Superior Molecular Representations from Intermediate Encoder Layers

Luis Pinto
pinto.luisc@gmail.com

Abstract

Pretrained molecular encoders have become indispensable in computational chemistry for tasks such as property prediction and molecular generation. However, the standard practice of relying solely on final-layer embeddings for downstream tasks may discard valuable information. In this work, we first analyze the information flow in five diverse molecular encoders and find that intermediate layers retain more general-purpose features, whereas the final-layer specializes and compresses information. We then perform an empirical layer-wise evaluation across 22 property prediction tasks. We find that using frozen embeddings from optimal intermediate layers improves downstream performance by an average of 5.4%, up to 28.6%, compared to the final-layer. Furthermore, finetuning encoders truncated at intermediate depths achieves even greater average improvements of 8.5%, with increases as high as 40.8%, obtaining new state-of-the-art results on several benchmarks. These findings highlight the importance of exploring the full representational depth of molecular encoders to achieve substantial performance improvements and computational efficiency. The code is made publicly available at https://github.com/luispintoc/Unlocking-Chemical-Insights.

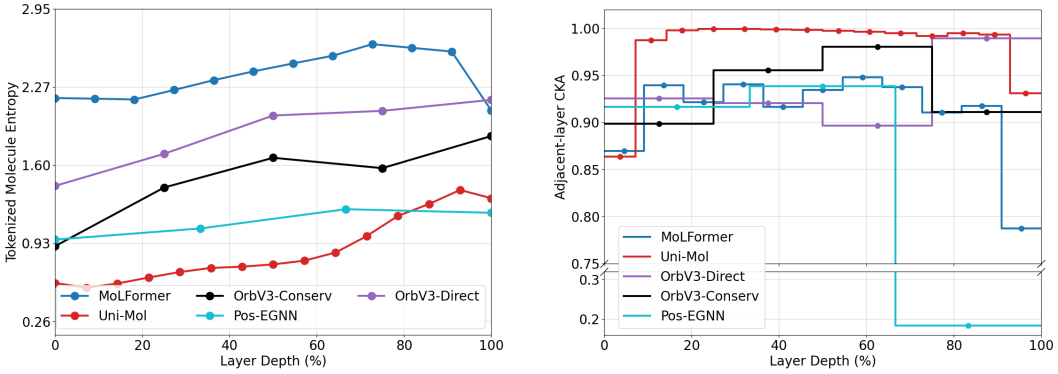
1 Introduction

Deep learning has reshaped molecular science, where pretrained molecular encoders are essential tools for applications from virtual screening in drug discovery to designing novel materials with desired properties [1, 2, 3]. These encoders, built on architectures such as Transformers [4] and Graph Neural Networks (GNNs) [5, 6], learn rich representations of molecular structures that can be applied to a wide range of downstream predictive tasks. The common practice of extracting molecular representations from the final encoder layer, while simple and widely adopted, rests on the implicit assumption that this layer consistently provides the most informative and task-relevant features.

However, this assumption is increasingly challenged by findings from Natural Language Processing (NLP) and Computer Vision, where intermediate layers often encode richer, more generalizable, or task-specific information, leading to significant performance gains [7, 8, 9, 10]. Despite the unique data modalities (e.g. molecular graphs, 3D conformers) and distinct pretraining objectives inherent to cheminformatics, systematic layer-wise studies of molecular encoders have been notably scarce. This leaves a critical question unanswered: are we overlooking superior molecular representations by adhering to the final-layer convention in chemistry?

Our work consists of a two-stage analysis across five diverse pretrained encoders. First, we map the information flow through their hidden layers, and second, we empirically evaluate the performance of these layer-wise representations on a suite of downstream tasks.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI4Mat-NeurIPS 2025.



(a) Tokenized-molecule Entropy

(b) Adjacent-layer CKA

Figure 1: Left: Tokenized-molecule entropy rises with depth but falls at the final block for MolFormer, Uni-Mol and PosEGNN, indicating compression; Orb models maintain higher spread. Right: Adjacent-layer CKA shows small changes across interior layers and a pronounced last-step change for most models. Depth is normalized from first encoder block (0%) to last (100%).

Our main contributions are as follows:

1. We use two label-free probes, tokenized-molecule entropy and adjacent-layer linear centered kernel alignment (CKA), to characterize information flow within molecules and across model depth. Our analysis reveals that while the middle layers remain stable and capture general, transferable features, the pronounced change in the final layer signals both information compression and specialization for the pretraining task.
2. We conduct an empirical study evaluating frozen embeddings across 22 absorption, distribution, metabolism, excretion, and toxicity (ADMET) tasks. We demonstrate that in over 81% of model-task combinations, an intermediate layer outperforms the conventionally used final-layer, yielding an average increase of 5.4% in downstream performance.
3. We further show that these advantages persist when the encoders are partially finetuned. For the same set of tasks, we observe that finetuning up to an optimal intermediate layer leads to notable improvements over finetuning the complete encoder in over 71% of cases, with an average gain of 8.5%.

2 Related Work

2.1 Molecular Encoders

Molecular encoders are deep learning models designed to transform chemical structures into fixed-length numerical vectors, also known as embeddings, that can be used for downstream tasks such as property prediction, molecular generation, or virtual screening. These models play a central role in computational chemistry and drug discovery by enabling machine learning algorithms to efficiently capture the relevant chemical, structural, and physical information inherent to molecules. The effectiveness of a molecular encoder is critically dependent on both the representational richness of its learned embeddings and its ability to generalize across diverse chemical spaces.

Molecular encoders are generally differentiated by two main aspects: the type of molecular input they process, for example SMILES [11] strings, molecular graphs, or 3D conformers; and the underlying computational architecture, which could be Transformers, Graph Neural Networks (GNNs), or other specialized architectures.

Transformer-based sequence models such as ChemBERTa [12], ChemGPT [13], and MolFormer [14] are pretrained on textual representations of molecules and apply masked language modeling [15] or autoregressive objectives [16] to learn general purpose embeddings. ChemBERTa and MolFormer operate on tokenized SMILES strings, whereas ChemGPT is trained on SELFIES [17], a robust string grammar that guarantees valid molecules. These models inherit the architectural benefits of NLP transformers, including multi-head self-attention and position embeddings.

Hybrid 2D/3D transformer encoders like Uni-Mol [18] extend transformer architectures to directly incorporate both 2D topological information and 3D atomic coordinates. Uni-Mol leverages pairwise distance matrices and spatial positional encodings to inject geometric priors, enabling the model to learn both conformation-aware and topology-sensitive representations. Its architecture retains the global receptive field of transformers while being sensitive to the spatial layout of molecules.

Graph Neural Networks for 2D molecular graphs represent a distinct branch of molecular representation learning. These models operate directly on the 2D graph structure of molecules, where atoms are nodes and bonds are edges. Through iterative message passing steps, GNNs aggregate information from neighboring atoms and bonds to learn representations that capture molecular topology and chemical features. Models like the Directed Message Passing Neural Network (D-MPNN) [19] are often trained with supervised learning on specific molecular properties. Others, such as MolCLR [20], employ self-supervised learning strategies, like contrastive learning between positive and negative molecular pairs, to learn general purpose embeddings.

GNN-based 3D models employ graph neural networks to process three dimensional molecular structures. Some of these models incorporate equivariant architectures to inherently respect physical symmetries, while others may utilize data augmentation techniques to learn these invariances. A specific subset of these 3D GNN models includes machine learning force fields [21, 22], which are trained to predict molecular energies and forces from DFT calculations [23]. For instance, MACE [24] and Pos-EGNN [25] are examples that use an equivariant architecture to predict such properties. In contrast, Orb models [26] do not rely on equivariant architectures but instead leverage augmentations.

2.2 Layer-Wise Probing in Language and Vision Models

Most relevant to our study is the recent work of Skean et al. (2025) [27], who performed the most comprehensive analysis to date of intermediate-layer representations in deep neural networks. Their layer-by-layer investigation across transformer and state-space models in both language and vision domains revealed that intermediate layers often exhibit higher utility than final ones across a range of probing tasks. Using a unified framework integrating metrics from information theory, geometry, and invariance, they demonstrated that the most informative representations, outperforming final-layer embeddings by up to 16% on 32 tasks, tend to emerge midway through the model. At this stage, task-relevant signals are preserved while over-specialization and noise accumulation, potentially more prevalent in final-layers, have not yet occurred. Their work critically challenges the widely adopted practice of extracting representations solely from the final-layer and uncovers how different architectural and training paradigms shape internal information flow and compression.

Despite these advances, layer-wise representational analysis has not been applied to molecular encoders, which differ from NLP and vision models in both data modality and pretraining objectives. Molecular encoders often combine structural and chemical priors, and their performance is often assessed using challenging benchmarks that include both regression and classification tasks, operate with limited data, or employ scaffold splits to test distinct generalization capabilities. Yet nearly all such models, from SMILES transformers to equivariant GNNs, default to using the final-layer as the molecular embedding, without examining the structure or quality of intermediate representations.

2.3 Surrogate Evaluators for Embedding Quality

Evaluating the quality of learned representations without resorting to computationally expensive end-to-end finetuning is crucial for efficient model development and analysis. This has led to the use of various surrogate models to probe the information encoded in frozen embeddings. These surrogates provide an estimate of the downstream utility across different tasks.

Linear probes, such as logistic regression or linear regression, are frequently used to assess the linear separability of concepts within embedding spaces [28]. More complex, tree-based ensemble models such as Random Forests or Gradient Boosted Decision Trees (e.g., XGBoost [29], LightGBM [30], CatBoost [31]) are often effective when applied to frozen embeddings [32, 33]. In some cases, small Multi-Layer Perceptrons (MLPs) are also trained as probes [34, 35]. More recently, models leveraging in-context learning principles have demonstrated significant promise. TabPFN [36], built upon a transformer architecture and pretrained on vast amounts of synthetic tabular data, can make accurate predictions without requiring task-specific gradient updates or hyperparameter tuning. These surrogate evaluators enable efficient assessment of learned embeddings by pairing powerful pretrained

encoders with lightweight predictors. From linear models to in-context transformer models, they offer a scalable alternative to finetuning, balancing speed and predictive reliability for downstream evaluation.

3 Methods

This section details the molecular encoder architectures investigated, the benchmark datasets and evaluation metrics employed, the information flow analysis and the experimental protocols for evaluating layer-wise representations through both frozen embeddings and finetuning.

3.1 Molecular Encoder Models

Our study investigates the layer-wise representations from five diverse pretrained molecular encoder architectures, selected to cover various input modalities and model designs relevant to molecular representation learning. These include:

- MolFormer [14]: A transformer architecture pretrained on tokenized SMILES strings with a masked language modeling objective. This model consists of 12 layers.
- Uni-Mol [18]: We evaluated the first model in the Uni-Mol family, which extends transformer architectures to directly incorporate both 2D topological information and 3D atomic coordinates. It features 15 encoder layers.
- Orb Family [26]: We investigated two specific variants from the Orb family of machine learning force fields. They do not rely explicitly on equivariant message passing architectures but incorporate invariances through their data augmentation. The variants we used are:
  - Orb-v3-conservative-omat: This variant is trained with a "conservative" objective, where forces are derived as the negative gradient of the predicted energy. The checkpoint used was trained on the ab initio molecular dynamics subset of OMat24 [37]. It features 5 interaction layers.
  - Orb-v3-direct-mpa: This variant is trained with a "direct" objective to predict forces explicitly, alongside energy. This checkpoint was trained on the combination of MPTraj [38] and Alexandria (PBE) [39] datasets. It also features 5 interaction layers.
- Pos-EGNN [25]: A Position-based Equivariant Graph Neural Network (Pos-EGNN) that functions as a machine learning force field. This foundation model for chemistry and materials utilizes equivariant GNNs operating directly on 3D molecular conformers. Representations were extracted from its 4 equivariant message passing blocks, which we treat as distinct layers in our analysis.

To generate 3D conformations for all molecules, we employed the ETKDG algorithm [40]. These conformations were subsequently optimized using the MMFF94 force field [41] to ensure physically plausible geometries before input into models requiring 3D coordinates. It should be noted that fewer than five data points per task ($< 0.5\%$ of the total datasets) did not yield valid 3D conformers using this protocol, and these instances were excluded from downstream analysis.

Finally, we emphasize that while the Orb models and Pos-EGNN are machine learning force fields trained on inorganic datasets, we investigated their learned representations here in an organic property prediction context for the first time. As these models were not trained for standard QSAR or ADMET prediction, our evaluation provides insight into the generality and transferability of their intermediate representations beyond their original design objectives.

3.2 Benchmark and Evaluation Protocol

We evaluated molecular encoder representations on the TDCommons (TDC) benchmark [42], a curated suite of 22 ADMET-related tasks spanning both regression and classification settings. Each task requires the prediction of a single target, a specific physicochemical or biological property, using molecular structures as input. ADMET properties critically influence the development and safety profiles of pharmaceutical compounds; thus, maximizing representational quality has tangible benefits for reducing experimental costs and enhancing predictive accuracy in drug discovery [43, 44].

To address the challenge of generalizing to novel chemical scaffolds, TDC provides scaffold-based data splits that partition each dataset into training, validation, and testing sets. This ensures that compounds in the test set have distinct core structures from those in the training set, enabling a rigorous assessment of a model's capacity to extrapolate to unseen chemical space.

We aggregated the training partitions across tasks to construct an unlabeled corpus for the information flow analysis. For the empirical experiments, we used the prescribed splits per task without modification. Following TDC benchmark guidelines, performance for each task was quantified using its designated metric. For regression tasks, this metric was either Mean Absolute Error (MAE) or Spearman Rank Correlation Coefficient, while for classification tasks, it was either the Area Under the Receiver Operating Characteristic curve (AUROC) or the Area Under the Precision-Recall Curve (AUCPR). Thus, our comprehensive evaluation framework employs these four performance metrics. Further details on each task are provided in Appendix A.

3.3 Information Flow Probes

We use two label-free probes to diagnose how information evolves across layers and to test the hypothesis that the final-layer is often over-specialized for pretraining. Let x_i denote a molecule and ℓ an encoder layer. The token matrix is $H_{i,\ell} \in \mathbb{R}^{T_i \times d}$, where T_i is the number of valid tokens/atoms for molecule x_i and d is the hidden dimensionality of layer ℓ. To derive a single vector representation per molecule from each layer, we followed the conventions established in the original publications of the respective models. For instance, MolFormer uses mean pooling over token embeddings, while Uni-Mol utilizes the representation of the special [CLS] token. For GNN-based models such as Pos-EGNN and the Orb variants, mean pooling over all scalar node embeddings was employed to obtain the graph-level representation.

Tokenized-molecule entropy. We first quantify the diversity of within-molecule token signals with a matrix-based entropy [45]. For each x_i and layer ℓ, we form the token Gram $K_{i,\ell} = H_{i,\ell} H_{i,\ell}^\top$, take its eigenvalues $\{\lambda_t\}$, normalize $p_t = \lambda_t / \sum_u \lambda_u$, and average the Shannon entropy of this spectrum across molecules:

$$\mathrm{TME}(\ell) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \Big(- \sum_t p_t \log p_t \Big).$$

Higher values indicate that token embeddings are spread across many principal directions, expressing greater variety and lower redundancy. Lower values indicate that variance concentrates in a small number of directions, reflecting higher redundancy and representational compression, a collapse towards a low-rank subspace.

Adjacent-layer CKA on pooled molecule vectors. We assess changes in the pooled space by comparing adjacent layers using linear centered kernel alignment (CKA) [46]. For two adjacent layers ℓ and $\ell{+}1$, we form X by stacking, row-wise, the pooled molecule vectors from layer ℓ (one row per molecule), and likewise Y from layer $\ell{+}1$. We then center each matrix across examples by subtracting the corresponding row-mean vector, yielding $\tilde{X}$ and $\tilde{Y}$. The centered second moments are then defined as

$$S_{xx}^{(\ell)} = \tilde{X}^\top \tilde{X}, \qquad S_{yy}^{(\ell+1)} = \tilde{Y}^\top \tilde{Y}, \qquad S_{xy}^{(\ell,\ell+1)} = \tilde{X}^\top \tilde{Y}.$$

And the linear CKA between layers is

$$\mathrm{CKA}_{\ell \to \ell+1} = \frac{\|S_{xy}^{(\ell,\ell+1)}\|_F^2}{\|S_{xx}^{(\ell)}\|_F \, \|S_{yy}^{(\ell+1)}\|_F} \in [0,1],$$

Values of $\mathrm{CKA}_{\ell \to \ell+1}$ close to 1 indicate that consecutive layers produce highly similar molecule embeddings, while values closer to 0 indicate larger geometric changes between layers. We report adjacent-layer CKA across depth to quantify, in a label-free and scale-free manner, how representation changes attenuate or persist throughout the encoder.

3.4 Evaluating Frozen Layer-wise Embeddings

To assess the predictive utility of representations from different layers, we use the hidden states from every encoder layer as frozen molecule embeddings for each TDC task. As explained in Section

3.3, we obtained a single vector representation for each molecule at every layer by following the conventions outlined in the original publications of the corresponding models.

These fixed-length embeddings then served as input to TabPFNv2 [47], hereafter simply TabPFN, a transformer-based in-context learning model pretrained on a vast array of synthetic tabular data. We selected TabPFN as our lightweight downstream predictive model for the following key reasons:

1. **No Hyperparameter Tuning:** TabPFN makes predictions for new tasks in a single forward pass without requiring task-specific hyperparameter optimization. This was crucial for our study, as it allows for a fair comparison of embedding quality across different layers and models, minimizing the confound of surrogate model tuning.

2. **Strong Validated Performance:** TabPFN has demonstrated state-of-the-art performance on diverse small to medium-scale tabular classification and regression tasks, making it a robust choice for efficiently probing embedding utility. Furthermore, as detailed in Appendix B, our own preliminary experiments confirmed that its performance when using cheminformatic features as input is comparable to that of well-established tree-based models on the same tasks, further validating its suitability as a reliable and efficient evaluator when applied to learned embeddings in this analysis.

For every TDC task, we retrieved the embeddings produced by each individual layer, trained TabPFN on the scaffold-split training portion of those embeddings and their labels, generated predictions for the matching test embeddings, and scored them with the metrics from Section 3.2.

3.5 Finetuning with Layer-wise Representations

We also performed a comprehensive finetuning study across all pretrained molecular encoders and TDC tasks. For each encoder and task, we extracted molecular representations from individual encoder layers, using the same pooling strategy as Section 3.4, and fed them into a task-specific prediction head. This prediction head consisted of two linear layers: the first projected to a hidden dimension, followed by a SiLU activation [48], 10% dropout, and a final linear layer projecting to the output dimension.

For each specific encoder layer being evaluated on each task, key hyperparameters, namely the prediction head's hidden dimension selected from {32, 256} and the learning rate for the AdamW optimizer [49] selected from {1e-5, 2e-5, 5e-5, 1e-4, 2e-4}, were optimized via an independent hyperparameter search. Hyperparameter ranges were selected based on preliminary studies and align with configurations commonly used in the original publications of the respective models, effectively capturing performance variance without excessive computational demands.

During training, both the parameters of the encoder up to and including the selected layer and those of the prediction head were jointly finetuned; any encoder layers beyond the selected layer were excluded. Models were trained for 50 epochs with a fixed batch size of 64 and the learning rate was linearly warmed up over the first 5% steps and decayed over the remaining steps. Validation performance was monitored at each epoch, and the model checkpoint yielding the best validation metric was used for test set evaluation.

4 Results and Discussion

In this section, we begin with a label-free analysis of representation dynamics using the probes in Section 3.3. These depth-wise diagnostics reveal how token diversity and layer-to-layer geometry evolve inside each encoder. We then validate these findings by evaluating model performance across 22 TDC ADMET tasks in two settings. We first show that intermediate layer representations consistently outperform the final-layer when used as inputs to TabPFN. We then show that these advantages largely persist when the encoders are finetuned up to an optimal intermediate layer. Finally, we establish a correlation between these two evaluation paradigms, suggesting an efficient strategy for selecting optimal layers for finetuning.

