Zero-Shot Generalization in Combinatorial IoBT Optimization using Hybrid GNN-DRL

Athanasios Gkelias
EEE Department
Imperial College London
London, UK
a.gkelias@imperial.ac.uk

Kin K. Leung
EEE Department
Imperial College London
London, UK
kin.leung@imperial.ac.uk

Patrick J. Baker
Rapid Capabilities Office
Royal Air Force
Farnborough, UK
pbaker@dstl.gov.uk

Olwen Worthington
Cyber & Information Systems
DSTL
Porton Down, UK
olworthington@dstl.gov.uk

***Abstract*—The Internet of Battlespace Things (IoBT) requires the rapid orchestration of heterogeneous assets where communication bandwidth and computational processing power are tightly coupled constraints. In such highly dynamic and adversarial environments, traditional exact optimization methods fail to scale, rendering them obsolete for real-time tactical decision-making. To address this, we propose a Deep Reinforcement Learning (DRL) framework for the Joint Routing and Processing Assignment problem. Our architecture integrates a scale-invariant Graph Neural Network (GNN) with Edge-Conditioned Convolutions to capture topological shifts and a Linear Attention mechanism to resolve inter-task contention with linear complexity O(M). This design allows the agent to generalize zero-shot to network topologies significantly larger than those seen during training. Extensive simulations demonstrate that our framework maintains a low optimality gap (approximately 5.5%) relative to an exact MIP solver while drastically reducing inference latency. For instance, it successfully orchestrates massive surges of 1500 tasks in under 60 seconds—translating to millisecond-level sequential decision-making—proving its viability for sub-second, real-time orchestration in large-scale battlespace networks.**

This paper was originally presented at the International Conference on Military Communication and Information Systems (ICMCIS), organized by the Information Systems Technology (IST) Scientific and Technical Committee, IST-224-RSY – the ICMCIS, held in Bath, United Kingdom, 12-13 May 2026.

***Index Terms*—Internet of Battlespace Things (IoBT), Deep Reinforcement Learning (DRL), Graph Neural Network (GNN), Combinatorial Optimization**

I. Introduction

The Internet of Battlespace Things (IoBT) envisions a highly dynamic, adversarial, and heterogeneous network environment where interconnected assets, ranging from sensors and wearables to autonomous vehicles, generate continuous streams of mission-critical data [1]. Unlike commercial IoT deployments, IoBT networks operate under severe constraints and volatility. Consider a forward-deployed swarm of reconnaissance UAVs generating high-fidelity video feeds (tasks) that must be routed through a contested, multi-hop mesh network to mobile ground vehicles (sinks) for immediate target-recognition processing. In such scenarios, network topology may shift rapidly due to node mobility or destruction, and link quality is subject to stochastic variation and adversarial jamming.

The challenge, however, extends beyond traditional packet routing. Modern tactical applications (e.g., real-time video analytics, sensor fusion) require not only data transmission bandwidth but also significant computational resources for processing [2]. Consequently, the network must solve a *Joint Communication and Computation Resource Allocation* problem. Tasks generated at edge nodes must be selectively admitted, routed through the network, and assigned to suitable processing nodes (sinks) that possess the required computational capacity.

The operational volatility of IoBT environments renders traditional exact optimization approaches inadequate. While the underlying resource allocation problem can be modelled as a Mixed-Integer Linear Program (MILP) [3], solving such problems to optimality is computationally intractable for large-scale networks. Furthermore, in a battlespace where network states change within seconds, an "optimal" solution calculated over minutes is likely obsolete by the time it is deployed. Therefore, the critical requirement is not necessarily finding the global optimum, but rather employing algorithms that prioritize inference speed and robustness, delivering high-quality suboptimal solutions with a minimal optimality gap to maintain operational effectiveness.

In this context, we consider a Software-Defined Networking (SDN) architecture working in tandem with an IoBT Digital Twin [1]—a real-time, high-fidelity virtual replica of the physical battlespace. The Digital Twin serves as the critical synthetic environment where a learning-based agent can safely train on massive volumes of simulated topological shifts, adversarial jamming, and task surges. Once trained, the agent can deploy its zero-shot routing policies to the physical SDN controller without risking live operational collapse.

To power this controller, this paper proposes a Deep Reinforcement Learning (DRL) framework designed to solve the joint admission, routing, and processing problem in real-time. A critical requirement for such a system is *multidimensional scalability*: the ability to handle both significantly larger network topologies and much higher volumes of concurrent tasks than seen during training. Traditional exact solvers fail

This work was supported by the Royal Air Force (RAF), UK, under the "Software Defined Slicing" Project. Athanasios Gkelias and Kin K. Leung were also supported by the EPSRC grant EP/Y037243/1. Gemini AI (Google) was used for editing and grammar enhancement.

to scale with task density due to combinatorial explosion, while heuristic approaches often lack the structural awareness to route effectively in larger maps. Consequently, our approach utilizes a scale-invariant Graph Neural Network (GNN) [4], [5] coupled with a linear-complexity attention mechanism. This allows the agent to transfer knowledge zero-shot to larger battlespace networks while maintaining sub-second inference speeds even under heavy task loads, ensuring robustness against both structural shifts and surges in operational tempo.

A. Related Work

The problem of joint resource allocation in such dynamic environments has attracted considerable research attention. Recent approaches have increasingly leveraged Graph Neural Networks (GNNs) and DRL to address the computational intractability of traditional optimization methods, though these efforts have largely treated the communication and computation layers in isolation.

Pioneering the use of GNNs in networking, Rusek *et al.* [6] demonstrated with *RouteNet* that message-passing architectures can generalize to unseen topologies, accurately predicting delay distributions in Software-Defined Networks (SDN). Building on this topological awareness, Yang *et al.* [7] proposed GROM, a generalized method combining DRL with GNNs to solve routing problems across varying network sizes. Extending these concepts to tactical environments, Lent [8] applied GNNs to "challenged networks," showing that graph-based learning can maintain routing efficiency even under intermittent connectivity. However, these works primarily optimize packet forwarding and link-level metrics, largely abstracting away the computational constraints of the destination nodes (sinks), which are the critical bottleneck in data-intensive IoBT tasks.

