

Using Graph Neural Networks and the Internet Graph Topology to Learn Autonomous System Relationships

Valentina Esteban^{1,2,*}, Sebastián Ferrada^{1,3,4,5} and Ivana Bachmann^{1,2}

¹DCC, Universidad de Chile, Santiago, Chile

²NIC Chile Research Labs, Santiago, Chile

¹IDIA, Universidad de Chile, Santiago, Chile

⁴Millennium Institute for Foundational Research on Data (IMFD), Santiago, Chile

⁵National Center for Artificial Intelligence Research (CENIA), Santiago, Chile

Abstract

The Internet is built upon thousands of interconnected Autonomous Systems (ASes) that exchange routing information through the Border Gateway Protocol (BGP). The commercial relationships between ASes classified as peer-to-peer (P2P), customer-to-provider (C2P), or provider-to-customer (P2C), shape the paths that packets follow across the network. However, this information is rarely disclosed publicly and must be inferred from indirect sources. Accurate inference of AS relationships is essential for analyzing the Internet structure, understanding routing policies, and detecting anomalies. In this work, we propose using Graph Neural Networks (GNNs) as a novel approach to this inference task. We construct an AS-level Internet graph with 79K nodes and 438K edges from BGP Routing Information Bases (RIBs) collected via RouteViews and RIPE RIS, enriching node representations with attributes from PeeringDB and one-hot encoding. We present a GNN approach that evaluates two GNN architectures (GCN and GraphSAGE) end-to-end, with the GNN and an MLP decoder jointly trained for relationship classification. We compare against established graph embedding baselines, DeepWalk and BGP2Vec. The best model, a GCN + MLP architecture with PeeringDB attributes, achieves 0.952 accuracy and 0.950 macro-F1, surpassing BGP2Vec in macro-F1 (0.946) while requiring half the training time (10 vs 20 min), suggesting that GNNs offer a scalable and structurally motivated alternative for AS relationship inference.

1. Introduction

The Internet is composed of thousands of interconnected Autonomous Systems (ASes) that exchange routing information through the Border Gateway Protocol (BGP). An AS is an administratively independent network operated by entities such as Internet service providers, enterprises, universities, or content networks. BGP is the interdomain routing protocol that allows these networks to advertise reachable IP prefixes and learn paths to external destinations. More importantly, BGP is a policy-based protocol: route selection and propagation are determined not solely by shortest-path criteria but by routing policies defined by each AS. These policies determine which routes are accepted, preferred, and exported to neighboring ASes, and are strongly influenced by commercial agreements; primarily *customer-to-provider* (C2P) and *peer-to-peer* (P2P) relationships.

The BGP propagates routing information across the Internet through route announcements and withdrawals. As routes are exchanged, routers maintain Routing Information Bases (RIBs), which store the paths (`AS_PATH`) learned from neighboring ASes. The `AS_PATH` records the sequence of ASes followed before reaching a given AS. This attribute provides an observable view of interdomain routing and enables the reconstruction of an approximation of the AS-level Internet topology.

Because interdomain routing is policy-driven, understanding AS relationships is essential for studying the structure and behavior of the Internet [1, 2]. These relationships affect how routes are exported, which paths become visible in BGP, and ultimately how traffic flows across the network. However,

AMW 2026: 17th Alberto Mendelzon International Workshop on Foundations of Data Management, November 9th–November 13th, 2026, Arequipa, Peru

*Corresponding author.

✉ valentina.esteban@ug.uchile.cl (V. Esteban)

🆔 0009-0004-3279-8368 (V. Esteban); 0000-0002-9421-8566 (S. Ferrada); 0000-0002-3189-1796 (I. Bachmann)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

the commercial agreements behind these relationships are rarely disclosed publicly. As a result, the problem of classifying each AS link as customer-to-provider (C2P), provider-to-customer (P2C), or peer-to-peer (P2P), has become a long-standing research topic in Internet topology analysis.

Accurate AS relationship inference is useful in several contexts. It helps characterize the economic and hierarchical organization of the Internet, improves topology modeling and simulation, and supports the analysis of resilience, dependency, and routing anomalies. A more realistic view of AS relationships also improves interpretations of observed BGP behavior, since many routing decisions are constrained by underlying business policies rather than shortest-path considerations alone.

