Enabling Large Language Model based Data Synthesis in Wireless Mesh Network Configuration for Internet of Things

Aitian Ma, Jean Marco Cruz, Dongsheng Luo, Mo Sha
Knight Foundation School of Computing and Information Sciences
Florida International University
{aima, jcruz290, dluo, msha}@fiu.edu

***Abstract*—Wireless Mesh Networks (WMNs) are essential for many Internet of Things (IoT) applications, such as industrial automation, environmental monitoring, and smart cities. Today, configuring a WMN to meet its stringent performance requirements remains a significant challenge due to dynamic real-world wireless conditions and operating environments. The simulation-to-reality gap in network configuration further complicates the generalization of models trained solely with simulation data, leading to suboptimal performance in physical deployments. To address such challenges, we develop *WMN-LLM-DS*, a novel framework that integrates Large Language Models (LLMs) for synthetic data generation with domain adaptation techniques to better configure WMNs. Leveraging LLMs, WMN-LLM-DS generates high-quality, diverse synthetic datasets conditioned on real-world constraints, effectively bridging the simulation-to-reality gap and enriching the diversity of training data. WMN-LLM-DS employs a teacher-student architecture to transfer network configuration knowledge learned from simulations to physical deployments, enhanced by custom loss functions to align feature representations across different domains. Experiments conducted on datasets collected from a physical testbed and network simulators demonstrate that WMN-LLM-DS outperforms existing solutions, achieving an improvement of up to 10.8% in prediction accuracy while also exhibiting strong domain generalization capabilities.**

***Index Terms*—Internet of Things, Wireless Mesh Networks, Network Configuration, Large Language Models, Data Synthesis**

I. Introduction

Wireless Mesh Networks (WMNs) are critical for enabling robust, scalable, and flexible communication across a diverse range of Internet of Things (IoT) applications, including industrial automation [1], environmental monitoring [2], [3], [4], and smart cities [5]. Despite decades of research advancing the capabilities of WMNs, configuring these networks remains a complex challenge due to diverse operating conditions, stringent performance requirements for reliability, latency, and energy efficiency, and the dynamic nature of real-world deployments [6]. These challenges are particularly pronounced in applications with strict real-time communication constraints [7].

Simulations have long been employed as a cost-effective alternative to experimentation on physical testbeds for identifying optimal WMN configuration. They provide significant advantages, including reduced setup time, lower overhead, and the ability to explore various configurations under controlled conditions. However, simulations often fail to capture the extensive uncertainties, dynamics, and variations inherent in real-world WMNs. Such a limitation leads to a significant simulation-to-reality gap, where network configuration derived from simulations may not perform as expected in physical deployments. Recent research has emphasized the importance of bridging this gap, often leveraging domain adaptation techniques to align simulation-derived insights with real-world requirements [8], [9], [10].

Domain adaptation has proven effective for scenarios where labeled data is abundant in a source domain (e.g., simulations) but limited in a target domain (e.g., physical deployments). Traditional domain adaptation approaches focus on aligning feature distributions between domains to enable models trained on source data to generalize effectively to the target domain [11]. However, existing methods for WMN configuration have limitations as they often fail to consider the variability in the simulation-to-reality gap across different network configurations [8], [10]. This oversight may lead to suboptimal alignment, where samples from different network configurations are misaligned, leading to poor generalization. Furthermore, these methods may overfit source domain data, resulting in decision boundaries that are less discriminative for target domain samples, particularly in complex, real-world settings.

The rise of Large Language Models (LLMs) has introduced new opportunities for addressing challenges in domain adaptation and data synthesis. These models, such as GPT and related architectures, are pre-trained on vast amounts of data and exhibit exceptional generalization capabilities across various tasks. Despite their transformative potential, fine-tuning LLMs for specific applications remains resource-intensive, often requiring substantial computational power, large-scale datasets, and significant training time. Additionally, the optimization of hyperparameters during fine-tuning further contributes to the complexity and cost of deployment. However, the ability of LLMs to generate high-quality, diverse, and contextually rich synthetic data has opened new possibilities for tackling domain adaptation challenges. By leveraging their inherent un-

derstanding of natural language and their capacity to condition outputs on metadata, LLMs can produce synthetic datasets that capture nuanced domain-specific features and relationships. This capability allows LLMs to significantly expand training datasets, improve representation diversity, and enhance model generalization, making them a powerful tool for bridging the simulation-to-reality gap.

To address these limitations, we develop *WMN-LLM-DS*, a novel framework that leverages LLMs for data synthesis to enhance domain adaptation in WMN configuration tasks. By integrating LLM-driven synthetic data generation with traditional domain adaptation, WMN-LLM-DS significantly expands the diversity and representativeness of the training dataset. Such an approach enables the model to capture both common and configuration-specific features, effectively bridging the simulation-to-reality gap. Unlike existing methods, WMN-LLM-DS focuses on augmenting the limited physical data in the target domain ($\mathcal{D}_T$) with high-quality synthetic data ($D_{\text{synthetic}}$) generated from LLMs. These synthetic datasets are conditioned on metadata descriptions that encode domain-specific constraints and performance requirements, ensuring that the generated data is both realistic and diverse.

Additionally, WMN-LLM-DS employs a teacher-student model framework to facilitate knowledge transfer from simulations to physical deployments. The teacher model, trained on abundant simulation data ($\mathcal{D}_S$), serves as a knowledge base, while the student model is trained on the augmented dataset ($\mathcal{D}_{\text{augmented}}$) to adapt to the real-world domain. To address domain shift effectively, WMN-LLM-DS introduces multiple loss functions, including classification loss, network configuration loss, and simulation-to-reality loss. These losses ensure robust alignment of feature representations across domains, enabling the model to generalize well to physical deployments.

This paper makes the following contributions:

1) We propose WMN-LLM-DS, a framework that combines LLM-driven data synthesis with domain adaptation to address the complexities of WMN configuration across diverse network conditions.
2) To the best of our knowledge, WMN-LLM-DS is the first framework to leverage LLMs for generating synthetic datasets specifically tailored for domain adaptation tasks. This approach enhances the diversity, representativeness, and quality of training data.
3) We validate the effectiveness of WMN-LLM-DS through experiments on datasets collected from physical WMN testbeds and multiple wireless simulators, demonstrating substantial improvements in prediction accuracy and domain generalization.

The remainder of this paper is organized as follows. Section II presents our empirical study on the simulation-to-reality gap and motivates the use of synthetic data generation. Section III details the design of WMN-LLM-DS and our training method. Section IV presents experimental results. Section V discusses related work. Section VI concludes this paper.

II. Empirical Study

This section presents our empirical study that investigates the simulation-to-reality gap in WMN configuration, quantifies its effects on prediction accuracy, and explores the feasibility of using synthetic data generation as a bridging solution. We conducted the study using data collected from a 50-node WirelessHART testbed and four network simulators [12].

