Jens Alexander Lagemann

Prediction of ADME properties using curriculum learning

MASTER'S THESIS

supervised by Dr. Caroline König and Prof. Dr. Alfredo Vellido

Master's Degree in Biomedical Data Science

Madrid, September 3rd 2024

Abstract

The accurate prediction of the pharmacokinetics of arbitrary molecules is the holy grail of computer aided drug development. Being able to correctly model how a compound will behave in the human body could increase the chances of a new medication making it to market significantly. One of the current lines of research involves a kind of deep learning model, called message passing neural networks, applied to molecular graphs. However, as most neural network based techniques, their performance is highly dependent on the dataset provided. To compound this, in the case of molecular properties, datasets are often relatively small. In order to improve the accuracy under these constraints, curriculum learning techniques can be applied to optimize the training procedure. This thesis will study the effect that such a curriculum sampler has when trying to predict a variety of molecular properties, using datasets describing absorption, distribution, metabolism and excretion of various samples. Making a quantitative analysis of how models trained in different configurations compare. The results show that the ordering of samples during training is a relevant factor in how well a model learns to generalize, although the type of molecular complexity measures depends on the application. As well as the importance of repeated trials in deep learning because of inherent variance, introduced via random initialization and shuffling.

Keywords: Curriculum Learning, Molecular Property Prediction, Message Passing Neural Networks, ADME properties

Dr. Caroline König, certifies that the student Jens Lagemann has elaborated the work under her direction and she authorizes the presentation of this Master's Thesis for its evaluation.

Firmado por KONIG CAROLINE LEONORE - ****2524* el día 03/09/2024 con un certificado emitido por AC FNMT Usuarios

Advisor signature: . . . . . .

Contents

List of Figures

List of Tables

1 Introduction

This chapter will provide an overview of the relevance of molecular property prediction and introduce the concepts and terminology necessary to understand the conducted study.

1.1 Motivation

Developing new medication is a very long, arduous and expensive process [1]. As much as 90% of new potential drugs fail in clinical trials [2], meaning they have already passed pre-clinical screenings. The reasons for failure are varied, either they might not show the desired effects or possibly be too toxic to justify their use.

Potential candidate compounds do not only have to be effective in their desired target receptor, but furthermore they have to be correctly absorbed by the body, reach the relevant area in the body and not be too toxic or lead to unacceptable side effects. Therefore, the prediction of molecular properties is a problem as hard as it is relevant.

Improved computational models to better predict the behavior of those molecules could decrease the number of failures and significantly reduce the costs of the development process. But those improvements are not easily achieved. The chemical space is incredibly large, with estimates of between $10^{20} - 10^{60}$ relevant molecules [3]. Besides the sheer number of molecules, there is the complexity of modelling chemical interactions. Classical computational tools that are built on modelling physics are extremely slow in solving all of the required formulas and more limited in their application.

This has lead to the popularity of using some machine learning (ML) approaches to tackle this problem. While they are much more versatile and, at least in inference mode, significantly faster, there is the disadvantage of being potentially unstable or hard to train to achieve state of the art accuracy [4].

One technique that promises potential improvements in making the predictions more exact and possibly increasing convergences during training is called curriculum learning (CL). It is inspired by the human learning process of starting with easier examples and only increase difficulty over time with increased understanding of the problems [5]. It has been applied in a variety of different areas, from classification problems, over natural language challenges to reinforcement learning [6]. This work attempts to design and evaluate such a technique for predicting a variety of molecular properties.

1.2 Objectives

The objective of this work is to explore the effect that a CL approach will have on the performance of Message Passing Neural Network (MPNN) models [7] in predicting various Absorption, Distribution, Metabolism, Excretion (ADME) properties. More specifically a set of different parameters are compared to evaluate the CL performance, including:

1. The change of final root mean square error (RMSE) depending on the function used to calculate the molecular complexity

2. The number of subsets in which the dataset is split for training

3. Evaluating whether an ascending or descending order of molecular complexity is more beneficial to the training outcome

Given as the technique has been successfully applied in a variety of different domains, the hypothesis is that some configurations of CL perform better than a typical randomized sequence of samples.

1.3 State-of-the-art

There is a variety of different graph neural network architectures in use, among which we have MPNN [7].

