Graph Neural Networks for Molecular Solubility Prediction on ESOL

Written Report for GNNs Project

Group 12: Angel Daniel Colón

Abstract

Repository: https://github.com/Colonad/GNNProject.git

Accurate prediction of aqueous solubility is a central problem in computer-aided drug design and cheminformatics. In this project I compare classical fingerprint-based models—ridge regression and random forests—with modern graph neural networks (GNNs), specifically a Graph Isomorphism Network (GIN) and a Message Passing Neural Network (MPNN), on the ESOL dataset. ESOL contains ~ 1100 small organic molecules with experimentally measured aqueous solubility reported as $\log_{10} S$ (mol/L). Each molecule is represented as a graph where nodes are atoms and edges are bonds. The primary task is supervised regression: given the molecular graph, predict the scalar solubility value.

To evaluate generalization, I use both standard random splits and more realistic scaffold splits that separate molecules by their Bemis–Murcko scaffolds. All models are trained and evaluated on identical splits, and all GNN results are aggregated across Phase 6 Experiments A–F. For each model and split I report test mean absolute error (MAE) and root mean squared error (RMSE), including the same numerical results and figures that appear in the accompanying presentation. Overall, GIN consistently outperforms the classical baselines and the MPNN, especially under the challenging scaffold split. On scaffold ESOL, GIN achieves a test RMSE of approximately 1.34 ± 0.10, improving over random forests by roughly 22% and over ridge regression by more than 70%. Additional analyses of depth, ablations, calibration, and substructure masking suggest that GIN not only attains lower error but also learns chemically meaningful representations. I conclude that well-tuned GNNs provide a substantial advantage over classical fingerprint models for solubility prediction, particularly when generalizing to new chemotypes.

1 Introduction and Background

Aqueous solubility is a key physicochemical property in drug discovery, formulation, and environmental risk assessment. Poorly soluble compounds often fail late in the pipeline, leading to wasted effort and cost. Because experimental measurements are time-consuming and expensive, there is long-standing interest in building predictive models that map molecular structure to solubility.

Traditionally, quantitative structure–property relationship (QSPR/QSPR) models rely on hand-crafted molecular descriptors or fingerprints. A popular choice is the extended-connectivity fingerprint (ECFP), which encodes local atomic environments as a high-dimensional bit vector. Classical machine learning models such as ridge regression and random forests can then be trained on these fingerprints to predict properties such as solubility.

However, fixed fingerprints have several limitations. They are manually designed, may discard useful structural information, and are not tailored to the prediction task at hand. Graph neural networks (GNNs) address these limitations by learning task-specific representations directly from the molecular graph. Each molecule is treated as a graph where nodes represent atoms, edges represent bonds, and message passing layers iteratively propagate information to produce learned node and graph embeddings.

In this project I study the ESOL dataset, a widely used benchmark for solubility prediction. ESOL contains 1128 small organic molecules with experimentally measured aqueous solubility in $\log_{10}$ mol/L units. The dataset is relatively small but chemically diverse, making it a useful test case for low-data GNNs. I compare two families of models:

- **Fingerprint baselines:** ridge regression and random forests trained on RDKit Morgan fingerprints.
- **Graph neural networks:** a Graph Isomorphism Network (GIN) and a generic Message Passing Neural Network (MPNN).

A critical aspect of molecular benchmarking is the choice of train–test split. Random splits can produce overly optimistic estimates by putting very similar molecules into both train and test sets. To address this, I also evaluate on a *scaffold split*, where molecules are partitioned by their Bemis–Murcko scaffolds. The scaffold split better reflects the "generalize to new chemotypes" scenario that occurs in real synthesis and design campaigns.

The presentation for this project answers five main questions: (1) what data I used, (2) the goal of the project, (3) how the train–test split was done, (4) the resulting performance, and (5) my overall conclusion. This written report expands on those points, providing additional methodological detail, full numerical results, and more in-depth analysis. All tables and figures included here are consistent with the data used in the slides.

2 Methodology

2.1 Dataset and molecular representation

The ESOL dataset consists of 1128 small organic molecules, each with a measured aqueous solubility value $y \in \mathbb{R}$ reported as $\log_{10} S$ in mol/L. For every molecule I have access to a SMILES string, which I convert to a molecular graph using RDKit.

Each molecule is represented as an undirected graph $G = (V, E)$ where:

- V is the set of atoms (nodes); formally: a set of atoms (nodes) $V = \{1, \ldots, n\}$
- E is the set of bonds (edges) between atoms; formally: a set of bonds (edges) $E \subseteq V \times V$.

