Deep Learning for Failure Recovery and Efficiency Enhancement in Hypercube Networks

Turkan Ahmed Khaleel

Computer Engineering Department, College of Engineering, University of Mosul, Mosul, Iraq

*Corresponding author E-mail: turkan@uomosul.edu.iq

https://doi.org/10.29072/basjs.20260110

ARTICLE INFO

Received: 12 October 2025

Accepted: 17 February 2026

Published: 30 April 2026

Keywords:

Hypercube Networks; Deep Learning; Fault Tolerance; Node Failures; Link Failures; Failure Recovery.

ABSTRACT

Hypercube networks are widely used in parallel and distributed computing systems due to their high scalability and inherent fault tolerance. However, node or link failures lead to significant performance degradation, negatively impacting data throughput and network availability. This research presents a novel deep learning-based framework for intelligent fault recovery in hypercube networks, utilizing dimensionally sensitive convolutional neural networks. The proposed model is designed to operate across different dimensions of the hypercube (2D, 4D, and 6D) and is trained using artificially generated fault scenarios involving node and link failures. Experimental results demonstrate a significant performance improvement compared to traditional methods. In the 2-dimensional hypercube, the recovery time decreased to 0.1905 seconds, the throughput increased to 0.5, and the response time was reduced to 1.9608 seconds, compared to 3.8462 seconds in the traditional approach. In the 4-dimensional cube configuration, recovery time improved from 0.2319 seconds to 0.1825 seconds, throughput increased from 0.1484 to 0.25, and response time decreased from 6.3116 seconds to 3.8462 seconds. In the 6-dimensional configuration, cycles per second decreased from 0.2014 to 0.1709, throughput increased from 0.0405 to 0.0938, and response time decreased from 19.7913 seconds to 9.6386 seconds. These results confirm the effectiveness of deep learning in enabling adaptive, resilient, and self-healing networking, thereby enhancing the reliability and performance of large-scale distributed systems in fault-prone environments.

1. Introduction

Hypercube networks comprise a basic and common interconnected system in parallel and distributed computing systems. The number of nodes in an n-dimensional hypercube is n^2, and each node has n neighbors. This design creates a very symmetric network that has a logarithmic diameter and high connectivity [1]. Due to these features, hypercube networks perform well in high-performance computing systems, scalable multiprocessor systems, and large-scale data centers where high-performance and stability of communication are critical [2-3]. Although the benefits of hypercube networks are obvious, the security of their work is complicated with the increasing size of the network. Failure of nodes or links results in poor performance, communication loss, and reduced reliability. Among the earliest research methods for fault tolerance in computer systems were probabilistic modeling and redundancy-based methods.

They are based on the forecast of failures and reducing their consequences through replication of components or data [3-7]. These methods were rather reactive and not very successful in dealing with dynamic and complex failure patterns, despite enhancing reliability to a certain degree. Other research focused on structural changes to enhance fault resilience in topologies of network structures. The routing schemes proposed by Samavi and Khadivi [8], which direct traffic to prevent faulty nodes and topological indices, were used in the studies by Sarkar et al. [9] and Gao and Ahmed [10] to evaluate the reliability and connectivity of hierarchical and hypercube-based networks. The disjoint cycle embeddings and the spanning structures have been investigated by Wu and Sabir [11,12] and were found to be significant in maintaining connectivity in the event of failures of either the vertex or edge. Liu [13,14] discussed the vertex fault-tolerant cycles and conditional fault model of augmented and hierarchical hypercubes.

Zhao and Hao [15], Fang et al. [16], and Guo et al. [17] analyzed fault diagnosis, irregularity index, and reliability models in a hierarchical hypercube. These works generally highlight the role of structural and topological characteristics in determining the reliability of networks, but the majority of methods are static, reactive, and do not have the ability to learn adaptively. Machine learning (ML) also began creating applications that design networks and fault management tools with the emergence of smart systems. Seidu et al. [18] demonstrated that latent ML models, which use Latin Hypercube Sampling in combination with Bayesian Optimization, can be utilized successfully in identifying faults in power systems. Gupta et al.

[19] used ML to enhance the energy efficiency of hypercube-based wireless sensor networks to prove that learning based optimization can be useful in structured network structures.

At the system level, reliable multiplexing schemes [7, 22, 23] and blockchain-based secure aggregation [20, 21] have also enhanced the reliability and fault tolerance of hierarchical networks based on a hypercube. Nonetheless, these methods are, to a large extent, rule-based or topology-based and do not actively rely on predictive intelligence to perform proactive fault recovery. Specifically, adaptive fault recovery can be achieved in Deep Learning (DL), in particular, Convolutional Neural Networks (CNNs).

The CNNs are best at extracting features in a hierarchy, performing cheap computations, strong training, and scaling; thus are best suited for real-time and dimension-sensitive fault recovery in large networks [4,5,8,9]. Nonetheless, none of the previous studies systematically implements CNN-based DL in hypercube fault recovery or enhancing its performance.