A. Problem Formulation of WMN Configuration

We formulated the WMN configuration task as a supervised learning problem that maps network performance requirements to optimal parameter settings. Given a target performance vector $\mathbf{p} = [L_{\text{target}}, B_{\text{target}}, E_{\text{target}}]$ specifying desired latency, battery lifetime, and reliability, the goal is to predict the configuration vector $\mathbf{c} = [R, C, A]$ that achieves these requirements in a physical deployment. The WMN configuration parameters are as follows:

- *Packet Reception Ratio (PRR) threshold* ($R \in [0.7, 0.9]$): Minimum link quality for route inclusion;
- *Communication channels* ($C \in \{1, 2, ..., 8\}$): Number of frequency channels allocated;
- *Transmission attempts* ($A \in \{1, 2, 3\}$): Maximum transmission attempts per packet.

The WMN performance metrics are as follows:

- *End-to-end latency* (L): Time for packet delivery from source to destination (ms);
- *Battery lifetime* (B): Expected operational duration based on energy consumption ($hours$);
- *End-to-end reliability* (E): Packet delivery ratio from source to destination during the evaluation period (%).

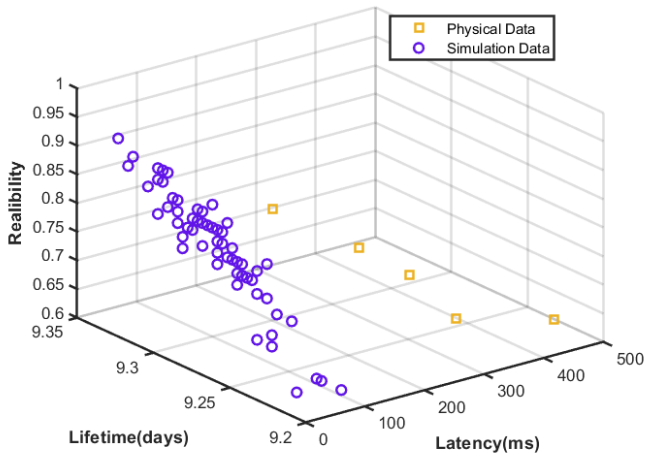
We collected performance measurements for 88 different network configurations, with 75 samples per configuration from the physical deployment and each of four network simulators (NS-3, Cooja, TOSSIM, and OMNeT++).

B. Quantifying the Simulation-to-Reality Gap

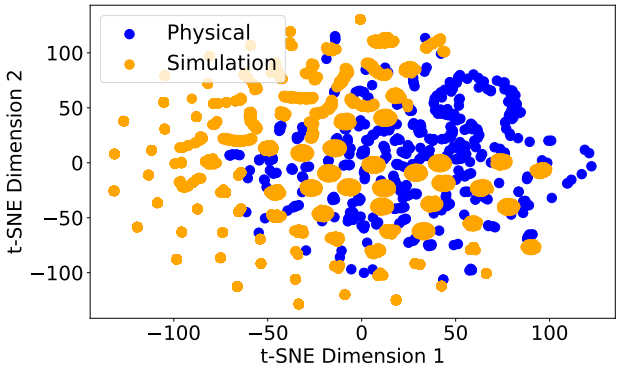
To analyze the simulation-to-reality mismatch, we trained identical neural network models on each simulator's data and evaluated their cross-domain performance on physical testbed measurements. Table I lists the accuracy when simulation-trained models are applied to data collected from physical network. Models achieving 85–92% accuracy within their source simulator domain suffer 25–40% accuracy degradation on target physical data, with NS-3 showing the smallest gap (15.2% loss) and OMNeT++ the largest (31.4% loss).

TABLE I: Prediction accuracy drops when simulation-trained models are applied to data collected from physical network.

Simulator	Source Accuracy	Target Accuracy	Gap
NS-3	89.3 ± 1.2%	74.1 ± 2.3%	15.2%
Cooja	86.7 ± 1.8%	58.9 ± 3.1%	27.8%
TOSSIM	91.2 ± 0.9%	64.8 ± 2.7%	26.4%
OMNeT++	85.4 ± 2.1%	54.0 ± 3.4%	31.4%



(a) Distribution mismatch in performance space



(b) Feature misalignment in latent space

Fig. 1: Quantitative analysis of the simulation-to-reality gap. Physical and simulation data exhibit significant distributional misalignment in both performance metrics (a) and learned feature representations (b).

The simulation-to-reality gap varies significantly across network configurations. Table I lists per-configuration accuracy loss, revealing that certain parameter combinations (high channel count + low transmission attempts) suffer disproportionate degradation. This heterogeneity motivates our configuration-aware synthetic data generation approach

Figure 1(a) shows the systematic bias in simulation models through visualization of performance space. Simulated latency distributions exhibit a 200-500ms range that completely misses the physical data's primary mode at 50-150ms. Similarly, reliability predictions show compression artifacts where simulators predict narrow ranges (0.85-0.95) while physical measurements span broader distributions (0.70-0.98). The latent space analysis in Figure 1(b) reveals deeper structural misalignment. Using t-SNE visualization of learned feature representations, physical and simulation data form distinct clusters with a Silhouette coefficient of 0.73, indicating strong separation. This suggests that simulation-trained models learn fundamentally different feature mappings that poorly transfer to physical domains.

We applied statistical testing to validate the observed domain gaps: Kolmogorov-Smirnov tests reject the null hypothesis that physical and simulation performance distributions are identical ($p < 0.001$ for all simulators and metrics). Jensen-Shannon divergence quantifies the separation: $D_{JS}(\text{Physical}, \text{NS-3}) = 0.34$, indicating substantial distributional differences. Maximum Mean Discrepancy (MMD) tests confirm significant feature distribution differences between domains. Using RBF kernels with varying bandwidths, MMD values range from 0.18 (NS-3) to 0.47 (OMNeT++), all statistically significant with $p < 0.001$.

C. Feasibility Analysis: Synthetic Data as Bridge

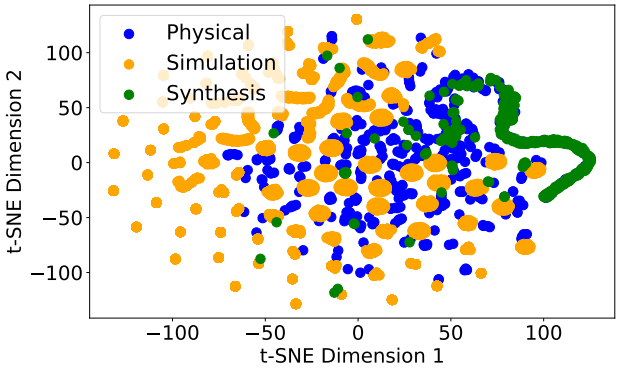


Fig. 2: Synthetic data effectively bridges the simulation-to-reality gap. LLM-generated samples (green) overlap both physical (blue) and simulation (red) distributions, providing intermediate representations that facilitate domain adaptation.

