# Applying Machine Learning Methods to Improve All-Terminal Network Reliability

José Azucena, University of Arkansas
Farid Hashemian, University of Arkansas
Haitao Liao, Ph.D., University of Arkansas
Edward Pohl, Ph.D., University of Arkansas

Key Words: reliability models, reliability growth analysis, deep reinforcement learning

*SUMMARY & CONCLUSIONS*

One essential task in practice is to quantify and improve the reliability of an infrastructure network in terms of the connectivity of network components (i.e., all-terminal reliability). However, as the number of edges and nodes in the network increases, computing the all-terminal network reliability using exact algorithms becomes prohibitive. This is extremely burdensome in network designs requiring repeated computations. In this paper, we propose a novel machine learning-based framework for evaluating and improving all-terminal network reliability using Deep Neural Networks (DNNs) and Deep Reinforcement Learning (DRL). With the help of DNNs and Stochastic Variational Inference (SVI), we can effectively compute the all-terminal reliability for different network configurations in DRL. Furthermore, the Bayesian nature of the proposed SVI+DNN model allows for quantifying the estimation uncertainty while enforcing regularization and reducing overfitting. Our numerical experiment and case study show that the proposed framework provides an effective tool for infrastructure network reliability improvement.

## 1 INTRODUCTION

Interconnected infrastructures provide critical services for the general population. Such critical infrastructures can take the form of power networks, communication networks, water networks, and transportation networks. Like other physical entities, these networks are prone to failure due to natural degradation and adverse events. An essential task in practice is to evaluate the reliability of an infrastructure network in terms of the connectivity of its components. Focusing on the probability that all components remain communicated with each other, the requirement can be conceptualized as the all-terminal reliability of the network [1].

Consider an $n$-node network $(V, E)$ with edge topology $X = [x_{1,2}, \dots, x_{i,j}, \dots, x_{n-1,n}]$ with $x_{i,j} = 1$, if edge $e_{i,j}$ is present; 0, otherwise. Let $p(x_{i,j})$ be the reliability of edge $(i,j)$, and $\Omega$ be all operational states with all nodes being connected. Then, the general equation for computing the all-terminal network reliability is [2]:

$$R = \sum_{X' \in \Omega} \left[ \prod_{(i,j) \in X'} p(x_{i,j}) \right] \left[ \prod_{(k,l) \in (X \setminus X')} (1 - p(x_{k,l})) \right] \quad (1)$$

The computational effort of exact algorithms for computing the all-terminal reliability of a network scales exponentially as the number of edges and nodes increases and eventually becomes prohibitive [3]. As a result, this computational step can become burdensome in applications that require repeated computations of all-terminal reliability for different network configurations. As an alternative to exact methods, approximate methods have been developed with varying degrees of precision. The gamut of such methods includes simulation-based methods such as Monte Carlo simulation [4], surrogate models like Artificial Neural Networks (ANNs) [2], and highly flexible models such as Deep Neural Networks (DNNs) supported by Graph Embeddings [2].

Beyond network reliability evaluation, improving the all-terminal reliability of a network is an essential task and can be implemented in different ways. These include, but are not limited to, Dynamic Programming and evolution-based methods [5] such as Simulated Annealing, Ant Colony Optimization, and Artificial Bee Colony algorithm. By taking advantage of machine learning methods, it would be practically valuable and more efficient to tackle the network reliability improvement problem using machine learning.

In this work, our goal is to develop a machine learning-based framework for evaluating and improving all-terminal network reliability. In particular, the reliability improvement problem focuses on data-driven network design optimization, for which for an initial network structure, we determine the best sequence for adding links to maximize the metrics related to the all-terminal reliability over a finite time horizon.

It is worth pointing out that for our problem of interest, the complexity of the problem is heightened by making the design space larger, as multiple link options of varying quality are considered. Additional practical constraints, such as budget constraints, pose tracking the feasibility of each action as an essential factor to consider in algorithm development. Given these, we consider the network design as a Reinforcement Learning (RL) problem. Furthermore, by using Deep Neural Networks to model network reliability, we frame the design problem as a Deep Reinforcement Learning (DRL) problem [6].

## 2 RELATED WORK

With the extent of interconnected networks in everyday life, there is an essential need to understand network reliability and achieve better resilience in such critical infrastructures. Numerous studies in this area try to maximize all-terminal network reliability, and such problems are classified as NP-hard problems [3]. Indeed, in solving such problems, calculating all-terminal network reliability, and optimizing the network structure are of the most computational challenges.