Furthermore, some work has been made in exploring the use of CL for molecular property prediction, such as that in Gu et al [8]. In their work, authors evaluated five different graph neural network architectures and compared them across 9 benchmark datasets, with the number of samples varying between 642 and 41127. More in detail, FreeSolv, ESOL, Monoamine, JAK2, HERG and Lipohlicity are regression benchmarks while BBBP, BACE and HIV are classification problems. Their implementation used a difficulty measure, a cumulative distribution function to normalize the distribution of difficulty values in the dataset. Finally, they used a training scheduler to present samples according to a competence function. So, starting with only simple examples, as the model progresses through training and improves the loss, the training scheduler includes samples with higher difficulty ratings. In their work, they achieved an improvement in the test set on 11 out of 12 regression experiments over the control randomized sampler. The relative improvement of RMSE varied between 2% and 25% on these benchmarks. In the classification experiments, their curriculum sampler performed better in each of the 20 trials, with a relative improvement between 0.146% and 4.838% of the area under the ROC curve.

2 Background

The following sections will cover relevant information to contextualize the field of application, as well as technical implementations.

2.1 Drug Design Definitions

One of the main goals of this line of research is to contribute to the computer-aided drug discovery pipeline. The following paragraphs will introduce the core concepts underlying this goal.

2.1.1 ADME properties

A variety of molecular properties are of interest in the de novo drug design process. Those properties are usually grouped into absorption, distribution, metabolism and excretion (ADME, as previously described) [9].
Absorption, the first of those properties, refers to how a drug enters the bloodstream after administration. Key factors include solubility, permeability, and stability in the gastrointestinal tract. Lipophilicity and molecular size are also critical, as they affect how easily a drug crosses cellular membranes.
After being absorbed, a drug is distributed throughout the body. The extent and rate of distribution depend on factors like blood flow, tissue permeability, and binding to plasma proteins.
Drug metabolism refers to the enzymatic conversion of drugs into more water-soluble compounds for excretion. The liver is the primary site for this process. After the compound has been fully metabolized, remaining traces and waste products are excreted, usually via renal and biliary routes.

All of these are important factors during the evaluation of drug candidates, as they determine the safety and efficacy of a substance and since a medication can only take effect if it is correctly absorbed and transported within the body to the reactive endpoints.

2.1.2 Molecule Representation

In order to be able to use computational tools, molecules have to be represented in a simplified format, apt for digital encoding. There are three main types that are normally used for property prediction tasks, as described next.

The first is ***molecular fingerprints***. These are hashed representations, calculated on the presence or absence of specific substructures or features of the molecules, providing a descriptive vector [10]. An example is the Morgan fingerprint [11], calculated by applying the Morgan algorithm on a molecular graph, well suited for typical ML tools that use vector-formatted data. Table 1 shows some bits of such a morgan fingerprint for Aspirin, which are usually of length 2048. Each bit encodes the presence of some specific substructure in the molecule.

Bit	98513984	132611095	509662800	673156540	864662311	...
Value	2	1	1	1	1	...

Table 1: Some bits of a Morgan Fingerprint of Aspirin

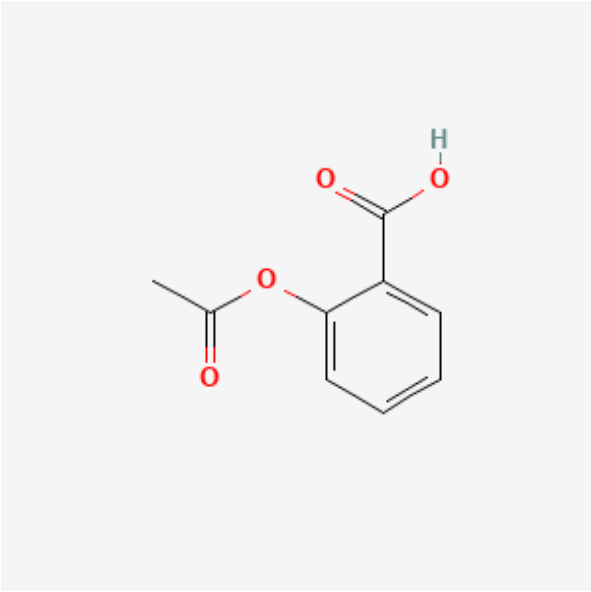


Figure 1: Molecular graph of Aspirin

The second approach is the so called Simplified Molecular Input Line Entry System (SMILES) [12], which consists of an ASCII string. Atoms are represented by their atomic symbols (e.g., C for carbon, O for oxygen) and bonds are implied by adjacency, or explicitly represented using symbols (e.g., = for double bonds). Branching and ring closures are indicated by special characters like parentheses and numbers. SMILES is widely used for database searches, structure storage, and input for chem-informatics tools. Its compact form makes it suitable for text processing and easy exchange between software tools. The SMILES string for Aspirin would be: CC(=O)OC1=CC=CC=C1C(=O)O, where C stands for Carbon atoms, O for Oxygen, = symbolizes a double bond and the 1 indicates the start and respective end of a ring structure. The hydrogen atoms in the molecule are typically left out and fill out the remaining required bonds in the molecule.

