



CHALMERS
UNIVERSITY OF TECHNOLOGY



UNIVERSITY OF GOTHENBURG



Photo by Dimitry Kooijmans on Unsplash

Symmetry-Embedded Siamese Neural Networks for Regression Tasks

Master's thesis in Complex Adaptive Systems

Gustav Burman

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

MASTER'S THESIS 2024

Symmetry-Embedded Siamese Neural Networks for Regression Tasks

GUSTAV BURMAN



UNIVERSITY OF
GOTHENBURG



CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Division of Data Science and AI
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Symmetry-Embedded Siamese Neural Networks for Regression Tasks
GUSTAV BURMAN

© GUSTAV BURMAN, 2024.

Supervisor: Hongtao Zhao, AstraZeneca; Janosch Menke, Division of Data Science and AI
Examiner: Ola Engkvist, Division of Data Science and AI

Master's Thesis 2024
Department of Computer Science and Engineering
Division of Data Science and AI
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in L^AT_EX
Gothenburg, Sweden 2024

Symmetry-Embedded Siamese Neural Networks for Regression Tasks

GUSTAV BURMAN

Department of Computer Science and Engineering

Chalmers University of Technology and University of Gothenburg

Abstract

Machine learning (ML) has gained momentum in early drug discovery by reducing the number of expensive and time consuming real-world experiments. One class of ML that has recently shown promise are siamese neural networks (SNN) which take a pair of inputs and predicts the difference in a property, rather than the absolute property. It follows that such a model should be anti-symmetric with regards to the order of the inputs. We introduce a new design for an SNN that has this symmetry embedded directly into the architecture to increase the reliability of the model. By pairing a test compound with a series of reference compounds the model can produce an ensemble of predictions where the mean can be treated as the final prediction and the variance can be used as an uncertainty measure. The symmetry embedded SNN (SE-SNN) shows comparable performance to baseline models on five chemical datasets.

Keywords: Machine Learning, Siamese Neural Network, Symmetry, Uncertainty, Drug Discovery

Acknowledgements

First and foremost I would like to express my deepest gratitude to Hongtao Zhao and Janosch Menke for supervising this project. Thank you for your continued support and guidance throughout the entire project, I couldn't have done this without you. I would also like to extend my sincere thanks to my examiner Ola Engkvist, for making the project a reality. I am also thankful to Eva Nittinger and Christian Tyrchan for their valuable feedback during the course of the project. Special thanks to Mattias Wiklund for acting as opponent for my project, and for his invaluable feedback. Lastly I'd like to acknowledge my friends and family, who have kept my spirits high during this process.

Gustav Burman, Gothenburg, June 2024

Contents

1	Introduction	1
2	Method	2
2.1	Network architecture	2
2.2	Pairing strategies	3
2.2.1	Similarity based pairing	3
2.2.2	SALI pairing	4
2.2.3	Random pairing	4
2.3	Data preparation and model evaluation	4
2.4	Baseline models	5
3	Results	5
3.1	Importance of Anti-symmetry	5
3.2	Effect of different pairings on performance	6
3.3	Applicability domain	7
3.4	Effect of number of reference compounds	8
3.5	SNN with embedded symmetry compared to baseline models	9
4	Discussion	10
4.1	Origin of uncertainty	10
4.2	Correlation between uncertainty and model performance	11
5	Conclusion	12
A	Datasets	I
B	SNN without embedded symmetry	II
C	Tanimoto similarity scatter plots	III

1 Introduction

The drug discovery process can be very expensive and time-consuming. It involves optimizing multiple properties of the drug, and ensuring that they have appropriate absorption, distribution, metabolism, excretion, and toxicity (ADMET) profiles [1]. In the search for such compounds, numerous candidates are usually experimentally tested to evaluate their properties and determine the most viable option. Unfortunately, these experiments are very time-consuming and labor intensive [2], which limits the number of compounds that can be investigated. To accelerate the drug discovery process, and to enable investigation of a larger chemical space, computational approaches are commonly introduced into the process [3]. This enables the researchers to try and predict the chemical properties of a set of potential drugs, and only synthesize the most promising candidates. Thus reducing the number of real world experiments necessary.

However, creating accurate predictive models presents several challenges. One major issue is the so-called activity cliffs, where small changes in the chemical structure can lead to large changes in the physicochemical properties [4]. Additionally, non-additivity of molecular properties poses a problem, as the property of a molecule cannot always be directly inferred from its parts [5, 6]. Today, many computational approaches are based on deep neural networks [3]. One class of neural networks that has seen a rise in popularity for molecular property prediction are so called Siamese Neural Networks (SNN) [7, 8]. Unlike traditional neural networks, SNNs take two inputs and predicts a relationship between these. Originally they were introduced as a similarity measure between images, with use cases like facial recognition [9, 10] and signature verification [11, 12], but was later adapted to regression tasks like chemical property prediction [7, 13, 8]. During regression the network is trained to predict the difference in a property between the two inputs. The goal of the SNN then becomes to approximate the function, where

$$f(x_i; x_j) = y_i - y_j \quad (1)$$

Consequently, to estimate the absolute value of an input x_i , it must be matched to an input x_j whose label is already known, allowing the calculation of the absolute prediction according to:

$$\hat{y}_i = f(x_i; x_j) + y_j \quad (2)$$

One advantage of predicting the difference between two inputs is that each input can be paired with multiple references x_j ($0; 1; \dots; n$) whose label is already known, which means that it becomes very easy to create an ensemble of predictions. The mean prediction of this ensemble can be used as the final prediction, and the variance can be used as an uncertainty estimation.

$$y_i = \frac{\sum_{j=1}^n [f(x_i; x_j) + y_j]}{n} \quad (3)$$

$$\sigma_i^2 = \frac{\sum_{j=1}^n [f(x_i; x_j) + y_j - y_i]^2}{n - 1} \quad (4)$$

