Test-time Adaptation for Graph-based Molecular Solubility Prediction

Philipp Sepin
165.164 Selected Topics in Theoretical Chemistry

June 6, 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 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 in drug development [?].

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 [?].

However, when there is a certain distribution shift between the training and test data, as with *OChemUnseen* [?] and *AqSolDB* [?], 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 trains a model on a source domain, then adapts 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 [?, ?].

2 Methods

2.1 Dataset

For this project, the *AqSolDB* dataset [?] was used for training and validation. It contains about 8000 molecules as SMILES strings, along with their solubility values. For testing, the *OChemUnseen* dataset [?] was 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 were converted to molecular graphs using the RDKit library [?], and one-hot encoded node and edge features were 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 were then converted to PyTorch Geometric [?] data objects.

To enhance the distribution shift, the datasets were 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, which consists of a shared encoder, which branches into a decoder and a prediction head, as shown in Figure **??**.

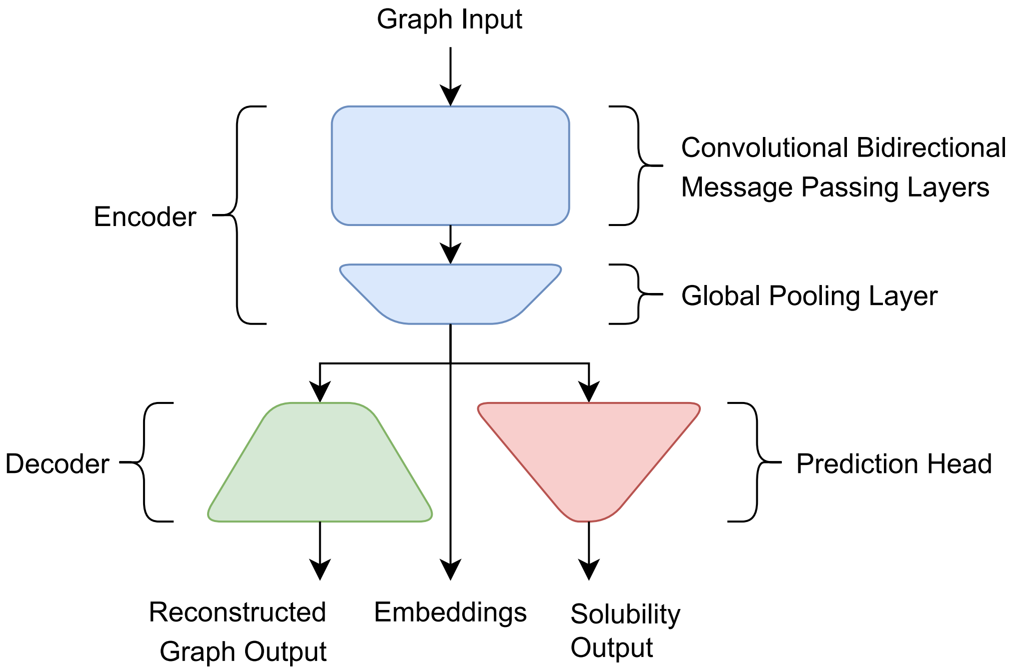

Figure 1: The model architecture.

The encoder is a convolutional bidirectional message passing neural network (MPNN), which is 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, followed by a global pooling layer that aggregates the node features into a single embedding vector of size 16 for each graph. This embedding vector is an information-dense representation of the molecular graph. A 2D projection of the embedding space with corresponding solubility values is shown in Figure **??**, visualized using both t-SNE [?] and UMAP [?].

The decoder consists of two fully connected layers that reconstruct node and edge features from the embedding vectors. The prediction head also employs two fully connected layers that map the same embedding vectors to the predicted solubility value, creating a multi-task learning architecture.