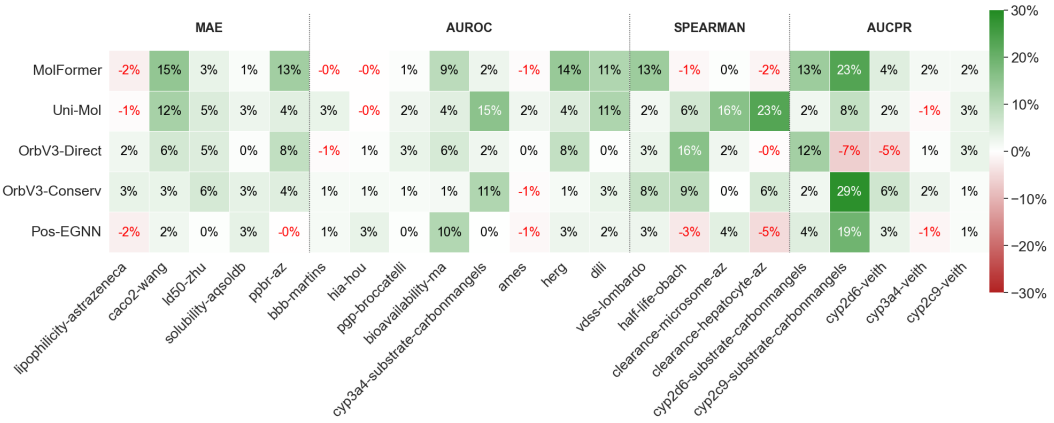


Figure 2: Percentage improvement in test metric of the best intermediate layer relative to the final-layer when evaluating frozen embeddings. Positive values mean the best non-final layer improves over the final-layer. Negative values mean the final-layer outperforms the best non-final layer.

4.1 Unsupervised Probes of Representation Dynamics

Figure 1 summarizes depth-wise behavior across models using two label-free probes. First, an analysis of token diversity using tokenized-molecule entropy reveals a pattern of end-of-encoder compression in several models. MolFormer and Uni-Mol exhibit a sharp drop in entropy in their final-layer, with Pos-EGNN showing a milder version of this effect. This drop indicates that the token representations collapse into a low-rank subspace, concentrating their variance into fewer principal directions. Such behavior is consistent with the final-layer becoming highly tailored to its pretraining objective. Conversely, both Orb variants maintain a broader spread across principal directions, indicating that token information stays more distributed instead of collapsing at the end.

Second, geometric stability between adjacent layers, captured by CKA, reveals that Uni-Mol sits near 1 across depth, indicating very similar pooled molecule embeddings from one block to the next. The remaining encoders live in the 0.90 to 0.95 range, which points to small but meaningful updates as depth increases. Most models then show a pronounced last-step drop in CKA signaling a large geometric remapping right before the output, that aligns with the pretraining task specialization and coincides with the token-level compression above. Viewed jointly, these trends support the view that, regardless of the specific architecture or pretraining objective, the final-layer is often over-specialized and does not always carry the most transferable features.

4.2 Superior Performance of Intermediate Layer Embeddings

We evaluate downstream utility by using each layer's hidden states as fixed feature embeddings with TabPFN. Figure 2 reports, for each model and task pair, the signed percent change of the best non-final layer relative to the final-layer, where green values indicate gains over the final-layer and red values indicate that the final-layer is better.

Across all evaluations, 81% of model and task combinations prefer a non-final layer. Negative values are present, although they are small in magnitude; when the final-layer wins, its average margin over the best non-final layer is about 1%. The choice of encoder influences these gains. MolFormer and Uni-Mol benefit the most from intermediate layers, with average improvements of 7.9% and 6.7%, respectively. In contrast, the comparatively shallow models show an average improvement of only $\sim$4%.

Moreover, while prior work in language and vision reports that the most informative representations often emerge midway through the model [27], our empirical study did not identify a universally optimal depth for molecular encoders. Instead, the ideal layer varies by architecture and ADMET task, as illustrated by the full performance curves in Appendix C. This pattern aligns with the information flow findings. Across the interior of each encoder, adjacent-layer CKA is high and nearly constant, indicating that successive layers produce very similar pooled molecule vectors; in practice this means many intermediate layers supply comparably good features, so no single depth consistently dominates.

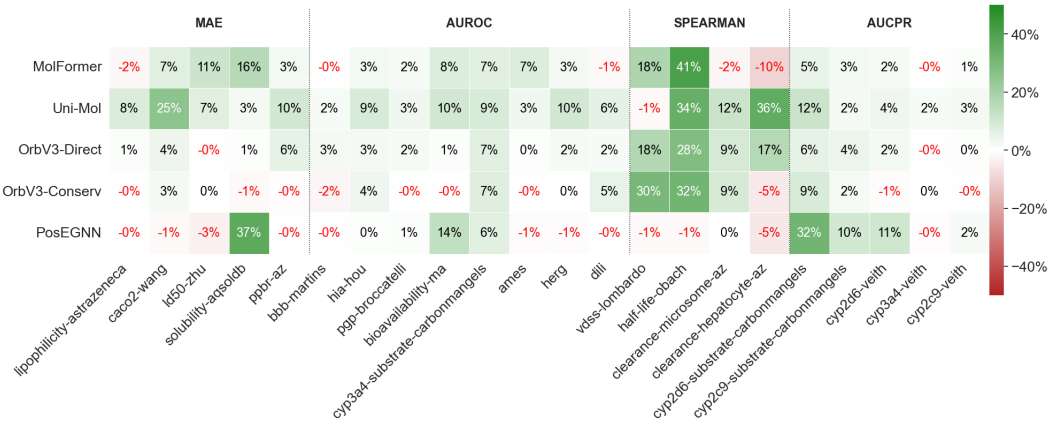


Figure 3: Percentage change in test metric achieved by finetuning up to the best intermediate layer compared to finetuning up to the final-layer. Positive values mean the best non-final layer improves over the final-layer. Negative values mean the final-layer outperforms the best non-final layer.

In contrast, between the final two layers we observe a pronounced CKA drop, and tokenized-molecule entropy shows a concurrent decrease, both consistent with a last-stage geometric remapping and increased compression tailored to the pretraining objective. In aggregate, these observations explain why intermediate layers frequently outperform the final-layer while the precise winning depth varies by model–task pair.

Task-specific benefits were also evident. For instance, *cyp2c9-substrate-carbonmangels* saw substantial improvements from intermediate layers, with average gains of 19.8%. Conversely, tasks like *clearance-hepatocyte-az* and *ames* showed limited advantage, with intermediate layers outperforming the final-layer in only two of the five models examined. Furthermore, using intermediate layers also yielded new state-of-the-art (SOTA) results, for example: Pos-EGNN Layer 0 on *hia-hou* (AUROC 0.994 vs. 0.989), Uni-Mol Layer 3 on *clearance-microsome-az* (Spearman 0.641 vs. 0.630), and Pos-EGNN Layer 3 on *dili* (AUROC 0.942 vs. 0.925) [50, 51].

4.3 Superiority of Intermediate Layers Persists with Finetuning

Next, we investigated whether the observed benefits of intermediate layers translate to a scenario where the encoder, up to a selected layer, and a task-specific prediction head are jointly finetuned. Figure 3 shows the relative difference in the test metric when finetuning to the optimal intermediate layer compared to finetuning to the final-layer, again with each cell representing a specific model-task pair and showing the numerical percentage gain. Layer-by-layer performance visualizations for all models and tasks are provided in Appendix C.

We observed pronounced benefits from employing intermediate layers even in this finetuning context. Averaged across all pairs, the optimal intermediate layer improves the test metric by 8.5%, with 71% of pairs showing gains. The gains were model-dependent but consistently substantial: Pos-EGNN saw the highest average improvement at 11.2%, closely followed by Uni-Mol at 9.9%, while MolFormer and OrbV3-Direct each experienced average gains of 8.4%. OrbV3-Conservative exhibited smaller, yet still notable, gains of 5.8%. Consistent with the information flow diagnostics and frozen embedding results, no single depth was best, which point to broadly similar interior representations and a specialized final block.

Tasks evaluated with Spearman correlation benefited the most from intermediate layers (e.g., *half-life-obach* +33.8%, *vdss-lombardo* +22%), although negatives do occur and *clearance-hepatocyte-az* shows both large gains and large drops across models, underscoring task-specific sensitivity to depth. Moreover, several intermediate layers deliver better results than prior SOTA benchmarks even without extensive hyperparameter tuning. For instance, Pos-EGNN Layer 3 on *lipophilicity-astrazeneca* (MAE 0.459 vs. 0.467), MolFormer Layer 8 on *ppbr-az* (MAE 7.221 vs. 7.526) and Uni-Mol Layer 13 on *cyp3a4-substrate-carbonmangels* (AUROC 0.688 vs. 0.662)[50, 51, 52, 53].

Beyond predictive improvements, leveraging intermediate layers substantially reduces the number of trainable parameters. Since the number of parameters scales nearly linearly with encoder depth,

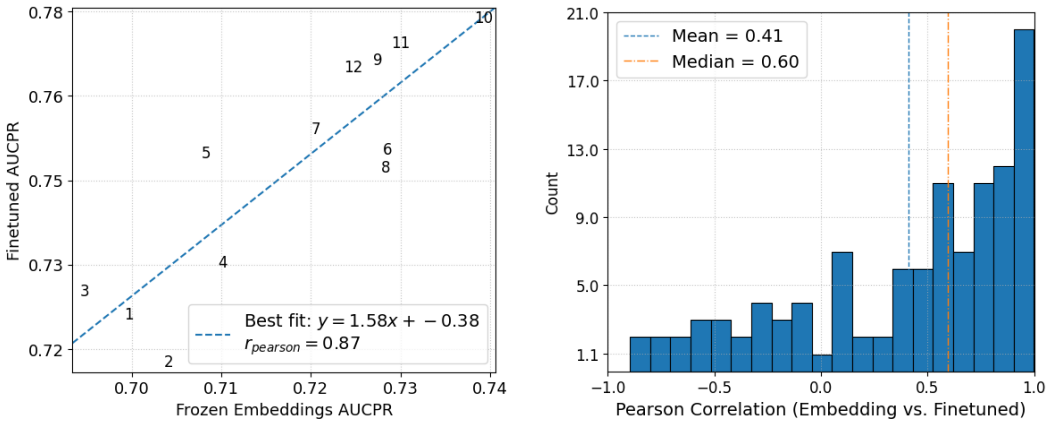


Figure 4: Left: Example of scatter plot of frozen embedding AUCPR vs. finetuned AUCPR for each MolFormer layer on task *cyp2c9-veith*. Each point is annotated with its corresponding layer number. Right: Histogram of embedding-to-finetuned correlations for all 110 model-task combinations.

stopping finetuning at an intermediate layer proportionally reduces the computational burden. For instance, MolFormer truncated at layer 8 of its 12 total layers prunes the final third of the network, decreasing trainable parameters by approximately 33%. This yields models that not only train faster but also require fewer computational resources, facilitating efficient experimentation and deployment without compromising, and frequently even enhancing, predictive performance.

4.4 Frozen Embedding Performance as a Proxy for Finetuning Layer Selection

For each encoder-task pair in the TDC, we computed the Pearson correlation between the layer-wise frozen embedding scores and the corresponding finetuned scores, thereby quantifying how well embedding performance anticipates finetuning gains. Figure 4 shows the correlation distribution. The left panel shows one representative scatter plot for MolFormer on the *cyp2c9-veith* task, where each data point corresponds to a layer, with a Pearson correlation of 0.87. This near linear relationship indicates that layers scoring highest in frozen embedding metric are typically the same layers that achieve the top metric after finetuning. Individual scatter plots for all model-task combinations are provided in Appendix D – H. The right panel aggregates this analysis across all 110 model–task pairs. The distribution is heavily skewed toward positive correlations: the median is 0.60, signaling a strong alignment for half of the model-task pairs and indicating frozen embedding scores can serve as a practical first-order filter to prioritize layers for finetuning, substantially reducing the search space.

5 Conclusion

This work systematically challenges the common practice of relying on final-layer representations from pretrained molecular encoders. Our comprehensive study, spanning five diverse molecular encoders and 22 ADMET tasks, demonstrates that intermediate layers frequently offer superior downstream performance. This superiority is evident when evaluating frozen embeddings, where in over 81% of model-task combinations an intermediate layer outperformed the final-layer, yielding an average 5.4% performance increase. With finetuning, intermediate layers led to notable improvements in over 71% of cases, with an average gain of 8.5%, alongside significant savings in parameters and compute time. These empirical gains are supported by our label-free analysis, which shows that the final-layer's specialization often leads to representational compression, leaving intermediate layers with more general transferable features.

While our findings are encouraging, we acknowledge certain limitations. Although constrained by computational resources to five encoders and 22 ADMET tasks, extending this analysis to a broader set of molecular encoders and chemical domains could further validate our findings. Future work with additional architectures, multiple random seeds, and more exhaustive hyperparameter searches would provide deeper insights and more robust generalizations. Nevertheless, our results strongly advocate for a shift beyond the final-layer default, urging exploration of the rich representational landscape within molecular encoders.

References

[1] Garrett B. Goh, Nathan O. Hodas, Charles Siegel, and Abhinav Vishnu. Smiles2vec: An interpretable general-purpose deep neural network for predicting chemical properties, 2018.

[2] Maciej Sypetkowski, Frederik Wenkel, Farimah Poursafaei, Nia Dickson, Karush Suri, Philip Fradkin, and Dominique Beaini. On the scalability of GNNs for molecular graphs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[3] Thalita Cirino, Luis Pinto, Mateusz Iwan, Alexis Dougha, Bono Lučić, Antonija Kraljević, Zaven Navoyan, Ani Tevosyan, Hrach Yeghiazaryan, Lusine Khondkaryan, Narek Abelyan, Vahe Atoyan, Nelly Babayan, Yuma Iwashita, Kyosuke Kimura, Tomoya Komasaka, Koki Shishido, Taichi Nakamura, Mizuho Asada, Sankalp Jain, Alexey V. Zakharov, Haobo Wang, Wenjia Liu, Vladimir Chupakhin, and Yoshihiro Uesawa. Consensus modeling strategies for predicting transthyretin binding affinity from tox24 challenge data. *Chemical Research in Toxicology*, 38(6):1061–1071, 2025. PMID: 40371923.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[5] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[6] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[7] Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. Linguistic knowledge and transferability of contextual representations. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. Exploring concept depth: How large language models acquire knowledge and concept at different layers? In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 558–573, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[9] Wes Gurnee and Max Tegmark. Language models represent space and time. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Lucrezia Valeriani, Diego Doimo, Francesca Cuturello, Alessandro Laio, Alessio Ansuini, and Alberto Cazzaniga. The geometry of hidden representations of large transformer models, 2023.

[11] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988.

[12] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta: Large-scale self-supervised pretraining for molecular property prediction, 2020.

[13] Nathan Frey, Ryan Soklaski, Simon Axelrod, Siddharth Samsi, Rafael Gomez-Bombarelli, Connor Coley, and Vijay Gadepally. Neural scaling of deep chemical models. *ChemRxiv*, 2022.

[14] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12):1256–1264, Dec 2022.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[16] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[17] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100string representation. *Machine Learning: Science and Technology*, 1(4):045024, oct 2020.

[18] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3d molecular representation learning framework. In *The Eleventh International Conference on Learning Representations*, 2023.

[19] Xu Han, Ming Jia, Yachao Chang, Yaopeng Li, and Shaohua Wu. Directed message passing neural network (d-mpnn) with graph edge attention (gea) for property prediction of biofuel-relevant species. *Energy and AI*, 10:100201, 2022.

[20] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287, Mar 2022.

[21] Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of Chemical Physics*, 145(17):170901, November 2016.

[22] Oliver T. Unke, Stefan Chmiela, Huziel E. Sauceda, Michael Gastegger, Igor Poltavsky, Kristof T. Schütt, Alexandre Tkatchenko, and Klaus-Robert Müller. Machine learning force fields. *Chemical Reviews*, 121(16):10142–10186, 2021. PMID: 33705118.

[23] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133–A1138, Nov 1965.

[24] Ilyes Batatia, David Peter Kovacs, Gregor N. C. Simm, Christoph Ortner, and Gabor Csanyi. MACE: Higher order equivariant message passing neural networks for fast and accurate force fields. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[25] Rodrigo Neumann Barros Ferreira. Pos-egnn. `https://github.com/IBM/materials/tree/main/models/pos_egnn`, 2025. Accessed: 2025-05-18.

[26] Benjamin Rhodes, Sander Vandenhaute, Vaidotas Šimkus, James Gin, Jonathan Godwin, Tim Duignan, and Mark Neumann. Orb-v3: atomistic simulation at scale, 2025.

[27] Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models, 2025.

[28] G. Tripepi, K. J. Jager, F. W. Dekker, and C. Zoccali. Linear and logistic regression analysis. *Kidney International*, 73(7):806–810, Apr 2008.

[29] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: a highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc. event-place: Long Beach, California, USA.

[31] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support, 2018.

[32] Indra Priyadarsini, Vidushi Sharma, Seiji Takeda, Akihiro Kishimoto, Lisa Hamada, and Hajime Shinohara. Improving performance prediction of electrolyte formulations with transformer-based molecular representation model, 2024.

[33] Eduardo Soares, Zeynep Sumer, Emilio Vital Brazil, Dave Braines, and Richard L Anderson. Representing surfactants by foundation models. In *AI for Accelerated Materials Design - ICLR 2025*, 2025.

[34] Han Li, Ruotian Zhang, Yaosen Min, Dacheng Ma, Dan Zhao, and Jianyang Zeng. A knowledge-guided pre-training framework for improving molecular representation learning. *Nature Communications*, 14(1):7568, Nov 2023.

[35] Muhammad Arslan Masood, Samuel Kaski, and Tianyu Cui. Molecular property prediction using pretrained-bert and bayesian active learning: a data-efficient approach to drug design. *Journal of Cheminformatics*, 17(1):58, Apr 2025.

[36] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second, 2023.

[37] Luis Barroso-Luque, Muhammed Shuaibi, Xiang Fu, Brandon M. Wood, Misko Dzamba, Meng Gao, Ammar Rizvi, C. Lawrence Zitnick, and Zachary W. Ulissi. Open materials 2024 (omat24) inorganic materials dataset and models, 2024.

[38] Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002, 07 2013.

[39] Jonathan Schmidt, Noah Hoffmann, Hai-Chen Wang, Pedro Borlido, Pedro J M A Carriço, Tiago F T Cerqueira, Silvana Botti, and Miguel A L Marques. Machine-Learning-Assisted determination of the global Zero-Temperature phase diagram of materials. *Adv Mater*, 35(22):e2210788, April 2023.

[40] Sereina Riniker and Gregory A. Landrum. Better informed distance geometry: Using what we know to improve conformation generation. *Journal of Chemical Information and Modeling*, 55(12):2562–2574, 2015. PMID: 26575315.

[41] Thomas A. Halgren. Merck molecular force field. i. basis, form, scope, parameterization, and performance of mmff94. *Journal of computational chemistry*, 17(5-6):490–519, 1996.

[42] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Proceedings of Neural Information Processing Systems, NeurIPS Datasets and Benchmarks*, 2021.

[43] Stephanie A. Eytcheson and Igor V. Tetko. Which modern ai methods provide accurate predictions of toxicological end points? analysis of tox24 challenge results. *Chemical Research in Toxicology*, 0(0):null, 0. PMID: 40779334.

[44] H. MacDermott-Opeskin, J. Scheen, C. Wognum, J. T. Horton, D. West, A. M. Payne, et al. A computational community blind challenge on pan-coronavirus drug discovery data. *ChemRxiv*, 2025.

[45] Luis G. Sanchez Giraldo, Murali Rao, and Jose C. Principe. Measures of entropy from data using infinitely divisible kernels, 2014.

[46] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019.

[47] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 01 2025.

[48] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. Special issue on deep reinforcement learning.

[49] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[50] James H. Notwell and Michael W. Wood. Admet property prediction through combinations of molecular fingerprints, 2023.

[51] Gemma Turon, Jason Hlozek, John G. Woodland, Kelly Chibale, and Miquel Duran-Frigola. First fully-automated ai/ml virtual screening cascade implemented at a drug discovery centre in africa. *bioRxiv*, 2022.

[52] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, Andrew Palmer, Volker Settels, Tommi Jaakkola, Klavs Jensen, and Regina Barzilay. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388, 2019. PMID: 31361484.

[53] Kexin Huang, Tianfan Fu, Lucas M Glass, Marinka Zitnik, Cao Xiao, and Jimeng Sun. Deeppurpose: a deep learning library for drug–target interaction prediction. *Bioinformatics*, 36(22-23):5545–5547, 12 2020.

[54] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. PMID: 20426451.

[55] Peter Gedeck, Bernhard Rohde, and Christian Bartels. Qsar - how good is it in practice? comparison of descriptor sets on an unbiased cross section of corporate data sets. *Journal of Chemical Information and Modeling*, 46(5):1924–1936, 2006. PMID: 16995723.

