Test-time Adaptation for Graph-based Molecular Solubility Prediction

Philipp Sepin
165.164 Selected Topics in Theoretical Chemistry

July 15, 2025

Abstract

Molecular solubility prediction is a critical task in drug development, but models often struggle with distribution shifts between training and test data. This project addresses this challenge by implementing test-time adaptation for graph neural networks and applying it to molecular solubility prediction.

This project was carried out as part of the seminar 165.164 Selected Topics in Theoretical Chemistry at TU Wien, under the supervision of Prof. Esther Heid.

1 Introduction

Molecular solubility prediction is a critical task in drug development, directly impacting a compound's bioavailability and therapeutic potential. Experimental solubility measurement requires substantial time and resources, making computational prediction essential for screening large molecular databases [5].

Recent advances in solubility prediction have been driven by deep learning architectures and molecular embedding approaches. Feature-based neural networks, graph-based neural networks (GNNs), and structural attention methods have emerged as powerful predictive models [5].

However, when there is a distribution shift between the training and test data, as with *AqSolDB* [7] and *OChemUnseen* [5], these models often struggle to generalize. This project aims to solve this by utilizing test-time adaptation (TTA) for graph neural networks (GNNs) to shift test set distributions towards the training set distribution, thereby improving generalization.

TTA involves training a model on a source domain, then adapting it at test time by performing a few self-supervised learning (SSL) steps on each test sample before prediction. It has been applied in various domains, such as semantic segmentation, object detection, medical image processing, video depth prediction, question answering, sentiment analysis, entity recognition, speech processing, social network analysis, as well as in protein and enzyme classification [4, 1].

2 Methods

2.1 Dataset

For this project, the *AqSolDB* dataset [7] is used for training and validation. It contains about 8000 molecules as SMILES strings, along with their solubility values. For testing, the *OChemUnseen* dataset [5] is used, which contains about 2000 molecules as SMILES strings, along with their solubility values. This dataset is fully orthogonal to the training dataset.

The SMILES strings are converted to molecular graphs using the RDKit library [3], and one-hot encoded node and edge features are added. The node features included element type, number of bonds, electric charge, aromaticity atomic mass, and orbital hybridization, while the

edge features included bond order, aromaticity, conjugation, and whether the bond is in a ring. The graphs are then converted to PyTorch Geometric [2] data objects.

To enhance the distribution shift, the datasets are filtered as follows:

- The training set contains molecules from *AqSolDB* with 6-19 atoms and no amino groups.
- The validation set contains molecules from *AqSolDB* with ≤ 5 atoms and no amino groups.
- The first test set (20 Atom set) contains molecules from *OChemUnseen* with ≥ 20 atoms.
- The second test set (NH2 set) contains molecules from *OChemUnseen* with ≥ 20 atoms and amino groups.

2.2 Model

The model used for this project is a Y-shaped architecture, consisting of a shared encoder, which branches into a decoder and a prediction head, as shown in Figure 1.