The third approach uses graphs to represent molecules, where atoms are nodes, and bonds are edges connecting these nodes [13]. In graph-based models, the molecular structure is visualized as a network. Nodes are labeled with atom types, and edges are labeled with bond types (e.g., single, double). This representation is highly flexible and can be used to capture complex structures and relationships within a molecule. For example, the molecular graph for Aspirin is shown in Fig. 1.

2.1.3 Molecular Complexity

Molecular complexity refers to the structural and chemical structure of a molecule, encompassing various aspects of its composition, number of atoms, connectivity and stereochemistry [14]. The concept of molecular complexity is crucial in fields such as drug discovery, as it often correlates with a molecule's biological activity, chemical reactivity, and other properties. It has been a field of study in computational chemistry, with a variety of proposals on how to quantify it. Since there is no clear single ideal measure of molecular complexity, this study used three different formulas. Atom bond count and fraction of sp 3 (Fsp3 hybridized carbons both were performed well in the study by Gu et al [8]), and the Spacial Score proposed by Krzyzanowski et al. [15], built on the basis of Fsp3, but further considering stereo chemical properties of the molecule.

The first method involves a straightforward structural assessment by summing up the number of atoms and bonds in a molecule. This metric assumes that molecules with fewer atoms and bonds are inherently simpler, as their smaller size limits the diversity and intricacy of possible interactions within the molecule.

The second approach focuses on the fraction of sp3 hybridized carbons within all carbon rings of the molecule. Sp3 hybridization is a concept in chemistry that describes the mixing of one s orbital and three p orbitals in a carbon atom to form four equivalent hybrid orbitals. This is indicative of the molecule's three-dimensional structure and flexibility, with a higher fraction suggesting greater structural complexity due to the geometry of sp3 carbons [16].

Lastly, the molecular spatial score, provides a more nuanced measure of complexity by evaluating the spatial distribution and arrangement of atoms within the molecule. This score accounts for the three-dimensional arrangement of atoms and the resulting steric effects, which can significantly influence molecular properties[15].

2.2 Deep Learning Techniques

2.2.1 Graph Neural Networks

This graph representation of molecules motivates the use of graph neural networks (GNNs) [17]. They are a type of deep learning (DL) model especially developed to work on networks. They usually consist of a step in which node information is propagated over edges, thereby maintaining the interdependence of features provided by the graph structure. This step is then usually followed by some kind of local pooling layer to condense local dependencies. Followed finally by a readout stage which consists of some global pooling layers to build the actual output.

2.2.2 Message Passing Neural Networks

MPNNs are a specific type of GNNs. Here, each node of the graph has a hidden state which gets updated during a message passing phase, as illustrated in Fig. 2. For each node, a message is calculated as a sum over every neighbor of the node, using a message function M. This message function can be

any non-linear differentiable function, but usually some multi-layer perceptron (MLP) with learnable parameters is used. Then the hidden state of the node is updated based on the current hidden state of the node and the message calculated from the neighbors. The update function introduces additional learnable parameters, typically another MLP, but any parameterized function could be used [18].

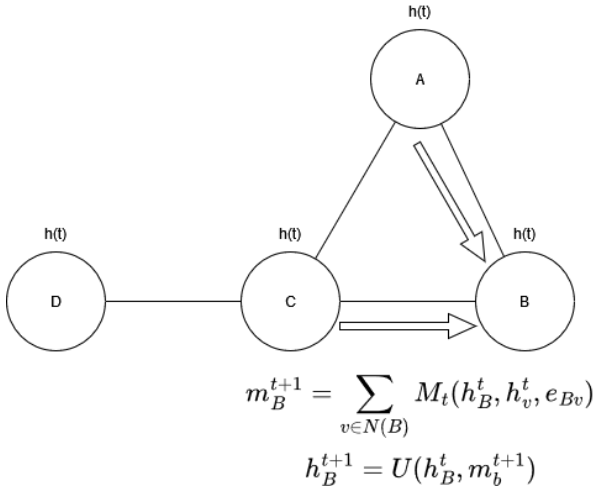


Figure 2: MPNN message passing phase

The message passing phase is followed by the readout phase, in which a readout function R is applied on the hidden states of the graph to produce a descriptive vector. Depending on the use case, this feeds a predictor model or other downstream tasks.