[56] Nikolaus Stiefl, Ian A. Watson, Knut Baumann, and Andrea Zaliani. Erg: 2d pharmacophore descriptions for scaffold hopping. *Journal of Chemical Information and Modeling*, 46(1):208–220, 2006. PMID: 16426057.

[57] Greg Landrum and RDKit Contributors. RDKit: Open-source cheminformatics, 2025. This concept DOI covers all RDKit versions on Zenodo and always resolves to the latest release.

[58] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2019.

Appendix

A TDC-ADMET Task Summary

Table 1 summarizes the 22 tasks from the Therapeutics Data Commons ADMET benchmark suite used in our experiments. Each row corresponds to a distinct molecular property prediction task, with associated details including a brief description, the number of compounds available in the dataset, and the evaluation metric used.

Dataset	Description	# compounds	Metric
lipophilicity-astrazeneca	Measures the ability of a drug to dissolve in a lipid	4 200	MAE
caco2-wang	Estimates intestinal permeability	906	MAE
ld50-zhu	Indicates drug's lethal dose threshold	7 385	MAE
solubility-aqsoldb	Measures a drug's ability to dissolve in water	9 982	MAE
ppbr-az	lasma protein binding percentage influences drug delivery efficiency	1 614	MAE
bbb-martins	Drug penetration across the blood-brain barrier	1 975	AUROC
hia-hou	Human intestinal absorption impacting oral drug delivery	578	AUROC
pgp-broccatelli	P-glycoprotein inhibition affects drug bioavailability, resistance	1 212	AUROC
bioavailability-ma	Oral bioavailability determines systemic drug exposure	640	AUROC
cyp3a4-substrate-carbonmangels	CYP3A4 metabolizes drugs for bodily clearance	667	AUROC
ames	Mutagenicity assesses genetic damage via Ames test	7 255	AUROC
herg	hERG inhibition linked to cardiac safety risks	648	AUROC
dili	Drug-induced liver injury impacting drug safety	475	AUROC
vdss-lombardo	Volume of distribution reflects tissue drug concentration	1 130	SPEARMAN
half-life-obach	Half-life indicates duration of drug activity	667	SPEARMAN
clearance-microsome-az	Drug clearance measures rate of systemic elimination	1 102	SPEARMAN
clearance-hepatocyte-az	Drug clearance measures rate of systemic elimination	1 1020	SPEARMAN
cyp2d6-substrate-carbonmangels	CYP2D6 enzyme involved in drug metabolism	664	AUCPR
cyp2c9-substrate-carbonmangels	Octanol/water distribution coefficient of molecules	666	AUCPR
cyp2d6-veith	CYP2C9 catalyzes metabolism of various compounds	13 130	AUCPR
cyp3a4-veith	CYP3A4 oxidizes xenobiotics for drug clearance	12 328	AUCPR
cyp2c9-veith	CYP2C9 mediates oxidation of cellular compounds	12 092	AUCPR

Table 1: Evaluated datasets description.

B Validating TabPFN Against a Strong Tree-Based Baseline

To ensure that TabPFN is an adequate surrogate for assessing embedding quality, we compared it to a state-of-the-art gradient-boosted decision-tree model. Specifically, we followed the CatBoost pipeline described in [50]: each molecule is represented by the concatenation of ECFP-1024 [54], Avalon-1024 [55], ErG-315 [56], 200 descriptors from RDKit [57], and a 300-dimensional GIN embedding [58], yielding a 2863-feature input that achieved top-3 accuracy on 19/22 TDC-ADMET leaderboards.

Due to compute constraints, we evaluated TabPFN and CatBoost on only 16 of the 22 ADMET benchmarks, as shown in Figure 5. On these 16 tasks, TabPFN matched CatBoost's overall performance and even outperformed it on five tasks. Remarkably, TabPFN achieved this with a single forward pass, no hyperparameter tuning and far more features than it was pretrained on (a maximum of 500 features). This confirms our choice of TabPFN as a lightweight, hyperparameter-free probe: it isolates the impact of encoder representations without introducing variability from per-task model optimization.

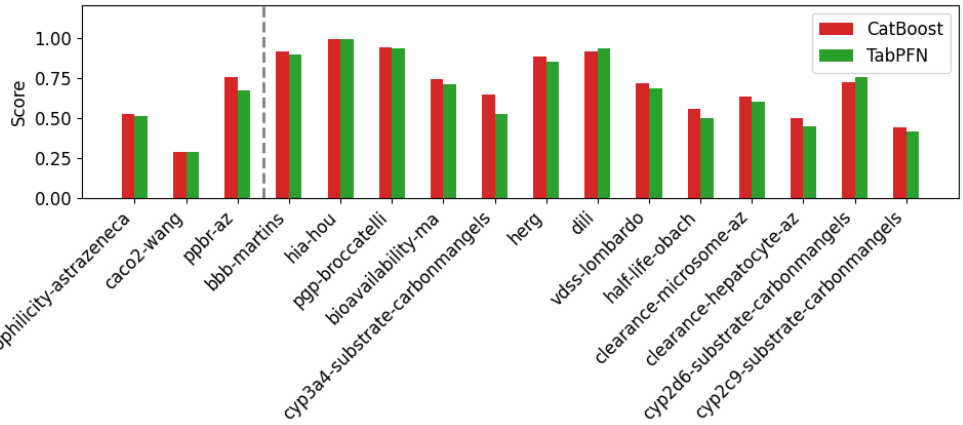


Figure 5: Comparison of CatBoost and TabPFN across a suite of ADMET benchmarks. The vertical dashed line separates the MAE-based tasks on the left (lower is better) from the remaining tasks on the right (higher is better). To display *ppbr-az* on a 0–1 scale, its MAE was divided by 10.

C Layer-wise Performance

Each figure in this appendix presents the complete layer-wise trajectories for all five encoders on a given TDC task. The left panel reports the frozen embedding performance obtained by freezing the encoder at successive depths and training a lightweight TabPFN surrogate on top; the right panel shows the outcome when the encoder is finetuned up to the same depth together with a task-specific prediction head. Finetuning hyperparameters for each model and layer were determined by conducting a limited grid search over a few learning rates and hidden layer sizes. Depth is expressed as a percentage of the total number of encoder blocks (0% = first encoder layer; 100 % = final block), and performance is plotted according to its task metric.

As expected, the two Orb variants, pretrained on inorganic data, generally perform worse than the other models across both evaluation paradigms, highlighting the significance of domain-aligned pretraining for small-molecule property prediction. However, Pos-EGNN, trained on a subset of the OrbV3-Direct dataset, consistently demonstrates strong performance, frequently matching or surpassing models like MolFormer and Uni-Mol. This surprising outcome indicates that the lower performance of the Orb variants may not result solely from domain mismatch, but also from insufficient hyperparameter tuning.

Comparing the two panels reveals that frozen embedding curves frequently, but not always, anticipate the relative ordering observed after fine-tuning. In some tasks (e.g. *lipophilicity-astrazeneca* and *ppbr-az*) the layer that minimizes/maximizes the given metric in the surrogate setting also yields the best or near best score after finetuning; in others, the correspondence weakens, suggesting that perhaps a broader hyperparameter search is needed. Additionally, it should be noted that for a small percentage

of cases, such as *vdss-lombardo* and *half-life-obach*, finetuning provided lower performance than using frozen embeddings. This variability likely contributes to the long tail observed in the histogram of Pearson correlations (Figure 4).

Finally, no single region of the network consistently outperforms the rest. Optimal depths vary not only across encoders but also across tasks within the same encoder. This heterogeneity reinforces the utility of our two-step protocol: inexpensive frozen embedding evaluation first narrows the set of candidate layers, after which targeted finetuning can focus computational resources on the most promising depths.

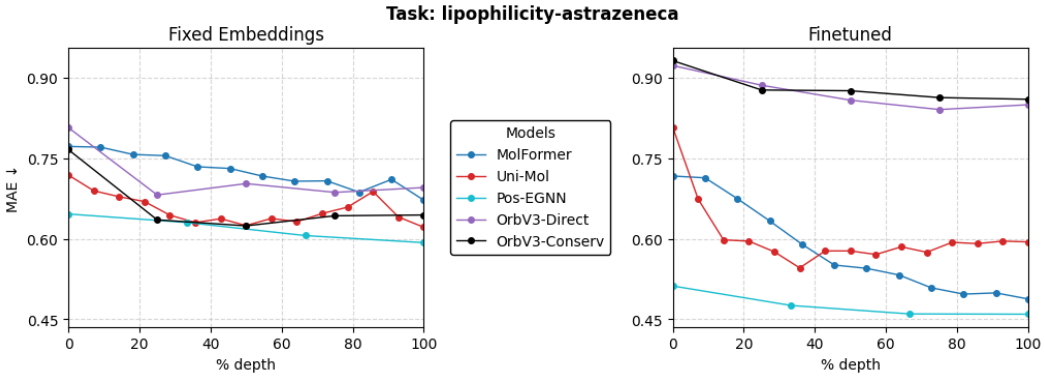


Figure 6: MAE (↓, the lower the better) on the ***lipophilicity-astrazeneca*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

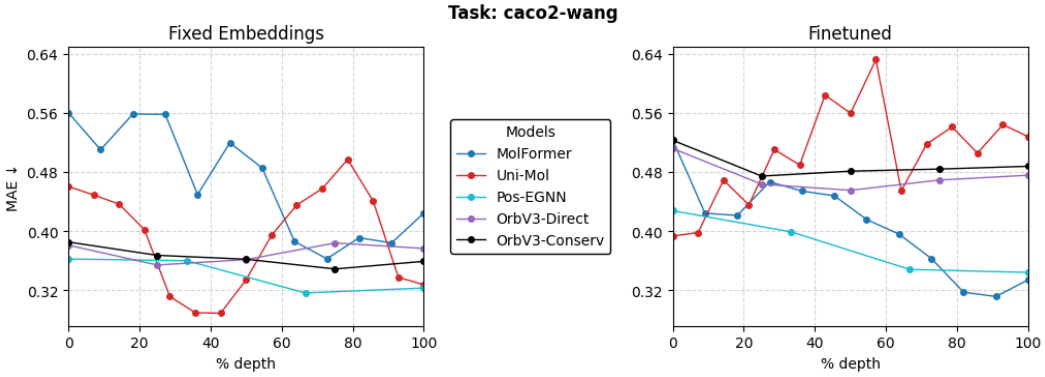


Figure 7: MAE (↓, the lower the better) on the ***caco2-wang*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

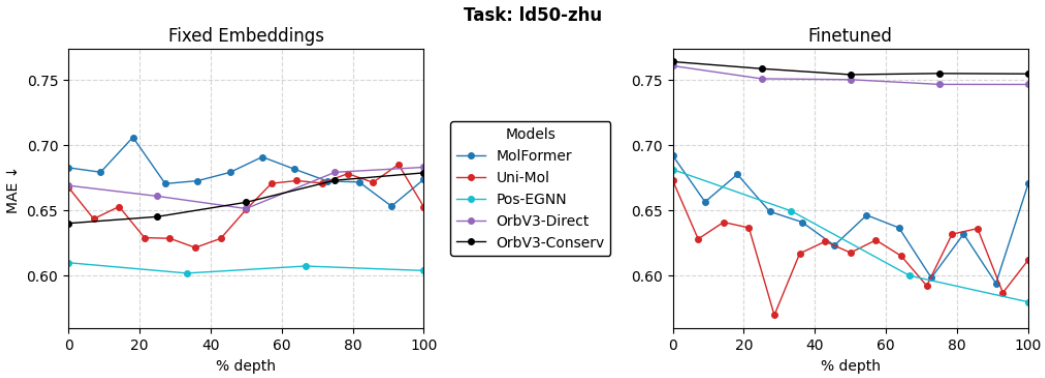


Figure 8: MAE (↓, the lower the better) on the ***ld50-zhu*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

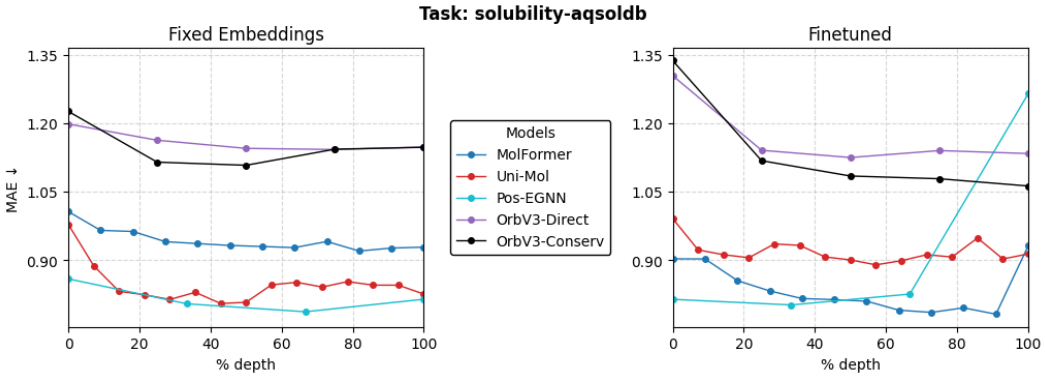


Figure 9: MAE (↓, the lower the better) on the ***solubility-aqsoldb*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

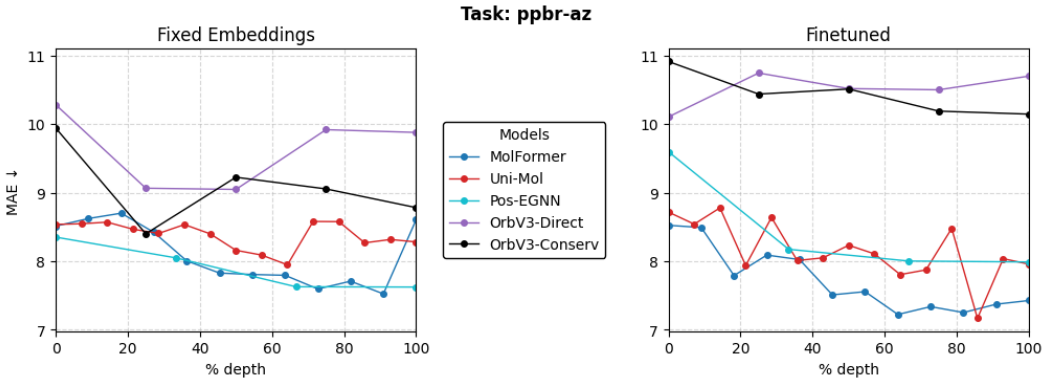


Figure 10: MAE (↓, the lower the better) on the ***ppbr-az*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

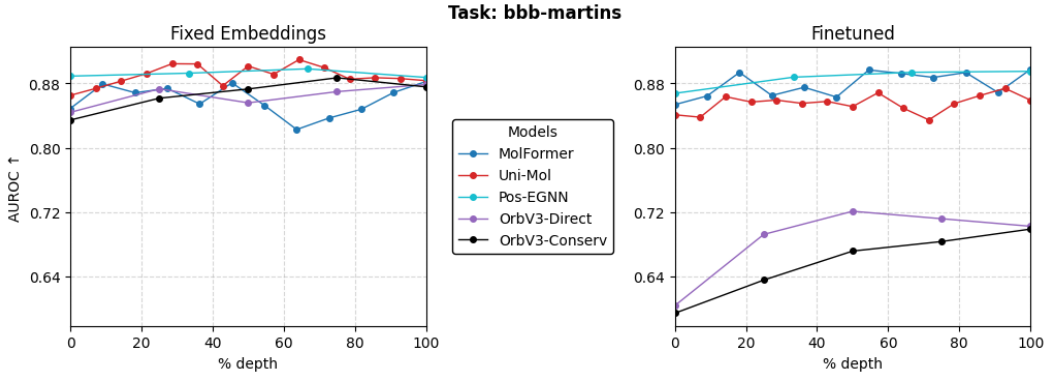


Figure 11: AUROC (↑, the higher the better) on the ***bbb-martins*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

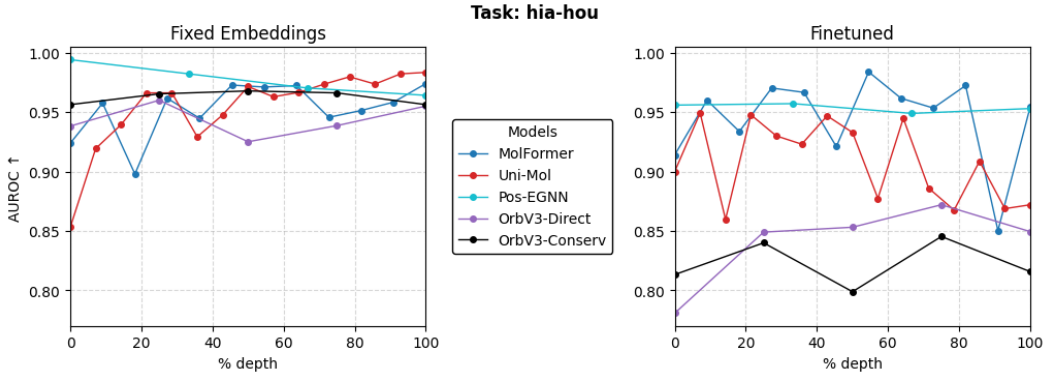


Figure 12: AUROC (↑, the higher the better) on the ***hia-hou*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

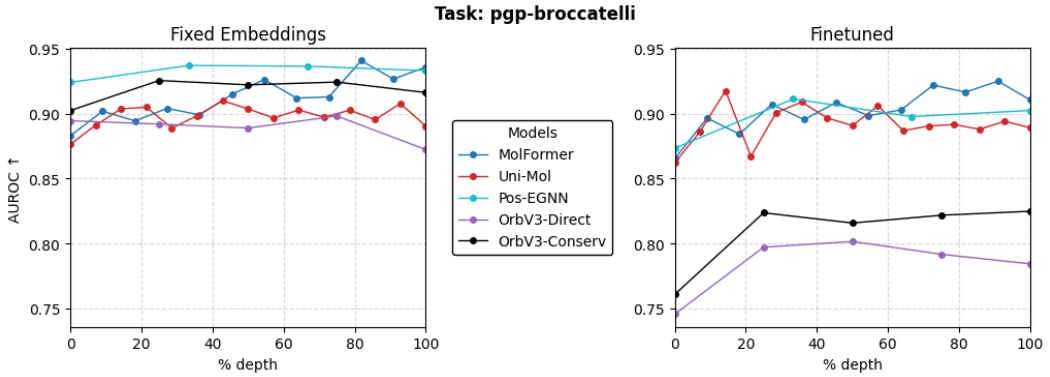


Figure 13: AUROC (↑, the higher the better) on the ***pgp-broccatelli*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

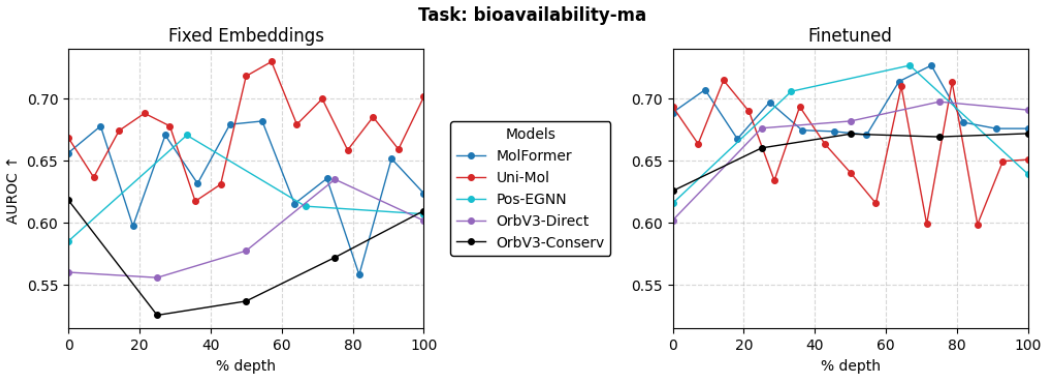


Figure 14: AUROC (↑, the higher the better) on the ***bioavailability-ma*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