The present paper presents a CNN-based DL system that can be used to intelligently repair performance and faults in the hypercube networks. The framework is a hybrid of topology-sensitive feature encoding and dimension-sensitive convolutional learning. It is able to perform proactive and self-healing behaviors over different hypercube sizes. In contrast to earlier systems, it combines structural consciousness, fault tolerance, connectivity, and efficiency in a single model of adaptation that minimizes the gap between old structural methods and modern intelligence based on data. In contrast to their previous ML-based and topology-based counterparts, our CNN framework is a unification of topology awareness, dimension sensitivity, and proactive fault recovery in one adaptive framework, which previous studies have not managed to do.

2. Key Contributions

This paper proposes a new failure recovery algorithm of hypercube networks that uses dimension-sensitive DL with CNNs. The old techniques were based on deterministic routing policy or on a heuristic on an individual route basis. By contrast, our model exploits the topological characteristics of multi-dimensional hypercubes to produce dynamic recovery schemes hypercube of diverse synthetic failure conditions. The main enhancement is the fact that the model can scale in hypercube dimensions (2D, 4D, 6D) without having to use a different recovery logic in each case. Besides, it is the first attempt at using the DL model, which can adjust its behavior to the dimensional structure of a hypercube, that allows proactive, large-scale, self-healing communication in high-dimensional distributed systems. The framework

transforms the traditional reactive failure handling engines into a predictive, topology-aware failure handling system, which is a major shift compared to past models.

3. Literature Review

Hypercube networks have received a lot of research on fault tolerance and reliability. Initial research on the topic focused mainly on structural approaches. For example, Samavi and Khadivi [8] recommended the avoidance of faulty node routing strategies. To assess the topological indices, Sarkar et al. [9] and Gao and Ahmed [10] tested reliability and connectivity. Although they emphasized the importance of structural properties, these works employed non-adaptive, non-dynamic models. Other research tackled the improvement of connectivity and embedding of cycles in the hypercube variants.

Wu and Sabir [11,12] examined disjoint cycles and spanning structures that maintain the network by ensuring that the network does not suffer disconnection due to faults. This work was later developed by Liu [13,14], who looked at vertex-fault-tolerant cycles and conditional fault models. However, these methods lack adaptive/learning-based mechanisms. Hierarchy that has been used in fault diagnosis and reliability modeling in hierarchical hypercubes has used the irregularity indices and diagnosability metrics [15-17]. These approaches are mostly reactive, pre-established, and do not scale.

Fault detection and optimization in machine learning applications such as power systems [18] and wireless sensor networks [19] have been explored, but adaptive fault recovery of hypercube networks has not explored yet. There are system-level reliability boosts, including blockchain-based aggregation [20,21], adaptive link recovery [22], and multiplexing schemes [7,23], which enhance robustness, but they are largely rule-based and topology-driven, with no predictive intelligence to facilitate real-time recovery. This framework incorporates structural awareness, adaptive learning, and performance optimization. CNNs, unlike Graph Neural Networks (GNNs), scale well to the grid-like topology of a hypercube and require less computational work, stable training, and can easily be scaled to higher-dimensional networks. CNNs are therefore useful and can serve as an effective solution when fault recovery tasks are required in real-time, and the dimension is sensitive.

Table 1: Summary of Literature Review

Ref.	Network Type	Methodology	Focus	Limitation
[8], [9], [10]	Hierarchical / Hypercube	Topological indices	Fault tolerance	Static, non-adaptive
[11], [12], [13], [14]	Hypercube variants	Cycle embedding, spanning structures	Connectivity under faults	No learning capability
[15]–[17]	Hierarchical hypercube	Fault diagnosis models	Diagnosability	Reactive, predefined
[18]	Power systems	Hybrid ML (LHS + BO)	Fault detection	Not network-oriented, no adaptive recovery
[19]	WSN (hypercube-based)	ML optimization	Energy efficiency	Non-adaptive, no-fault recovery
[20], [21]	Hypercube-based systems	Blockchain mechanisms	Secure recovery	Rule-based, non-predictive
[7], [22], [23]	Hierarchical hypercube	Multiplexing schemes	Reliability	No predictive capability

4. Motivation

In contrast to the previous work, which either dealt with topological enhancements or endeavored to provide static recovery, the model presented in this study is the first to integrate dimension-sensitive failure handling and, based upon learning, recovery in hypercube networks. The hypercube networks have the advantage of being scalable, symmetric, and efficient in routing, making them suitable for parallel and distributed systems. However, they are easily affected by node and link failures; thus, in n high-dimensional systems, such failures can seriously degrade their performance, resulting in latency, throughput, and extended recovery times. More traditional recovery mechanisms are generally either reactive or static, and are usually unable to deal with concurrent/complex failures effectively, or even respond in real-time to different failure patterns.

The shortcoming serves as the basis to drive the inclusion of DL solutions, especially CNNs, to facilitate proactive intelligent failure recovery. CNNs can learn structural patterns and generalize across many failure cases, allowing them to convert damaged network topologies into trained ones easily. The system will apply DL models to broken adjacency matrices and obtain recovery paths. This will minimize the time to recover and to enhance the throughput and resilience of the system. CNNs will read damaged adjacency matrices and generate recovery routes, resulting in less recovery latency, a greater throughput, and an overall increase in resiliency.