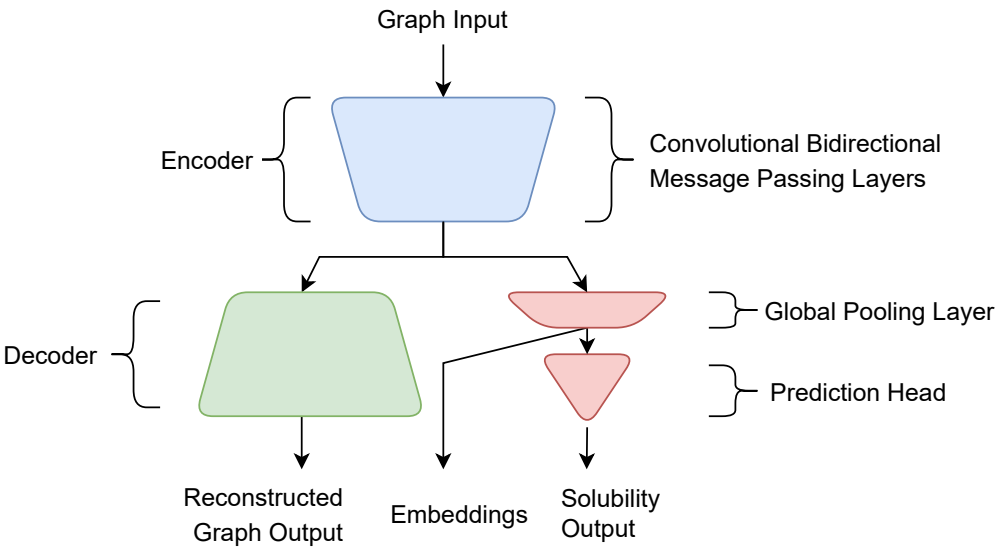

Figure 1: The model architecture.

The encoder is a convolutional bidirectional message passing neural network (MPNN), a GNN that applies convolutional operations to aggregate information from neighboring nodes in both directions through iterative message passing. It consists of two graph convolutional layers which return an embedding vector of size 16 for each node. These nodal embedding vectors are an information-dense representation of the molecular graph. A 2D projection of the mean-pooled 16-dimensional embedding space with corresponding solubility values is shown in Figure 2, visualized using two dimensionality reduction techniques, t-SNE [8] and UMAP [6].

The decoder consists of two fully connected layers that reconstruct node and edge features from the nodal embedding vectors. The prediction head employs a global pooling layer which aggregates the nodal embedding vectors into a single global embedding vector by taking their mean, followed by two fully connected layers that map the global embedding vector to the predicted solubility value, creating a multi-task learning architecture.

2.3 Training

Following our architecture design, the model can be trained on two tasks. The first one is a self-supervised task, where masked node and edge features are denoised and reconstructed. For this,

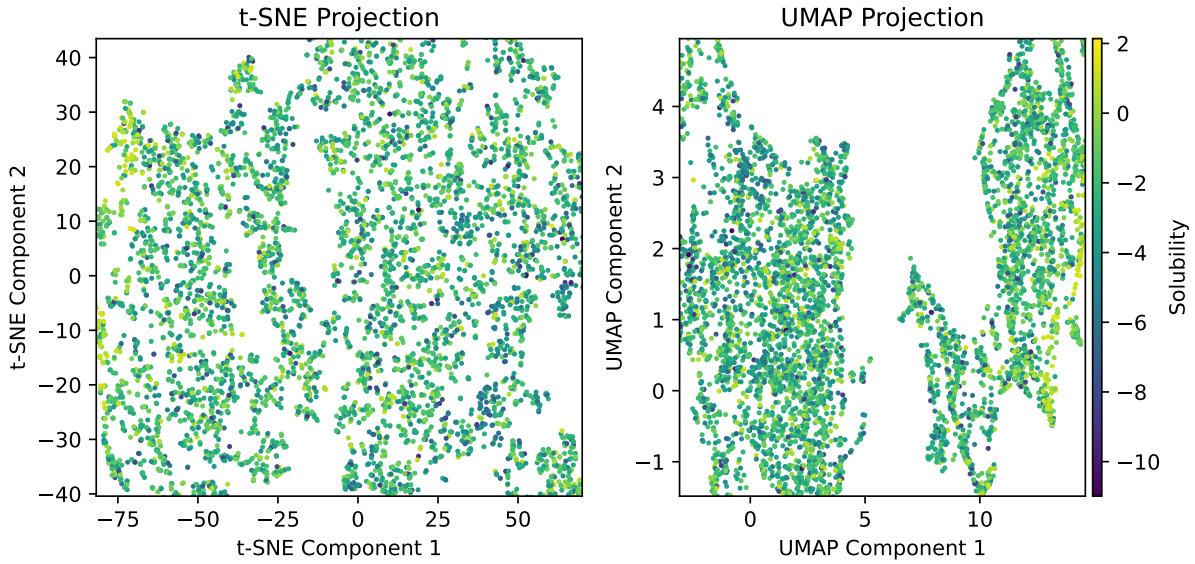

Figure 2: 2D projection of the embedding vectors with their corresponding solubility.

all features of a certain percentage of nodes and edges are masked. The encoder then learns to create an information-dense representation in form of the nodal embedding vectors, from which the decoder learns to reconstruct the denoised node and edge features. The second task is the supervised task, where the encoder also learns to create nodal embedding vectors, from which the prediction head learns to predict the solubility value.