It follows intuitively from equation 1 that the network should be antisymmetric, e.g. $f(x_i; x_j) = y_i - y_j = -f(x_j; x_i) = -(y_j - y_i)$. Previously, this condition has only been enforced during

Figure 1: Schematic of the network structure. The network takes two inputs ($x_1; x_2$) and processes them through two identical DL models with shared weights. The hidden states are stacked horizontally into matrices ($h_1h_2; h_2h_1$), and a series of convolutional layers are applied to these matrices. The resulting feature vectors are concatenated, passed through a feed forward neural network, and the difference $z = z_1 - z_2$ is trained against y , ensuring antisymmetry in the network $SE-SNN(x_1; x_2) = -SE-SNN(x_2; x_1)$.

training. What we propose on the other hand, is a Symmetry Embedded SNN (SE-SNN), which has symmetry integrated directly into the design of the network, fulfilling this condition regardless of the trainable weights.

2 Method

2.1 Network architecture

The SE-SNN was implemented in Python using the Pytorch library. A schematic representation of our proposed network structure is provided in Figure 1. The proposed architecture accepts two molecules ($x_1; x_2$) as input, each of which is processed by an identical deep learning (DL) model that maps these inputs to the latent vector ($h_1; h_2$) which are then stacked horizontally to form the matrices ($h_1h_2; h_2h_1$). These matrices undergo convolutional layers, each with kernel size (1; 2), and the resulting vectors are then concatenated and processed through identical readout models. The final stage involves computing the difference between the output states of these layers, denoted as $z = z_1 - z_2$. The absolute property of a molecule can then be predicted by pairing it with a molecule with a known property and adding the value of the known molecule to the model output according to equation 2.

For our DL model we used a simple multilayer perceptron (MLP) comprising three hidden

layers. Layer sizes varying from (x 64 x 32 x 8) to (x 2048 x 1024 x 256) by powers of two were tried. However, with the exception of the two architectures with the smallest layers, which had a higher error, no significant effect on performance was observed. Eventually an architecture with layers of size (x 512 x 256 x 64) interspersed with ReLU activation functions was chosen, as this architecture has previously shown good performance in regression tasks [14]. A single convolutional filter was used, and the readout model consisted of one hidden layer with a ReLU activation function and 64 neurons. The count-based ECFP4 (extended connectivity fingerprint with a radius of 2) fingerprint folded into 1024 bits was used as input.

2.2 Pairing strategies

Given N compounds in a training set, there are a total of N^2 possible pairs. When we embed symmetry into the network the order of the pairs doesn't matter, e.g. if we create the pair $(i; j)$ then the pair $(j; i)$ becomes redundant. Likewise, pairing a compound with itself will always predict 0 regardless of the weights of the network. This leaves us with $\frac{N^2 - N}{2}$ potential pairs. It was previously shown that using only the most similar pair for each molecule in the training set can be enough to train an SNN [15]. Here we investigate further strategies for pairing the training molecules and their effect on the model performance.

2.2.1 Similarity based pairing

One assumption is that the network would find it easier to predict more similar compounds, and thus we can tune the network towards this easier problem by training and evaluating on the most similar pairs. On the other hand, the difference between two similar molecules contains less chemical information than the difference between less similar molecules. Therefore pairing each molecule with both its most and its least similar counterparts was tried.

The similarity between compounds, measured using the Tanimoto coefficient, was determined using RDKit (<https://www.rdkit.org/>) employing the count-based ECFP4. Compounds were ranked by their highest similarity, and the similarity matrix was computed. The top ranked molecule was then paired with the most similar compounds, and the conjugate pairs were removed from consideration to prevent redundancy (Fig. 2).

Figure 2: Example of most similar pairing with 5 training molecules and a single pair per training molecule. Cell $(i; j)$ contains the Tanimoto similarity for molecules $(i; j)$, excluding the diagonal $(i; i)$. Starting at Mol 1; each molecule is paired with its most similar counterpart, and the conjugate pair is removed from consideration. Dark blue denotes the selected pairs, crossed out grey indicates the pairs that have been excluded, and light blue represents pairs that were not chosen.

2.2.2 SALI pairing

Activity cliffs provide a big challenge while developing computational models, since small changes in molecular structure can have a large impact on the chemical properties [4, 16]. Training the model on these activity cliffs might give it a better chance to learn where small changes in structure lead to large changes in property.

The Structure-Activity Landscape Index (SALI) can be used to pinpoint these activity cliffs [17]. The SALI score can be calculated by taking the absolute difference of the measured property and dividing it by the distance between the compounds (Equation (5)). The SALI score was calculated for each possible pair in the training set and the pairs with the highest SALI score were chosen the same way as the similarity pairing (Figure 2).

$$\text{SALI}(i;j) = \frac{|P_i - P_j|}{1 - s(i;j)} \quad (5)$$

2.2.3 Random pairing

To evaluate the impact of different pairing strategies, a baseline is needed. The simplest strategy for pairing the training compounds is to simply pair each compound with another random compound within the training set. This random pairing was used as a reference for all other pairing strategies.

2.3 Data preparation and model evaluation

For all analyses, we used four of the six endpoints in the public ADME dataset made available by Biogen [18]. We used the endpoints solubility (logS), human and rat liver microsomal stability (logHLM, logRLM), and the MDR1_MDCK efflux ratio (logER). Human and rat plasma protein binding data sets were omitted, as they contained only 194 and 168 compounds respectively. Furthermore, we used the Lipophilicity (logD) dataset available at MoleculeNet [19]. All data sets contained log-transformed data.

