollback, escrow, or atomic commit mechanisms to guarantee consistency in the event of partial failures or complex multi-shard dependencies. As such, SPRING may be less robust in scenarios involving intricate, interdependent cross-shard transactions.

With an emphasis on improving cross-shard transactions to increase scalability, reliability, and efficiency, these diverse approaches (e.g., [24,52]) demonstrate the ongoing innovation in blockchain technology. However, each approach has unique drawbacks, particularly in terms of added complexity and potential performance compromises. A major challenge in creating reliable and effective blockchain operations is finding the optimal balance between the demands of large-scale, decentralized networks and the need for robust performance, scalability, and security. Several works, such as AEROChain [55] and SkyChain [73], do not provide explicit guarantees for transactional atomicity or rollback mechanisms to manage partial cross-shard failures. This absence makes it difficult to comprehensively evaluate their fault tolerance, as there is insufficient evidence regarding their behavior under adverse or failure scenarios. Additionally, methods employing batching or multi-phase coordination, including Kronos [66] and S-BAC [43], inherently introduce latency under conditions of low transaction volume or in time-sensitive contexts such as real-time financial systems and IoT applications, where rapid transaction processing is critical. Critically, these protocols lack detailed mitigation strategies, including adaptive coordination methods or simplified fallback procedures, to effectively minimize latency under these less demanding operational scenarios. Consequently, this unaddressed latency significantly impacts the evaluation of their practical responsiveness and reliability, particularly in high-throughput or adversarial environments.

7.5. Cross-Shard Capacity

Cross-shard Capacity is one of the unique methods to optimize shard operation and transaction processing across diverse blockchain environments. It involves adjusting the number and size of shards to decrease cross-shard communication. Notably, this method is not employed in Static or Layered Sharding techniques.

C. Chen et al. in [69] propose a method in Dynamic Sharding that modifies user distribution according to real-time network conditions in a public blockchain. This method differs from conventional sharding techniques that often allocate users randomly, hence enhancing long-term system performance in dynamic settings. This protocol includes stages such as the Validator Redistribution Approach and the Validator Vote and System Reconfiguration Approach, which allow for ongoing adjustments to shard composition in response to changing network demands. Although this dynamic protocol improves flexibility, the frequent need for reconfiguration and redistribution may increase operational complexity and affect the stability of the system as a whole in larger networks.

Concurrently, Tao et al. in [25] present a methodology that correlates the quantity of miners per shard with the transaction volume within each shard. This methodology, illustrated by the MaxShard system, guarantees that shards experiencing elevated transaction

volumes are allocated adequate processing resources to sustain performance. Furthermore, it utilizes a combination of Inter-shard Merging, Intra-shard Transaction Selection, and Parameter Unification Method to optimize cross-shard communication, enabling transaction validation within each shard, thus reducing latency and increasing TPS. However, if demand fluctuates rapidly, allocating miners to transaction volumes may result in resource imbalances amongst shards. This could therefore worsen computing overhead and bottlenecks.

Effective Sharding Consensus Mechanism in [52] focuses on the initial assignment of nodes to shards and the subsequent redistribution as shards evolve. This dynamic allocation helps maintain balance across the network and adapt to changes (e.g., shard addition or deletion). The Node Remapping process, integral to this approach, ensures nodes are spread equitably across the remaining shards, maintaining the integrity and efficiency of the network. Even if dynamic allocation improves balance, frequent node remapping can increase computational load and cause delays, especially in networks with high shard volatility.

The Dynamic Sharding Protocol Design for Consortium Blockchains in [72] utilizes Unspecified Agents to oversee transactions inside a consortium blockchain framework. These agents allocate transactions to shards according to the sender's information and oversee the consensus process throughout the network. The concept uses random integers to construct unpredictable routing tables for each epoch and incorporates both Boss Shard and Normal Shard components, hence enhancing security against attacks on shard integrity. For larger deployments, the greater system complexity that comes with enhanced security through Unspecified Agents and random routing must be weighed against significant scalability and maintainability challenges.

By creating public and private keys and exchanging identifying information, each node in this method [72] initializes and guarantees secure connections inside the consortium blockchain architecture. In order to ensure reliable transaction processing and system resilience, the method also includes successive phases such as Transaction Sharding and Micro Block Generation, Full Block Generation, and Synchronization. Despite its resilience, the multi-phase design may result in synchronization delays (particularly during the aggregation and dissemination of micro-blocks by the boss shard and their verification across other shards). This delay is notably intensified under severe transaction loads, where large volumes of simultaneous transactions increase complexity, communication overhead, and processing latency, potentially impacting overall system performance.

DYNASHARD [68] dynamically regulates its transaction load through the support of shard splitting and merging, informed by real-time assessments of transaction volume and system resource use. When a shard's transaction volume or resource utilization surpasses a specified splitting threshold, the shard is divided into numerous smaller shards to evenly distribute the load and preserve transaction processing efficiency. Should a shard exhibit continuously low activity, as evidenced by its transaction volume and resource utilization falling beneath a designated merging threshold for multiple consecutive epochs, DYNASHARD may merge it with other underutilized shards to improve overall resource efficiency and system performance. Because of this adaptable approach, DYNASHARD can continue to function reliably even in the event of transaction volume fluctuations or low levels.

However, the shard splitting and merging process introduces considerable complexity in maintaining system consistency. During a split, the shard's entire state, including ongoing transactions, must be atomically divided among the new child shards. Validators must update their local views of the network topology, while ensuring that no transactions are lost, duplicated, or incorrectly assigned. Similarly, merging shards requires reconcil-

ing multiple independent shard states into a single coherent ledger without introducing transaction conflicts or state corruption. These transitions are highly sensitive to timing and synchronization accuracy, and if validators operate on outdated shard mappings or fail to synchronize their views properly, temporary inconsistencies could arise, making the system vulnerable to double-spending attacks or transaction replay.

Moreover, there is currently no specified rollback or recovery plan in place for DYNASHARD in the event that a split or merge operation fails in the middle of the process, such as validator crashes, partial network partitions, or anomalies in transaction queuing. Incomplete shard state changes, validator discord, or partial ledger divergence may arise from a failed migration if transactional atomicity guarantees are absent during shard reconfigurations. Even though dynamic shard restructuring greatly enhances scalability and resource efficiency, the problems show that it needs strong migration protections, fault-tolerant processes, and strict state validation mechanisms to keep the system's integrity during transitional times.

AEROChain [55] offers AERO, a DRL-based optimizer that balances shard loads and minimizes cross-shard transactions by utilizing prefix-level account grouping. The DRL policy is taught on actual Ethereum data and evaluated against five benchmarks (e.g., SPRING [74] and Monoxide [42]). It attained a 31.77% increase in TPS compared to state-of-the-art approaches, due to diminished cross-shard transaction ratios and enhanced migration operations. Nonetheless, its adaptability is reactive and limited in adversarial robustness evaluation. Although the prefix abstraction increases training efficiency, performance may be impaired by abrupt behavioral changes or malicious behaviors.

SkyChain [73] facilitates the modification of shard quantity and block size via DRL-driven re-sharding. It characterizes sharding dynamics as a Markov Decision Process and adjusts parameters to optimize security and performance. Nonetheless, its limitations include a lack of granularity in workload data and insufficient learning-based workload adaptability, unlike approaches such as AEROChain [55]. No migration plan or optimization focused on cross-shard transaction is provided. Simulation results demonstrate that SkyChain attains approximately 30–35% enhancement in overall TPS relative to fixed sharding baselines. However, the assessment is confined to aggregate measures and fails to examine performance on a per-shard or per-transaction basis, hence neglecting real transaction-level implications.

Enhancing cross-shard communication and transaction processing by adjusting shard size and quantity to maximize Cross-shard Capacity is a common theme among these approaches (e.g., [52,69]). Methods such as dynamic user allocation [69] (where a user denotes an account that submits transactions to the blockchain) and node reassignment [52] (with nodes defined as servers or computers tasked with processing and validating transactions) facilitate workload distribution and ensure efficient transaction processing across shards. Moreover, the emphasis on minimizing cross-shard communication in On Sharding Open Blockchains [25] and enhancing TPS across every approach highlights the persistent endeavors to enhance latency and TPS rates, which are critical metrics for blockchain network performance. While these approaches enhance TPS and inter-shard communication, dynamic shard management may lead to inefficiencies, especially in extensive or volatile networks. The approaches analyzed in the existing literature offer a thorough review of strategies for improving Cross-shard Capacity in both public and consortium blockchains. Considering their potential, these approaches encounter considerable problems regarding complexity, scalability, and adaptability, especially in dynamic environments. Continuous adjustments are essential to maintain resilience and resilient performance.

Zhang and Xue in [48] reformulated the shard allocation problem in the FLPShard model as a Single-Source Capacitated Facility Location Problem, allowing for variable

node assignment based on inter-node interdependence, latency, and geographic proximity. The concept improves intra-shard efficiency and diminishes the frequency and expense of cross-shard transactions by clustering highly interacting nodes. This approach is especially effective in Industrial IoT settings, where physical infrastructure and organizational hierarchies affect communication dynamics. Although FLPShard demonstrates formal rigor and quantifiable enhancements in TPS and latency, it introduces significant computing complexity and assumes constant environmental conditions, which may restrict its applicability in more dynamic or diverse deployment scenarios.

