Metaheuristic Optimization Review

♦ ISSN: 3066-280X Vol: 07, No. 02, 2026 ♦ PP. 51–72

♦ Article DOI: https://doi.org/10.54216/MOR.070204

REVIEW ARTICLE

Intelligent Network Management Systems: A Comprehensive Review of Artificial Intelligence, Machine Learning, and Metaheuristic Optimization Approaches

Amir Ibrahim Neamatallah[1,*] **Raneem Magdy Elagmy**[1] **Ahmed Mohamed Yousry Ibrahim**[1] **Mohamed Ahmed Ibrahim Ali**[1]

[1] Department of Communications and Electronics, Delta Higher Institute of Engineering and Technology, Mansoura 35111, Egypt

Emails: CH2000075@dhiet.edu.eg; CH2100343@dhiet.edu.eg; CH1800091@dhiet.edu.eg; CH2100156@dhiet.edu.eg

Received: June 12, 2026 Revised: August 06, 2026 Accepted: September 05, 2026 ⋆ Corresponding author

ABSTRACT

Modern communication networks have become increasingly heterogeneous, programmable, distributed, and service-driven, creating management requirements that exceed the capabilities of purely manual or static rule-based operation. This review examines the evolution of intelligent Network Management Systems (NMSs) through the integration of machine learning, deep learning, predictive analytics, feature selection, metaheuristic optimization, network telemetry, software-defined control, and automated decision mechanisms. The reviewed methodological directions support traffic and performance prediction, anomaly and intrusion detection, fault localization, configuration optimization, resource allocation, routing assistance, quality-of-service management, capacity planning, and closed-loop remediation. Intelligent approaches can strengthen situational awareness and reduce the time required to identify, diagnose, and respond to rapidly changing network conditions, while metaheuristic algorithms provide flexible mechanisms for feature-space reduction, model calibration, multi-objective resource optimization, and search within large configuration spaces. Practical deployment nevertheless remains constrained by telemetry quality, concept drift, computational overhead, interoperability, multi-vendor heterogeneity, security, privacy, explainability, scalability, and the risk of unsafe automated actions. Future research should therefore prioritize trustworthy, policy-aware, scalable, secure, and adaptive NMS architectures capable of combining continuous observability with verifiable closed-loop automation across enterprise, cloud, edge, Internet of Things, and multi-domain network environments.

Keywords: Network Management System ▪ Artificial intelligence ▪ Machine learning ▪ Network automation ▪ Network telemetry ▪ Software-defined networking ▪ Metaheuristic optimization

1. INTRODUCTION

The continuing evolution of communication infrastructure has substantially increased the complexity associated with monitoring, configuring, protecting, optimizing, and maintaining modern networks. Earlier network environments were commonly composed of comparatively stable device populations, fixed topologies, limited service diversity, and management procedures dominated by command-line configuration and manually defined thresholds. Contemporary environments differ markedly from this operational model. Enterprise networks, cloud infrastructures, edge platforms, Internet of Things deployments, wireless systems, virtualized services, and software-defined networks can contain large numbers of heterogeneous devices and logical functions whose states change continuously. Traffic patterns fluctuate according to

application demand, users move across access domains, virtual functions can be instantiated dynamically, and service chains may span infrastructure controlled by different administrative entities. These characteristics create a management problem in which the network must be observed and adjusted continuously rather than configured once and subsequently maintained through occasional intervention.

A Network Management System (NMS) provides the operational layer through which administrators and automated controllers obtain visibility into network state and coordinate management actions. Its functions can include topology and inventory discovery, performance monitoring, fault and alarm management, configuration control, security supervision, event correlation, traffic analysis, capacity assessment, policy enforcement, and reporting. The increasing scale of managed infrastructure, however, makes it difficult for human operators to interpret every metric, alarm, flow record, and configuration change directly. Modern NMS architectures are therefore progressing from descriptive monitoring platforms toward decision-support and automation systems capable of converting telemetry into operational knowledge and subsequently translating that knowledge into controlled actions.

Figure 1 presents the conceptual organization adopted throughout this review. The architecture treats network management as a layered process that begins with heterogeneous network infrastructure and ends with decisions or actions that influence the managed environment. The figure is intentionally broader than a single protocol or product because intelligent management depends on the coordination of several functions: data collection, state representation, analytics, optimization, policy evaluation, visualization, storage, and controlled actuation. This perspective is important because a highly accurate analytical model provides limited operational value when it is disconnected from reliable telemetry or when its recommended actions cannot be translated safely into network configuration.

The growth of connected devices further increases the scale of the resource-allocation and coordination problem faced by NMS platforms. Large device populations differ in processing capability, traffic demand, interface capacity, communication technology, service priority, and battery constraints, and the management layer may need to allocate communication or computational resources under dynamically changing constraints. Research concerned with nature-inspired optimization for resource allocation in connected-device environments demonstrates the suitability of metaheuristic search when the decision space is large, heterogeneous, and difficult to solve exhaustively [1]. Within NMS design, this principle is relevant to bandwidth assignment, path selection, controller placement, service placement, channel allocation, and other management tasks in which multiple feasible configurations must be evaluated under competing operational objectives.

A second requirement is the extraction of useful operational information from high-volume telemetry. Interface counters, packet and flow statistics, queue states, device logs, alarms, routing events, configuration changes, latency measurements, and application-level indicators can exhibit nonlinear and time-varying relationships. Static thresholds remain useful for known conditions, but they can become inadequate when the meaning of a metric depends on topology, service context, temporal behavior, or interactions among several variables. Hybrid computational approaches that combine machine learning with metaheuristic optimization have demonstrated the ability to address detection and prediction problems involving dynamically changing data [2]. For NMS applications, the methodological significance lies in coupling adaptive learning with optimization so that the management platform can identify patterns that are difficult to represent through fixed rules alone.

The dimensionality of network telemetry introduces an additional analytical difficulty. A large NMS may collect thousands of metrics from interfaces, devices, virtual functions, hosts, and applications. Not all of these observations are equally informative for a particular management objective. Some variables may be redundant, strongly correlated, stale, or unrelated to the event being analyzed. Processing every available metric can increase model-training time, storage requirements, communication overhead, and inference latency without guaranteeing improved performance. Multi-objective feature-selection research demonstrates that dimensionality reduction can be formulated as a trade-off between retaining predictive information and minimizing the number of selected variables [3]. This formulation is directly applicable to network management when the analytical layer must maintain strong diagnostic or predictive capability while operating under practical constraints on computation and telemetry transport.

The effectiveness of feature selection should nevertheless be interpreted in relation to the target management function rather than through dimensional reduction alone. The subset that is useful for traffic forecasting may differ from the subset required for fault diagnosis, anomaly detection, quality-of-service prediction, or configuration-risk assessment. Methodological work emphasizing project-specific evaluation of feature-selection and analytical procedures highlights the need to evaluate preprocessing choices according to the actual objective and constraints of the downstream task [4]. This principle is particularly important in NMS research because one network may prioritize low-latency inference, another may emphasize interpretability, and a third may require robustness across several device vendors or service domains.

The search for informative telemetry subsets becomes more challenging as the number of candidate variables increases. In binary feature selection, every additional variable enlarges the number of possible subsets exponentially. Exhaustive search consequently becomes unrealistic for large monitoring datasets. Population-based metaheuristic strategies provide an alternative mechanism for navigating this combinatorial space, and investigations of Grey Wolf Optimizer variants for high-dimensional feature selection illustrate the broader ability of stochastic search to identify compact subsets without enumerating every combination [5]. Within an NMS, such search mechanisms can be used to identify the counters, events, or derived indicators that contribute most strongly to a specific prediction or classification objective.

Predictive modeling represents another important component of intelligent network management because many operational actions become more effective when future conditions can be estimated before degradation is visible to end users. Traffic

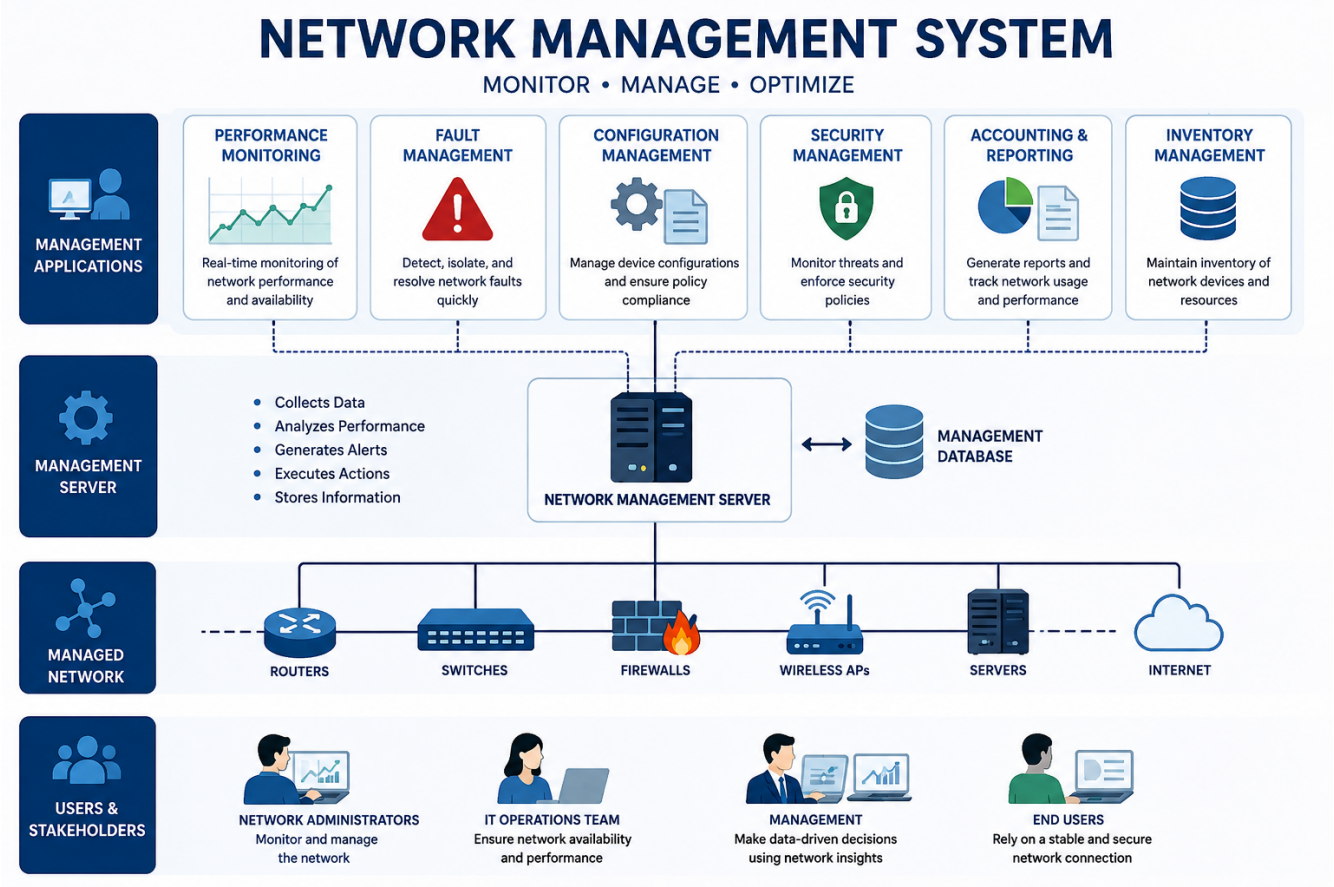

Figure 1. Conceptual architecture of a Network Management System integrating network infrastructure.

demand, interface utilization, queue growth, latency variation, capacity exhaustion, service quality, and failure likelihood can all be treated as forecasting targets. Optimization-assisted predictive modeling has been investigated in other nonlinear time-series environments, demonstrating how metaheuristic search can contribute to the configuration of learning systems used for complex prediction tasks [6]. Transferred to NMS design, the same methodological principle allows the optimizer to refine model parameters or input configurations before the predictor is embedded within a capacity-planning, routing, or proactive remediation workflow.

Structured preprocessing can further reduce unnecessary search effort when candidate telemetry variables exhibit different levels of association with the target quantity. Correlation analysis, dependency measures, or domain-informed filtering can be used to identify groups of variables that deserve greater attention before more computationally expensive optimization is performed. Research combining Pearson correlation and chi-square screening illustrates how statistical feature selection can reduce the computational requirements of intrusion detection [7]. In NMS applications, this approach can be adapted to relationships among throughput, latency, loss, queue occupancy, routing state, retransmission behavior, and application demand, thereby providing a more structured input space for subsequent learning.