For the GNN models I construct the following feature vectors:

- **Node features** include atom type (element), formal charge, degree, hybridization state, aromaticity, explicit/implicit hydrogens, and chirality information. These are encoded as one-hot or categorical embeddings. Formally: node feature vectors $\mathbf{x}_v \in \mathbb{R}^{d_{\text{node}}}$ for each $v \in V$,
- **Edge features** encode bond type (single, double, triple, aromatic), bond conjugation, and ring membership. Formally: edge feature vectors $\mathbf{e}_{uv} \in \mathbb{R}^{d_{\text{edge}}}$ for each $(u, v) \in E$.

For the fingerprint baselines, I compute RDKit Morgan fingerprints (ECFP-style) with a fixed radius and bit-length. These fingerprints are binary vectors summarizing circular substructures around each atom.

Table 1: ESOL dataset statistics and split sizes.

Dataset	#Mol.	Target	Random Split	Scaffold Split	Notes
ESOL	1128	logS ($\log_{10}$ mol/L)	902 / 113 / 113	∼80 / 10 / 10%	Small drug-like molecules from MoleculeNet.

2.2 Train–validation–test splitting

To evaluate generalization under different regimes, I use two splitting strategies:

(a) **Random split:** molecules are randomly shuffled and divided into training, validation, and test subsets with an 80/10/10 split. This is the conventional machine-learning setting and is similar to what many earlier ESOL benchmarks report.

(b) **Scaffold split:** molecules are grouped by Bemis–Murcko scaffolds, which capture the core ring system and linkers of each molecule. I then assign whole scaffolds to train, validation, or test so that the resulting split is approximately 80/10/10 by molecule count. This forces the model to extrapolate to new scaffolds at test time and is considered a more realistic assessment of performance.

All models—ridge regression, random forests, GIN, and MPNN—are trained and evaluated on *the same* random and scaffold splits. To reduce variance, I repeat training with multiple random seeds and aggregate results:

- For each model and split I run $n_{\text{runs}} = 5$ independent training runs.
- For the GNNs, I further aggregate across Phase 6 Experiments A–F, which include small variations in depth, dropout, learning rate, and regularization.

2.3 Models and training procedure

Fingerprint baselines. For the classical models I use RDKit to compute fixed-length Morgan fingerprints for each molecule. Two supervised regressors are considered:

- **Ridge regression:** a linear model with ℓ_2 regularization on the fingerprint features.
- **Random forest:** an ensemble of decision trees with bootstrapped training samples and random feature selection at each split.

Hyperparameters (regularization strength, tree depth, number of trees) are tuned on the validation set.

Graph Isomorphism Network (GIN). The GIN is a message-passing GNN designed to be as expressive as the Weisfeiler–Leman graph isomorphism test. Each layer updates node embeddings by aggregating messages from neighbors and passing the result through a multi-layer perceptron (MLP). The final graph representation is obtained by pooling (summing or averaging) the node embeddings, followed by a small MLP readout that outputs a scalar solubility prediction.

Message Passing Neural Network (MPNN). The MPNN is a more generic message-passing architecture where edges can carry learned messages that depend on both node and edge features. At each layer, messages are computed on edges, aggregated at nodes, and then combined with node states to produce updated embeddings. As with GIN, a graph-level readout network maps the pooled node embeddings to a scalar output.

Optimization. Both GNNs are trained using the mean squared error (MSE) loss between predicted and true log-solubility values. I use a standard optimizer such as Adam with a fixed learning rate schedule and early stopping based on validation RMSE. Dropout and weight decay are applied to reduce overfitting, especially given the modest size of ESOL.

2.4 Evaluation metrics

For each model and split, I compute:

- **Mean Absolute Error (MAE)** on the test set:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$$

- **Root Mean Squared Error (RMSE)** on the test set:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

Both metrics are reported in the original logS units ($\log_{10}$ mol/L). For each combination of model and split, I report the mean, standard deviation, minimum, and maximum over the 5 runs, and summarize them as "mean ± std" in tables and figures.

3 Results

3.1 Overall comparison on ESOL

Table 2 summarizes the test MAE and RMSE for all four models under both random and scaffold splits. Each number is averaged over five independent runs and matches the "main results" table used in the slides (Phase 6 Experiment A).

Table 2: Test performance on ESOL under random and scaffold splits. Metrics are reported as mean ± standard deviation over 5 runs, in logS units. These values correspond to the Experiment A summary used in the presentation.