For the HLM and RLM endpoints, 958 and 346 compounds, respectively, were below the detection range and thus excluded from the analysis. The RLM data set also had two compounds above the detection range, which were also excluded. In addition, the SMILES inputs were preprocessed by removing salts, canonicalizing, filtering compounds containing elements other than H, C, N, O, S, F and Cl, and filtering duplicates. Statistics of the processed datasets can be seen in Table 1. The distributions of the datasets can be seen in Appendix A.

Table 1: Summary of the 5 physicochemical datasets

	logS	logHLM	logRLM	logER	logD
Size	2127	2084	2648	2585	4089
Min	-1.00	0.68	1.03	-1.16	-1.50
Max	2.18	3.37	3.85	2.73	4.50
Mean	1.26	1.61	2.41	0.40	2.18
Median	1.54	1.56	2.44	0.16	2.35
SD	0.68	0.54	0.65	0.69	1.20

The model was evaluated using 5-fold cross-validation, where the same seed was used for all

analyses. To prevent data leakage, the data was split into train and test sets prior to pairing. While training the model, training set molecules were then paired with other training set molecules. When evaluating the model on the test set, molecules in the test set were paired with molecules in the training set and the absolute property of the test set was inferred using Equation (2).

2.4 Baseline models

To evaluate the performance of the SE-SNN we compared it to a range of already established models. These being random forest (RF), multilayer perceptron (MLP), XGBoost (XGB), support vector machine (SVM), Gaussian process regression (GPR) and k-nearest neighbor regression (kNN). A random search was used to optimize the hyperparameters of the baseline models.

The MLP used the same architecture as the DL model in the SE-SNN with an included output ($x 512 \times 256 \times 64 \times 1$), using ReLu activation functions between every layer. RF, XGB, SVM and GPR were implemented in scikit-learn. RF using 1478 trees, a max depth of 30 and a minimum number of samples to split of 5. XGB using 830 trees, a max depth of 7 and a learning rate of 0.01. SVM with an rbf kernel, a C of 10, a degree of 3, of 0.1, and a of 0.01. The GPR used an rbf kernel with length scale of 1.0 and an of 0.09. The kNN used the ve nearest neighbors. All other parameters was kept as the scikit-learn default. All baseline models used the count-based ECFP4 ngerprint folded into 1024 bits as input, and they all processed molecules individually to directly predict the absolute value of the respective properties.

3 Results

3.1 Importance of Anti-symmetry

To show the importance of anti-symmetrical predictions for the performance of the network, a model with embedded anti-symmetry, and a model without embedded anti-symmetry (network structure can be seen in Appendix B) were trained for 50 epochs using the most similar pair. The nal weights for the convolutional lter were compared to the MAE of the predictions. The results are shown in Figure 3. The network without embedded anti-symmetry requires the convolutional weights to converge to subtraction, which is the only anti-symmetric operation using a single convolutional lter, for accurate predictions. However, for the network with embedded anti-symmetry the nal weights of the convolutional lter do not impact the performance of the model. Indicating that building antisymmetry into the model improves its robustness.

Figure 3: Weights w_1 and w_2 of the convolutional layer for a network without embedded symmetry (top), and a network with embedded symmetry (bottom). Both networks were trained on the most similar pair for 50 epochs. The dashed orange line corresponds to weights equivalent to subtracting the hidden states.

3.2 Effect of different pairings on performance

The performance of different pairing strategies in the 5-fold cross-validation is shown in Figure 4. This evaluation encompasses training on a single pair per column in the similarity matrix (A); specifically, the most similar, the least similar, a randomly chosen pair, and the pair with the highest SALI score. As can be seen in the figure, the various pairing strategies demonstrate comparable performance for all data sets, with the exception of the SALI pairing, which performs significantly worse on all datasets.

Likewise, a few combined strategies were implemented for training (B). These being the most similar + the least similar pairs, the most similar + the least similar + the pair with the highest SALI score, three random pairs and the three most similar pairs. Again most pairing strategies have similar performance. But interestingly, training on the most similar and the least similar pairs (2 pairs) performed marginally better than training on the most similar, least similar, and the pair with the highest SALI score (3 pairs).

One reason for the worse performance of SALI pairings could be that, since the network is exposed to pairs manifested by activity cliffs, which represents a small fraction of all possible pairs, it overfits to the activity cliffs and fails to generalize to the rest of the data.

The similarity between compounds in the Biogen datasets is generally quite low, which could mean that there is less of an advantage to train on the most similar pairs. The logD dataset on the other hand has a higher similarity between the compounds, and looking at Figure 4 it can

be noted that using the three most similar pairs leads to marginally better performance than the other strategies for this dataset. Because of this the three most similar pairs were used for the further analyses.

Figure 4: MAE for models trained with various pairs. Most similar in blue, least similar in green, random in yellow, highest SALI in red (A). MAE for models using different combinations of pairing strategies: most similar and least similar in blue; most similar, least similar, and highest SALI in green; three random pairs in yellow; and three most similar pairs in red (B). All pairing strategies were evaluated on the test-train pair with the highest Tanimoto similarity.

3.3 Applicability domain

Figure 5 shows the difference in experimental property as a function of the Tanimoto similarity, where each compound within the dataset is paired based on the maximal Tanimoto similarity (top row). The logD dataset shows two peaks at similarities approximating 0.2 and 0.8 respectively (left), whereas the logHLM dataset exhibit a singular peak at approximately 0.2 (right). Furthermore, the logHLM dataset demonstrates a narrower distribution with respect to the property difference compared to the logD dataset. The other Biogen datasets showed similar behaviour to the logHLM dataset, and can be seen in Appendix C.

It is expected that molecules with high similarity will exhibit a small difference in property [20]. And there does indeed seem to exist a correlation between increasing Tanimoto similarity of the compound pairs and a smaller difference in their respective properties.