Parallel research has focused on the computational aspect, specifically task offloading in Mobile Edge Computing (MEC). Zhao *et al.* [9] developed a framework for UAV-assisted MEC, using Multi-Agent DRL to optimize offloading decisions to minimize system cost. Similarly, Jia *et al.* [10] applied Multi-Agent RL to satellite edge computing, addressing resource allocation in highly dynamic non-terrestrial networks. While Liu *et al.* [11] proposed distributed learning strategies for joint communication and computation in smart cities, recent advancements by Sun *et al.* [12] have integrated attention mechanisms to weigh the importance of user requests dynamically. Nevertheless, these MEC-focused approaches often simplify the underlying multi-hop network into a single-hop abstraction or assume reliable links, failing to account for the adversarial interference (e.g., jamming) prevalent in the battlefield.

Our work distinguishes itself by unifying these fragmented perspectives into a holistic *Joint Routing and Processing* framework. We address the limitations of routing-centric studies [6]–[8], which optimize packet flow but ignore the critical computational bottlenecks at destination sinks, and MEC-centric studies [9], [10], [12], which optimize task placement but frequently neglect the adversarial link degradation inherent to multi-hop tactical networks. Furthermore, we significantly advance beyond the foundational admission control framework we introduced in [13]. While that work relied on standard Graph Attention Networks (GATs) with quadratic $\mathcal{O}(M^2)$ complexity, the current architecture integrates a *Linear Attention* mechanism ($\mathcal{O}(M)$) with *Edge-Conditioned Convolution*. This architectural evolution allows our agent to fully couple link bandwidth with sink CPU constraints and achieve superior zero-shot scalability across both dimensions, handling larger network topologies and denser task environments where traditional solvers and previous GNN iterations become computationally intractable.

The main contributions of this paper are threefold:

- We formulate the *Joint Routing and Processing* problem for IoBT as a holistic optimization challenge, unifying the traditionally isolated domains of packet routing and task offloading into a single decision framework that accounts for simultaneous bandwidth and CPU constraints.
- We propose a scalable Neural Architecture that integrates *Edge-Conditioned Convolutions* with a *Linear Attention* mechanism. This reduces the complexity of inter-task contention modeling from quadratic $\mathcal{O}(M^2)$ to linear $\mathcal{O}(M)$, enabling the system to handle hyper-dense task loads that are computationally intractable for standard Transformer or GAT-based approaches.
- We demonstrate superior *Zero-Shot Generalization* through a rigorous "Constant Load Factor" scaling protocol. We prove that an agent trained on small, static graphs can be deployed effectively on large, dynamic networks without fine-tuning, achieving an optimality gap of $\approx 5.5\%$ relative to the theoretical upper bound.

The remainder of this paper is organized as follows. Section II details the system model and the exact MILP formulation of the optimization problem. Section III describes the proposed Neural Architecture, including the slack feature engineering and the Linear Attention mechanism. Section IV presents the performance evaluation, analyzing the agent's scalability and optimality gap against exact baselines. Finally, Section V concludes the paper.

II. System Model and Problem Formulation

We model the IoBT network as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{N} \cup \mathcal{S}$. The set $\mathcal{N}$ represents generic network nodes (sources and relays), and the set $\mathcal{S}$ represents specialized sink nodes equipped with computational resources. The set $\mathcal{E}$ denotes the communication links between these nodes.

We define a set of candidate tasks $\mathcal{I}$. Each task $i \in \mathcal{I}$ is characterized by a tuple $\{o_i, r_i, \omega_i, u_i\}$, where $o_i \in \mathcal{N}$ is the fixed origin node, r_i is the required data transmission rate (bandwidth), ω_i is the required computational workload (e.g., CPU cycles per second), and u_i is the utility value gained if the task is successfully completed.

The network resources are capacity-constrained:

- Each link $(u, v) \in \mathcal{E}$ has a maximum bandwidth capacity $C_{u,v}$.

- Each sink node $s \in \mathcal{S}$ has a maximum processing capacity P_s.

The objective is to maximize the total utility of admitted tasks by jointly determining: (1) which tasks to admit, (2) which sink node will process each admitted task, and (3) the routing path from the origin to the selected sink.

A. Decision Variables

We introduce the following binary decision variables:

- $x_i \in \{0, 1\}$: Indicates if task i is admitted (1) or rejected (0).
- $y_{i,s} \in \{0, 1\}$: Indicates if task i is processed at sink s.
- $f_{i,(u,v)} \in \{0, 1\}$: Indicates if the flow for task i traverses link (u, v).

B. Optimization Problem

The Joint Routing and Processing Assignment problem is formulated as follows:

$$\text{Maximize} \quad \sum_{i \in \mathcal{I}} u_i x_i \tag{1}$$

Subject to the following constraints:

1) Link Capacity Constraints:

$$\sum_{i \in \mathcal{I}} r_i f_{i,(u,v)} \leq C_{u,v}, \quad \forall (u, v) \in \mathcal{E}. \tag{2}$$

This inequality ensures that the total bandwidth consumed by all traffic flows traversing a specific link (u, v) does not exceed the link's physical bandwidth capacity.

2) Node Processing Constraints:

$$\sum_{i \in \mathcal{I}} \omega_i y_{i,s} \leq P_s, \quad \forall s \in \mathcal{S}. \tag{3}$$

This constraint guarantees that the aggregated computational workload of all tasks assigned to sink s remains within its maximum processing power.

3) Sink Assignment Constraint:

$$\sum_{s \in \mathcal{S}} y_{i,s} = x_i, \quad \forall i \in \mathcal{I}. \tag{4}$$

This constraint ensures that every admitted task ($x_i = 1$) is assigned to exactly one processing sink, while rejected tasks are assigned to none.

