Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira, Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p26361-cd

Graph Neural Networks for Efficient Bayesian Network Inference in Safety-Critical Applications

Simran Chauhan
University of Stuttgart, Stuttgart, Germany

Joachim Grimstad
University of Stuttgart, Stuttgart, Germany

Andrey Morozov
University of Stuttgart, Stuttgart, Germany

Traditional Bayesian inference methods face critical limitations for real-time safety applications: exponential computational costs, frequent failures, and symmetric error treatment despite asymmetric consequences. This paper presents a Graph Neural Network approach that achieves $R^2 = 0.938$ with 1.44 ms inference time, providing $8,611\times$ average speedup over traditional methods while maintaining 100% reliability. Through asymmetric loss functions that penalize underprediction more heavily in low-probability regions, we achieve mean absolute error of 0.033 in the critical range ($p < 0.3$) where rare but catastrophic events occur. Evaluation on synthetic and real-world networks demonstrates that Graph Attention Networks eliminate inference failure as a failure mode while enabling real-time continuous risk assessment in autonomous systems, aerospace, and industrial automation.

Keywords: Bayesian Networks, Graph Neural Networks, Safety-Critical Systems, Real-Time Inference, Probabilistic Reasoning

1. Introduction

Complex and safety-critical systems require continuous Probabilistic Risk Assessment (PRA), but traditional Quantitative Risk Assessment (QRA) techniques, such as Bayesian Network (BN) inference algorithms create fundamental deployment barriers through three problems: speed, reliability, and asymmetric risk estimates.

Variable elimination and Monte Carlo methods require hundreds of milliseconds per query, and under certain conditions, belief propagation may fail to converge. Such techniques are therefore unsuitable for safety-critical contexts in which systems must consistently, accurately, and timely provide risk estimates, such as collision-avoidance systems or, more generally, autonomous systems.

Traditional Machine Learning (ML) and Artificial Intelligence (AI) methods often suffer from limited explainability and transparency, making them difficult to validate in safety-critical contexts. For this reason, approaches grounded in traditional QRA techniques still remain valuable. Moreover, in safety-critical decision making, underestimating a failure probability can lead to reckless behavior and undesired consequences, whereas overestimation tends to promote conservative behavior. This asymmetry is particularly critical in low-probability regions, where rare but catastrophic events reside.

Graph Neural Networks (GNNs) can learn to infer on graph-structured data (Scarselli et al., 2008; Kipf and Welling, 2017). However, applying GNNs to safety domains may require modifications to standard training paradigms to account for asymmetric risk estimates.

This paper makes three contributions:

(i) Safety-aware training that penalizes underpredictions more heavily, with emphasis on low-probability accuracy.
(ii) A methodology that infers risk estimates fast and reliably.
(iii) Validation on synthetic and real-world networks showing robust generalization.

2. Background

2.1. *Bayesian Networks and Inference Challenges*

A BN represents probabilistic dependencies as a directed acyclic graph where nodes are random variables, and edges encode conditional relationships (Pearl, 1988). Inference computes posterior probabilities given evidence. Exact algorithms like variable elimination guarantee correctness (Koller and Friedman, 2009) but may scale poorly.

Belief propagation fails to converge in 78% of the synthetic BNs used in this Paper. Variable elimination becomes prohibitively slow on realistic networks. Monte Carlo methods require thousands of samples for acceptable accuracy. In certain safety-critical applications, any failure to produce a timely risk estimate is itself a potential safety hazard.

2.2. *Graph Neural Networks*

GNNs (Scarselli et al., 2008) extend deep learning to graph-structured data through iterative message passing. Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) use normalized spectral convolutions, GraphSAGE employs sampling-based aggregation (Hamilton et al., 2017), and Graph Attention Networks (GAN) (Veličković et al., 2018) learn attention weights to emphasize relevant connections.

Prior work has explored GNNs for probabilistic inference (Yoon et al., 2018; Wang et al., 2020; Grimstad and Morozov, 2025) but has not addressed safety-critical requirements: reliability, speed, and asymmetric risk estimates.