The open-endedness allows incorporating such a learning-based model into hypercube-based infrastructures, which brings the adaptive and self-healing abilities in the hypercube-based systems, a necessary precondition in achieving robust and intelligent distributed systems. Table 2 shows the significant distinctions between conventional recovery mechanisms and the suggested CNN-based model. In this comparative analysis, we can see how our approach can be further developed and how it can be adapted and scaled to a wide range of hypercube dimensions and failure conditions.

Table 2: The Comparative Analysis of Conventional and CNN-Based Recovery Systems

Feature	Traditional Recovery	CNN-Based Recovery Systems
Recovery Strategy	Static	Adaptive, Learned
Failure Detection	Manual / Reactive	Predictive mediante Deep Learning
Topology Sensitivity	Fixed	Dimension-aware
High failure performance	Limited	Robust
Scalability	Poor	Excellent

5. Objective

The main goal of this research is to build an intelligent, scalable, and adaptive mechanism for the failure recovery of the hypercube networks based on DL methods. More precisely, the research focuses on designing and constructing dimension-aware CNN models that will explore, learn, and successfully reconstruct patterns of outages of nodes and links to ensure network connectivity with the lowest latency and maximum throughput. In this work, an attempt is made to narrow the distance between static recovery methods that are widely used and intelligent systems, ensuring adaptability relying on the data that will help to find a new way of protecting the network in a changing (dynamic) environment.

The framework tests and validates DL opportunities to provide fast, robust, and resilient self-healing communication infrastructure systems in distributed environments at scale through extensive simulations and scenarios in 2D, 4D, and 6D hypercube topologies.

6. Failure Recovery in Hypercube Networks

Hypercube networks are known to be the most widely used distribution networks and the least expensive to implement, topology-wise. They have symmetrical topology, low diameter, and high path redundancy, forming a perfect model of fault-tolerant distributed systems. Nevertheless, node and/or link failure may dramatically affect the system's

communication and reliability, despite its original strong nature. Innovative recovery systems should be built to identify and overcome such failures in the ongoing process to ensure uninterrupted operations. A four-dimensional hypercube (n=4) of size 16 [see Figure1] shows how. Each node of the mesh is linked to four neighbors.

The dimensionality is a tradeoff between the complexity of the structure and network resilience since each new dimension creates exponentially more ways, which increases fault tolerance. Specifically, the 4D may result in multiple alternate routes at the risk of congestion, even when multiple nodes or links fail, allowing the system to maintain communication despite not being entirely disconnected. In Figure 1, the operational units are colored blue, whereas the failed (to be) components are labeled red.

Active links are marked using blue lines, and failures are marked using red dashed lines. For example, the dead state of nodes 5 and 9 isolates neighboring paths and highlights how node failure affects not only its neighbors but also spreads to other nodes through their connections and undermines the performance of a network. This situation makes it evident how elastic recovery plans are essential because they dynamically reroute to restore connectivity.

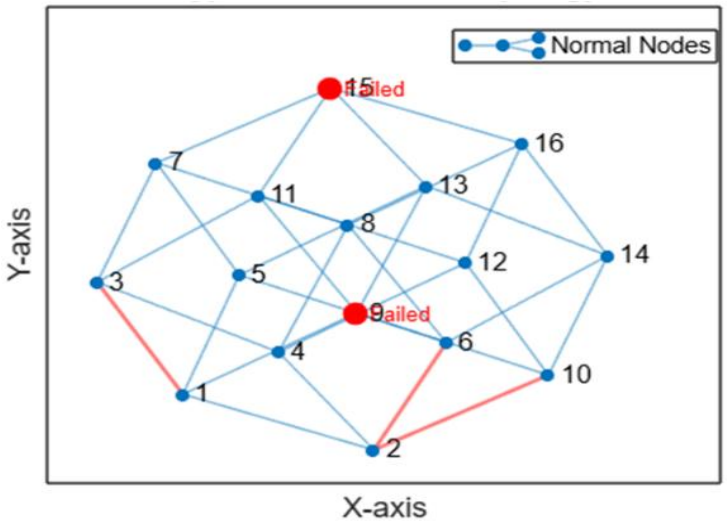


Figure 1: Hypercube Network Topology (n=4)

Hypercube networks have their way of dealing with such difficulties by using fault-aware algorithms so that they behave differently according to the type of failure they detect. As

the subsequent subsections show, these mechanisms can be classified into node and link failures.

a. Node Failures

The failure of nodes may be a result of the failure of the hardware, software failure, or a power outage. Every node in a 4D hypercube will be connected with four neighbors. When a node fails, it not only goes away, but also removes its four connections. This reduces the redundancy of the network and its overall reliability. Adaptive routing algorithms identify a failed node and re-route traffic to be redirected to other functioning nodes. The hypercube has dimensional storage so that despite numerous failures of nodes, several working paths are still accessible, hence improving the level of recovery and speed.