Following the literature, both tasks are trained simultaneously by combining their respective losses [1, 9]. We implemented this by summing the weighted denoising and prediction losses, weighting the denoising loss twice as high as the prediction loss.

The model was trained for 70 epochs with a batch size of 1024 using the Adam optimizer. Hyperparameters were optimized on the validation set. The learning rate was optimized to $5 \cdot 10^{-3}$ and training was performed on a single NVIDIA GeForce GTX 960M GPU.

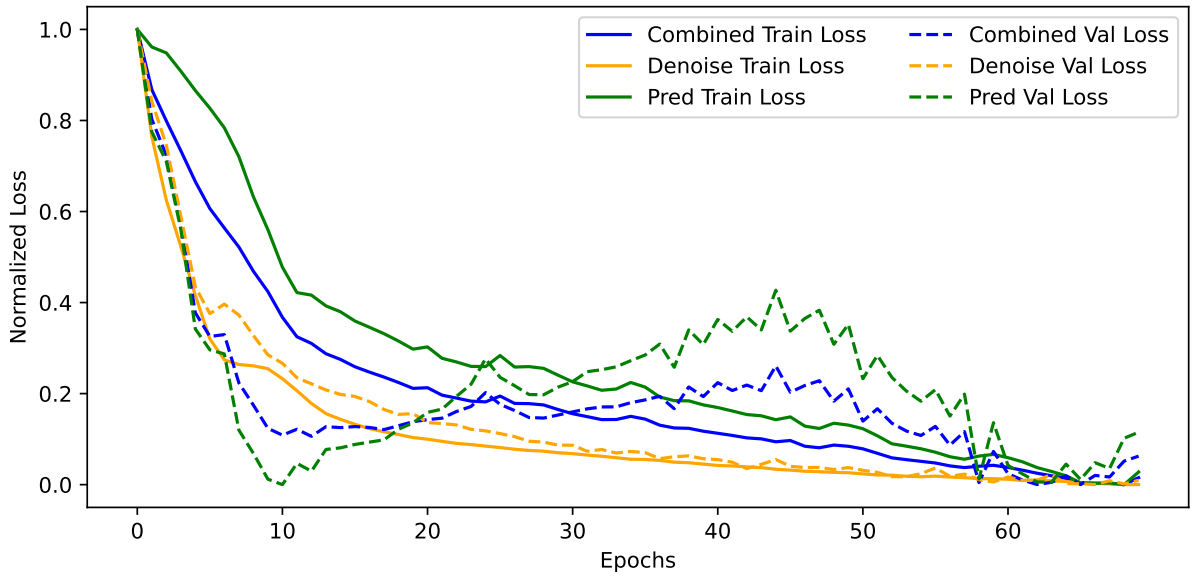

Figure 3: Training and validation losses.

As shown in Figure 3, the denoising validation loss steadily decreases, whereas the prediction validation loss showed an interesting double-descent behaviour, where it first converges at around 10 epochs, then increases again, and finally converges at around 60 epochs.

2.4 Test-time Adaptation and Prediction

For TTA, each test sample is processed individually. The encoder is adapted to the specific molecular structure through a few gradient descent steps on the self-supervised loss, after which the adapted model predicts the solubility using the standard prediction head. The model is then reset to its original state for the next test sample. The step size and number of gradient steps were optimized to $1.9 \cdot 10^{-3}$ and 5 steps, respectively.

The distribution shift and effect of TTA can be seen in the 2D projections of the embedding space shown in Figures 5 and 4.