Dengiz et al. [5] used a genetic algorithm to maximize a network's all-terminal reliability. Ramirez-Marquez and Rocco [4] suggested a novel algorithm based on hybrid optimization, which combines a probabilistic solution discovery with a Monte Carlo simulation to estimate the all-terminal reliability of the network. Recently, Goharshady and Mohammadi [7] suggested an innovative method for computing reliability for networks with small treewidth (as in subway networks), which can be scaled to networks with a higher count of vertexes.

Scholars have attempted to estimate network reliability by developing ANNs as surrogate models. Srivaree-Ratana and Smith [2] developed an ANN model that used link reliability and a reliability upper bound as inputs and computed the all-terminal reliability as its target value for a network with ten nodes. Davila-Frias et al. [8] proposed an approach to predict the network's reliability with varying graph sizes. They used different graph embedding methods to represent the network as the input of the DNN model. Recently, Davila-Frias and Yadav [9] addressed all-terminal reliability estimation using Convolutional Neural Networks (CNNs). They defined a multi-dimensional vector representing the network adjacency matrix, link reliability, and topological attributes as the input of the CNN model. The output layer is a regression preceded by a sigmoid layer that predicts the network's reliability. This allows for exploiting the power of CNNs to identify correlations in the spatial patterns of the adjacency matrix that might be relevant for minimizing the loss function during training.

## 3 METHODOLOGY

DNN models are a powerful tool for creating a surrogate model for all-terminal reliability. We propose using a Bayesian variant of a DNN model to estimate the reliability of networks with vertex counts in a defined range. Exploiting the flexibility of this type of models to estimate all-terminal reliability will be balanced by the regularization capabilities of the Bayesian component in the inference procedures.

A sequence of subtasks will be tackled before obtaining a viable surrogate model. We first create a dataset comprised of several sample networks with a relatively wide range of all-terminal and individual edge reliability values. Then, the all-terminal reliability is calculated using an exact state enumeration algorithm. Next, we create a tensor representation of these networks based on adjacency matrices and spectral analysis. As these tensors arise from 2-dimensional arrays, we consider they encode relevant spatial information that can be extracted. To exploit this in a data-driven fashion, we train a baseline CNN model to create two desired outputs: an initial estimation of the all-terminal reliability and a data-driven embedding of the sample networks. We then train a DNN model in a Bayesian framework using Stochastic Variational Inference with the CNN embedding as input. This procedure enables inference on the all-terminal reliability values while quantifying the uncertainty of the estimates through Bayesian reasoning by sampling from the posterior distributions using DNNs as learnable transformations conditioned on the observed data. The complete modeling pipeline from tensor representation to SVI+DNN is now the surrogate model.

One of the most valuable use cases for this surrogate model is to speed up network design optimization tasks. Furthermore, as this approximation is not as computationally expensive as those exact methods, we can use iterative optimization methods and take sequential samples from the decision space. To take full advantage of this, we propose using a Deep Reinforcement Learning framework in a setup where it will actively learn from observed sequences of network designs while maximizing all-terminal network reliability.

### 3.1 *Dataset Generation*

Network graphs are generated with sizes of 8, 9, and 10 nodes with random numbers of undirected edges. We generate $n$ graphs of each size for a total $N = 3n$ samples. We consider all nodes to be perfectly reliable, all edge failures are independent, and edge reliabilities can take varying values on pre-defined levels. Any given edge connecting two nodes is included with a random uniform chance defined by $p_{add}$. Link reliabilities are randomly assigned and uniformly chosen from a list of values $r_{list}$. For this dataset, all the graphs are connected: all nodes can communicate with each other, so their all-terminal reliability values are always non-zero.

The all-terminal reliability is first computed using an enumeration algorithm. This calculation is validated by another automated procedure developed to compute the reliability polynomial of an arbitrary graph [1]. Finally, the pairs of graphs and their exact all-terminal reliability values are stored for each sample. The generated sets of random graphs have varying numbers of edges and nodes, topology, and edge reliability configurations. This procedure was inspired by the work in [2] and [9]. However, more complexity was added by including multiple vertex counts in a single modeling pipeline and by making link reliability values vary both within a single graph and across graphs.