To assess the impact of Tanimoto similarity on prediction error, each molecule in the test set was paired with the 100 most similar molecules in the training set. These pairs were fed to an SE-SNN model that was trained using the three most similar pairs. The prediction errors were then calculated for each pair and measured against the degree of similarity between the molecules in each pair (bottom row of Fig. 5). A mild correlation was observed between prediction errors and Tanimoto similarity, suggesting that higher similarity improves prediction accuracy.

Figure 5: Correlation of experimental logD (left) and logHLM (right) with the Tanimoto similarity of compounds, each paired with the singularly most similar compound across the dataset (top). Correlation of the prediction error and Tanimoto similarity for each compound in the test split paired with the 100 most similar compounds from the training split (bottom). The dashed red lines correspond to the 95th percentile of the distribution.

3.4 Effect of number of reference compounds

Next the effect on the number of references during evaluation was investigated. A model was evaluated by pairing each test compound with the 1-20 most similar training compounds, and the mean of the predictions was used. As can be seen in Figure 6, the number of references have very small effects on model accuracy. Only the logD dataset showed any effect on performance, where using more than 3 references leads to a slowly increasing error. The lowest error (using 3 reference) and the highest error (using 20 references) differed by ≈ 0.01 . For all other datasets, no effect of the number of references on performance was observed. Since the test-train pairs are based on maximal Tanimoto similarity, using more pairs for the ensemble will lead to a lower average similarity of the ensemble. And as previously observed, more similar pairs tend to have a lower prediction error. Average similarity of the most similar pairs for the logD dataset is rather high, while for the Biogen datasets it is much lower, which could explain why there is an upward trend in the MAE for the logD dataset but not for the others. To balance performance of the model while still using enough references to reliably calculate the variance of the ensemble, ten references were chosen.

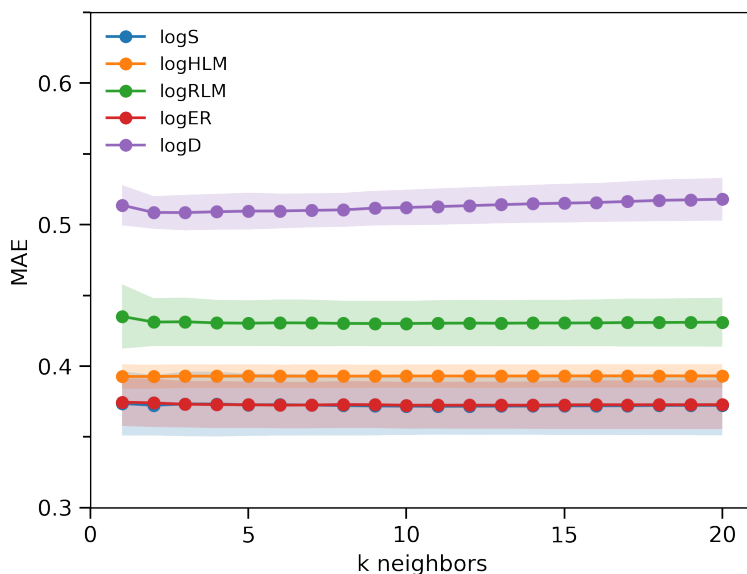


Figure 6: MAE as a function of the number of reference compounds used for evaluation of a model trained on the 3 most similar pairs. The shaded area corresponds to the standard deviation from the 5-fold cross validation.

3.5 SNN with embedded symmetry compared to baseline models

To compare the performance of our model (SE-SNN) with a range of baseline models, we trained random forests (RF), multilayer perceptrons (MLP), XGBoost (XGB), support vector machines (SVM), gaussian process regression (GPR) and k-nearest neighbour regression (kNN).

Table 2 shows the mean absolute error of the SE-SNN compared to the baseline models for the five data sets.

The SE-SNN shows performance comparable to the baseline models. Performing better than RF, XGBoost, and kNN on all datasets and having a very similar performance to MLP. The SE-SNN performs slightly worse than SVM and GPR on the logHLM and logRLM datasets, but slightly better on the logS dataset.

Table 2: MAE for symmetry embedded Siamese Neural Network (SE-SNN) compared to baseline models; Random Forest (RF), Multilayer Perceptron (MLP), XGBoost (XGB), Support Vector Machine (SVM) and k-Nearest Neighbors (kNN) with $k = 5$. The data points correspond to the mean and standard deviation for the 5-fold cross validation.

Model	Solubility	HLM	RLM	MDR1_MDCK	LogD
RF	0.40 (0.01)	0.40 (0.01)	0.45 (0.01)	0.41 (0.01)	0.61 (0.02)
MLP	0.37 (0.01)	0.40 (0.02)	0.43 (0.01)	0.39 (0.02)	0.54 (0.01)
XGB	0.40 (0.01)	0.40 (0.01)	0.44 (0.01)	0.40 (0.01)	0.62 (0.01)
SVM	0.39 (0.01)	0.37 (0.01)	0.40 (0.01)	0.37 (0.01)	0.51 (0.01)
GPR	0.39 (0.01)	0.37 (0.00)	0.39 (0.02)	0.37 (0.01)	0.51 (0.01)
kNN($k = 5$)	0.44 (0.02)	0.45 (0.01)	0.53 (0.01)	0.49 (0.01)	0.76 (0.02)
SE-SNN	0.37 (0.02)	0.39 (0.01)	0.43 (0.02)	0.37 (0.02)	0.51 (0.01)

4 Discussion

The SE-SNN has a competitive performance compared to the baseline models. However, a large advantage of the SE-SNN is its ability to create ensembles of prediction and using the variance as an uncertainty measure. By approximating the SE-SNN as a linear model we can investigate where this uncertainty actually comes from.