4) Flow Conservation (Flexible Destination): At Origin Node o_i:

$$\sum_{v \in \mathcal{V}} f_{i,(o_i,v)} - \sum_{v \in \mathcal{V}} f_{i,(v,o_i)} = x_i, \quad \forall i \in \mathcal{I} \tag{5}$$

At Sink Nodes $s \in \mathcal{S}$:

$$\sum_{v \in \mathcal{V}} f_{i,(v,s)} - \sum_{v \in \mathcal{V}} f_{i,(s,v)} = y_{i,s}, \quad \forall i \in \mathcal{I}, \forall s \in \mathcal{S} \tag{6}$$

At Intermediate Nodes $n \in \mathcal{V} \setminus (\{o_i\} \cup \mathcal{S})$:

$$\sum_{v \in \mathcal{V}} f_{i,(v,n)} - \sum_{v \in \mathcal{V}} f_{i,(n,v)} = 0, \quad \forall i \in \mathcal{I}\ . \tag{7}$$

These equations enforce standard flow conservation with a dynamic destination. Traffic originates at o_i if admitted, terminates at the specific sink s responsible for processing (where $y_{i,s} = 1$), and is conserved at all other intermediate relays.

5) Binary Constraints:

$$x_i, y_{i,s}, f_{i,(u,v)} \in \{0, 1\}\ . \tag{8}$$

Defines the domain of all decision variables as binary.

C. Complexity Analysis

The problem formulated above is a Combinatorial Optimization Problem (COP) characterized by the coupling of multidimensional resource constraints. We analyze its complexity by considering its reduction to known NP-hard problems.

1) Relation to the Unsplittable Flow Problem (UFP): If we relax the computational constraints (i.e., set $P_s = \infty$) and fix a specific destination for each task, the problem reduces to the Unsplittable Flow Problem (UFP) [14]. In the UFP, we seek to select a subset of flows to route through a capacity-constrained graph to maximize profit. The UFP is known to be strongly NP-hard. Since our problem contains UFP as a special case, it is at least as hard as UFP.

2) Relation to the Generalized Assignment Problem (GAP): Conversely, if we relax the network constraints (i.e., set $C_{u,v} = \infty$), the problem effectively becomes a task placement challenge. We must assign tasks (items), each with a specific weight ω_i, to sinks (bins) with capacity P_s to maximize utility. "This maps directly to the Generalized Assignment Problem (GAP) [15], which is closely related to knapsack-type assignment models."

3) Path-Constrained Relaxation and Complexity Reduction: The proposed formulation is strictly harder than either UFP or GAP individually because it requires the simultaneous satisfaction of both constraints. A solution is valid only if a feasible path exists to a sink that also has available processing capacity. The discrete nature of the decision variables ($x_i, y_{i,s}, f_{i,(u,v)}$) introduces a combinatorial explosion in the state space.

To mitigate the intractability of the general formulation described above, in our practical implementation (both for the MIP baseline and the proposed DRL framework), we introduce a *path-constrained relaxation*. Instead of searching the infinite space of all possible graph walks, we pre-compute a fixed set of K candidate paths (e.g., K-shortest paths) for *each valid task-sink pair*. This implies that for every task i, the algorithm considers $K \times |\mathcal{S}|$ potential routing options (plus the rejection option), rather than just K global paths. This relaxation transforms the problem structure significantly. By eliminating the flow conservation constraints (Eq. 5–7) and the continuous flow variables, the problem collapses into a *Multidimensional Knapsack Problem (MDKP)* [16]. Here, each "item" corresponds to a specific tuple (task i, path p, sink s). Selecting an item consumes resources across multiple dimensions: the bandwidth capacities of the specific links in path p and the processing capacity of sink s.

While the MDKP remains NP-hard, the complexity of this relaxed formulation is drastically lower than the original

Unsplittable Flow Problem. The number of decision variables is reduced from $\mathcal{O}(|\mathcal{I}|\cdot|\mathcal{E}|)$ (edge-based flow) to $\mathcal{O}(|\mathcal{I}|\cdot|\mathcal{S}|\cdot K)$ (selection-based). Consequently, the search space complexity is decoupled from the graph diameter and density, scaling instead with the number of processing nodes and the pre-selected path depth K. For the DRL agent, this discretizes the action space into a manageable size ($|\mathcal{S}| \cdot K + 1$ actions per task), enabling efficient policy learning without traversing the explosive combinatorial space of arbitrary routing. Nevertheless, even with this relaxation, the problem remains NP-hard. As the number of tasks $|\mathcal{I}|$, sinks $|\mathcal{S}|$, and candidate paths K increase, the computational time required to find the global optimum using exact methods (e.g., Branch-and-Bound [17]) grows exponentially. This persisting intractability, particularly under strict latency constraints, necessitates the development of learning-based heuristics suitable for practical IoBT deployment.

III. Neural Architecture and Data Flow

As established in the Complexity Analysis (Section II-C), the joint routing and processing problem is NP-hard, with an action space that grows exponentially with network scale. To overcome this intractability, we propose a *Hybrid GNN-DRL* architecture. It is termed "hybrid" because it actively fuses distinct neural paradigms: Edge-Conditioned Convolutions (NNConv) to capture dynamic link physics, Graph Attention (GATConv) for structural topology, and a Linear Attention mechanism to resolve inter-task contention. We deploy this architecture as our primary decision-making agent—hereafter referred to as the *Joint-DRL* agent, reflecting its capacity to concurrently optimize both communication routing and computational processing. This design explicitly decouples inference complexity from network size, enabling zero-shot generalization to larger, unseen topologies.