On the other hand, the AI-Shard framework in [49] utilizes a predictive and adaptive approach for capacity management. It utilizes a GCN-GAN model to predict changing node interaction matrices and leverages a DRL controller to dynamically optimize shard configurations. This anticipatory technique enables AI-Shard to proactively synchronize shard structures with fluctuating communication patterns, reducing cross-shard interactions and enhancing TPS, particularly in Building IoT applications. The dual-layer architecture of AI-Shard, comprising a coordinating main shard and adaptive sub-shards, enhances scalable collaboration in densely populated IoT contexts. The reliance on past data for training, sensitivity to DRL parameter adjustment, and the possibility of unstable convergence, however, provide significant difficulties that could compromise real-time responsiveness in hostile or volatile networks.

Lin et al. in [87] emphasize improving the scalability and communication efficiency of blockchain-based Federated Learning (FL) systems within Intelligent Transportation Systems. It tackles the constraints of current two-layer blockchain systems that depend on static shard parameters and entail substantial inter-chain communication expenses. The primary contributions are a reputation-based shard selection method to eradicate bad nodes, a streamlined shard transmission strategy to minimize overhead, and a DRL-based adaptive sharding controller to dynamically optimize shard configurations. The suggested approach incorporates DRL to perpetually adjust shard configurations based on environmental insights, while subjective logic is employed to represent trust both within and among shards. The simulation findings indicate that the framework enhances throughput and decreases latency while maintaining FL accuracy. The main advantage resides in the efficient integration of trust management with DRL-based shard adaptation, directly tackling Cross-shard Capacity challenges. Since the work was assessed using a simulator, a real-world validation is required to determine its practical efficacy.

Chen et al. in [88] focus on scalability and flexibility in blockchain-enabled IoT systems by tackling reconfiguration delays and suboptimal shard allocation. To organize IoT devices into optimized shards, the proposed Block-K Clustering approach integrates multiple algorithms, including K-means clustering, Genetic Algorithms, and the Cuckoo Rule, to enhance shard formation and efficiency. A DRL model is incorporated to dynamically modify the number of shards and consensus parameters according to transactional patterns and device clustering dynamics. Their system represents the network as a Dynamic Transaction Flow Graph, utilizing a DRL component designed to enhance cluster modularity while concurrently reducing inter-shard communication. Simulation outcomes validate that the approach significantly boosts TPS, diminishes reconfiguration duration, and strengthens system resilience against malicious activity. A primary advantage is the integration of many algorithms that facilitates real-time, performance-sensitive shard modification. Nonetheless, this work does not investigate the framework's scalability in exceedingly large IoT networks, nor does it thoroughly assess the communication costs associated with frequent reconfiguration.

Taken together, both frameworks in [87,88] emphasize that optimizing Cross-shard Capacity requires not only increasing shard count but also the smart management of shard

composition, interaction density, and responsiveness to fluctuating demands. Adaptive capacity optimization is an underexplored design component, yet it is critical for developing scalable, robust sharded blockchains capable of supporting sophisticated, high-throughput applications in IoT and beyond. However, many of these techniques rely primarily on simulation-based evaluations, leaving little insight into their efficacy in real-world deployment conditions or hostile network scenarios. Furthermore, numerous systems, especially those utilizing DRL-based reconfiguration, inadequately highlight the coordination of shard rebalancing during state transitions, as well as the maintenance of consistency amid delayed synchronization or partial node updates. These unresolved issues hinder the evaluation of robustness and practicality in dynamic settings.

7.6. DAG Block Structure

The integration of DAG offers a sophisticated approach to managing data structures and transaction processing [89,90]. This method naturally aligns with the inherent characteristics of blockchain transactions and object interactions, in which each transaction may invalidate certain inputs and generate new active objects as outputs, thereby forming a DAG. This feature is predominantly observed in Static Sharding.

For example, OmniLedger [51] employs a block-based DAG (BlockDAG) to enhance the concurrent processing of blocks within its architecture. In this approach, each block can reference multiple parent blocks, effectively capturing the simultaneous actions and interdependencies of transactions across different blocks. Similarly, the Hash-DAG structure [43] represents transactions and objects in a directed graph format, illustrating the dynamic relationships among existing objects, transactional modifications, and newly created outputs. However, the simultaneous interdependencies in both BlockDAG and Hash-DAG structures can complicate data consistency, especially under high transaction loads where tracking multiple parent blocks may delay consensus.

In the approach proposed by Lee and Kim in [91], the integration of DAGs into blockchain sharding frameworks provides an advanced method for managing asynchronous transactional data, particularly in FL environments with non-independent and identically distributed data distributions. Each shard maintains an independent DAG ledger, enabling heterogeneous nodes to asynchronously record and aggregate model updates without serialization bottlenecks. Their architecture aligns naturally with FL, where updates frequently diverge from earlier model states.

To secure aggregation under adversarial conditions, the framework employs a novel tip selection algorithm based on model accuracy, similarity, and update multiplicity, mitigating risks such as poisoned models. While integrating a DAG enhances TPS and resilience, it also presents new challenges. For instance, careful coordination is needed to verify transaction multiplicity and maintain aggregation consistency during asynchronous updates. Additionally, if the random tip selection is poorly tuned, it can hinder model convergence. Nonetheless, empirical evaluations demonstrate that the DAG-enabled system significantly outperforms chain-based FL in robustness and accuracy.

While Lee and Kim [91] primarily design their layered DAG model for FL rather than general-purpose blockchain sharding, their work illustrates the broader potential of DAGs to facilitate scalable, parallel transaction aggregation in decentralized environments, albeit with some trade-offs in communication overhead for general transactional systems.

The adoption of DAG structures, as seen in [43,51], offers significant advantages in terms of transaction processing speed and data integrity by enabling more flexible and efficient block validation and propagation mechanisms. Nevertheless, achieving a consistent transaction history within a DAG remains challenging because there is no

straightforward method for tracking the complete sequence of interdependent transactions, making discrepancy recovery difficult.

In summary, while DAGs can enhance the flexibility and speed of blockchain transaction processing, managing complex transaction histories and resolving inconsistencies continue to pose significant challenges. These issues could compromise the long-term integrity and reliability of sharded blockchains, indicating that further research and development are necessary to fully realize the potential of DAG-based approaches in improving blockchain scalability and efficiency. Moreover, unlike traditional linear blockchains, DAG-based systems require more sophisticated conflict resolution and ordering strategies, particularly in multi-shard environments where concurrent updates may propagate with inconsistent timing. While domain-specific implementations such as FL have shown promise, general-purpose blockchain networks integrating DAGs must contend with broader coordination, synchronization, and validation overheads that can impact scalability.

7.7. Availability Enhancement

In a consortium blockchain network, the primary goal of availability enhancement is to improve the resilience of the blockchain network by serving as a contingency for shard failures. To the best of our knowledge, this mechanism is uniquely implemented in the Static Sharding approach in [71].

The work presented in Meepo [71] focuses on enhancing shard robustness through a backup mechanism called Shadow Shard-Based Recovery. In this approach, each shard is equipped with several shadow shards that act as backup servers. When a primary shard encounters issues, a consortium member can seamlessly switch to a corresponding backup shard to ensure uninterrupted blockchain operations. Zheng et al. in [71] implement shadow shards to reinforce the robustness of each shard, ensuring that the overall system continues to function despite individual component failures. While redundancy strategies improve system availability compared to centralized systems that are prone to single points of failure, managing shadow shards can increase overhead. This issue becomes increasingly evident in large-scale deployments. The frequent switching between main and backup shards could slow down operations and need more resources, which raises concerns about scalability and overall resource use.

The implementation of Shadow Shard-Based Recovery within Meepo's architecture [71] offers a practical approach to the critical challenge of maintaining high availability and reliability in blockchain networks. Despite its benefits in improving resilience, the method can complicate maintenance and elevate operating expenses as the number of shards and shadow shards grows. This trade-off between resource efficiency and robustness underscores the challenges of achieving high availability in large-scale blockchain networks.

Kronos [66] integrates rollback recovery into its cross-shard transaction framework, enabling the system to maintain liveness even when certain cross-shard transactions enter the unhappy path. If a cross-shard transaction partially commits to one input shard and another input shard subsequently determines its input is invalid, Kronos triggers an atomic rollback to revert the affected transaction. However, experimental results show that as the frequency of rollbacks increases, system TPS decreases, reflecting the overhead of handling unhappy paths. Simulation results demonstrate that Kronos remains live and continues processing transactions in the presence of rollbacks, though at reduced TPS.

Further complexity arises when transactions produce cascading effects before final commitment, such as state transitions that indirectly influence other shards or involve off-chain acknowledgments. In such cases, rolling back a single transaction may not be sufficient to fully restore system integrity unless additional compensatory actions are taken. Moreover, while rollback improves fault tolerance, it increases coordination overhead

during failure scenarios, as shards must synchronize their rollback states and confirm consistency before resuming operations. Under adversarial conditions where failures are deliberately triggered, frequent rollbacks could degrade system performance and expose shards to liveliness bottlenecks. Although Kronos' integrated rollback mechanism represents a significant advance over sharding mechanisms lacking explicit recovery protocols, ensuring efficient and fully coherent rollback under complex dependency chains remains an open engineering challenge.