4.1 Origin of uncertainty

It has previously been shown that if an SNN is pairwise separable the resulting variance of an ensemble of predictions is solely dependent on the reference set used [8]. If we approximate the model as linear we can separate the two inputs and we get

$$\Delta \hat{y}_i = \text{SE-SNN}(x_i, x_j) - f(x_i) - f(x_j) \quad (6)$$

Furthermore, if we calculate the mean of an ensemble using this approximation we get

$$\bar{y}_i = \frac{\sum_{j=1}^n [\text{SE-SNN}(x_i, x_j) + y_j]}{n} \quad (7)$$

$$\frac{\sum_{j=1}^n [f(x_i) - f(x_j) + y_j]}{n} \quad (8)$$

$$= f(x_i) + \frac{\sum_{j=1}^n [y_j - f(x_j)]}{n} \quad (9)$$

We can see that the second term is the average fitted error of the reference compounds. For a given reference set this will be a constant and we can define a variable

$$\bar{e} = \frac{\sum_{j=1}^n [y_j - f(x_j)]}{n} \quad (10)$$

as the average fitted error for the reference set. And thus $\bar{y}_i = f(x_i) + \bar{e}$. Using this to calculate the variance for an ensemble $\hat{\sigma}_i^2$ we get

$$\hat{\sigma}_i^2 = \frac{\sum_{j=1}^n [\text{SE-SNN}(x_i, x_j) + y_j - \bar{y}_i]^2}{n - 1} \quad (11)$$

$$\frac{\sum_{j=1}^n [f(x_i) - f(x_j) + y_j - f(x_i) - \bar{e}]^2}{n - 1} \quad (12)$$

$$= \frac{\sum_{j=1}^n [\hat{e}_j - \bar{e}]^2}{n - 1} \quad (13)$$

where $\hat{e}_j = y_j - f(x_j)$ is the fitted error of reference compound j . Thus the variance $\hat{\sigma}_i^2$ is equivalent to the variance of the fitted errors for the reference set.

Figure 7 shows the experimentally measured logD against the predicted logD for 20 different test compounds, each paired with the same 50 reference compounds, for an SE-SNN. Each point corresponds to a test-ref prediction, and the different lines of points correspond to the different test compounds. We can see that the distributions of predictions for the different test compounds are very similar. Additionally the standard deviation of each test compound was calculated with an average of $\mu(\hat{\sigma}_i) = 0.18$ and a standard deviation $(\hat{\sigma}_i) = 0.0016$. The

small standard deviation with regards to the mean indicates that the standard deviation for each test compound is in large part determined by the choice of reference compounds.

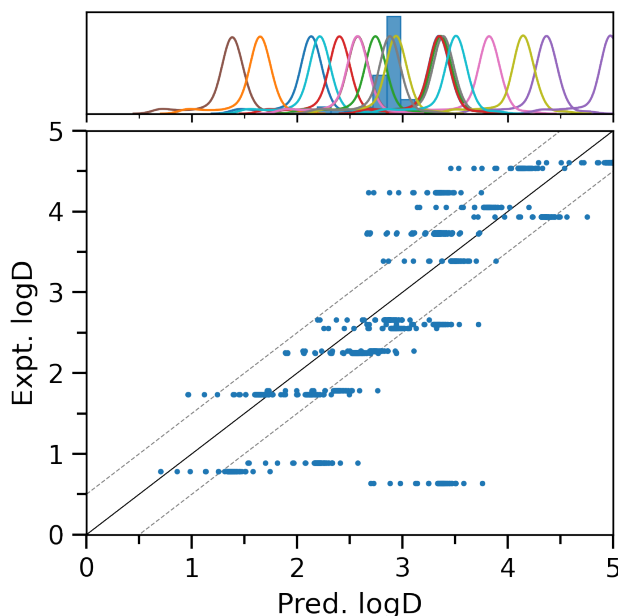


Figure 7: Predicted logD vs experimentally measured logD for 20 test compounds paired with the same 50 random references. Top figure shows the approximated density distribution of the predictions for each test compound.

4.2 Correlation between uncertainty and model performance

To visualize the correlation between uncertainty and performance, confidence curve plotting was used, where data points are ordered by their uncertainty and variations in performance are evaluated during sequential removal of the most uncertain data points [21]. The confidence curves for standard deviation (σ) and mean absolute error (MAE) for the SE-SNN and the GPR can be seen in Figure 8.

The plot for σ shows the ordered standard deviation for each prediction, corresponding to the distribution of the σ . We can see that when predicting on the Biogen datasets using the SE-SNN 70% of the predictions have a σ very near to zero. When predicting on the logD dataset however, the predictions have a more varied distribution of σ . Likewise, for all datasets using the GPR we can see a varied distribution of σ .

For the confidence plot for the MAE the MAE is calculated based on the remaining predictions during the sequential removal of the most uncertain predictions. Looking at how this removal of uncertain predictions affect the MAE we can see the same general trends for both the SE-SNN and the GPR. For the logS and logD datasets the MAE for both models is steadily decreasing with decreasing uncertainty, and for the logHLM and logRLM datasets both models show little to no correlation between uncertainty and prediction error. Where we do see a difference is for the ER dataset, where the GPR shows a decrease in error for less uncertain predictions while the SE-SNN does not. These results seem to indicate that the relationships between uncertainty and model performance is not only model dependent, but also dataset dependent. Another thing to notice is that the GPR shows a stronger correlation between uncertainty and

prediction error for the logD and logER datasets than the SE-SNN for the same datasets. Additionally, while the GPR and the SE-SNN have about equal performance on the logD and logER datasets when 100% of the data is included, the GPR outperforms the SE-SNN when the predictions with highest uncertainty are excluded. Meaning, in turn, that the SE-SNN outperforms the GPR on predictions with a high uncertainty.