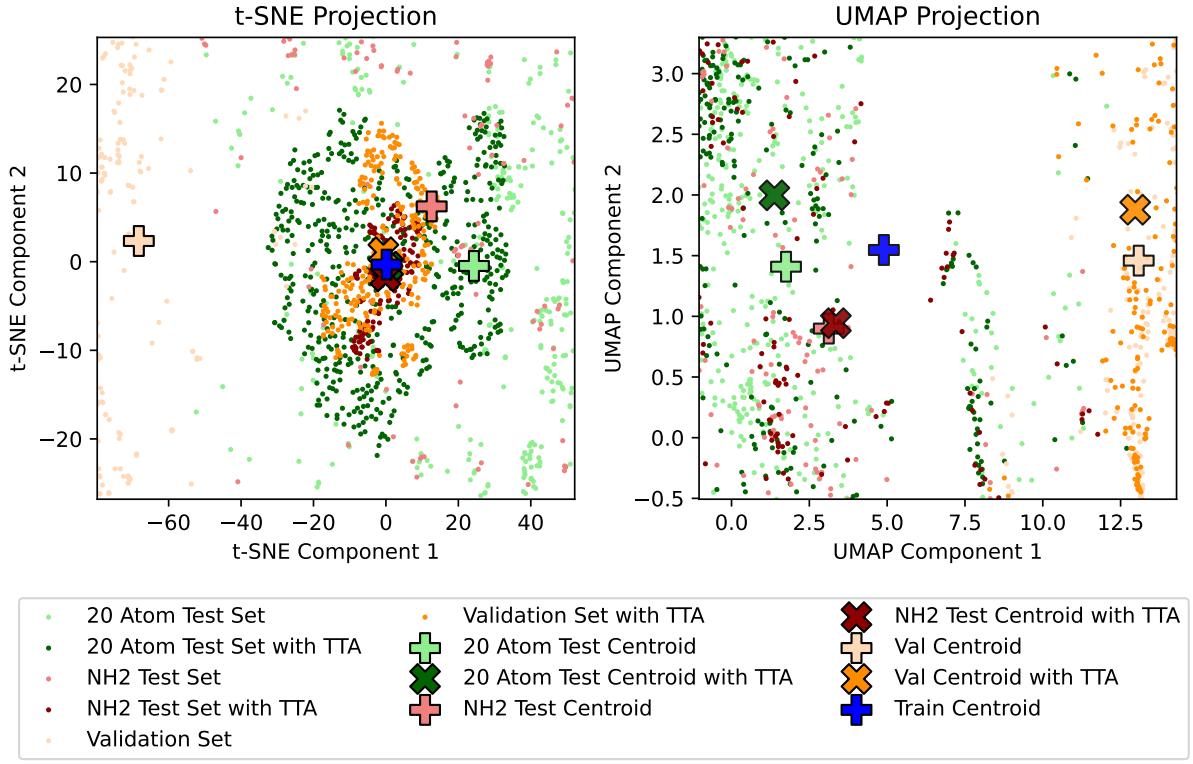

Figure 4: 2D projection of the embedding vectors before and after TTA.

3 Results

Model	Validation RMSE	Test RMSE (20 Atom set)	Test RMSE (NH2 set)
Model without TTA	1.3886	1.8705	1.8898
Model with TTA	1.2675	2.0093	2.1690
Reference Model	1.2540	2.2291	2.1379

Table 1: Performance comparison of different model configurations.

As shown in Table 1, TTA worsened performance on both test sets, but improved validation performance. Figure 4 shows that TTA shifted the test and validation set embeddings. The t-SNE projection shows a clear shift towards the training set distribution, whereas the UMAP projection shows shifts in other directions, which aligns with the observed performance degradation. This suggests that UMAP, with its focus on preserving global structure, may be more suitable for analyzing the effects of TTA.

A reference model trained only on the prediction task shows that SSL training can help generalization, as the reference model had worse performance on both test sets, but slightly

better validation performance.

Additional figures are provided in the appendix. The code for this project is available at github.com/p0017/Molecular-Test-Time-Adaptation under the GPL-3.0 license.