b. Link Failures

Link failures take place when communication between two intact nodes is simultaneously broken. Link failures can result from physical damage, intrusion, or overcrowding. In failures, link-state surveilling schemes may identify damaged links and set into motion re-routing activities. The 4D topology permits each node to have multiple connections with its neighbors individually, and thus, the effect of any broken connection is minimal, and this topology offers high fault tolerance. Figure 2 shows a 4-dimensional hypercube network with 16 nodes, which shows how the failure of nodes and links may affect network connectivity. The nodes in blue symbolize the operational (active) parts, whereas the nodes in red symbolize the failed parts, which have ceased to perform as desired. The thick black lines represent operational anomalies between nodes that vary by Hamming distance of the hypercube.

On the other hand, red dashed lines reflect failed linkages interrupted by node or communication breakages. The break in the connection attempt shown in the graph illustrates the problem of how defects in a few nodes can lead to many broken links, thus lowering the overall reliability of the communication system. This illustration shows the significance of planning innovative, fault-tolerant routing elements that can identify such failures and readjust communication links dynamically to ensure that communication is preserved.

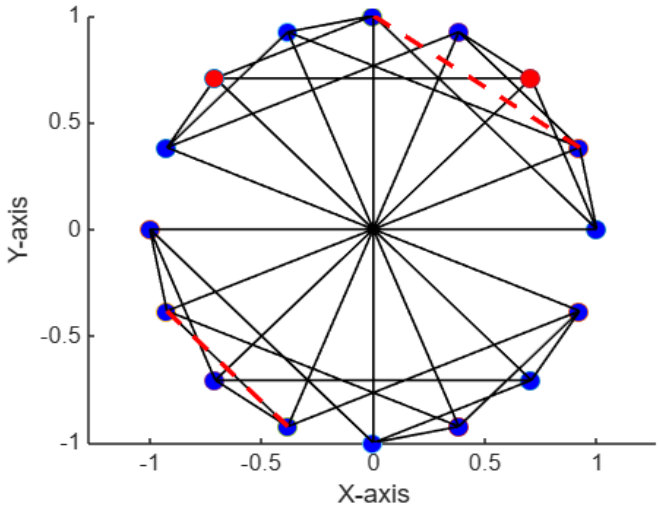


Figure 2: Node and link failure of a 4-dimensional hypercube network

7. DL Based Hypercube Network Adaptive Fault Recovery

DL is an emerging methodology for network fault management, providing adaptive, predictive, and scalable solutions that fit complex topologies, like hypercube networks. In contrast to conventional stationary techniques, DL-based approaches can learn real-time network dynamics, so that systems can learn to identify, label, and react to anomalies, so that faults are self-detected. This paper has demonstrated that hybrid models implementing artificial neural networks, Latin hypercube sampling, and Bayesian optimization effectively enhance the accuracy of fault detection in communication systems [5]. These models, albeit not applicable to hypercube networks, show how effective DL is in dealing with the complexities associated with faults in large infrastructures. The fault tolerance problem in hypercube networks has been investigated probabilistically [2] and by structural means (edge partitioning) [4], the latter being used to obtain enhanced reliability but not dynamicity. Structural fault joints like even-cycle fault resilience [24], fault diameter assessment of structures [25], and performance maximization of message passing across hypercube structures [26].

The combination of these developments gives a high potential for integrating DL algorithms, which can read the pattern of failures and act optimally on the networks. The DL method used in the study is not only an advanced networking addition but also a strategic way of addressing inherent constraints in conventional recovery methods. The long-term objective of the proposed system is to take advantage of the strengths of DL, i.e., pattern recognition, predictive and dynamic adaptations, better fault tolerance, scalability, and energy efficiency of a hypercube network, to eventually transform the hypercube networks into a communication infrastructure that is intelligent and self-healing.

8. Simulation Setting and Proposed Methodology

The presented methodology proposes a systematic pattern to fabricate, train, and test the data of the failure recovery framework in the hypercube network using DL. Significant elements of the methodology are mentioned below:

a. Hypercube Network Construct

Adjacency matrices generate the hypercube topologies of 2D, 4D, and 6D networks, each containing 2d nodes (i.e., 4, 16, and 64, respectively). Simulating communication in scalable distributed systems is based on these structures, as illustrated in Figure 3 .