Before detailing the mathematical formulations, it is helpful to establish the intuitive rationale behind our design choices. Traditional neural networks struggle with dynamic routing because they expect fixed-size inputs, essentially memorizing specific node identities or static map layouts. By utilizing a GNN, our agent instead learns the generalized "physics" of network flow—evaluating how traffic moves through bottlenecks and how sink capacities drain. Furthermore, by incorporating Edge-Conditioned Convolutions, the agent learns to interpret dynamic link conditions, such as adversarial jamming, as varying "pipe widths" that restrict flow, regardless of where those pipes are located.

Additionally, to handle massive task surges, the agent must efficiently resolve contention (i.e., multiple tasks competing for the same bandwidth or CPU). Standard attention mechanisms compare every task against every other task, leading to quadratic computational overhead that scales poorly. Our Linear Attention mechanism avoids this by first creating a single "global context" summary of all network demand, and then evaluating each candidate task against this overall summary. This reduces the complexity from quadratic to linear, enabling the millisecond-level inference speeds required for real-time tactical orchestration.

The core innovation lies in treating the network state not as a fixed-size vector, but as a permutation-invariant graph structure. By learning these local physical rules of flow conservation and resource contention, the agent can effectively transfer its policy to significantly larger and topologically distinct networks without retraining.

A. State Representation and Slack Features

The environment is modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node v possesses features $\mathbf{x}_v \in \mathbb{R}^4$ (Type, CPU Util, Capacity, Degree), and each edge e_{ij} possesses features $\mathbf{e}_{ij} \in \mathbb{R}^3$ (Link Util, Bandwidth, Capacity Ratio).

1) Dynamic Slack Features: To enhance responsiveness to network bottlenecks, we compute a set of heuristic *slack features* $\mathbf{x}_{\text{slack}}^{(k)}$ for each candidate path P_k:

$$\mathbf{x}_{\text{slack}}^{(k)} = \left[\tilde{B}_{\text{bottle}}^{(k)}, \tilde{C}_{\text{sink}}^{(k)}, E_{\text{eff}}^{(k)}\right], \tag{9}$$

where $\tilde{B}_{\text{bottle}}^{(k)} = \min_{e\in P_k}(C_e^{\text{resid}})/C_{\max}$ identifies the critical bottleneck, and $\tilde{C}_{\text{sink}}^{(k)}$ is the normalized residual CPU at the sink. The Efficiency Ratio $E_{\text{eff}}^{(k)}$ relates task utility u_i to resource consumption:

$$E_{\text{eff}}^{(k)} = \frac{u_i}{\left(r_i \cdot \frac{|P_k|}{|\mathcal{V}|}\right) + \omega_i + \epsilon}\ . \tag{10}$$

Intuitively, this metric quantifies the overall resource efficiency of a task. It evaluates the operational utility gained relative to the combined cost of network bandwidth and sink processing power required for its execution. Here, r_i and ω_i are bandwidth and CPU requirements. Crucially, the path length $|P_k|$ is normalized by the total network size $|\mathcal{V}|$. This normalization is a key enabler of *scale invariance*, ensuring that the agent perceives relative efficiency consistently whether operating on a 10-node training graph or a 100-node deployment network.

B. Topology-Aware Graph Encoder

The encoder transforms raw observations into latent node embeddings $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}|\times d}$ ($d = 64$). To capture the impact of edge constraints (e.g., jamming) on node reachability, we employ an *Edge-Conditioned Convolution (NNConv)*. Intuitively, while a standard GCN treats all network connections as identical wires, NNConv treats them as pipes of varying widths—where a degraded link actively throttles the flow of information between nodes during the embedding phase. It uses a learned kernel generating network (MLP) to dynamically modulate message passing based on edge attributes $\mathbf{e}_{ij}$. The update rule for node i is:

$$\mathbf{h}_i' = \mathbf{\Theta}\mathbf{x}_i + \sum_{j\in N(i)} \text{MLP}(\mathbf{e}_{ij}) \cdot \mathbf{x}_j\ . \tag{11}$$

This mechanism serves a dual purpose: (1) it allows the network to adapt to fluctuating link qualities, identifying jammed links as "high-cost" edges, and (2) it ensures topological generalization by processing local neighborhoods identically

regardless of global graph size. Once physical link conditions are captured, this is followed by a **Graph Attention Layer (GATConv)** [5]. This allows nodes to weigh the relative structural importance of their neighbors, enabling the agent to prioritize signals from critical central routers while ignoring isolated nodes.

C. Multi-View Path Aggregation

Once node embeddings are generated, the agent must summarize the quality of a candidate path P_k. A simple mean embedding is insufficient for routing, as it masks critical outliers. For example, a path with five high-capacity links and one completely severed link might yield an acceptable mean, but is operationally useless. To capture these extremes, we introduce a *Multi-View Path Aggregator* that explicitly tracks the minimum, maximum, sum, and variance of the nodes along P_k:

$$\mathbf{h}_{P_k} = \text{Concat}\left(\mathbf{h}_{\max}, \mathbf{h}_{\text{mean}}, \mathbf{h}'_{\text{sum}}, \mathbf{h}_{\text{std}}, \mathbf{h}_{\min}\right) . \tag{12}$$

Crucially, the sum component utilizes a square-root normalization $\mathbf{h}'_{\text{sum}} = \frac{1}{\sqrt{|P_k|}} \sum_{v \in P_k} \mathbf{h}_v$. This ensures the embedding magnitude remains invariant to path length variations, preventing gradient explosion for longer routes in large networks and further supporting the transferability of the learned policy to larger topologies.

D. Contention-Aware Linear Attention

To resolve inter-task dependencies and resource contention, the agent must evaluate candidate k against all other M candidates. Standard Self-Attention scales quadratically $O(M^2)$, which is prohibitive for high task loads. We implement a *Linear Attention* mechanism with $O(M)$ complexity.