Generalization is equally important because network conditions can change after a model has been trained. New applications may alter traffic distributions, devices can be upgraded, routing policies may change, virtual functions can be relocated, and previously unseen failure modes can appear. A model that performs strongly on one operating period may therefore deteriorate after deployment. Research combining metaheuristic feature selection with machine learning for scalable diagnostic modeling has explicitly considered performance across diverse operating conditions [8]. The corresponding lesson for NMS design is that model quality should be evaluated not only through average test accuracy but also through robustness to topology changes, workload variation, device heterogeneity, and temporal drift.

Security forms another inseparable component of intelligent network management. The NMS itself is a high-value operational asset because it can possess broad visibility and potentially privileged control over the managed infrastructure. Compromised telemetry can mislead analytical models, unauthorized configuration access can alter network behavior, and malicious events can be hidden among large volumes of legitimate operational data. Optimization-based feature selection and machine-learning intrusion detection have been examined in wireless mesh and smart-grid communication environments [9], illustrating how intelligent analytical mechanisms can contribute to the identification of malicious or abnormal behavior within networked infrastructure. For an NMS, security analytics should therefore be integrated with performance and configuration management rather than treated as an unrelated monitoring function.

This requirement becomes even more significant in industrial and Internet of Things environments, where the boundary between communication infrastructure and physical operation can be narrow. A network incident may interrupt monitoring, control, or safety-related communication, while an incorrect

automated remediation can affect a large number of connected devices. Hybrid artificial-intelligence frameworks developed for intrusion detection in industrial connected-device environments demonstrate the broader role of intelligent detection in protecting highly interconnected infrastructures [10]. Such evidence supports an NMS architecture in which anomaly detection, access control, policy validation, and operational analytics are coordinated rather than developed as independent subsystems.

Taken together, these developments show that an intelligent NMS is not defined by a single machine-learning model or optimization algorithm. It is better understood as a layered operational architecture. Collection mechanisms acquire telemetry and events from the managed network; normalization and preprocessing convert heterogeneous observations into consistent representations; feature-selection procedures identify informative variables; predictive and diagnostic models estimate present or future conditions; optimization procedures search configuration or resource spaces; policy mechanisms determine whether proposed actions are permitted; and actuation interfaces apply validated changes to the infrastructure. The quality of network management consequently depends on the interaction among these layers rather than on the performance of any one component in isolation.

This interaction also explains why improvements at one stage cannot automatically compensate for weaknesses elsewhere. A sophisticated anomaly detector cannot provide reliable results when its input telemetry is incomplete or delayed. A high-quality traffic predictor cannot improve service performance if the NMS lacks a safe mechanism for translating predictions into routing or capacity decisions. Similarly, an optimizer can return mathematically favorable configurations that are unsuitable when topology constraints, maintenance policies, security rules, or service-level commitments are missing from the objective formulation. Intelligent network management should therefore be evaluated as an end-to-end decision process.

The need for this integrated perspective becomes more apparent across the diversity of modern network environments. Enterprise networks require visibility across users, applications, wired and wireless access infrastructure, and security controls. Data-center networks must coordinate high-capacity fabrics, virtual workloads, and rapidly changing service dependencies. Telecommunications environments include access, transport, core, edge, and service-management domains, often supplied by multiple vendors. Internet of Things deployments introduce large numbers of constrained devices, while industrial networks can impose strict requirements on availability and latency. Although the physical technologies differ, each environment requires the NMS to transform heterogeneous observations into operationally meaningful decisions.

Artificial intelligence contributes to this process through several distinct functions rather than through one universal model. Classification can distinguish normal and abnormal operating states; regression and time-series models can predict traffic or performance; clustering can reveal previously unknown behavioral patterns; feature-selection mechanisms can reduce telemetry dimensionality; and optimization can search model, routing, placement, or configuration spaces. The value of combining these functions lies in enabling management systems to progress from reactive observation toward anticipatory and adaptive operation.

Metaheuristic optimization is particularly relevant because its role can appear at several different levels of the NMS. Before model training, it can identify informative telemetry features. During model development, it can calibrate hyperparameters or architecture choices. During operation, it can search for routing, placement, scheduling, or resource-allocation decisions. These roles should remain conceptually distinct because the candidate solutions, objective functions, and computational budgets differ. Recognizing the location of optimization within the management pipeline is essential for meaningful comparison between studies and for avoiding the misleading assumption that all optimizer-assisted NMS approaches solve the same problem.

The motivation for the present review therefore arises from the need to examine network management as an integrated computational field in which telemetry, machine learning, feature selection, metaheuristic optimization, security, policy, and automation perform complementary roles. Rather than presenting an inventory of algorithms, the review focuses on how computational components are positioned within the management lifecycle and how decisions made at one stage influence the reliability, scalability, and practicality of subsequent stages.

The scope of the review encompasses intelligent computational approaches relevant to network monitoring, state estimation, traffic and performance prediction, anomaly and intrusion detection, fault diagnosis, information selection, model calibration, resource optimization, automated configuration, and secure closed-loop operation. Particular attention is given to the transition from static and predominantly manual management toward systems capable of extracting knowledge from continuous telemetry and using that knowledge within controlled decision processes.

The principal contribution of this review is to establish a coherent conceptual foundation for evaluating intelligent NMS technologies according to the specific function performed by each computational component. By distinguishing data acquisition, state representation, predictive learning, feature-space optimization, model calibration, operational decision-making, policy verification, and actuation, the review provides a structured basis for comparing methodological approaches without conflating fundamentally different tasks.

The remainder of the paper is organized into three principal sections. The Literature Review synthesizes representative methodological developments and examines how optimization, learning, and hybrid computational architectures can support network-management functions. The Discussion interprets the broader implications of intelligent and automated NMS design, with emphasis on reliability, uncertainty, safe closed-loop operation, scalability, and lifecycle management. Finally, the Conclusion consolidates the main observations and identifies the requirements for dependable, adaptive, and increasingly autonomous network-management systems.

2. LITERATURE REVIEW

The literature relevant to intelligent network management increasingly reflects a methodological movement away from

isolated predictive models toward computational architectures in which data reduction, learning, optimization, and decision formation are deliberately coupled. This development is significant because the effectiveness of an intelligent system is determined not only by the individual algorithm used at a particular stage but also by the manner in which computational components interact throughout the analytical pipeline. Recent studies in closely related data-intensive and interconnected environments demonstrate several distinct patterns of integration, including optimizer-assisted data reduction, optimization-guided classifier construction, dynamically coordinated metaheuristic search, hybrid population-based optimization, and ensemble-driven deep-learning architectures. Although many of these studies originate from cybersecurity, distributed computing, industrial connected systems, and cloud environments rather than direct network-management applications, they provide useful methodological evidence regarding how intelligent computational components can be organized within complex operational systems. Their relevance to intelligent network management therefore lies primarily in the architectural and optimization principles they establish rather than in direct application-specific conclusions about traffic behavior or operational resource allocation.

This architectural arrangement introduces an important distinction in the interpretation of optimization-assisted intelligent systems. The optimizer does not necessarily determine the final operational decision directly. Instead, it can alter the information space from which the subsequent learning mechanism constructs its decision boundary. This position gives the optimization stage an indirect but potentially substantial influence over the final analytical outcome. Such a configuration requires careful coordination between the criterion used to evaluate candidate variable subsets and the behavior of the downstream analytical model. A subset that appears compact from a dimensional perspective may not be useful if it removes variables required to discriminate relevant operating states, whereas a comparatively larger subset may preserve unnecessary information and increase computational requirements. Consequently, the quality of an optimization-assisted selection mechanism should be interpreted according to its interaction with the subsequent learner rather than according to the reduction ratio alone. This observation provides a useful basis for evaluating intelligent architectures in which multiple computational components influence one another sequentially.

A related branch of the literature explicitly combines machine-learning procedures with metaheuristic optimization for simultaneous feature-selection and classification objectives. Work on botnet attack detection, for example, has examined machine learning together with metaheuristic optimization algorithms as a coordinated framework in which the search procedure determines informative feature combinations and the learning mechanism evaluates the discriminatory usefulness of those combinations [11]. This formulation differs from conventional workflows in which feature reduction and classification are selected independently. The optimizer and learner instead become mutually dependent: the search process requires an evaluation mechanism capable of indicating whether a candidate subset is useful, while the learning model depends on the search process to define the representation on which its subsequent classification behavior is based. The methodology therefore creates a feedback relationship between data representation and model performance.

This interaction has important implications for the design of intelligent computational frameworks. When machine learning is embedded inside a metaheuristic search process, the objective function becomes one of the most consequential methodological decisions. An optimization procedure can only search for solutions according to the information supplied by its fitness formulation. If the objective considers only predictive performance, the resulting search may favor comparatively large candidate subsets whenever they provide even a marginal improvement in classification quality. Conversely, an objective that emphasizes dimensional reduction excessively may produce highly compact representations that omit relevant information. Multi-criteria formulations can address this tension, but they introduce additional decisions concerning weighting, normalization, dominance relations, or trade-off selection. Accordingly, the effectiveness of a hybrid machine-learning–metaheuristic framework depends as much on the formulation of the search objective as on the identity of the optimizer itself. This methodological issue is particularly important when different optimization algorithms are compared because an apparent difference in algorithmic performance may actually result from differences in fitness construction, stopping criteria, or learner configuration.

The literature has also progressed toward hybrid search mechanisms that modify the optimization process itself rather than simply combining one optimizer with one predictive model. A feature-selection framework for network intrusion detection employed a hybrid metaheuristic dynamic optimization strategy, illustrating a design in which feature-space exploration is performed through a dynamically coordinated search mechanism rather than through a single fixed search procedure [12]. The significance of this approach lies in the treatment of the search process as a configurable computational system. Instead of assuming that one search behavior is equally effective throughout the entire optimization trajectory, dynamic hybridization provides a basis for changing the relative contribution of different search behaviors as candidate solutions evolve.

This dynamic-search perspective raises a separate methodological issue concerning how exploration and refinement should be distributed across the optimization process. Search mechanisms that produce strong diversification during the early iterations may be beneficial when the candidate space contains many distant promising regions, whereas stronger local refinement may become more useful once high-quality regions have been identified. A dynamically coordinated metaheuristic framework can therefore be interpreted as an attempt to control the search trajectory rather than merely accelerate convergence. This is distinct from simple parameter adjustment because the underlying search behavior itself may be altered according to the state of the optimization process. For intelligent computational applications, such mechanisms create an additional design dimension: optimization performance depends not only on the selected algorithm but also on when and how different search operators are activated.