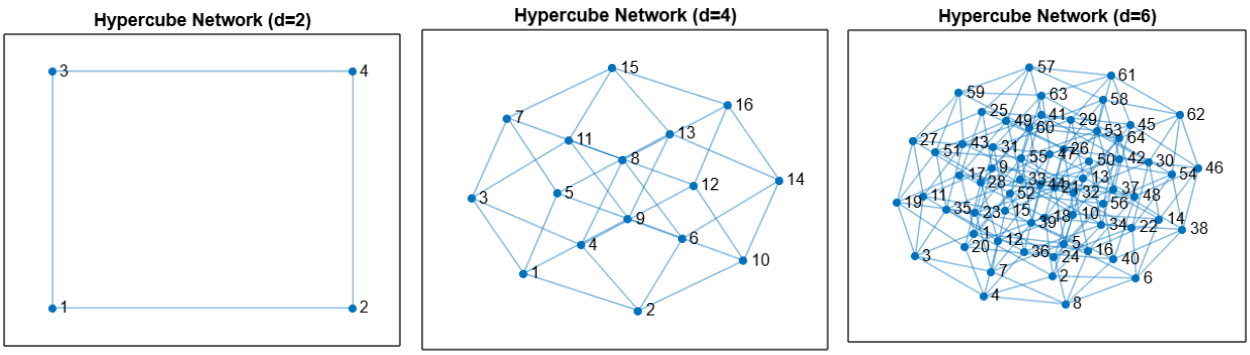


Figure 3: Hypercube topologies of 2D, 4D, and 6D networks

b. Failure Scenario Modeling and Simulation

To test the sensitivity of the system and its resistance to route failures at different magnitudes, three scenarios were adopted where the dimensions of the network and the number of simulated failures varied. In 2D (Scenario 1), a node failure rate of 0.1 and a link failure rate of 0.05 were assumed. Scenario 2 (4D) implied the high failure probability of nodes (0.2) and links (0.1). Scenario 3 (6D) was the worst-case scenario, with the highest node failure and link rates of (0.3 and 0.15, respectively). The experiments with such scenarios aimed to create realistic operational conditions and rigorously test the strength of the proposed routing and recovery mechanisms, as shown in Table 3.

c. Preparing Synthetic Datasets

Training samples are created by using random failure patterns and corrupting clean hypercube networks by generating large sample patterns. The input samples are a corruption of an adjacency matrix corresponding to a failed state of a network, and the target is the failure-free structure. Using this in-synthetic dataset, the CNN can learn to relate failed and functional states of the network, as shown in Table 3.

d. Architecture Design and Supervised Learning Strategy of CNN

Different dimensions are provided with different CNNs whose input size corresponds to the corresponding adjacency matrix (e.g., [4x4x1] in 2D, [16x16x1] in 4D, and [64x64x1] in 6D), as shown in Table 3. The trained models apply supervised learning to use mean squared error (MSE) loss and the Adam optimizer. Both models learn to estimate the original topology using damaged, but partially complete, input. The trained CNN model is used with a failed network, and the resultant network is returned during testing; a predicted adjacency matrix is then produced. This matrix is then post-processed (thresholded and symmetrized) to constitute a legitimate undirected graph. There are three important metrics to assess recovery performance: Recovery Time (s), throughput (individual links active/active links total), and Latency (s) (mean average shortest path length). The DL technique is compared with a conventional topology-aware failure masking technique. To demonstrate the improvements brought about by the proposed framework, bar plots are generated for all three scenarios, so as to clearly show the comparison between the values of recovery time, throughput, and latency.

Table 3 summarizes the set of simulation parameters and experimental conditions under which the proposed failure recovery framework was evaluated at various dimensions of hypercubes. All the CNN models used were identical in all dimensions, and the input dimensions were normalized according to the respective adjacency matrices. All the models included three layers of a convolutional neural network (3x3 kernels) and batch normalization. The two fully connected layers were fed with the output, then the network's topology was reconstructed with a sigmoid activation. This architecture allowed this model to generalize structural patterns in different failure cases.

Table 3: Experimental Set-ups and CNN System to Recover the Failure in Hypercube Networks

Parameter	Scenario 1 (2D)	Scenario 2 (4D)	Scenario 3 (6D)
Number of Dimensions (d)	2	4	6
Number of Nodes	4	16	64
Node Failure Rate	0.1	0.2	0.3
Link Failure Rate	0.05	0.10	0.15
Input Matrix Size	4×4	16×16	64×64
CNN Input Size	4×4×1	16×16×1	64×64×1
CNN Layers	multicolumn{3}{c}{3 Conv Layers (3×3, ReLU) + Batch Norm + 2 Dense Layers (128, N×N, Sigmoid)}		
Loss Function	multicolumn{3}{c}{Mean Squared Error (MSE)}		
Optimizer	multicolumn{3}{c}{Adam (learning rate = 0.001)}		
Training Samples	2000	3000	4000
Evaluation Metrics	multicolumn{3}{c}{Recovery Time, Throughput, Latency}		

9. Results and Discussion

In this paper, the performance of the framework is contrasted with a conventional recovery mechanism in three network structures, 2-D, 4-D, and 6-D hypercubes. In each case, different node and link failure rates are investigated, and the comparison is made with the use of three important measurements, which are recovery time, throughput, and latency.

a. Scenario 1: Hypercube 2D

In the 2D hypercube case, the node failure rate is established at 0.1, and the link failure rate is established at 0.05. In this scenario, the classical recovery strategy is able to reach a recovery time of 0.2204 -1, throughput of 0.25, and a latency of 3.8462 -1. Conversely, the CNN-based recovery mechanism has better results in all the measured metrics. In particular, recovery time is decreased to 0.1998 -1, or 9.4 -1. Moreover, the throughput doubled (0.25 to 0.50), but the latency reduced by an estimated 49% to reach 1.9608s .