Even though Meepo's Shadow Shard-Based Recovery [71] method significantly enhances system resilience, scalability may be limited by the increased complexity and resource requirements associated with managing shadow shards. As the number of primary shards grows, the associated shadow shards must also be proportionally maintained, which can lead to increased storage, communication, and synchronization overhead. This fixed redundancy model does not incorporate dynamic adaptation to traffic or failure patterns, potentially resulting in underutilized resources. Furthermore, the authors do not provide empirical analysis or simulation results evaluating performance impacts under varying network sizes or failure rates, leaving the practical limits of this mechanism unquantified. This observation highlights the ongoing trade-off between high availability and the operational expenses of large-scale blockchain networks. In contrast, Kronos [66] approaches availability through rollback-based recovery embedded in its consensus logic, which avoids shadow redundancy but introduces coordination complexity during failure handling.

7.8. Summary

Although many of the features covered in this section, such as Trust Establishment, Consensus Selection, Epoch Randomness, Cross-shard Algorithm, Cross-shard Capacity, DAG Block Structures, and Availability Enhancement, are widely utilized in general blockchain systems, they remain fundamental to the design of sharded blockchain architectures. Within sharding, these features are not simply adopted from broader blockchain frameworks. They are selectively adapted, extended, or re-engineered to address the specific demands of decentralized shard management, cross-shard consistency, and dynamic system scaling.

Most sharding approaches, whether Static, Dynamic, or Layered, rely on specialized implementations or variations of these core features to maintain TPS, resilience, and decentralization at scale. Their continued evolution is critical, particularly as sharded systems face increasingly complex operational and adversarial challenges.

In addition to these traditional approaches, recent developments have introduced AI-augmented techniques that incorporate predictive modeling, reinforcement learning, and reputation-driven strategies. While these AI-based methods are not directly rooted in the existing Static, Dynamic, or Layered Sharding approaches discussed in this section, they introduce novel features aimed at improving shard adaptability, workload prediction, security hardening, and resource optimization. As such, they represent an important emerging direction that complements and extends the evolution of sharding technologies.

The interplay between the features covered in this section, their specialization across different sharding techniques, and the gradual shift from broad to more targeted optimizations will be discussed in the next section, providing a deeper view of how sharding techniques have evolved over time.

8. Discussion

Referring to Figure 8 and the timeline infographic depicted in Figure 2, which illustrate the features of Static, Dynamic, and Layered Sharding techniques, we initially observe a broad emphasis on enhancing all relevant aspects. Over time, however, each technique

appears to narrow its focus toward a specific feature. This evolution is further confirmed by Figure 8 which illustrates the overlapping and unique features of Static, Dynamic, and Layered sharding. Static Sharding is primarily associated with Availability Enhancement and the DAG Block Structure, while Dynamic Sharding emphasizes Cross-shard Capacity. Some features are shared across techniques: the Cross-shard Algorithm appears in both Static and Layered techniques. Trust establishment, Consensus Selection, and Epoch Randomness are fundamental to all three techniques, underscoring their universal importance in the effective operation of shard systems.

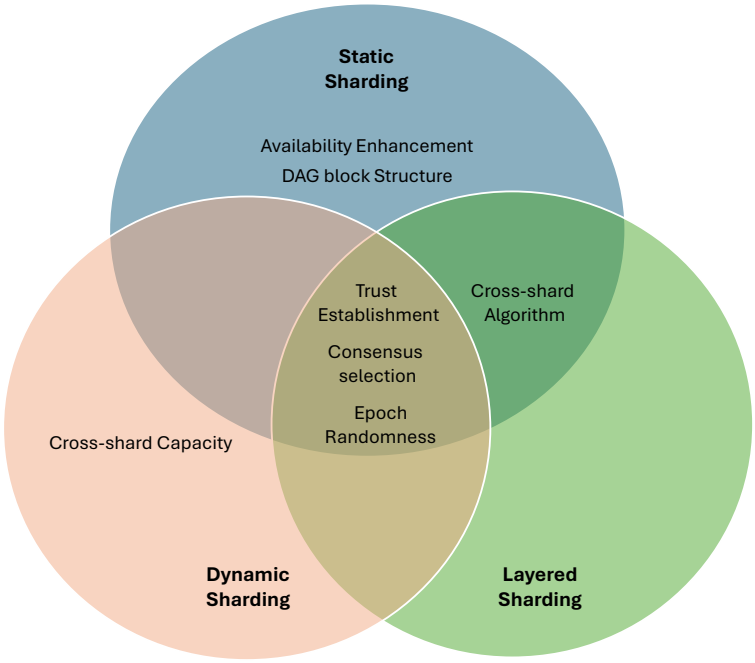

Figure 8. Three sharding techniques and their features, with overlapping features identified in the Venn diagram framework.

When sharding first came into play with efforts like Elastico [23], the goal was to improve all seven features and bring in new ways to select consensus. These early efforts often led to security issues and did not bring much improvement. Later, the introduction of Epoch Randomness aimed to make blockchain sharding more secure. However, this often slowed down performance, and many approaches merely replicated the security features already provided by existing consensus mechanisms and trust setups.

Additionally, while Availability Enhancement usually acts as a backup within group settings, adding a DAG Block Structure makes sharding more complex because of how different DAG structures are. The Cross-shard Algorithm introduces new ways to interact with different shards in the same blockchain network. However, improving Cross-shard Capacity could also enhance the approach without needing new Cross-shard algorithms or methods by simply focusing on how to manage the number of shards and their sizes.

The comparative analysis in Section 7 demonstrate that sharding-based approaches yield tangible improvements in blockchain scalability, but a direct benchmarking against state-of-the-art baselines clarifies both their potential and their limitations. Static approaches such as Elastico [23], OmniLedger [51], and RapidChain [70] improve throughput and reduce latency compared to conventional blockchains like Bitcoin (7 TPS, approximately 10 min) and Ethereum (15 TPS, approximately 19 s), yet they remain far below the industrial benchmark exemplified by VISA's 24,000 TPS. Dynamic approaches such as DYNASHARD [68] and AEROChain [55] further enhance adaptability and resilience through secure committee formation and DRL-driven migration, but they introduce non-trivial

computational and communication overheads that restrict their suitability for latency-sensitive applications. Layered approaches such as Pyramid [24] and SPRING [74] reduce cross-shard transaction ratios but still face unresolved challenges in maintaining atomicity and fairness at scale. Emerging AI-assisted designs such as PoAI and hybrid PoS-AI frameworks show measurable gains in simulated environments, yet they raise critical concerns regarding transparency, adversarial robustness, and fairness of model-driven validator selection. By situating these outcomes against both baseline systems and the gaps noted in prior reviews (Table 1), our discussion underscores that while sharding is a promising route to scalability, current implementations remain transitional and require further empirical validation before claims of robustness and generalizability can be substantiated.

Recent developments show a clear trend: instead of attempting to optimize all features simultaneously, newer sharding approaches increasingly specialize in specific feature enhancements based on their target environment. For example, Dynamic Sharding approaches have prioritized Cross-shard Capacity, while Layered Sharding approaches have focused on optimizing cross-shard transaction consistency and delegation structure which is Cross-shard Algorithm.

In parallel with these targeted optimizations, a complementary shift is emerging with the introduction of AI-augmented frameworks. Although these AI-based approaches, such as AI-Shard [49] and TBDD [78], are not directly derived from existing Static, Dynamic, or Layered Sharding approaches, the works in [49,78] introduce novel dimensions centered on predictive load management, trust-based dynamic re-sharding, and intelligent fault recovery. Their presence (e.g., [49,78]) indicates a growing need for adaptivity and proactivity in handling the increasingly volatile and complex transaction environments that modern sharded blockchains must operate within. Thus, while the foundational sharding mechanisms continue to mature through domain-specific refinements, AI-enhanced innovations are beginning to push the boundaries beyond conventional optimization, hinting at a future convergence between architectural specialization and intelligent, learning-based system adaptation.

Nevertheless, we argue that introducing fundamentally new sharding techniques, rather than merely augmenting existing ones with additional AI components, remains a more robust path for advancing blockchain scalability and resilience. Pure AI-driven optimizations often adapt at the operational layer without modifying the underlying shard formation, validation, or cross-shard interaction principles. As demonstrated by the evolution from Static to Dynamic and Layered Sharding, rethinking the sharding paradigm itself has historically enabled breakthroughs in overcoming limitations such as static load distribution, high cross-shard overhead, and rigid trust models. By designing novel sharding architectures that embed adaptivity, fault tolerance, and load balancing at the structural level rather than as external enhancements, the system can inherently support higher scalability, better security, and more predictable performance guarantees. Therefore, while AI innovations offer valuable short-term gains, the pursuit of new native sharding methodologies remains essential for sustaining long-term blockchain evolution under increasingly decentralized, adversarial, and heterogeneous conditions. We can also observe the interrelationship (e.g., [43,71]) between the number of shards, shard size, and the different sharding techniques, including Static, Dynamic, and Layered. We summarize the reviewed approaches in Section 7 focusing improvement of different features under Static, Dynamic and Layer Sharding techniques in Table 2, Table 3 and Table 4, respectively, emphasizing the technical specifics of each method. Furthermore, Table 5 provides a summary of these features and their impact on TPS and latency performance of blockchain network.

Table 2. Existing approaches in Static Sharding technique contributing to improve different features.