First, we generate a candidate vector $\mathbf{c}_k$ via *Fusion Layer* 1 (Local Candidate Fusion), concatenating the task features $\mathbf{f}_{\text{task}}$, the path vector $\mathbf{h}_{P_k}$, and the slack features $\mathbf{x}^{(k)}_{\text{slack}}$. Let $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{M \times d}$ be the query, key, and value matrices projected from the batch of candidate vectors. Instead of the standard Softmax operation—which strictly requires pairwise comparisons of all elements—we compute a global context vector using a positive feature map $\phi(x) = \text{elu}(x) + 1$:

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \frac{\phi(\mathbf{Q}) \sum_{j=1}^{M} (\phi(\mathbf{k}_j)^T \mathbf{v}_j)}{\phi(\mathbf{Q}) \sum_{j=1}^{M} \phi(\mathbf{k}_j)^T} . \tag{13}$$

By replacing Softmax with this mathematically decoupled formulation, the agent can construct a "Global Contention Context"—a unified summary of all competing requests—without ever building the massive $M \times M$ attention matrix. This *linear scaling* is essential for handling the increased task density encountered when scaling from training scenarios to complex operational environments.

E. Global Fusion and Q-Value Estimation

The final stage, *Fusion Layer* 2 (Global State Fusion), assembles the comprehensive state for the decoder. For each candidate, we concatenate: (1) The local Candidate Vector $\mathbf{c}_k$, (2) The Global Context Vector from the attention layer, and (3) A Global Graph Embedding (aggregated from all nodes). This global embedding acts as a macro-level weather report for the network, informing the agent whether the overall battlespace is relatively clear or heavily congested. This comprehensive state vector is passed through an MLP Decoder ($256 \to 128 \to 64 \to 1$) to predict the scalar Q-value. The network is trained to minimize the Temporal Difference (TD) error using a Double DQN objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathcal{B}} \left[\left(y_j - Q(s, a; \theta) \right)^2 \right], \tag{14}$$

where the target $y_j = r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta^-)$ uses the target network θ^-. In highly volatile IoBT environments, standard Q-learning tends to be overly optimistic about the value of unexplored paths. The Double DQN approach mitigates this by decoupling action selection from value evaluation, ensuring the agent learns a robust, conservative policy rather than chasing artificially inflated routing options.

F. Inference Complexity and Scalability Analysis

As established in Section IV, the exact MIP formulation suffers from exponential complexity, bounded in the worst case by $\mathcal{O}(2^{|\mathcal{I}| \cdot |\mathcal{S}| \cdot K})$. While acceptable for offline benchmarking, this intractability renders exact solvers obsolete for real-time tactical decisions.

In contrast, the computational cost of the proposed DRL agent during deployment is deterministic and polynomial. The inference complexity is driven by two primary stages:

a) Topology Encoding: The GNN processes the network state using sparse message passing. Since IoBT networks are sparse graphs (average degree $d \ll |\mathcal{V}|$), the complexity of the `NNConv` and `GAT` layers scales with the number of edges rather than the square of nodes:

$$\mathcal{T}_{\text{GNN}} \in \mathcal{O}\left(L \cdot (|\mathcal{V}| + |\mathcal{E}|) \cdot d_{\text{model}}\right), \tag{15}$$

where L is the number of GNN layers and d_{model} is the feature embedding size.

b) Linear Attention Scoring: Standard attention mechanisms scale quadratically $\mathcal{O}(M^2)$ with the number of task candidates M. To ensure scalability, we implement *Linear Attention*. By computing the global context $\mathbf{C} = \sum_j \phi(\mathbf{k}_j)^T \mathbf{v}_j$ first, we avoid constructing the massive $M \times M$ attention matrix. The complexity of the scoring head is:

$$\mathcal{T}_{\text{Score}} \in \mathcal{O}\left(M \cdot (d^2_{\text{model}} + l_{\text{path}})\right), \tag{16}$$

where l_{path} is the maximum path length for feature extraction.

The aggregated inference complexity is linear with respect to the number of tasks:

$$\mathcal{T}_{\text{Total}} \approx \mathcal{O}(|\mathcal{E}|) + \mathcal{O}(M) . \tag{17}$$

This linear scaling $\mathcal{O}(M)$ versus the exponential scaling $\mathcal{O}(2^M)$ of the MIP solver represents the fundamental architectural advantage, allowing the agent to generate valid routing decisions in milliseconds independent of the exponentially growing state space.

IV. Performance Evaluation

To validate the proposed framework, we conducted extensive simulations comparing the DRL agent against an exact optimal solver. Our evaluation focuses on two key performance indicators: the Optimality Gap, which measures the quality of the DRL decisions relative to the theoretical upper bound, and Inference Scalability, demonstrating the architecture's ability to maintain sub-second decision-making as network size and task volume grow.

A. Simulation Environment

We simulate the IoBT network as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ generated via the Erdős–Rényi model [18] to approximate ad-hoc tactical deployments. The node set $\mathcal{V}$ is partitioned into generic network nodes $\mathcal{N}$ (sources and relays) and specialized sink nodes $\mathcal{S}$ equipped with computational resources. Traffic is generated as a batch of tasks $\mathcal{I}$. Each task $i \in \mathcal{I}$ is assigned a random origin $o_i \in \mathcal{N}$, a bandwidth requirement r_i, a processing requirement ω_i, and a mission utility u_i derived from a uniform distribution.

To demonstrate the zero-shot generalization capabilities of the GNN-based architecture, we employ distinct regimes for training and testing. The agent is trained on small-scale scenarios ($|\mathcal{N}| = 10$, $|\mathcal{I}| = 100$) to allow for rapid episode iteration. Conversely, testing is performed on scaled instances with $|\mathcal{N}|$ reaching up to 30 and the task volume $|\mathcal{I}|$ increasing up to 1500. This represents a $3\times$ increase in graph size and a $15\times$ increase in task density, rigorously testing the model's adaptability to unseen environments.