### 3.2 *Tensor representation*

For each of the generated graphs in the dataset, we created a tensor of size 10×10×3 comprised of three matrices of encoded information about the network structure. Each matrix is of size 10×10 as the largest vertex count in our samples is ten nodes. For graphs with 8 and 9 nodes, we added zeroes on the last 1 or 2 rows and columns correspondingly.

The adjacency matrix encoding the network edge and node structure is the first matrix. The second matrix is analog to the adjacency matrix but replaces the encoded values for each edge with their corresponding reliability values. We refer to this as the edge reliability matrix in this paper. The third matrix is the normalized Laplacian matrix. Given the adjacency matrix $\boldsymbol{A}$ and the node degrees in diagonal matrix form $\boldsymbol{D}$, the Laplacian matrix $\boldsymbol{L}$ and normalized Laplacian matrix $\boldsymbol{N}$ are computed by:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} \tag{2}$$

$$\boldsymbol{N} = \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} \tag{3}$$

We hypothesize that as this matrix encodes spectral information about the graph structure, it is a suitable input for a data-driven model with capabilities for exploiting spatial information. Therefore, the spatial information is encoded in the matrix arrays and will be used for the CNN embedding.

### 3.3 *CNN embedding*

Inspired by the work in [9], we use a CNN architecture for obtaining baseline estimation of all-terminal reliability. The following is a list of the sequential layers in this model:

- Convolution: 8 filters, 3×3×3
- Leaky ReLU
- Average Pool 2×2
- Convolution: 16 filters, 3×3×8
- Leaky ReLU
- Average Pool 2×2
- Convolution: 32 filters, 3×3×16
- Leaky ReLU
- Average Pool 2×2
- Convolution: 32 filters, 3×3×32
- Leaky ReLU
- Average Pool 2×2
- Fully Connected Layer (288,512)
- Leaky ReLU
- Fully Connected Layer (512,1024)
- Leaky ReLU
- Fully Connected Layer (1024,1)
- Sigmoid

We use a sigmoid function as the final activation to guarantee that the output is a real number between 0 and 1. We hypothesize that the three final fully connected layers work as a latent space embedding of the features extracted from the tensor representation.

The outputs of the second to last fully connected layer are used as a representation of the graph in the latent space. We refer to this as the data-driven CNN embedding in this paper. This vector of size 1024, together with the baseline estimates, is used as the input for the SVI+DNN model.

### 3.4 *SVI+DNN surrogate model*

To create a surrogate model for calculating all-terminal reliability, we use a Bayesian framework to train the DNN model [10]. In addition, we selected the Pyro probabilistic programming framework [11] and used their proposed terminology to describe the procedures.

On the weights and biases of the layers of the DNN, we place probabilistic priors. Conditioning this on the observed data will lead to an intractable posterior formulation. To resolve the challenge, sampling methods such as Hamiltonian Monte Carlo (HMC) can be applied. However, such an approach would be computationally costly for models with many parameters. Instead, we approximate the intractable posterior using Stochastic Variational Inference (SVI) [12]. This involves optimizing the Evidence Lower Bound (ELBO) using stochastic gradient steps, where the said lower bound is related to the Kulback-Leibler (KL) divergence of a proposed surrogate distribution function, called *guide*, and the true posterior. This proposed guide is based on the DNN model and a selected family of probability distributions. As the distribution family, we selected the Delta distribution, leading to Maximum a Posteriori estimates of the posterior distribution parameters.

The following equation shows the ELBO as an expectation with regards to the guide distribution:

$$ELBO = E_{q_{\phi(z)}}\left[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{x},\mathbf{z})\right] \tag{4}$$

where observations are represented by $\mathbf{x}$ and latent random variables $\mathbf{z}$, and $p_\theta(\mathbf{x},\mathbf{z})$ is their joint probability distribution with parameters θ. $q_\phi(\mathbf{x},\mathbf{z})$ is the guide distribution with parameters $\phi$. Stochastic Gradient Descent is performed in the parameter space for $\phi$ to find a guide to decrease the divergence between the guide and posterior.

As this approximation would scale inefficiently for fully parameterized guides, we exploited the idea of amortization, which involves using a DNN to map the inputs to the required parameters in the guide model to reduce the number of trainable parameters. It is an exchange of parameters for a functional mapping.