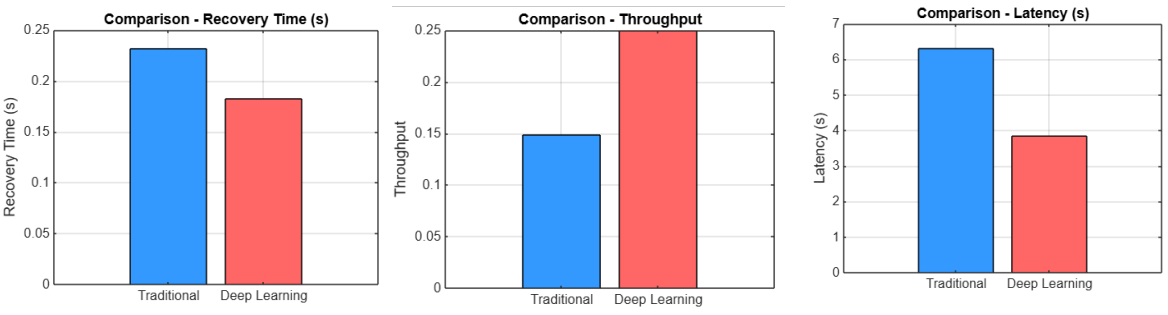
The findings suggest that despite the relatively mild failure conditions in low-dimensional hypercube networks, the CNN-based model can learn the recovery patterns with considerable efficiency and react more effectively than traditional patterns. The mechanical comparison of the recovery time, throughput, and latency is shown in Table 4, whereas the performance differences between the two methods are visualized in Figure 4(a), 4(b), and 4(c).

(a) (b) (c)

Figure 4: Comparison of Recovery Time, Throughput, and Latency between the Traditional and CNN-Based Approaches in 2D Hypercube

9.2 Scenario 2: Hypercube 4D

In the second scenario, a hypercube network with a high failure ratio is considered. The node and link failure rates are set at 0.2 and 0.1, respectively. With this setup, the standard mechanism of recovery takes a recovery time of 0.2876 s, a throughput of 0.1484, and a latency of 6.3116 s. The model presented in this study, after being trained on 3000 samples with CNN, can reach a recovery time of 0.180696s, an improved throughput of 0.25, and a lower latency of 3.8462s. These results reflect a 37% decrease in recovery time, 1.68 times more throughput, and 39%. decrease in latency compared to the traditional approach. The improvements observed indicate that the CNN-based recovery framework can be scaled to large network dimensionality and failure rates. The comparison between the 4D approach and the traditional approach is shown in Table 4 and illustrated in Figures 5(a), 5(b), and 5(c).



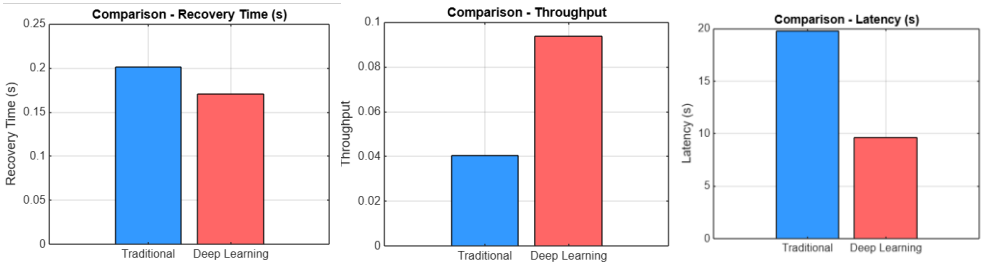
(a) (b) (c)

Figure 5: Comparison of Recovery Time, Throughput, and Latency between the Traditional and CNN-Based Approaches in 4D Hypercube

b. Scenario 3: Hypercube 6D

In the third scenario, the high-dimensional network is a 6D hypercube network with high failure rate conditions. The rates of node failures are 0.3, and link failure is 0.15. Using the traditional recovery method, the system requires a time of 0.2898 to recover, and it has a throughput of 0.0405 and a latency of 19.7913. The suggested CNN-based solution, in turn, reduces the recovery time to 0.1514s, increases throughput to 0.0938, and decreases latency to 9.6386s. These values indicate that recovery time is decreased by 47.8%, throughput doubled, and latency fell by approximately 51%. This gain is increased with the increase in the dimensionality of the network, which means that the CNN technique is particularly efficient in the case of intricate and large-scale topology of a hypercube .

Table 4 and Figures 6(a), 6(b), 6(c) show the comparisons of the numerical and graphic results of the 6D scenario.