B. Resource Scaling and Constraint Tightness

A critical methodological challenge in benchmarking routing algorithms is distinguishing between architectural scalability and simple resource saturation. If network capacities are held fixed while the task volume $|\mathcal{I}|$ increases (e.g., from 100 to 1500 tasks), the optimization problem degenerates from a complex joint routing challenge into a trivial admission control problem, where the optimal policy is simply to reject the vast majority of traffic due to gross infeasibility.

To ensure a rigorous evaluation of the combinatorial packing capabilities of our agent, we adopt a *Constant Load Factor* scaling approach. As the number of tasks $|\mathcal{I}|$ increases, we scale link capacities $C_{u,v}$ and sink processing limits P_s linearly, such that the aggregate system capacity grows extensively with demand (i.e., $\sum C_{u,v} \propto |\mathcal{I}|$ and $\sum P_s \propto |\mathcal{I}|$). This protocol maintains a consistent demand-to-capacity ratio (constraint tightness) across all test scales. By keeping the system in the critical phase transition region—where resources are scarce enough to require intelligent trade-offs but sufficient to allow for feasible high-utility solutions—we preserve the NP-hard nature of the problem. This allows us to isolate the impact of network scale on inference latency and solution quality without the confounding effects of saturation.

C. Baselines and Metrics

We evaluate the performance of our proposed Joint-DRL agent against an Optimal MIP Solver (SCIP). We implement the exact formulation described in Section II-B using the SCIP optimization suite [19]. Because the complexity of NP-hard problems varies significantly depending on the specific random topology and task parameters, exact solvability is evaluated on an episode-by-episode basis. For individual episodes where SCIP converges within the 600-second time limit, we obtain the ground-truth global optimum. For episodes that hit the timeout—which becomes the dominant outcome as task density grows—we utilize the best incumbent solution found by SCIP as the baseline (U_{baseline}). Crucially, across all such timeout instances, the solver's internal MIP gap (the difference between the incumbent solution and the theoretical upper bound) remained strictly below 2%, guaranteeing that our baseline represents a highly rigorous near-optimal threshold.

Performance is reported using two primary metrics. First, the Optimality Gap (%), defined as $(1 - \frac{U_{\text{DRL}}}{U_{\text{baseline}}}) \times 100$, quantifies how close the DRL solution is to this exact baseline. Second, we measure Execution Time (s), representing the wall-clock time required to generate a valid assignment for the entire batch $\mathcal{I}$, which serves as a proxy for the feasibility of deployment in real-time tactical loops.

D. Training Regimes and Generalization Strategy

To systematically evaluate the agent's ability to learn generalized networking rules versus simply memorizing specific maps, we define three distinct environmental complexity regimes for training. In the *Static Infrastructure* regime, the network topology and resource capacities remain constant, representing a stable deployment where agents might risk overfitting to specific node identities. The *Variable Resources* regime fixes the physical graph but randomizes link and sink capacities at the start of each episode, simulating adversarial jamming or background traffic spikes that force the agent to interpret real-time state features rather than static properties. Finally, the *Dynamic Topology* regime regenerates the entire graph structure and resource allocations in every episode, compelling the GNN to learn abstract, transferable topological rules that apply universally regardless of the specific network layout.

Crucially, regardless of the training regime employed, we strictly enforce a "Zero-Shot" evaluation protocol. All testing is conducted exclusively in the *Dynamic Topology* setting on scaled-up networks ($|\mathcal{V}| = 30$) that are significantly larger and structurally distinct from the training instances ($|\mathcal{V}| = 10$). This approach rigorously validates that the learned policy is scale-invariant and robust to the topological shifts inherent in ad-hoc battlespace networks, ensuring the agent has learned to optimize generic graph flows rather than memorizing a specific training environment.

E. Numerical Results

We evaluate the performance of the proposed Joint-DRL agent on a test set of 50 randomly generated episodes per task

density point. To ensure reproducible baseline comparisons, all simulations and timing benchmarks were executed on a standard workstation equipped with an Intel Core i7-8700K CPU @ 3.70GHz and 64 GB of RAM, strictly utilizing CPU processing without GPU acceleration. The MIP baseline performance is reported with a relaxed time limit of 600 seconds (10 minutes) to approximate a high-quality "ground truth."

1) Solution Quality and Generalization: Table I and Fig. 1 present the total utility and optimality gap achieved by the agent across the three different training regimes. The results reveal a clear hierarchy of generalization. As shown in Table I, the agent trained on *Dynamic Topologies* consistently achieves the highest utility, closely tracking the MIP baseline (which, as previously established, represents either the global optimum or a near-optimal solution with a sub-2% internal gap). For the largest instance ($|\mathcal{I}| = 1500$), the Dynamic agent achieves a utility of 3318.2 compared to the MIP solver's 3509.1. The *Variable Resources* agent follows closely, adapting well to bottlenecks but suffering slightly from unseen topological structures. In contrast, the agent trained on *Static Infrastructure* exhibits the lowest utility (3089.9 at $|\mathcal{I}| = 1500$), indicating that it partially overfitted to its training graph. This trend is mirrored in the optimality gap (Fig. 1), where the Dynamic regime maintains a low gap between 5.4% and 6.6%, significantly outperforming the Static baseline ($\approx 12\%$).

2) Inference Scalability: The decisive advantage of the proposed framework is quantified in Fig. 2. For small instances ($|\mathcal{I}| = 100$), the MIP solver is efficient (15.95s). However, as the combinatorial search space expands, MIP execution time grows exponentially. As explicitly depicted in Fig.2, the execution time for the MIP solver abruptly plateaus at exactly 600 seconds for instances larger than $|\mathcal{I}| = 750$. This flatline occurs not because the solver successfully found an optimal solution, but because it triggered our predefined 10-minute timeout limit, failing to converge within a practical timeframe.