2. Dataset and Graph Construction

To build the AS-level topology, we collected public BGP routing data from RouteViews [3] and RIPE RIS [4]. We use BGP RIBs as our primary data source because `AS_PATHS` are the core input used by existing inference methods, including heuristic [1, 5, 6], probabilistic [7, 8], and learning-based approaches [9]. As for the AS relationships, we use data available from the Center for Applied Internet Data Analysis or CAIDA [10], which provides inferred relationships regarding AS-level network. These inferred relationships are the only available resources, since relations between ASes are typically undisclosed.

We collected RIBs over a 24-hour period on March 1st, 2026, and observed that after that period, the changes in the collected data diminished significantly. After this period, we obtained 27,456,277 BGP AS paths, from which we constructed a graph with 79,504 AS nodes and 438,520 edges that approximates the observable AS-level Internet topology. The limited number and geographic distribution of BGP collectors mean that some AS links may not be visible from any available vantage point [11, 12, 13].

We enrich the node representations with structural features derived from the AS-level graph such as *degree*, *betweenness*, and *PageRank*, as well as PeeringDB attributes such as the number of facilities the AS operates in (*fac_count*), the number of IPv4 prefixes announced (*info_prefixes4*), the general peering policy (*policy_general*), estimated traffic level (*info_traffic*), and network type (*info_type*) [14, 15].

Here, we distinguish two types of PeeringDB features: (1) Categorical attributes such as *info_traffic* or *info_type*, which are converted to one-hot-encoded vectors for compatibility with neural network processing, and (2) Numerical attributes such as *fac_count* or *info_prefixes4*, which exhibit high variability and long-tail distributions. Thus, we first apply a logarithmic transformation to compress the range, followed by Min-Max normalization to scale values to [0, 1].

Edges present in our BGP-derived graph that are not present in CAIDA and, thus, have no assigned relation (about 10% of all edges), are excluded from training, validation, and test sets. For the remaining labeled edges, we augmented the dataset by adding the inverse relationship for each directed edge (e.g., adding P2C for each C2P), ensuring that both directions of every link are represented.

The resulting labeled dataset exhibits class imbalance: P2P relationships account for 46.3% of edges, while C2P and P2C each account for 21.78%, with the remaining 10.07% corresponding to edges with an unknown relationship type (absent from CAIDA). To account for this imbalance, we use stratified splits for training, validation, and test sets to ensure proportional representation of all three classes.

3. GNNs for AS Relationship Inference

We define the Internet AS-level topology as a graph $G = (V, E)$, where V is the set of ASes and E is the set of adjacencies observed in BGP RIBs. Each node $v \in V$ is represented by a vector $\mathbf{x}_v \in \mathbb{R}^d$ encoding PeeringDB attributes and structural metrics. Thus, the AS relationship inference is formulated as a *directed edge classification* problem. Each edge $(u, v) \in E$ is assigned a label $y_{uv} \in \{\text{C2P}, \text{P2C}, \text{P2P}\}$, corresponding to the three commercial relationship types. The goal is to learn a mapping function

$$\min_{\theta} \sum_{(u,v) \in E_{\text{train}}} \mathcal{L}(f_{\theta}(u, v), y_{uv})$$

that minimizes the classification loss over the set of labeled training edges E_{train} .

Table 1

Performance metrics across all experimental configurations. Best per column is in bold; second-best is underlined.