To validate synthetic data generation as a viable solution, we conducted a preliminary study using Gaussian Process regression to generate intermediate samples between physical and simulation domains. Figure 2 shows that carefully generated synthetic data can effectively fill the gap between physical and simulation distributions. Synthetic samples (marked in green) exhibit substantial overlap with both domains, achieving coverage coefficients of 0.82 with physical data and 0.79 with simulation data.

We evaluated synthetic sample quality through statistical consistency metrics:

- *Mean preservation*: Synthetic latency distribution maintains $\mu = 142.7$ms vs. physical $\mu = 138.4$ms (2.4% error);
- *Variance matching*: Synthetic reliability variance $\sigma^2 = 0.041$ vs. physical $\sigma^2 = 0.038$ (7.9% error);
- *Constraint satisfaction*: 96.3% of synthetic samples satisfy physical constraints (positive latency, bounded reliability).

Using 10-fold cross-validation, models trained on augmented datasets (physical + synthetic) achieve 12.3-18.7% accuracy

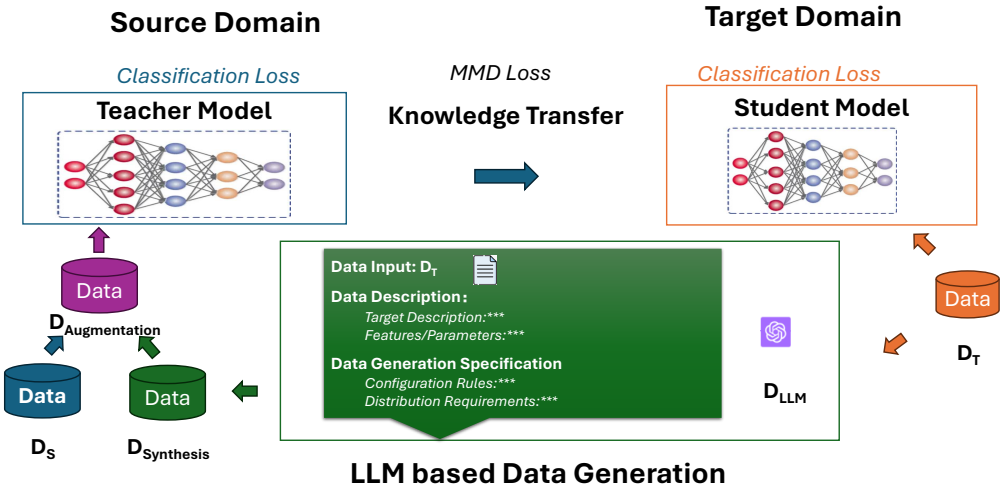


Fig. 3: WMN-LLM-DS system architecture. The framework integrates three core components: (1) constraint-aware LLM data generation, (2) teacher-student knowledge transfer, and (3) adaptive domain alignment for robust wireless network configuration.

improvements over physical-only training across all simulators. This preliminary result strongly supports synthetic data augmentation as an effective bridging strategy.

Our analysis reveals that synthetic data generation becomes increasingly valuable as physical data becomes scarce. With only 5 physical samples per configuration, synthetic augmentation provides 23.4% accuracy improvement. This efficiency is critical for practical deployment, where physical data collection is expensive and time-consuming.

D. Key Empirical Observations

Our systematic analysis yields three key observations that motivate our design of WMN-LLM-DS. The simulation-to-reality mismatch varies by up to 2.1× under different network configurations, indicating the need for adaptive rather than uniform bridging strategies. All observed domain gaps are statistically significant ($p < 0.001$), with effect sizes ranging from medium (Cohen's $d = 0.52$) to large (Cohen's $d = 1.34$). Preliminary experiments further show that high-quality synthetic data can yield substantial performance gains, particularly in low-data regimes common in real-world deployments. These observations demonstrate the critical need to close the simulation-to-reality gap in network configuration and the viability of synthetic data generation as a solution.

III. WMN-LLM-DS System Design

This section presents the design of WMN-LLM-DS, a framework that bridges the simulation-to-reality gap in wireless network configuration through LLM-based synthetic data generation and adaptive domain transfer. We detail the system components, data generation pipeline, and training methodology that enable robust configuration prediction under real-world deployment constraints.

A. Design Overview

WMN-LLM-DS employs a three-stage architecture designed to systematically bridge the simulation-to-reality gap through synthetic data augmentation and domain-aware knowledge transfer. By processing limited physical measurements ($\mathcal{D}_T$) alongside abundant simulation data ($\mathcal{D}_S$), WMN-LLM-DS produces a configuration model that generalizes effectively to real-world deployments. As Figure 3 shows, the architecture is defined by four core functional units: a *LLM-based Data Generator* synthesizes constraint-aware samples conditioned on network metadata and physical data characteristics, a *Teacher Network* extracts comprehensive configuration mappings from the large-scale simulation environment, a *Student Network* adapts this distilled knowledge to the target physical domain, and a *Domain Alignment Module* facilitates this transition by minimizing feature distribution discrepancies through MMD-based loss functions. This system architecture enables scalable processing of diverse network configurations while maintaining computational efficiency suitable for practical deployment scenarios.

B. Constraint-Aware Synthetic Data Generation

Algorithm 1 illustrates a constraint-aware framework for generating high-fidelity synthetic datasets using LLMs, while preserving both statistical properties and domain-specific feasibility constraints. The algorithm takes a target dataset $\mathcal{D}_T$, metadata $\mathcal{M}$ encoding system constraints and schema semantics, and a pretrained LLM f_{LLM} as input, and outputs a validated synthetic dataset $D_{synthetic}$ suitable for downstream modeling and simulation.