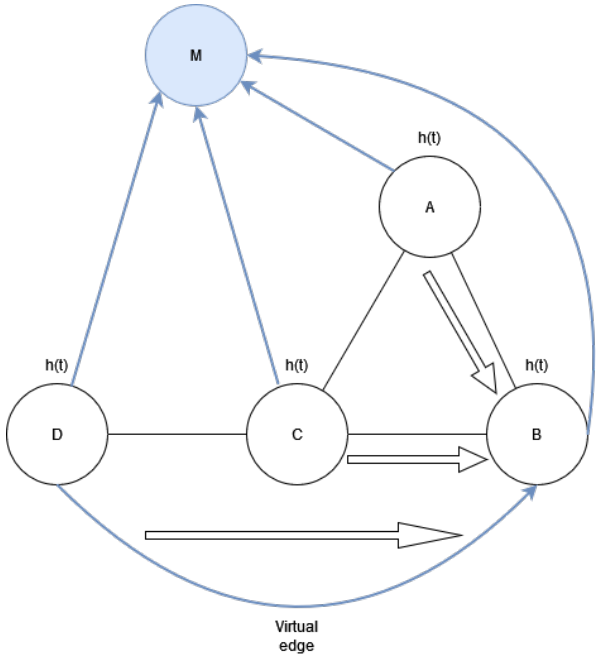


Figure 3: MPNN model design

Some MPNN implementations add virtual master nodes, as shown in Fig. 3, which have edges

connecting to every other node of the graph. This facilitates the learning of relationships between distant nodes. Since otherwise the effect between nodes gets diluted with every degree of separation.

2.2.3 Curriculum Learning

CL is an ML technique inspired by the way humans learn, where models are trained on tasks that gradually increase in difficulty, with the application to DL first formalized by Bengio et al [5]. It consists of ranking the difficulty of the training samples and structuring them accordingly during training. This approach can lead to faster convergence, better generalization, and improved model performance.

2.3 Gaussian Inference

The training of artificial neural networks usually introduces some randomness in two different stages. First is the initialization of weights when building the network. With no previous knowledge of good model weights, the common approach is assigning random ones to the nodes in the network [19]. Therefore starting the training process from a random position on the loss landscape. Even if all of the training samples were to be presented in the exact same order, two models with different starting weights will most likely end training with some different weights configuration and some difference in final performance. While in the academic environment seeds for random number generators are provided to be able to reproduce results, this is not realistic for real world applications, as previous knowledge would be required of which is a good seed.

Secondly, we must consider the sequence of training samples, from which the loss is calculated, and give the direction of the update steps. A different sequence of samples means a different sequence of update steps and therefore a different path through the loss landscape, meaning a different final weights configuration and model performance. So, again, training two identical models on the same data can lead to different final performances.

While a perfect convergence to the global minimum is theoretically possible, it is highly unlikely in real world conditions, with limits on computation time. Given these two sources of randomness, there is some inherent variance to the final performance of models [20]. This has to be accounted for when trying to evaluate new techniques, like CL in this case, since it could easily be that some difference in final performance is due to normal variance rather than a significant change in performance that is likely to be reproduced in practical applications.

The loss landscape in DL represents the surface defined by the loss function across the high-dimensional space of a neural network's parameters. Each point in this landscape corresponds to a specific set of model parameters, and the height of the landscape at any point indicates the loss, or

error, associated with those parameters. The primary goal of training a DL model is to navigate this landscape, typically using optimization algorithms like stochastic gradient descent (SGD) [21], to find the lowest possible point—where the loss is minimized, and the model makes the most accurate predictions.

Due to the complex nature of DL models, this landscape is often rugged, featuring numerous local minima, saddle points, and flat regions [22]. Despite this complexity, modern optimization methods are designed to effectively explore the landscape, finding parameter configurations that generalize well to unseen data. The shape and characteristics of the loss landscape play a crucial role in the training process, influencing how easily and effectively a model can learn from data.

3 Methods

The methods section of this thesis outlines the approach and techniques employed to investigate CL as a strategy for improving molecular property prediction, also detailing the data, model architecture and training parameters. Furthermore, it discusses the statistical methods applied to evaluate the effectiveness of CL in this application area.

3.1 Tools

The experiments were conducted using the Python programming language and a variety of libraries provided for it, including:

- **Chemprop** [23]

  Providing the implementation of molecular datasets, MPNN and training loop in pytorch.

- **rdkit** [24]

  A toolkit for cheminformatics that was used to construct molecular graph representations from SMILES and calculating the complexity scores.