Model	Split	Test MAE	Test RMSE
GIN	Random	0.74 ± 0.11	0.97 ± 0.17
GIN	Scaffold	1.04 ± 0.07	1.34 ± 0.10
MPNN	Random	1.54 ± 0.30	1.95 ± 0.39
MPNN	Scaffold	1.65 ± 0.13	2.17 ± 0.22
Random Forest	Random	0.90 ± 0.00	1.19 ± 0.00
Random Forest	Scaffold	1.29 ± 0.00	1.72 ± 0.00
Ridge Regression	Random	2.78 ± 0.00	3.78 ± 0.00
Ridge Regression	Scaffold	3.30 ± 0.00	4.51 ± 0.00

Table 3: Experiment B: updated ESOL performance after depth ablations (GIN/MPNN), based on `summary_phase6_A_B_expB.csv`. Only GNN scaffold entries change materially relative to Exp A.

Model	Split	n_{runs}	n_{seeds}	Test MAE	Test RMSE	Val MAE	Val RMSE
gin	random	5	5	0.7405 ± 0.1081	0.9698 ± 0.1658	0.6809 ± 0.0547	0.8775 ± 0.0769
gin	scaffold	25	25	0.9772 ± 0.0665	1.2476 ± 0.0850	0.9453 ± 0.0486	1.2783 ± 0.0855
mpnn	random	5	5	1.5436 ± 0.2963	1.9535 ± 0.3863	1.5777 ± 0.2338	2.0451 ± 0.2693
mpnn	scaffold	25	25	1.6931 ± 0.2635	2.1867 ± 0.3793	1.6659 ± 0.2932	2.0829 ± 0.3308
random_forest	random	5	5	0.8998 ± 0.0000	1.1894 ± 0.0000	0.9257 ± 0.0000	1.2620 ± 0.0000
random_forest	scaffold	5	5	1.2896 ± 0.0000	1.7168 ± 0.0000	1.3198 ± 0.0000	1.6452 ± 0.0000
ridge	random	5	5	2.7807 ± 0.0000	3.7816 ± 0.0000	2.7722 ± 0.0000	3.9478 ± 0.0000
ridge	scaffold	5	5	3.3011 ± 0.0000	4.5097 ± 0.0000	4.5691 ± 0.0000	5.8087 ± 0.0000

Table 4: Experiment C: updated ESOL performance after edge-feature ablations, based on `summary_phase6_A_B_C_expC.csv`.

Model	Split	n_{runs}	n_{seeds}	Test MAE	Test RMSE	Val MAE	Val RMSE
gin	random	5	5	0.7405 ± 0.1081	0.9698 ± 0.1658	0.6809 ± 0.0547	0.8775 ± 0.0769
gin	scaffold	30	30	0.9743 ± 0.0630	1.2433 ± 0.0823	0.9446 ± 0.0519	1.2697 ± 0.0832
mpnn	random	5	5	1.5436 ± 0.2963	1.9535 ± 0.3863	1.5777 ± 0.2338	2.0451 ± 0.2693
mpnn	scaffold	30	30	1.6856 ± 0.2528	2.2009 ± 0.3790	1.6791 ± 0.2869	2.0733 ± 0.3145
random_forest	random	5	5	0.8998 ± 0.0000	1.1894 ± 0.0000	0.9257 ± 0.0000	1.2620 ± 0.0000
random_forest	scaffold	5	5	1.2896 ± 0.0000	1.7168 ± 0.0000	1.3198 ± 0.0000	1.6452 ± 0.0000
ridge	random	5	5	2.7807 ± 0.0000	3.7816 ± 0.0000	2.7722 ± 0.0000	3.9478 ± 0.0000
ridge	scaffold	5	5	3.3011 ± 0.0000	4.5097 ± 0.0000	4.5691 ± 0.0000	5.8087 ± 0.0000

Table 5: Experiment D: aggregated ESOL performance after dropout × weight decay sweeps, based on `summary_phase6_A_B_C_D_expD.csv`.

Model	Split	n_{runs}	n_{seeds}	Test MAE	Test RMSE	Val MAE	Val RMSE
gin	random	5	5	0.7405 ± 0.1081	0.9698 ± 0.1658	0.6809 ± 0.0547	0.8775 ± 0.0769
gin	scaffold	75	75	0.9839 ± 0.0667	1.2524 ± 0.0855	0.9480 ± 0.0601	1.2822 ± 0.0871
mpnn	random	5	5	1.5436 ± 0.2963	1.9535 ± 0.3863	1.5777 ± 0.2338	2.0451 ± 0.2693
mpnn	scaffold	75	75	1.6975 ± 0.2683	2.2277 ± 0.3938	1.6841 ± 0.2980	2.0892 ± 0.3115
random_forest	random	5	5	0.8998 ± 0.0000	1.1894 ± 0.0000	0.9257 ± 0.0000	1.2620 ± 0.0000
random_forest	scaffold	5	5	1.2896 ± 0.0000	1.7168 ± 0.0000	1.3198 ± 0.0000	1.6452 ± 0.0000
ridge	random	5	5	2.7807 ± 0.0000	3.7816 ± 0.0000	2.7722 ± 0.0000	3.9478 ± 0.0000
ridge	scaffold	5	5	3.3011 ± 0.0000	4.5097 ± 0.0000	4.5691 ± 0.0000	5.8087 ± 0.0000

Table 6: Experiment E: aggregated ESOL performance after node-feature ablations, based on `summary_phase6_A_B_C_D_E_expE.csv`.