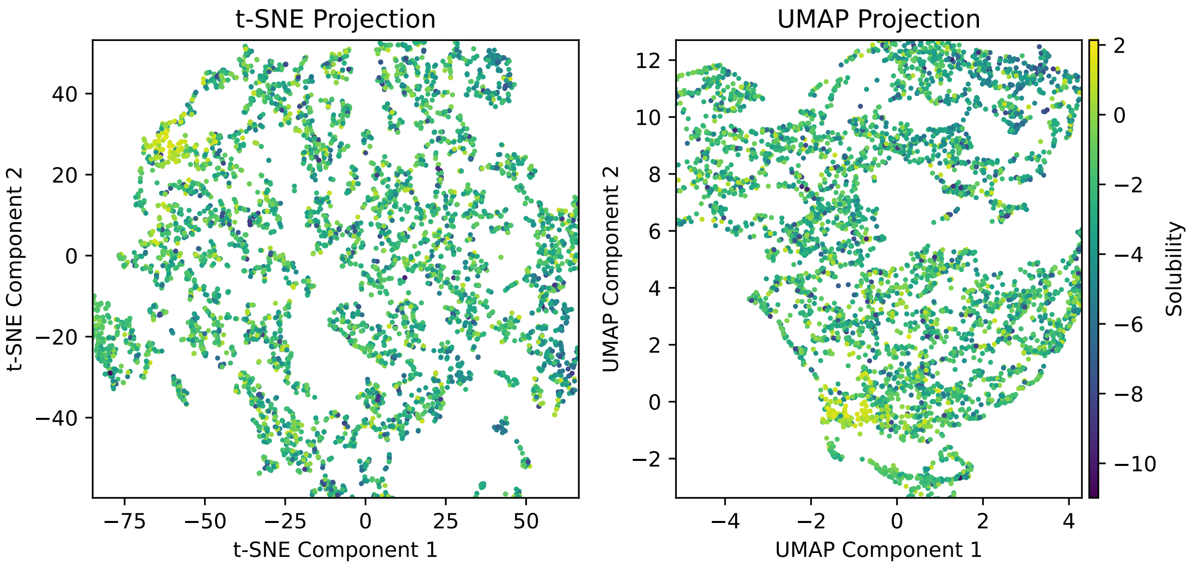

Figure 2: 2D projection of the embedding vectors with solubility.

2.3 Training

Following our architecture design, the model can be trained on two tasks. The first one is a self-supervised task, were node and edge features are denoised and reconstructed. For this, the features were pertrubed by randomly flipping a percentage of the one-hot encoded node and edge features, as well as randomly deleting edges. The encoder then learns to create an information-dense representation in form of the embedding vector, 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 a representation, from which the preidtcion head learns to predict the solubility value.

Following established approaches in the literature, both tasks are trained simultaneously by combining their respective losses [?, ?]. We implemented this by summing the denoising and prediction losses, and also experimented with weighted combinations of the task-specific losses, though this did not yield performance improvements.

The model was trained for 25 epochs with a batch size of 1024 using the Adam optimizer. Hyperparameters were optimized on the validation set. Different regularization strategies were applied to each component: the decoder used higher weight decay of 10^{-2} and a dropout of 0.4 to prevent overfitting on the self-supervised task, while the encoder and prediction head used more a moderate weight decay of 10^{-4} and a dropout of 0.2. The learning rate was set to $9.5 \cdot 10^{-3}$ and training was performed on a single NVIDIA GeForce GTX 960M GPU.

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 $1.2 \cdot 10^{-3}$ and 5 steps, respectively. This process shifts the embedding vectors of test samples towards the training set distribution, theoretically improving generalization.

The distribution shift and effect of TTA can be seen in the 2D projections of the embedding space shown in Figures **??** and **??**. Interestingly, while t-SNE shows a clear shift of the test and validation sets towards the training set, UMAP does not show any shift. This discrepancy likely arises from the differences between these dimensionality reduction techniques: t-SNE focuses