The algorithm begins with a statistical profiling phase, in which first- and second-order moments of key performance metrics are extracted from $\mathcal{D}_T$. Specifically, the mean and standard deviation of latency and reliability, denoted by (μ_L, σ_L)

Algorithm 1: Constraint-Aware Synthetic Data Generation

Input : Target data $\mathcal{D}_T$, Metadata $\mathcal{M}$, LLM f_{LLM}
Output: Validated synthetic dataset $D_{synthetic}$

Phase 1: Statistical Analysis
$\mu_L, \sigma_L \leftarrow$ ComputeStatistics($\mathcal{D}_T$.latency)
$\mu_E, \sigma_E \leftarrow$ ComputeStatistics($\mathcal{D}_T$.reliability)
$\rho_{LE} \leftarrow$ CorrelationAnalysis($\mathcal{D}_T$.latency, $\mathcal{D}_T$.reliability)
Phase 2: Prompt Construction
$prompt \leftarrow$ ConstructPrompt($\mathcal{M}$, $\{\mu_L, \sigma_L, \mu_E, \sigma_E, \rho_{LE}\}$)
Phase 3: Iterative Generation
$D_{synthetic} \leftarrow \emptyset$
for $i = 1$ *to* N_{target} **do**
  $sample \leftarrow f_{LLM}(prompt)$
  if *ValidateConstraints(sample, $\mathcal{M}$)* **then**
    $D_{synthetic} \leftarrow D_{synthetic} \cup \{sample\}$
Phase 4: Quality Assurance
$D_{synthetic} \leftarrow$ StatisticalValidation($D_{synthetic}$, $\mathcal{D}_T$)
return $D_{synthetic}$

and (μ_E, σ_E), respectively, are computed, along with their Pearson correlation coefficient ρ_{LE}. These statistics capture both marginal distributions and cross-metric dependencies and serve as soft constraints that anchor the generation process to the empirical data distribution.

Next, the extracted statistics are integrated with metadata $\mathcal{M}$ to construct a structured prompt that conditions the LLM on valid value ranges, inter-feature dependencies, and operational semantics. This prompt transforms the LLM into a constraint-aware conditional generator rather than a free-form sampler, thereby significantly reducing infeasible outputs while preserving diversity.

The algorithm then performs iterative generation, where candidate samples are drawn from $f_{LLM}(prompt)$ and validated against domain constraints encoded in $\mathcal{M}$. These constraints enforce physical feasibility, such as non-negative delays and bounded reliability, as well as system-level invariants and cross-field consistency relationships. Only samples satisfying all constraints are retained in $D_{synthetic}$, and this process continues until the target dataset size is reached, ensuring semantic correctness of all generated records.

Finally, a statistical quality assurance phase evaluates the synthetic dataset against the target dataset $\mathcal{D}_T$ to verify alignment in marginal distributions and correlation structure. Discrepancies beyond predefined tolerances trigger resampling or regeneration, ensuring that the final output preserves both local feasibility and global statistical fidelity. This design enables scalable and safe synthetic data generation for data-scarce scenarios, such as Ultra-Reliable Low-Latency Communications (URLLC) modeling, wireless digital twins, and network control policy training.

C. Teacher-Student Knowledge Transfer Architecture

The teacher-student framework facilitates effective knowledge transfer from simulation-rich environments to data-scarce physical domains through carefully designed neural architectures and loss functions.

1) Teacher Network Design: The teacher network $f_{teacher} : \mathbb{R}^3 \to \mathbb{R}^3$ learns comprehensive configuration mappings from abundant simulation data:

$$f_{teacher}(\mathbf{x}) = \text{MLP}(\mathbf{x}; \theta_T) \tag{1}$$

$$\theta_T^* = \arg\min_{\theta_T} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_S}[\ell(\mathbf{y}, f_{teacher}(\mathbf{x}))] \tag{2}$$

The teacher architecture employs a 4-layer MLP with 256-128-64-32 hidden units, ReLU activations, and dropout regularization (p=0.3) to prevent overfitting to simulation-specific patterns.

2) Student Network Architecture: The student network adapts teacher knowledge to real-world domains using augmented training data. The multi-objective loss function addresses three complementary learning objectives:

Classification Loss: Ensures accurate prediction on augmented target data.

$$\mathcal{L}_{cls} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_{aug}}[\ell(\mathbf{y}, f_{student}(\mathbf{x}))] \tag{3}$$

MMD Domain Alignment Loss: Minimizes feature distribution mismatch between domains.

$$\mathcal{L}_{MMD} = \text{MMD}^2(\phi(f_{teacher}(\mathcal{X}_S)), \phi(f_{student}(\mathcal{X}_T))) \tag{4}$$

Knowledge Consistency Loss: Maintains consistency with teacher predictions where appropriate.

$$\mathcal{L}_{cons} = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{overlap}}[\|\text{softmax}(f_{teacher}(\mathbf{x})) - \text{softmax}(f_{student}(\mathbf{x}))\|_2^2] \tag{5}$$

The total loss combines these objectives:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \lambda_{MMD}\mathcal{L}_{MMD} + \lambda_{cons}\mathcal{L}_{cons} \tag{6}$$

D. System Training Pipeline

Algorithm 2 illustrates the training pipeline that integrates LLM-based synthetic data generation with adaptive knowledge transfer for WMN configuration and performance modeling. The pipeline takes a source dataset $\mathcal{D}_S$, a limited target dataset $\mathcal{D}_T$, metadata $\mathcal{M}$ encoding system constraints and domain semantics, and a pretrained LLM f_{LLM} as input, and outputs an optimized student model $f_{student}^*$ tailored to the target deployment environment.

The procedure begins by training a high-capacity teacher network on the source dataset $\mathcal{D}_S$, yielding optimized parameters θ_T^* that capture transferable structural knowledge from data-rich domains. This teacher model serves as a supervisory signal for subsequent student adaptation and provides soft labels and feature representations that improve generalization under data scarcity.

Algorithm 2: WMN-LLM-DS Training Pipeline

Input: Source data $\mathcal{D}_S$, Target data $\mathcal{D}_T$, LLM f_{LLM}, Metadata $\mathcal{M}$
Output: Optimized student model $f^*_{student}$

Stage 1: Teacher Network Training
$\theta^*_T \leftarrow$ TrainTeacher($\mathcal{D}_S$, epochs=100, lr=1e-3)
Stage 2: Synthetic Data Generation
$D_{syn} \leftarrow$ GenerateSynthetic($\mathcal{D}_T$, $\mathcal{M}$, f_{LLM})
$\mathcal{D}_{aug} \leftarrow \mathcal{D}_T \cup D_{syn}$
Stage 3: Student Network Training
Initialize $f_{student}$ with Xavier initialization
for *epoch = 1 to* E_{max} **do**
  for *batch* $\mathcal{B} \subset \mathcal{D}_{aug}$ **do**
    $\mathcal{L}_{cls} \leftarrow$ ComputeClassificationLoss($\mathcal{B}$, $f_{student}$)
    $\mathcal{L}_{mmd} \leftarrow$ ComputeMMDLoss($f_{teacher}$, $f_{student}$, $\mathcal{B}$)
    $\mathcal{L}_{total} \leftarrow \mathcal{L}_{cls} + \lambda_{MMD}\mathcal{L}_{mmd}$
    $\theta_S \leftarrow \theta_S - \alpha\nabla_{\theta_S}\mathcal{L}_{total}$
  if *ValidationAccuracy*($f_{student}$) *converges* **then**
    break
Stage 4: Model Validation and Selection
$acc_{val} \leftarrow$ EvaluateModel($f_{student}$, $\mathcal{D}_{val}$)
if $acc_{val} >$ *threshold* **then**
  return $f_{student}$
else
  Adjust hyperparameters and repeat Stage 3

Next, the algorithm invokes the constraint-aware synthetic data generator to produce a synthetic dataset D_{syn} conditioned on the limited target data $\mathcal{D}_T$ and metadata $\mathcal{M}$. The synthetic samples are merged with the real target data to form an augmented dataset $\mathcal{D}_{aug} = \mathcal{D}_T \cup D_{syn}$, which expands coverage of rare but operationally important regimes while preserving statistical fidelity and domain feasibility.

