

GNNs for Long-Distance Relation Correction: Neurosymbolic Edge Reweighting and Loss Coupling

Usman Zafar, Ph.D.

Fourth paper of a series of seven

16 March 2026

Abstract

This paper addresses persistent failures of parsers and relation extractors on long-distance relations (LDRs) by introducing a neurosymbolic Graph Neural Network (GNN) framework that (1) learns multiplicative edge reweighting factors to amplify multi-hop supporting paths, (2) couples a differentiable neurosymbolic constraint loss to enforce linguistic invariants, and (3) provides theoretical and empirical analysis of signal amplification and stability. The method integrates with structured alignment and calibration primitives from the program’s earlier work, improves recall on LDRs while preserving precision, and includes algorithms, proofs (appendix), and a comprehensive experimental plan.

1 Introduction

Long-distance relations — semantic or syntactic links between nodes separated by many hops in a combined parse graph — are a major source of error for downstream information extraction. Standard GNN message passing attenuates signals across long paths and is sensitive to noisy intermediate structure. This paper proposes a principled remedy: learnable edge reweighting that amplifies paths supporting candidate relations, combined with a neurosymbolic arbiter loss that penalizes violations of linguistically motivated constraints. The approach is designed to be plug-in for parse+AMR graphs and to be trained end-to-end with calibration and alignment primitives.

Contributions.

1. A **learnable edge reweighting module** that outputs multiplicative factors α_e for edges, implemented as an edge-GNN or path-feature MLP.
2. A **neurosymbolic arbiter loss** \mathcal{L}_{sym} that enforces soft linguistic constraints via differentiable surrogates.
3. Theoretical analysis of **signal amplification** and stability under reweighting, including a bound on multi-hop SNR improvement.
4. Algorithms, pseudocode, and a detailed experimental plan with datasets, baselines, and ablations.
5. Full proofs and implementation notes in the appendix.

2 Related work

Graph neural networks (GNNs) are a standard tool for structured NLP tasks; early influential models include graph convolutional networks (GCNs) and graph attention networks (GATs) [1, 2]. Recent

work has focused on improving long-range information flow in graphs and on transformer-style architectures for graphs (e.g., Graphormer) that explicitly model global context and positional encodings [3]. Benchmarks and diagnostics for long-range graph reasoning (e.g., the Long Range Graph Benchmark) highlight settings where standard message passing underperforms and motivate specialized architectures or training strategies [4].

Edge-centric modeling and explicit path features have been explored as ways to capture relational evidence beyond local neighborhoods; these approaches motivate parameterizations that treat edges as first-class objects (edge-GNNs, edge co-embeddings) or that aggregate path statistics for downstream scoring. Neurosymbolic methods integrate logical constraints or symbolic priors into neural training via differentiable surrogates or constraint losses [5, 6]; such techniques provide a principled way to enforce linguistic invariants (e.g., role uniqueness) while retaining end-to-end differentiability.

Finally, practical training and stability techniques for deeper or more expressive GNNs (residual connections, normalization, careful optimization) are relevant when amplifying multi-hop signals to avoid runaway amplification of noise. Calibration and uncertainty quantification for graph models are complementary concerns when amplified scores are used for downstream decision thresholds.

3 Problem formulation

Let a sentence produce a heterogeneous graph $G = (V, E)$ combining UD edges, AMR relations, and lexical adjacency. Each node $v \in V$ has an initial feature vector x_v (contextual embedding, POS, lemma). Each edge $e = (u \rightarrow v) \in E$ has an initial weight $w_e^0 > 0$ derived from relation type and lexical signals.

The goal is to predict a set of target relations \mathcal{R} (typed edges) between node pairs, with particular focus on pairs whose shortest path length in G exceeds a threshold h_{LDR} (long-distance relations).