Types of Blockchain		**Public Blockchain**							**Consortium Blockchain**
Approaches		**Luu et al. (2016) [23]**	**Kokoris-Kogias et al. (2017) [51]**	**Zamani et al. (2018) [70]**	**Al-Bassam et al. (2018) [43]**	**Wang & Wang (2019) [42]**	**Lin et al. (2024) [67]**	**Liu et al. (2024) [66]**	**Zheng et al. (2021) [71]**
Category	**Components**								
Features	Trust Establishment	• Identity Establishment and Committee Formation • Overlay Setup for Committees	Identity Block Creation	Peer Discovery	Present but no specific method mentioned or proposed	Present but no specific method mentioned or proposed	VRF role assignment + VDF	PoW + committee verification	Present but provided by PoA
	Consensus Selection	• PoW • Intracommittee Consensus (PBFT) • Final Consensus Broadcast	ByzcoinX	• Intra-committee Consensus (Enhanced by Block pipelining) • IDA Gossiping Protocol • Synchronous Consensus	• S-BAC • Byzantine Agreement • Atomic Commit	• Chu-Ko-Nu mining • Asynchronous Consensus Zones	FBFT	Multi-phase BFT with rollback support	PoA
	Epoch Randomness	• Distributed Commit-and-XOR	• Randhound • Shard Ledger Pruning	• Protocol Reconfiguration • Cuckoo Rule/ Bounded Cuckoo Rule	N/A	N/A	VRF-based randomness (internally verifiable)	PRP seeded from PoW (internal, not publicly verifiable)	N/A
	Cross-Shard Algorithm	N/A	• Atomix • (Optional) Trust-but-verify transaction	• Cross-shard Verification • Kademlia	S-BAC	Eventual Atomicity	Relay-based metadata forwarding without rollback	Batch certification using Merkle proofs + vector commitments	• Cross-epoch • Cross-call • Partial Cross-call Merging Strategy • Replay-epoch
	Cross-Shard Capacity	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	DAG Block Structure	N/A	BlockDAG	N/A	Hash-DAG	N/A	N/A	N/A	N/A
	Availability Enhancement	N/A	N/A	N/A	N/A	N/A	N/A	Rollback recovery embedded in consensus	Shadow Shard Based Recovery
Implementation Details	Smart Contracts	N/A	N/A	N/A	Yes	N/A	Yes	N/A	Yes
	Applications	C++, Own PBFT, Amazon EC2	GO, Deterlab	GO	Python, Java, Amazon EC2	C++, Botan Cryptography, Intel IPP cryptography, own test network	GO, Harmony, Amazon EC2	Amazon EC2, Python	Go, AliCloud

Table 3. Existing approaches in Dynamic Sharding technique contributing to improve different features.

Types of Blockchain		**Public Blockchain**							**Consortium Blockchain**
Approaches		**Tao et al. (2020) [25]**	**Zhang et al. (2020) [73]**	**C. Chen et al. (2021) [69]**	**R. Chen et al. (2022) [52]**	**Tennakoon & Gramoli (2022) [53]**	**Liu et al. (2024) [68]**	**Song et al. (2025) [55]**	**Zhou et al. (2020) [72]**
Category	**Components**								
Features	Trust Establishment	Present but no specific method mentioned or proposed	Deterministic identity establishment and reconfiguration	Adopt the Louvain Algorithm	Present but no specific method mentioned or proposed	Present but no specific method mentioned or proposed	Random committee selection per epoch	Secure node assignment & PBFT (dual layer); static trust	Present but no specific method mentioned or proposed
	Consensus Selection	PoW	Standard BFT consensus	N/A	PBFT	DBFT	Hybrid intra-shard BFT + inter-shard MPC + threshold signing	PBFT at intra-shard and logical shard layers	N/A
	Epoch Randomness	Based on OmniLedger	Epoch-based resharding; randomness source unspecified	N/A	Shard Reconfiguration	Shard Committee Rotation	Internal randomness for committee selection	Epoch-based deterministic seed (not verifiable)	N/A
	Cross-shard Algorithm	N/A	Claims support; lacks architectural details	N/A	• Anchorhash • Jump Consistent Hash Algorithm	N/A	MPC + threshold signatures for atomic cross-shard validation	Relay-based via CSTM; no rollback/atomicity	N/A
	Cross-shard Capacity	• Inter-shard Merging Algorithm • Intra-shard Transaction Selection Mechanism • Parameter Unification Method	DRL-driven resharding	• Open Jackson Queueing Network Model • Shard-based Blockchain Game • Polynomial Time Algorithm • Dynamic User Distribution • Validator Redistribution Approach • User Distribution Table • User Redistribution Epoch • Validator Vote and System Reconfiguration Approach	• Node Mapping • Node Remapping	N/A	Dynamic shard splitting and merging based on live metrics	DRL-based prefixgrouped account migration	• Transaction Sharding (Boss Shard and Normal Shard) • Network Sharding • Epoch-switching Preparation • Transaction Sharding and Micro Block Generation • Full Block Generation and Synchronization
	DAG Block Structure	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Availability Enhancement	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Implementation Details	Smart Contracts	Yes	N/A	N/A	N/A	Yes	N/A	N/A	N/A
	Applications	go-Ethereum v1.8.0; AWS; Python 3.0	Tensorflow; Python 3.6	Python; Matlab	N/A	Collachain; Solidity	Python; C++	Go; Python; BlockEmulator	Omnet++

Table 4. Existing approaches in Layered Sharding technique contributing to improve different features.

Types of Blockchain		**Public Blockchain**		
Approaches		**Hong et al. (2021) [24]**	**Liu et al. (2021) [54]**	**Li et al. (2024) [74]**
Category	**Components**			
Features	Trust Establishment	Assignment System	Present but no specific method mentioned or proposed	PoW at registration only
	Consensus Selection	PoW	PoW	PBFT intra-shard consensus
	Epoch Randomness	Randomness Generation	N/A	Initial PoW bits for assignment; periodic VRF for node reshuffling (not static)
	Cross-Shard Algorithm	b-Shard, i-Shard	Actual shard and Virtual shard	DRL-based preassignment, no rollback
	Cross-Shard Capacity	N/A	N/A	N/A
	DAG Block Structure	N/A	N/A	N/A
	Availability Enhancement	N/A	N/A	N/A
Implementation Details	Smart Contracts	Yes	N/A	Yes
	Applications	Go	Python	Python, AliCloud

Table 5. Summary of the three sharding techniques and their distinguishing features in different blockchain types. In the features column, the ✓indicates the criterion is met.

Types of Blockchain		**Public Blockchain**																	**Consortium Blockchain**	
Techniques of Sharding		**Static Sharding**					**Dynamic Sharding**								**Layered Sharding**				**Static Sharding**	**Dynamic Sharding**
	Approaches	**Luu et al. (2016) [23]**	**Kokoris-Kogias et al. (2017) [51]**	**Zamani et al. (2018) [70]**	**Al-Bassam et al. (2018) [43]**	**Wang & Wang (2019) [42]**	**Lin et al. (2024) [67]**	**Liu et al. (2024) [66]**	**Tao et al. (2020) [25]**	**Zhang et al. (2020) [73]**	**C. Chen et al. (2021) [69]**	**R. Chen et al. (2022) [52]**	**Tennakoon & Gramoli (2022) [53]**	**Liu et al. (2024) [68]**	**Song et al. (2025) [55]**	**Hong et al. (2021) [24]**	**Liu et al. (2021) [54]**	**Li et al. (2024) [74]**	**Zheng et al. (2021) [71]**	**Zhou et al. (2020) [72]**
Category	**Components**																			
Features	Trust Establishment	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Consensus Selection	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	
	Epoch Randomness	✓	✓	✓			✓	✓	✓	✓		✓	✓	✓	✓	✓		✓		
	Cross-shard Algorithm		✓	✓	✓	✓	✓	✓		✓		✓		✓	✓	✓	✓	✓	✓	
	Cross-shard Capacity								✓	✓	✓	✓		✓	✓					✓
	DAG Block Structure		✓		✓															
	Availability Enhancement							✓											✓	
Performance	TPS	40	4000	7380	350	11,694.89	20,000	320,000	108	110,000	1900	N/A	14,000	8938	Estimated 1652	12,000	N/A	5444	124,583.7	1383
	Latency	103 s	1.38 s	8.7 s	69 ms	13 s	N/A	2 s	212 s	N/A	1 s	N/A	N/A	N/A	N/A	4 s	N/A	N/A	120 ms	N/A

9. Lesson Learned, Open Research Issues, and Future Directions

As shown in the previous sections, blockchain technology, especially with sharding techniques, keeps changing to meet the need for faster and more efficient networks. While much has been done to improve sharding and consensus mechanisms over the years, more research is needed as blockchain technology and performance expectations evolve along with society's needs. In this section, we share key lessons from the earlier sections and point out specific challenges that need more research in the future.

9.1. Number of Shards and Shard Size Manipulation

The manipulation of the number of shards and shard size plays a fundamental role in enhancing the scalability of blockchain networks, as covered in Section 7.5. By dynamically adjusting the number and size of shards, blockchain architectures can manage workloads more effectively, reducing bottlenecks in cross-shard communication and improving overall performance. The adjustment process seeks to balance the distribution of transactions across shards to prevent overload on any single shard and ensure that all shards contribute evenly to the processing power of the network.

The challenge lies in determining the optimal number of shards and their respective sizes based on real-time network demands. In static environments, a fixed shard configuration might suffice, but in highly dynamic settings where transaction volumes and validator participation vary, a more flexible approach is needed. Dynamic Sharding must react to changes in TPS and validator availability, potentially splitting or merging shards as needed. However, frequent changes in shard composition can increase the complexity of cross-shard communication and consistency, as well as introduce overhead in terms of computation and synchronization.