This framework allows for estimating predictive distributions for all-terminal network reliability using the network CNN embedding and the baseline estimates as the inputs. The architecture of the model network is as follows:

- Bayesian Linear Layer (1025,128)
- Leaky ReLU
- Bayesian Linear Layer (128,128)
- Leaky ReLU
- Bayesian Linear Layer (128,2)
- Sigmoid → (Output[1], Output[2])
- Normal Distribution
  - Mean: initial estimation + Output[1]
  - Variance: Output[2]

### 3.5 *DRL for network design optimization*

We use DRL to solve the sequential network design problem. The choice for DRL stems from the sequential nature of the design problem. Technically, feasibility constraints are factored-in by using Maskable Proximal Policy Optimization (M-PPO) as our optimization algorithm [13], which is a variant of Proximal Policy Optimization (PPO) that can take into account rule-based action feasibility.

We formulate the design problem with a reliability improvement objective. The DRL agent will decide the best next edge to be included in the graph for a given initial network to maximize the all-terminal reliability. We enable discrete levels of edge quality with varying levels of cost and reliability. Moreover, we also consider a cost constraint such that a budget limits how many edges can be added. The agent must find a finite sequence of edge decisions that maximize its reward.

Mathematically, the problem is formulated as:

$$\max_{\boldsymbol{A_t}|\boldsymbol{O_t}} \mathcal{R}_t = \log(R_t) - \log(1 - R_t) + \lambda \mathcal{R}_{t-1} \quad (5)$$

$$\boldsymbol{A_t} = [x_{ij}, q_{ij}] \quad (6)$$

$$\boldsymbol{O_t} = [x_{\boldsymbol{ij}}, c_{ij}, C_{t-1}] \quad (7)$$

$$r_{ij} = p(x_{ij}, q_{ij}) \quad (8)$$

$$\boldsymbol{s.t.} \sum_i \sum_j c_{ij} x_{ij} = C_t \leq B \quad (9)$$

For all decision steps $t$, the DRL decides the actions $\boldsymbol{A_t}$ to take given the observations $\boldsymbol{O_t}$ to maximize the reward $\mathcal{R}_t$ which depends on the all-terminal reliability $R_t$ and a discount factor $\lambda$ that balances the cumulative rewards. The available actions include adding a new edge $x_{ij}$ with a chosen quality level $q_{ij}$. The quality level affects the cost of adding said link, $c_{ij}$, and the link reliability value, $r_{ij}$. The observations include the graph topology represented by the links in the network, the cost of each of the present links, and the total cost of the network up to the previous decision step $C_{t-1}$. The last equation shows the budget constraint to keep the total cost below $B$.

For implementation, we created an environment using the OpenAI-Gym [14] framework and trained the agents based on the Stable Baselines specified models [15]. The SVI+DNN surrogate model is used to approximate the all-terminal reliability to speed up the required computations.

## *4 NUMERICAL EXPERIMENT*

We conducted a numerical experiment by creating a dataset with $N = 6000$ randomly generated networks with vertex counts of 8, 9, and 10 nodes (i.e., $n = 2000$). Edge creation probability is $p_{add} = 0.3$. We considered five options for the levels of edge reliability: $r_{list} = [0.8, 0.85, 0.9, 0.95, 0.99]$. For model validation purposes, we separate these samples into "train," "validation," and "test" subsets, with proportions of 75%, 15%, and 10%, respectively.

We run our experiment on a Google Collaboratory Virtual Machine with 8 Intel Xeon processors with 4-Cores@2.2Ghz and 51GB of RAM. First, all the graphs in the sample were transformed to their respective tensor representation. This step took between 4ms and 6ms per graph. The initial CNN model was adjusted on the training set using an Adam optimizer to minimize an MSE loss metric. The learning rate was set on 0.002 following a cosine annealing schedule until 0.001. We use mini-batch updates of 128 samples during 1000 epochs. This procedure took 13m and 46s (i.e., 0.8s per epoch). This step output the initial estimation and the CNN embedding. For a new tensor representation, obtaining an initial estimation and embedding took between 4 and 8ms.

We trained the SVI+DNN model during 20 optimization steps, using mini-batch updates of 32 graphs and a ClippedAdam optimizer. The learning rate was set to 0.001. The training took 14m and 51s with around 40s per step for training and 5s per step to evaluate performance on the validation set. We sampled 100 times from the predictive posterior distribution for estimations and use the mean as the final prediction. Given a graph, executing the complete estimation from tensor representation to the surrogate model took around 200ms. For graphs of similar size, our implementation of the enumeration algorithm took 30s for an exact calculation of the corresponding all-terminal reliability.