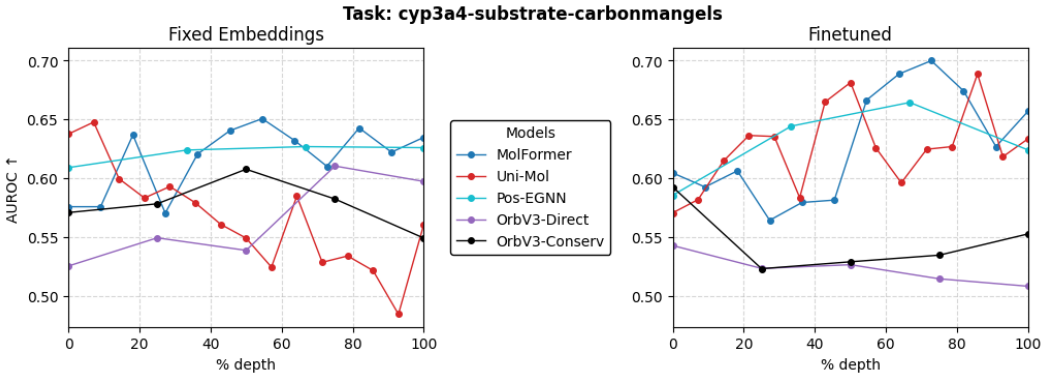


Figure 15: AUROC (↑, the higher the better) on the ***cyp3a4-substrate-carbonmangels*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

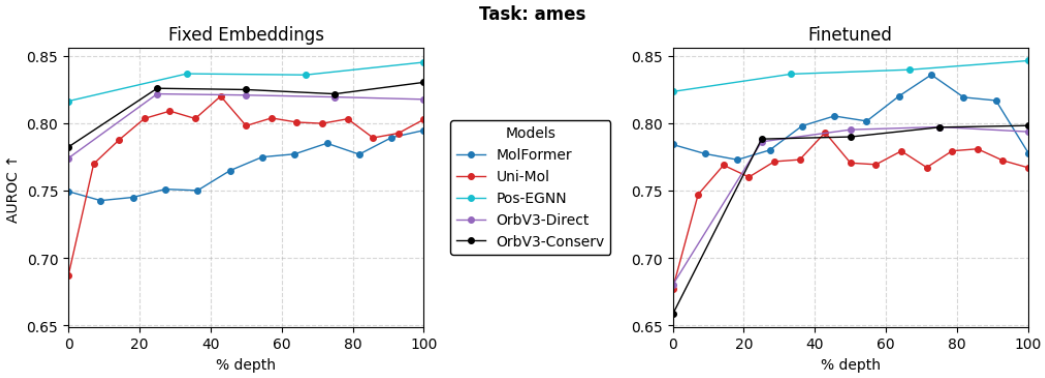


Figure 16: AUROC (↑, the higher the better) on the ***ames*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

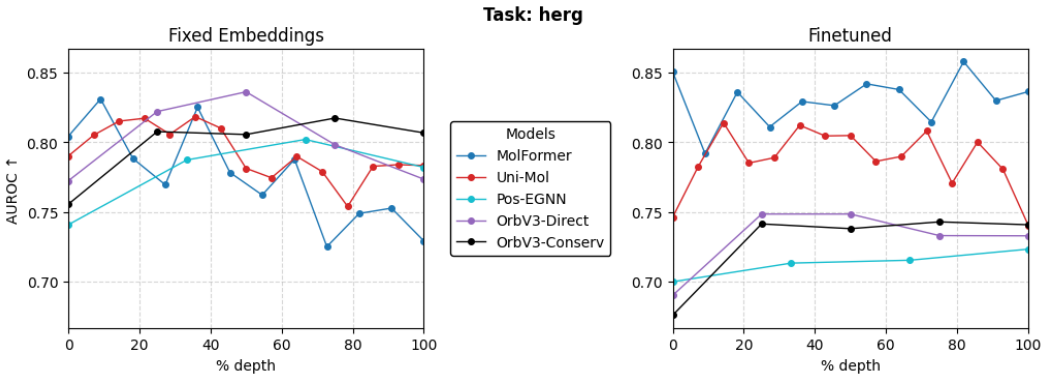


Figure 17: AUROC (↑, the higher the better) on the ***herg*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

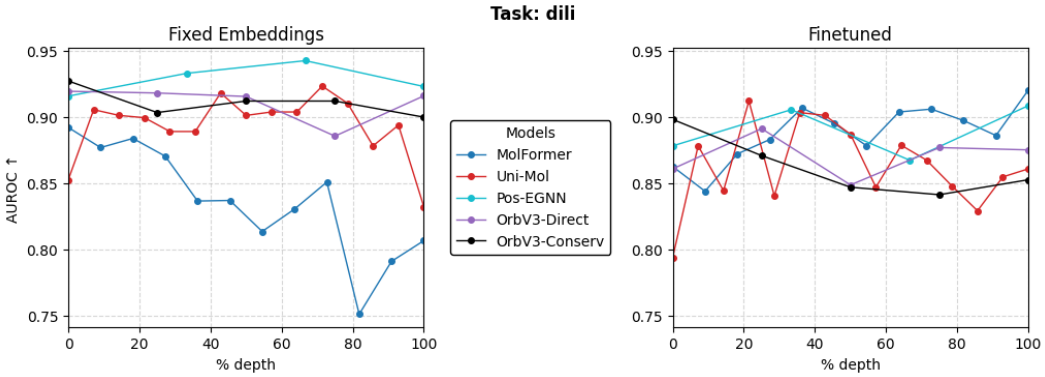


Figure 18: AUROC (↑, the higher the better) on the ***dili*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

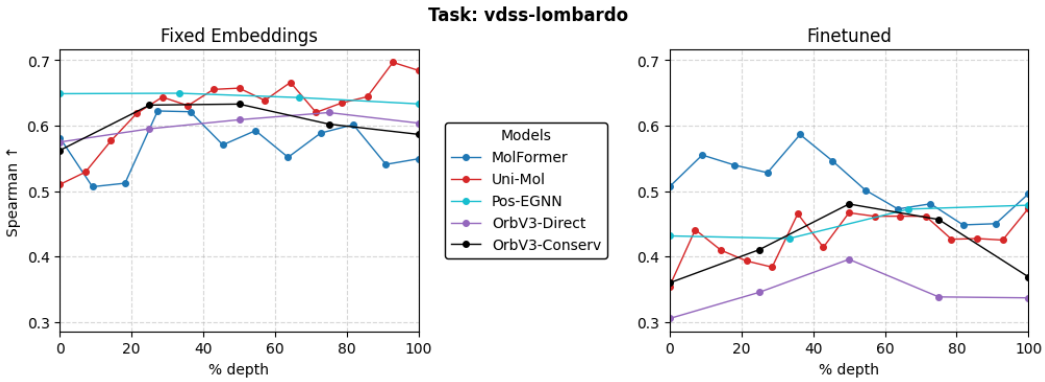


Figure 19: SPEARMAN (↑, the higher the better) on the ***vdss-lombardo*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

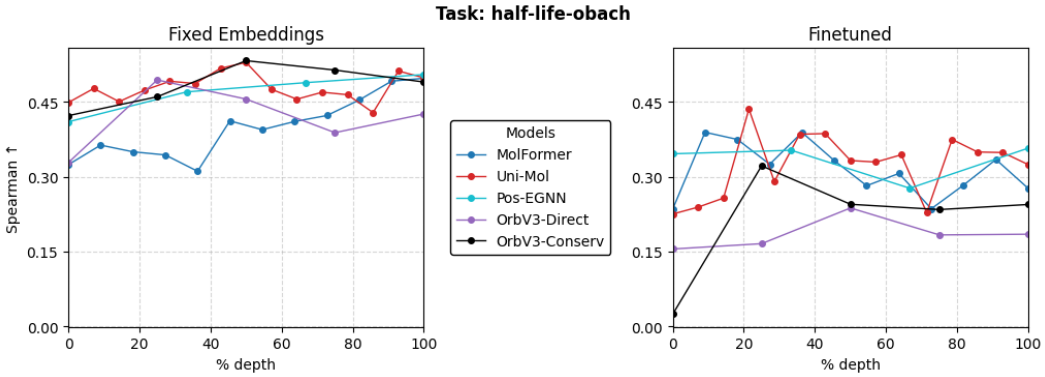


Figure 20: SPEARMAN (↑, the higher the better) on the ***half-life-obach*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

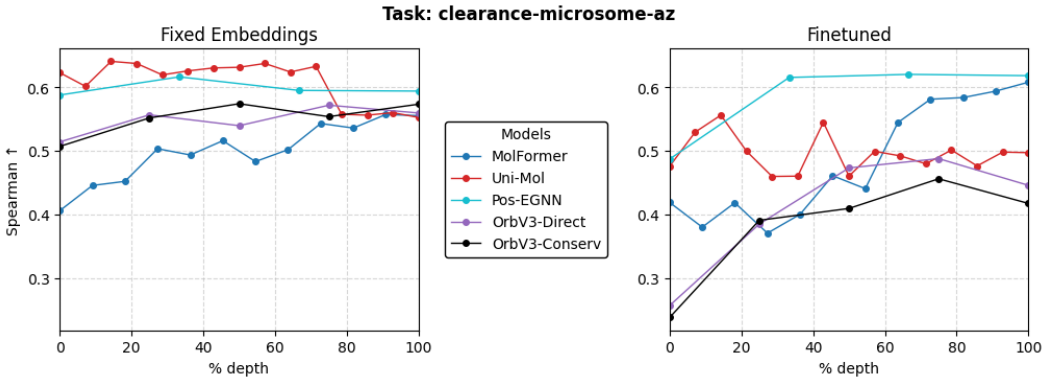


Figure 21: SPEARMAN (↑, the higher the better) on the ***clearance-microsome-az*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

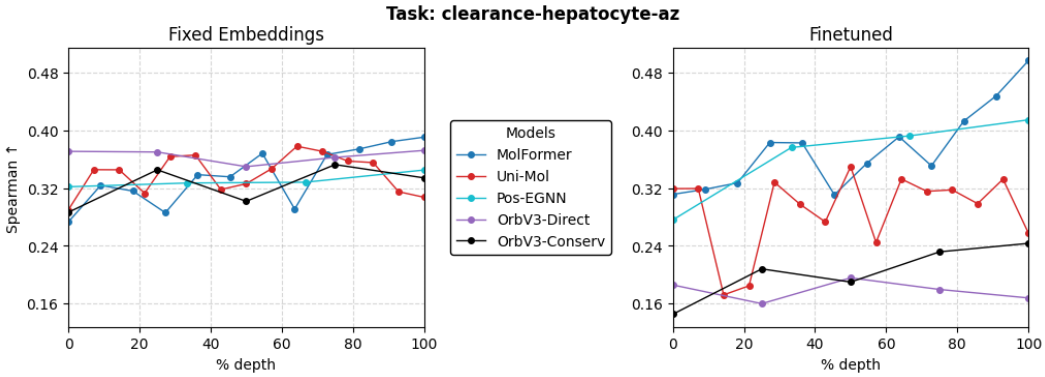


Figure 22: SPEARMAN (↑, the higher the better) on the ***clearance-hepatocyte-az*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

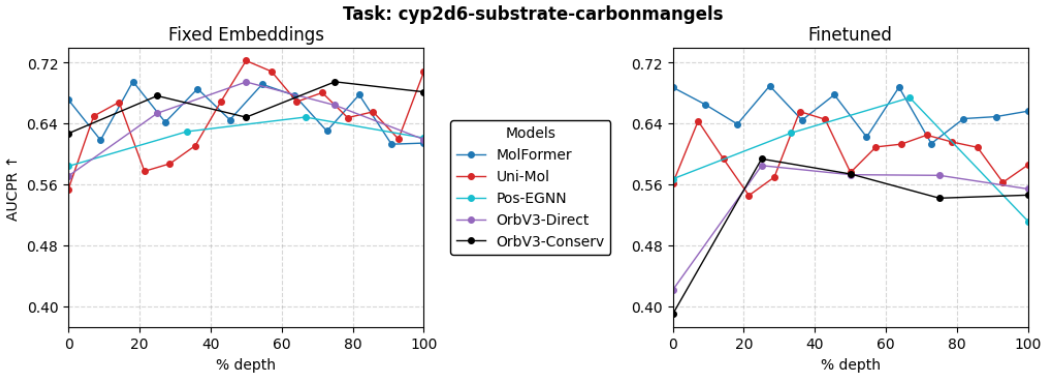


Figure 23: SPEARMAN (↑, the higher the better) on the ***cyp2d6-substrate-carbonmangels*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

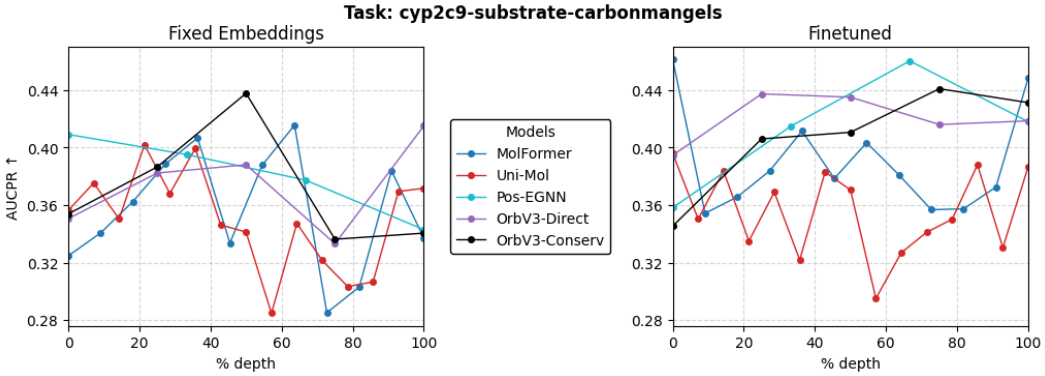


Figure 24: AUCPR (↑, the higher the better) on the ***cyp2c9-substrate-carbonmangels*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

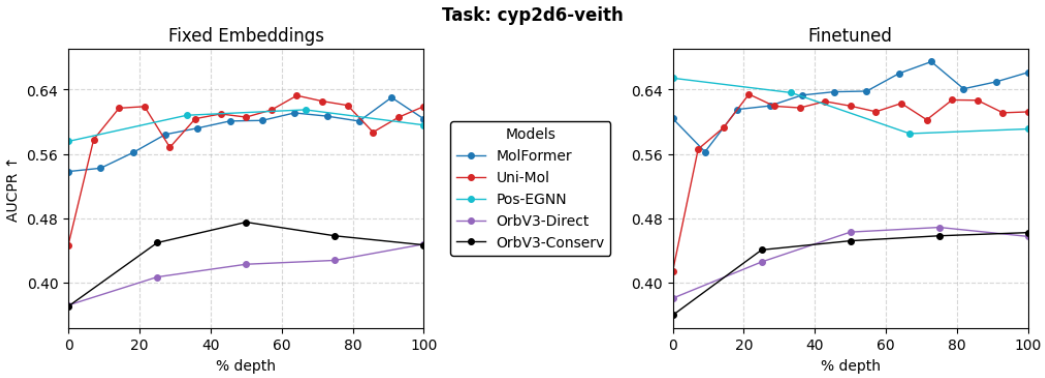


Figure 25: AUCPR (↑, the higher the better) on the ***cyp2d6-veith*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

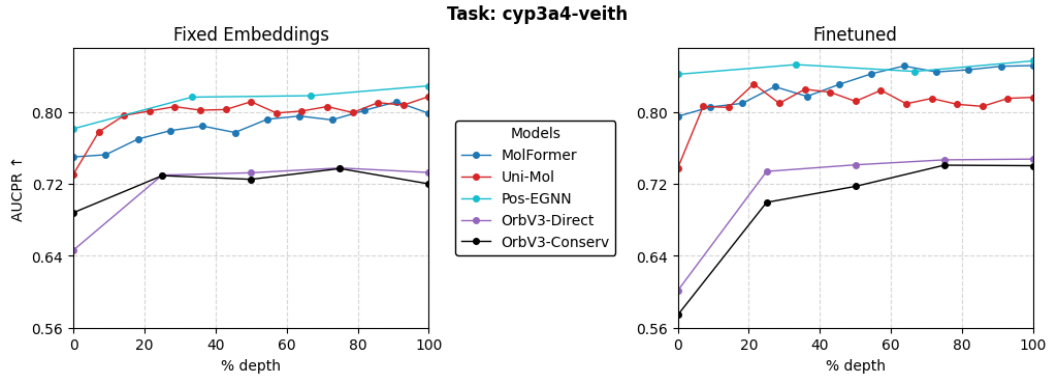


Figure 26: AUCPR (↑, the higher the better) on the ***cyp3a4-veith*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

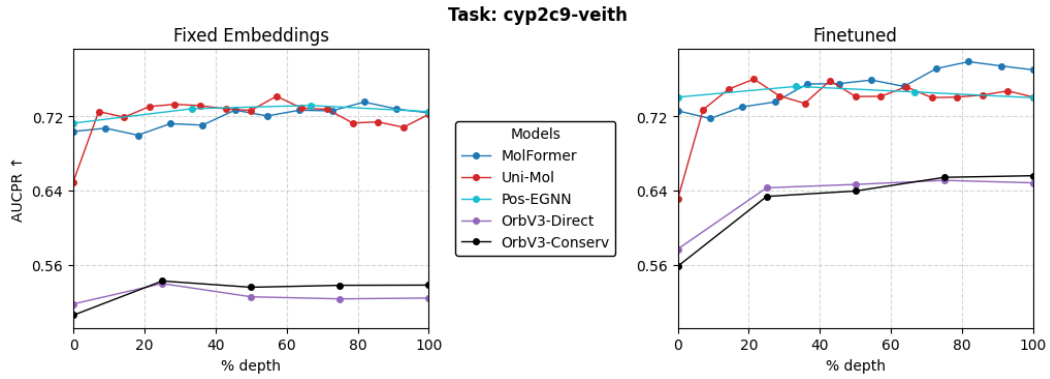


Figure 27: AUCPR (↑, the higher the better) on the ***cyp2c9-veith*** dataset as a function of encoder depth (% depth) for frozen embeddings (left) versus finetuned embeddings (right).

D MolFormer: Frozen Embedding vs. Finetuned Scatter Plots

This appendix section presents a series of scatter plots dedicated to the MolFormer model. Each plot illustrates the relationship between frozen embedding performance and full finetuning performance across the model's different layers for a specific downstream task. The plots were ordered by Pearson correlation.