4 Model

4.1 Edge reweighting module

A learnable reweighting function r_θ maps edge features and local path statistics to a multiplicative factor:

$$\alpha_e = r_\theta(\text{feat}(e), \text{path_stats}(e; G)), \quad \alpha_e \geq 0.$$

The reweighted adjacency is $w_e = \alpha_e \cdot w_e^0$. Practical parameterizations include:

- **Edge-GNN:** a small GNN operating on edge features (treating edges as nodes in a line graph).
- **Path-MLP:** an MLP taking aggregated path counts, max/min similarity along short supporting paths, and relation type encodings.

4.2 GNN with reweighted adjacency

A standard message-passing GNN is applied on the reweighted graph:

$$h_v^{(t+1)} = \sigma \left(W^{(t)} h_v^{(t)} + \sum_{u \in \mathcal{N}(v)} w_{(u,v)} \cdot M^{(t)}(h_u^{(t)}, h_v^{(t)}, e_{(u,v)}) \right),$$

where $h_v^{(0)} = x_v$, $M^{(t)}$ is a message function, and $w_{(u,v)} = \alpha_{(u,v)} w_{(u,v)}^0$.

4.3 Relation prediction head

For a candidate pair (i, j) , a relation score is computed from final node representations:

$$s_{ij} = \text{MLP}_\phi([h_i^{(T)}; h_j^{(T)}; \psi_{\text{pair}}(i, j)]),$$

and relation probabilities follow via sigmoid/softmax depending on multi-label or multi-class setup.

4.4 Neurosymbolic arbiter loss

Define a set of symbolic constraints \mathcal{S} (e.g., “a predicate has at most one ARG0 within the same clause”). For each constraint $s \in \mathcal{S}$, define a differentiable violation surrogate $\text{viol}_s(\hat{\mathcal{R}})$ computed from soft predictions $p_{ij} = \sigma(s_{ij})$. The neurosymbolic loss is:

$$\mathcal{L}_{\text{sym}} = \sum_{s \in \mathcal{S}} \lambda_s \cdot \text{viol}_s(\{p_{ij}\}).$$

Example: for a uniqueness constraint on role r for predicate p , use hinge surrogate:

$$\text{viol}_{\text{uniq}}(p) = \max\left(0, \sum_j p_{p,j}^{(r)} - 1\right).$$

4.5 Full training objective

The model is trained to minimize:

$$\mathcal{L} = \mathcal{L}_{\text{sup}}(\{p_{ij}\}, \mathcal{R}^{\text{gold}}) + \eta \mathcal{L}_{\text{sym}} + \rho \mathcal{L}_{\text{reg}}(\alpha),$$

where \mathcal{L}_{sup} is a supervised cross-entropy or binary cross-entropy loss, and $\mathcal{L}_{\text{reg}}(\alpha)$ regularizes reweighting factors (e.g., ℓ_2 penalty or KL to 1) to avoid degenerate amplification.

5 Theoretical analysis

This section provides a simplified analysis of how multiplicative reweighting amplifies multi-hop supporting signals and affects signal-to-noise ratio (SNR) for LDR candidates.

Definition 5.1 (Supporting path). A path $P = (v_0, \dots, v_L)$ is said to *support* a candidate relation between v_0 and v_L if intermediate edges and nodes provide lexical or structural evidence (e.g., predicate–argument cues). Let the path score be $S(P) = \prod_{t=1}^L w_{(v_{t-1}, v_t)}$.

Proposition 5.2 (Multiplicative amplification of path scores). *Let P be a supporting path of length L . Under reweighting $w_e = \alpha_e w_e^0$, the path score becomes*

$$S(P) = \left(\prod_{t=1}^L \alpha_{e_t} \right) \cdot S_0(P),$$

where $S_0(P)$ is the original path score under w_e^0 .