This observation is particularly relevant to high-dimensional analytical problems because the topology of a search space

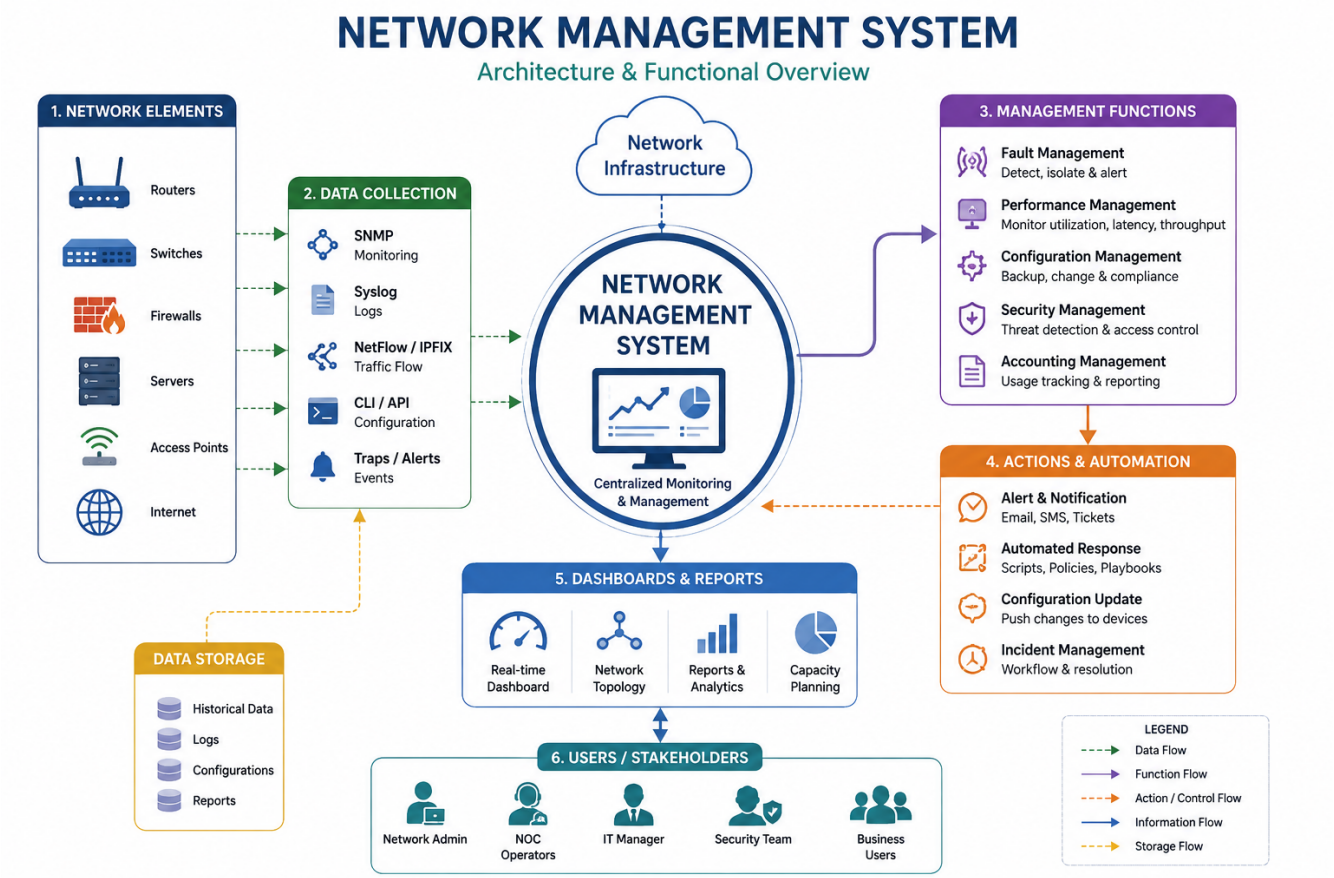

Figure 2. Conceptual architecture of a Network Management System integrating network elements, data collection, centralized management functions, automated actions, dashboards, data storage, and stakeholder interaction.

can change substantially as variables are added or removed. In a binary feature-selection problem, every additional candidate feature doubles the number of possible subsets, causing the theoretical solution space to expand exponentially. An optimizer operating in such an environment must therefore balance sufficient diversification to avoid becoming confined to a limited region with adequate refinement to distinguish among similar high-quality subsets. Dynamic hybridization provides one mechanism for managing this balance. However, it also increases implementation complexity because the framework must specify how search components interact, how transitions are controlled, and whether additional mechanisms improve the final solution sufficiently to justify their computational overhead.

A further line of research has investigated explicit hybridization between established population-based optimization algorithms. A feature-selection framework combines binary Grey Wolf Optimization (GWO) with Cuckoo Search (CS), using adaptive search control and probabilistic variation to coordinate local refinement and wider exploration [13]. This differs from benchmarking GWO and CS as independent optimizers because the hybrid deliberately assigns complementary roles to both search mechanisms within one feature-selection process. The study evaluates classification-oriented benchmark datasets rather than network-attack deployment, so its relevance to intelligent NMS design is methodological: it illustrates how a composite search mechanism can be constructed and assessed before transfer to a specific management task.

Hybrid optimizer design creates several analytical questions that should be distinguished from conventional algorithm comparison. The first concerns the level at which the constituent methods interact. A sequential hybrid may allow one optimizer to produce an initial solution population that is subsequently refined by another, whereas an embedded hybrid may exchange candidate information continuously during the same optimization process. A parallel architecture can maintain multiple search processes simultaneously and subsequently combine their results. A hierarchical architecture may assign different optimization responsibilities to different layers. These configurations are not methodologically equivalent, even when they contain the same constituent optimizers. Consequently, describing a framework only according to the names of the included algorithms can conceal the mechanism through which the hybrid system actually searches the solution space.

A second issue concerns the computational cost introduced through hybridization. Combining multiple optimizers can increase the number of objective-function evaluations, population-management operations, model-training cycles, or information-exchange procedures required before termination. Therefore, superiority in a final predictive metric cannot automatically establish that a hybrid framework is operationally preferable. The computational budget required to obtain that performance must also be considered. This concern becomes particularly important for intelligent systems intended to support frequent model updating or near-real-time analytical decisions, because optimization latency can become part of the practical suitability of the frame-

work. Accordingly, comparative evaluation of hybrid optimizers should distinguish improvements produced by genuinely more effective search behavior from improvements obtained primarily through a substantially larger computational budget.

A more structurally elaborate design appears in research combining ensemble-based feature selection with optimization-driven deep learning for attack detection in cloud-computing environments [14]. This type of architecture introduces multiple analytical stages before the final decision is produced. Ensemble-based selection attempts to determine informative inputs using the combined behavior of several selection mechanisms, while optimization is subsequently associated with the configuration of the deep-learning component. The architecture therefore distributes computational intelligence across different levels of the pipeline rather than expecting one algorithm to perform all analytical responsibilities. Such a design represents a progression from single-stage hybridization toward modular intelligent systems in which separate mechanisms are assigned to information screening, model construction, and final analytical inference.

The ensemble component introduces a methodological principle that differs from the population-search concepts discussed previously. Whereas a metaheuristic searches candidate solutions through iterative population evolution, an ensemble-based selection mechanism can exploit agreement, complementarity, or aggregation among multiple selectors. The purpose is not necessarily to locate one candidate solution through a single search trajectory but to construct a more stable representation from the outputs of several analytical mechanisms. This distinction becomes important when evaluating multi-stage frameworks because an ensemble and a metaheuristic should not be interpreted as interchangeable computational tools. They solve different methodological problems. The ensemble addresses diversity among analytical judgments, while the optimizer addresses the search for an appropriate configuration according to an objective function.

The coexistence of these mechanisms within one framework also introduces the issue of error propagation across computational stages. In a multi-stage intelligent architecture, an early analytical decision influences every subsequent component. If the input-selection stage removes useful information, later optimization cannot recreate variables that are no longer available. Similarly, a high-quality feature representation does not guarantee strong final performance if the optimized learning architecture is poorly matched to the structure of the target problem. Evaluation of multi-stage systems should therefore examine the contribution of each stage separately rather than reporting only the final output of the complete pipeline. Ablation analysis is particularly useful in this context because it can determine whether each additional module produces a measurable contribution when the remaining architecture is held constant. Without such analysis, increasingly complex frameworks can appear effective while leaving uncertainty regarding which component is actually responsible for the observed improvement.

Taken collectively, the reviewed methodological evidence reveals a progression in the structural complexity of optimization-assisted intelligent systems. The earliest architectural form considered here places a metaheuristic directly before a classifier as an attribute-selection mechanism. A second pattern couples the learning algorithm with the optimization procedure so that candidate representations are evaluated according to downstream predictive behavior. Dynamic hybridization then modifies the internal search trajectory itself, while explicit multi-optimizer frameworks combine distinct population-based search mechanisms. Finally, ensemble-driven architectures distribute analytical responsibilities across several computational layers. These configurations form a useful taxonomy because they demonstrate that the phrase “metaheuristic-assisted intelligence” can describe substantially different system architectures. Meaningful comparison therefore requires identification of the exact role played by optimization rather than merely recording whether an optimizer was included.

This architectural distinction is particularly useful when transferring methodological lessons into intelligent network-management research. The key design question is not simply which optimizer achieves the highest performance on a particular benchmark. A more informative question concerns where optimization is positioned within the complete management architecture and which computational object it modifies. Optimization may influence the representation of measured information, the internal configuration of a learning model, the structure of a hybrid analytical pipeline, or a higher-level decision mechanism. These placements generate different objective functions, different decision spaces, and different computational costs. Consequently, comparisons across studies should first determine whether the competing algorithms actually solve the same optimization problem. Comparing a method used for input selection with another used for model configuration, for example, would provide little methodological insight unless the different optimization roles were explicitly recognized.

Another important theme emerging from the literature concerns evaluation granularity. Intelligent computational frameworks often report one final performance indicator even though several stages contribute to the result. A more rigorous evaluation strategy should assign metrics to the stage they are intended to measure. A feature-selection component can be assessed according to dimensional reduction, subset stability, and its influence on downstream performance. An optimizer can be evaluated according to convergence behavior, computational budget, robustness across independent runs, and sensitivity to initialization. A predictive learner can be evaluated using task-specific predictive metrics. A complete operational framework may additionally require assessment of latency, scalability, memory requirements, or decision feasibility. Maintaining this separation prevents one strong final metric from concealing weaknesses elsewhere in the architecture.

The stochastic character of most metaheuristic approaches creates an additional requirement for experimental design. A result from a single optimization run provides limited information regarding algorithmic reliability because different random initializations can produce different candidate trajectories and final solutions. Comparative studies should therefore distinguish best-case performance from typical performance and should report the distribution of outcomes across repeated independent runs. Measures such as the mean, median, standard deviation, interquartile range, or confidence

intervals can provide information regarding the stability of the optimizer, while appropriate statistical tests can determine whether observed differences between competing methods are sufficiently consistent to support comparative claims. This is particularly important when performance differences are numerically small, since a marginal improvement in a single run may fall within ordinary stochastic variability.

Reproducibility should also be considered independently from predictive quality. Hybrid intelligent frameworks can contain numerous implementation decisions, including population size, iteration limits, termination conditions, initialization procedures, algorithmic parameters, preprocessing choices, model configurations, feature bounds, and objective-function definitions. Incomplete reporting of these settings makes it difficult to determine whether an observed improvement originates from the proposed methodological contribution or from differences in experimental configuration. Computational-budget reporting is equally important when several optimizers are compared because equal iteration counts do not necessarily correspond to equal computational effort if the algorithms perform different numbers of objective evaluations per iteration. A reproducible evaluation should therefore expose enough implementation detail to allow the complete search and learning process to be reconstructed.

The reviewed literature further suggests that increasing architectural complexity should not automatically be interpreted as methodological advancement. Adding more optimizers, selection stages, ensemble components, or learning layers can increase representational capacity, but every additional component also introduces parameters, computational overhead, and potential sources of instability. A simpler framework may therefore be preferable when it achieves comparable performance with lower computational demand and clearer analytical behavior. This principle is particularly important for practically deployed intelligent systems because real-world implementation often imposes resource constraints that are absent from offline experimental evaluation. Model complexity should consequently be justified by demonstrable functional contribution rather than by architectural novelty alone.

An additional consideration concerns the relationship between benchmark-specific success and cross-domain methodological transfer. The five studies examined in this section primarily evaluate cybersecurity, network, industrial connected-device, and cloud-computing problems. Their results should therefore not be interpreted as direct empirical evidence that identical performance would be achieved in intelligent network-management applications. Data distributions, objective functions, temporal dependencies, physical constraints, and operational consequences differ substantially between these domains. Nevertheless, the studies provide methodological evidence concerning the organization of hybrid intelligent systems, including how optimization can interact with learning, how multiple search mechanisms can be combined, how ensemble selection can precede deep models, and how architectural complexity can be distributed across several computational stages. Their value to an intelligent network-management review is thus methodological rather than application-specific.

This distinction is essential for maintaining rigorous interpretation of the literature. Transferability should be evaluated at the level of computational principle rather than at the level of reported numerical performance. A feature-selection mechanism demonstrated in network data may motivate investigation of an analogous search structure for another high-dimensional dataset, but its original accuracy cannot be transferred to a different task. Similarly, the successful combination of two optimizers within one classification framework demonstrates the feasibility of hybrid search but does not establish that the same hybridization will be optimal under a different objective landscape. Recognizing this boundary prevents cross-domain evidence from being overstated while still allowing useful computational principles to inform subsequent research.

The literature can therefore be understood according to an emerging hierarchy of methodological integration. At the lowest level, optimization is attached to one isolated analytical task. More integrated systems couple optimization directly with learning. More advanced configurations coordinate several search behaviors or multiple optimizers. Multi-stage architectures then distribute distinct responsibilities among selection, optimization, and deep-learning components. This hierarchy provides a useful analytical framework for reviewing intelligent systems because it enables methodological complexity to be described according to functional organization rather than according to the number of algorithms listed in a model name.

