Adapting Wireless Mesh Network Configuration from Simulation to Reality via Deep Learning based Domain Adaptation

Junyang Shi*, Mo Sha*, and Xi Peng+

*Department of Computer Science, State University of New York at Binghamton
+Department of Computer & Information Sciences, University of Delaware
Wireless Mesh Networks (WMNs)

- Rapid deployments in recent years
  - For industrial automation, military operations, smart energy, etc.
Wireless Mesh Networks (WMNs)

- Rapid deployments in recent years
  - For industrial automation, military operations, smart energy, etc.
- Industrial wireless sensor-actuator networks (WSANs)
  - Connect sensors, actuators, and controllers in industrial facilities, such as steel mills, oil refineries, and chemical plants
Wireless Mesh Networks (WMNs)

- Rapid deployments in recent years
  - For industrial automation, military operations, smart energy, etc.
- Industrial wireless sensor-actuator networks (WSANs)
  - Connect sensors, actuators, and controllers in industrial facilities, such as steel mills, oil refineries, and chemical plants
Wireless Mesh Networks (WMNs)
- Rapid deployments in recent years
  - For industrial automation, military operations, smart energy, etc.
- Industrial wireless sensor-actuator networks (WSANs)
  - Connect sensors, actuators, and controllers in industrial facilities, such as steel mills, oil refineries, and chemical plants
  - Standards: *WirelessHART*, ISA100, 6TiSCH, etc.

Credit: Emerson Process Management
Credit: FieldComm Group
Network configuration: a complex process
- Involving theoretical computation, simulation, and field testing, among other tasks
Network configuration: a complex process
  - Involving theoretical computation, simulation, and field testing, among other tasks

Using simulations to identify good network configurations
  - Simulations can be set up in less time, introduce less overhead, and allow for different configurations to be tested under exactly the same conditions
WMN Configuration

- Network configuration: a complex process
  - Involving theoretical computation, simulation, and field testing, among other tasks

- Using simulations to identify good network configurations
  - Simulations can be set up in less time, introduce less overhead, and allow for different configurations to be tested under exactly the same conditions
  - Wireless simulators: TOSSIM, Cooja, OMNet++, NS-3, etc.
Network configuration: a complex process
- Involving theoretical computation, simulation, and field testing, among other tasks

Using simulations to identify good network configurations
- Simulations can be set up in less time, introduce less overhead, and allow for different configurations to be tested under exactly the same conditions
- Wireless simulators: TOSSIM, Cooja, OMNet++, NS-3, etc.
- Challenge: hard to capture extensive uncertainties, variations, and dynamics in real-world deployments
- Issue: questionable credibility of the simulation results
Empirical Study

- Experimental setup and data collection
  - Adopt an open-source implementation of WirelessHART networks provided by Li et al. at Washington University in St. Louis
  - Configure six data flow on our testbed with 50 TelosB motes
Empirical Study

- Experimental setup and data collection
  - Adopt an open-source implementation of WirelessHART networks provided by Li et al. at Washington University in St. Louis
  - Configure six data flow on our testbed with 50 TelosB motes
  - Consider three configurable parameters: 88 distinct configurations
    - \( R \): the PRR threshold for link selection
    - \( C \): the number of channels used in the network
    - \( A \): the number of transmission attempts scheduled for each packet
  - Consider three network performance metrics:
    - \( L \): the end-to-end latency
    - \( B \): the battery lifetime
    - \( E \): the end-to-end reliability
Empirical Study

- Experimental setup and data collection
  - Adopt an open-source implementation of WirelessHART networks provided by Li et al. at Washington University in St. Louis
  - Configure six data flow on our testbed with 50 TelosB motes
  - Consider three configurable parameters: 88 distinct configurations
    - R: the PRR threshold for link selection
    - C: the number of channels used in the network
    - A: the number of transmission attempts scheduled for each packet
  - Consider three network performance metrics:
    - L: the end-to-end latency
    - B: the battery lifetime
    - E: the end-to-end reliability
  - Simulation data $D_s$: 6,600 traces; Physical data $D_p$: 6,600 traces
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping $f_\theta(\cdot): x \rightarrow y$
    - $x = \text{concatenation}(L,B,E)$: the given performance requirements
    - $y = \text{concatenation}(R,C,A)$: the network configuration
    - $\theta$: the model parameters that are learned from data
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping \( f_\theta(\cdot) : x \rightarrow y \)
    - \( x = \text{concatenation}(L,B,E) \): the given performance requirements
    - \( y = \text{concatenation}(R,C,A) \): the network configuration
    - \( \theta \): the model parameters that are learned from data

![Accuracy Comparison](image)
Empirical Study

Problem formulation

- Formulate our network configuration prediction task as a machine learning problem
- Our goal: to learn a nonlinear mapping $f_\theta(\cdot) : x \rightarrow y$
  - $x = \text{concatenation}(L,B,E)$: the given performance requirements
  - $y = \text{concatenation}(R,C,A)$: the network configuration
  - $\theta$: the model parameters that are learned from data
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping $f_\theta(\cdot): x \rightarrow y$
    - $x = \text{concatenation}(L,B,E)$: the given performance requirements
    - $y = \text{concatenation}(R,C,A)$: the network configuration
    - $\theta$: the model parameters that are learned from data
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping $f_{\theta}(\cdot): x \rightarrow y$
    - $x = \text{concatenation}(L, B, E)$: the given performance requirements
    - $y = \text{concatenation}(R, C, A)$: the network configuration
    - $\theta$: the model parameters that are learned from data