The future direction of research on shard manipulation might focus on predictive algorithms capable of anticipating network load fluctuations and adjusting shard parameters in real time to maintain efficiency. Additionally, the interplay between cross-shard communication and shard size needs to be more thoroughly explored, as larger shards may reduce communication overhead but could also lead to inefficiencies if transactions within the shard become too complex to manage effectively. By improving these shard manipulation strategies, blockchain networks can enhance their scalability without sacrificing performance.

9.2. Hybrid Allocation Approaches for Robust Static Sharding

In Section 7.1, Static Sharding is recognized for its reliability in ensuring secure, predictable blockchain operations. Static Sharding assigns nodes to fixed shards and maintains this structure regardless of changes in network conditions. This approach allows for the establishment of trust in a stable environment, where nodes within a shard can reliably verify each other's identities through predefined mechanisms like PoW or other consensus mechanism. This enhances security, as the fixed nature of Static Sharding reduces the opportunities for malicious actors to exploit the approach by frequently changing shards.

However, the primary limitation of Static Sharding is its inflexibility. While this fixed structure is beneficial for security, it limits the blockchain's ability to adapt to dynamic conditions, such as sudden surges in transaction volume or shifts in network topology. In such scenarios, the mechanism in Static Sharding may struggle to distribute workload efficiently, potentially leading to congestion in some shards while others remain underutilized. To address these challenges, future research could explore hybrid models that incorporate aspects of both Static and Dynamic Sharding. Such models would allow blockchain networks to maintain the security and trust benefits of Static Sharding while introducing limited flexibility to adapt to changing conditions. For example, a Static Sharding

could include mechanisms that temporarily adjust shard boundaries or redistribute certain transactions during periods of high demand. These enhancements would help optimize the performance of Static Sharding approaches without compromising their inherent stability and security.

9.3. Advanced Optimization and Intelligent Transaction Routing in Dynamic Sharding

In Section 7.5, existing works in Dynamic Sharding detailed a flexible and scalable approach to managing blockchain networks by continuously adjusting the composition of shards based on real-time demands. Unlike Static Sharding, Dynamic Sharding is designed to adapt to fluctuating transaction volumes and validator participation, allowing for the reconfiguration of shards as needed to optimize performance. This flexibility is key to ensuring that the network remains efficient and responsive, even as conditions change unpredictably.

However, Dynamic Sharding introduces new complexities, particularly in maintaining consistency across shards. As shards are created, merged, or split to handle varying workloads, ensuring that transactions are properly validated and synchronized across all shards becomes a more challenging task. Without proper coordination, these frequent adjustments could lead to issues such as double-spending, inconsistencies in transaction history, or delays in transaction processing. Moreover, Dynamic Sharding often requires sophisticated consensus mechanisms that can accommodate the frequent changes in shard composition, which can be computationally expensive and difficult to implement.

Future research in this area could focus on optimizing the mechanisms used to adjust shard configurations dynamically. Specifically, improving the algorithms that determine when and how to adjust shard boundaries could reduce the overhead associated with dynamic reconfiguration. Additionally, more advanced transaction routing protocols could help streamline cross-shard communication, ensuring that transactions are processed quickly and efficiently, even as shard configurations shift. By refining these approaches, Dynamic Sharding can continue to offer a scalable approach for blockchain networks while minimizing the potential downsides of its inherent complexity.

9.4. Layered Sharding Enhancement Through Efficient Coordination and Proactive Shard Operations

In Section 7.4, existing works in Layered Sharding represent an advanced strategy for optimizing transaction processing across multiple layers of shards. By dividing the blockchain network into different layers of shards, each responsible for specific types of transactions or operations, Layered Sharding can enhance both TPS and efficiency. This technique allows for parallel processing of transactions, as different shards operate semi-independently, reducing the likelihood of bottlenecks and ensuring that the approach can handle high volumes of transactions without sacrificing speed or security.

However, the interdependencies between different layers of shards present challenges in maintaining consistency and ensuring smooth communication across layers. When transactions span multiple shards or layers, coordinating the validation and finalization of these transactions requires robust algorithms to manage the process effectively. Without such coordination, there is a risk of transaction failures, delays, or inconsistencies in the ledger. Moreover, as the number of layers increases, the complexity of managing these interactions grows, potentially leading to inefficiencies if not carefully controlled.

Enhancing Layered Sharding techniques could involve refining the Cross-shard Algorithms that manage transactions across different layers, ensuring that transactions are validated and finalized as efficiently as possible. Additionally, developing more intelligent and dynamically manageable routing mechanisms to direct transactions to the appropriate shards or layers could further optimize performance. Another potential area of focus is the

use of machine learning or predictive analytics to anticipate transaction flows and adjust shard operations preemptively, reducing the likelihood of congestion or delays. These advancements would help Layered Sharding become a more scalable and reliable approach for high-performance blockchain networks.

In conclusion, while Layered Sharding offers significant potential for improving blockchain scalability, continued research and development are needed to address the inherent challenges associated with cross-layer communication and transaction coordination. By refining these techniques, blockchain networks can fully realize the benefits of Layered Sharding, enhancing both TPS and efficiency while maintaining the security and integrity of the ledger.

9.5. AI-Driven Techniques for Trust, Sharding, and Scalability in Blockchain

AI techniques, particularly DRL and machine learning, have been applied across various aspects of sharded blockchain operation, including node classification, task scheduling, behavior prediction, and protocol parameter tuning as observed in Sections 7.2, 7.3 and 7.5. While these approaches improve responsiveness and performance, they also raise concerns regarding transparency, fairness, and robustness. Key research directions include the development of explainable AI methods to enhance interpretability, online learning frameworks to adapt to dynamic environments, and safeguards against biased training data that could lead to unfair validator assignments or centralized influence. Moreover, the computational cost and tuning complexity of deep models highlight the need for efficient, self-adjusting alternatives that balance intelligence with scalability.

9.5.1. Machine Learning for Dynamic Trust and Reputation Management

AI and machine learning models are leveraged for node classification, reputation assessment, and dynamic identity verification. These methods enable real-time detection of anomalous or malicious behaviors, providing greater flexibility than static, rule-based systems. However, as seen in recent studies (see Section 7.1), there remain challenges in ensuring the transparency and fairness of AI-driven trust assessments. Many current approaches are difficult to audit, potentially introducing bias or centralization risks, especially when training data are skewed or adversarial. Thus, future research should prioritize explainable, auditable, and decentralized AI trust mechanisms that support decentralized governance, are resistant to adversarial manipulation, and incorporate online learning frameworks to adapt to changing environments.

9.5.2. Reinforcement Learning for Adaptive Shard Configuration

DRL models have shown promise in optimizing shard configurations, transaction assignments, and protocol parameter tuning in response to fluctuating workloads. Such adaptability, discussed in Sections 7.3 and 7.5, improves throughput and resource allocation. However, the computational demands and tuning complexity of deep models can limit scalability. These systems may also overfit to recent network states, reducing robustness to sudden changes. There is a growing need for lightweight, self-tuning algorithms that balance responsiveness and stability. Research should focus on designing real-time monitoring layers and distributed control mechanisms that support consistent and coordinated decision-making. Evaluating the cost and impact of frequent reconfiguration, including synchronization delays and system overhead, is essential to ensure that local adaptations align with broader performance goals.

9.5.3. Predictive Analytics and Clustering for Efficient Cross-Shard Communication

AI-powered clustering and prediction techniques are used to reduce inter-shard dependencies by grouping highly interactive nodes or predicting communication patterns

(see Sections 7.4 and 7.5). These approaches can lower transaction latency and network congestion, but they may introduce computational overhead, react slowly to rapid or adversarial behavioral changes, and their robustness under dynamic or adversarial conditions requires further evaluation. Future work should investigate asynchronous or optimistic protocols for cross-shard transactions, lightweight verification schemes, and advanced workload profiling techniques. Mechanisms that cluster accounts or nodes based on shared characteristics show promise, but must be validated under diverse operational scenarios.

9.5.4. AI-Augmented Data Structures for Parallelism and Consistency

The integration of AI with advanced data structures, such as DAGs (see Section 7.6), enables asynchronous and parallel transaction processing in sharded blockchains. AI methods can support conflict resolution, efficient state management, and tip selection. Nonetheless, challenges remain in maintaining transactional integrity across concurrent updates, as well as managing synchronization and conflict resolution at scale. Further research should investigate scalable, lightweight protocols for maintaining consistency in DAG-based sharded environments, and optimize traversal and tip selection strategies. Lightweight synchronization protocols and conflict resolution mechanisms tailored for DAG usage in sharded settings are also important areas for further study.

9.5.5. Summary and Emerging Directions

In addition to addressing trust, AI integration, communication efficiency, shard adaptability, and consistency in data structures, future research should also consider the role of FL frameworks. These approaches support decentralized training of AI models while preserving data privacy, aligning well with the distributed nature of blockchain systems. Furthermore, addressing scalability and interoperability remains essential, particularly in application domains such as healthcare and the Internet of Things.

Emerging technologies such as Agentic AI [92] offer promising avenues for enhancing sharding mechanisms under real-time and complex operational demands. By decomposing high-level objectives into smaller, manageable sub-tasks, Agentic AI can facilitate intelligent shard configuration, adaptive cross-shard communication, and real-time anomaly detection. These capabilities are particularly relevant for latency-sensitive applications, including smart factories and haptic communication systems, where autonomous, fine-grained control is essential for reliable and secure performance.