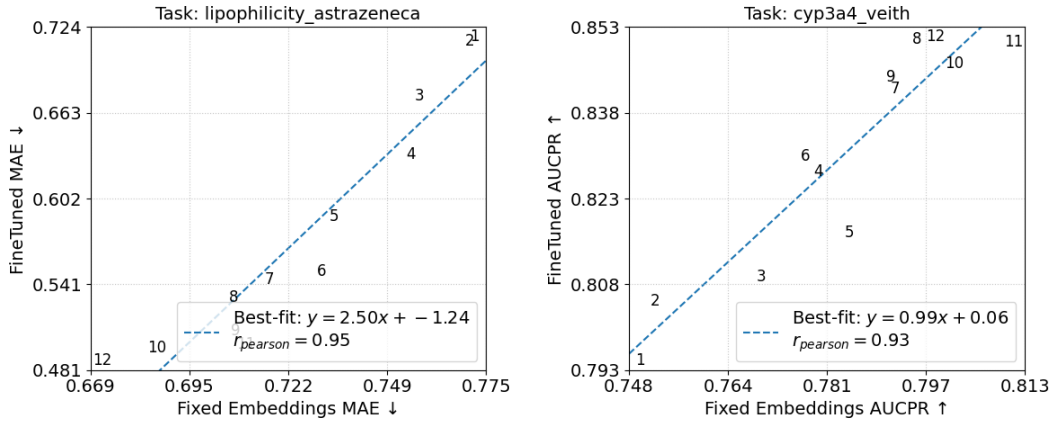


Figure 28: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

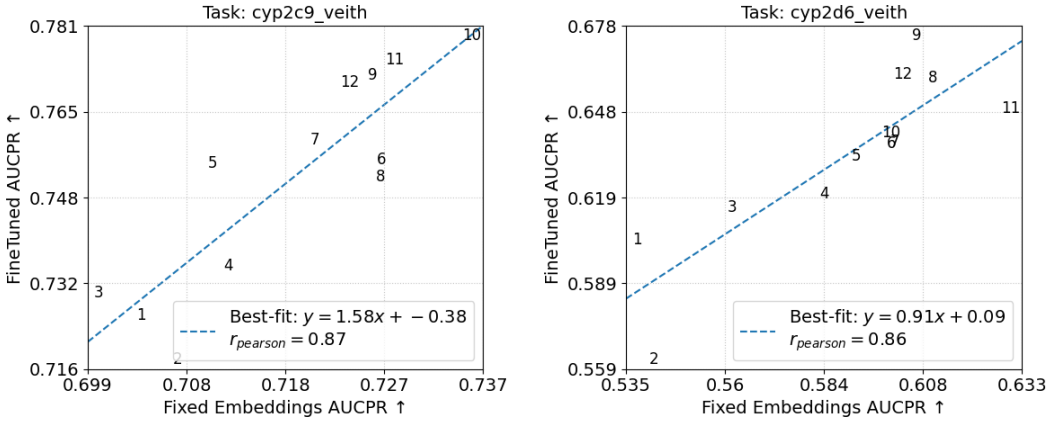


Figure 29: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

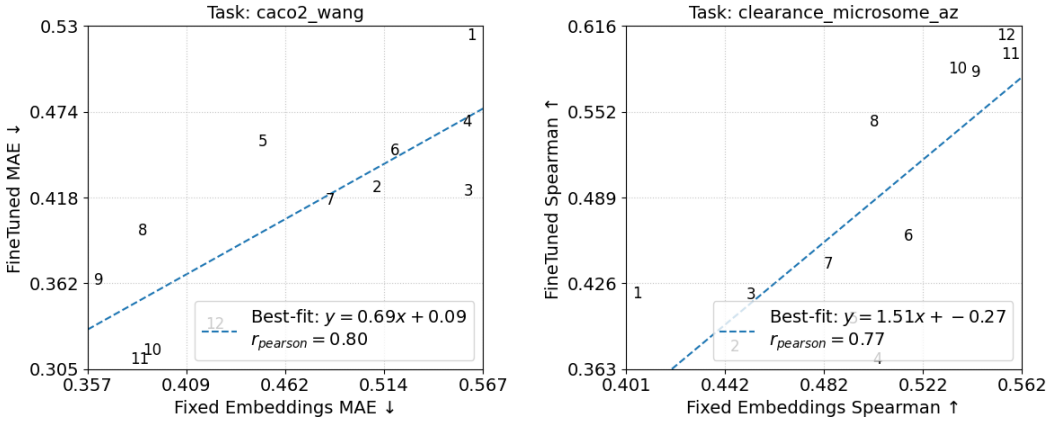


Figure 30: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

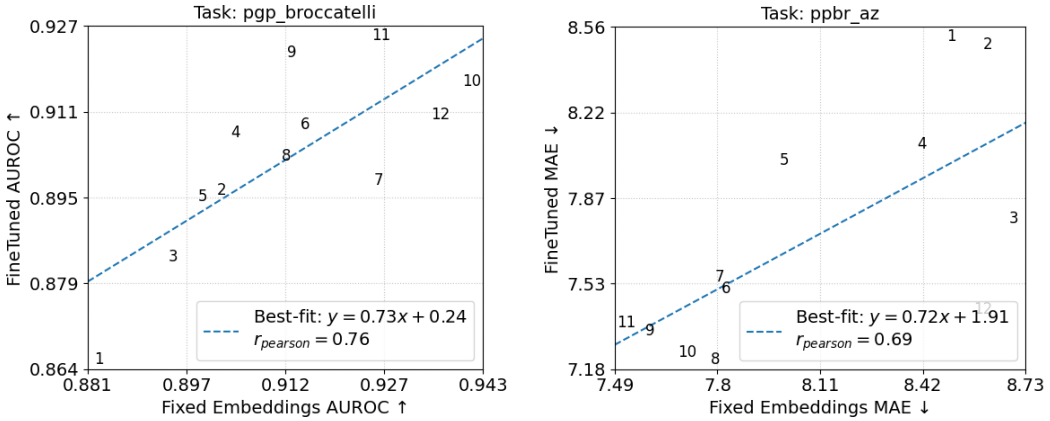


Figure 31: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

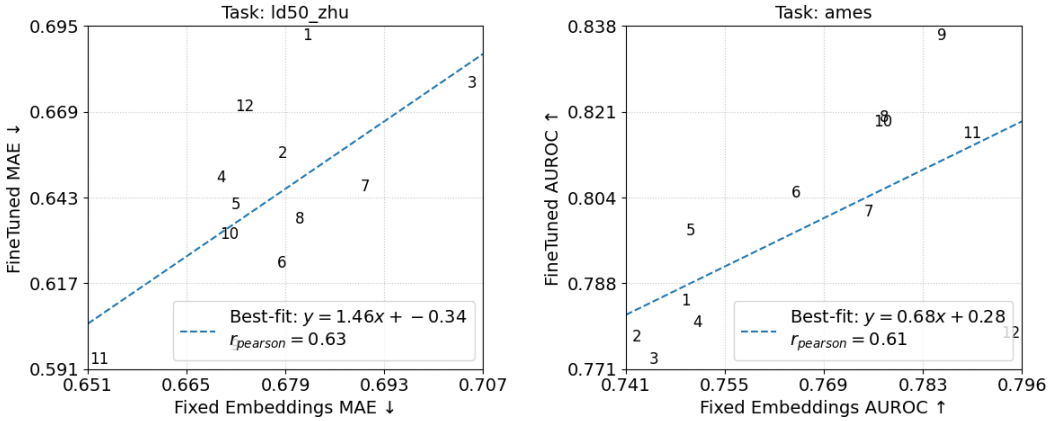


Figure 32: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

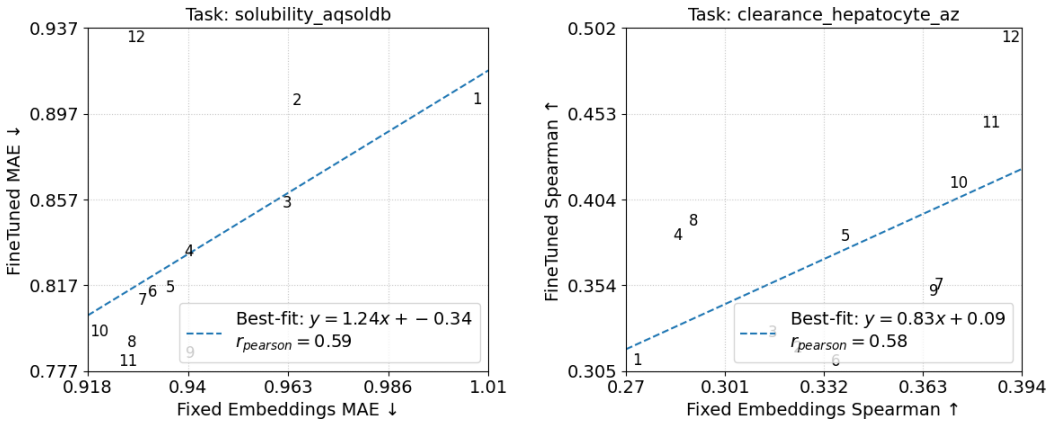


Figure 33: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

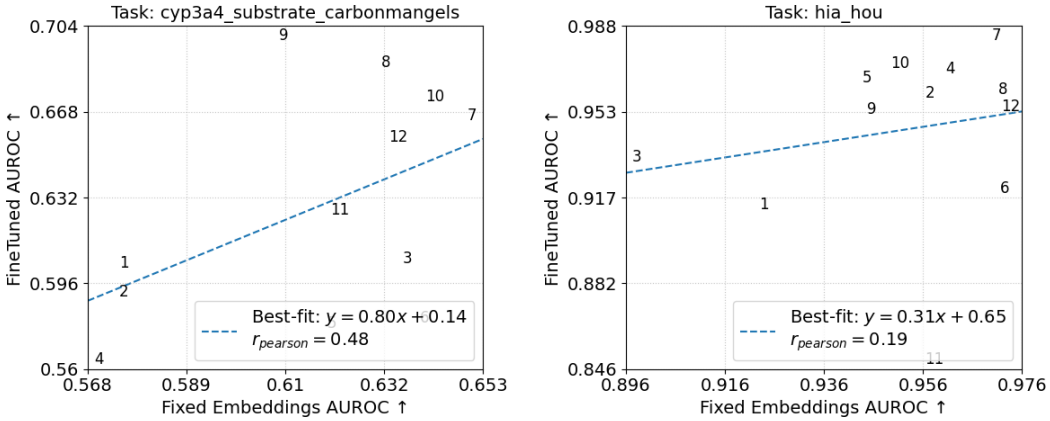


Figure 34: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

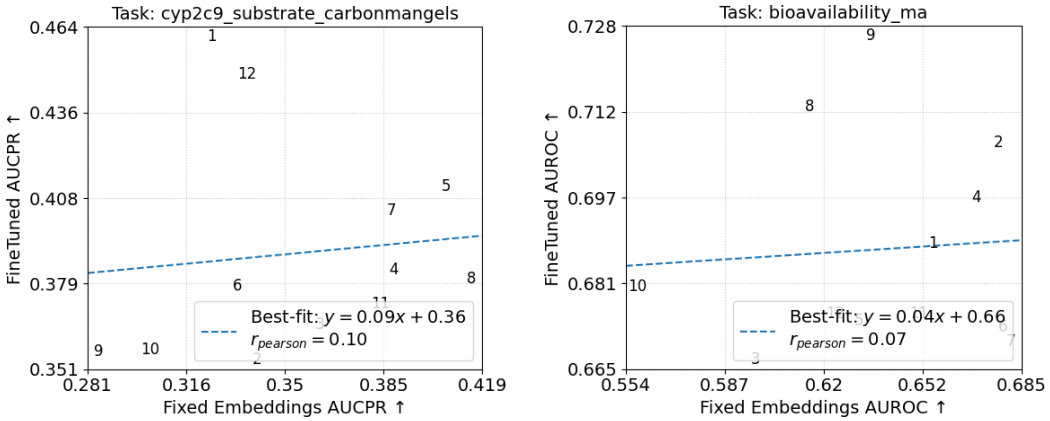


Figure 35: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

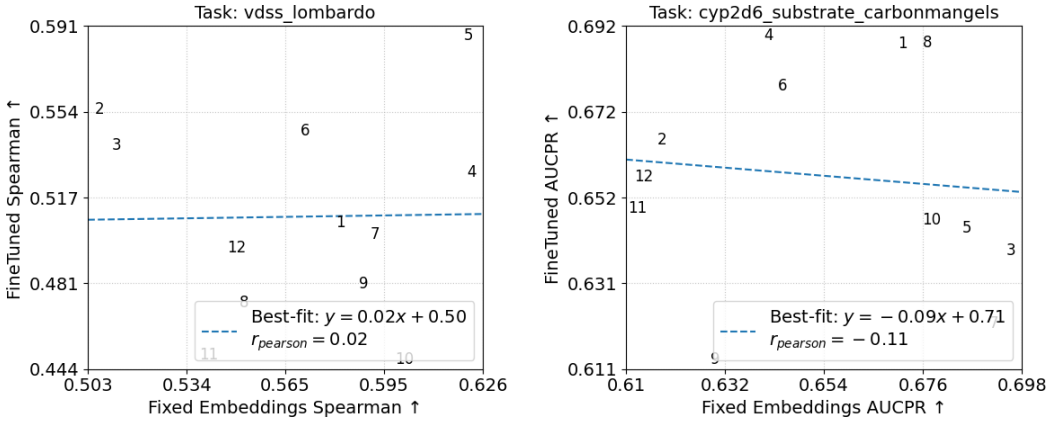


Figure 36: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

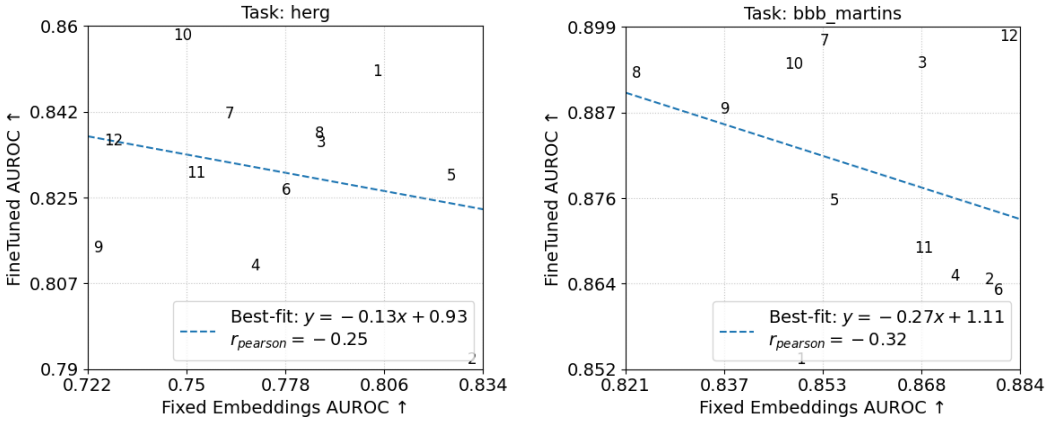


Figure 37: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

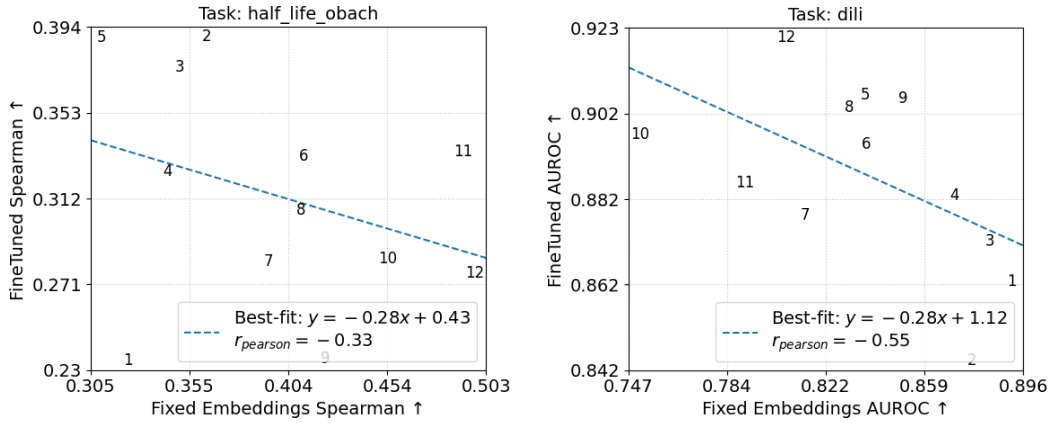


Figure 38: Frozen embeddings vs. finetuned performance for every layer of MolFormer.

E Uni-Mol: Frozen embedding vs. Finetuned Scatter Plots

This appendix section presents a series of scatter plots dedicated to the Uni-Mol model. Each plot illustrates the relationship between frozen embedding performance and full finetuning performance across the model's different layers for a specific downstream task. The plots were ordered by Pearson correlation.