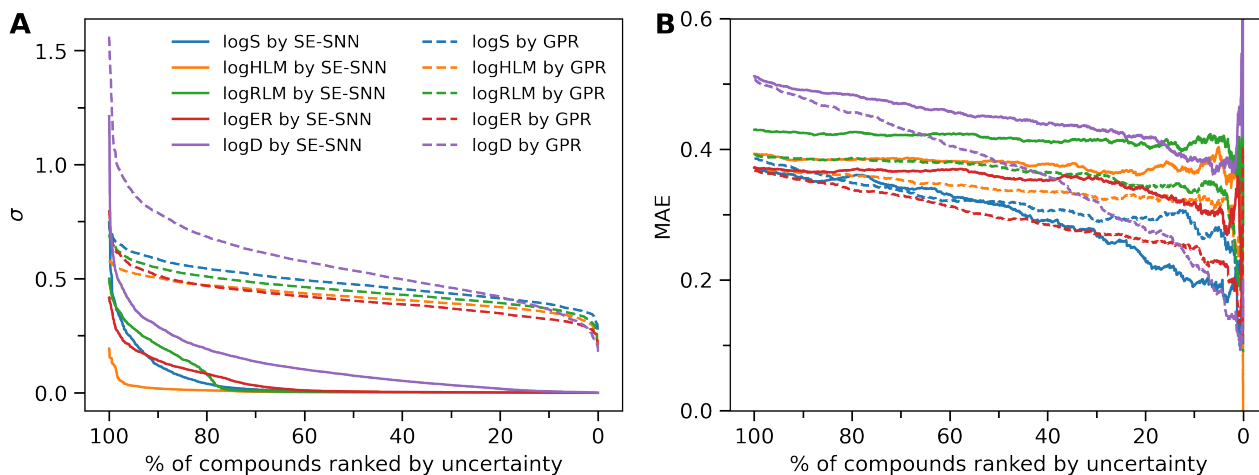


Figure 8: Standard deviation (A) and mean absolute error (B) as a function of the fraction of the least uncertain compounds included.

5 Conclusion

We have proposed embedding symmetry into the architecture of a Siamese Neural Network. Our results show that the model is robust with regards to hyperparameters, and while SNNs without embedded symmetry can in certain cases run into convergence problems, these convergence issues are in large part ameliorated by embedding symmetry into the model. The SE-SNN can easily create ensembles of predictions where the standard deviation of the ensemble can be used as an uncertainty measurement, and in some cases this uncertainty correlates with the prediction error for the model.

References

- [1] H. van de Waterbeemd and E. Gifford, “Admet in silico modelling: towards prediction paradise?,” *Nature Reviews Drug Discovery*, vol. 2, pp. 192–204, Mar 2003.
- [2] J. W. Scannell, A. Blanckley, H. Boldon, and B. Warrington, “Diagnosing the decline in pharmaceutical r&d efficiency,” *Nature Reviews Drug Discovery*, vol. 11, pp. 191–200, Mar 2012.
- [3] A. Cherkasov, E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dearden, P. Gramatica, Y. C. Martin, R. Todeschini, V. Consonni, V. E. Kuz'min, R. Cramer, R. Benigni, C. Yang, J. Rathman, L. Terfloth, J. Gasteiger, A. Richard, and A. Tropsha, “Qsar modeling: Where have you been? where are you going to?,” *Journal of Medicinal Chemistry*, vol. 57, pp. 4977–5010, Jun 2014.
- [4] M. Cruz-Monteagudo, J. L. Medina-Franco, Y. Pérez-Castillo, O. Nicolotti, M. N. D. Cordeiro, and F. Borges, “Activity cliffs in drug discovery: Dr jekyll or mr hyde?,” *Drug Discovery Today*, vol. 19, no. 8, pp. 1069–1080, 2014.
- [5] C. Kramer, “Nonadditivity analysis,” *Journal of Chemical Information and Modeling*, vol. 59, pp. 4034–4042, Sep 2019.
- [6] K. Kwapien, E. Nittinger, J. He, C. Margreitter, A. Voronov, and C. Tyrchan, “Implications of additivity and nonadditivity for machine learning and deep learning models in drug design,” *ACS Omega*, vol. 7, pp. 26573–26581, Aug 2022.
- [7] Z. Fralish, A. Chen, P. Skaluba, and D. Reker, “Deepdelta: predicting admet improvements of molecular derivatives with deep learning,” *Journal of Cheminformatics*, 2023.
- [8] M. Tynes, W. Gao, D. J. Burrill, E. R. Batista, D. Perez, P. Yang, and N. Lubbers, “Pairwise difference regression: A machine learning meta-algorithm for improved prediction and uncertainty quantification in chemical search,” *Journal of Chemical Information and Modeling*, vol. 61, pp. 3846–3857, Aug 2021.
- [9] E. Solomon, A. Woubie, and E. S. Emiru, “Deep learning based face recognition method using siamese network,” 2024.
- [10] H. Wu, Z. Xu, J. Zhang, W. Yan, and X. Ma, “Face recognition based on convolution siamese networks,” in *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1–5, 2017.
- [11] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Lladós, and U. Pal, “Signet: Convolutional siamese network for writer independent offline signature verification,” *CoRR*, vol. abs/1707.02131, 2017.
- [12] J. Bromley, I. Guyon, Y. LeCun, E. Säcker, and R. Shah, “Signature verification using a “siamese” time delay neural network,” in *Advances in Neural Information Processing Systems* (J. Cowan, G. Tesauero, and J. Alspector, eds.), vol. 6, Morgan-Kaufmann, 1993.
- [13] S. J. Wetzal, K. Ryczko, R. G. Melko, and I. Tamblyn, “Twin neural network regression,” *Applied AI Letters*, vol. 3, no. 4, p. e78, 2022.
- [14] Y. Zhou, S. Cahya, S. A. Combs, C. A. Nicolaou, J. Wang, P. V. Desai, and J. Shen, “Exploring tunable hyperparameters for deep neural networks with industrial adme data sets,” *Journal of Chemical Information and Modeling*, vol. 59, pp. 1005–1016, Mar 2019.
- [15] Y. Zhang, J. Menke, J. He, E. Nittinger, C. Tyrchan, O. Koch, and H. Zhao, “Similarity-based pairing improves efficiency of siamese neural networks for regression tasks and uncertainty quantification,” *Journal of Cheminformatics*, 2023.