Proof. Immediate from multiplicative definition: $S(P) = \prod_{t=1}^L w_{e_t} = \prod_{t=1}^L (\alpha_{e_t} w_{e_t}^0) = (\prod_{t=1}^L \alpha_{e_t}) \cdot (\prod_{t=1}^L w_{e_t}^0)$. \square

Theorem 5.3 (SNR improvement bound). *Consider a candidate relation between nodes i, j with a set of supporting paths \mathcal{P}_{ij} and a set of non-supporting (noise) paths \mathcal{N}_{ij} . Let the aggregated supporting signal be $S_{\text{sup}} = \sum_{P \in \mathcal{P}_{ij}} S(P)$ and noise $S_{\text{noise}} = \sum_{Q \in \mathcal{N}_{ij}} S(Q)$. Suppose reweighting factors satisfy $\alpha_e \in [\underline{\alpha}, \bar{\alpha}]$ with $\underline{\alpha} > 0$. Then the ratio*

$$\frac{S_{\text{sup}}}{S_{\text{noise}}}$$

is multiplied by at least $(\underline{\alpha}/\bar{\alpha})^{L_{\text{max}}}$ relative to the unweighted ratio, where L_{max} is the maximum path length considered.

Sketch. Each supporting path score is multiplied by the product of α along its edges; similarly for noise paths. The worst-case relative amplification between any supporting path of length $\leq L_{\text{max}}$ and any noise path is at least $(\underline{\alpha}/\bar{\alpha})^{L_{\text{max}}}$. Summing across paths yields the bound on aggregated ratio. \square

Implication. If the reweighting module can assign systematically larger α to edges on supporting paths than to noisy edges, the SNR for LDR candidates increases exponentially in path length, improving detectability.

6 Algorithms

6.1 Training loop (high level)

Algorithm 1 Edge Reweighting and Neurosymbolic Training

Require: Graphs $\{G_i\}$, gold relations $\{\mathcal{R}_i\}$, hyperparameters

- 1: Initialize encoder, edge reweighter r_θ , GNN, relation head
 - 2: **for** epoch = 1 to E **do**
 - 3: **for** batch **do**
 - 4: Compute initial edge features and path statistics
 - 5: Compute $\alpha_e = r_\theta(\cdot)$, set $w_e = \alpha_e w_e^0$
 - 6: Run GNN on reweighted graph to obtain node embeddings
 - 7: Compute relation scores s_{ij} and probabilities p_{ij}
 - 8: Compute $\mathcal{L} = \mathcal{L}_{\text{sup}} + \eta \mathcal{L}_{\text{sym}} + \rho \mathcal{L}_{\text{reg}}$
 - 9: Update parameters by gradient descent
 - 10: **end for**
 - 11: **end for**
-

6.2 Practical implementation notes

- **Stability:** constrain α_e via softplus or sigmoid scaling to keep values in a controlled range; use ℓ_2 or KL regularization toward 1.
- **Efficiency:** compute path statistics with bounded hop depth L_{max} and sample a subset of paths for large graphs.
- **Calibration:** integrate structured calibration (P2) to ensure amplified scores remain well-calibrated for downstream thresholds.
- **Optimization:** use residual connections, layer normalization, and careful learning-rate schedules when training deep message-passing stacks to avoid vanishing/exploding signals.

7 Experimental plan

The experimental plan focuses on demonstrating improved LDR recall, controlled precision, and downstream extraction gains.

7.1 Datasets

- **UD treebanks (English EWT)** combined with **AMR 3.0** parses for sentence-level graphs.
- **Long-distance relation subsets:** construct subsets by selecting gold relations whose shortest path length in the combined graph exceeds $h_{\text{LDR}} = 4$.
- **Relation extraction benchmarks** with annotated long-range phenomena (custom splits).
- **Small adversarial suite** (from P1/P6 pipeline) containing syntactic obfuscations and null-node cases.

7.2 Baselines

- Standard GNN without reweighting.
- GNN with static heuristic reweighting (e.g., inverse distance).
- Path-based classifier using handcrafted path features.
- Transformer-only baseline (no explicit graph).

7.3 Metrics

- **Edge recall/precision/F1** for long-distance relations (primary).
- **Downstream extraction F1** on tasks using predicted relations.
- **Constraint violation rate** (how often symbolic constraints are breached).
- **Calibration metrics** (structured Brier score, sECE) for predicted relation confidences.
- **Efficiency:** runtime and memory per sentence.