(a) (b) (c)

Figure 6: The Comparison of Recovery Time, Throughput, and Latency of the Traditional and CNN-Based Approaches in 6D Hypercube

The general comparison of the three scenarios is presented in Table 4, which demonstrates that the CNN-based recovery framework always scores higher than the traditional approach in all of the measured aspects. The approach suggested in this paper minimizes recovery time, throughput, and Latency. The profitability increases with both the dimension of the hypercube and the probability of failure, which proves that the CNN model observes the patterns of faults as a matter of dimension and adjusts its risk mitigation practice to them. Although the results prove the viability and efficiency of the new approach, it is important to remember that the test was conducted in controlled simulated conditions. Repeating the experiment and statistical analysis, and creating a hypercube of higher dimension, would help confirm the accuracy and generalizability of the results. Nonetheless, the results provided have a firm ground in the sense that the incorporation of DL into hypercube fault recovery enhances efficiency and flexibility.

Table 4: Aggregate Comparison Among Scenarios

Hypercube Dimension	Metric	Traditional	Deep Learning	Improvement
2D	Recovery Time (s)	0.2204	0.1998	9.4% faster
	Throughput	0.25	0.50	2x higher
	Latency (s)	3.8462	1.9608	49% reduction
4D	Recovery Time (s)	0.2876	0.1806	37% faster
	Throughput	0.1484	0.25	1.68x higher
	Latency (s)	6.3116	3.8462	39% reduction
6D	Recovery Time (s)	0.2898	0.1514	47.8% faster
	Throughput	0.0405	0.0938	2.3x higher
	Latency (s)	19.7913	9.6386	51% reduction

10. Conclusion

This paper demonstrates that integrating topology-sensitive hypercube representation with DL provides a powerful, trustworthy technique of fault tolerance in distributed computing, particularly in failure-prone systems. The architecture is easily scalable to large network sizes; its performance is not affected, and hence can be used in large and complex systems. It has an adaptive, predictive layer of recovery that reduces recovery time, recovery latency, increases throughput, and systematically excels the traditional fault-management strategies. Also, the model becomes more effective in larger networks and more stringent failure situations, which is why it should be used in large-scale deployments. Despite being tested only in controlled simulations, the findings can be applicable in practice in distributed systems in the real world and provide a sound base for a future study of intelligent, self-adaptive fault-tolerant networking solutions.

References

[1] M. Abd El Barr and F. Gebali, "Reliability analysis and fault tolerance for hypercube multi-computer networks," *Information Sciences*, vol. 276, pp. 295–318, 2014, https://doi.org/10.1016/j.ins.2013.10.031.

[2] R. W. Hung, "The connectivity of DVcube networks: A survey," *Mathematics*, vol. 13, no. 11, art. no. 1836, 2025, https://doi.org/10.3390/math13111836.

[3] J. Chen and Y. He, "Hypercube network fault tolerance: A probabilistic approach," *Journal of Interconnection Networks*, vol. 6, no. 1, pp. 1–14, Jan. 2005.

[4] X. Liu, B. Cheng, Y. Wang, J. Yu, and J. Fan, "Enhancing fault tolerance of balanced hypercube networks by the edge partition method," *Theoretical Computer Science*, vol. 986, art. no. 114340, 2023.

[5] H. Dong, Z. Zhang, and Y. Li, "A theoretical model for reliability evaluation of half-hypercube networks," *Journal of Systems Architecture*, vol. 137, art. no. 102693, 2023.

[6] X. Wang and E. Sabir, "Super spanning connectivity of the generalized hypercube network," *Theoretical Computer Science*, vol. 1029, art. no. 115038, 2025.

[7] X. Liu, B. Cheng, Y. Wang, J. Yu, and J. Fan, "A highly reliable multiplexing scheme in hypercube-structured hierarchical networks," *IEEE Transactions on Computers*, vol. 74, no. 10, pp. 3462–3475, 2025.

[8] S. Samavi and P. Khadivi, "Fault-tolerant routing in hypercube networks through avoidance of faulty nodes," *Computers & Electrical Engineering*, vol. 77, pp. 127–138, 2019.

[9] P. Sarkar, N. De, and A. Pal, "Topological indices of hierarchical hypercube networks for fault tolerance," *Journal of Computational and Theoretical Nanoscience*, vol. 18, no. 2, pp. 389–397, 2021.

[10] T. Gao and I. Ahmed, "Reliability of hypercube-based networks using topological indices," *Applied Mathematics and Computation*, vol. 399, art. no. 125969, 2021.

[11] W. Wu and E. Sabir, "On embedding spanning disjoint cycles in hypercube networks with prescribed edges," *Axioms*, vol. 12, no. 9, art. no. 861, 2023, https://doi.org/10.3390/axioms12090861.

[12] W. Wu and E. Sabir, "Embedding disjoint cycles in hypercube networks with prescribed edges," *International Journal of Computer Mathematics*, vol. 100, no. 3, pp. 601–615, 2023.

[13] M. Liu, "Vertex-fault-tolerant cycle embedding in hypercube networks," *Journal of Parallel and Distributed Computing*, vol. 180, pp. 23–35, 2024.