Our review further highlights that the integration of AI with sharding is not merely conceptual but has already been supported by experimental studies. For example, SPRING [74] demonstrates that DRL-based reconfiguration reduces cross-shard transactions and improves TPS, while TBDD [78] reports up to 13% TPS gains through a trust-driven DRL framework. Similarly, MBPOBS [83] employs reinforcement learning with Gaussian Process Regression and achieves 1.1–1.26× TPS improvements compared to baseline models. AEROChain [55] and SkyChain [73] also employ DRL-based migration and re-sharding strategies validated in simulation, focusing on TPS, latency, and security metrics. These cases illustrate that AI-assisted sharding approaches are already backed by empirical evidence, though further real-world validation remains essential for assessing their scalability and robustness.

In summary, AI techniques are enabling advances in trust management, adaptive shard configuration, efficient cross-shard communication, and state consistency in sharded blockchains. However, these benefits come with new challenges, including the need for transparency, fairness, and robustness, particularly in adversarial or dynamic environments. Research should continue to focus on explainable AI, decentralized and privacy-

preserving learning (e.g., FL), and real-time adaptive control, especially for sensitive and large-scale applications.

10. Conclusions

Our review has classified sharding into three distinct techniques: Static, Dynamic, and Layered, along with the features inherent to each sharding mechanism. It highlights how the number of shards and shard sizes are crucial in managing nodes and transactions, subsequently impacting performance metrics such as TPS and latency. For instance, Static Sharding, constrained by a predetermined number of shards and sizes, can only accommodate a limited number of nodes and transactions. This limitation highlights the need to explore Dynamic and Layered Sharding, but also acknowledges that these approaches may still be insufficient when it comes to catering to the emerging blockchain-integrated, low-latency and high-reliability demanding applications (e.g., Tactile Internet and Haptic Feedback and Smart Factories). Furthermore, AI integration can break down complex tasks into smaller, manageable units, offering a promising approach to enhancing sharding mechanisms and meeting the real-time demands of applications. While AI innovations offer valuable short-term gains, it remains essential to rethink the underlying shard formation, validation, and interaction logic in order to embed adaptivity, fault tolerance, and load balancing directly into the architecture. In addition, compared with prior state-of-the-art surveys, this review provides broader and more up-to-date coverage. Earlier reviews often remained conceptual, limited their focus to static sharding, or lacked integration of evolving trends. By contrast, our study consolidates the classifications of Static, Dynamic, and Layered Sharding, incorporates emerging AI-assisted mechanisms, and uniquely integrates standardization efforts such as ITU-T Recommendation F.751.19 (2024) and ETSI GR PDL-032 (2025). This positions our work as a more complete and forward-looking contribution. While numerous studies report improvements in TPS and reduced latency through sharding-based approaches, these claims are often supported by limited or incomplete experimental evidence. In particular, many works do not provide detailed descriptions of their experimental settings, datasets, or analyses, which makes independent verification difficult. This gap underscores the need for more rigorous benchmarking across diverse environments and transparent reporting of evaluation methodologies in future research. Looking ahead, future research should focus on developing lightweight cross-shard communication protocols, dynamic trust reassessment models, and transparent AI-driven frameworks that enhance fairness and resilience. Furthermore, more prescriptive standards are needed to guide practical implementations and ensure interoperability. By highlighting both comparative advantages over prior surveys and outlining these future directions, this review not only synthesizes the current state of blockchain sharding but also charts a clear roadmap for advancing scalability, security, and practical deployment. Finally, we identified here key open issues and future research challenges within the context of blockchain sharding. This work will inspire and guide future researchers in exploring more advanced sharding techniques and approaches to improve blockchain scalability and efficiency.

Author Contributions: Conceptualization, M.M., S.T., S.H.S.N. and R.K.P.; methodology, M.M., S.T., S.H.S.N. and R.K.P.; investigation, M.M., S.T., S.H.S.N. and R.K.P.; writing—original draft preparation, M.M., S.T., S.H.S.N. and R.K.P.; writing—review and editing, M.M., S.T., S.H.S.N., R.K.P. and G.M.L.; visualization, M.M., S.T., S.H.S.N. and R.K.P.; supervision, M.M., S.T., S.H.S.N. and R.K.P.; funding acquisition, G.M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Universiti Teknologi Brunei (UTB), Brunei Darussalam.

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: During the preparation of this work the author(s) used OpenAI's ChatGPT o3-mini (https://chatgpt.com/) in order to improve the readability and grammatical correction of some parts of this manuscript. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication. This research has been supported by Universiti Teknologi Brunei (UTB), Brunei Darussalam.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Haber, S.; Stornetta, W.S. How to Time-Stamp a Digital Document. In *Advances in Cryptology—CRYPTO '90*; Springer: Berlin/Heidelberg, Germany, 1991; pp. 437–455.
2. González, C.D.; Mena, D.F.; Muñoz, A.M.; Rojas, O.; Sosa-Gómez, G. Electronic Voting System Using an Enterprise Blockchain. *Appl. Sci.* **2022**, *12*, 531. [CrossRef]
3. Wang, Q.; Huang, L.; Chen, S.; Xiang, Y. Blockchain Enables Your Bill Safer. *IEEE Internet Things J.* **2020**, *9*, 14162–14171. [CrossRef]
4. Esposito, C.; Ficco, M.; Gupta, B.B. Blockchain-based authentication and authorization for smart city applications. *Inf. Process. Manag.* **2021**, *58*, 102468. [CrossRef]
5. Shuaib, M.; Alam, S.; Shabbir Alam, M.; Shahnawaz Nasir, M. Self-sovereign identity for healthcare using blockchain. *Mater. Today Proc.* **2021**, *81*, 203–207. [CrossRef]
6. Dunphy, P.; Petitcolas, F.A. A first look at identity management schemes on the blockchain. *IEEE Secur. Priv.* **2018**, *16*, 20–29. [CrossRef]
7. Kundu, S.; Acharya, U.S.; Mukherjee, S. Identity Management in Internet of Things: A Software-Defined Networking Approach. In Proceedings of the 2nd International Conference on Communication, Devices and Computing, Haldia, India, 14–15 March 2019; Volume 602. [CrossRef]
8. Martinson, P. *Estonia—The Digital Republic Secured by Blockchain*; PwC: Tallinn, Estonia, 2019; pp. 1–12.
9. Hakak, S.; Khan, W.Z.; Gilkar, G.A.; Imran, M.; Guizani, N. Securing Smart Cities through Blockchain Technology: Architecture, Requirements, and Challenges. *IEEE Netw.* **2020**, *34*, 8–14. [CrossRef]
10. Alammary, A.; Alhazmi, S.; Almasri, M.; Gillani, S. Blockchain-Based Applications in Education: A Systematic Review. *Appl. Sci.* **2019**, *9*, 2400. [CrossRef]
11. Shaikh, E.; Mohammad, N. Applications of Blockchain Technology for Smart Cities. In Proceedings of the 4th International Conference on Inventive Systems and Control, ICISC 2020, Coimbatore, India, 8–10 January 2020; pp. 186–191. [CrossRef]
12. Treiblmaier, H.; Rejeb, A.; Strebinger, A. Blockchain as a Driver for Smart City Development: Application Fields and a Comprehensive Research Agenda. *Smart Cities* **2020**, *3*, 853–872. [CrossRef]
13. Shen, C.; Pena-Mora, F. Blockchain for Cities—A Systematic Literature Review. *IEEE Access* **2018**, *6*, 76787–76819. [CrossRef]
14. Tselios, C.; Politis, I.; Kotsopoulos, S. Enhancing SDN security for iot-related deployments through blockchain. In Proceedings of the 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2017, Berlin, Germany, 6–8 November 2017; pp. 303–308. [CrossRef]
15. Nam Nguyen, H.; Anh Tran, H.; Fowler, S.; Souihi, S. A survey of Blockchain technologies applied to software-defined networking: Research challenges and solutions. *IET Wirel. Sens. Syst.* **2021**, *11*, 233–247. [CrossRef]
16. Aujla, G.S.; Singh, M.; Bose, A.; Kumar, N.; Han, G.; Buyya, R. BlockSDN: Blockchain as a Service for Software Defined Networking in Smart City Applications. *IEEE Netw.* **2020**, *34*, 83–91. [CrossRef]
17. Rathee, T.; Singh, P. A systematic literature mapping on secure identity management using blockchain technology. *J. King Saud Univ.-Comput. Inf. Sci.* **2021**, *34*, 5782–5796. [CrossRef]
18. Lu, Q.; Paik, H.Y. Design Patterns for Blockchain-based Self-Sovereign Identity. *arXiv* **2005**, arXiv:2005.12112.
19. Ordano, E.; Meilich, A.; Jardi, Y.; Araoz, M. *Decentraland White Paper*; 2017 ; p. 15. Available online: https://decentraland.org/whitepaper.pdf (accessed on 15 December 2024).
20. Nasir, M.H.; Arshad, J.; Khan, M.M.; Fatima, M.; Salah, K.; Jayaraman, R. Scalable blockchains—A systematic review. *Future Gener. Comput. Syst.* **2022**, *126*, 136–162. [CrossRef]
21. Siriwardena, P. The Mystery Behind Block Time—Medium.facilelogin.com. Available online: https://medium.facilelogin.com/the-mystery-behind-block-time-63351e35603a#:~:text=In%20bitcoin%2C%20the%20expected%20block,between%2010%20to%2019%20seconds (accessed on 15 December 2024).
22. Hafid, A.; Hafid, A.S.; Samih, M. Scaling Blockchains: A Comprehensive Survey. *IEEE Access* **2020**, *8*, 125244–125262. [CrossRef]
23. Luu, L.; Narayanan, V.; Zheng, C.; Baweja, K.; Gilbert, S.; Saxena, P. A secure sharding protocol for open blockchains. In Proceedings of the ACM Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 17–30. [CrossRef]