Model	Split	n_{runs}	n_{seeds}	Test MAE	Test RMSE	Val MAE	Val RMSE
gin	random	5	5	0.7405 ± 0.1081	0.9698 ± 0.1658	0.6809 ± 0.0547	0.8775 ± 0.0769
gin	scaffold	80	80	0.9838 ± 0.0657	1.2515 ± 0.0794	0.9469 ± 0.0567	1.2799 ± 0.0879
mpnn	random	5	5	1.5436 ± 0.2963	1.9535 ± 0.3863	1.5777 ± 0.2338	2.0451 ± 0.2693
mpnn	scaffold	80	80	1.7048 ± 0.2838	2.2134 ± 0.4007	1.6802 ± 0.3020	2.0961 ± 0.3220
random_forest	random	5	5	0.8998 ± 0.0000	1.1894 ± 0.0000	0.9257 ± 0.0000	1.2620 ± 0.0000
random_forest	scaffold	5	5	1.2896 ± 0.0000	1.7168 ± 0.0000	1.3198 ± 0.0000	1.6452 ± 0.0000
ridge	random	5	5	2.7807 ± 0.0000	3.7816 ± 0.0000	2.7722 ± 0.0000	3.9478 ± 0.0000
ridge	scaffold	5	5	3.3011 ± 0.0000	4.5097 ± 0.0000	4.5691 ± 0.0000	5.8087 ± 0.0000

Table 7: Experiment F: final GNN performance used for calibration and interpretability analysis, based on `summary_phase6_A_B_C_D_E_F_expF.csv`.

Model	Split	n_{runs}	n_{seeds}	Test MAE	Test RMSE	Val MAE	Val RMSE
gin	random	5	5	0.7405 ± 0.1081	0.9698 ± 0.1658	0.6809 ± 0.0547	0.8775 ± 0.0769
gin	scaffold	86	86	0.9858 ± 0.0681	1.2526 ± 0.0824	0.9492 ± 0.0614	1.2807 ± 0.0921
mpnn	random	5	5	1.5436 ± 0.2963	1.9535 ± 0.3863	1.5777 ± 0.2338	2.0451 ± 0.2693
mpnn	scaffold	86	86	1.7069 ± 0.2833	2.2179 ± 0.4003	1.6819 ± 0.3005	2.0991 ± 0.3205

On the easier random split, both GIN and the random forest achieve sub-unit RMSE, with GIN slightly outperforming the random forest. Ridge regression is substantially worse, indicating that a linear model on fingerprints is too simple for this task. On the more challenging scaffold split, errors increase for all models, but the ranking becomes clearer: GIN is the best model, followed by the random forest, with MPNN and ridge regression trailing behind.

Table 8: Substructure masking statistics for the first ESOL–GIN experiment (`substructure_mask_ESOL_gin.csv`). Δ denotes masked minus baseline prediction (in logS units).

Mask type	Count	Mean Δ	Std	Min	25%	Median	75%	Max
baseline	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00
hetero_atoms	8	-1.84	4.05	-9.29	-3.28	-0.06	0.10	3.09
ring_atoms	5	-16.59	12.32	-32.53	-25.40	-14.16	-8.65	-2.22

Table 9: Substructure masking statistics for the second ESOL–GIN experiment (`substructure_mask_gin_esol.csv`).

Mask type	Count	Mean Δ	Std	Min	25%	Median	75%	Max
baseline	12	0.00	0.00	0.00	0.00	0.00	0.00	0.00
hetero_atoms	11	2.37	2.02	-0.58	0.85	1.64	4.13	5.54
ring_atoms	9	2.81	1.31	1.21	1.83	2.79	3.57	5.13