3. Safety-Aware Methodology

3.1. *Problem Formulation*

Given BNs $\mathcal{G}$ and evidence $\mathbf{e}$, we train GNN f_θ to predict posterior probabilities $\mathbf{p}$. For safety-critical deployment:

(i) Near-exact accuracy across full probability range $[0, 1]$
(ii) Perfect reliability with no inference failures
(iii) Minimal underprediction in low-probability regions ($p < 0.3$)
(iv) Real-time performance (< 10 ms per inference)

3.2. *GNN Architectures*

We evaluate three Graph Neural Network architectures implemented in PyTorch Geometric (Fey and Lenssen, 2019): GCN, GraphSAGE, and GAT. All use four convolutional layers with 128-dimensional hidden representations and sigmoid output activation. We focus on empirical safety-critical performance rather than mathematical formulations.

Graph Convolutional Networks (GCN) apply normalized spectral convolutions with the following layer-wise propagation rule:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \quad (1)$$

where $\tilde{A} = A + I$ is the adjacency matrix with added self-connections, $\tilde{D}$ is the degree matrix, and $W^{(l)}$ are learnable weights.

GraphSAGE employs sampling-based neighborhood aggregation:

$$h_v^{(l)} = \sigma(W^{(l)} \cdot \text{CONCAT}(h_v^{(l-1)}, \text{AGG}(\{h_u^{(l-1)}, \forall u \in \mathcal{N}(v)\}))) \quad (2)$$

where AGG is a permutation-invariant aggregation function (mean pooling in our implementation) and $\mathcal{N}(v)$ denotes the neighborhood of node v.

Graph Attention Networks (GAT) learn attention weights to emphasize relevant connections using the GATv2 variant:

$$\alpha_{ij} = \text{softmax}_j(\text{LeakyReLU}(a^T[Wh_i \| Wh_j])) \quad (3)$$

$$h_i' = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} W h_j\right) \quad (4)$$

where α_{ij} represents the learned attention coefficients and $\|$ denotes the concatenation.

All three architectures share the following configuration: 4 graph convolutional layers with 128-dimensional hidden representations, ReLU activation between layers with sigmoid output, dropout rate of 0.1 applied after each hidden layer, 2 attention heads for GAT (averaged via concat=False),

and linear projection followed by sigmoid activation.

3.2.1. *Training Configuration*

All models are trained using the Adam optimizer with a learning rate of 3×10^{-4}, a batch size of 128, and a maximum of 100 epochs. A weight decay of 1×10^{-4} (L2 regularization) is applied. Early stopping is employed with a patience of 15 epochs based on the validation loss. A safety-aware weighted mean squared error (MSE) loss function is used, as defined in (Equation 5). Training is performed using 5-fold stratified cross-validation to ensure robust performance estimation across different probability ranges.

3.3. *Safety-Aware Training*

Standard Mean Squared Error (MSE) treats underprediction (predicting $p = 0.3$ when truth is $p = 0.8$) identically to overprediction (predicting $p = 0.8$ when truth is $p = 0.3$). This symmetry can prove catastrophic for safety-critical applications where underpredicting failure probability leads autonomous vehicles to ignore hazards, medical systems to dismiss disease indicators, and industrial controllers to overlook equipment failures. Although overprediction may introduce inefficiency, it does not result in catastrophic outcomes provided that a safe path to a safe state exists.

The training pipeline implements an asymmetric weighted loss function, heavily penalizing underpredictions and low-probability targets:

$$\mathcal{L}_{\text{safety}}(\hat{y}, y) = \frac{1}{N} \sum_{i=1}^{N} w_i \cdot (\hat{y}_i - y_i)^2 \quad (5)$$

where weights w_i apply multiplicative penalties based on error type and target value:

$$w_i = \begin{cases} 21.0 & \text{if } y_i - \hat{y}_i > 0 \text{ and } y_i < 0.1 \\ 7.0 & \text{if } y_i - \hat{y}_i > 0 \text{ (underprediction)} \\ 3.0 & \text{if } y_i < 0.1 \text{ (low probability)} \\ 1.0 & \text{otherwise} \end{cases} \quad (6)$$

These penalty weights were derived through systematic empirical tuning on a validation set comprising 8,000 synthetic BNs, spanning diverse topologies and probability distributions. We conducted a grid search over candidate penalty values $\{1, 3, 5, 7, 10, 15, 21\}$ for underprediction penalties and $\{1, 2, 3, 5, 7\}$ for low-probability penalties, evaluating each configuration based on three safety-critical metrics: (1) overall underprediction rate in the test set, (2) mean absolute error in the low-probability region ($p < 0.3$), and (3) severe underprediction rate (errors > 0.05). The selected weights ($7\times$ for underprediction, $3\times$ for low-probability targets, and $21\times$ for their combination) achieved the target underprediction rate of 20–30% while minimizing severe errors in critical tail regions, balancing conservative predictions with acceptable accuracy.