The student model $f_{student}$ is then initialized using Xavier initialization and trained on $\mathcal{D}_{aug}$ using a joint optimization objective that combines supervised classification loss $\mathcal{L}_{cls}$ with an MMD regularization term $\mathcal{L}_{mmd}$ that aligns the student's latent representations with those of the teacher network. The total loss $\mathcal{L}_{total} = \mathcal{L}_{cls} + \lambda_{MMD}\mathcal{L}_{mmd}$ is minimized using stochastic gradient descent, and training proceeds iteratively until convergence is detected on a held-out validation set, ensuring both predictive accuracy and distributional alignment across domains.

Finally, the trained student model is evaluated on a validation dataset $\mathcal{D}_{val}$, and if the achieved accuracy exceeds a predefined threshold, the model is selected as the final deployment model $f^*_{student}$. Otherwise, hyperparameters are adjusted, and the student training stage is repeated, enabling robust convergence under heterogeneous domain shifts.

a) Computational Complexity.: The computational cost of synthetic data generation scales as $O(N \cdot L_{prompt} \cdot C_{LLM})$, where N is the number of synthetic samples, L_{prompt} is the prompt length, and C_{LLM} denotes the LLM inference cost per token. The student training stage scales as $O(E \cdot B \cdot (|D_{aug}| \cdot C_{forward} + C_{MMD}))$, where E is the number of epochs, B is the batch size, $C_{forward}$ is the forward/backward propagation cost, and C_{MMD} is the cost of computing the MMD regularizer. The memory footprint scales as $O(|\theta_T| + |\theta_S| + |D_{aug}|)$ to store teacher parameters, student parameters, and the augmented dataset. Overall, the pipeline exhibits linear scaling in dataset size while delivering substantially improved domain adaptation performance under limited target supervision.

The system design addresses the key challenges of simulation-to-reality transfer through principled synthetic data generation, adaptive knowledge transfer, and efficient implementation optimizations suitable for practical wireless network deployment.

IV. Evaluation

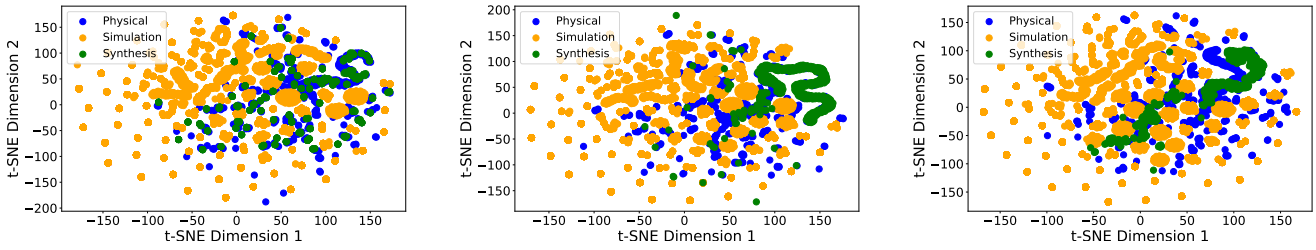
In this section, we first introduce our experimental setup and the dataset used for our experiments. We then present the performance of WMN-LLM-DS and our study on the effects of two key parameters on WMN-LLM-DS's performance. Finally, we present the performance of WMN-LLM-DS when different amounts of physical data are used to train the network configuration model.

A. Experimental Setup and Dataset

We implement WMN-LLM-DS and our baseline TS-DA, the domain adaptation method developed by Shi el al. [8], using PyTorch [13]. We train WMN-LLM-DS with the Adam optimizer [14] and run all experiments on a single NVIDIA A100 GPU with $80GB$ of memory. The physical data used in our experiment is collected from the WirelessHART network that runs on the testbed built by Shi et al. [8]. The testbed consists of 50 TelosB motes [15]. The data contains 88 distinct network configurations with three network parameters: $R \in \{0.7, 0.71, 0.72, \ldots, 0.9\}$, $C \in \{1, 2, 3, \ldots, 8\}$, and $A \in \{1, 2, 3\}$. Each network configuration corresponds to three performance metrics: end-to-end latency (L), battery lifetime (B), and end-to-end reliability (E). In total, we have 6,600 data traces (88 network configurations × 75 traces per configuration). We implement the same WirelessHART network in four wireless simulators, including NS-3 [16], Cooja [17], TOSSIM [18], and OMNet++ [19]. In total, we collect 6,600 data traces (88 network configurations × 75 traces per configuration) from each simulator.

B. Performance of WMN-LLM-DS

We first measure the performance of WMN-LLM-DS in selecting configurations to achieve optimal network performance. The experiments are conducted using 75 shots (each shot contains one data sample under each of 88 network configurations) of simulation data and 5 shots of physical data. To ensure robustness, the experiments are repeated with simulation data collected from various simulators. Table II lists the prediction performance of the proposed WMN-LLM-DS model



(a) Gemini. (b) GPT-4o. (c) Grok2.

Fig. 4: T-SNE Visualization of Physical, Simulation, and Synthetic Data Across Different LLMs

compared to two baseline models: TS-DA (Teacher-Student Model) and WMN-GP-DS (Gaussian Process Data Synthesis). This evaluation is carried out across four network simulators: NS-3, Cooja, TOSSIM, and OMNet++. The primary objective is to assess the effectiveness of WMN-LLM-DS in enhancing prediction accuracy across diverse simulation environments.

Among the models, TS-DA demonstrates the lowest performance, with accuracy ranging from 60.9% to 71.1%. WMN-GP-DS shows moderate improvements over TS-DA, achieving higher scores, such as 73.5% in TOSSIM and 66.6% in OMNet++. In contrast, WMN-LLM-DS consistently outperforms both baselines, achieving top scores of 77.4% (NS-3), 78.0% (Cooja), 77.3% (TOSSIM), and 77.4% (OMNet++). The improvement row highlights the percentage gains of WMN-LLM-DS over WMN-GP-DS, with significant enhancements such as 10.8% in OMNet++ and 7.9% in Cooja. This analysis underscores the robust predictive capabilities of WMN-LLM-DS, particularly in challenging simulation environments.