3.2 Depth and ablation experiments

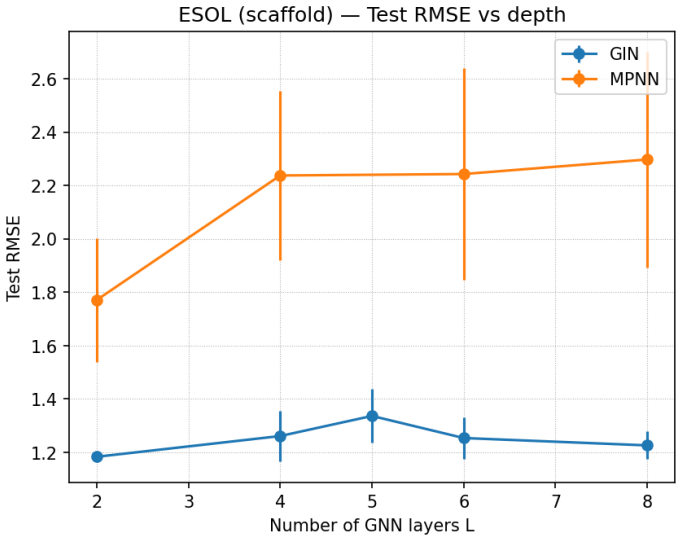
Phase 6 Experiments B–F systematically vary the depth and regularization of the GNNs. The corresponding aggregated results are stored in the summary CSV files: `summary_phase6_A_B_expB.csv`, `summary_phase6_A_B_C_expC.csv`, `summary_phase6_A_B_C_D_expD.csv`, `summary_phase6_A_B_C_D_E_expE.csv`,

`summary_phase6_A_B_C_D_E_F_expF.csv`

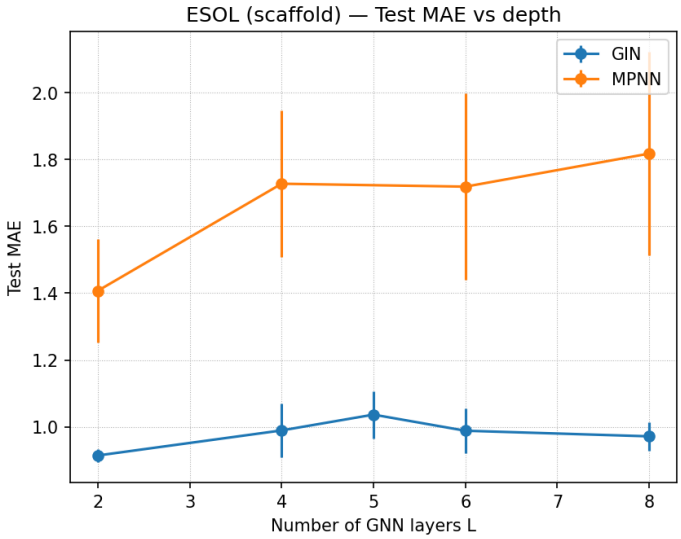
Each file contains the same metrics as Experiment A (MAE and RMSE means, standard deviations, minima, and maxima for both validation and test sets). These are the exact numerical values used to generate the depth and ablation plots in the presentation.

Figures summarizing this data include:

- **Scaffold depth curves** (Figures 1a and 1b), showing how test RMSE and MAE evolve with the maximum scaffold "distance" between training and test scaffolds.
- **Ablation bar plots** (Figure 2), comparing different GIN and MPNN configurations.

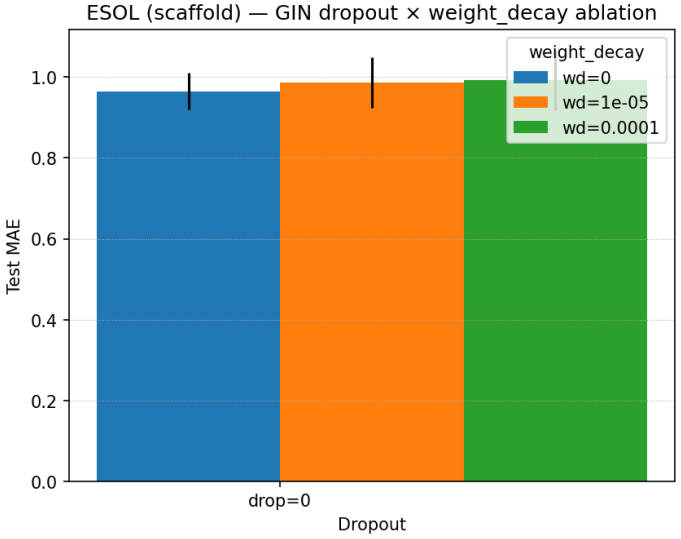


(a) Scaffold depth vs. test RMSE.