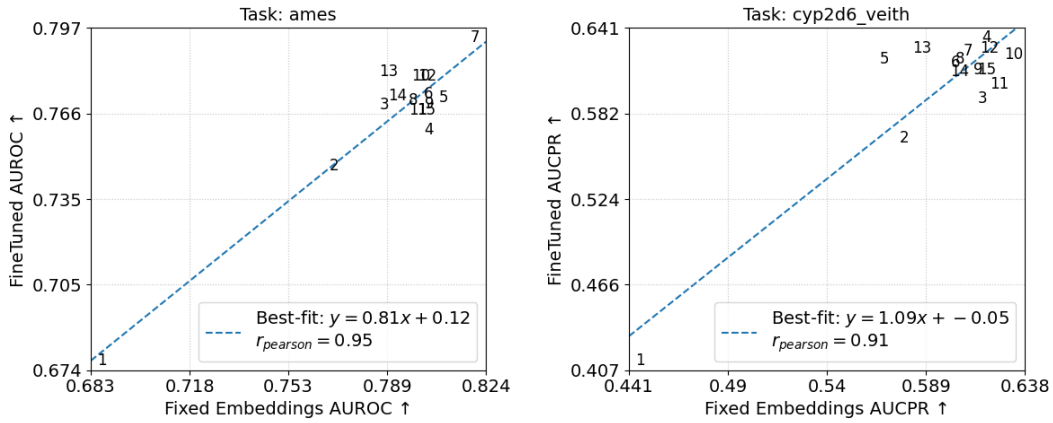


Figure 39: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

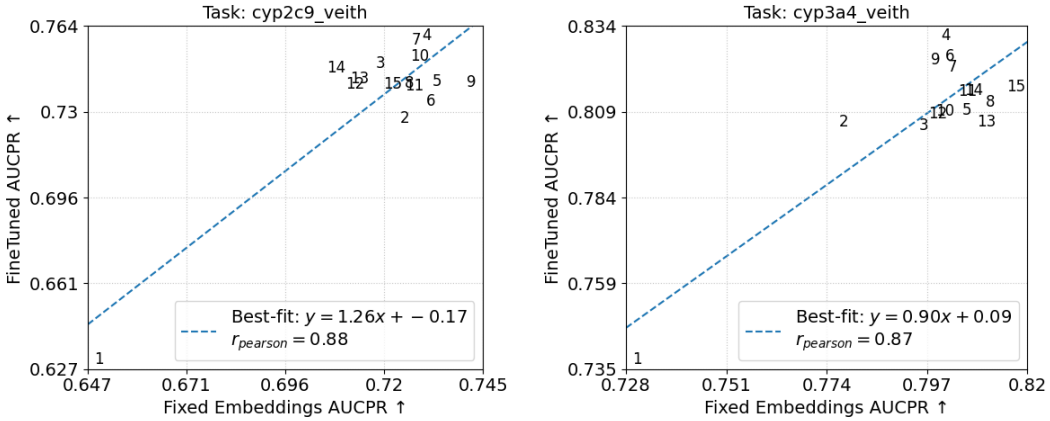


Figure 40: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

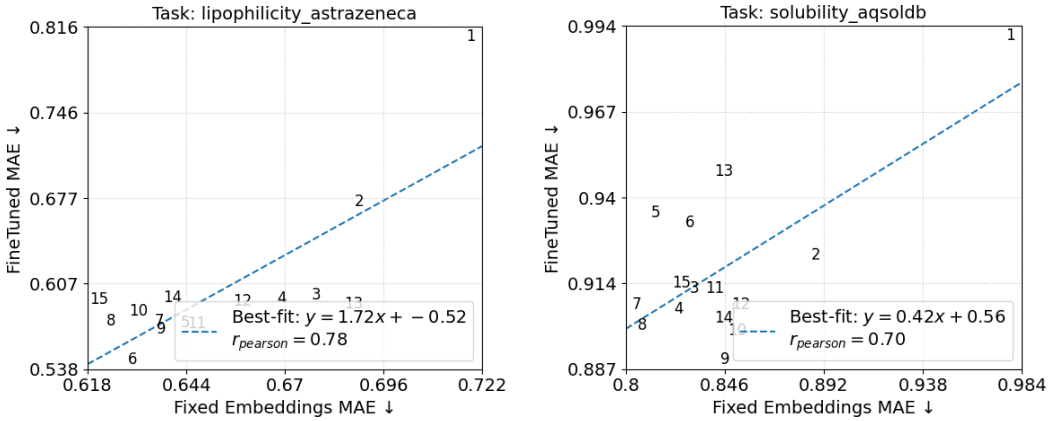


Figure 41: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

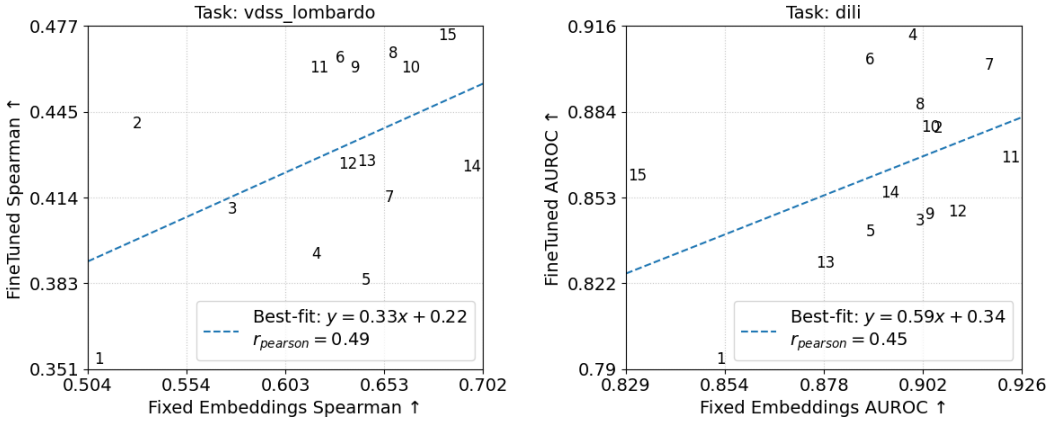


Figure 42: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

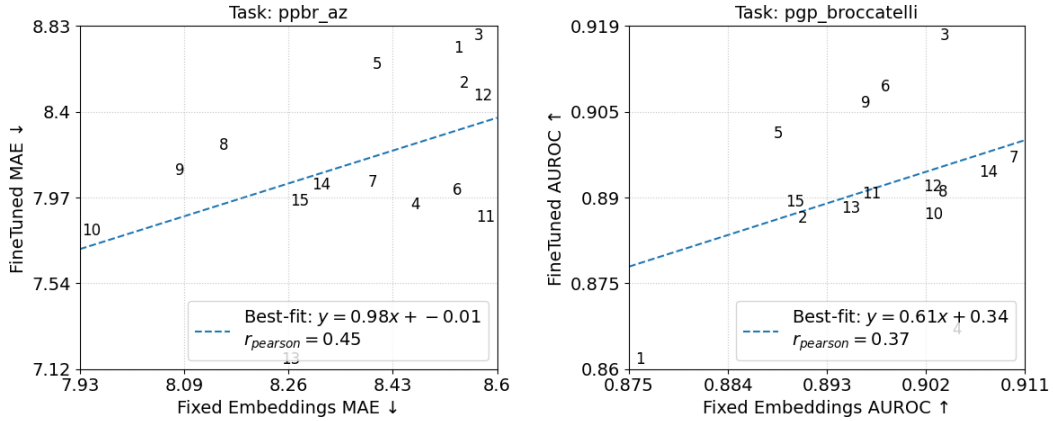


Figure 43: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

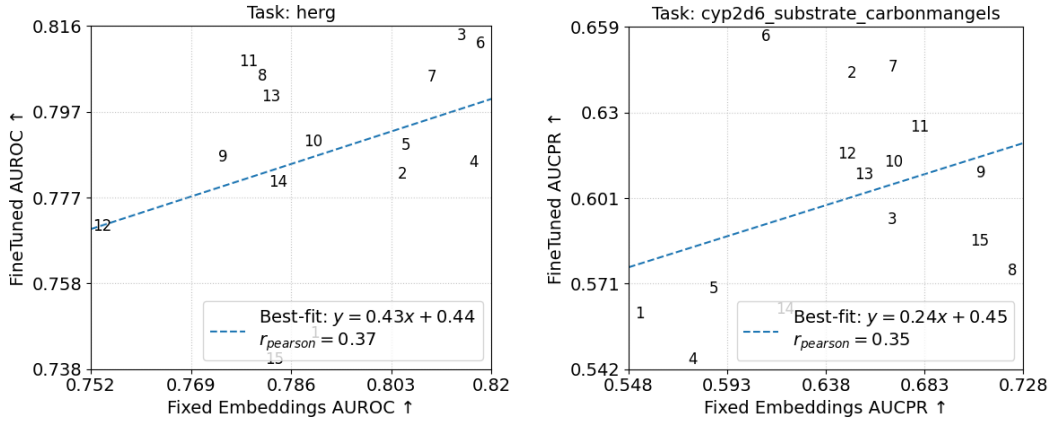


Figure 44: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

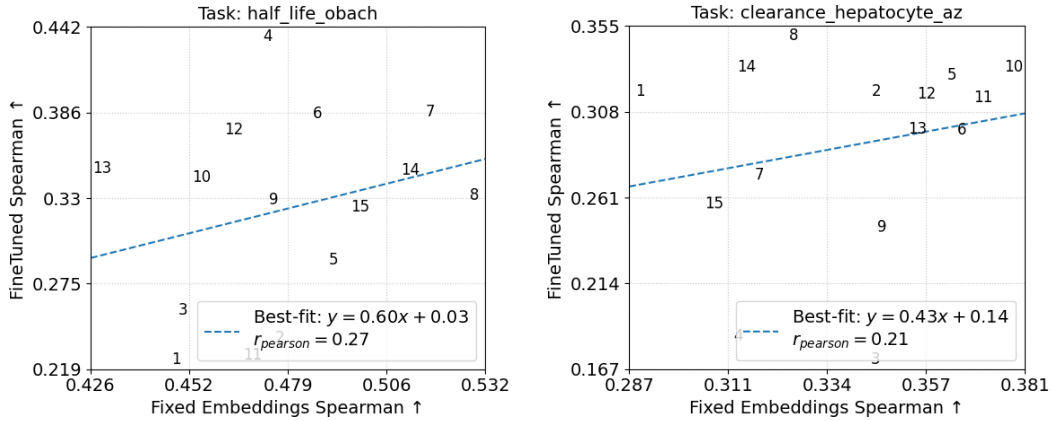


Figure 45: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

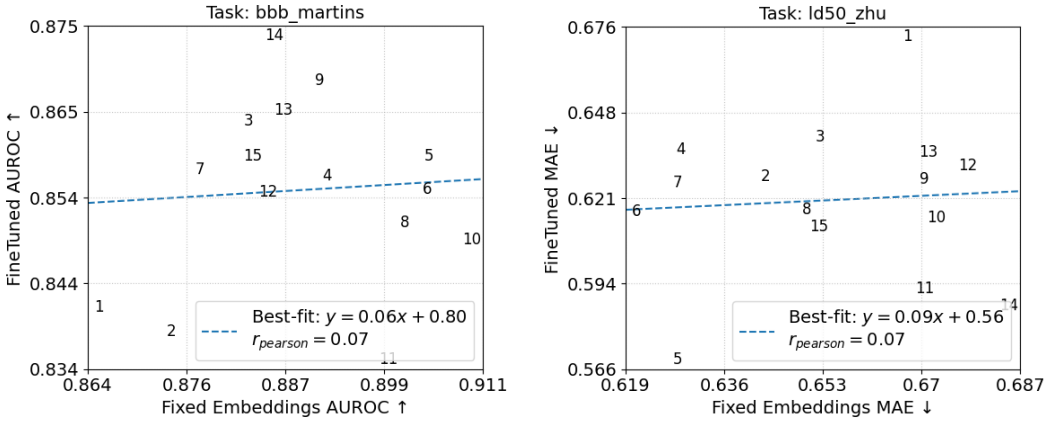


Figure 46: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

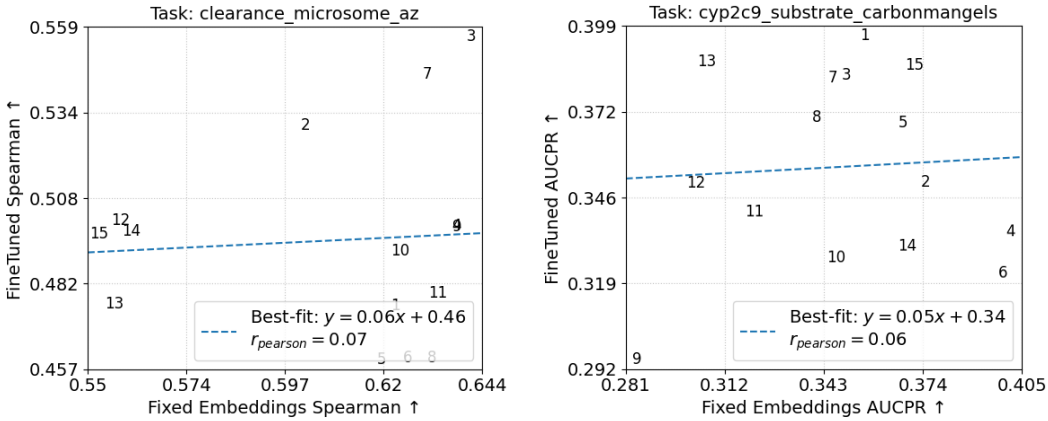


Figure 47: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

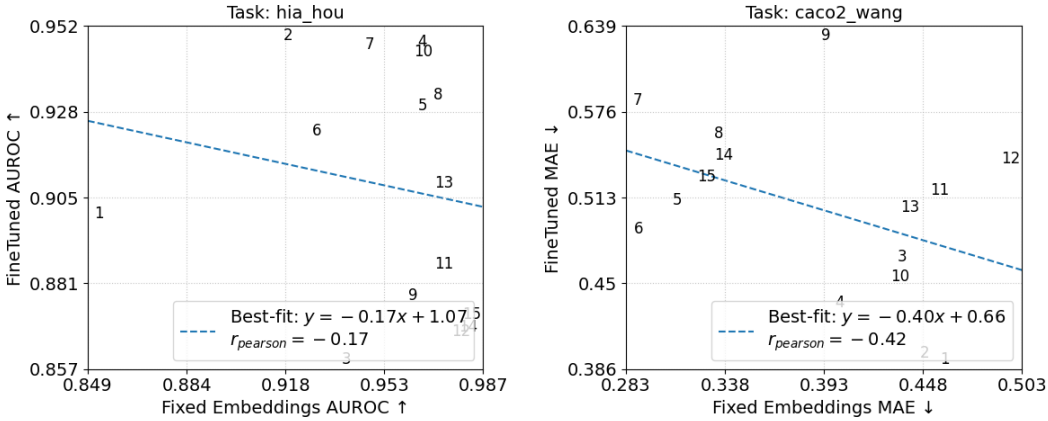


Figure 48: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

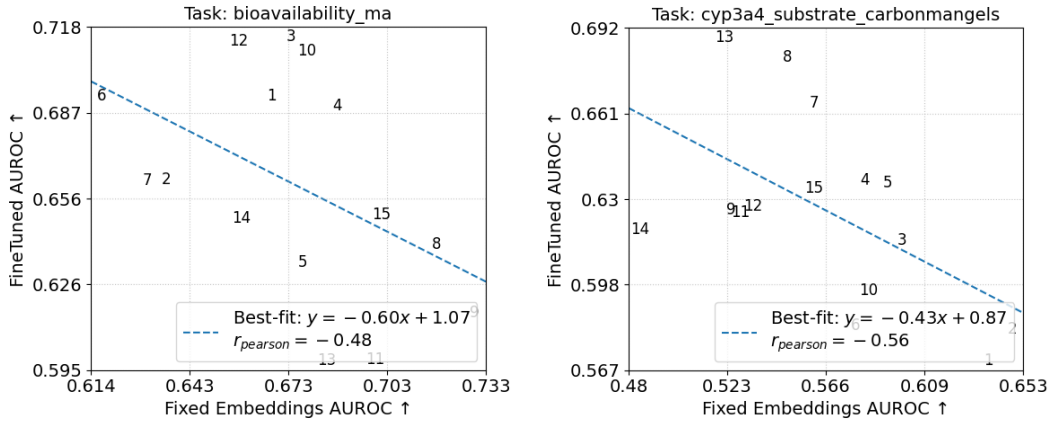


Figure 49: Frozen embeddings vs. finetuned performance for every layer of Uni-Mol.

F OrbV3-Direct: Frozen embedding vs. Finetuned Scatter Plots

This appendix section presents a series of scatter plots dedicated to the OrbV3-Direct model. Each plot illustrates the relationship between frozen embedding performance and full finetuning performance across the model's different layers for a specific downstream task. The plots were ordered by Pearson correlation.