For intelligent network-management research, such an organization can support more meaningful evaluation of future computational architectures. The central issue is whether additional computational layers provide clearly identifiable value within the complete decision process. Each component should possess a defined function, its contribution should be independently measurable, and its computational requirements should be reported. This perspective also encourages researchers to avoid constructing hybrid models in which algorithms are combined primarily to create novelty without establishing a clear functional reason for each component.

Overall, the literature reviewed in this section demonstrates that contemporary intelligent computational research is progressively shifting toward deliberately structured hybrid architectures. The principal methodological development is not simply the growing use of metaheuristic algorithms but the increasing diversity of roles assigned to optimization within larger analytical systems. Feature-space modification, learner-dependent search, dynamic operator coordination, multi-optimizer hybridization, and ensemble-supported deep architectures represent distinct design strategies with different computational implications. Recognizing these distinctions provides a more precise foundation for assessing intelligent network-management frameworks and prevents fundamentally different optimization tasks from being grouped together under a single broad methodological category. The subsequent Discussion can therefore concentrate on the higher-level implications arising from this literature without reproducing the individual methodological patterns examined here.

To provide a structured methodological view without repeating the analytical arguments developed in the preceding literature synthesis, Table 1 organizes representative computational architectures according to the specific methodological function implemented in each study. The included works originate from several data-intensive application domains and are therefore not presented as direct empirical evidence for network-management performance. Instead, the table isolates transferable computational designs that can inform intelligent network-management research, including alternative strategies for feature-space construction, model calibration, multi-objective learning, hybrid optimization, comparative optimizer assessment, explainability, environmental modeling, and decision-oriented prediction. This organization distinguishes the functional contribution of each methodology and avoids treating fundamentally different computational tasks as equivalent merely because they employ machine learning or metaheuristic optimization.

Table 1. Representative computational methodologies relevant to intelligent network-management research.

Study focus	Data/application type	Methodological approach	Main methodological objective
Hybrid feature-selection framework [15]	Distributed computing network data	Hybrid machine-learning architecture incorporating feature selection before intrusion classification	Constructs a reduced and discriminative input representation before model training, illustrating how data-space refinement can be positioned as an integrated stage of an intelligent analytical pipeline.
Search-driven variable selection [16]	IoT network traffic observations	Statistical screening followed by SVM recursive feature elimination and PSO, with LightGBM and XGBoost classification	Combines inexpensive input screening with model-based subset refinement before classification, separating complementary feature-selection responsibilities.
Sensitivity-informed modeling [17]	Ransomware-related behavioral data	Feature selection combined with sensitivity analysis and an optimized hybrid predictive model	Evaluates variable influence alongside optimized model construction so that model development incorporates information regarding the relative contribution of candidate inputs rather than relying only on final predictive output.
Multi-objective network design [18]	Transit routes and service frequencies under station congestion	Bilevel optimization and a multi-objective evolutionary algorithm based on objective-space decomposition	Balances passenger and operator costs while representing congestion-sensitive passenger assignment, illustrating explicit treatment of competing network objectives.
Binary hybrid search [19]	UNSW-NB15 network traffic data	Hybrid Grey Wolf Optimization and Quantum Binary Bat feature selection with machine-learning classifiers	Combines two search behaviors to choose a compact feature subset for intrusion detection, linking discrete representation decisions with classifier evaluation.
Cross-model feature-selection assessment [20]	Landslide-conditioning geospatial variables	Ensemble recursive feature elimination combined with multiple classifiers and a meta-classifier	Examines feature selection across different learners and combines their predictions, distinguishing representation choices from the contribution of individual classifiers.
Swarm-assisted neural modeling [21]	Clinical diagnostic datasets	Swarm-intelligence metaheuristic optimization integrated with neural-network models	Synthesizes the role of swarm-based search in neural-model development and demonstrates how population-level optimization can be used to support configuration of nonlinear predictive architectures.
Multi-objective network fine-tuning [22]	Clinical diabetes data	Multi-objective metaheuristic-assisted fine-tuning of a deep network	Treats deep-model configuration as a multi-objective search problem, allowing competing optimization requirements to be considered simultaneously instead of reducing model selection to a single performance criterion.
Deep-model input optimization [23]	Cotton leaf image data	Metaheuristic feature-selection algorithms coupled with deep learning	Reduces the information supplied to the deep-learning stage through search-based feature selection, providing a framework in which input optimization precedes high-capacity representation learning.
Optimization-guided deep framework [24]	Software-defined network DDoS traffic	Binary Narwhal feature selection, attention-enhanced CNN–BiGRU classification, and Seagull hyperparameter optimization	Assigns separate roles to input selection, temporal and spatial feature modeling, and model tuning within a layered intrusion-detection framework.
Model-level metaheuristic calibration [25]	Parkinson's disease clinical variables	Metaheuristic optimization of machine-learning models	Uses optimization at the model-configuration level rather than exclusively at the input-selection level, demonstrating a distinct role for search algorithms in refining predictive-model settings.

Table 1. Representative computational methodologies relevant to intelligent network-management research (continued).

Study focus	Data/application type	Methodological approach	Main methodological objective
Multi-component forecasting architecture [26]	Cryptocurrency time-series data	Hybrid Transformer-based forecasting with attention and correlation mechanisms	Combines complementary temporal representations across forecasting scales, providing an example of modular predictive modeling rather than metaheuristic feature selection.
Explainable hybrid prediction [27]	Chronic kidney disease clinical data	Binary Bat feature selection with Extra Trees classification and explainable artificial intelligence	Combines input optimization, ensemble classification, and interpretability so that predictions can be examined through the contributions of clinical features.
Connected-device hybrid prediction [28]	Biomedical IoT health-data streams	Temporal adaptive neural evolutionary algorithm for predictive disease modeling	Combines temporal neural modeling with evolutionary adaptation in a connected sensing setting, linking data acquisition with optimized inference.
Metaheuristic-enhanced environmental modeling [29]	Suspended sediment, streamflow, and river water-level observations	Extreme Learning Machines tuned with PSO, Henry Gas Solubility, Electromagnetic Field, and Nuclear Reaction optimization	Uses alternative optimization strategies to calibrate an efficient nonlinear predictor for a continuously varying environmental quantity.
Hybrid credit-risk feature optimization [30]	Peer-to-peer lending records	Hybrid modified Grey Wolf Optimizer with an optimized decision-tree model	Coordinates feature-space search with a decision-oriented predictive model, illustrating how optimization can be distributed between representation construction and the subsequent learner.
Hybrid spatial localization [31]	Microseismic monitoring observations	Backpropagation neural network, Genetic Algorithm, and Gauss–Newton hybrid localization	Coordinates learning, global search, and local numerical refinement to estimate a physical source location rather than only selecting features.
Hybrid susceptibility mapping [32]	Flood-related geospatial, climate, and land-use variables	PSO-optimized XGBoost, Random Forest, and Support Vector Machine models	Tunes alternative predictive learners for spatial flood-susceptibility estimation while incorporating climate and land-use scenarios.

Table 1 demonstrates that the methodological landscape represented by the reviewed studies cannot be characterized through a single dominant computational architecture. Instead, the selected references illustrate a gradual movement from relatively focused feature-selection procedures toward increasingly integrated systems in which data representation, model configuration, optimization, prediction, and decision support are distributed across several computational stages. The significance of this progression lies in the changing role assigned to optimization. In some methodologies, optimization acts directly on the available input variables; in others, it modifies the internal configuration of the predictive model; more elaborate frameworks distribute optimization across several components or combine it with deep representation learning, ensemble analysis, or explainability. This methodological diversity is particularly relevant to intelligent network-management research because it shows that optimization should not automatically be regarded as a single interchangeable processing step. Its practical contribution depends on the computational object being optimized, the criterion used to measure candidate quality, and the position of the optimizer within the overall analytical pipeline.

The hybrid feature-selection framework represented in the first methodological category illustrates an important architecture in which information reduction occurs before the principal predictive or classification stage. This organization establishes a clear separation between determining which information should be retained and determining how the retained information should subsequently be interpreted. Such separation can be valuable in data-rich intelligent systems because raw datasets may contain variables with substantially different predictive contributions. Rather than forcing the predictive model to process the complete input space, a preliminary selection mechanism attempts to construct a more concentrated representation. Methodologically, this approach can reduce the effective dimensionality of the learning problem while allowing the subsequent learner to operate on a deliberately selected subset. The important principle is therefore not simply the removal of variables, but the architectural decision to place an information-screening mechanism before the main learner. For intelligent network-management applications involving measurements from multiple meters, sensors, operational devices, environmental sources, and temporal records, such an architecture provides a general mechanism for separating data-space refinement from subsequent prediction or control.

The dual-layer statistical and PSO-assisted methodology represents a more explicit search-oriented interpretation of feature selection. In this formulation, variable selection is treated as an optimization problem rather than as a deterministic ranking process. Every candidate subset can be regarded as a possible solution within a very large combinatorial search space, and the optimizer is responsible for navigating that space toward configurations associated with favorable analytical performance. The methodological implication is substantial because the search procedure must determine not merely whether an individual variable appears useful in isolation, but whether a combination of variables collectively provides an effective representation. Variables that appear weak individually may become important when interacting with other inputs, whereas several individually informative variables may contain overlapping information. Search-driven feature selection therefore provides a mechanism for considering combinations rather than isolated relevance scores. For intelligent network-management datasets containing correlated traffic counters, flow statistics, topology attributes, temporal

indicators, protocol states, and device telemetry, the same methodological principle can support the identification of compact input configurations without requiring exhaustive examination of every possible subset.

The sensitivity-informed hybrid methodology introduces a different analytical dimension because it places explicit attention on the influence of variables in addition to their inclusion or exclusion. Sensitivity analysis can provide information regarding how changes in particular inputs affect model behavior or predicted outcomes, thereby extending the methodological objective beyond simple subset construction. When combined with feature selection and optimized predictive modeling, this creates a pipeline in which the data are not only reduced but also examined according to their relative influence on the analytical process. Such a design is especially valuable when the objective includes understanding why a model responds strongly to certain measurements. Within intelligent network management, this distinction could become important when variables represent operationally meaningful quantities such as traffic intensity, queue states, interface utilization, routing changes, alarm patterns, or service-level indicators. An optimization framework that identifies a strong predictive configuration provides one type of knowledge, whereas sensitivity analysis contributes another by revealing how individual inputs influence the resulting model response. The methodological value lies in maintaining these two analytical objectives as complementary rather than treating predictive performance as the only criterion of interest.

The multi-objective transit methodology represented in the table addresses network design and service-frequency setting under station congestion. It combines a bilevel model of operator and passenger costs with an evolutionary algorithm based on objective-space decomposition. Evaluation on a shared transit benchmark provides a setting for studying trade-offs between competing operational objectives. For intelligent network-management research, the transferable principle is that alternative solutions should be assessed under consistent constraints and comparable computational budgets. A favorable result in one application does not establish universal optimizer superiority, but the explicit modeling of conflicting objectives can inform the design of network resource-allocation and service-quality studies.

The hybrid Grey Wolf–Quantum Binary Bat methodology addresses the special characteristics of feature-selection problems in which the decision associated with each candidate variable is inherently discrete. A feature is generally either retained or excluded, meaning that the optimizer must operate within a binary representation even when its original search behavior was developed for continuous variables. This requires an explicit mechanism through which search movements are translated into binary selection decisions. Hybridizing two population-based strategies introduces an additional layer because the resulting method attempts to combine distinct exploration characteristics within the same binary search environment. The methodological interest lies in the construction of a search process suited to combinatorial decisions rather than the simple transfer of a continuous optimizer without adaptation. This consideration is directly relevant to intelligent network-management modeling when the objective is to determine whether specific sensors, measurements, lagged observations, contextual variables, or derived features should participate in the predictive framework. The underlying decision is categorical, and the optimizer must therefore represent that structure appropriately.

The cross-model feature-selection methodology provides an important means of examining the interaction between data representation and learner choice. A feature subset cannot always be considered independently of the predictive algorithm that subsequently processes it. Different learning models possess different capacities for handling nonlinear relationships, redundant inputs, interactions, and complex decision boundaries. Consequently, a subset that performs particularly well with one learner may not necessarily provide equivalent behavior with another. Comparing multiple machine-learning algorithms together with ensemble recursive feature elimination enables the evaluation to distinguish between improvements associated with better information representation and those associated with the intrinsic characteristics of the learner. This methodological separation is highly relevant to intelligent network management because comparative studies frequently report a winning model without establishing whether its superiority results from model architecture, preprocessing, feature selection, or optimization. Cross-model assessment provides a more rigorous framework for identifying where the improvement actually originates.