In sharp contrast, the DRL agent exhibits polynomial scalability. To ensure a rigorous and fair comparison, the MIP solver and all three DRL training regimes were evaluated on the exact same set of 50 test topologies per task density point. While the MIP execution time is averaged over these 50 episodes, the DRL inference time—which rises linearly to only 58.6s for the massive 1500-task batch—is averaged across 150 evaluations (the 50 topologies $\times$ 3 regimes). Because the neural network architecture is structurally identical across all regimes, inference complexity is strictly independent of the learned weights, allowing us to safely aggregate these execution times for a highly robust latency measurement.

Furthermore, because the DRL agent processes tasks sequentially, this aggregate batch execution time corresponds to an individual decision latency of under 40 milliseconds. *Operationally, this allows the network controller to begin pipelining and deploying tasks immediately as they are approved, whereas exact solvers require the entire network to wait—often for minutes—until a simultaneous global assignment is found.* This represents a speed-up factor of $> 10\times$ at the largest scale, validating that the architecture effectively decouples computational complexity from the exponential growth of the solution space.

TABLE I
TOTAL UTILITY COMPARISON VS. TASK DENSITY

Tasks ($\lvert\mathcal{I}\rvert$)	MIP Baseline	Joint-DRL Training Regimes		
		Dynamic	Variable	Static
100	223.8	209.4	204.8	202.7
250	568.2	533.7	525.8	506.0
500	1161.2	1095.6	1083.4	1026.0
750	1740.1*	1639.9	1616.5	1527.1
1000	2333.1*	2200.6	2182.5	2052.1
1500	3509.1*	3318.2	3304.7	3089.9

*Average execution time hit the 600s timeout; baseline contains incumbent solutions (gap $< 2\%$).

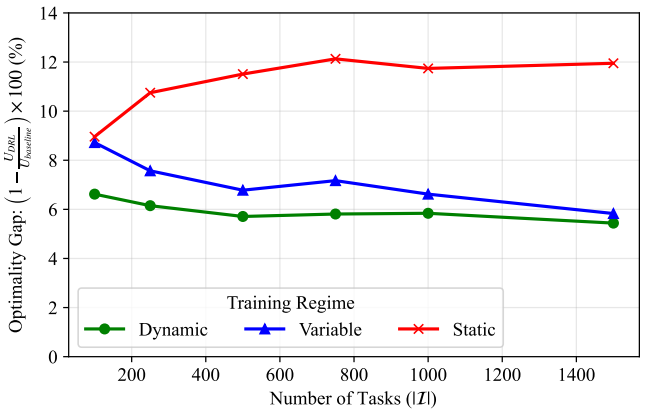

Fig. 1. Optimality Gap comparison. The gap is calculated relative to the exact MIP baseline, which represents either the guaranteed global optimum or a high-quality incumbent solution (sub-2% internal MIP gap) upon reaching the 600s timeout. The Dynamic training regime achieves a significantly lower gap ($\approx 5.5\%$) than the Static baseline ($\approx 12\%$).

V. BROADER APPLICABILITY AND GENERALIZED USE CASES

While the proposed Hybrid GNN-DRL framework is evaluated within the context of IoBT communication and computation routing, the underlying mathematical formulation—a dynamic, multidimensional resource allocation problem over a permutation-invariant graph—has broad applicability. Traditional exact solvers fail in any domain characterized by severe spatiotemporal volatility and combinatorial explosion. Consequently, the scale-invariant architecture proposed in this paper can be generalized to several other critical defense and civilian domains where rapid decision-making is paramount.

In modern air force operations, dynamic mission optimization is structurally identical to the joint routing problem. Airbases and targets can be abstracted as nodes, flight corridors as edges, and strike or reconnaissance missions as tasks. Flight corridors are highly volatile due to pop-up anti-access/area denial (A2/AD) threat zones or stochastic radar coverage, while aircraft face multidimensional constraints such as fuel capacity and payload limits. The Edge-Conditioned Convolutions in our architecture naturally translate to modeling dynamic threat

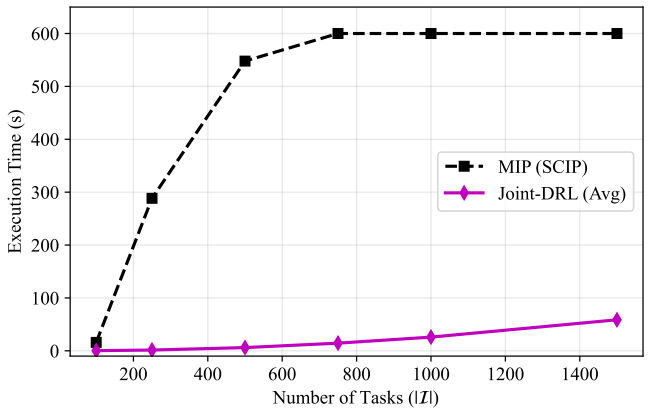

Fig. 2. Execution Time vs. Task Density. Both methods were evaluated on the exact same 50 test topologies per data point. The average execution time of the exact solver hits the 600s limit for $|\mathcal{I}| \geq 750$ (with individual episodes varying between exact convergence and timeout), while the Joint-DRL agent scales linearly. The Joint-DRL runtime is reported as the average (Avg) across 150 runs (the identical 50 topologies evaluated across all 3 training regimes), as inference complexity is independent of the learned policy.

levels along a flight path. A corridor blocked by a sudden surface-to-air missile threat is mathematically equivalent to a jammed communication link. Our DRL agent can provide near real-time, zero-shot re-routing and target reallocation for autonomous drone swarms mid-flight, whereas traditional solvers would stall the mission while computing a new global optimum.

Similarly, military logistics in contested environments require the swift routing of physical goods through a network of supply depots and forward operating bases. Supply routes are frequently disrupted by adversarial ambushes or destroyed infrastructure, and transport vehicles act as processing sinks with strict weight and volume constraints. Surges in resupply requests during a kinetic engagement create a massive combinatorial bottleneck. Our Linear Attention mechanism, which reduces inter-task contention modeling to linear complexity, allows a centralized logistics controller to prioritize and assign hundreds of concurrent supply tasks to available convoys instantaneously. The GNN's topological awareness ensures that if a primary route is severed, the policy inherently understands how to route through secondary paths without requiring retraining.