4 Conclusion

This study implemented test-time adaptation (TTA) for graph neural networks (GNNs) and applied it to molecular solubility prediction. While TTA successfully shifted the distributions of the validation and test sets, the shift did not consistently move them closer to the training set distribution, resulting in partially worse predictive performance. Future work could explore alternative self-supervised learning (SSL) tasks, different adaptation strategies, or investigate TTA effectiveness on datasets with different distribution shifts.

References

[1] Taoyong Cui, Chenyu Tang, Dongzhan Zhou, Yuqiang Li, Xingao Gong, Wanli Ouyang, Mao Su, and Shufei Zhang. Online test-time adaptation for better generalization of interatomic potentials to out-of-distribution data. *Nature Communications*, 16(1):1891, 2025.

[2] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[3] Greg Landrum, Paolo Tosco, Brian Kelley, Ricardo Rodriguez, David Cosgrove, Riccardo Vianello, sriniker, Peter Gedeck, Gareth Jones, Eisuke Kawashima, NadineSchneider, Dan Nealschneider, Andrew Dalke, Matt Swain, Brian Cole, tadhurst cdd, Samo Turk, Aleksandr Savelev, Alain Vaucher, and guillaume godin. rdkit/rdkit: 2025_03_3 (Q1 2025) Release, 2025.

[4] Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 133(1):31–64, 2025.

[5] P Llompart, C Minoletti, S Baybekov, Dragos Horvath, G Marcou, and A Varnek. Will we ever be able to accurately predict solubility? *Scientific Data*, 11(1):303, 2024.

[6] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[7] Murat Cihan Sorkun, Abhishek Khetan, and Süleyman Er. Aqsoldb, a curated reference set of aqueous solubility and 2d descriptors for a diverse set of compounds. *Scientific data*, 6(1):143, 2019.

[8] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[9] Yiqi Wang, Chaozhuo Li, Wei Jin, Rui Li, Jianan Zhao, Jiliang Tang, and Xing Xie. Test-time training for graph neural networks. *arXiv preprint arXiv:2210.08813*, 2022.