24. Hong, Z.; Guo, S.; Li, P.; Chen, W. Pyramid: A layered sharding blockchain system. In Proceedings of the IEEE INFOCOM, Vancouver, BC, Canada, 10–13 May 2021. [CrossRef]
25. Tao, Y.; Li, B.; Jiang, J.; Ng, H.C.; Wang, C.; Li, B. On sharding open blockchains with smart contracts. In Proceedings of the International Conference on Data Engineering, Dallas, TX, USA, 20–24 April 2020; pp. 1357–1368. [CrossRef]
26. Syta, E.; Jovanovic, P.; Kogias, E.K.; Gailly, N.; Gasser, L.; Khoffi, I.; Fischer, M.J.; Ford, B. Scalable Bias-Resistant Distributed Randomness. In Proceedings of the IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–26 May 2017; pp. 444–460. [CrossRef]
27. Zhou, Q.; Huang, H.; Zheng, Z.; Bian, J. Solutions to Scalability of Blockchain: A Survey. *IEEE Access* **2020**, *8*, 16440–16455. [CrossRef]
28. Sanka, A.I.; Cheung, R.C. A systematic review of blockchain scalability: Issues, solutions, analysis and future research. *J. Netw. Comput. Appl.* **2021**, *195*, 103232. [CrossRef]
29. Khan, D.; Jung, L.T.; Hashmani, M.A. Systematic literature review of challenges in blockchain scalability. *Appl. Sci.* **2021**, *11*, 9372. [CrossRef]
30. Shalaby, S.; Abdellatif, A.A.; Al-Ali, A.; Mohamed, A.; Erbad, A.; Guizani, M. Performance Evaluation of Hyperledger Fabric. In Proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies, ICIoT 2020, Doha, Qatar, 2–5 February 2020; pp. 608–613. [CrossRef]
31. Chaudhry, N.; Yousaf, M.M. Consensus Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities. In Proceedings of the 2018 12th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 19–21 December 2018; pp. 54–63. [CrossRef]
32. Lashkari, B.; Musilek, P. A Comprehensive Review of Blockchain Consensus Mechanisms. *IEEE Access* **2021**, *9*, 43620–43652. [CrossRef]
33. Fu, X.; Wang, H.; Shi, P. A survey of Blockchain consensus algorithms: Mechanism, design and applications. *Sci. China Inf. Sci.* **2021**, *64*, 121101. [CrossRef]
34. Gramoli, V. From blockchain consensus back to Byzantine consensus. *Future Gener. Comput. Syst.* **2020**, *107*, 760–769. [CrossRef]
35. Oyinloye, D.P.; Teh, J.S.; Jamil, N.; Alawida, M. Blockchain Consensus: An Overview of Alternative Protocols. *Symmetry* **2021**, *13*, 1363. [CrossRef]
36. Dreyer, J.; Fischer, M.; Tönjes, R. Performance analysis of hyperledger fabric 2.0 blockchain platform. In Proceedings of the CCIoT 2020—2020 Cloud Continuum Services for Smart IoT Systems, Part of SenSys 2020, Virtual Event, 16–19 November 2020; pp. 32–38. [CrossRef]
37. Ampel, B.; Patton, M.; Chen, H. Performance modeling of hyperledger sawtooth blockchain. In Proceedings of the 2019 IEEE International Conference on Intelligence and Security Informatics, ISI 2019, Shenzhen, China, 1–3 July 2019; pp. 59–61. [CrossRef]
38. Khan, D.; Jung, L.T.; Hashmani, M.A.; Cheong, M.K. Empirical Performance Analysis of Hyperledger LTS for Small and Medium Enterprises. *Sensors* **2022**, *22*, 915. [CrossRef] [PubMed]
39. Xie, J.; Zhang, K.; Lu, Y.L.; Zhang, Y. Resource-Efficient DAG Blockchain with Sharding for 6G Networks. *IEEE Netw.* **2022**, *36*, 189–196. [CrossRef]
40. Tekeoglu, A.; Ahmed, N. TangoChain: A lightweight distributed ledger for internet of things devices in smart cities. In Proceedings of the 5th IEEE International Smart Cities Conference, ISC2 2019, Casablanca, Morocco, 14–17 October 2019; pp. 18–21. [CrossRef]
41. Doku, R.; Rawat, D.B.; Garuba, M.; Njilla, L. LightChain: On the lightweight blockchain for the internet-of-things. In Proceedings of the 2019 IEEE International Conference on Smart Computing, SMARTCOMP 2019, Washington, DC, USA, 12–15 June 2019; pp. 444–448. [CrossRef]
42. Wang, J.; Wang, H. Monoxide: Scale out blockchain with asynchronous consensus zones. In Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2019, Boston, MA, USA, 26–28 February 2019; pp. 95–112.
43. Al-Bassam, M.; Sonnino, A.; Bano, S.; Hrycyszyn, D.; Danezis, G. Chainspace: A Sharded Smart Contracts Platform. In Proceedings of the Network and Distributed Systems Security (NDSS) Symposium 2018, San Diego, CA, USA, 18–21 February 2018. [CrossRef]
44. Shahid, A.R.; Pissinou, N.; Staier, C.; Kwan, R. Sensor-chain: A lightweight scalable blockchain framework for internet of things. In Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Atlanta, GA, USA, 14–17 July 2019; pp. 1154–1161.
45. Cheng, F.; Xiao, J.; Liu, C.; Zhang, S.; Zhou, Y.; Li, B.; Li, B.; Jin, H. Shardag: Scaling dag-based blockchains via adaptive sharding. In Proceedings of the 2024 IEEE 40th International Conference on Data Engineering (ICDE), Utrecht, The Netherlands, 13–16 May 2024; pp. 2068–2081.
46. Wang, Q.; Yu, J.; Chen, S.; Xiang, Y. SoK: DAG-based blockchain systems. *ACM Comput. Surv.* **2023**, *55*, 1–38. [CrossRef]