Figure 1 shows the predictions on the training set (blue line) compared to the sorted actual values (black line). The shaded area around these lines represents the 90% credible interval for the posterior estimates. One can see that the predictions are generally close to the true values. For example, the RMSE values of the predictions on the train, validation, and test sets are 0.003, 0.08, and 0.08, respectively. Compared to the results presented in the related studies [2], our proposed methodology achieves adequate precision in tackling a more difficult task of estimating all-terminal network reliability for sets of graphs with varying node counts and edge reliability values on a single modeling pipeline.

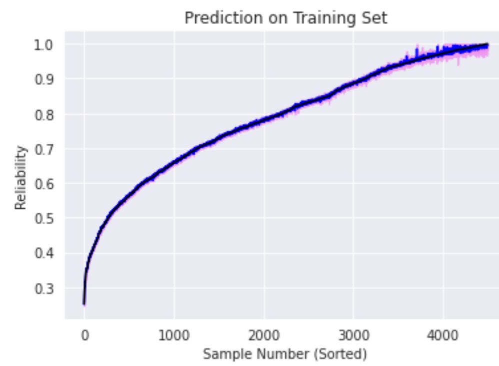


*Figure 1. Prediction comparison*

After completing the surrogate model, the model was used to estimate the reliability of sequential graphs during the training of our M-PPO agents. We defined a DRL environment with an initial path network with eight nodes. For the initial edges, the reliability was set to be 0.8. For any given decision step, the agent can add edges with one of three quality levels: 0.9, 0.95, and 0.99, and these edges have costs of 1, 2, and 3 units. Moreover, there is a maximum budget of 10 units of cost, and the agent stops adding edges when the budget is exceeded.

We trained the M-PPO agent during 5000 episodes, and it was completed in 24m and 42s. For a similar setup using a Reliability Polynomial for the all-terminal exact calculations, the observed training time was close to 16 hours.

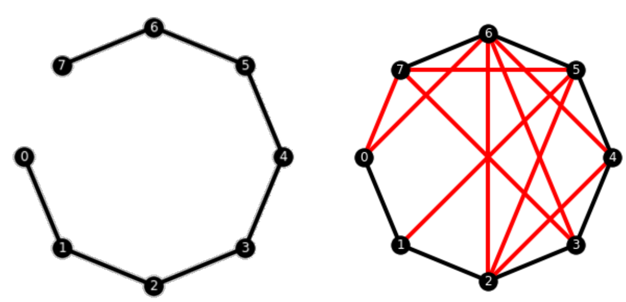

*Figure 2. Initial (left) and final (right) graphs*

The designed final network prioritizes connectivity over individual edge quality. This is observed from the exclusive use of links with the lowest quality and cost. The reliability estimated by the surrogate model is 0.9452, and the exact calculation is 0.9927, within the bounds of our validation error. Similar designs maximizing connectivity are also preferred by the M-PPO agent when the training is done using the exact computations in similar problem setups. Figure 2 shows the initial and final graphs for the best solution found after training, where the black links represent the existing edgesinitial network.

## 5 CASE STUDY

To further illustrate the proposed methodology, we conduct a case study involving budget constraints. We focus our attention on the backbone computer network of the Gazi University in Ankara, Turkey [16]. The computer network with 11 nodes is shown in Figure 3. Three types of links can be chosen with individual edge reliability values of [0.99, 0.995, 0.999]. The quality level of each edge has an associated cost per meter of distance covered: [$12, $17, $28].