Swarm-assisted neural modeling illustrates a broader integration between population-based optimization and nonlinear learning systems. Neural models contain numerous configurable elements whose selection can substantially influence their behavior. Swarm-based optimization offers a mechanism for searching this configuration space without relying solely on manual trial-and-error procedures. The important methodological contribution is the conversion of neural-model design into an explicit search problem. Candidate configurations can be represented as solutions, evaluated according to a defined criterion, and iteratively refined. This process can be used to explore settings that would otherwise require repeated manual experimentation. For intelligent network-management modeling, where neural networks may be employed to represent nonlinear temporal or operational relationships, such optimization can provide a systematic mechanism for identifying model configurations suited to a particular dataset. The methodology therefore differs from feature selection because the optimization target is located inside the predictive architecture rather than exclusively within the input representation.

The multi-objective deep-network fine-tuning methodology extends this concept by rejecting the assumption that model configuration should necessarily be optimized according to one criterion only. Intelligent computational systems frequently involve competing objectives. A configuration associated with strong predictive performance may require greater computational complexity, while a simpler model may offer lower computational demand at the cost of a modest reduction in predictive quality. Multi-objective optimization provides a formal mechanism for representing such competing requirements rather than combining them implicitly or ignoring all but one. The resulting search process can identify a set of trade-off solutions rather than a single universally optimal configuration. This methodological perspective has particular

relevance for intelligent network management because practical deployments may require simultaneous consideration of accuracy, execution time, memory requirements, processing overhead, robustness, or other operational criteria. Multi-objective fine-tuning therefore provides a conceptual basis for designing predictive intelligence according to practical system requirements rather than according to one isolated metric.

The methodology involving metaheuristic feature selection before deep learning illustrates a sequential organization in which the information supplied to a high-capacity model is optimized before representation learning begins. Deep-learning models are often capable of automatically extracting complex representations from data, but this does not imply that every possible input variable should necessarily be supplied without prior screening. A preliminary optimization stage can restrict the information space, after which the deep architecture concentrates on learning higher-level relationships within the selected representation. The methodological significance lies in the explicit division between variable-level search and hierarchical representation learning. Rather than expecting the deep model to perform both functions implicitly, the architecture assigns them to separate computational components. In intelligent network-management applications, this type of separation may be particularly useful when structured measurements from many sources are supplied to a deep predictive architecture and only a subset provides meaningful information for the target task.

The binary-narwhal-optimized framework adds another perspective by linking feature-space optimization with a specialized deep-learning architecture. Its methodological structure demonstrates how a study can deliberately separate the optimization of the input representation from the nonlinear processing performed by the deep-learning component. The optimizer addresses the question of which variables should be retained, while the subsequent attention-enhanced CNN–BiGRU classifier addresses how the retained information should be transformed into the final prediction or classification output. This separation can facilitate more precise analysis of the computational responsibilities assigned to each stage. It also demonstrates why hybrid architectures should not be described merely by listing their constituent algorithms. The position and function of each component determine the actual methodology. For intelligent network-management research, this provides a useful design principle: when several computational methods are combined, each should have a clearly defined role rather than being introduced simply to increase architectural complexity.

The model-level metaheuristic calibration methodology moves the optimization target away from the input variables entirely. Instead of determining which information enters the model, the search process is used to improve the configuration of the machine-learning models themselves. This distinction is important because feature selection and model optimization operate on different decision spaces. The former searches combinations of input variables, whereas the latter searches parameter or configuration alternatives governing how the learner processes those inputs. Treating these tasks separately allows researchers to identify whether predictive improvements are being generated by improved representation or improved model configuration. In intelligent network-management systems, this distinction becomes particularly important when the same dataset is evaluated using several predictive models. A model should not necessarily be considered inferior simply because it was evaluated under an arbitrary default configuration while another received extensive optimization. Model-level metaheuristic calibration therefore contributes to fairer and more systematic predictive comparison.

The multi-component forecasting architecture included in the table combines attention and correlation mechanisms within a Transformer-based framework for cryptocurrency forecasting. Unlike a feature-selection optimizer, this architecture focuses on temporal representation across multiple forecasting scales. Its methodological relevance is the separation of complementary representation mechanisms rather than the presence of a particular metaheuristic. Such designs can motivate investigations of network traffic and performance forecasting, where short-term fluctuations and long-term dependencies may require different forms of modeling. Any transfer to network management would still need task-specific validation and component-level comparisons to establish whether each part of the architecture contributes useful information.

The explainable hybrid prediction methodology introduces interpretability as an explicit architectural requirement rather than treating the predictive output as the final endpoint. Feature optimization and ensemble prediction can generate powerful decision systems, but complex models can make it difficult to understand the basis on which particular outputs are produced. An explainable framework attempts to supplement predictive capability with mechanisms that provide insight into the factors influencing the model. This represents a methodological shift because model evaluation is no longer confined to determining whether predictions are correct; the analytical framework must also provide information that supports interpretation. Intelligent network management can particularly benefit from this principle when automated recommendations affect routing, configuration, resource allocation, fault remediation, or other operationally significant network decisions. Operators may require an explanation of the conditions that contributed to a recommendation before accepting or acting upon it. Explainability therefore becomes a separate functional component of the intelligent system rather than simply a desirable descriptive property.

The connected-device hybrid methodology demonstrates how sensing infrastructure can be integrated with optimization and supervised learning within one analytical environment. The method combines an Internet of Things-enabled data-acquisition context with a temporal adaptive neural evolutionary framework. Its methodological significance is that the analytical model is positioned within a connected sensing ecosystem rather than treated as an isolated offline experiment. Data acquisition, communication, optimization, and inference therefore belong to the same operational chain. For intelligent network management, this type of architecture closely reflects the conditions encountered in practical smart-network environments, where measurements originate from distributed devices and must travel through communication infrastructure before analytical decisions can be generated. The quality of the final model consequently depends not only

on the learning procedure but also on the availability and reliability of the sensing layer that supplies its inputs.

The metaheuristic-enhanced extreme learning machine methodology illustrates the application of optimization-assisted prediction to continuously varying environmental measurements. This methodology is useful for understanding how a relatively efficient learning architecture can be combined with a search mechanism when the target phenomenon exhibits nonlinear variation. The optimizer can contribute to identifying an improved configuration, while the learning model provides the mapping from observed variables to the continuous target quantity. Methodologically, this demonstrates that metaheuristic-assisted learning is not restricted to categorical classification problems. The same computational principle can be applied to regression and continuous estimation. This distinction is highly relevant to intelligent network management because many core variables, including latency, throughput, interface utilization, queue occupancy, packet-loss rate, and link availability, are continuous rather than categorical. Optimization-assisted regression therefore represents a different methodological pathway from intrusion classification or disease detection.

The hybrid modified Grey Wolf Optimizer methodology combined with an optimized decision tree provides another example of distributed optimization, but with a structurally interpretable learner. The feature-selection component controls the input representation, while the decision-tree stage produces the subsequent risk-oriented prediction. This type of architecture is methodologically interesting because it balances search-based dimensional reduction with a model whose decision structure can generally be inspected more readily than that of many deep architectures. The resulting framework demonstrates that hybrid optimization does not inherently require a highly complex downstream learner. Intelligent computational systems can instead combine advanced search with comparatively transparent predictive mechanisms when interpretability, implementation simplicity, or explicit decision structure is valuable. For intelligent network management, such a design can be relevant when operators require understandable decision rules in addition to predictive capability.

The hybrid microseismic localization methodology represents a distinctly different role for metaheuristic optimization because the optimizer is associated with a search over a physical or spatial solution domain rather than only with feature or parameter selection. A backpropagation neural network, a Genetic Algorithm, and Gauss–Newton refinement cooperate to estimate the microseismic source location. This architecture demonstrates how optimization can interact with a learned representation while still operating on a decision space that corresponds to the underlying physical problem. The methodological lesson for intelligent network management is that optimization need not remain confined to model construction. It can also support physical-state estimation, localization, or other operational search tasks after predictive information has already been generated. This expands the potential role of metaheuristics from preprocessing and training into inference-stage decision formation.

The final susceptibility-mapping methodology combines PSO with XGBoost, Random Forest, and Support Vector Machine models to construct spatially oriented predictive models. Its relevance lies in the integration of a structured predictive learner with optimization for a problem characterized by heterogeneous geographic information. Spatial prediction differs from conventional independent-sample classification because nearby observations may share environmental characteristics and because the resulting output must often be interpreted over a geographic surface. The use of hybrid models in this setting illustrates how metaheuristic optimization can support structured prediction beyond purely tabular or temporal datasets. Intelligent network-management systems increasingly incorporate geographically distributed nodes, links, data centers, radio sites, edge resources, and regional traffic patterns. Although the original application differs, the methodological principle demonstrates the adaptability of optimization-assisted learning to spatially structured decision environments.

When the methodologies in Table 1 are considered collectively, a clear functional classification emerges. Some methods optimize the information entering the predictive system; others optimize the model that interprets that information; another group combines multiple computational mechanisms to distribute responsibilities across a larger architecture; and still others apply optimization directly to a physical or decision-oriented search space. These categories should not be collapsed into one general notion of "metaheuristic optimization" because the mathematical representation of the candidate solution, the objective function, and the interpretation of the optimized output differ substantially. A binary feature-selection vector represents a fundamentally different decision object from a set of model parameters or a candidate physical location. Recognizing these distinctions is essential when transferring methodologies into intelligent network-management applications.

The reviewed methodologies also reveal an important progression in how computational intelligence is organized. Early or comparatively simple architectures can be represented as a sequence in which data are first processed and then passed to a learner. More integrated approaches create feedback between the optimizer and the learner, meaning that candidate solutions are evaluated according to model behavior. Multi-component systems extend this interaction by assigning separate responsibilities to several algorithms, while explainable and sensing-integrated frameworks incorporate additional requirements beyond prediction itself. This progression indicates that the concept of an intelligent system is becoming increasingly architectural. Performance is determined by the coordination of computational functions rather than by the identity of one isolated algorithm.

From the perspective of intelligent network management, the methodological evidence contained in the table suggests that future computational architectures can be designed according to the specific location of uncertainty or complexity within the management process. If the primary problem is excessive or redundant sensing information, optimization may be most useful at the feature-selection stage. If the input representation is sufficiently informative but model configuration is difficult, search can be directed toward hyperparameter or architectural calibration. When several objectives must be balanced, multi-objective optimization provides an alternative to a single aggregated fitness criterion. If decision sequences

involve distinct analytical responsibilities, modular hybrid frameworks may be more appropriate. When operator trust is important, an explainability layer can be integrated after prediction. If measurements originate from distributed infrastructure, sensing and communication must become explicit elements of the architecture rather than invisible assumptions.

This functional interpretation also prevents unnecessary algorithmic complexity. A common risk in hybrid computational research is the assumption that adding another optimizer or predictive component will inherently improve the system. The methodologies summarized in the table instead suggest that every additional component should be justified according to a unique analytical function. A feature-selection module should solve an information-dimensionality problem; a model optimizer should solve a configuration problem; an attention mechanism should address selective representation; an explainability component should address interpretive transparency; and a physical search mechanism should solve a decision-space problem. When two components perform essentially the same role without a clear reason for their coexistence, architectural complexity can increase without providing a correspondingly distinct methodological contribution.

Another insight derived from the combined methodologies concerns the importance of stage-specific evaluation. Because hybrid architectures distribute intelligence across several components, evaluating only the final prediction may obscure the contribution of intermediate stages. An optimizer used for feature selection should be evaluated according to the quality and compactness of the selected representation. A model-calibration procedure should be examined according to the improvement obtained over an equivalent non-optimized configuration. A hybrid search should demonstrate that the combination provides value beyond its individual components. An explainability mechanism should be assessed according to whether it produces meaningful and consistent interpretations. A sensing-integrated system should consider whether the complete acquisition-to-inference chain can operate under realistic conditions. This stage-specific perspective is essential for determining whether a complex architecture provides genuine functional improvement.