5 Appendix

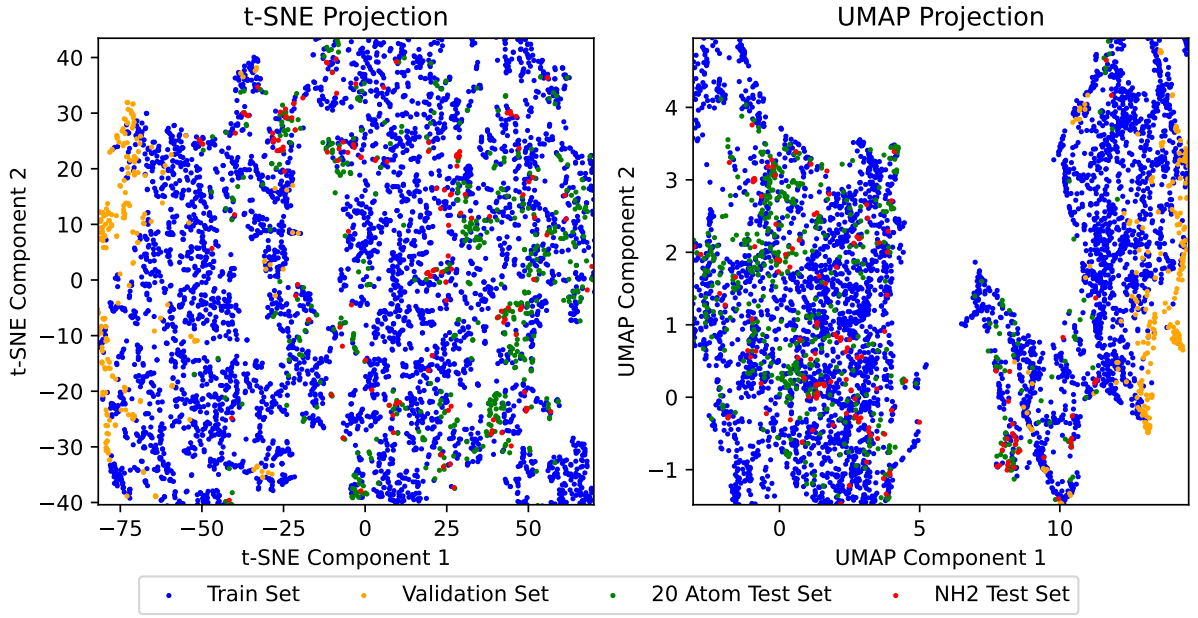

Figure 5: 2D projection of the embedding vectors with their corresponding sets.

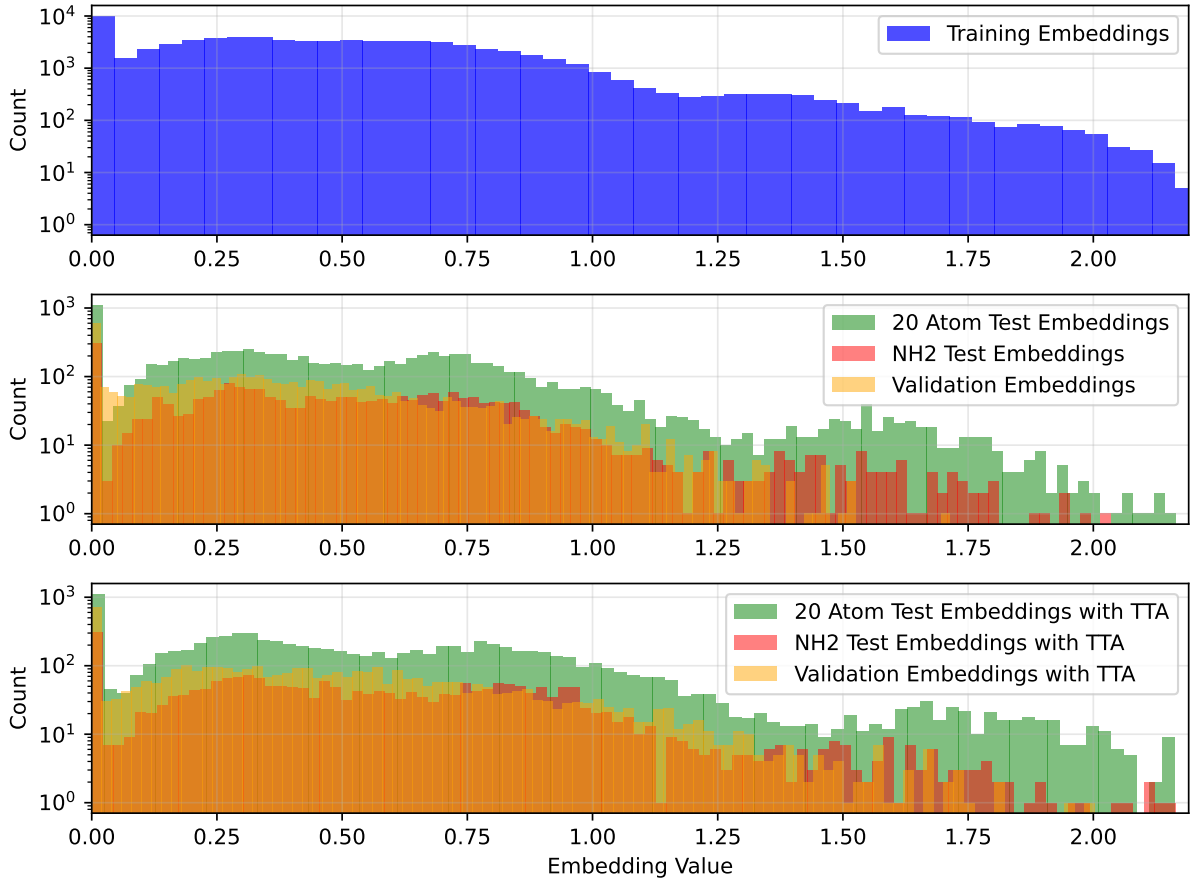

Figure 6: Histogram of the embedding vector components.