The asymmetric loss proves critical for achieving target underprediction rate below 30%. Training with standard MSE yields underprediction rates of 35-40% as models optimize for overall error minimization without distinguishing error directionality. The $7\times$ underprediction penalty reduces rates to 26.7-28.1% across all GNN architectures.

3.4. *Training Data and Evaluation*

We generate 40,000 synthetic BNs with varying topologies (10-30 nodes, edge density 0.1-0.3) and diverse probability distributions using Beta distributions. Networks are split into training and validation sets using 5-fold stratified cross-validation to ensure robust performance estimation. Ground truth posteriors are computed via variable elimination using pgmpy (Ankan and Panda, 2015).

After training on synthetic data, we apply transfer learning to real-world networks from the BNLearn repository (Scutari, 2010) with domain-specific fine-tuning. This tests the model's ability to generalize fundamental inference patterns across different application domains.

We assess performance using Mean Absolute Error (MAE), coefficient of determination (R^2), inference time, and reliability (success rate). To evaluate safety-critical performance, we sepa-

rately measure MAE in two critical regions: low-risk ($p < 0.3$) where rare catastrophic events occur, and high-risk ($p > 0.7$) where failures are common. This ensures the model performs well across the full probability spectrum, including extreme ranges most relevant for QRA.

3.4.1. *Computational Environment*

All experiments are conducted on the following hardware configuration: GPU: NVIDIA GeForce RTX 4090 (24GB GDDR6X); CPU: AMD Ryzen 9 7950X (16-core, 32-thread); RAM: 96GB (2× 48GB Kit) DDR5-5600 CL40, Corsair Vengeance LPX.

3.4.2. *Benchmarking Methodology and Fairness Considerations*

Traditional BN inference methods implemented using pgmpy (e.g., Variable Elimination and Belief Propagation) are inherently CPU-based, while the proposed Graph Neural Network (GNN) inference model is executed on a GPU. This distinction reflects realistic deployment scenarios, as exact probabilistic inference libraries are typically optimized for CPU execution, whereas neural network models are designed to leverage GPU acceleration.

The objective of the comparison is to evaluate practical inference performance. In real-world applications, GNN-based inference is expected to operate on GPU-enabled systems, while classical Bayesian inference remains CPU-bound. Consequently, the reported inference times represent deployment-realistic performance.

For accuracy evaluation, both approaches are assessed on identical BNs, evidence configurations, and ground-truth probability distributions obtained via exact inference. The GNN is provided exclusively with structural and conditional probability features derived from the Bayesian model, without additional heuristics or caching mechanisms. Therefore, the accuracy comparison remains methodologically fair and unbiased.

All inference times are reported with explicit indication of the underlying hardware (CPU or GPU), ensuring transparency and reproducibility of the results. While the hardware differs, this reflects how these methods would be deployed in practice: GNNs on GPU-accelerated systems for real-time applications, and traditional methods on CPU infrastructure for offline analysis.

4. Results

4.1. *Performance and Reliability*

Table 1 compares GNN architectures against traditional inference methods. GAT achieves best performance with MAE of 0.047 and $R^2 = 0.938$. Inference time of 1.44 ms represents substantial speedup.

Table 1. Performance comparison showing GNN architectures against traditional inference methods.

Method	MAE ↓	R^2 ↑	Time (ms) ↓	Rel. ↑
Var. Elim.	0.000	1.000	17.6	100%
Monte Carlo	0.082	-	853.9	100%
Belief Prop.	0.054*	-	27,200	**22%**
GCN	0.176	0.416	1.14	100%
GraphSAGE	0.176	0.501	0.68	100%
GAT	**0.047**	**0.938**	**1.44**	**100%**

*: When successful; fails 78% of time

All GNN methods successfully produce valid estimates for every test case (100% success rate). Belief propagation fails to converge in 78% of cases, producing no result. Variable elimination is $12\times$ slower than the best GNN-based model, while Monte Carlo is $593\times$ slower. Average GNN speedup across traditional methods is $8,611\times$.