- **pystan** [25]

  A Python API to the stan [26] package for statistical modelling.

- **numpy** [27]

  For array operations on the datasets.

- **pandas** [28]

  For reading csv files and formatting data in dataframes.

- **matplotlib** [29]

  To create the plots seen in this work.

3.2 Data

This thesis analyzes five different datasets of different ADME properties. Four of these, abbreviated as HLM, hPPB, MDR1 ER and Solubility, were collected and experimentally validated in a study by Fang et al [30]. Additionally, a fifth dataset (TDC Solubility), created by Sorkun et al [31], measuring the solubility of molecules in water, was used, as it contained a larger number of samples. The latter is publicly available from tdcommons.ai. Table 2 lists the number of samples included in the training and testing sets for each ADME property.

The HLM dataset measures human liver microsomal stability. This represents the rate at which a drug is metabolised in a human liver. The experimental setup to test the mircrosomal stability consists of incubating the compound with human liver enzymes and monitoring the change over time. Here, the stability was reported as intrinsic clearance, which is not limited by blood flow or other external

ADME endpoint	Training set	Test set
HLM	2469	618
hPPB	1446	362
MDR1 ER	2113	529
Solubility	1738	436
TDC Solubility	6988	1996

Table 2: The number of samples in each dataset

factors and is provided in milliliter per minute per kg of tissue. The values were transformed by the authors using the logarithmic function.

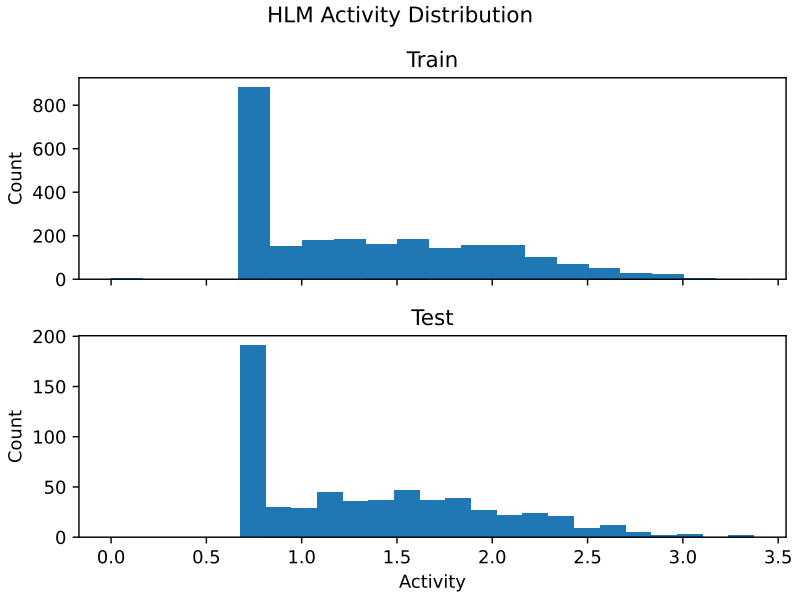


Figure 4: A Histogram of transformed human liver microsomal stability

Figures 4 - 8 show how the activity values are distributed in the training and testing sets, the activity mapped on the x-axis, and the number of molecules on the y-axis. The TDC Solubility dataset was split into 30 bins, because of the larger number of samples. All of the other figures used 20 bins.

As seen in Figure 4, the distribution of activity of molecules is very similar between the training and testing sets, with a significant peak, as many molecules shared the value 0.675687.

hPPB is a dataset on the binding behavior of the compounds to human plasma proteins. The dataset shows the percentage of molecules of a compound that remain unbound, since only the unbound fraction of a drug can cross cell membranes and interact with its target.

Figure 5 shows the distribution of activity values in the hPPB training and testing sets. Taking into