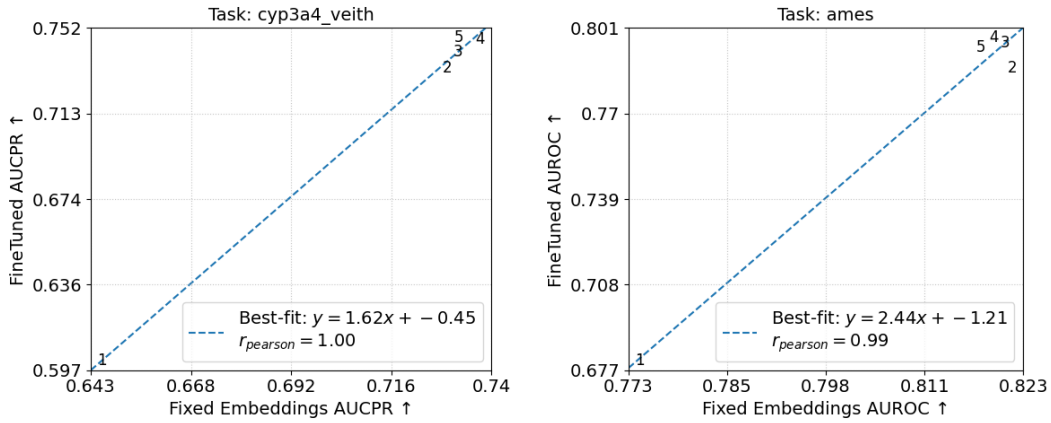


Figure 50: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

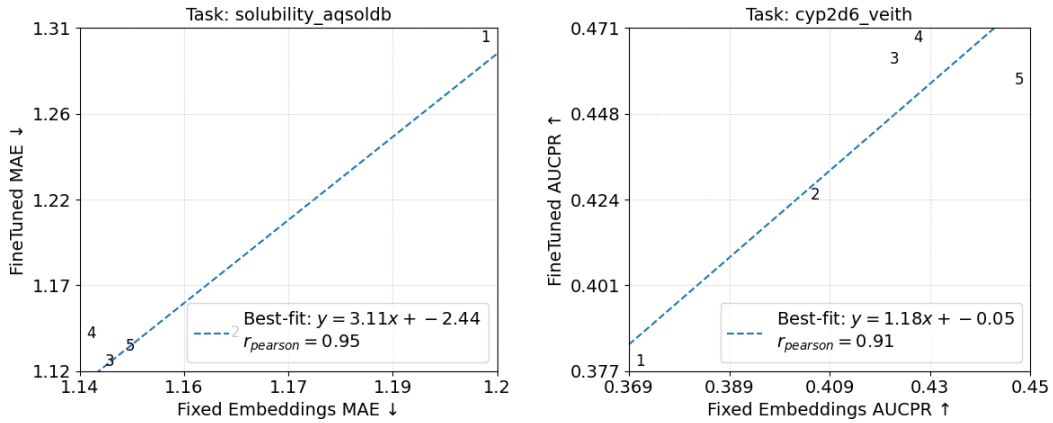


Figure 51: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

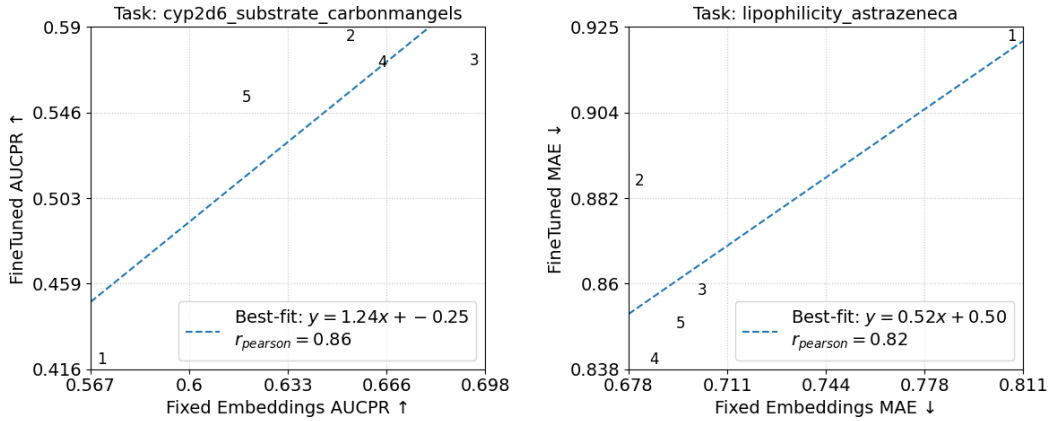


Figure 52: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

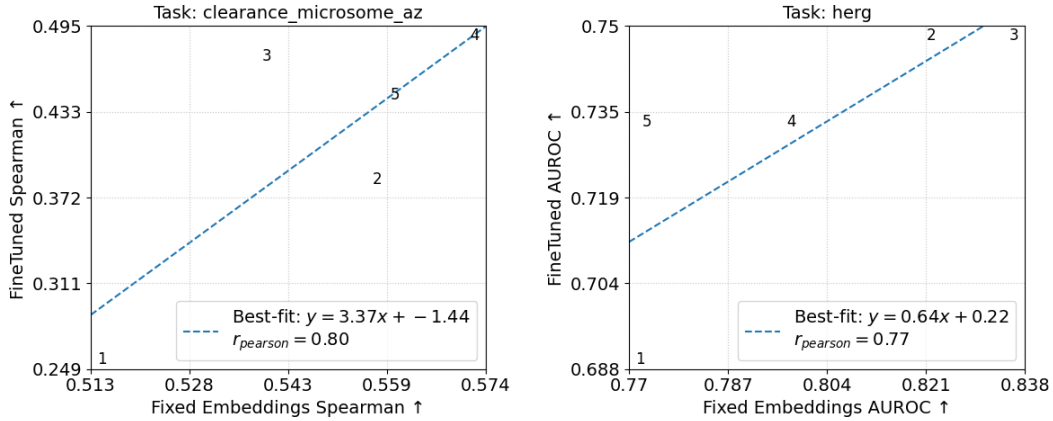


Figure 53: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

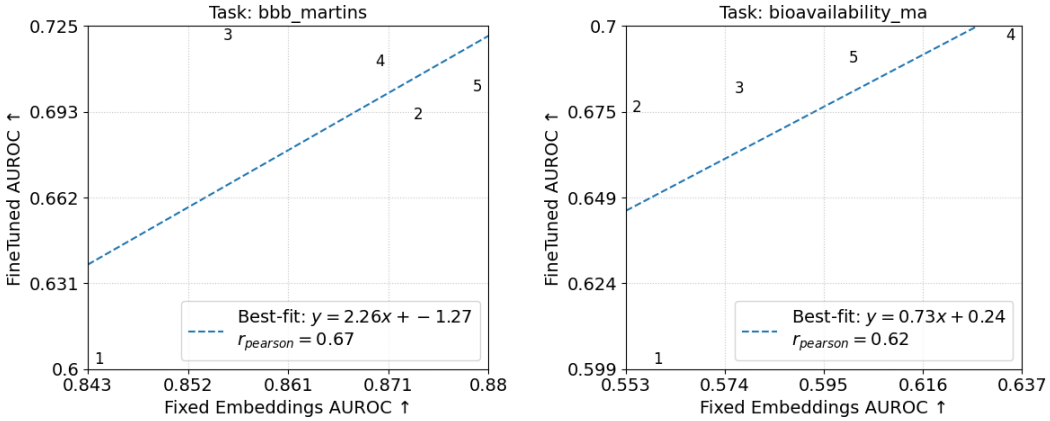


Figure 54: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

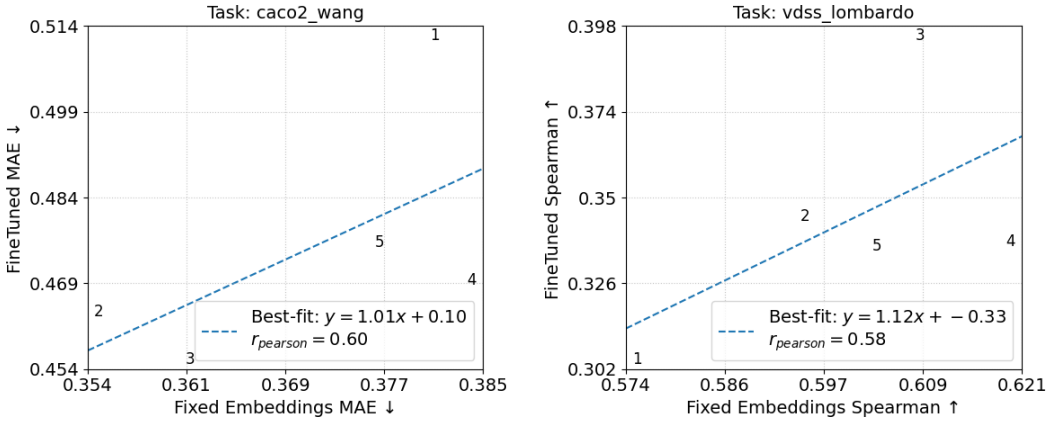


Figure 55: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

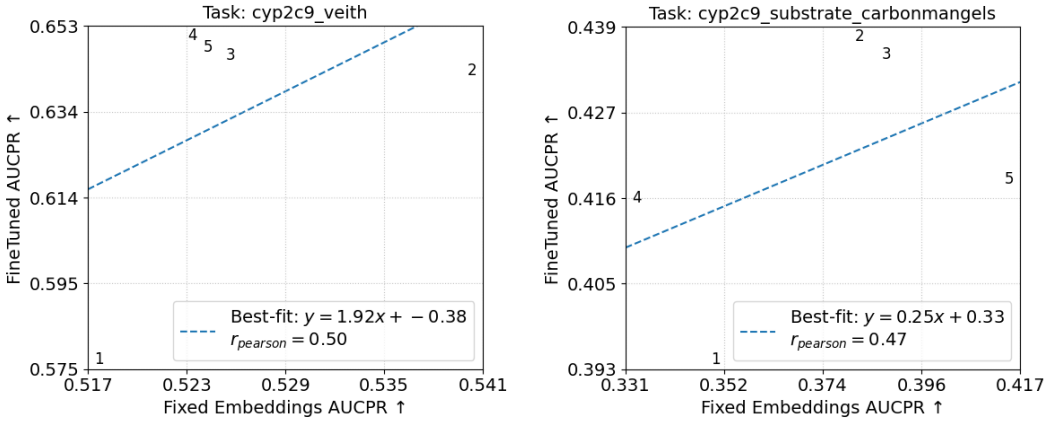


Figure 56: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

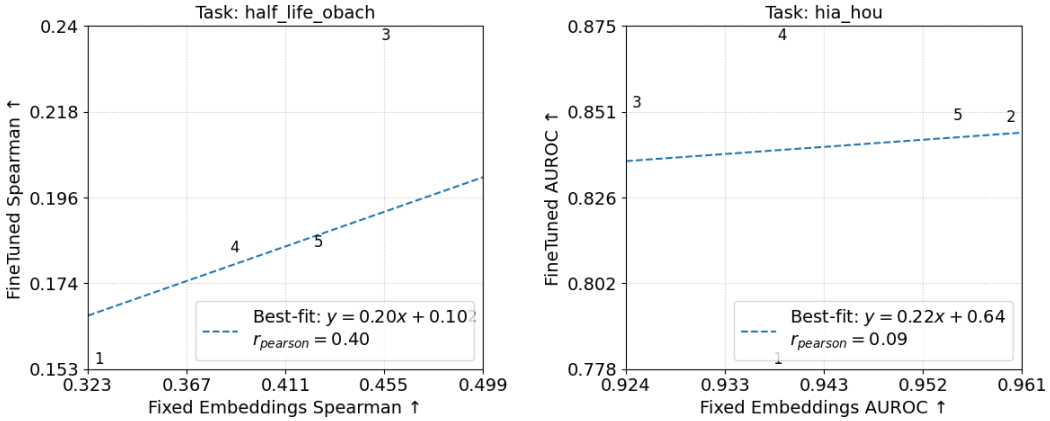


Figure 57: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

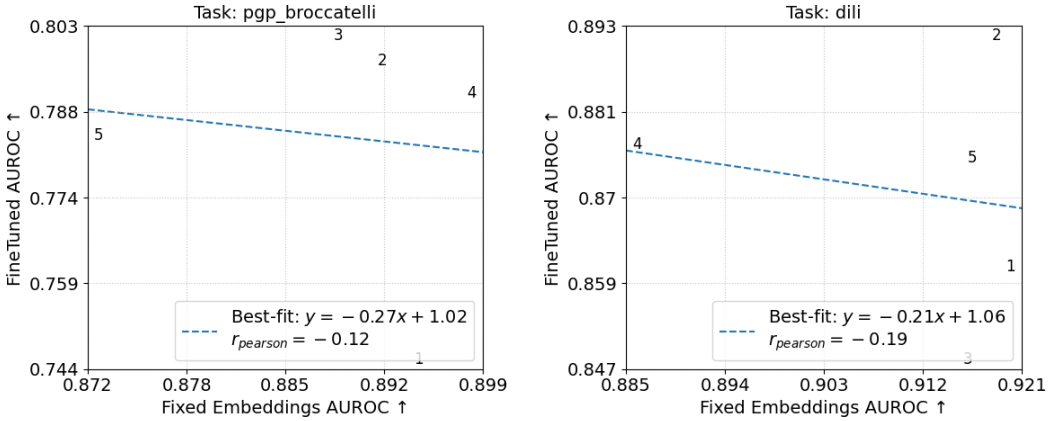


Figure 58: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

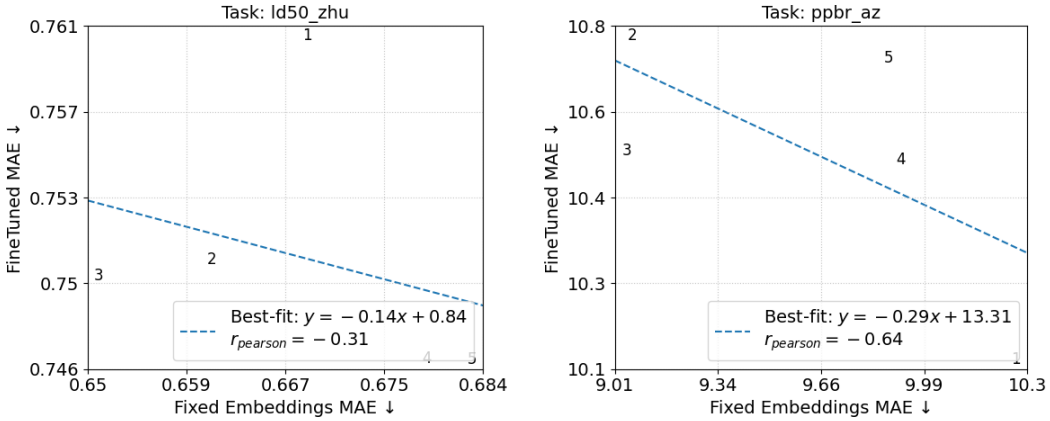


Figure 59: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

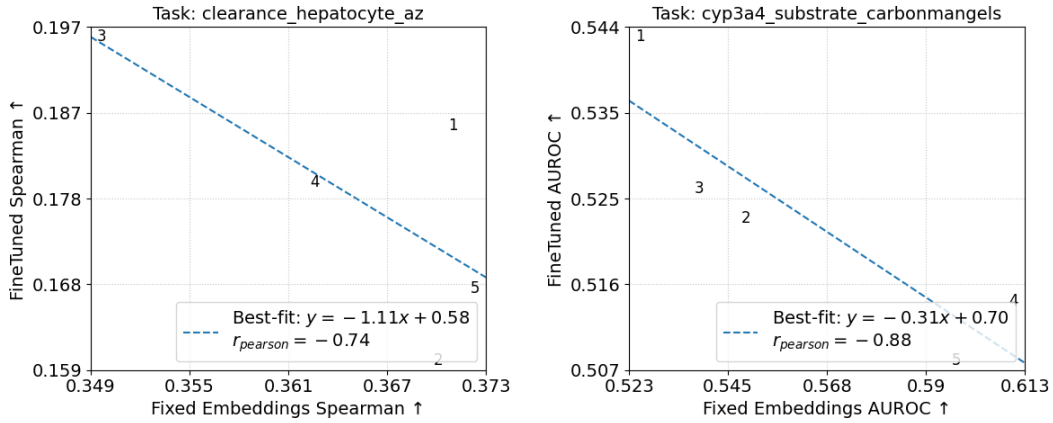


Figure 60: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Direct.

G OrbV3-Conservative: Frozen embedding vs. Finetuned Scatter Plots

This appendix section presents a series of scatter plots dedicated to the OrbV3-Conservative model. Each plot illustrates the relationship between frozen embedding performance and full finetuning performance across the model's different layers for a specific downstream task. The plots were ordered by Pearson correlation.