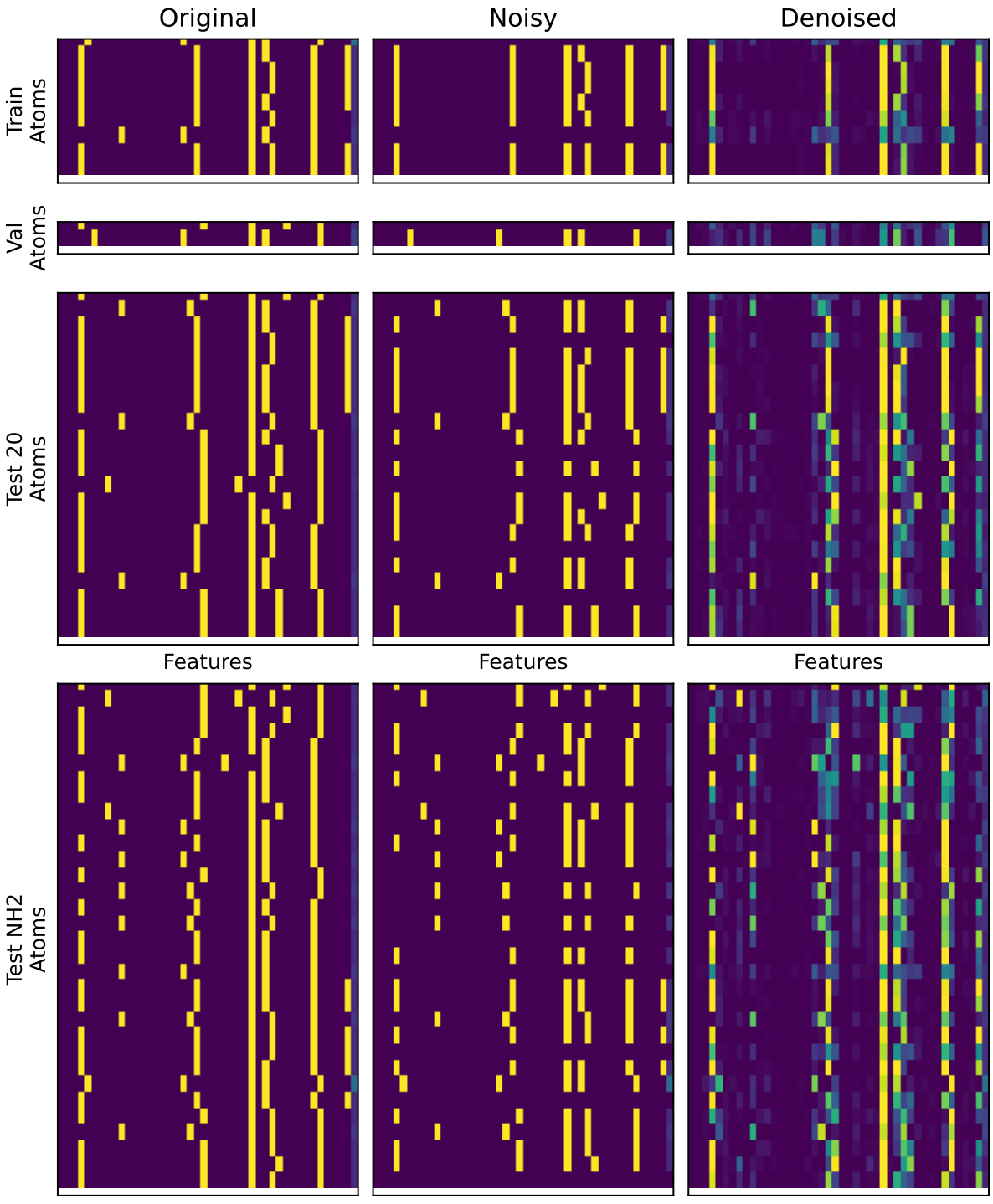

Figure 7: Original, masked, and reconstructed node features.