7.4 Ablations

1. Learned reweighting vs no reweighting vs heuristic reweighting.
2. With and without neurosymbolic loss \mathcal{L}_{sym} .
3. Regularization strength ρ sweep.
4. Path statistic features ablation (counts, max similarity, type signals).
5. Integration with calibration (P2) vs no calibration.

7.5 Reproducibility checklist

- Release code for edge reweighter and neurosymbolic loss with unit tests.
- Provide scripts to construct long-distance subsets from UD+AMR.
- Include seed and hyperparameter settings for reported runs.

8 Discussion

The proposed neurosymbolic reweighting framework offers a principled mechanism to counteract message attenuation in GNNs for long paths while preserving interpretability through explicit constraint losses. The multiplicative nature of reweighting yields exponential gains in path scores when supporting edges are consistently upweighted, but it requires careful regularization and

calibration to avoid amplifying noise. Integration with structured calibration (P2) and alignment primitives (P1) is essential to produce trustworthy, high-confidence relation predictions suitable for downstream decision making.

9 Conclusion

This paper introduced a neurosymbolic GNN approach for long-distance relation correction that learns edge reweighting factors and couples them with differentiable symbolic constraints. Theoretical bounds, algorithms, and a focused experimental plan are provided to validate the approach. The methods are designed to integrate with the broader program of work on syntactic–semantic governance and robustness.

Acknowledgements

The author acknowledges the assistance of AI tools in drafting and implementation. This paper is the fourth in a planned series of seven papers exploring syntactic–semantic governance, adversarial resilience, and legal-grade provenance.

References

- [1] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- [2] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *ICLR*, 2018.
- [3] R. Ying, D. Cai, S. Zhang, Y. Zhang, J. Huang, and J. Leskovec. Do transformers really perform badly for graph representation? In *NeurIPS*, 2021.
- [4] V. P. Dwivedi, A. Bresson, and others. Long Range Graph Benchmark. In *NeurIPS Datasets and Benchmarks Track*, 2022.
- [5] Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. P. Xing. Harnessing deep neural networks with logic rules. In *ACL*, 2016.
- [6] M. Diligenti, G. Roychowdhury, and M. Gori. Semantic constraints in learning. In *AAAI*, 2017.

A Appendix A: Proofs and technical lemmas

A.1 Proof of Theorem 5.3 (detailed sketch)

Proof. Let \mathcal{P}_{ij} denote supporting paths and \mathcal{N}_{ij} denote noise paths. Under reweighting, each path P of length L_P is multiplied by $\prod_{e \in P} \alpha_e$. Let $\underline{\alpha} = \min_e \alpha_e$ and $\bar{\alpha} = \max_e \alpha_e$. Then for any supporting path P with length $\leq L_{\max}$,

$$\prod_{e \in P} \alpha_e \geq \underline{\alpha}^{L_P} \geq \underline{\alpha}^{L_{\max}}.$$

Similarly, for any noise path Q ,

$$\prod_{e \in Q} \alpha_e \leq \bar{\alpha}^{L_Q} \leq \bar{\alpha}^{L_{\max}}.$$

Therefore each supporting path score is amplified by at least $\underline{\alpha}^{L_{\max}}$ and each noise path by at most $\overline{\alpha}^{L_{\max}}$. Aggregating across paths yields the stated multiplicative lower bound on the SNR ratio. \square

B Appendix B: Implementation notes and hyperparameters

- Default α parameterization: $\alpha_e = 1 + \text{softplus}(g_\theta(\cdot))$ clipped to $[0.1, 10]$.
- Regularization: $\mathcal{L}_{\text{reg}}(\alpha) = \lambda_\alpha \sum_e (\log \alpha_e)^2$ with $\lambda_\alpha \in [10^{-3}, 10^{-1}]$.
- Optimizer: AdamW with learning rate 1e−4 for encoders; 1e−3 for edge reweighter.
- Path depth $L_{\max} = 6$ for path statistics; sample up to 50 paths per candidate pair.