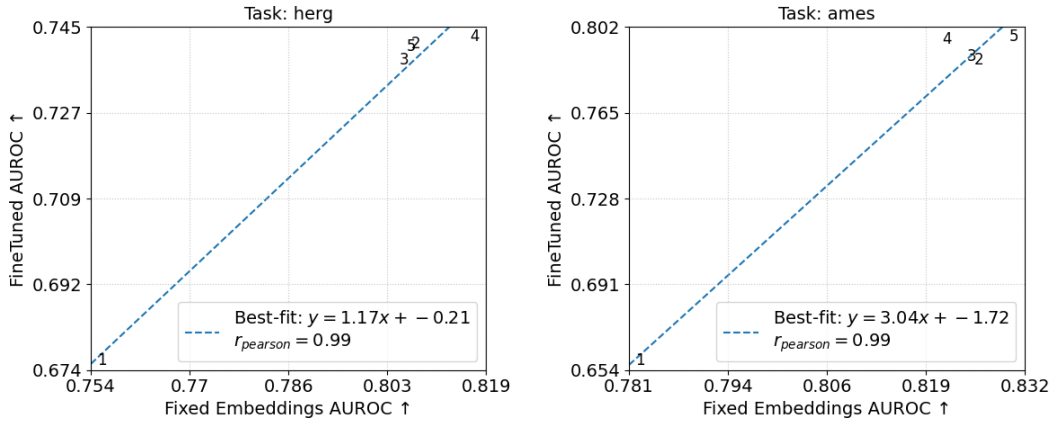


Figure 61: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

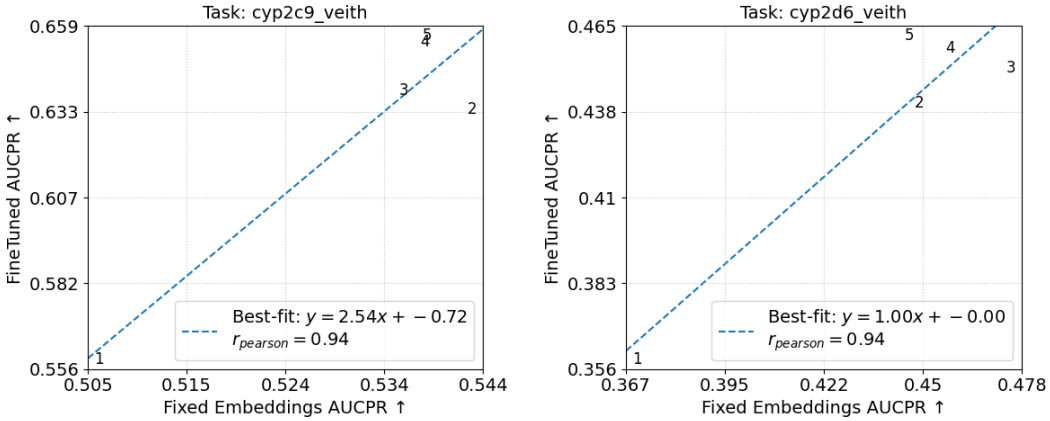


Figure 62: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

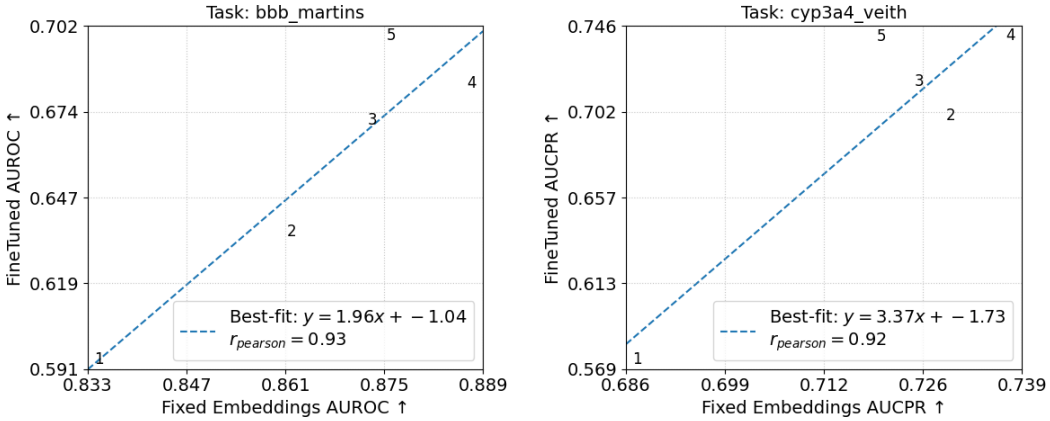


Figure 63: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

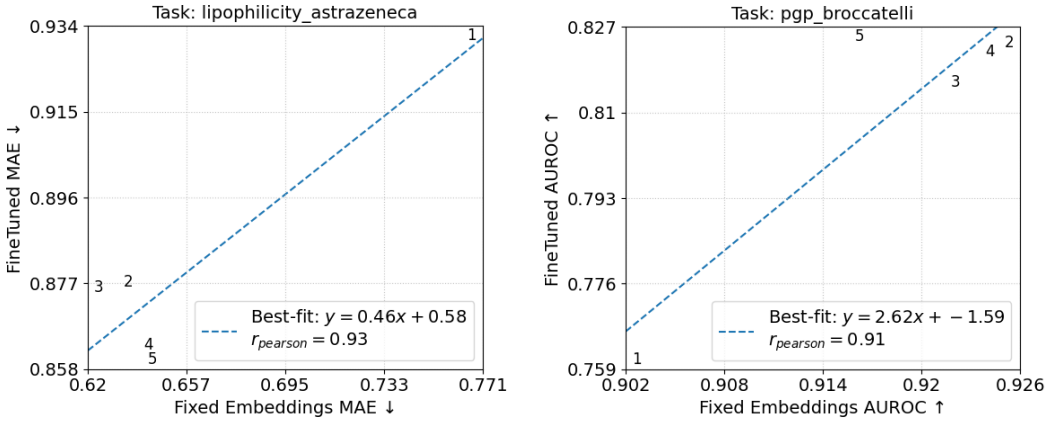


Figure 64: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

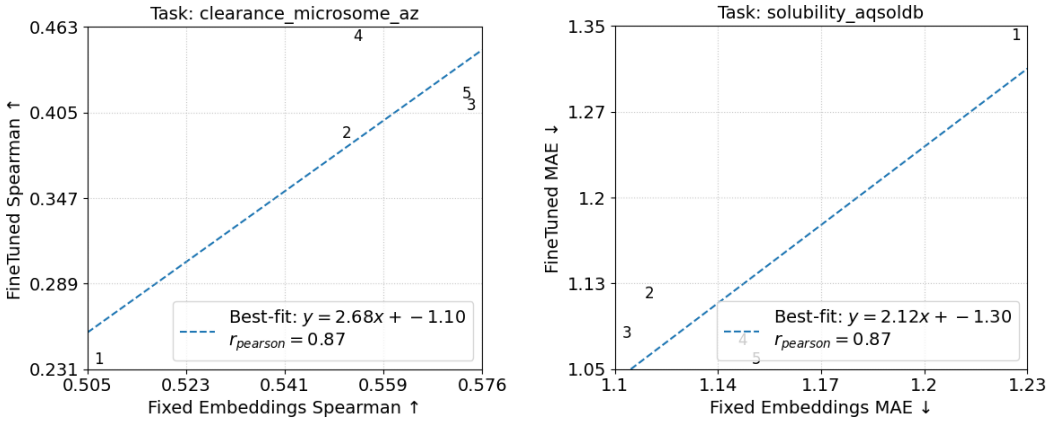


Figure 65: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

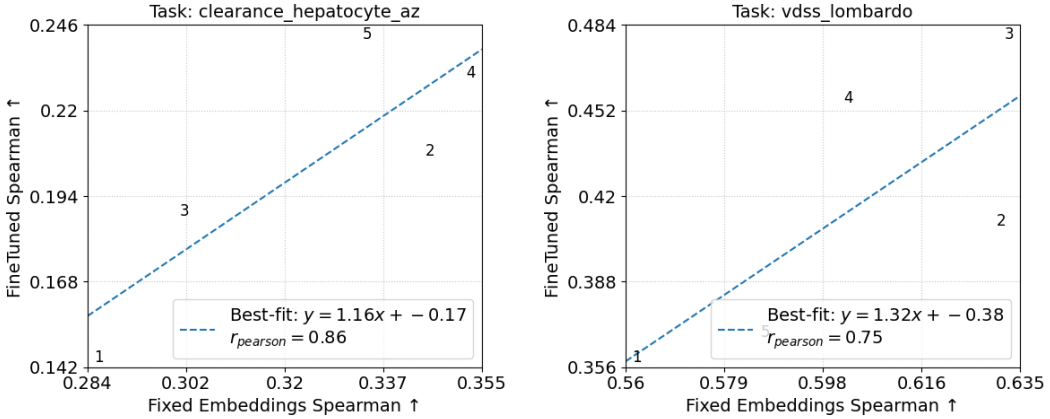


Figure 66: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

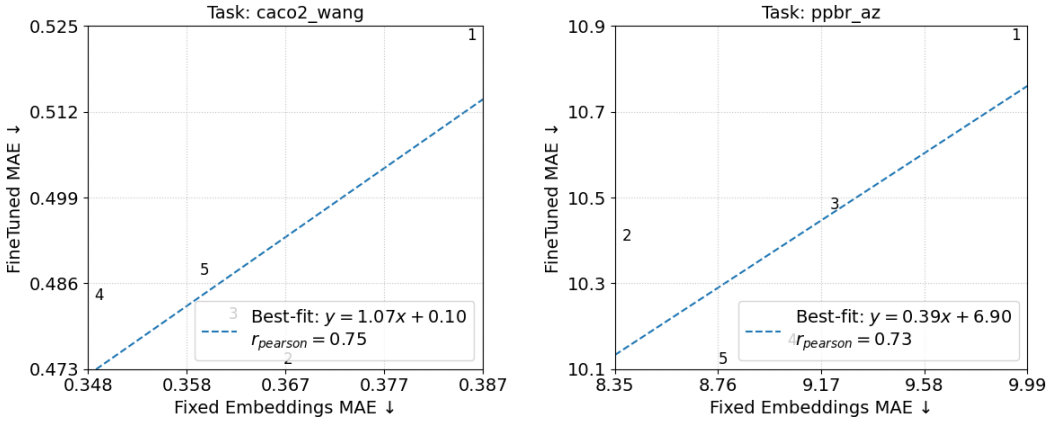


Figure 67: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

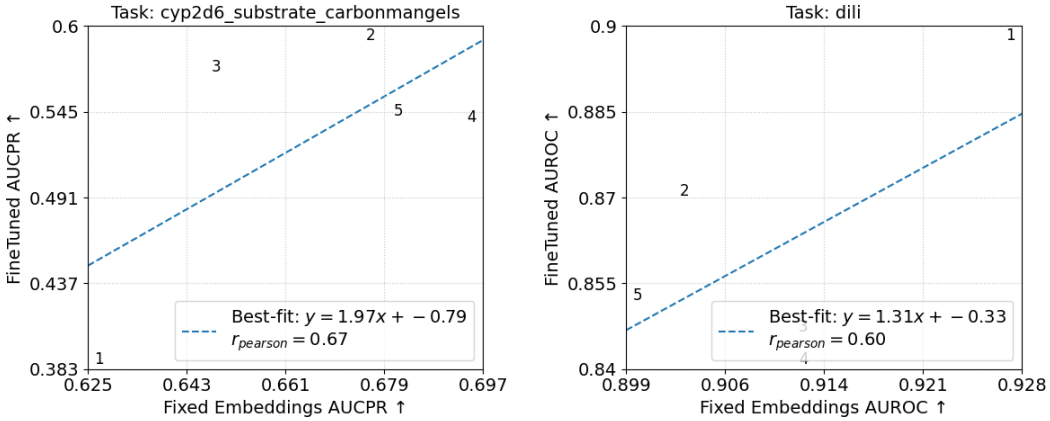


Figure 68: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

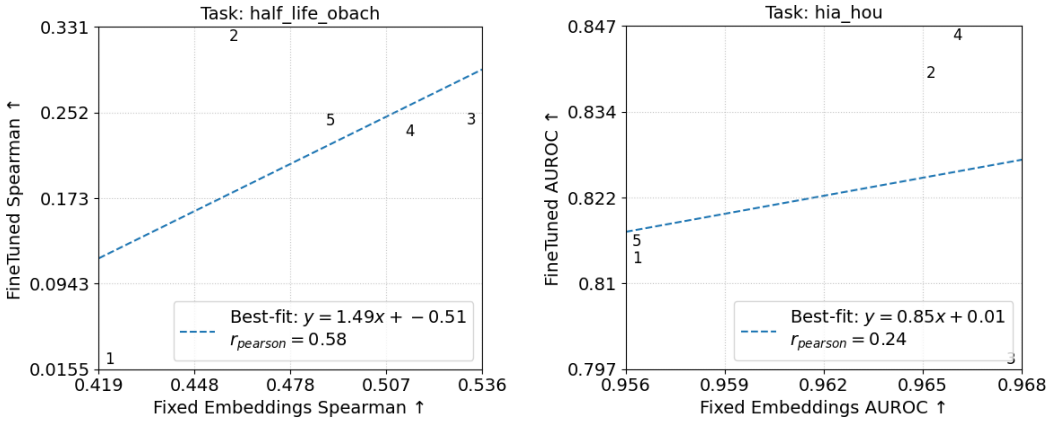


Figure 69: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

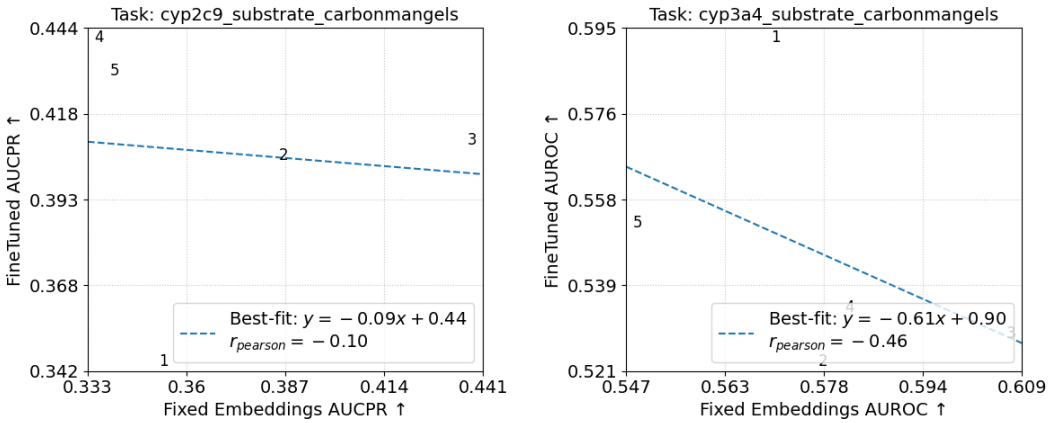


Figure 70: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

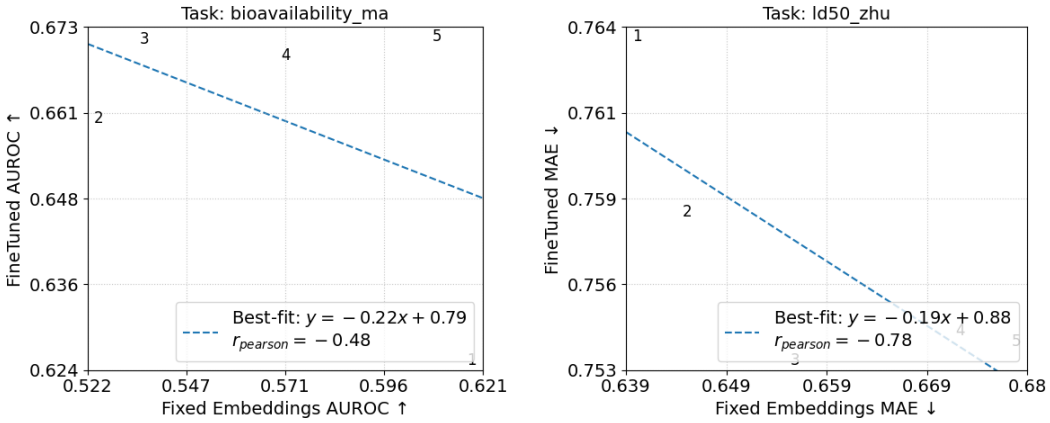


Figure 71: Frozen embeddings vs. finetuned performance for every layer of OrbV3-Conservative.

H Pos-EGNN: Frozen embedding vs. Finetuned Scatter Plots

This appendix section presents a series of scatter plots dedicated to the Pos-EGNN model. Each plot illustrates the relationship between frozen embedding performance and full finetuning performance across the model's different layers for a specific downstream task. The plots were ordered by Pearson correlation.

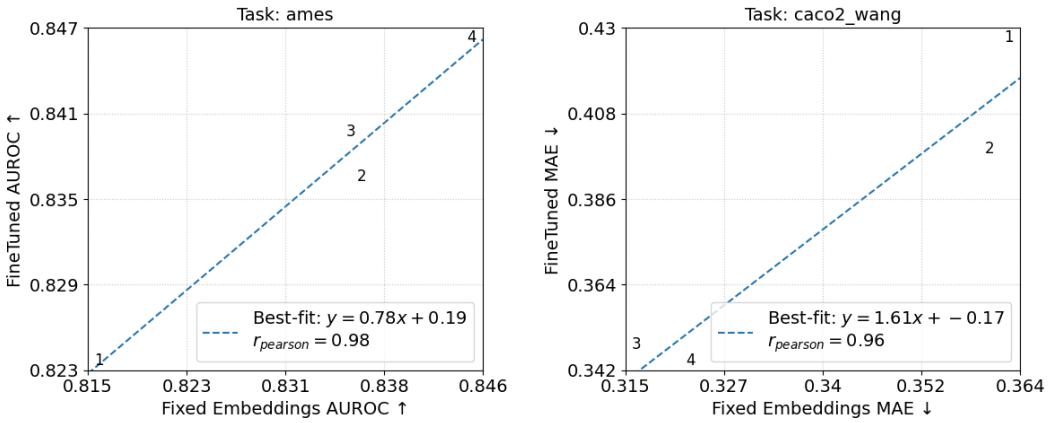


Figure 72: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

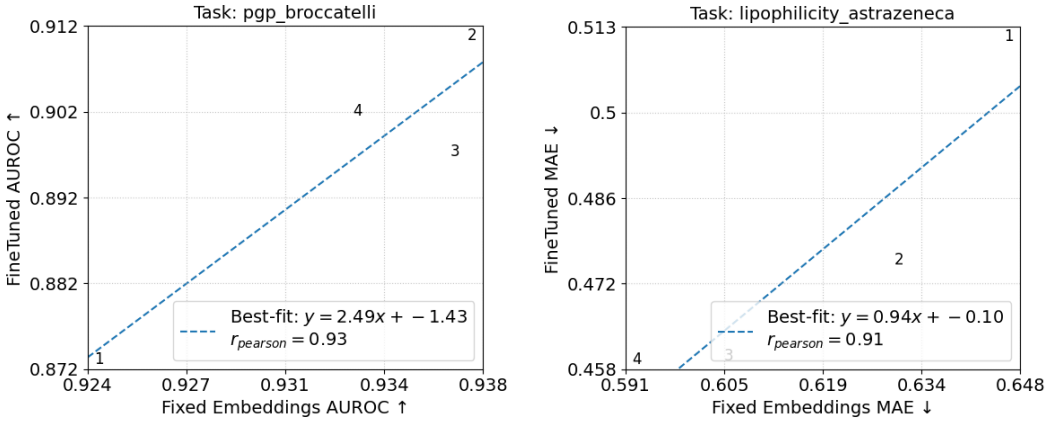


Figure 73: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

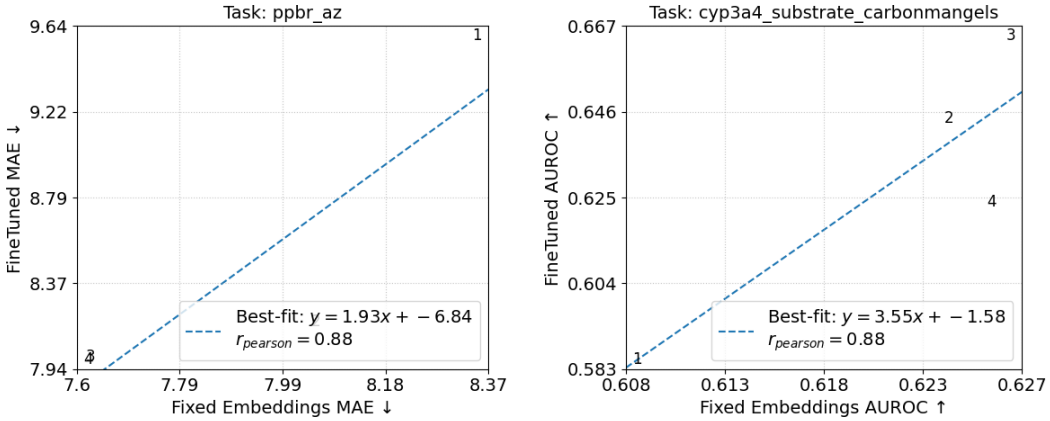


Figure 74: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

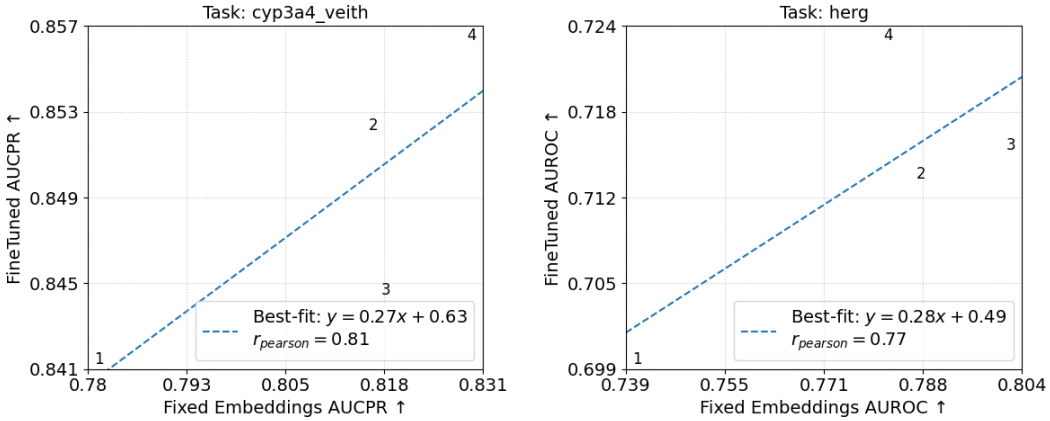


Figure 75: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

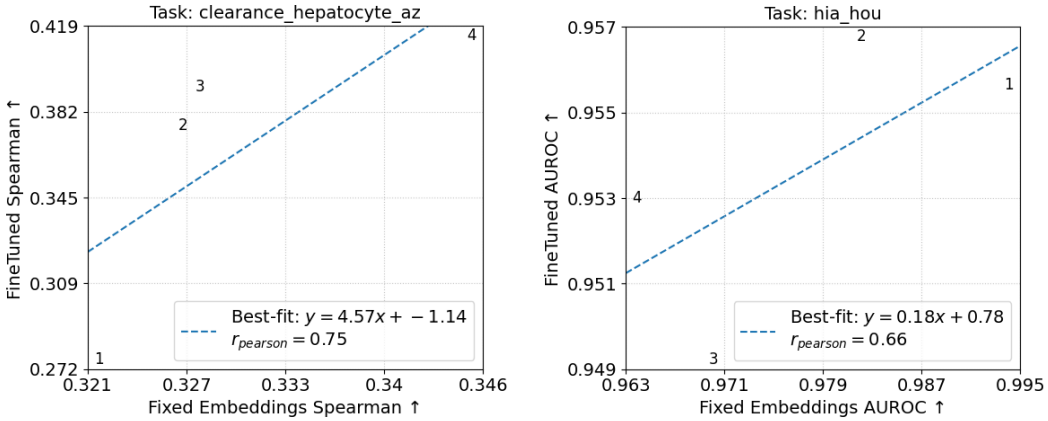


Figure 76: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

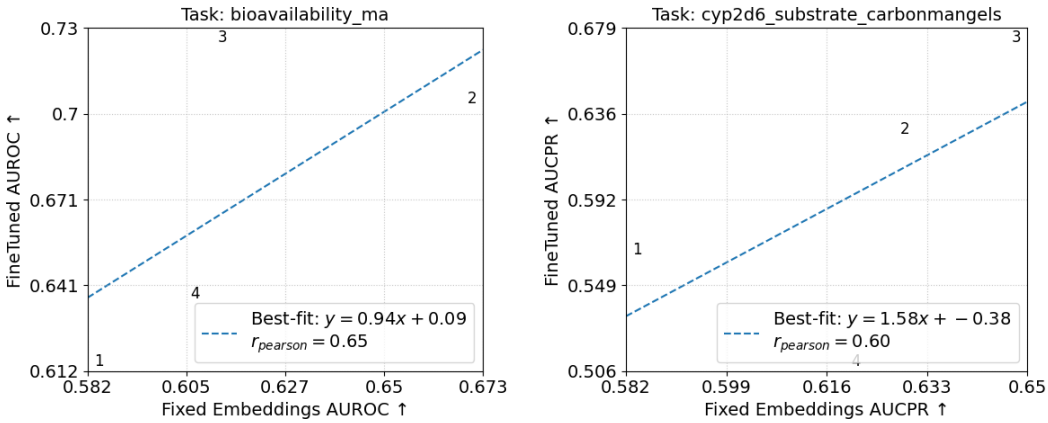


Figure 77: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

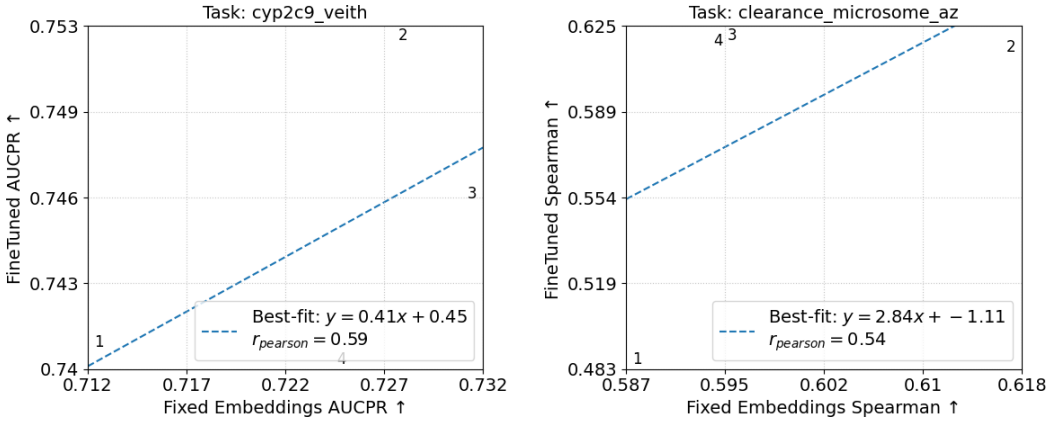


Figure 78: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

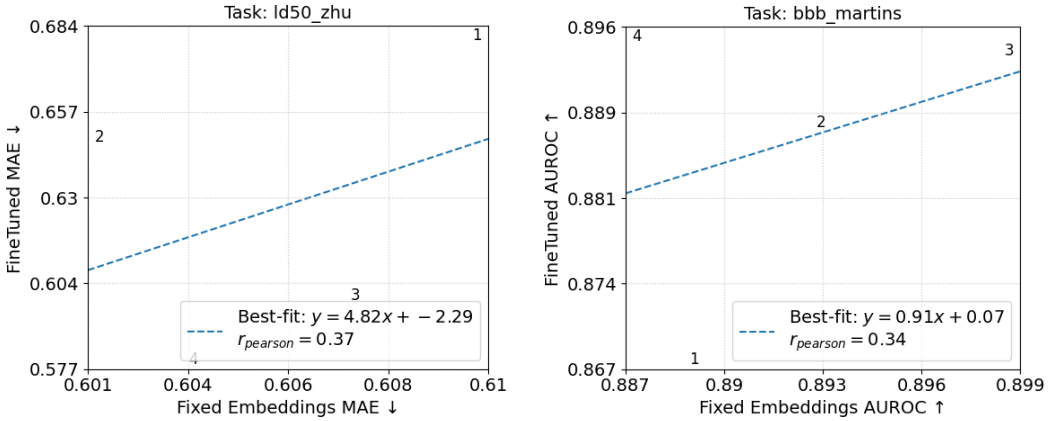


Figure 79: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

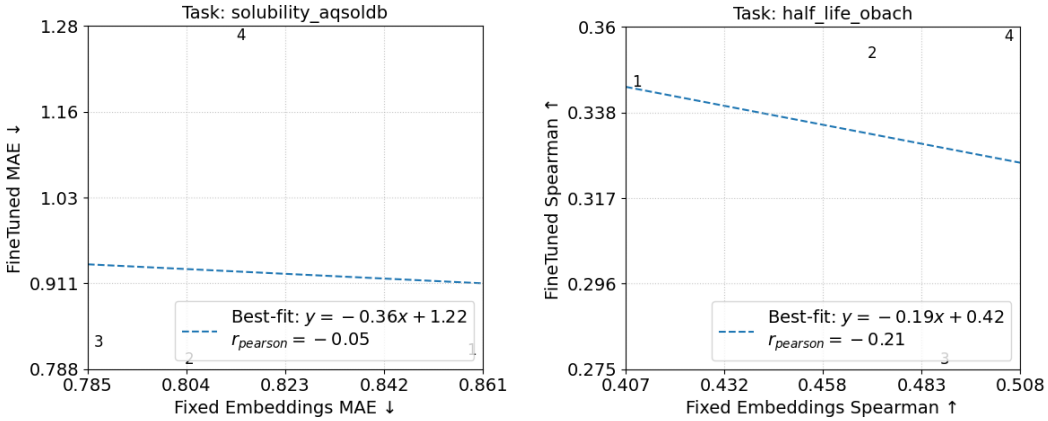


Figure 80: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

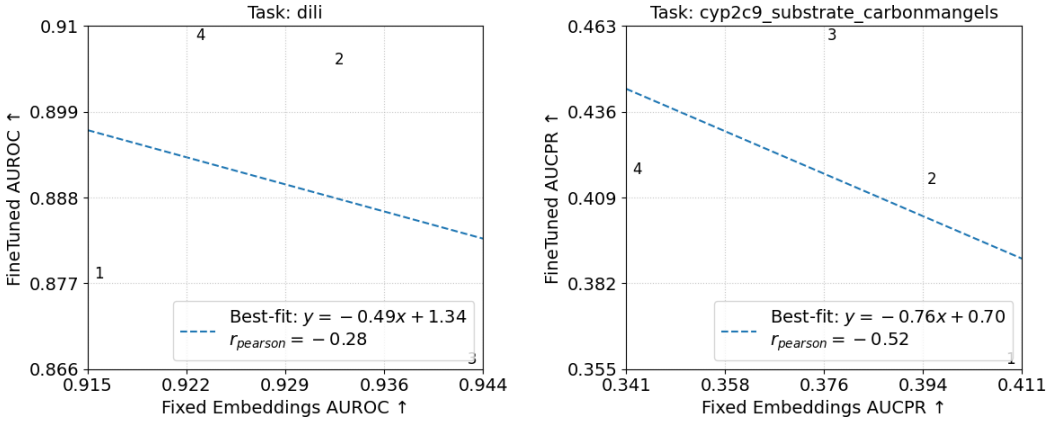


Figure 81: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.

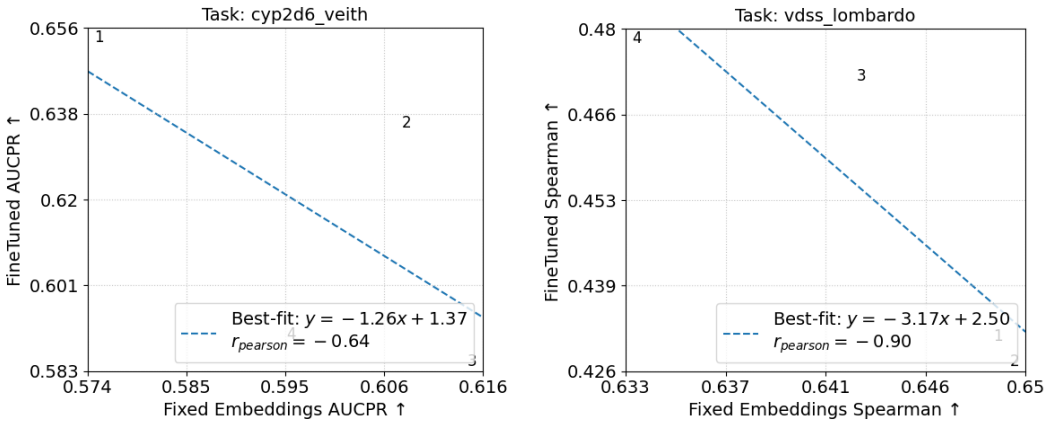


Figure 82: Frozen embeddings vs. finetuned performance for every layer of Pos-EGNN.