-
- [16] G. M. Maggiora, "On outliers and activity cliffs - why qsar often disappoints," *Journal of Chemical Information and Modeling*, vol. 46, pp. 1535–1535, Jul 2006.
- [17] R. Guha and J. H. Van Drie, "Structureactivity landscape index: Identifying and quantifying activity cliffs," *Journal of Chemical Information and Modeling*, vol. 48, no. 3, pp. 646–658, 2008. PMID: 18303878.
- [18] C. Fang, Y. Wang, R. Grater, S. Kapadnis, C. Black, P. Trapa, and S. Sciabola, "Prospective validation of machine learning algorithms for absorption, distribution, metabolism, and excretion prediction: An industrial perspective," *Journal of Chemical Information and Modeling*, vol. 63, no. 11, pp. 3263–3274, 2023. PMID: 37216672.
- [19] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, "Moleculenet: A benchmark for molecular machine learning," 2018.
- [20] G. Maggiora, M. Vogt, D. Stumpfe, and J. Bajorath, "Molecular similarity in medicinal chemistry," *Journal of Medicinal Chemistry*, vol. 57, pp. 3186–3204, Apr 2014.
- [21] G. Scalia, C. A. Grambow, B. Pernici, Y.-P. Li, and W. H. Green, "Evaluating scalable uncertainty estimation methods for deep learning-based molecular property prediction," *Journal of Chemical Information and Modeling*, vol. 60, pp. 2697–2717, Jun 2020.

A Datasets

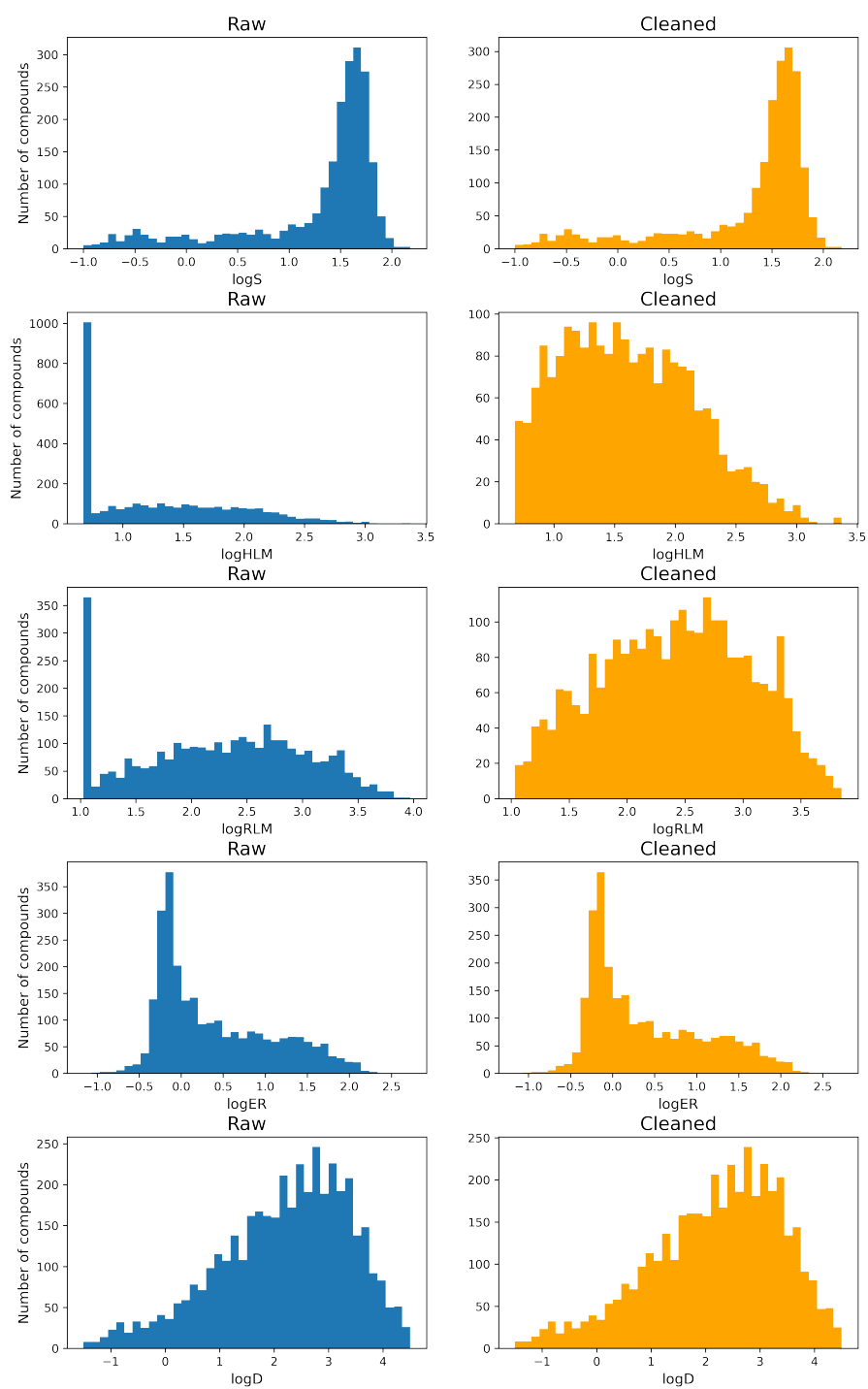


Figure 9: Distribution of the five datasets used in the analyses. The left column contains the raw data and the right column contains the data after preprocessing.