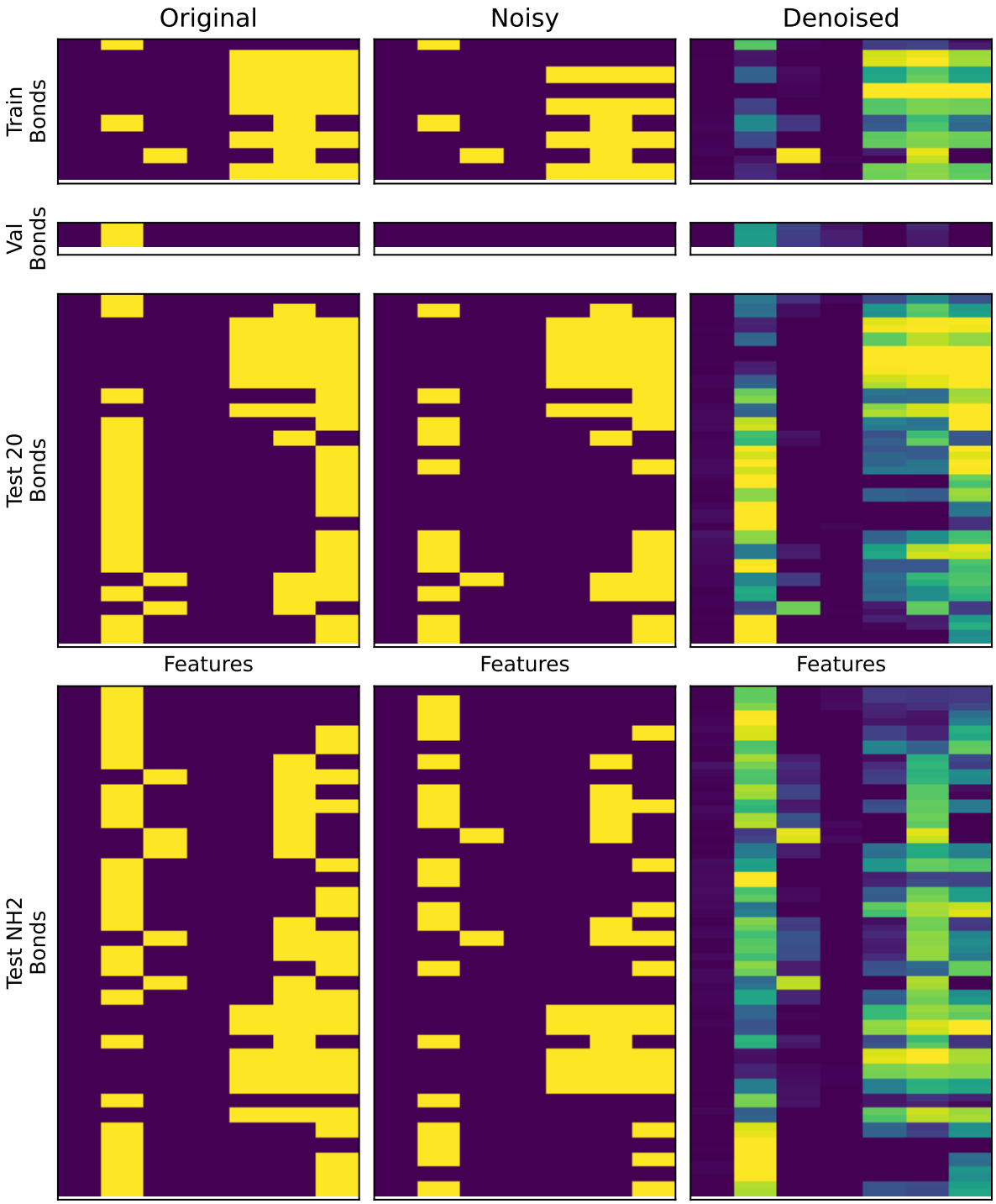

Figure 8: Original, masked, and reconstructed edge features.