Approach / Case	Model Arch.	Acc.	Precision			M-Prec.	Recall			M-Rec.	M-F1	Train Time
			P2P	C2P	P2C		P2P	C2P	P2C			
End-to-End Training												
Case E0	GraphSAGE + MLP	0.944	0.956	0.927	0.927	0.938	<u>0.971</u>	0.915	0.916	0.934	0.934	40 min
Case E1	GraphSAGE + MLP	0.813	0.678	0.933	0.931	0.844	0.962	0.736	0.734	0.813	0.823	15 min
Case E2	GCN + MLP	<u>0.952</u>	0.965	<u>0.946</u>	0.945	0.952	0.951	0.952	<u>0.954</u>	<u>0.952</u>	0.950	10 min
Case E3	GCN + MLP	0.947	0.948	0.943	<u>0.949</u>	<u>0.947</u>	0.907	<u>0.956</u>	0.952	0.938	0.941	2 min
Case E4	GraphSAGE + MLP	0.853	0.716	0.968	0.971	0.885	0.984	0.791	0.785	0.853	0.811	20 min
Baselines												
BGP2Vec [9]	ANN	0.960	0.990	0.906	0.900	0.932	0.959	0.960	0.964	0.961	<u>0.946</u>	20 min
DeepWalk [18]	ANN	<u>0.952</u>	<u>0.988</u>	0.887	0.882	0.919	0.951	0.950	<u>0.954</u>	<u>0.952</u>	0.935	45 min

As a training strategy, we use **End-to-End Training**, which jointly optimizes the encoder and decoder in a single training loop, using the relationship classification loss as the sole objective. The encoder produces node embeddings that are passed directly to an edge decoder, and backpropagation updates all components simultaneously, allowing the model to learn representations optimized for the target classification task.

For the encoder architectures, we use a novel approach using GNNs. Here, we evaluate two GNNs, each implemented as a two-layer message-passing network with ReLU activations and dropout for regularization: (1) Graph Convolutional Network (GCN) [16], and (2) GraphSAGE [17].

As for the decoder, which transforms the node embeddings $\mathbf{h}_u$ and $\mathbf{h}_v$ of a candidate edge (u, v) into a class prediction, we use a Multilayer Perceptron (MLP) Decoder for each case.

Finally, we compare our GNN-based models against two graph embedding baselines (BGP2Vec [9] and DeepWalk [18]), both paired with an ANN classifier trained on their output embeddings.

4. Experimental Setup

We define five *end-to-end* configurations (Cases E0–E4). Case E0 corresponds to the reference configuration that uses PeeringDB attributes across the full graph. Case E1 replaces PeeringDB attributes with centrality features only (degree, betweenness, and PageRank) to assess whether a lighter, topology-derived feature set can achieve competitive performance. Cases E2 and E3 evaluate mini-batch training alternatives under the same PeeringDB feature set as E0, using random neighbor sampling and ClusterGCN community-based [19] partitioning, respectively, to assess whether scalable training strategies preserve model performance. Case E4 is a structural control in which all node features are replaced by random vectors of the same dimensionality, isolating what the GNN learns from graph topology alone and quantifying the contribution of node attributes to model performance.

All GNN models were trained using the Adam optimizer with an initial learning rate of 0.001, for a maximum of 300 epochs (50 epochs for the mini-batch configurations E2 and E3), with early stopping if the validation loss did not improve for 20 consecutive epochs. To mitigate the effect of class imbalance, the cross-entropy loss was weighted by class frequency. All architectures use 2 GNN layers, with hidden embedding dimensions of 128 or 256 and output embedding dimensions of 64 or 128, a dropout rate of 0.3, and ReLU activation. For GraphSAGE, we use mean aggregation with a random neighbor sample size of 15. The MLP decoder consists of 2 fully-connected layers.

Labeled edges are partitioned into training, validation, and test sets using a **70/15/15%** ratio; reciprocal edges are always assigned to the same partition to prevent information leakage. The full graph is visible during training, but the loss is computed only over training-set edges (in a transductive setting).

5. Results and Discussion

Table 1 reports performance across all configurations, using the best-performing encoder per case.

Comparing E0 (PeeringDB, 0.944), E1 (centrality only, 0.813), and E4 (random features, 0.853) shows that random features outperform hand-crafted centrality by 4 points in accuracy. This is likely because centrality metrics are redundant with what GNNs already learn via message passing, while random features break the symmetry between structurally identical nodes, giving the model a discriminative signal that centrality alone can not [20]. We must note that this is not the case for macro-F1, however, where E1 (0.823) surpasses E4 (0.811) by one point. The gap in accuracy between E4 (0.853) and E0 (0.944) confirms that PeeringDB attributes add information that topology cannot recover.