C. Effects of Prompts

We then examine the effects of range specification (RS) prompts (e.g., $A \in [1,3]$, $C \in [1,8]$) on the prediction performance of the model across four network simulators: NS-3, Cooja, TOSSIM, and OMNet++. The primary objective is to assess how RS prompts enhance prediction accuracy. The analysis focuses on two key parameters: the number of channels used in the network (C), representing the total communication channels allocated for network operations, and the number of transmission attempts per packet (A), which defines the maximum number of retransmissions allowed before a packet is considered lost. The results in Table III are divided into two scenarios: predictions made without RS prompts "without RS", and predictions made with RS prompts "RS". Without RS prompts, the model achieves prediction accuracies of 75.0% (NS-3), 70.9% (Cooja), 75.7% (TOSSIM), and 65.4% (OMNet++). When RS prompts are incorporated, the model exhibits improved performance across all simulators, achieving scores of 77.4% (NS-3), 78.0% (Cooja), 77.3% (TOSSIM), and 77.4% (OMNet++). These findings underscore the importance of RS prompts in enhancing prediction accuracy. Notably, significant improvements are observed in simulators such as Cooja and OMNet++, highlighting the value of incorporating RS prompts in the model's predictive processes. This demonstrates that RS prompts play a critical role in refining the model's predictive capabilities across diverse simulation environments.

TABLE II: Prediction performance when using different simulators with various training methods. "Imp." represents the improvement of WMN-LLM-DS compared to the best performance of our baseline "TS-DA" (Teacher-student Model) and "WMN-GP-DS" (Gaussian Process data synthesis method).

Simulator	NS-3	Cooja	TOSSIM	OMNet++
TS-DA	70.1%	69.8%	71.1%	60.9%
WMN-GP-DS	71.3%	70.1%	73.5%	66.6%
WMN-LLM-DS	**77.4%**	**78.0%**	**77.3%**	**77.4%**
Imp.	+6.1%	+7.9%	+3.8%	+10.8%

D. Performance When Using Different LLMs

We also evaluate the prediction performance of different LLMs across four network simulators: NS-3, Cooja, TOSSIM, and OMNet++. Figure 4 presents t-SNE visualizations of data distributions for three LLMs: Gemini, GPT-4o, and Grok2. For Gemini, synthetic data successfully bridges the gaps between physical and simulation data, overlapping both distributions and reducing the simulation-to-reality gap. GPT-4o demonstrates a more distinct clustering of synthetic data, showcasing its ability to capture domain-specific nuances and enhance alignment between physical and simulation data. Similarly, Grok2 effectively generates contextually rich and diverse synthetic datasets, further bridging the simulation-to-reality gap.

Figure 5 summarizes the performance results. GPT-4 demonstrates strong predictive capabilities, particularly in NS-3 (77.8%) and Cooja (77.2%), though it shows slightly lower performance in OMNet++ (69.5%). GPT-4o consistently achieves

TABLE III: Prediction performance when with or without range specification prompts.

Distribution	NS-3	Cooja	TOSSIM	OMNet++
without RS	75.0%	70.9%	75.7%	65.4%
RS	**77.4%**	**78.0%**	**77.3%**	**77.4%**

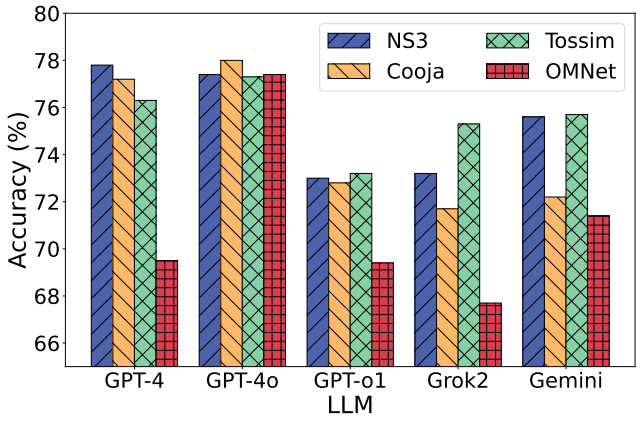


Fig. 5: Prediction performance when using different LLMs.

high accuracy, outperforming other models in Cooja (78.0%) and excelling in OMNet++ (77.4%). GPT-o1 delivers moderate performance, with its best results in NS-3 (73.0%) and TOSSIM (73.2%). Grok2 achieves its highest performance in TOSSIM (73.3%) but remains below 73% across other simulators. Gemini provides balanced performance, with its best results in OMNet++ (71.4%) and TOSSIM (75.7%). Overall, GPT-4o emerges as the most effective model, consistently achieving the highest or near-highest performance across all simulators, particularly in challenging environments such as Cooja and OMNet++. This analysis underscores the importance of selecting the appropriate LLM to optimize predictive accuracy for simulation-based tasks.

E. Performance with Different Amounts of Synthesis Data

We further evaluate the prediction performance of the model under different ratios of synthetic to physical data across four network simulators: NS-3, Cooja, TOSSIM, and OMNet++. The ratio represents the proportion of synthetic data to physical data used during training. The goal of this evaluation is to determine the optimal data composition ratio that maximizes prediction accuracy. The results in Table IV indicate variations in performance based on the ratio used. For most simulators, ratios closer to 4:5, 5:5, and 10:5 yield the highest prediction accuracy. Conversely, lower ratios such as 1:5 and higher ratios such as 9:5 result in reduced accuracy across all simulators. These findings highlight the importance of balancing synthetic and physical data to achieve optimal prediction performance. Ratios like 4:5, 5:5, and 10:5 demonstrate the ability to effectively leverage synthetic data while maintaining a strong correlation with physical data.

F. Performance with Different Amounts of Physical Data

Finally, we evaluate the model's prediction performance as we vary the number of physical data shots[1] across four network simulators: NS-3, Cooja, TOSSIM, and OMNet++. Figure 6 shows the corresponding results.

[1]The term "shots" refers to the number of physical data samples used during training.

TABLE IV: Performance When Using Different Ratios of Synthesis Data. The ratio represents the synthesis of physical data relationship.

Ratio	NS-3	Cooja	TOSSIM	OMNet++
1 : 5	74.7%	74.2%	76.5%	70.7%
2 : 5	75.8%	73.0%	76.9%	69.7%
3 : 5	76.6%	72.6%	75.7%	76.9%
4 : 5	**77.4%**	74.0%	76.6%	75.0%
5 : 5	77.2%	**78.0%**	75.6%	**77.4%**
6 : 5	71.1%	68.6%	75.3%	67.3%
7 : 5	72.2%	69.3%	72.1%	64.1%
8 : 5	70.9%	69.5%	71.3%	57.9%
9 : 5	72.0%	69.6%	69.5%	68.7%
10 : 5	77.3%	75.8%	**77.3%**	75.6%

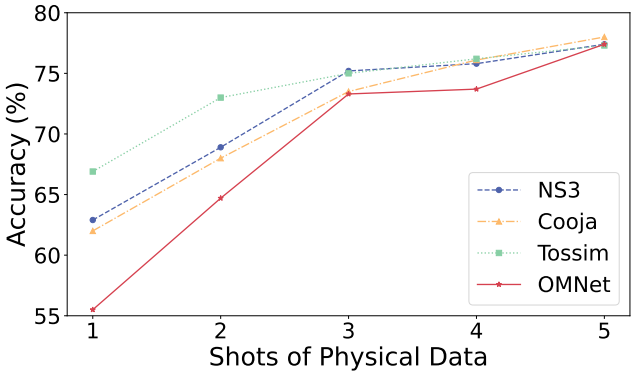


Fig. 6: Prediction performance across different simulators with different shots of physical data.