Beyond defense, the framework is directly applicable to civilian emergency management, such as post-disaster response operations. Disasters physically alter the graph topology of a city, rendering roads impassable and dynamically degrading the capacity of local hospitals and emergency shelters. In this scenario, tasks represent civilian rescue requests, while sinks represent medical facilities with finite processing capacities. The heuristic slack features designed into our state representation actively monitor the residual capacity of these sinks. The GNN-DRL agent can autonomously orchestrate a fleet of emergency vehicles, continuously adapting to the degrading infrastructure and ensuring that critical tasks are rapidly routed to the nearest viable facility.

VI. Conclusion

This paper presented a robust, Hybrid GNN-DRL framework for solving the NP-hard Joint Routing and Processing problem in dynamic IoBT environments. By moving beyond fixed-input representations to a permutation-invariant graph architecture, we successfully decoupled inference complexity from network size. The integration of Edge-Conditioned Convolutions, Linear Attention, and heuristic slack features allowed our Joint-DRL agent to learn generalized, physical rules of resource contention. Experimental results confirm that the proposed agent achieves superior zero-shot generalization, maintaining a strict optimality gap of approximately 5.5% relative to the exact SCIP baseline on scaled, unseen topologies. Crucially, by reducing contention modeling to linear $\mathcal{O}(M)$ complexity, the agent successfully orchestrated massive surges of 1500 tasks with sub-second execution times. Operationally, this sequential processing capability allows the network controller to begin pipelining and deploying tasks immediately as they are admitted, whereas traditional exact solvers force the entire battlespace network to stall—often for minutes—waiting for a simultaneous global assignment.

Looking ahead, a critical vulnerability of existing DRL applications in tactical networks is their fundamental reliance on correlational learning rather than causal inference. While current RL solutions excel at mapping observed states to high-reward actions, they inherently capture statistical correlations without understanding the underlying causal mechanisms—such as distinguishing between naturally congested links and targeted adversarial jamming. In mission-critical defense settings, acting on spurious correlations can lead to catastrophic policy collapse if an adversary abruptly shifts tactics. Therefore, future work will extend this framework to decentralized multi-agent settings and formally integrate causal representation learning. By building upon the interpretable, physics-based foundation established in this paper, we aim to develop agents capable of isolating and mitigating the true causal sources of network degradation.

References

[1] A. Gkelias, P. J. Baker, K. K. Leung, O. Worthington, and C. Melville, "Digital twins for Internet of Battlespace Things (IoBT) coalitions," in *Proc. Int. Conf. Mil. Commun. Inf. Syst. (ICMCIS)*, 2025.

[2] A. Gkelias, N. K. Panigrahy, M. Mobayenjarihani, K. K. Leung, D. Towsley, P. J. Baker, O. Worthington, and L. Fowkes, "Resource management in software defined coalitions (SDC) through slicing," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, San Diego, CA, USA, Nov./Dec. 2021.

[3] L. A. Wolsey, *Integer Programming*, 2nd ed. Hoboken, NJ, USA: John Wiley & Sons, 2020.

[4] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[5] P. Veličković *et al.*, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[6] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, "RouteNet: Leveraging graph neural networks for network modeling and optimization in SDN," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2260–2270, Oct. 2020.

[7] Q. Yang, S. Xia, T. Zhang, and Z. Li, "GROM: A generalized routing optimization method with graph neural network," *Comput. Commun.*, vol. 220, pp. 12–23, Jun. 2024.
[8] R. Lent, "Dynamic routing in challenged networks with graph neural networks," in *Proc. IEEE Latin-Amer. Conf. Commun. (LATINCOM)*, Rio de Janeiro, Brazil, Nov.–Dec., pp. 1–6, 2022.
[9] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, and D. Niyato, "Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 6949–6960, Sep. 2022.
[10] M. Jia, L. Zhang, J. Wu, Q. Guo, G. Zhang, and X. Gu, "Deep multiagent reinforcement learning for task offloading and resource allocation in satellite edge computing," *IEEE Internet Things J.*, vol. 12, no. 4, pp. 3832–3845, Feb. 2025.
[11] T. Liu, R. Luo, F. Xu, C. Fan, and C. Zhao, "Distributed learning based joint communication and computation strategy of IoT devices in smart cities," *Sensors*, vol. 20, no. 4, art. 973, Feb. 2020.
[12] Z. Sun, H. Yang, C. Li, Q. Yao, Y. Teng, J. Zhang, S. Liu, Y. Li, and A. V. Vasilakos, "A resource allocation scheme for edge computing network in smart city based on attention mechanism," *ACM Trans. Sen. Netw.*, vol. 20, no. 3, art. 60, Mar. 2024.
[13] A. Gkelias, K. K. Leung, P. J. Baker, and O. Worthington, "GNN-enabled reinforcement learning for robust task admission and routing in IoBT environments," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Los Angeles, CA, USA, 2025.
[14] J. M. Kleinberg, "Single-source unsplittable flow," in *Proc. 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 68–77, 1996.
[15] D. G. Cattrysse and L. N. Van Wassenhove, "A survey of algorithms for the generalized assignment problem," *Eur. J. Oper. Res.*, vol. 60, no. 3, pp. 260–272, 1992.
[16] A. Fréville, "The multidimensional 0-1 knapsack problem: An overview," *Eur. J. Oper. Res.*, vol. 155, no. 1, pp. 1–21, 2004.
[17] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960.
[18] P. Erdős and A. Rényi, "On random graphs I," *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.
[19] K. Bestuzheva *et al.*, "The SCIP Optimization Suite 8.0," Zuse Institute Berlin, ZIB-Report 21-41, Dec. 2021.