Computational efficiency forms another methodological consideration that becomes increasingly important as architectures become more elaborate. Metaheuristic algorithms can require repeated evaluation of candidate solutions, and when each candidate involves training or validating a learning model, the total computational burden can become substantial. A multi-stage architecture containing feature selection, model optimization, and deep learning can therefore incur a much larger computational cost than a conventional learner. In network-management environments requiring frequent updates or operational decisions within restricted time intervals, this cost can become an important practical constraint. Consequently, methodological assessment should consider not only whether optimization improves predictive behavior but also whether the additional search effort remains compatible with the intended deployment environment.

The methodologies also highlight the difference between offline model development and online intelligent operation. Feature selection and extensive hyperparameter optimization can often be performed offline because they are associated primarily with model construction. In contrast, physical search, adaptive decision-making, or some forms of operational optimization may need to execute repeatedly while the system is active. This distinction affects the acceptable computational budget. An algorithm that requires substantial processing may remain practical during occasional model retraining but become unsuitable when invoked for every real-time management decision. Intelligent network-management design should therefore position computationally expensive procedures at stages where their cost can be tolerated and reserve faster decision mechanisms for time-critical operation.

Finally, the methodological references summarized in Table 1 demonstrate that the broader contribution of metaheuristic optimization is its capacity to provide a flexible search layer between complex decision spaces and intelligent predictive systems. Its role changes according to the problem being addressed: it can identify variables, configure learning models, coordinate hybrid components, balance competing objectives, support physical-state estimation, or participate within connected sensing architectures. The methodological value of metaheuristics should therefore be evaluated according to this functional role rather than according to algorithmic novelty alone. For intelligent network management, such a perspective provides a more rigorous foundation for selecting computational methods because the optimizer can be chosen according to the structure of the network-management problem, the nature of its decision variables, the operational time scale, and the computational constraints of the deployment environment.

Accordingly, the methodology matrix should be interpreted as evidence of several distinct computational design patterns rather than as a ranking of algorithms. The selected references collectively illustrate how intelligent systems can be constructed through different combinations of information selection, predictive learning, model calibration, hybrid search, explainable decision support, connected sensing, and physical-space optimization. These patterns provide methodological building blocks that can be adapted to network-management problems when their functional relevance is clearly established. Importantly, the transfer of a methodology from another application domain should preserve the computational principle while allowing the objective function, constraints, input representation, and evaluation criteria to be reformulated according to the physical and operational characteristics of the target network system.

2.1 Real-Time Network State Awareness and Digital Representation

An increasingly important direction in intelligent network management concerns the construction of continuously updated representations of the current operating state of the managed infrastructure. Predictive analytics can estimate future traffic or performance, but an NMS must first possess an accurate representation of what is occurring at the present moment. This requirement has encouraged architectures in which telemetry collected from routers, switches, wireless controllers, virtual network functions, hosts, sensors, applications, and security platforms is continuously integrated into a unified operational state. Such a representation can include

topology, inventory, interface status, routing state, link utilization, queue occupancy, latency, packet loss, alarms, configuration versions, service dependencies, and other variables that influence subsequent management decisions.

The methodological importance of real-time state awareness arises from the difference between historical analysis and operational intelligence. Historical datasets are essential for model development and long-term trend discovery, whereas active management requires information that reflects the current condition of the network closely enough to support timely action. Delayed metrics, stale topology information, duplicated alarms, missing samples, asynchronous timestamps, and inconsistent device clocks can therefore reduce decision quality even when the analytical model itself is accurate. Intelligent NMS architectures must consequently include mechanisms for time alignment, state reconciliation, missing-data handling, event ordering, and freshness assessment.

A unified network-state representation can provide an intermediate layer between heterogeneous infrastructure and higher-level analytics. Rather than forcing each machine-learning or optimization component to interact directly with vendor-specific interfaces, a normalized state model can supply consistent information to multiple management functions. Traffic predictors, fault-localization models, anomaly detectors, capacity planners, and configuration optimizers can then operate on a shared representation of the network. This separation reduces direct dependence between individual devices and analytical algorithms and provides a more maintainable architecture for multi-vendor environments.

The value of a unified representation becomes especially significant in networks containing several technological and administrative domains. A service path can traverse wireless access, campus switching, IP routing, data-center fabrics, cloud gateways, virtualized functions, and remote application infrastructure. A problem observed at the application layer may therefore originate from congestion, packet loss, routing instability, resource exhaustion, or a configuration change elsewhere in the path. Network-state awareness should consequently capture relationships among components rather than treating device metrics as independent observations.

Topology is particularly important because network measurements acquire meaning through connectivity. High interface utilization on an isolated backup link may have little immediate significance, whereas a similar condition on a critical transit link can threaten multiple services. Likewise, a device alarm can have different operational priority depending on the number and criticality of services that depend on that device. An intelligent NMS therefore benefits from graph-oriented representations that connect nodes, interfaces, logical services, and dependencies so that analytics can interpret telemetry within structural context.

A continuously updated state model also provides a foundation for event correlation. Large infrastructures can generate many alarms from one underlying incident. A link failure, for example, may produce interface-down notifications, routing reconvergence events, unreachable-host alarms, application errors, and service-level violations. Treating each event independently can overwhelm operators and automated systems. A state-aware management layer can instead relate events according to temporal proximity, topology, shared dependencies, and causal hypotheses, thereby reducing alert duplication and improving fault localization.

Temporal resolution creates another methodological challenge. Different management decisions require different observation intervals. Fast congestion control or radio-resource adaptation may depend on sub-second or second-level telemetry, whereas capacity planning may use hourly or daily aggregates. Collecting every variable at the highest frequency can generate unnecessary transport and storage overhead. Intelligent monitoring should therefore adapt sampling and retention according to the sensitivity of the target management function, preserving high-frequency visibility where rapid change matters while aggregating slower variables when fine resolution offers little additional value.

Digital representations can be extended toward network digital twins, in which the operational model is capable not only of describing current state but also of evaluating hypothetical changes before they are applied to production infrastructure. Candidate routing modifications, access-control policies, traffic-engineering decisions, software upgrades, or configuration changes can be tested against a computational representation of topology and service dependencies. This creates a verification layer between decision generation and execution and can reduce the operational risk associated with autonomous management.

Such an arrangement is particularly valuable because configuration mistakes can propagate rapidly through programmable networks. A logically valid but poorly scoped policy can disconnect services, create routing loops, overload links, or violate security requirements. Testing proposed changes in a sufficiently representative digital environment provides an opportunity to detect conflicts before physical actuation. The usefulness of this approach nevertheless depends on synchronization: a digital twin that does not reflect current topology, software versions, or traffic conditions can provide misleading confidence.

State synchronization therefore becomes a central requirement. An intelligent NMS must specify how frequently its operational representation is refreshed, how inconsistencies among data sources are resolved, and how uncertainty is represented when observations disagree. Telemetry should not be treated automatically as ground truth because counters can reset, devices can report delayed information, collection systems can fail, and monitoring traffic itself can be lost. Confidence and freshness indicators can help downstream analytics distinguish well-observed states from uncertain ones.

Overall, real-time network-state awareness introduces a layer of intelligence that is conceptually distinct from prediction, optimization, or fault remediation. It establishes the operational context on which those mechanisms depend. Intelligent NMS design therefore benefits from coherent, synchronized, topology-aware, and uncertainty-aware state representations that connect raw telemetry with higher-level analytical and automation functions.

2.2 Intent-Aware and Policy-Constrained Network Management

Intelligent network management cannot be evaluated exclusively according to technical optimization metrics because operational decisions are constrained by business objectives,

service-level commitments, security requirements, maintenance policies, regulatory obligations, and administrator intent. A routing configuration that minimizes average latency may be unacceptable if it violates isolation requirements, sends traffic through an unapproved domain, or reduces resilience. Similarly, aggressive resource consolidation may improve utilization while creating insufficient redundancy for critical services. Intelligent management must therefore incorporate policy and intent directly into the decision process rather than treating them as external considerations applied after optimization.

Intent-aware management represents operational objectives at a higher level than device-specific configuration. Instead of specifying every command required to produce a desired outcome, an operator can express requirements such as maintaining service availability, limiting latency, preserving path diversity, isolating particular traffic classes, or meeting capacity thresholds. The management system is then responsible for translating these objectives into lower-level configuration while ensuring that the resulting actions remain consistent with network state and existing policy.

This translation introduces a nontrivial reasoning problem because one high-level intent can correspond to several technically feasible configurations. Candidate configurations may differ in cost, resilience, latency, bandwidth consumption, or implementation complexity. Optimization can therefore be used to search the feasible configuration space, but the objective function must be constrained by policy. A mathematically optimal solution that violates a mandatory security or compliance rule cannot be considered acceptable merely because it produces a favorable performance metric.

Policy conflict is another important issue. Large organizations can contain global policies, domain-specific rules, temporary maintenance restrictions, tenant-level requirements, and application-specific service objectives. These rules may interact or conflict. An intelligent NMS should therefore provide mechanisms for detecting incompatible requirements before automated changes are applied. Conflict resolution can depend on explicit priority, administrative scope, risk classification, or escalation to a human operator when automatic resolution would be ambiguous.

Human involvement remains relevant even in highly automated environments. Fully autonomous management can execute routine actions without intervention, whereas decision-support systems may only recommend changes. Between these extremes, semi-autonomous architectures can apply low-risk remediations automatically while requiring approval for actions that affect critical services or large portions of the network. The appropriate level of autonomy depends on operational maturity, confidence in the analytical model, reversibility of the proposed change, and the consequences of failure.

Operator knowledge can also provide contextual information that is difficult to infer from telemetry alone. Planned maintenance, business-critical events, temporary service migrations, unusual traffic campaigns, and organizational priorities may change the interpretation of current measurements. A human-aware NMS should therefore provide mechanisms through which operators can annotate context, constrain automation, override recommendations, or temporarily alter policy without requiring redesign of the underlying analytical model.

Override behavior itself can provide useful feedback. If operators repeatedly reject a particular class of automated recommendation, the pattern can indicate that the optimization objective or policy representation does not capture practical requirements adequately. Rather than treating every override as an isolated exception, an intelligent management platform can record and analyze intervention patterns to identify systematic disagreement between automated reasoning and operational practice.

Transparency is particularly important when machine learning influences operational decisions. A recommendation to reroute traffic, modify access control, change queue priorities, or isolate a device should ideally be accompanied by information explaining the evidence and constraints that produced the recommendation. Explanations can include the observed anomaly, affected services, predicted consequence of inaction, expected effect of the proposed change, and policy rules considered during decision formation. Such information supports operator trust and improves auditability.

Multi-tenant and shared infrastructures introduce fairness considerations. Resource-allocation decisions can improve performance for one tenant while degrading another. An optimizer focused only on aggregate utilization can therefore conceal unequal service impact. Policy-aware management should allow minimum guarantees, priority classes, fairness objectives, and contractual constraints to be represented explicitly so that optimization does not sacrifice protected workloads for global efficiency.

Security policy must also remain independent from purely performance-oriented optimization. Automated systems with broad configuration privileges can create substantial risk if analytical errors or compromised models are allowed to alter security controls without adequate validation. Role-based authorization, change scoping, rollback capability, and independent policy verification provide safeguards that limit the consequences of incorrect recommendations. These mechanisms are especially important when AI-generated actions are translated directly into device configuration.

Intent-aware and policy-constrained management consequently broadens the definition of NMS performance. A practically effective system must not only improve throughput, latency, utilization, or fault response; it must also preserve service objectives, comply with operational policy, produce auditable decisions, support appropriate human control, and restrict automation to actions that remain within authorized boundaries.

2.3 Multi-Domain and Cross-Layer Network Coordination

A further development in intelligent network management concerns the transition from isolated device or domain management toward coordinated operation across multiple technologies, layers, and administrative boundaries. Modern services commonly traverse several infrastructure domains, including access networks, transport networks, data-center fabrics, edge platforms, cloud infrastructure, security gateways, and application-delivery systems. Managing each domain independently can provide local visibility while failing to reveal the end-to-end relationships that determine user experience.

Multi-domain coordination changes the structure of the management problem because a service-level degradation may be caused by several interacting conditions. Increased application latency can originate from access congestion, inefficient routing, packet loss, virtual-function overload, cloud-resource contention, or an upstream service dependency. An end-to-end NMS must therefore correlate observations from different layers and determine whether local optimization in one domain improves or degrades the complete service path.