[14] M. Liu, "Even-cycle fault-free h-super edge connected augmented hypercubes under conditional fault model," *Scientific Reports*, vol. 15, no. 1, art. no. 2502, 2025.

[15] S. Zhao and R. Hao, "Reliability analysis of hierarchical hypercube networks using topological models," *Journal of Network and Computer Applications*, vol. 133, pp. 75–85, 2019.

[16] J. Fang, H. Yu, and L. Zhang, "Irregularity indices in hierarchical hypercube networks," *Physica A*, vol. 531, art. no. 121756, 2019.

[17] Y. Guo, L. Wang, and F. Liu, "Parallel diagnosis of faults in hypercube networks under test," *The Journal of Supercomputing*, vol. 75, pp. 8433–8451, 2019.

[18] Y. Seidu, E. Twumasi, and E. A. Frimpong, "Hybrid optimized artificial neural network using Latin Hypercube Sampling and Bayesian Optimization for detection, classification, and location of faults in transmission lines," *AIMS Electronics and Electrical Engineering*, vol. 8, no. 4, pp. 498–531, 2024, https://doi.org/10.3934/electreng.2024024.

[19] N. Gupta, K. S. Vaisla, and R. Kumar, "Design of a structured hypercube network chip topology model for energy efficiency in wireless sensor networks using machine

learning," *SN Computer Science*, vol. 2, art. no. 376, 2021, https://doi.org/10.1007/s42979-021-00766-7.

[20] G. C. Ismayilov and C. Özturan, "Trustless privacy-preserving data aggregation on Ethereum with hypercube network topology," *Computer Communications*, vol. 230, art. no. 108009, 2025.

[21] J. Ali, S. Ali, M. Babar, H. Naeem, and S. W. Kim, "Smart blockchain-enabled adaptive and reliable link recovery for IoT," *Future Generation Computer Systems*, vol. 141, pp. 483–494, 2023.

[22] Z. Wang, H. Xu, and Q. Chen, "On-the-fly failures handling during distributed training with in-memory checkpoints," in *Proceedings of IEEE Big Data*, pp. 1384–1393, 2023.

[23] Y. Chen and E. Sabir, "Spanning edge cyclability of enhanced hypercube networks," *Parallel Processing Letters*, vol. 35, no. 1–2, art. no. 2550006, 2025.

التعلم العميق لاسترداد الأعطال وتعزيز الكفاءة في شبكات المكعب الفائق

توركان احمد خليل

قسم هندسة الحاسوب، كلية الهندسة، جامعة الموصل، موصل، العراق

المستخلص:

تُستخدم شبكات المكعّب الفائق على نطاق واسع في أنظمة الحوسبة المتوازية والموزعة لما تتمتع به من قابلية عالية للتوسّع وقدرة متأصلة على تحمّل الأعطال. إلا أنّ تعرّض العقد أو الوصلات للفشل يؤدي إلى تدهور ملحوظ في الأداء، مما ينعكس سلبًا على كفاءة حركة البيانات وتوفّر الشبكة. يقدّم هذا البحث إطارًا جديدًا قائمًا على التعلّم العميق للاسترداد الذكي من الأعطال في شبكات المكعّب الفائق، بالاعتماد على الشبكات العصبية الالتفافية الحسّاسة للأبعاد تم تصميم النموذج المقترح للعمل عبر أبعاد مختلفة للمكعّب الفائق (2D، 4D، 6D)، وتم تدريبه باستخدام سيناريوهات أعطال مولّدة اصطناعيًا تشمل أعطال العقد والوصلات. أظهرت النتائج التجريبية تحسّنًا ملحوظًا في الأداء مقارنة بالأساليب التقليدية. ففي المكعّب ثنائي الأبعاد، انخفض زمن الاسترداد إلى 0.1905 ثانية، وارتفع معدل الإنتاجية إلى 0.5، مع تقليل زمن الاستجابة إلى 1.9608 ثانية مقارنةً بـ 3.8462 ثانية في النهج التقليدي. أمّا في المكعّب رباعي الأبعاد، فقد تحسّن زمن الاسترداد من 0.2319 ثانية إلى 0.1825 ثانية، وازداد معدل الإنتاجية من 0.1484 إلى 0.25، بينما انخفض زمن الاستجابة من 6.3116 ثانية إلى 3.8462 ثانية. وفي التكوين سداسي الأبعاد، انخفض عدد الدورات في الثانية من 0.2014 إلى 0.1709، وارتفع معدل الإنتاجية من 0.0405 إلى 0.0938، كما انخفض زمن الاستجابة من 19.7913 ثانية إلى 9.6386 ثانية. تؤكد هذه النتائج فعالية توظيف التعلّم العميق في تمكين الاتصال الشبكي التكيّفي، والمرن، والقادر على الاستشفاء الذاتي، بما يعزز موثوقية وأداء الأنظمة الموزعة واسعة النطاق في البيئات المعرضة للأعطال.

الكلمات المفتاحية: شبكات المكعّب الفائق، التعلّم العميق، تحمّل الأعطال، أعطال العقد، أعطال الوصلات، استرداد الأعطال.