B SNN without embedded symmetry

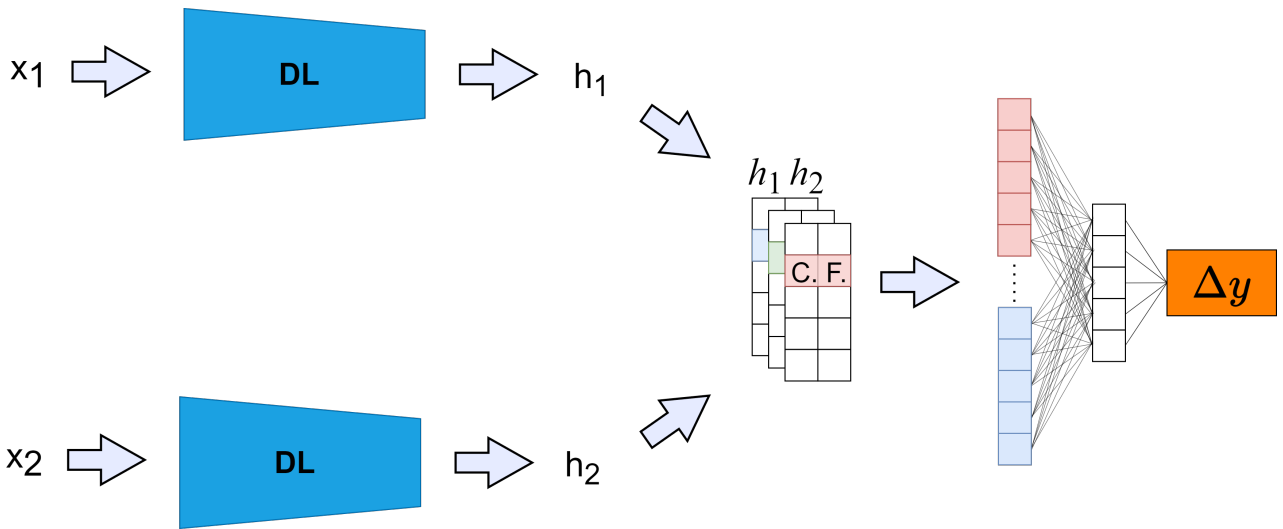


Figure 10: Schematic of the SNN without embedded symmetry.

C Tanimoto similarity scatter plots

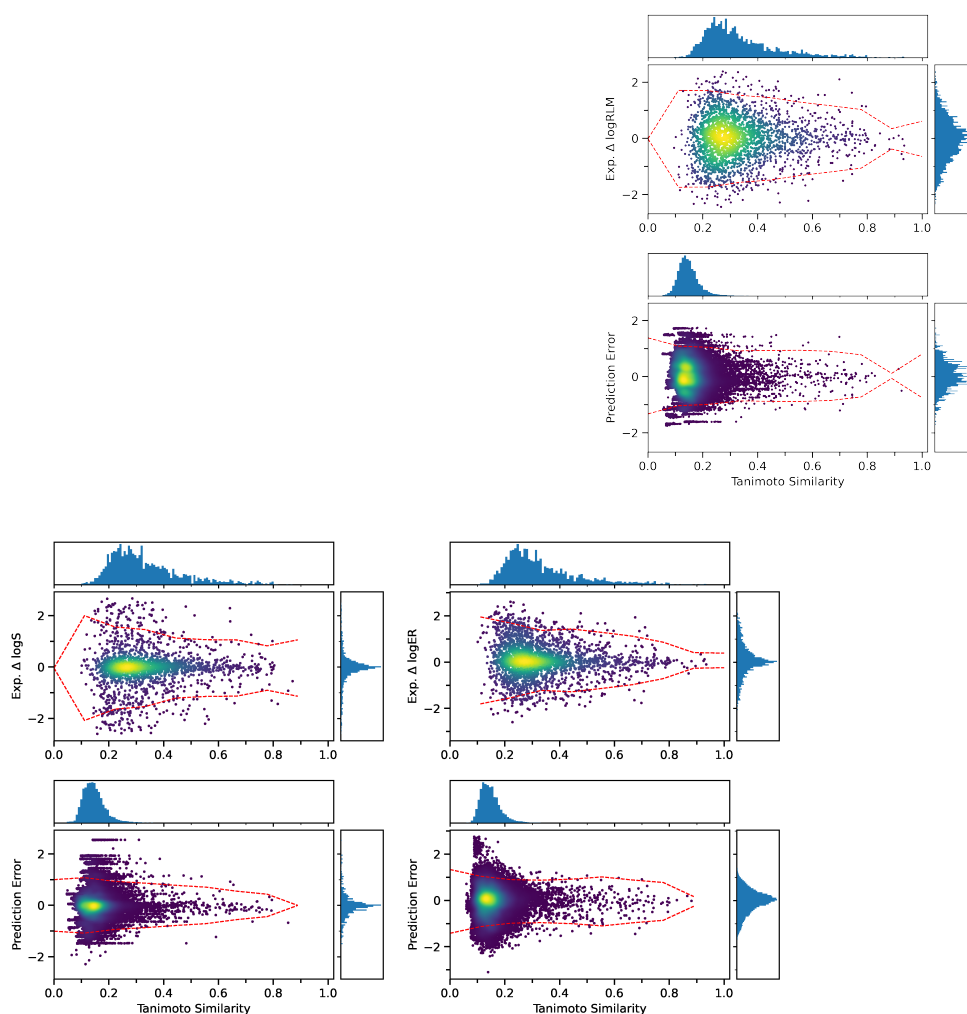


Figure 11: Correlation of experimental property with Tanimoto similarity for the most similar pairs in the dataset (top figures). Correlation of the prediction error and Tanimoto similarity for each compound in the test split paired with the 100 most similar compounds from the training split (bottom figures).