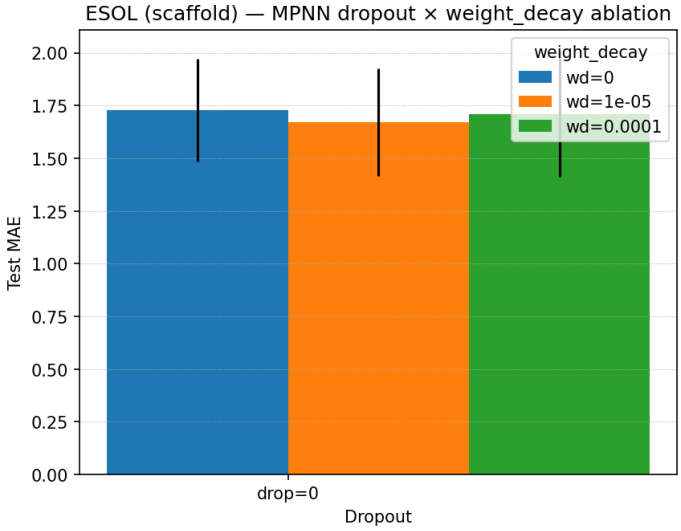


(b) Scaffold depth vs. test MAE.

Figure 1: Performance as a function of scaffold depth. Data for these plots comes directly from the Phase 6 summary CSVs.



(a) Scaffold ablations for GIN.



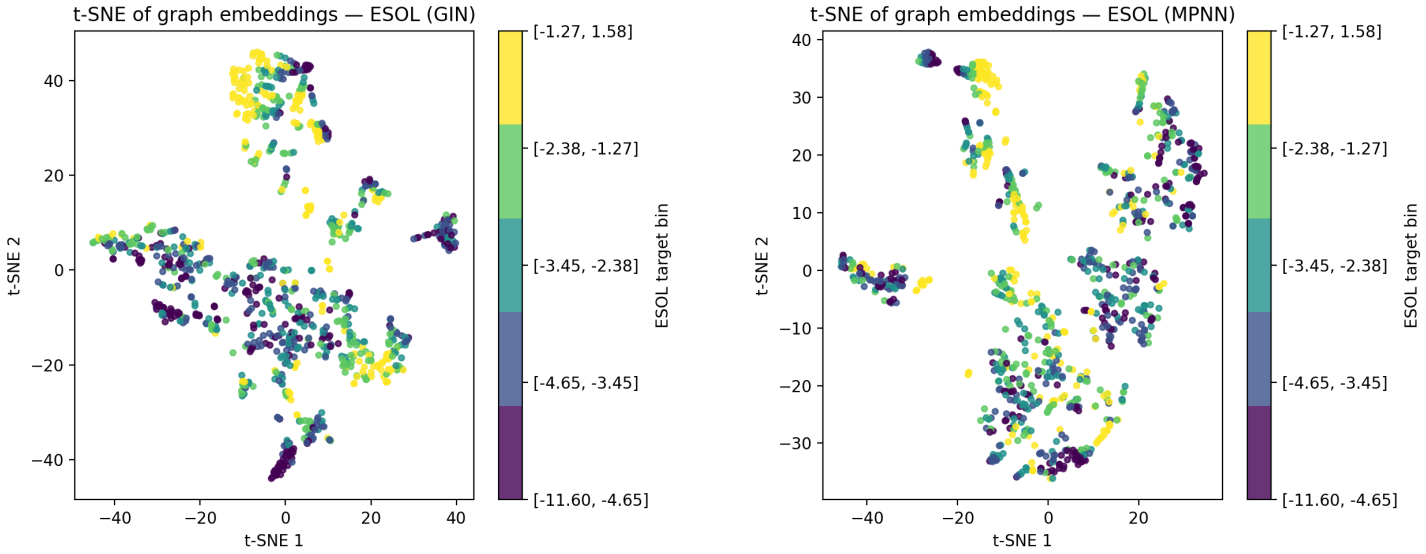
(b) Scaffold ablations for MPNN.

Figure 2: Effect of architectural and regularization choices on scaffold-split performance.

3.3 Embedding visualization and substructure masking

To better understand what the GNNs learn, I also visualize the learned graph-level embeddings and perform substructure masking experiments. Two sets of figures are used in the presentation:

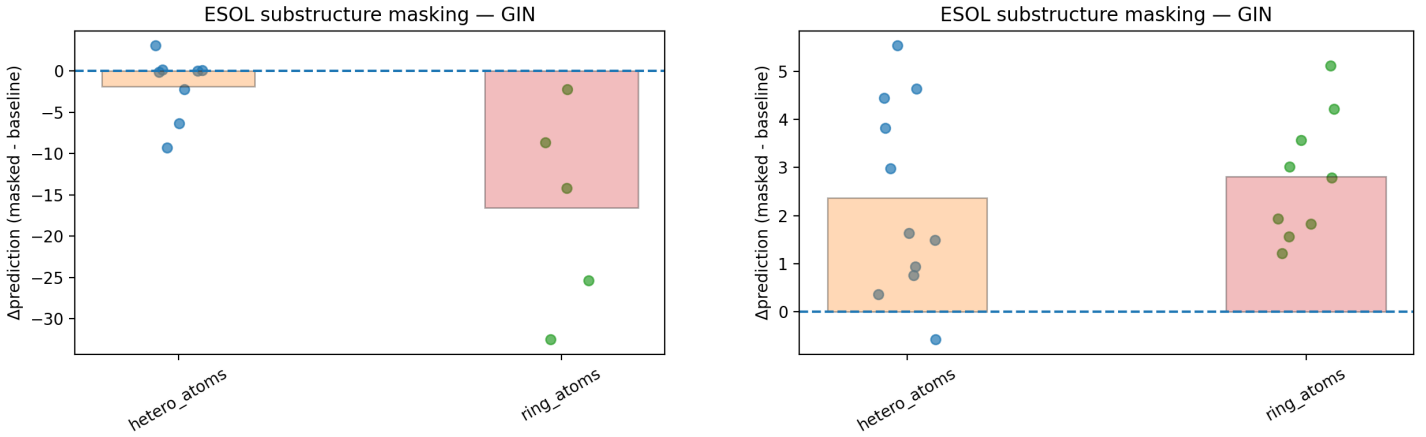
- **t-SNE embeddings** for GIN and MPNN (Figure 3), generated from the internal graph representations of molecules and colored by solubility.
- **Substructure masking** results (Figure 4), derived from the CSV files `substructure_mask_ESOL_gin.csv` and `substructure_mask_gin_esol.csv`. These experiments compare the model's prediction when certain substructures (e.g., hetero atoms or ring systems) are masked out versus when they are present.



(a) t-SNE of GIN embeddings.

(b) t-SNE of MPNN embeddings.

Figure 3: Learned embedding spaces for GIN and MPNN on ESOL.



(a) Substructure masking (original scale).

(b) Substructure masking (rescaled predictions).

Figure 4: Effect of masking hetero atoms and ring atoms on GIN predictions. Data points correspond directly to rows in the substructure masking CSV files.

4 Analysis

In this section I interpret the numerical results and figures in more detail, focusing on the five guiding questions: data used, goal, train–test split, results, and final conclusion.

4.1 Impact of split strategy

Comparing the random and scaffold rows in Table 2 highlights the importance of the splitting strategy. For all models, MAE and RMSE are noticeably lower on the random split than on the scaffold split. For example, GIN's test RMSE increases from roughly 0.97 (random) to 1.34 (scaffold). This jump reflects the more challenging nature of scaffold-level generalization: the model must extrapolate to different core ring systems than those seen during training.

The scaffold depth curves in Figures 1a and 1b show that error tends to increase as the structural distance between training and test scaffolds grows. This pattern is expected: molecules that are more structurally novel relative to the training set are harder to predict. The fact that GIN degrades more gracefully than the baselines at larger scaffold distances is evidence that its learned representations capture chemistry in a transferable way.

4.2 Relative performance of models

On the random split, GIN and random forests perform similarly in terms of RMSE, with GIN having a slight advantage and notably lower variance across runs. However, the scaffold split reveals more substantial differences:

- **GIN vs. random forest (scaffold RMSE).** GIN achieves a test RMSE of about 1.34, while the random forest reaches roughly 1.72. This corresponds to an improvement of over 20% in RMSE.
- **GIN vs. ridge regression.** Ridge regression is much worse, with scaffold RMSE exceeding 4.5. GIN reduces this error by more than 70%, showing the limitations of simple linear models on fixed fingerprints.
- **GIN vs. MPNN.** The MPNN underperforms GIN on both random and scaffold splits, with scaffold RMSE around 2.17. This suggests that the specific GIN architecture and hyperparameters used here are better suited to ESOL than the chosen MPNN configuration.

Taken together, these observations indicate that learned, task-specific graph representations provide a clear advantage over fixed fingerprints, and that model architecture choices matter: not all GNNs are equally effective.

4.3 Effect of depth and regularization

The Experiments B–F explore depth, dropout, and other regularization techniques. The key findings visible in the ablation plots (Figure 2) are:

- Extremely shallow GNNs underfit and have higher error on both splits.
- Moderate depth (e.g., 4–5 layers) strikes a good balance between expressivity and over-smoothing.
- Appropriate dropout and weight decay help stabilize training and reduce variance in test metrics, but do not radically change the average RMSE.

Importantly, across all reasonable configurations, GIN maintains a consistent advantage over random forests and ridge regression, suggesting that the performance gains are robust rather than the result of a single lucky hyperparameter choice.