Cross-layer management is similarly important. Decisions at the physical, link, network, transport, and application-related monitoring layers can influence one another. Increasing traffic on one IP path may overload an underlying optical or wireless resource. A security policy can alter routing behavior, while application placement can change traffic distribution across the network. Treating each layer as independent can therefore produce locally reasonable actions that conflict at system level.

Software-defined networking and network virtualization strengthen the possibility of coordinated management because control functions can operate through programmable interfaces and logical abstractions. Nevertheless, programmability does not automatically guarantee interoperability. Different controllers, orchestrators, telemetry systems, and vendor implementations may expose different data models and operational semantics. Intelligent coordination therefore requires consistent representations of topology, capability, policy, and service state across management components.

The placement of computational functions introduces another coordination dimension. Analytics can execute centrally, at the network edge, within domain controllers, or through distributed management services. Centralized analysis can obtain broad visibility but may introduce latency and scalability constraints. Local analysis can respond quickly but may lack global context. Hierarchical architectures can divide responsibilities so that domain-level controllers handle fast local decisions while higher-level components coordinate policies and end-to-end objectives.

Inter-domain management can also involve organizational boundaries. A service may cross infrastructure owned by several providers, while only limited operational information can be exchanged across those boundaries. Full centralization is therefore often unrealistic. Federated management approaches can coordinate objectives and summarized state while preserving administrative autonomy and limiting disclosure of sensitive internal information.

Intent decomposition becomes important in this context. A high-level end-to-end objective must be translated into domain-specific requirements that remain collectively consistent. For example, an availability objective may require redundant paths in the transport domain, sufficient compute redundancy in the edge domain, and appropriate failover behavior in the application layer. The NMS must therefore determine how a global requirement is distributed among local controllers and how compliance is verified after implementation.

The optimization horizon can differ across management layers. Fast path selection or queue adjustment may operate over seconds, while capacity planning and topology redesign can operate over weeks or months. Coordinating decisions across these horizons requires mechanisms that prevent short-term optimization from undermining longer-term objectives. Hierarchical scheduling and multi-timescale control can provide a structure in which strategic planning establishes constraints or targets that guide faster operational decisions.

Service assurance provides a natural integration point for multi-domain coordination. Instead of managing devices solely according to individual health metrics, the NMS can model the services that depend on those devices and determine how local events propagate to end-to-end outcomes. This service-centric perspective allows management priority to reflect business impact rather than device count alone and can improve fault localization by connecting infrastructure state with observed service degradation.

Resilience likewise requires coordination across domains. Redundancy at one layer may be ineffective when several supposedly independent paths share the same lower-layer dependency. An intelligent NMS should therefore identify shared-risk relationships and avoid assuming that logical diversity automatically implies physical diversity. Cross-layer topology information can reveal hidden dependencies that influence failover planning and restoration.

Multi-domain management also increases the importance of standardized models and interfaces. Consistent data representation allows telemetry, configuration, topology, and policy information to move among management components without requiring every integration to be implemented through vendor-specific translation. Interoperability should therefore be treated as a prerequisite for intelligent coordination rather than as a secondary implementation detail.

Overall, multi-domain and cross-layer coordination represents a substantial expansion of the NMS problem. Instead of optimizing isolated devices or one administrative domain, the management system must reason about relationships among technologies, service layers, controllers, policies, and organizations. The resulting architecture requires distributed state awareness, hierarchical decision-making, interoperable representations, and optimization mechanisms capable of balancing local objectives with end-to-end service requirements.

3. DISCUSSION

The evidence synthesized throughout this review indicates that the development of intelligent Network Management Systems is progressing toward operational architectures whose effectiveness depends increasingly on the quality of coordination between telemetry, analytical intelligence, policy, and programmable control. The central research challenge is therefore no longer restricted to identifying models capable of producing favorable classification or prediction results. A more consequential issue concerns whether the analytical mechanisms incorporated within an NMS remain operationally meaningful and sufficiently reliable when their outputs are translated into changes that affect active network infrastructure.

This distinction is important because network-management decisions operate within technical constraints that are not always visible to a purely data-driven model. Routing relationships, protocol timers, interface capacities, topology

dependencies, security policies, device capabilities, maintenance states, and service-level requirements can determine whether a proposed action is feasible. A learning model may identify a statistically favorable configuration while overlooking an operational restriction that makes the change unsafe. Intelligent NMS research should therefore move toward architectures in which network knowledge and policy constraints are incorporated directly into analytical and decision processes.

Topology-aware reasoning provides one mechanism for introducing this operational structure. Network measurements cannot always be interpreted correctly when devices and interfaces are treated as independent data points. The significance of utilization, loss, or a failure event depends on connectivity and service dependency. Graph-oriented representations can capture these relationships and provide a basis for fault localization, routing analysis, dependency discovery, and risk assessment. This direction also creates opportunities for combining graph-based learning with conventional telemetry analytics while retaining explicit structural information about the managed system.

A second major issue concerns uncertainty. Intelligent NMS platforms frequently rely on predictions or classifications that cannot be assumed to be perfectly correct. Traffic forecasts, anomaly scores, failure probabilities, and service-impact estimates all contain uncertainty, yet many automated workflows use their outputs as deterministic inputs to subsequent optimization. This treatment can produce fragile decisions because a configuration may appear optimal only under one predicted condition. The operational value of intelligent management would increase if the decision layer considered the confidence associated with analytical outputs and the sensitivity of candidate actions to estimation error.

Uncertainty-aware management therefore requires a different decision philosophy from simple best-case optimization. A network action that provides slightly lower nominal performance but remains safe across several plausible future states may be preferable to a highly optimized configuration that fails following a small prediction error. Confidence intervals, calibrated probabilities, scenario analysis, or robust optimization can help distinguish situations in which automation can act aggressively from situations that require larger safety margins or human review.

Related to uncertainty is the problem of concept drift. Network behavior can change after model deployment because applications evolve, user populations change, services migrate, routing policies are modified, devices are upgraded, and new attack patterns appear. An NMS that assumes stationary data distributions can therefore deteriorate without producing an obvious software failure. Continuous performance monitoring should be applied not only to the network but also to the analytical models that manage it. Drift detection, periodic recalibration, controlled retraining, and model-version tracking are consequently important components of the operational lifecycle.

The verification of automated actions represents another major requirement. As intelligent NMS platforms move toward closed-loop operation, the consequence of analytical error becomes more significant. A wrong prediction used only for reporting may have limited impact, whereas the same prediction can cause service interruption if it triggers an unsafe configuration change. The architecture should therefore distinguish decision generation from decision authorization. An analytical component can propose an action, while an independent verification layer checks topology, policy, syntax, reachability, service constraints, and rollback conditions before the change is executed.

Pre-deployment validation can be extended through digital-twin or sandbox environments. Proposed changes can be evaluated against a model of current topology and configuration before deployment. This approach is especially valuable for routing-policy modifications, access-control changes, software upgrades, or large-scale configuration templates whose consequences can propagate widely. The principal research challenge lies in maintaining sufficient fidelity between the validation environment and the production network so that successful simulation provides meaningful evidence rather than false assurance.

Rollback mechanisms should also be treated as part of intelligent automation rather than as an afterthought. Even carefully validated changes can produce unexpected effects because models and test environments are incomplete. An NMS should therefore define how the success of an action will be measured, how long the observation period should continue, which signals indicate deterioration, and under what conditions the previous configuration will be restored automatically. Closed-loop management is dependable only when action, observation, verification, and recovery are designed as one lifecycle.

Cybersecurity introduces a parallel set of concerns because an intelligent NMS combines privileged access, high-value telemetry, and automated decision capability. Attackers who manipulate telemetry can influence analytical conclusions without necessarily compromising the machine-learning model directly. Poisoned training data can alter future decisions, while unauthorized access to the actuation layer can transform a management platform into an attack mechanism. Security must consequently protect the entire analytical chain, including data collection, model training, model storage, inference, policy evaluation, and configuration execution.

The integrity of telemetry is particularly important. Intelligent analytics can become highly confident in an incorrect conclusion when several correlated inputs originate from the same compromised source. Cross-validation among independent data sources, device identity verification, signed telemetry where appropriate, provenance tracking, and anomaly detection at the collection layer can reduce this risk. The NMS should be able to distinguish between a network anomaly and an anomaly in the monitoring infrastructure itself.

Explainability constitutes another requirement for trustworthy operation. Network operators often need to understand why an automated system has classified an event as anomalous, prioritized a particular root cause, or recommended a configuration change. Explanations should be operationally meaningful rather than restricted to abstract model coefficients. They can identify the telemetry variables that changed, the topology or service dependencies involved, the policy constraints considered, and the expected effect of the proposed action. Such explanations improve auditability and make it easier to determine whether the model's reasoning is consistent with

engineering knowledge.

However, explainability should not be interpreted as a substitute for validation. A plausible explanation can accompany an incorrect recommendation. The role of interpretability is to support human understanding and diagnosis, while independent policy and safety checks determine whether an action is permitted. Separating these functions prevents user trust from depending entirely on the persuasiveness of a model-generated explanation.

Computational efficiency becomes increasingly important as NMS architectures incorporate multiple analytical stages. Metaheuristic optimization can require repeated evaluation of candidate solutions, and deep-learning models can impose substantial training or inference cost. Such computation may be acceptable for offline capacity planning or periodic model calibration but unsuitable for fast fault remediation or sub-second control. The placement of computationally intensive algorithms should therefore reflect the time scale of the management function.

This consideration naturally leads to distributed computation. Cloud or centralized platforms provide substantial processing capacity and global visibility, while edge or domain-local analytics can reduce latency and dependence on continuous connectivity. A hierarchical NMS can allocate functions according to their characteristics: long-term model training and global optimization can execute centrally, whereas time-critical inference and local remediation can remain close to the managed infrastructure. The challenge lies in coordinating these levels without creating inconsistent decisions.

Scalability should also be evaluated in terms of telemetry volume and management overhead rather than device count alone. Increasing sampling frequency can improve observability while consuming bandwidth, storage, and processing resources. Intelligent monitoring can therefore become adaptive, increasing collection frequency for unstable or critical resources and reducing it for stable portions of the network. Such mechanisms allow observability itself to become an optimization problem.

Interoperability remains a practical barrier to the deployment of intelligent management. Multi-vendor environments often contain different telemetry formats, APIs, configuration models, and operational semantics. Machine-learning models trained on one representation may not transfer directly to another. Standardized data models and management protocols can reduce this fragmentation, but semantic consistency remains necessary even when the transport mechanism is standardized. Future NMS research should distinguish protocol interoperability from true operational interoperability.

Administrative boundaries create an additional challenge in multi-domain environments. A service provider, cloud operator, enterprise, and application provider may each control a different part of the end-to-end service path. No single entity may possess complete telemetry or configuration authority. Intelligent management must therefore operate under partial observability and constrained control. Federated analytics, abstracted state exchange, and intent negotiation are possible directions for coordination without requiring every domain to expose internal details.

The appropriate level of abstraction is another important design choice. Highly detailed models can represent individual flows, queues, interfaces, and configuration states but can become computationally difficult to scale. More aggregated models reduce complexity but may conceal the local conditions that determine whether an action is safe. The required representation therefore depends on the management objective. Strategic capacity planning can tolerate aggregation, whereas fault isolation or policy validation may require fine-grained topology and configuration detail.

Hierarchical decision architectures provide a promising compromise. Higher-level components can determine service objectives, risk tolerances, or resource targets, while lower-level controllers translate them into domain-specific actions. This structure allows different analytical methods to operate at different temporal and spatial scales. The principal challenge is maintaining consistency: high-level intents must remain achievable under local constraints, and local actions must contribute to the end-to-end objective rather than optimizing one domain at the expense of another.

Evaluation methodology should evolve accordingly. Predictive accuracy alone is insufficient for judging an intelligent NMS because the ultimate purpose of the analytical model is to improve network operation. A traffic predictor with marginally higher error can still produce better capacity decisions if its errors occur in regions that do not alter management actions. Conversely, a highly accurate model can be operationally poor if its rare errors occur during congestion, failure, or security-critical conditions.

Decision-oriented evaluation connects analytical metrics with network outcomes. Instead of examining only classification accuracy or prediction error, researchers can measure whether the intelligent system reduces mean time to detect, mean time to repair, packet loss, service downtime, configuration failure rate, SLA violations, unnecessary alarms, or resource overprovisioning. For automated configuration, the evaluation should also consider rollback frequency, policy violations prevented, and the proportion of proposed actions that can be executed safely.