47. Li, Y.; Wang, J.; Zhang, H. A survey of state-of-the-art sharding blockchains: Models, components, and attack surfaces. *J. Netw. Comput. Appl.* **2023**, *217*, 103686. [CrossRef]
48. Zhang, H.; Xue, J. FLPSHARD: A Flexible and Efficient Blockchain Sharding Solution for IIoT. *Electronics* **2025**, *14*, 961. [CrossRef]
49. Wang, J.; Li, Y.; Wu, Y.; Zheng, W.; Zhou, S.; Xiong, X. Blockchain sharding scheme based on generative AI and DRL: Applied to building internet of things. *Internet Things Cyber-Phys. Syst.* **2024**, *4*, 333–349. [CrossRef]
50. *ITU-T Recommendation F.751.19*; Framework and Requirements for Distributed Ledger Technology Based on Sharding Technique. International Telecommunication Union: Geneva, Switzerland, 2024; ITU-T Series F: Non-Telephone Telecommunication Services, Multimedia Services.
51. Kokoris-Kogias, E.; Jovanovic, P.; Gasser, L.; Gailly, N.; Ford, B. OmniLedger: A Secure, Scale-Out, Decentralized Ledger. *IACR Cryptol. ePrint Arch.* **2017**, 406. Available online: https://eprint.iacr.org/2017/406.pdf (accessed on 15 December 2024).
52. Chen, R.; Wang, L.; Peng, C.; Zhu, R. An Effective Sharding Consensus Algorithm for Blockchain Systems. *Electronics* **2022**, *11*, 2597. [CrossRef]
53. Tennakoon, D.; Gramoli, V. Dynamic Blockchain Sharding. In Proceedings of the 5th International Symposium on Foundations and Applications of Blockchain 2022 (FAB 2022), Berkeley, CA, USA, 3 June 2022. [CrossRef]
54. Liu, Y.; Sun, H.; Song, X.; Chen, Z. OverlapShard: Overlap-based Sharding Mechanism. In Proceedings of the IEEE Symposium on Computers and Communications, Athens, Greece, 5–8 September 2021. [CrossRef]
55. Song, M.; Li, P.; Zhou, B.; Yin, S.; Xiao, Z.; Long, J. AERO: Enhancing sharding blockchain via deep reinforcement learning for account migration. In Proceedings of the ACM on Web Conference 2025, Sydney, Australia, 28 April–2 May 2025; pp. 706–716.
56. Yu, G.; Wang, X.; Yu, K.; Ni, W.; Zhang, J.A.; Liu, R.P. Survey: Sharding in Blockchains. *IEEE Access* **2020**, *8*, 14155–14181. [CrossRef]
57. Yu, G.; Wang, X.; Yu, K.; Ni, W.; Zhang, J.A.; Ren Ping Liu, J.A. Scaling-out blockchains with sharding: An extensive survey. In *Blockchains for Network Security: Principles, Technologies and Applications*; IET: London, UK, 2024.
58. Liu, Y.; Liu, J.; Salles, M.A.V.; Zhang, Z.; Li, T.; Hu, B.; Henglein, F.; Lu, R. Building Blocks of Sharding Blockchain Systems: Concepts, Approaches, and Open Problems. *Comput. Sci. Rev.* **2021**, *46*, 100513. [CrossRef]
59. Hashim, F.; Shuaib, K.; Zaki, N. Sharding for Scalable Blockchain Networks. *SN Comput. Sci.* **2023**, *4*, 2. [CrossRef]
60. Xiao, J.; Liang, W.; Cai, J.; Zhu, H.; Li, X.; Xie, S. An Investigation of Blockchain-Based Sharding. In *Smart Computing and Communication*; Qiu, M., Lu, Z., Zhang, C., Eds.; Springer: Cham, Switzerland, 2023; pp. 695–704.
61. Zhang, L.; Wang, Y.; Ding, Y.; Liang, H.; Yang, C.; Li, C. Sharding Technologies in Blockchain: Basics, State of the Art, and Challenges. In *Blockchain and Trustworthy Systems*; Chen, J., Wen, B., Chen, T., Eds.; Springer: Singapore, 2024; pp. 242–255.
62. Yang, Q.; Huang, H.; Yin, Z.; Lin, Y.; Chen, Q.; Luo, X.; Li, T.; Liu, X.; Zheng, Z. The State-of-the-Art and Promising Future of Blockchain Sharding. *IEEE Commun. Mag.* **2025**, *63*, 192–198. [CrossRef]
63. Liu, X.; Xie, H.; Yan, Z.; Liang, X. A survey on blockchain sharding. *ISA Trans.* **2023**, *141*, 30–43. [CrossRef] [PubMed]
64. *Permissioned Distributed Ledger (PDL); Artificial Intelligence for Permissioned Distributed Ledger*; Etsi Group Report gr pdl 032 v1.1.1; European Telecommunications Standards Institute (ETSI): Sophia Antipolis, France, 2025.
65. Kitchenham, B.; Brereton, O.P.; Budgen, D.; Turner, M.; Bailey, J.; Linkman, S. Systematic literature reviews in software engineering–a systematic literature review. *Inf. Softw. Technol.* **2009**, *51*, 7–15. [CrossRef]
66. Liu, Y.; Liu, A.; Lu, Y.; Pan, Z.; Li, Y.; Liu, J.; Bian, S.; Conti, M. Kronos: A secure and generic sharding blockchain consensus with optimized overhead. *Cryptol. ePrint Arch.* **2024**. Available online: https://eprint.iacr.org/2024/206.pdf (accessed on 15 December 2024).
67. Lin, Y.; Li, M.; Wei, Q.; Liu, Y.; Goh, S.M.R.; Zhang, J. DL-Chain: Scalable and Stable Blockchain Sharding with High Concurrency via Dual-Layer Consensus. *arXiv* **2024**, arXiv:2407.06882.
68. Liu, A.; Chen, J.; He, K.; Du, R.; Xu, J.; Wu, C.; Feng, Y.; Li, T.; Ma, J. Dynashard: Secure and adaptive blockchain sharding protocol with hybrid consensus and dynamic shard management. *IEEE Internet Things J.* **2024**, *12*, 5462–5475. [CrossRef]
69. Chen, C.; Ma, Q.; Chen, X.; Huang, J. User distributions in shard-based blockchain network: Queueing modeling, game analysis, and protocol design. In Proceedings of the Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing, Shanghai, China, 26–29 July 2021; pp. 221–230.
70. Zamani, M.; Movahedi, M.; Raykova, M. RapidChain: Scaling Blockchain via Full Sharding. In Proceedings of the ACM Conference on Computer and Communications Security (CCS), Toronto, ON, Canada, 15–19 October 2018; pp. 1–38.
71. Zheng, P.; Xu, Q.; Zheng, Z.; Zhou, Z.; Yan, Y.; Zhang, H. Meepo: Sharded consortium blockchain. In Proceedings of the International Conference on Data Engineering, Chania, Greece, 19–22 April 2021; pp. 1847–1852. [CrossRef]
72. Zhou, Z.; Qiu, Z.; Yu, Q.; Chen, H. A Dynamic Sharding Protocol Design for Consortium Blockchains. In Proceedings of the 2020 IEEE International Conference on Big Data, Big Data 2020, Atlanta, GA, USA, 10–13 December 2020; pp. 2590–2595. [CrossRef]
73. Zhang, J.; Hong, Z.; Qiu, X.; Zhan, Y.; Guo, S.; Chen, W. Skychain: A deep reinforcement learning-empowered dynamic blockchain sharding system. In Proceedings of the 49th International Conference on Parallel Processing, Edmonton, AB, Canada, 17–20 August 2020; pp. 1–11.

74. Li, P.; Song, M.; Xing, M.; Xiao, Z.; Ding, Q.; Guan, S.; Long, J. Spring: Improving the throughput of sharding blockchain via deep reinforcement learning based state placement. In Proceedings of the ACM Web Conference 2024, Singapore, 13–17 May 2024; pp. 2836–2846.
75. *ISO/TR 24332:2025*; Information and Documentation—Blockchain and Distributed Ledger Technology (DLT) in Relation to Authoritative Records, Records Systems and Records Management. International Organization for Standardization: Geneva, Switzerland, 2025.
76. Standards—IEEE Blockchain Technical Community. Available online: https://blockchain.ieee.org/standards (accessed on 15 December 2024).
77. Sabadello, M.; Zagidulin, D. Decentralized Identifier Resolution (DID Resolution) v0.3. W3C Working Draft. Available online: https://www.w3.org/TR/did-resolution/ (accessed on 22 May 2025).
78. Zhang, Z.; Yu, G.; Sun, C.; Wang, X.; Wang, Y.; Zhang, M.; Ni, W.; Liu, R.P.; Reeves, A.; Georgalas, N. TbDd: A new trust-based, DRL-driven framework for blockchain sharding in IoT. *Comput. Netw.* **2024**, *244*, 110343. [CrossRef]
79. Alzoubi, Y.I.; Mishra, A. Blockchain consensus mechanisms comparison in fog computing: A systematic review. *ICT Express* **2024**, *10*, 342–373. [CrossRef]
80. Deval, V.; Dwivedi, V.K.; Dixit, A.; Norta, A.; Shah, S.A.; Sharma, R.; Draheim, D. Mobile Smart Contracts: Exploring Scalability Challenges and Consensus Mechanisms. *IEEE Access* **2024**, *12*, 34265–34288. [CrossRef]
81. Mezouari, H.E.; Omary, F. Enhancing Blockchain Sharding Consensus through Artificial Intelligence and Hybridizing with Proof of Stake. *Int. J. Commun. Netw. Inf. Secur. (IJCNIS)* **2025**, *17*, 223–237.
82. Chen, J.; Duan, K.; Zhang, R.; Zeng, L.; Wang, W. An AI based super nodes selection algorithm in blockchain networks. *arXiv* **2018**, arXiv:1808.00216. [CrossRef]
83. Li, Y.; Wan, J.; Liu, C.; Li, L. Model-Based Throughput Optimization for Blockchain Sharding System. In Proceedings of the 2024 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA), Kaifeng, China, 30 October–2 November 2024; pp. 2054–2059.
84. Micali, S.; Rabin, M.; Vadhan, S. Verifiable random functions. In Proceedings of the 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039), New York, NY, USA, 17–19 October 1999; pp. 120–130.
85. Zhang, Y.; Yang, J.; Lei, H.; Bao, Z.; Lu, N.; Shi, W.; Chen, B. Verifiable Random Function Schemes Based on SM2 Digital Signature Algorithm and Its Applications for Committee Elections. *IEEE Open J. Comput. Soc.* **2024**, *5*, 480–490. [CrossRef]
86. Awerbuch, B.; Scheideler, C. Towards a Scalable and Robust DHT. *Theory Comput. Syst.* **2009**, *45*, 234–260. [CrossRef]
87. Lin, Y.; Gao, Z.; Du, H.; Kang, J.; Niyato, D.; Wang, Q.; Ruan, J.; Wan, S. DRL-based adaptive sharding for blockchain-based federated learning. *IEEE Trans. Commun.* **2023**, *71*, 5992–6004. [CrossRef]
88. Chen, T.; Lin, H.; Wu, L.; Wang, X. Optimized Block-K Clustering IoT Clustering and Blockchain Sharding Strategy Using Deep Reinforcement Learning. In Proceedings of the 2024 IEEE Cyber Science and Technology Congress (CyberSciTech), Boracay Island, Philippines, 5–8 November 2024; pp. 45–51.
89. Wu, H.Y.; Yang, X.; Yue, C.; Paik, H.Y.; Kanhere, S.S. Chain or DAG? Underlying data structures, architectures, topologies and consensus in distributed ledger technology: A review, taxonomy and research issues. *J. Syst. Archit.* **2022**, *131*, 102720. [CrossRef]
90. Lu, X.; Jiang, C.; Wang, P. A Survey on Consensus Algorithms of Blockchain Based on DAG. In Proceedings of the 2024 6th Blockchain and Internet of Things Conference, Fukuoka, Japan, 19–21 July 2024; pp. 50–58.
91. Lee, J.; Kim, W. Dag-based blockchain sharding for secure federated learning with non-iid data. *Sensors* **2022**, *22*, 8263. [CrossRef]
92. Acharya, D.B.; Kuppan, K.; Divya, B. Agentic AI: Autonomous Intelligence for Complex Goals—A Comprehensive Survey. *IEEE Access* **2025**, *13*, 18912–18936. [CrossRef]