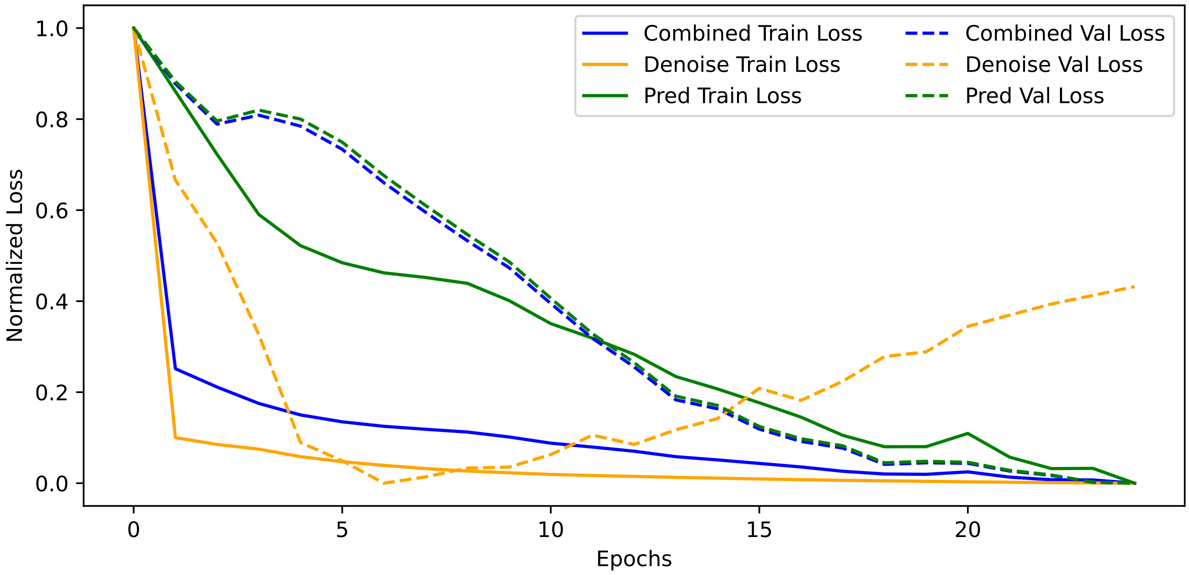

Figure 3: Training and validation losses.

on preserving local neighborhood structure and can amplify small differences, while UMAP prioritizes preservation of global structure. This suggests that the effects of TTA may be subtle and local for these particular datasets.

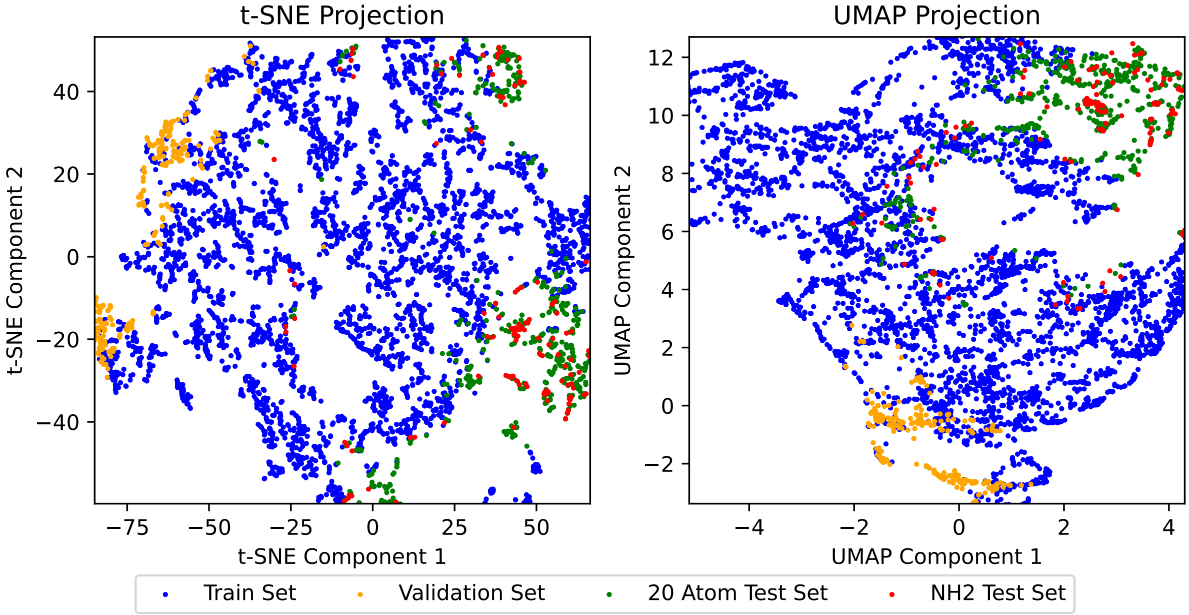

Figure 4: 2D projection of the embedding vectors with sets.