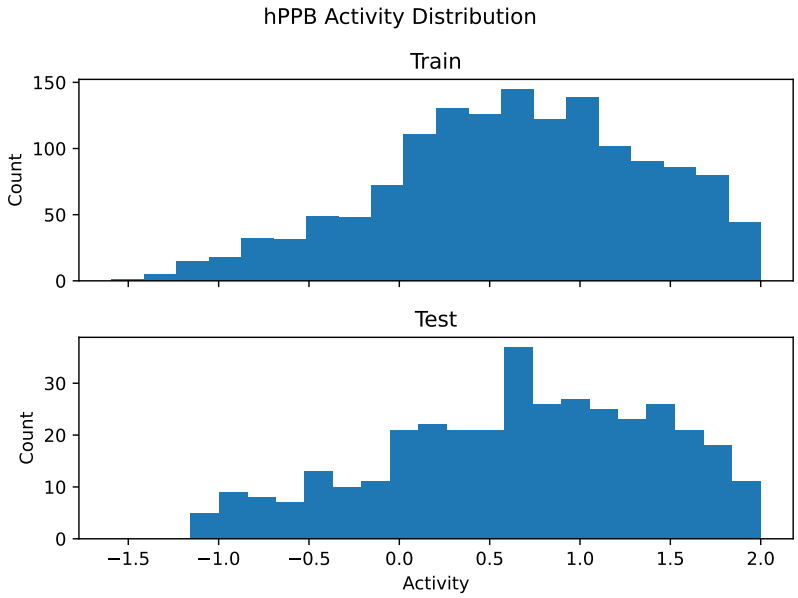


Figure 5: Distribution of activity of human plasma protein binding

account the logarithmic transform, a majority of samples have only 10% or fewer fraction of molecules unbound, which translates to an activity value of 1 in this plot. There are a few compounds that approach a very high ratio of unbound molecules, though. Both the testing and training sets show very similar distributions.

The MDR1 ER dataset studies the permeability of the compounds, experimentally measured as the efflux ratio (ER) in Madin-Darby Canine Kidney (MDCK) cells expressing the Multi-Drug resistance protein 1 (MDR1). The efflux ratio is calculated by comparing the permeability of the drug in two directions: from the apical side to the basolateral side (A-B) and from the basolateral side to the apical side (B-A) of the cell monolayer as

$$ER = \frac{\text{Permeability} B - A}{\text{Permeability} A - B} \tag{1}$$

Because of the logarithmic transform, an even efflux ratio of 1 corresponds to a value of 0 in Figure 6. Therefore, in this dataset more compounds have an ER below 1, meaning a higher permeability in the A-B direction. However, there are samples with much higher difference in permeability in the B-A direction, as can be seen by the fact that there are many samples with an activity value greater 0.5, but extremely few with an activity value below -0.5. This bias is equally present in both the training and testing sets.

The final ADME dataset, hereto further referred to as Solubility, collected by Fange et al.[30] used in this study, evaluates the solubility of the compounds in water with pH level at 6.8, reported as $\mu g/mL$, the values in the dataset have been converted with the logarithmic function.

Figure 7 shows the distribution of solubility of compounds registered in the Solubility dataset by Fang et al. The majority of molecules dissolve between 10 - 100 $\mu g/mL$, with both subsets showing