The MLP decoder, which preserves edge order through concatenation, yields balanced per-class precision (P2P: 0.956, C2P: 0.927, P2C: 0.927) despite class imbalance (P2P: 46.3% of edges), confirming the value of a direction-aware decoder under end-to-end supervision.

Mini-batch training (E2: random sampling; E3: ClusterGCN) outperforms full-graph training (E0: 0.944 accuracy / 0.934 macro-F1, 40 min) in both accuracy and speed: E2 reaches 0.952 / 0.950 in 10 min, E3 0.947 / 0.941 in 2 min. We attribute this to sampling’s implicit regularization against high-degree nodes (e.g., Tier-1 providers), showing that GNN training scales without sacrificing accuracy.

GraphSAGE leads under full-graph training (E0: 0.944), while GCN leads under sampling (E2: 0.952, E3: 0.947), suggesting that mini-batch sampling itself mitigates the degree-distribution challenge that otherwise favors GraphSAGE’s fixed-size neighborhoods. This trade-off matters beyond accuracy: GraphSAGE is *inductive*, generating embeddings for unseen nodes without retraining, whereas GCN, DeepWalk, and BGP2Vec are effectively *transductive* and require recomputation over the full graph when new ASes appear. Since the Internet’s topology evolves continuously, GraphSAGE (E0) is a more deployment-friendly choice despite its 0.008 accuracy gap relative to the GCN-based E2.

Against baselines, BGP2Vec leads in accuracy (0.960, F1: 0.946), but our best configuration E2 surpasses it in macro-F1 (0.950) at half the training time (10 vs. 20 min); given the class imbalance in our data, macro-F1 is the more informative metric, making E2 the strongest overall result. DeepWalk matches E2 in accuracy (0.952) but trails in macro-F1 (0.935) and takes 45 min to train, suggesting that random walks alone capture most of the predictive structural signal, while BGP2Vec’s routing semantics offer some added benefit for the harder C2P/P2C distinction.

6. Conclusions

This paper studied the use of Graph Neural Networks for AS relationship inference, using end-to-end training across two GNN architectures and evaluating the contribution of PeeringDB node attributes relative to topology-only and random-feature baselines. We constructed and publicly released an AS-level graph of 79,504 nodes and 438,520 edges, combining BGP routing data with PeeringDB attributes and CAIDA relationship labels, together with the full preprocessing and training pipeline.

We found that graph topology is the dominant signal for this task: even with random node features, GNNs achieve 0.853 accuracy by exploiting structural patterns alone. PeeringDB attributes provide complementary information that topology cannot recover, pushing end-to-end GraphSAGE + MLP to 0.944 accuracy on the full graph. With random neighbor sampling (E2), GCN reaches 0.952 accuracy and 0.950 macro-F1 in 10 min, surpassing BGP2Vec in macro-F1 (0.946) at half the training time and without BGP-specific sequence modeling. ClusterGCN partitioning (E3) exceeds full-graph accuracy (0.947 vs. 0.944) in just 2 min, enabling deployment on graphs where full-batch training is infeasible. GraphSAGE’s inductive design also offers a practical edge over transductive alternatives, generating embeddings for new ASes without retraining — a property worth evaluating directly in future work given the Internet’s continuously evolving topology.

All code, datasets, and trained model checkpoints are publicly available in a GitHub repository: <https://github.com/valesteban/gnn-as-relationships>. The repository includes scripts for BGP data extraction, graph construction, PeeringDB attribute enrichment, and all experimental configurations.

Acknowledgments

Ferrada was funded by ANID - Millennium Science Initiative Program - Code ICN17_00; by CENIA FB210017, Basal ANID; by FONDECYT Grant 11251322; and by the Vicerrectoría de Investigación y Desarrollo (VID) of Universidad de Chile, project code: UI-016/24.

Declaration on Generative AI

The authors used Claude Opus 5 to assist with writing. All generated outputs were reviewed and validated by the authors, who take full responsibility for the final content.

