Project Proposal: Applying Graph Neural Networks in Molecular Property Prediction with PyTorch Geometric

Nikita Zagainov n.zagainov@innopolis.university
Dmitry Tetkin d.tetkin@innopolis.university
Nikita Tsukanov n.tsukanov@innopolis.university

November 1, 2025

Summary

We propose to develop a tutorial/case study on applying modern graph neural networks (GNNs) to molecular property regression in computational chemistry and drug discovery, using PyTorch Geometric (PyG). The tutorial will be step-by-step and self-contained for readers who know PyTorch but are new to graph ML, with clear code snippets, extensive visualizations, and Google Colab notebooks for full reproducibility.

1 Application domain

Computational chemistry and drug discovery. Molecules are naturally graphs: atoms as nodes and chemical bonds as edges. Predicting molecular properties is central to virtual screening and lead optimization.

2 Which dataset are you planning to use?

Primary: ZINC constrained solubility regression (logP), as curated by the GNN benchmarking suite (see Benchmarking GNNs); available directly in PyG as `ZINC` (PyG ZINC).
Fallback/Optional extension: QM9 quantum chemistry dataset (PyG QM9; Sci. Data 2014).

3 Describe the dataset, prediction tasks, and metric

Graph schema.

- **Nodes:** atoms, with categorical features such as element type, formal charge; optionally degree, aromaticity flags.
- **Edges:** chemical bonds, with bond type (single/double/triple/aromatic) as categorical edge features; optionally conjugation and ring membership.
- **Labels:** graph-level scalar property.

Task: supervised *graph-level regression.*
Metric: mean absolute error (MAE) on the target property.
ZINC details: *Constrained solubility* (logP) regression; small molecules (up to 28 atoms) curated for benchmarking (Benchmarking GNNs). Standard splits: 12k train / 1k val / 1k test with MAE as the official metric (PyG ZINC; paper).
QM9 details (optional): Predict one or more quantum-mechanical properties (e.g., dipole moment, HOMO/LUMO, atomization energy). Standard MAE/% errors per-target (PyG QM9; dataset paper).

4 Why did you choose the dataset?

- **Small graphs, fast training:** molecules are typically ¡ 30 nodes in ZINC, enabling rapid iteration even on modest GPUs.
- **Excellent PyG support:** convenient loaders and baselines reduce boilerplate (PyG docs).
- **Standardized splits and benchmarks:** ensures fair comparison and reproducibility (Benchmarking GNNs; benchmarking-gnns repo).
- **Pedagogical clarity:** molecular graphs match the strengths of message passing, making concepts intuitive to newcomers.

5 Graph ML technique that you want to apply

Message Passing Neural Networks (MPNNs) with **edge-aware updates**. We will start with Graph Isomorphism Network (GIN; paper) and its edge-feature-aware variant GINE (available as `GINEConv` in PyG; docs). Pooling will use global add pooling, optionally with a virtual node.

6 Graph ML model you plan to use

Backbone: Stacked `GINEConv` layers with MLP message/update functions, each followed by BatchNorm, ReLU, and dropout.
Readout: Global add pooling to obtain a graph embedding, then a small MLP for scalar prediction.
Training: L2 loss (MSE), AdamW optimizer, cosine learning-rate schedule with warmup; early stopping on validation MSE.
Baselines: (i) RDKit descriptors + XGBoost (RDKit; XGBoost), (ii) an MLP on simple atom-type histograms.
Stretch (time-permitting): virtual node, Stochastic Weight Averaging (SWA), and scaffold split robustness analysis.

7 Describe the model

GIN/GINE (edge-aware MPNN). We will use Graph Isomorphism Network (GIN; paper) and its edge-aware variant GINE (PyG GINEConv). Intuitively, each layer updates an atom by aggregating representations from its neighbors; GINE augments this with bond-type information so messages depend on both neighboring atoms and the connecting bond.

Graph Attention (GAT/GATv2). As a complementary model, we will evaluate attention-based layers that learn importance weights over neighbors: GAT (paper, PyG GATConv) and GATv2 (paper, PyG GATv2Conv). For molecules, edge features can modulate attention or messages, e.g., as in AttentiveFP (paper).

Readout and training. We use a permutation-invariant global add pooling to obtain a molecule-level embedding, followed by a small MLP for scalar prediction. Training minimizes mean squared error (MSE) with AdamW and cosine learning-rate scheduling; early stopping monitors validation MSE. For reporting and comparison to benchmarks, we will evaluate using mean absolute error (MAE), as specified in the dataset protocols.
Planned architecture: Input node/edge embeddings -¿ GINEConv + BatchNorm + ReLU + Dropout (repeat) -¿ Global Add Pooling -¿ MLP Readout -¿ Scalar prediction.
Hyperparameters (to be tuned): layers 3/5/7; hidden size 64/128/256; dropout 0.0–0.5; learning rate 1e-4–3e-3.

8 Why the model is appropriate for the dataset

- **Expressivity:** GIN matches the Weisfeiler–Lehman (1-WL) test in discriminative power (Xu et al., 2019), aligning with motif/substructure sensitivity needed for molecular properties.
- **Edge features:** GINE directly incorporates bond types and related chemistry, crucial for properties like solubility and electronic energies.
- **Data regime:** For small graphs and moderate dataset sizes like ZINC, lightweight MPNNs with global pooling are strong and efficient baselines.
- **Simplicity and reproducibility:** Popular, well-supported layers in PyG minimize engineering complexity while providing competitive performance.

Links and references

- **PyTorch Geometric:** docs
- **ZINC in PyG:** dataset **Benchmarking GNNs:** paper code
- **QM9 in PyG:** dataset **QM9 paper:** Sci. Data 2014
- **GIN (How Powerful Are GNNs?):** paper
- **GAT:** paper PyG GATConv
- **GATv2:** paper PyG GATv2Conv
- **AttentiveFP (edge-aware attention for molecules):** paper
- **GINEConv in PyG:** docs
- **Papers With Code (SOTA browser):** link **OGB Leaderboards:** link
- **RDKit:** link **XGBoost:** link

Key references

- K. Xu, W. Hu, J. Leskovec, S. Jegelka, "How Powerful Are Graph Neural Networks?" ICLR 2019. arXiv:1810.00826.
- P. Veličković et al., "Graph Attention Networks." ICLR 2018. arXiv:1710.10903.
- B. Brody, U. Alon, E. Yahav, "How Attentive are Graph Attention Networks?" ICLR 2022. arXiv:2105.14491.
- S. Xiong et al., "Pushing the Boundaries of Molecular Representation for Drug Discovery with the Graph Attention Mechanism." (AttentiveFP) arXiv:1904.01279.
- V. Dwivedi et al., "Benchmarking Graph Neural Networks." arXiv:2003.00982.

Group size: 1–3 students. We will focus on well-established, classical graph ML methods and avoid research-heavy or novel architectures, prioritizing clarity and reproducibility.