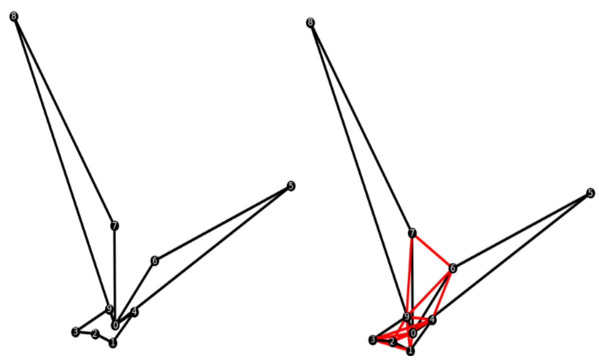

*Figure 3. Initial (left) and final (right) graphs for the Case Study*

Before applying our method, some adjustments to the original formulation are made. First, to apply the current surrogate model, we merge one of the nodes in the network to its neighbor for simplicity. Second, we assume that all the links in the original network are at the same quality level (set it at the lowest value). We use a budget value of $218,635, as in the solution with the lowest cost in the original formulation. Third, while the lengths of existing edges were provided in [16], to compute the distances not considered in the original formulation, we located the university buildings using Google Earth and measured the related distances so that the associated costs for adding new edges can be calculated.

We created a DRL agent to maximize the all-terminal reliability of the Gazi Network by adding new edges between nodes. The best solution in the original formulation [16] had an all-terminal reliability of 0.9945 and a cost of $322,865. This solution did not modify the network topology. Our best solution found has a higher reliability of 0.9998, and it is shown in Figure 3 as the final (right) network. The total cost is $199,140. It is worth pointing out that our solution added edges and modified the network topology. Moreover, our solution maximizes the number of edges by choosing the lowest quality level and adding edges with the lowest distances between pairs of nodes not yet in the network.

## 6 CONCLUSIONS AND FUTURE RESEARCH

In this paper, we proposed a surrogate model to efficiently evaluate all-terminal network reliability. With the goal of improving the all-terminal reliability of a network, a DRL environment defining the action space, the per-period rewards, feasibility constraints, system evolution conditions, and state transition dynamics was developed for solving the network design problem. The practical value of this work is twofold. First, the Bayesian data-driven model is flexible enough to approximate the all-terminal network reliability of arbitrary graphs of varying sizes. It can work with varying edge reliability values and does not need additional computations as input, such as reliability upper bounds. The SVI framework allows for quantifying estimation uncertainty and incentivizing regularization to avoid overfitting. Second, enhanced and supported by the SVI+DNN approximation method, exploring the use of DRL for network reliability improvement provides a useful data-driven algorithm capable of tackling complex design problems in ample design spaces.

Our future work will be focused on validating the network design solutions by diagnosing convergence and identifying suitable conditions. Indeed, validating such a data-driven model is challenging, but quantifying the optimality gap using total enumeration for small-scale problems is still possible. Furthermore, we have identified research opportunities for leveraging the estimation uncertainty naturally arising from the Bayesian model. We will exploit this by using other optimization methods such as Bayesian Optimization with Gaussian Process surrogate models or Thompson Sampling. In another direction, we will explore possibilities of balancing maximizing reliability with other objectives such as the equity across a network in humanitarian applications.

## AKNOWLEDGMENTS

The authors express their gratitude to the funding from National Science Foundation (NSF) EPSCoR RII Track-2 Program under the grant number OIA-2119691. The findings and opinions expressed in this article are those of the authors only and do not necessarily reflect the views of the sponsors.

## REFERENCES

1. Azucena, J., Wells, H., Liao, H., Sullivan, K., and Pohl, E. *Applying deep reinforcement learning to improve the reliability of an infrastructure network*. in *Proceedings of the 60th ESReDA Seminar*. 2022. Grenoble, France.
2. Srivaree-Ratana, C. and A.E. Smith. *Estimating all-terminal network reliability using a neural network*. in *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*. 1998. IEEE.
3. Valiant, L.G., *The complexity of enumeration and reliability problems.* SIAM Journal on Computing, 1979. **8**(3): p. 410-421.