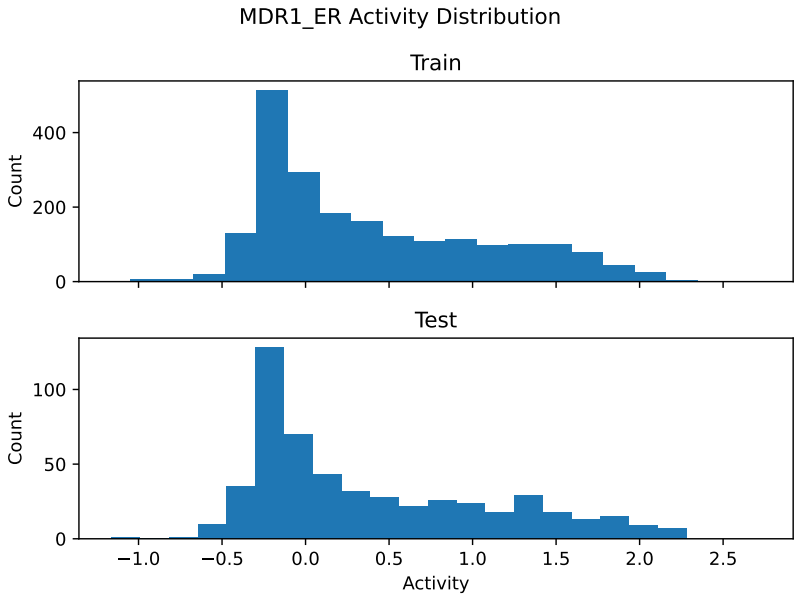


Figure 6: Histogram of MDR1 efflux ratio

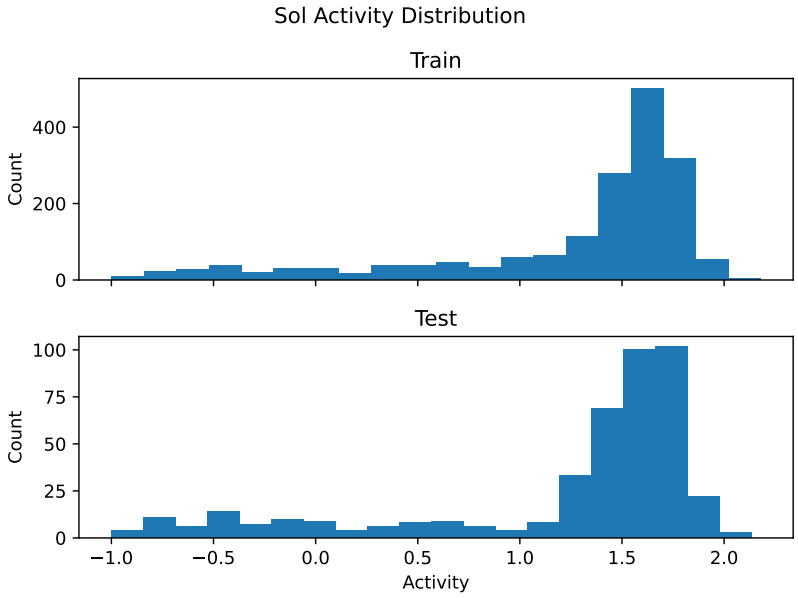


Figure 7: Solubility of compounds

a similar peek at around 1.7. But whether a compound is considered highly soluble is relative to the highest intended dosage [32].

In addition to the previously mentioned datasets, another dataset on molecule solubility in water was used, here called TDC Solubility, which contain a greater number of different compounds. The solubility in this case is provided in

$$log\left(\frac{\text{mol}}{L}\right) \tag{2}$$

and therefore not directly comparable to the previous solubility dataset. But since the goal of this study is to compare the effects of CL on models trained on the same dataset, the units were not converted.

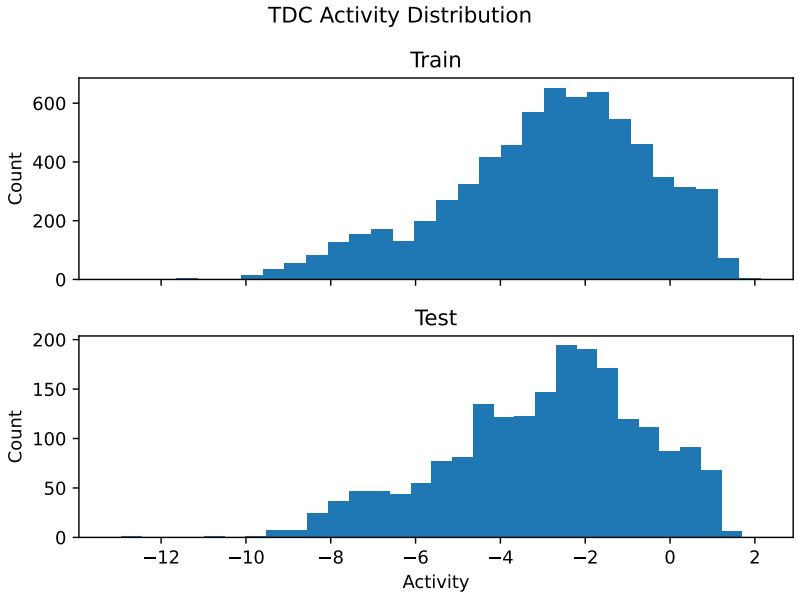


Figure 8: TDC solubility distribution

The compounds found in the TDC Solubility dataset, as seen in Figure 8, are more evenly distributed, with the highest density being at around -2, which translates to 0.01 mol/L. There does not appear to be a significant difference in the solubility of samples found in the test set or the training set.

In all of the datasets, the molecules were provided as SMILES. In the preprocessing of the dataset for training, molecular graphs were constructed from these SMILES. Additionally a random 10% of the training set was split off to be used as a validation set during the training process, thereby using a different validation set for each model trained.

3.3 Training Implementation

This section explains the technical implementation of the curriculum learning sampler as well as how the hyperparameters of the models were chosen for training.

3.3.1 Curriculum learning sampler

A classical CL approach was taken for the experiments in this thesis, which consists on classifying the training samples into different subsets based on perceived difficulty, then training on the different sets in a determined order.

In this context, the first challenge was determining the difficulty of a sample. To tackle the challenge, the concept of molecular complexity was used. A molecule with higher complexity is considered to be

a more difficult sample. Then, according to those complexity values, the dataset would be separated into even groups. During training, the batch sampler would only use samples from one group until it is depleted, before moving on to the next according to the predetermined order, as described in Algorithm 1. This way, the model would be presented with every sample once in each epoch, the same as in the control, with only the sequence changing in which they appear.