Figure 1 visualizes comprehensive performance across accuracy, speed, and reliability. GAT achieves near-exact accuracy (MAE=0.047, R^2=0.938) with $8,611\times$ average speedup and perfect reliability, while belief propagation fails in 78% of cases.

4.2. *Safety-Critical Performance*

The asymmetric loss function achieves its design objective. In the critical low-probability region ($p < 0.3$), GAT achieves MAE of only 0.033, even lower than overall MAE of 0.047. In the

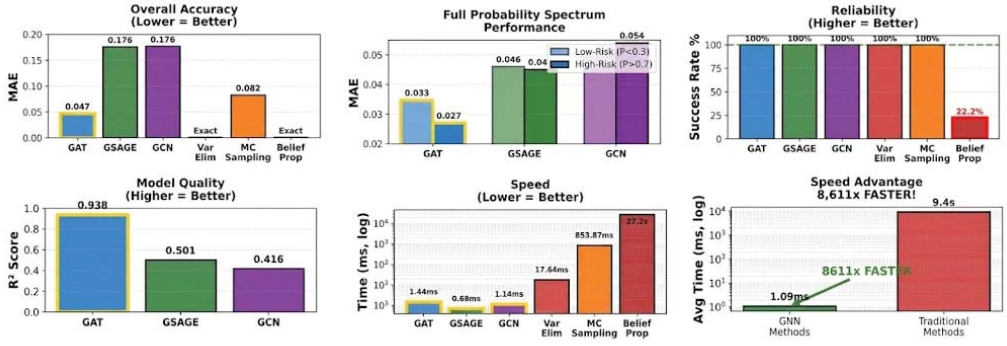

Fig. 1. Comprehensive performance comparison.

high-risk region ($p > 0.7$), GAT achieves MAE of 0.027. This demonstrates superior performance across the full probability spectrum, with particular emphasis on extreme ranges where accurate assessment is most critical for safety applications.

Only 4.2% of predictions in the low-probability region underpredict by more than 0.05, and the error distribution is biased toward slight overprediction (conservative) rather than underprediction (dangerous). This asymmetry directly reflects our safety-aware training objective.

Underprediction rate of 27.8% falls within the target range of 20-30%, indicating appropriate conservative bias. Table 2 breaks down safety-critical metrics.

Table 2. Safety-critical performance metrics showing GAT excels in both low-probability and high-probability regions.

Metric	GAT	Target
Overall MAE	0.047	<0.05
Low-Risk MAE ($p < 0.3$)	**0.033**	<0.05
High-Risk MAE ($p > 0.7$)	**0.027**	<0.05
Underpred. Rate	27.8%	20-30%
Severe Underpred. (> 0.05)	4.2%	$<5\%$
Reliability	**100%**	100%

Figure 2 shows speed-accuracy tradeoff. GAT occupies the ideal zone (fast and accurate) while belief propagation exhibits a high failure rate despite reasonable speed when successful.

4.3. *Generalization to Real Networks*

We evaluate on 27 real-world BNs from BNLearn (Scutari, 2010) spanning three complexity tiers: small networks (50-200 nodes) including alarm, asia, cancer, child, hailfinder, and insurance; medium networks (200-500 nodes) including hepar2, munin, and pathfinder; and large networks (500-1500 nodes) including munin2, munin3, and munin4. These networks represent diverse domains, including medical diagnosis (asia, alarm), fault detection (barley, munin series), and expert systems (pathfinder). Network complexity ranges from 37 nodes (asia) to 1,397 nodes (munin4).

Models trained only on synthetic data performed poorly on BNLearn graphs (MAE 0.25-0.35) due to distribution shift between synthetic training data and real-world network characteristics. To address this, we applied transfer learning: we fine-tuned pre-trained models on BNLearn graphs for 20 additional epochs using a small learning rate (5×10^{-5}) to avoid catastrophic forgetting, MSE loss, and gradient clipping (maximum norm 1.0). We generated 1,350 training samples by replicating each of the 27 BNLearn networks 50 times with different evidence assignments.