Overall, we observe a consistent improvement in prediction accuracy as the number of physical data shots increases across all simulators. With only one shot, performance remains relatively low, ranging from 55.5% on OMNet++ to 66.9% on TOSSIM, indicating limited generalization when physical data supervision is scarce. As the number of shots increases to five, prediction accuracy improves substantially, reaching 77.4% on both NS-3 and OMNet++, 78.0% on Cooja, and 77.3% on TOSSIM.

These results highlight the strong positive impact of incorporating additional physical data during training, particularly in more challenging simulation environments. The observed trend suggests that even a modest increase in physical data availability can significantly enhance model robustness and predictive reliability, underscoring the effectiveness of combining simulated and physical data sources.

V. Related Work

A. Wireless Mesh Networks

WMNs have become a fundamental communication substrate for many IoT applications due to their flexibility, scal-

ability, and robustness against device failures. Standards such as WirelessHART enable reliable low-power wireless communication in harsh industrial environments, but sustaining high performance remains challenging due to interference, multipath fading, and time-varying traffic patterns [20], [21]. Traditional WMN configuration approaches often rely on static heuristics such as channel blacklisting or routing based on fixed PRR thresholds [22]. However, recent studies show that such heuristic policies frequently fail to generalize across heterogeneous environments and dynamic workloads [23], [24].

To address these limitations, optimization-based and adaptive control approaches have been proposed for routing, scheduling, and channel assignment [25], [26]. While effective in structured scenarios, these methods struggle to scale to large networks and to capture complex cross-layer interactions in modern industrial deployments. More recently, machine learning techniques, particularly reinforcement learning and deep learning, have demonstrated strong potential for learning adaptive network control policies directly from data [27], [28]. However, their real-world applicability remains constrained by the difficulty and cost of collecting sufficiently large and diverse industrial datasets. Although simulators offer a scalable alternative, discrepancies between simulated and real environments continue to limit model transferability and robustness [8], [29].

B. Large Language Models

LLMs such as BERT [30], GPT-style autoregressive transformers [31], and more recent foundation models [32], [33] have demonstrated remarkable capabilities in representation learning, reasoning, and structured generation. Beyond natural language processing, LLMs are increasingly explored as general-purpose foundation models for scientific computing, robotics, and systems optimization [32]. Early work suggests that LLMs can assist in configuration synthesis, program generation, and system design automation.

In networking and wireless systems, the potential of LLMs lies in their ability to generate structured data and encode complex dependencies across configuration parameters and performance metrics. However, applying LLMs to networked systems introduces unique challenges, as generated samples must satisfy physical-layer constraints, protocol invariants, and operational semantics. General-purpose LLMs lack built-in inductive biases for such domains, making constraint modeling, prompt conditioning, and post-generation validation essential to ensure feasibility and realism [34], [35], [36], [37], [38], [39]. These challenges motivate the development of constraint-aware LLM pipelines tailored to networking applications.

C. Data Synthesis

Data synthesis has long been used to mitigate data scarcity in machine learning and system modeling. Classical approaches include statistical modeling and resampling methods [40], while modern techniques increasingly rely on generative models such as variational autoencoders and Generative Adversarial Networks (GANs) [41], [42]. In wireless networks, synthetic data has been employed to model channel dynamics, traffic patterns, and mobility behavior [43], [44], [45], [46].

Despite their success, existing synthesis methods face persistent challenges in capturing high-dimensional dependencies, rare operational regimes, and complex system constraints. Models trained on synthetic data often exhibit degraded performance when transferred to real deployments due to distribution shift and unmodeled interactions [47], [29]. Recent work has explored hybrid approaches that combine simulation, generative modeling, and domain adaptation to improve realism and transferability [48]. More recently, LLMs have emerged as flexible generative engines capable of producing structured and semantically rich samples [31], [49], [50], but without explicit constraint modeling and validation, LLM-generated data may violate domain invariants and physical feasibility. This motivates the need for principled, constraint-aware synthesis frameworks, particularly for safety-critical wireless networking applications.

VI. Conclusions

This paper presents WMN-LLM-DS, a novel framework that leverages LLMs for synthetic data generation to bridge the simulation-to-reality gap in Wireless Mesh Network configurations. By integrating LLM-driven data synthesis with domain adaptation techniques, WMN-LLM-DS enhances the diversity, representativeness, and quality of training data, enabling robust generalization across diverse network environments. Experiments show that WMN-LLM-DS outperforms existing methods in prediction accuracy and domain generalization across multiple simulators and physical testbeds. These findings highlight the potential of LLMs to revolutionize WMN configurations, offering a scalable and efficient solution to meet the demands of modern networked systems, and paving the way for future advancements in cyber-physical systems and intelligent wireless networks.

Acknowledgment

This work was supported in part by the National Science Foundation under grants CNS-2150010, ECCS-2242700, and IIS-2529283.

References

[1] B. Milic, S. Brack, and R. Naumann, "Hierarchical Configuration System for Wireless Mesh Networks in Manufacturing Industry," in *IFIP Wireless and Mobile Networking Conference (WMNC)*, 2014.

[2] H.-C. Lee and H.-H. Lin, "Design and Evaluation of An Open-source Wireless Mesh Networking Module for Environmental Monitoring," *IEEE Sensors Journal*, vol. 16, no. 7, pp. 2162–2171, 2015.

[3] A. Ma, J. T. Rodriguez, M. Sha, and D. Luo, "Sensorless Air Temperature Sensing Using LoRa Link Characteristics," in *IEEE International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2025.

[4] A. Ma, J. C. T. Rodriguez, and M. Sha, "Enabling Reliable Environmental Sensing with LoRa, Energy Harvesting, and Domain Adaptation," in *IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2024.

[5] C. Silva, Y. Oliveira, C. Celes, R. Braga, and C. Oliveira, "Performance Evaluation of Wireless Mesh Networks in Smart Cities Scenarios," in *Euro American Conference on Telematics and Information Systems (EATIS)*, 2018.

[6] I. F. Akyildiz, X. Wang, and W. Wang, "A Survey on Wireless Mesh Networks," *IEEE Communications Magazine*, vol. 43, no. 9, pp. S23–S30, 2005.