3 Results

As shown in Table ??, TTA did not improve the performance of the model on either test set, which is surprising given that the test set centroids clearly shift towards the training set centroid after TTA in the t-SNE projections, as shown in Figure ??. However, the absence of any such shifts in UMAP projections suggests that these changes may be more subtle, which could explain

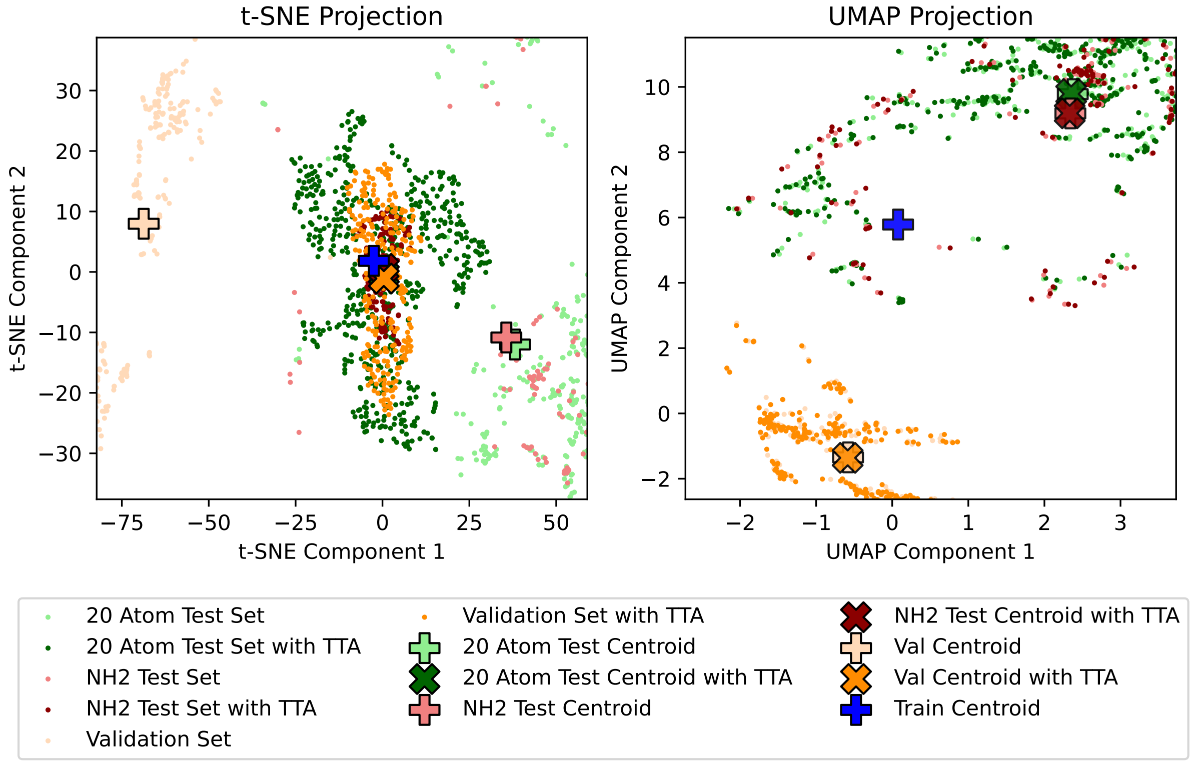

Figure 5: 2D projection of the embedding vectors with set centroids.

Model	**Validation RMSE**	**Test RMSE (20 Atom set)**	**Test RMSE (NH2 set)**
Model without TTA	1.0649	2.2453	2.3517
Model with TTA	1.0649	2.2451	2.3517
Reference Model	2.1926	2.6450	2.0751

Table 1: Performance comparison of different model configurations.

the lack of performance improvement.

For comparison, a reference model trained solely on the prediction task without any SSL was also included. The results demonstrate that SSL training clearly improves prediction performance, with the SSL-trained models achieving substantially lower RMSE values across both the validation set and the 20 Atom test set. Interestingly, the reference model performed better on the NH2 test set, suggesting that the amino group distribution shift may be particularly challenging for the SSL approach used in this study.

4 Code

The code for this project is available at github.com/p0017/Molecular-Test-Time-Adaptation under the GPL-3.0 license.

5 Conclusion

This study implemented test-time adaptation (TTA) for graph neural networks (GNNs) and applied it to molecular solubility prediction. While TTA successfully shifted test sample embeddings towards the training distribution in t-SNE visualizations, this did not translate into improved predictive performance. The discrepancy between t-SNE and UMAP visualizations suggests that the distribution shifts may be more subtle and local. Future work could explore alternative self-supervised learning (SSL) tasks, different adaptation strategies, or investigate TTA effectiveness on datasets with more pronounced 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] 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(q12025)release,$2025.

[3] 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.

[4] 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.

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

[6] 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.

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

[8] 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.