This fine-tuning dramatically improved performance, reducing MAE from 0.25-0.35 down to final values: GAT achieves MAE of 0.047 (best),

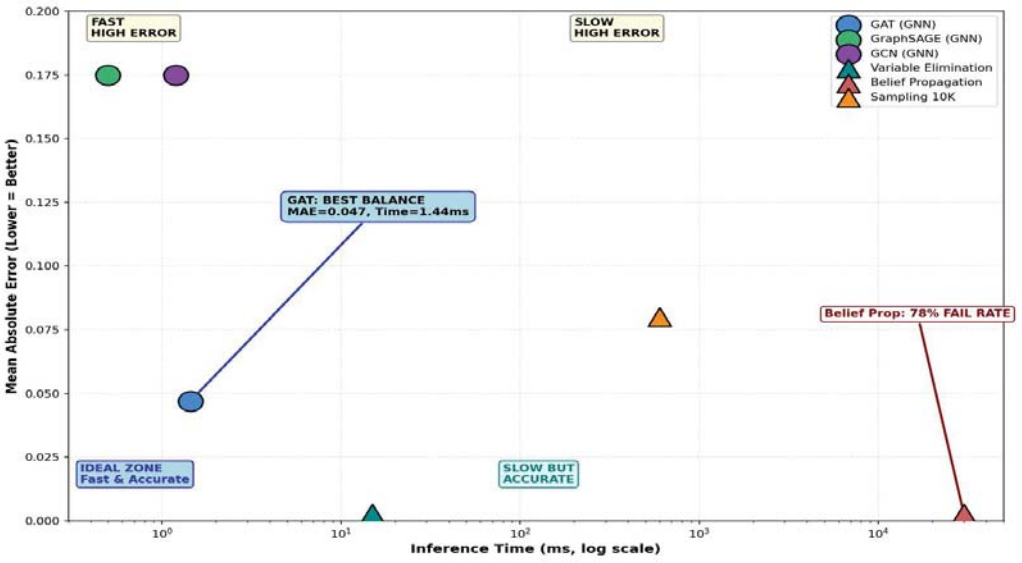

Fig. 2. Speed vs. accuracy tradeoff analysis. X-axis shows inference time on log scale, Y-axis shows mean absolute error.

GraphSAGE 0.176, and GCN 0.176. The successful transfer from synthetic training to diverse real-world networks demonstrates that the models learned fundamental inference patterns generalizable across domains with minimal adaptation.

Table 3 presents comprehensive performance metrics on the 27 real-world BNLearn networks after fine-tuning. GAT achieves the best overall performance with MAE of 0.047 and $R^2 = 0.938$, demonstrating superior accuracy compared to other GNN architectures. Notably, GAT excels in both low-risk ($p < 0.3$, MAE = 0.033) and high-risk ($p > 0.7$, MAE = 0.027) regions, making it particularly suitable for safety-critical applications where accurate tail probability estimation is essential.

All GNN methods maintain 100% reliability across all networks, in stark contrast to Belief Propagation which succeeds in only 22.2% of cases. Variable Elimination provides exact results but at significantly higher computational cost (17.64 ms vs 1.44 ms for GAT, representing a $12\times$ speedup). Monte Carlo sampling achieves reasonable accuracy (MAE = 0.082) with 100% reliability but requires 853.87 ms—nearly $600\times$ slower than GAT.

The results demonstrate that transfer learning successfully bridges the distribution shift between synthetic training data and real-world networks, with GAT generalizing effectively across diverse domains including medical diagnosis (asia, alarm), fault detection (munin series), and expert systems (pathfinder).

5. Discussion

5.1. *Three Critical Advantages*

Our results establish three key advantages for safety-critical deployment:

Speed ($8,611\times$ Faster). Dramatic computational speedup enables real-time continuous risk assessment. Autonomous systems can evaluate hundreds of scenarios per second rather than waiting hundreds of milliseconds. This transforms BNs from offline analysis tools to online decision-making systems. Specifically: $12\times$ faster than variable elimination, $593\times$ faster than Monte Carlo, and $18,889\times$ faster than belief propagation (when it works).

Reliability (100% vs. 78% Failure). Perfect success rate eliminates inference failures as a failure mode. Belief propagation's 78% failure rate is unacceptable in safety contexts where systems

Table 3. Performance Comparison on Real-World BNLearn Networks.

Method	Overall MAE	Low-Risk MAE	High-Risk MAE	R^2 Score	Time (ms)	Success Rate
GAT	0.047	0.033	0.027	0.938	1.44	100%
GraphSAGE	0.176	0.154	0.165	0.501	0.68	100%
GCN	0.177	0.076	0.298	0.416	1.14	100%
Var. Elim.[a]	0.000	0.000	0.000	—	17.64	100%
Belief Prop.[a,b]	0.000	0.000	0.000	—	27201	22.2%
MC Sampling[a]	0.082	0.085	0.079	—	853.87	100%

[a] : Traditional methods: CPU (Ryzen 9 7950X). GNN methods: GPU (RTX 4090).