[7] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems," *Proceedings of the IEEE, Special Issue on Industrial Cyber Physical Systems*, vol. 104, no. 5, pp. 1013–1024, 2016.

[8] J. Shi, M. Sha, and X. Peng, "Adapting Wireless Mesh Network Configuration from Simulation to Reality via Deep Learning based Domain Adaptation," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2021.

[9] X. Cheng, M. Sha, and D. Chen, "Configuring Industrial Wireless Mesh Networks via Multi-Source Domain Adaptation," in *ACM International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2024.

[10] J. Shi, A. Ma, X. Cheng, M. Sha, and X. Peng, "Adapting Wireless Network Configuration from Simulation to Reality via Deep Learning based Domain Adaptation," *IEEE/ACM Transactions on Networking*, vol. 32, no. 3, pp. 1983–1998, 2024.

[11] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of Representations for Domain Adaptation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2006.

[12] M. et al., "Network Configuration Data Traces," 2026. [Online]. Available: https://github.com/aitianma/WSNConfDomainAdaptation/

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[14] D. Kinga, J. B. Adam *et al.*, "A Method for Stochastic Optimization," in *International conference on learning representations (ICLR)*, 2015.

[15] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling Ultra-low Power Wireless Research," in *ACM/IEEE International Symposium on Information Processing in Sensor Networks (IPSN)*, 2005.

[16] "NS-3 Shadowing Model." [Online]. Available: https://www.nsnam.org/docs/release/3.10/doxygen/classns3_1_1_shadowing_loss_model.html

[17] "Source Code of Cooja," https://github.com/contiki-os/contiki/wiki/An-Introduction-to-Cooja, Cooja.

[18] "Source Code of TOSSIM," https://github.com/tinyos/tinyos-main/tree/master/tos/lib/tossim, TOSSIM.

[19] "OMNeT," https://github.com/omnetpp/omnetpp, OMNeT.

[20] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "Reliable and Real-Time Communication in Industrial Wireless Mesh Networks," *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2011.

[21] A. Willig, "Recent and Emerging Topics in Wireless Industrial Communications," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 2, pp. 102–124, 2008.

[22] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep Reinforcement Learning for Dynamic Multichannel Access in Wireless Networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 257–265, 2018.

[23] J. Shi and M. Sha, "Parameter Self-Configuration and Self-Adaptation in Industrial Wireless Sensor-Actuator Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2019.

[24] ——, "Parameter Self-Adaptation for Industrial Wireless Sensor-Actuator Networks," *ACM Transactions on Internet Technology*, 2020.

[25] E. Altman, T. Boulogne, R. El-Azouzi, T. Jiménez, and L. Wynter, "A Survey on Networking Games in Telecommunications," *Computers & Operations Research*, vol. 33, no. 2, pp. 286–311, 2006.

[26] Y. Li, C. Chen, and Q. Sun, "Joint Routing and Scheduling in Wireless Mesh Networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 12, pp. 5304–5314, 2008.

[27] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Reinforcement Learning for Context Awareness and Intelligence in Wireless Networks: Review, New Features and Open Issues," *Journal of Network and Computer Applications*, vol. 35, no. 1, pp. 253–267, 2012.

[28] L. Huang, S. Bi, and Y.-J. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-edge Computing Networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2581–2593, 2020.

[29] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[30] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019.

[31] T. Brown, B. Mann, N. Ryder, and et al., "Language models are few-shot learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[32] R. Bommasani, D. A. Hudson, E. Adeli, and et al., "On the Opportunities and Risks of Foundation Models," *arXiv preprint arXiv:2108.07258*, 2021.

[33] H. Touvron, T. Lavril, G. Izacard, and et al., "LLaMA: Open and Efficient Foundation Language Models," *arXiv preprint arXiv:2302.13971*, 2023.

[34] A. Fan, B. Gokkaya, M. Harman, M. Lyubarskiy, S. Sengupta, S. Yoo, and J. M. Zhang, "Large Language Models for Software Engineering: Survey and Open Problems," in *IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*, 2023.

[35] A. Havrilla, Y. Du, S. C. Raparthy, C. Nalmpantis, J. Dwivedi-Yu, M. Zhuravinskyi, E. Hambro, S. Sukhbaatar, and R. Raileanu, "Teaching large language models to reason with reinforcement learning," *arXiv preprint arXiv:2403.04642*, 2024.

[36] D. Ma, A. Ma, S. Luo, M. de Jode, A. Hudson-Smith, and M. Sha, "LORADOCTOR: LLM-Driven Diagnosis and Adaptive Policy Optimization for Reducing Packet Error Rate in LoRaWAN Networks," in *IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2025.

[37] A. Ma, D. Luo, A. Maatouk, R. Ying, and M. Sha, "Configuring Industrial Wireless Mesh Network via Dual-Mind Reasoning," in *IEEE International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2026.

[38] A. Ma and M. Sha, "WMN-CDA: Contrastive Domain Adaptation for Wireless Mesh Network Configuration," in *ACM Symposium On Applied Computing (SAC)*, 2025.

[39] J. T. Rodriguez, D. Sadekeen, M. Sha, and M. A. Rahman, "Leveraging LLMs to Close Simulation-to-Reality Gap in Wireless Mesh Network Configuration," in *IEEE International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2026.

[40] D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys*. Wiley, 1987.

[41] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[42] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[43] Y. Zhao, C.-X. Wang, and et al., "Deep Learning-Based Channel Modeling and Prediction," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 110–116, 2019.

[44] J. Wang, W. Zhang, and Q. Zhang, "Mobility Prediction in Wireless Networks: A Deep Learning Perspective," *IEEE Network*, vol. 35, no. 1, pp. 72–78, 2021.

[45] A. Ma, D. Luo, and M. Sha, "MixLinear: Extreme Low Resource Multivariate Time Series Forecasting with 0.1 K Parameters," in *International Conference on Learning Representations (ICLR)*, 2026.

[46] ——, "MMFNet: Multi-Scale Frequency Masking Neural Network for Multivariate Time Series Forecasting," in *ACM Symposium on Applied Computing (SAC)*, 2026.

[47] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from Simulated and Unsupervised Images through Adversarial Training," in *IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*, 2017.

[48] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[49] X. Zheng, T. Wang, W. Cheng, A. Ma, H. Chen, M. Sha, and D. Luo, "Parametric Augmentation for Time Series Contrastive Learning," in *International Conference on Learning Representations (ICLR)*, 2024.

[50] ——, "AutoTCL: Automated Time Series Contrastive Learning with Adaptive Augmentations," in *Second Workshop of Artificial Intelligence for Time Series Analysis: Theory, Algorithms, and Applications (AI4TS)*, 2023.