References

- [1] L. Gao, On inferring autonomous system relationships in the internet, *IEEE/ACM Trans. Netw.* 9 (2001) 733–745. URL: <https://doi.org/10.1109/90.974527>. doi:10.1109/90.974527.
- [2] L. Subramanian, S. Agarwal, J. Rexford, R. Katz, Characterizing the internet hierarchy from multiple vantage points, in: *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, 2002, pp. 618–627 vol.2. doi:10.1109/INFCOM.2002.1019307.
- [3] University of Oregon RouteViews Project, University of oregon routeviews project, <https://www.routeviews.org/routeviews/>, 2026. Accessed: 2026-07-31.
- [4] RIPE NCC, Ripe ncc routing information service (ris), <https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/>, 2026. Accessed: 2026-07-31.
- [5] L. Ruan, J. S. Varghese, Computing observed autonomous system relationships in the internet, 2014.
- [6] M. Luckie, B. Huffaker, A. Dhamdhere, V. Giotsas, k. claffy, As relationships, customer cones, and validation, in: *Proceedings of the 2013 Conference on Internet Measurement Conference, IMC '13*, Association for Computing Machinery, New York, NY, USA, 2013, pp. 243–256. URL: <https://doi.org/10.1145/2504730.2504735>. doi:10.1145/2504730.2504735.
- [7] Y. Jin, C. Scott, A. Dhamdhere, V. Giotsas, A. Krishnamurthy, S. Shenker, Stable and practical as relationship inference with problink, in: *Proceedings of the 16th USENIX Conference on Networked Systems Design and Implementation, NSDI'19*, USENIX Association, USA, 2019, pp. 581–597.
- [8] Z. Jin, X. Shi, Y. Yang, X. Yin, Z. Wang, J. Wu, Toposcope: Recover as relationships from fragmentary observations, in: *Proceedings of the ACM Internet Measurement Conference, IMC '20*, Association for Computing Machinery, New York, NY, USA, 2020, pp. 266–280. URL: <https://doi.org/10.1145/3419394.3423627>. doi:10.1145/3419394.3423627.
- [9] T. Shapira, Y. Shavitt, Unveiling the type of relationship between autonomous systems using deep learning, in: *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–6. doi:10.1109/NOMS47738.2020.9110358.
- [10] Center for Applied Internet Data Analysis, CAIDA, 2026. URL: <https://www.caida.org/>.
- [11] R. V. Oliveira, D. Pei, W. Willinger, B. Zhang, L. Zhang, The (In)completeness of the observed Internet AS-level structure, *IEEE/ACM Transactions on Networking* 18 (2010) 109–122. doi:10.1109/TNET.2009.2020798.
- [12] E. Gregori, A. Improta, L. Lenzini, L. Rossi, L. Sani, On the incompleteness of the AS-level graph: A novel methodology for BGP route collector placement, in: *Proceedings of ACM IMC, 2012*, pp. 253–264. doi:10.1145/2398776.2398803.
- [13] T. Alfroy, T. Holterbach, T. Krenc, K. Claffy, C. Pelsser, Measuring Internet routing from the most valuable points, 2024. arXiv:2405.13172.
- [14] PeeringDB, The interconnection database, 2022. URL: <https://www.peeringdb.com/>.
- [15] D. P. Giakatos, S. Kostoglou, P. Sermpezis, A. Vakali, Benchmarking graph neural networks

- for internet routing data, in: Proceedings of the 1st International Workshop on Graph Neural Networking, GNNet '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 1–6. URL: <https://doi.org/10.1145/3565473.3569187>. doi:10.1145/3565473.3569187.
- [16] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907 (2016).
 - [17] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems, volume 30, 2017.
 - [18] B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710. doi:10.1145/2623330.2623732.
 - [19] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, C.-J. Hsieh, Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 257–266. URL: <https://doi.org/10.1145/3292500.3330925>. doi:10.1145/3292500.3330925.
 - [20] C. T. Duong, T. D. Hoang, H. T. H. Dang, Q. V. H. Nguyen, K. Aberer, On node features for graph neural networks, 2019. URL: <https://arxiv.org/abs/1911.08795>. arXiv:1911.08795.