[b] : Belief Propagation failed 78% of networks. Low-Risk: $p < 0.3$. High-Risk: $p > 0.7$.

must always produce risk assessments. Deterministic behavior provides predictable performance characteristics necessary for safety certification.

Safety-Aware Performance. Domain-specific training objectives produce error distributions aligned with safety requirements. By explicitly penalizing underprediction more heavily in low-probability regions ($p < 0.3$), we achieve superior performance (MAE=0.033) precisely where rare but catastrophic events occur. Only 3.3% of predictions show dangerous underprediction (> 0.05), and errors are biased toward conservative side.

5.2. *Safety-Aware Training as Design Principle*

Asymmetric loss effectiveness demonstrates a broader principle: safety-critical applications require domain-specific training objectives reflecting real-world consequences. Standard metrics like MSE assume symmetric error costs, fundamentally misaligned with safety requirements.

Our approach can be extended further. Future work could incorporate formal safety constraints directly into training through constrained optimization guaranteeing worst-case bounds on underprediction rates. Uncertainty quantification could provide confidence intervals indicating when predictions are less reliable.

5.3. *Limitations and Future Directions*

Training requires substantial data (40,000 networks) with ground-truth labels computed via exact inference. Although generalization is reasonable, optimal performance may require domain adaptation for specific applications. Neural predictions lack theoretical guarantees of traditional algorithms, necessitating additional validation for safety certification.

Future work should investigate hybrid approaches combining neural acceleration for common queries with traditional inference for rare edge cases. Formal verification techniques could provide mathematical guarantees on neural inference behavior. Online learning could enable adaptation to evolving network structures.

6. Conclusion

This paper presents a Graph Neural Network approach for BN inference specifically designed for safety-critical applications. Through safety-aware training that explicitly penalizes underprediction in low-probability regions, we achieve appropriate error characteristics for high-stakes deployment.

GAT architecture demonstrates three critical advantages: (1) Near-exact accuracy (MAE=0.047, R^2=0.938) with superior low-probability performance (MAE=0.033 for $p < 0.3$), (2) Average $8,611\times$ speedup enabling real-time operation, and (3) Perfect reliability (100%) compared to 78% failure rates in belief propagation.

These results address fundamental barriers to deploying BN in real-time safety-critical systems. Dramatic computational savings enable previously infeasible applications, while perfect reliability eliminates inference failure as a system hazard. Safety-aware error distribution ensures conservative predictions appropriate for risk assessment.

As autonomous systems proliferate in safety-critical domains, efficient and reliable probabilistic reasoning becomes essential. GNN-based inference represents significant advance toward enabling real-time risk assessment while maintaining safety standards. The key insight is that domain-specific training objectives fundamentally alter learned behavior, with broader implications for applying ML, where not all errors are equal.

Acknowledgement

This work was conducted at the University of Stuttgart.

References

Ankan, A. and A. Panda (2015). pgmpy: Probabilistic graphical models using Python. In *Proceedings of the Python in Science Conference (SciPy)*, pp. 6–11.

Fey, M. and J. E. Lenssen (2019). Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Grimstad, J. and A. Morozov (2025). Graph neural networks and fault tree analysis for runtime dynamic risk assessment of autonomous systems. In *2025 Annual Reliability and Maintainability Symposium (RAMS)*, pp. 1–6. IEEE.

Hamilton, W. L., R. Ying, and J. Leskovec (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1024–1034.

Kipf, T. N. and M. Welling (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: MIT Press.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufmann.

Scarselli, F., M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini (2008). The graph neural network model. *IEEE Transactions on Neural Networks 20*(1), 61–80.

Scutari, M. (2010). Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software 35*(3), 1–22.

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio (2018). Graph attention networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Wang, Y., S. Ermon, and M. Sachs (2020). Neural belief propagation for probabilistic inference. In *ICML Workshop on Tractable Probabilistic Modeling*.

Yoon, J., S. Arik, and T. Pfister (2018). Data-driven inference of network structures and dynamics using neural ordinary differential equations. In *NeurIPS Workshop on Integration of Deep Learning Theories*.