![Accuracy Comparison]

<table>
<thead>
<tr>
<th>Model</th>
<th>Train: $D^5$, Test: $D^5$</th>
<th>Train: $D^5$, Test: $D^p$</th>
<th>Train: $D^p$, Test: $D^p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>88.92%</td>
<td>79.83%</td>
<td>69.12%</td>
</tr>
<tr>
<td>SVM</td>
<td>25.70%</td>
<td></td>
<td>52.90%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping \( f_\theta(\cdot) : x \rightarrow y \)
    \( x = \text{concatenation}(L,B,E) \): the given performance requirements
    \( y = \text{concatenation}(R,C,A) \): the network configuration
    \( \theta \): the model parameters that are learned from data

<table>
<thead>
<tr>
<th># of Data Samples Used for Training</th>
<th>From a Physical Network (Train: ( D^p ), Test: ( D^p ))</th>
<th>From Simulation (Train: ( D^s ), Test: ( D^p ))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy (%)</td>
<td>Collection Time (s)</td>
</tr>
<tr>
<td>88</td>
<td>19.39</td>
<td>4.40 \times 10^3</td>
</tr>
<tr>
<td>528</td>
<td>42.16</td>
<td>2.64 \times 10^4</td>
</tr>
<tr>
<td>968</td>
<td>57.92</td>
<td>4.84 \times 10^4</td>
</tr>
<tr>
<td>2,024</td>
<td>67.68</td>
<td>1.01 \times 10^5</td>
</tr>
<tr>
<td>3,080</td>
<td>78.82</td>
<td>1.54 \times 10^5</td>
</tr>
<tr>
<td>3,960</td>
<td>79.83</td>
<td>1.98 \times 10^5</td>
</tr>
</tbody>
</table>
Empirical Study

- Problem formulation
  - Formulate our network configuration prediction task as a machine learning problem
  - Our goal: to learn a nonlinear mapping $f_\theta(\cdot): x \rightarrow y$
    - $x = \text{concatenation}(L,B,E)$: the given performance requirements
    - $y = \text{concatenation}(R,C,A)$: the network configuration
    - $\theta$: the model parameters that are learned from data

<table>
<thead>
<tr>
<th># of Data Samples Used for Training</th>
<th>From a Physical Network (Train: $D^p$, Test: $D^p$)</th>
<th>From Simulation (Train: $D^s$, Test: $D^p$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy (%)</td>
<td>Collection Time (s)</td>
</tr>
<tr>
<td>88</td>
<td>19.39</td>
<td>$4.40 \times 10^3$</td>
</tr>
<tr>
<td>528</td>
<td>42.16</td>
<td>$2.64 \times 10^4$</td>
</tr>
<tr>
<td>968</td>
<td>57.92</td>
<td>$4.84 \times 10^4$</td>
</tr>
<tr>
<td>2,024</td>
<td>67.68</td>
<td>$1.01 \times 10^5$</td>
</tr>
<tr>
<td>3,080</td>
<td>78.82</td>
<td>$1.54 \times 10^5$</td>
</tr>
<tr>
<td>5,960</td>
<td>79.83</td>
<td>$1.98 \times 10^5$</td>
</tr>
</tbody>
</table>
Domain Adaptation

- Close the gap by domain adaptation
  - Idea: to construct a deep learning model that can learn transferable features that bridge the cross-domain discrepancy and build a classifier \( y = f_{\theta}(x) \), which can maximize the target domain accuracy (\( f_s \rightarrow f_p \)) by using a small amount of physical data.
Domain Adaptation

- **Teacher Neural Network**
  - Taking advantage of the large amount of simulation data for training
  - Learning its parameters by minimizing the cross-entropy loss

- **Student Neural Network**
  - Trained based on the physical data with the help of the teacher
  - Classification loss: \( L_{cls} = - \mathbb{E}_{x \sim D_p} y \log(f_{\theta_2}(x)) \)
  - Distillation: \( L_{dis} = - \mathbb{E}_{x \sim D_s} q \log(f_{\theta_2}(x)) \)
  - Domain-consistent loss: \( L_{mmd} = \| \mathbb{E}_{x \sim D_s} f_{\theta_1}(x) - \mathbb{E}_{x \sim D_p} f_{\theta_2}(x) \| \)
Evaluation

- Using our testbed and four simulators: TOSSIM, Cooja, OMNeT++, and NS-3
- Compare against seven baselines
  - Seven baselines: TPTP, TSTP, FT, CCSA, DaNN, RSM, and Kriging
Testing accuracy and energy consumption
Evaluation

- Testing accuracy and energy consumption

![Chart showing accuracy and energy consumption over number of shots]

- Accuracy: 50.12%
- Energy consumption: 70.24%
Testing accuracy and energy consumption

Accuracy vs. Number of Shots

- TPTP
- TSTP
- FT
- CCSA
- DaNN
- Ours
- RSM
- Kriging

Accuracies:
- 41.21%
- 30.1%
Our Contributions

- We present the simulation-to-reality gap in network configurations
- We formulate the network configuration into a machine learning problem and develop a teacher-student neural network to close the gap
- We implement and evaluate our method through testbed experimentation: our method effectively closes the gap and increases the accuracy of predicting a good network configuration from 30.10% to 70.24%
Thanks for your attention! Questions?