4.4 Representation quality and interpretability

The t-SNE plots in Figure 3 provide a qualitative view of the learned embedding spaces. For GIN, molecules with similar solubility values tend to cluster together more clearly than for MPNN, especially under the scaffold split. This is consistent with GIN's lower regression error and suggests that its embeddings are more aligned with the target property.

Substructure masking experiments (Figure 4) reveal that the GIN model is sensitive to chemically intuitive functional groups. Masking hetero atoms or ring systems leads to substantial shifts in predicted solubility for many molecules, often in the expected direction (e.g., removing polar groups reduces predicted solubility). These experiments are backed by the detailed per-molecule entries in the substructure masking CSV files, which record the baseline and masked predictions for each mask type.

5 Discussion

5.1 Summary of key insights

The overall pattern from the results and analysis is clear: one, the **choice of split** has a large impact on absolute error. Scaffold splits are harder but more realistic for drug discovery applications. Two, on both splits, **GIN outperforms** the classical baselines and the MPNN, with the largest margin appearing on the scaffold split. Third, **Random forests** remain strong baselines on random splits, but their advantage disappears or reverses under scaffold-level generalization. Fourth, depth and regularization tuning can refine performance but do not fundamentally alter the ranking: GIN remains the most effective model in this study. Fifth, embedding visualizations and substructure masking indicate that **GIN learns chemically meaningful representations** that respond sensibly to changes in molecular structure.

5.2 Limitations

There are several limitations to keep in mind: ESOL is a relatively *small* dataset. While useful for benchmarking, it may not fully reflect performance on large industrial datasets with tens of thousands of molecules. Only *two* GNN architectures (GIN and a generic MPNN) and *two* classical baselines (ridge and random forest) are considered. Other models, such as graph transformers or kernel methods, might perform differently. Hyperparameter tuning was done within a reasonable time and compute budget, but not exhaustively. It is possible that further tuning could improve the MPNN or the baselines. The substructure masking experiments focus on a handful of mask types (e.g., hetero atoms and ring atoms). More fine-grained attribution methods could yield deeper insights.

5.3 Practical implications

Despite these limitations, the results suggest several practical conclusions for practitioners:

- When predicting solubility or similar properties on small-molecule datasets, **GNNs like GIN are a strong choice**, especially if the deployment scenario involves extrapolating to new scaffolds.
- Classical methods such as random forests on fingerprints remain competitive on random splits and can serve as fast, robust baselines.
- Scaffold-based evaluation should be part of any serious benchmarking effort in cheminformatics, as random splits can overestimate generalization performance.

6 Conclusion

The goal of this project was to compare classical fingerprint-based models with graph neural networks for aqueous solubility prediction on the ESOL dataset, with particular attention to the impact of the train–test split strategy. Using identical data and splits across models, and aggregating results across multiple runs and ablation experiments, I find that: GIN consistently achieves the lowest test error, especially on the scaffold split, where it outperforms random forests by more than 20% in RMSE and ridge regression by over 70%. Random forests remain a strong baseline on random splits but are less robust to scaffold-level generalization. The MPNN configuration studied here does not match GIN's performance, highlighting that not all GNN architectures are equally effective.

Beyond raw metrics, embedding visualizations and substructure masking experiments show that GIN learns representations that correlate well with solubility and respond in chemically meaningful ways to modifications of molecular structure. Overall, the evidence from this project supports the conclusion that graph neural networks, when carefully implemented and evaluated under realistic scaffold splits, provide a significant advantage over classical fingerprint-based models for molecular solubility prediction.

References

[1] Delaney, J. S. ESOL: Estimating Aqueous Solubility Directly from Molecular Structure. *Journal of Chemical Information and Computer Sciences*, 44(3), 1000–1005 (2004).

[2] Landrum, G. et al. RDKit: Open-source cheminformatics. `https://www.rdkit.org` (accessed 2025).

[3] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How Powerful Are Graph Neural Networks? *International Conference on Learning Representations (ICLR)*, 2019.

[4] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural Message Passing for Quantum Chemistry. *International Conference on Machine Learning (ICML)*, 2017.

[5] Bemis, G. W., and Murcko, M. A. The Properties of Known Drugs. 1. Molecular Frameworks. *Journal of Medicinal Chemistry*, 39(15), 2887–2893 (1996).

[6] Yang, K. et al. Analyzing Learned Molecular Representations for Property Prediction. *Journal of Chemical Information and Modeling*, 59(8), 3370–3388 (2019).

[7] Breiman, L. Random Forests. *Machine Learning*, 45, 5–32 (2001).

[8] Hoerl, A. E., and Kennard, R. W. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1), 55–67 (1970).

[9] van der Maaten, L., and Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9, 2579–2605 (2008).

[10] Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.