Reproducibility remains particularly important for metaheuristic and hybrid NMS approaches. Population-based optimizers are stochastic, and differences observed in a single run may not reflect stable superiority. Comparative studies should report repeated independent runs, computational budgets, convergence behavior, stopping criteria, population sizes, objective-function evaluations, and model-training conditions. Equal iteration counts do not necessarily imply equal computational effort when algorithms perform different work per iteration.

The same principle applies to hybrid architectures. Adding more optimization stages, deep models, ensembles, or attention mechanisms should not automatically be interpreted as methodological progress. Every additional component should have a distinct function whose contribution can be demonstrated through ablation or controlled comparison. A simpler NMS pipeline may be preferable when it achieves comparable operational benefit with lower latency, clearer behavior, and easier maintenance.

Lifecycle management represents a further research requirement. Intelligent models embedded within an NMS become

software assets that must be versioned, monitored, updated, and retired. The system should record which model produced a recommendation, which telemetry version was used, which configuration was applied, and what operational outcome followed. This provenance is necessary for debugging, audit, and controlled rollback, especially when automated decisions accumulate over long periods.

Governance is therefore inseparable from technical intelligence. Organizations need explicit definitions of which decisions can be automated, which require approval, who is responsible for model updates, how exceptions are handled, and how safety incidents are investigated. Increasing autonomy without equivalent governance can enlarge operational risk even when model accuracy improves.

The broader implication is that intelligent network management is moving from algorithm-centered experimentation toward dependable operational intelligence. The next stage of progress should be measured less by the number of computational techniques combined in one architecture and more by the trustworthiness of the complete path from observation to decision and from decision to verified network action. Systems that remain robust under uncertainty, explainable to operators, secure against manipulation, interoperable across heterogeneous infrastructure, and recoverable after unexpected outcomes will provide greater practical value than models optimized solely for benchmark performance.

4. CONCLUSION

Network management is evolving from predominantly reactive monitoring and manually executed configuration toward an integrated decision-oriented discipline in which telemetry, machine learning, optimization, policy, and programmable control operate within the same management lifecycle. The principal significance of this transition lies not simply in the ability of artificial intelligence to analyze large volumes of network data, but in its potential to transform continuous observability into timely and operationally meaningful actions.

The review demonstrates that intelligent NMS architectures can employ computational intelligence at several distinct levels. Feature-selection mechanisms can reduce telemetry dimensionality, predictive models can estimate traffic and performance conditions, classification can support anomaly and fault detection, metaheuristic optimization can refine models or search configuration and resource spaces, and policy-aware automation can convert analytical results into controlled network changes. These functions should remain clearly distinguished because they solve different problems and require different forms of validation.

Future progress depends on strengthening the connection between analytical performance and dependable operation. Intelligent management systems should quantify uncertainty, tolerate concept drift, preserve topology and policy constraints, verify actions before execution, provide rollback mechanisms, protect telemetry and models against manipulation, and maintain transparent records of automated decisions. Interoperable state representations and hierarchical multi-domain coordination will also become increasingly important as services span enterprise, cloud, edge, wireless, virtualized, and provider-controlled infrastructures.

Ultimately, an intelligent NMS should be evaluated as an operational system rather than as a standalone predictive model. Its value is determined by whether it improves service reliability, reduces fault-resolution time, prevents unsafe configuration, supports efficient resource utilization, and enables automation without sacrificing security or operator control. Achieving these characteristics would move network management beyond isolated AI-assisted analytics toward trustworthy, adaptive, and increasingly autonomous management of complex communication infrastructures.

REFERENCES

[1] F. Amirghafouri, A. A. Neghabi, H. Shakeri, and Y. E. Sola, "Nature-Inspired Meta-Heuristic Algorithms for Resource Allocation in the Internet of Things," *International Journal of Communication Systems*, vol. 38, no. 5, p. e6141, Mar. 2025.

[2] T. Etem, M. R. Baker, and S. Buyrukoğlu, "Hybrid approach for traffic flow detection using metaheuristics optimization and machine learning algorithms," *Journal of the Chinese Institute of Engineers*, pp. 1–21, Dec. 2025.

[3] H. Ouyang, X. Lin, S. Li, L. Gao, and E. H. Houssein, "Recent metaheuristic algorithms for multi-objective feature selection: review, applications, open issues and challenges," *Cluster Computing*, vol. 28, no. 7, p. 467, Sep. 2025.

[4] H. Von Linde and O. Riedel, "A methodology for evaluating feature selection and clustering methods with project-specific requirements," *International Journal of Production Research*, vol. 63, no. 5, pp. 1692–1706, Mar. 2025.

[5] Y. Wang, Y. Yin, H. Zhao, J. Liu, C. Xu, and W. Dong, "Grey wolf optimizer with self-repulsion strategy for feature selection," *Scientific Reports*, vol. 15, Art. no. 12807, 2025.

[6] G. Gelete, H. Gökçekus, Z. M. Yaseen, and T. Gichamo, "Extreme learning machine coupled with heuristic algorithms for daily streamflow modeling at Lake Ziway Watershed, Ethiopia," *Journal of Hydrology*, vol. 660, Art. no. 133345, 2025.

[7] S. Kaushik, A. Bhardwaj, A. Almogren, S. Bharany, A. Altameem, A. Ur Rehman, S. Hussen, and H. Hamam, "Robust machine learning based intrusion detection system using simple statistical techniques in feature selection," *Scientific Reports*, vol. 15, Art. no. 3970, 2025.

[8] B. R. Nayana, R. Subha, R. Radhakrishnan, and P. Geethanjali, "Scalable bearing fault diagnosis using metaheuristic feature selection and machine learning for diverse operating conditions," *Systems Science & Control Engineering*, vol. 13, no. 1, p. 2469606, Dec. 2025.

[9] J. P. Roselyn, "Golden eagle optimization approach for feature selection and XGBoost algorithm-based intrusion detection system in wireless mesh networks of

smart grid." *Journal of the Chinese Institute of Engineers*, vol. 48, no. 8, p. 1312, 2025.

[10] A. Dhiya Eddine, G. Abdelkader, and B. Mourad, "Enhancing IIoT security: a hybrid AI framework for intrusion detection powered by Gen-X AI," *International Journal of Computers and Applications*, vol. 47, no. 9, pp. 749–765, Sep. 2025.

[11] M. Maazalahi and S. Hosseini, "Machine learning and metaheuristic optimization algorithms for feature selection and botnet attack detection," *Knowledge and Information Systems*, vol. 67, no. 4, pp. 3549–3597, Apr. 2025.

[12] X. Gong, Y. Yang, Y. Zhang, N. Li, Y. Guan, and R. Jiang, "Feature selection method for network intrusion based on hybrid meta-heuristic dynamic optimization algorithm," *Computers & Security*, vol. 156, p. 104512, 2025.

[13] X. Liu and H. Tian, "A novel hybrid feature selection method combining binary grey wolf optimization and cuckoo search," *Scientific Reports*, vol. 15, Art. no. 45190, 2025.

[14] S. Nagarajan, D. F. S. Jayapalan, and P. P. L. Devaraj, "Ensemble-based feature selection and optimization-driven deep learning for attack detection in cloud computing," *Knowledge-Based Systems*, vol. 325, p. 113984, 2025.

[15] Y. Pourardebil Khah, M. Hosseini Shirvani, and H. Motameni, "A hybrid machine learning approach for feature selection in designing intrusion detection systems (IDS) model for distributed computing networks," *The Journal of Supercomputing*, vol. 81, no. 1, p. 254, Jan. 2025.

[16] G. Logeswari, K. Thangaramya, M. Selvi, and J. D. Roselind, "An improved synergistic dual-layer feature selection algorithm with two type classifier for efficient intrusion detection in IoT environment," *Scientific Reports*, vol. 15, Art. no. 8050, 2025.

[17] K. Zhang, Y. Wang, U. A. Bhatti, Y. Zhou, and M. Jin, "Enhanced ransomware attacks detection using feature selection, sensitivity analysis, and optimized hybrid model," *Journal of Big Data*, vol. 12, no. 1, p. 245, Nov. 2025.

[18] M. Liang, M. Xu, and S. Wang, "A novel multi-objective evolutionary algorithm for transit network design and frequency-setting problem considering passengers' choice behaviors under station congestion," *Transportation Research Part B: Methodological*, vol. 197, Art. no. 103238, 2025.

[19] M. Alotaibi, H. A. Mengash, H. Alqahtani, A. M. Al-Sharafi, A. E. Yahya, S. R. Alotaibi, A. O. Khadidos, and A. Yafoz, "Hybrid GWQBBA model for optimized classification of attacks in Intrusion Detection System," *Alexandria Engineering Journal*, vol. 116, pp. 9–19, 2025.

[20] K. Halder, A. K. Srivastava, A. Ghosh, S. Das, S. Banerjee, S. C. Pal, U. Chatterjee, D. Bisai, F. Ewert, and T. Gaiser, "Improving landslide susceptibility prediction through ensemble recursive feature elimination and meta-learning framework," *Scientific Reports*, vol. 15, Art. no. 5170, 2025.

[21] K. Veeranjaneyulu, M. Lakshmi, and S. Janakiraman, "Swarm Intelligent Metaheuristic Optimization Algorithms-Based Artificial Neural Network Models for Breast Cancer Diagnosis: Emerging Trends, Challenges and Future Research Directions," *Archives of Computational Methods in Engineering*, vol. 32, no. 1, pp. 381–398, Jan. 2025.

[22] P. Dhal, B. Pradhan, U. Fiore, S. A. J. Francis, and D. S. Roy, "A clinical diabetes prediction based support system based on the multi-objective metaheuristic inspired fine tuning deep network," *Information Fusion*, vol. 122, p. 103188, 2025.

[23] N. Ullah, F. Martínez-Álvarez, I. De Falco, and G. Sannino, "Optimizing deep learning for cotton leaf disease detection using meta-heuristic feature selection algorithms," in *2025 20th Conference on Computer Science and Intelligence Systems (FedCSIS)*," IEEE, pp. 605–614, 2025.

[24] C. L. Kumar, S. Betam, D. Pustokhin, E. L. Lydia, K. Bala, R. Aluvalu, and B. S. Panigrahi, "Metaparameter optimized hybrid deep learning model for next generation cybersecurity in software defined networking environment," *Scientific Reports*, vol. 15, Art. no. 14166, 2025.

[25] A. A. Soladoye, D. B. Olawade, A. O. Esan, N. Aderinto, B. A. Omodunbi, I. A. Adeyanju, and S. Boussios, "Enhancing Parkinson's disease prediction using meta-heuristic optimized machine learning models," *Personalized Medicine*, vol. 22, no. 4, pp. 223–234, Jul. 2025.

[26] R. Younas, H. M. R. Ur Rehman, and G. S. Choi, "Crypto foretell: a novel hybrid attention-correlation based forecasting approach for cryptocurrency," *Journal of Big Data*, vol. 12, Art. no. 229, 2025.

[27] A. M. Elshewey, E. Selem, and A. H. Abed, "Improved CKD classification based on explainable artificial intelligence with extra trees and BBFS," *Scientific Reports*, vol. 15, Art. no. 17861, 2025.

[28] S. Chandragandhi, C. Arvind, and K. Srihari, "Advanced predictive disease modeling in biomedical IoT using the temporal adaptive neural evolutionary algorithm," *Scientific Reports*, vol. 15, Art. no. 20378, 2025.

[29] A. Saleh and M. A. Zulkifley, "Prediction of suspended sediment load in Sungai Semenyih using extreme learning machines and metaheuristic optimization approach," *Journal of Environmental Management*, vol. 380, Art. no. 124987, 2025.

[30] M. Sam'an, M. Mat Deris, and Farikhin, "Feature selection in peer-to-peer lending based on hybrid modified grey wolf optimization with optimized decision tree for

credit risk assessment," *International Journal of Management Science and Engineering Management*, vol. 20, no. 1, pp. 141–158, Jan. 2025.

[31] Y. Wang, N. Yang, and S. Zhao, "A microseismic location method based on BP-GA-GN hybrid algorithm," *Applied Sciences*, vol. 15, no. 23, Art. no. 12569, 2025.

[32] H. Khodaei, F. Nasiri Saleh, A. Nobakht Dalir, and E. Zarei, "Future flood susceptibility mapping under climate and land use change," *Scientific Reports*, vol. 15, Art. no. 12394, 2025.