4. Ramirez-Marquez, J.E. and C.M. Rocco, *All-terminal network reliability optimization via probabilistic solution discovery.* Reliability Engineering & System Safety, 2008. **93**(11): p. 1689-1697.
5. Dengiz, B., F. Altiparmak, and A.E. Smith, *Efficient optimization of all-terminal reliable networks, using an evolutionary approach.* IEEE transactions on Reliability, 1997. **46**(1): p. 18-26.
6. Arulkumaran, K., et al., *Deep reinforcement learning: A brief survey.* IEEE Signal Processing Magazine, 2017. **34**(6): p. 26-38.
7. Goharshady, A.K. and F. Mohammadi, *An efficient algorithm for computing network reliability in small treewidth.* Reliability Engineering & System Safety, 2020. **193**: p. 106665.
8. Davila-Frias, A., S. Salem, and O.P. Yadav. *Deep Neural Networks for All-Terminal Network Reliability Estimation*. in *2021 Annual Reliability and Maintainability Symposium (RAMS)*. 2021. IEEE.
9. Davila-Frias, A. and O.P. Yadav, *All-terminal network reliability estimation using convolutional neural networks.* Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 2020: p. 1748006X20969465.
10. Kingma, D.P., T. Salimans, and M. Welling, *Variational dropout and the local reparameterization trick.* Advances in neural information processing systems, 2015. **28**.
11. Bingham, E., et al., *Pyro: Deep universal probabilistic programming.* The Journal of Machine Learning Research, 2019. **20**(1): p. 973-978.
12. Hoffman, M.D., et al., *Stochastic variational inference.* Journal of Machine Learning Research, 2013.
13. Huang, S. and S. Ontañón, *A closer look at invalid action masking in policy gradient algorithms.* arXiv preprint arXiv:2006.14171, 2020.
14. Brockman, G., et al., *Openai gym.* arXiv preprint arXiv:1606.01540, 2016.
15. Raffin, A., et al. *Stable baselines3*. 2019; Available from: *https://github.com/Stable-Baselines-Team/stable-baselines3-contrib*.
16. Altiparmak, F., B. Dengiz, and A.E. Smith, *Optimal design of reliable computer networks: A comparison of metaheuristics.* Journal of heuristics, 2003. **9**(6): p. 471-487.

## *BIOGRAPHIES*

José Carlos Hernández Azucena
Department of Industrial Engineering
University of Arkansas
4117 Bell Engineering Center
Fayetteville, Arkansas 72701 USA

e-mail: jch051@uark.edu

Mr. Azucena is a Ph.D. student in the Industrial Engineering Department at the University of Arkansas. In 2014, he received a B.S. degree in Business Engineering from ESEN, Santa Tecla, El Salvador. His research interests include Network Reliability, Deep Learning, and Deep Reinforcement Learning.

Seyyed Farid Hashemian
Department of Industrial Engineering
University of Arkansas
4121 Bell Engineering Center
Fayetteville, Arkansas 72701 USA

e-mail: sfhashem@uark.edu

Mr. Farid Hashemian received his bachelor's and master's degree in Industrial Engineering in 2018 and 2020. Currently, he is a Ph.D. student in Industrial Engineering at the University of Arkansas. He is passionate about data-driven decision-making and real-world problem-solving using Machine Learning, Data Analysis, and Optimization.

Dr. Haitao Liao
Department of Industrial Engineering
University of Arkansas
4207 Bell Engineering Center
Fayetteville, Arkansas 72701 USA

e-mail: liao@uark.edu

Dr. Haitao Liao is Professor, and John and Mary Lib White Endowed Systems Integration Chair in the Industrial Engineering Department at the University of Arkansas. Haitao received his Ph.D. degree in 2004 from the Department of Industrial and Systems Engineering at Rutgers University. He also received a B.S.E.E. from Beijing Institute of Technology. His research interests focus on reliability models, smart maintenance, and data analytics. He received a National Science Foundation CAREER Award in 2010, William A. J. Golomski Award for four times, 2015 Stan Ofsthun Best Paper Award, and 2017 Alan O. Plait Award for Tutorial Excellence. He is a Fellow of IISE, a member of INFORMS, and a lifetime member of SRE.

Dr. Edward Pohl
Department of Industrial Engineering
University of Arkansas
4207 Bell Engineering Center
Fayetteville, Arkansas 72701 USA

e-mail: epohl@uark.edu

Dr. Ed Pohl is a Professor and Head of the Department of Industrial Engineering at the University of Arkansas. Ed spent twenty years in the United States Air Force, where he served in a variety of engineering, analysis, and academic positions. Ed received his Ph.D. in systems and industrial engineering from the University of Arizona, an M.S. in reliability engineering from the University of Arizona, an M.S. in systems engineering from AFIT, an M.S. in engineering management from the University of Dayton, and a B.S.E.E. from Boston University. Ed is a Fellow of IISE, a Fellow of SRE, a Fellow of ASEM, a senior member of ASQ, a senior member of IEEE, and member of INFORMS. He is a two-time winner of the Alan Plait award for Outstanding Tutorial at RAMS, as well as a two-time winner of the William A. J. Golomski Award.