Algorithm 1 Curriculum Batch Sampler

```
Number of batches = round up(Number of samples / Batch size)
Sort Molecules by complexity
Split Samples into n subsets
for subset in subsets do
    shuffle subset
end for
for number of batches do
    for Batch size do
        if subset not empty then
            pop sample from subset
        else
            take next subset
            pop sample from subset
        end if
    end for
end for
```

3.3.2 Hyperparameter Tuning

To determine the hyperparameters of the model, a search was conducted with tools included in the *chemprop* library. It used the Tree-structured Parzen Estimators algorithm [33] to search the best values for the number of message passing steps on the graph previous to the readout, the number of layers in the fully connected neural network (FNN) used for the regression, the dropout rate in the MPNN and FNN layers, the dimension of the hidden representation produced by the MPNN and the number of neurons in each FNN layer. Further, some manual trials were performed to determine whether to use atom message passing or bond message passing, the number of epochs, and the activation function to use in the fully connected layers.

The final configuration of the model used in the experiments is the following:

- mean aggregation (as the local pooling in the MPNN)

- Depth = 2 (number of message passing steps)
- number of FNN layers = 2
- Droupout = 0.2
- hidden dimension = 100
- Atom message passing
- Epochs = 30
- activation function = leaky relu

3.4 Experimental Design

In order to explore the impact that a CL approach might have, different experiments were proposed. Firstly, different functions were utilized to assign a complexity score to the molecules, since there is no clear single indicator of a molecules complexity. Trials were conducted using each of the AtomBondCount, fraction of sp3 hybridized carbon atoms and the Spacialscore.
Further, the datasets were split into different numbers of difficulty subsets. A higher number of splits gives a more fine grained ordering during the training of samples, whereas a lower number of splits allows for a bigger difference in complexity between samples within a batch. In the reported experiments, either 2, 3 or 4 such subsets were made from the training set for the curriculum sampler. For the control, there was no splitting of the samples. Since the samples are randomly shuffled, creating subsets would not influence their order.

Finally, both sorting the samples in ascending complexity and descending complexity were tested, that is, from least complex molecules first and most complex last and vice-versa.

These three experimental degrees of freedom with 3, 3 and 2 possible values respectively provide 18 possible combinations, as seen in table 3, in addition to the control group, which were tried for each of the 5 datasets. For each trial, 30 models were trained to account for the randomness of initialization and batch sampling.

Thus, an overall total of $(18 + 1) \cdot 5 \cdot 30 = 570$ models were trained and then evaluated on the appropriate test set. The final root mean squared error is taken as the overall model performance and used for comparison.

3.5 Statistical analysis

To compare the results from the different experiments, a distribution underlying the 30 RMSEs of the models, from each experimental configuration, was determined. For this, the Pystan library was

Complexity function	Order	Splits
AtomBondCount	Ascending	2
AtomBondCount	Ascending	3
AtomBondCount	Ascending	4
AtomBondCount	Descending	2
AtomBondCount	Descending	3
AtomBondCount	Descending	4
Fraction of sp3 carbons	Ascending	2
Fraction of sp3 carbons	Ascending	3
Fraction of sp3 carbons	Ascending	4
Fraction of sp3 carbons	Descending	2
Fraction of sp3 carbons	Descending	3
Fraction of sp3 carbons	Descending	4
Spacialscore	Ascending	2
Spacialscore	Ascending	3
Spacialscore	Ascending	4
Spacialscore	Descending	2
Spacialscore	Descending	3
Spacialscore	Descending	4

Table 3: Experimental configurations

used to estimate the mean and standard deviation of a normal distribution using a Bayesian approach. The calculation process involves defining a probabilistic model that describes the likelihood of the observed data given these parameters. Combined with a prior distribution specifying a possible range of values for the mean and standard deviation, a posterior distribution is calculated which represents the updated belief of what values of the parameters are likely, given the observed data. Then Markov Chain Monte Carlo (MCMC) sampling was used to generate samples from the posterior distribution of possible means and standard deviations.

From the generated samples, a 95% confidence interval can be extracted of the mean and standard deviation to describe the expected outcome of training a model with the respective configuration.

4 Results

Tables 4 - 8 summarize the results from the experiments. The first column indicates the order in which the molecules were ordered, ascending or descending. The second column provides the number of subsets in which the data was split by the curriculum learning sampler. Then follow the complexity functions used, as described in the methods section. The final column shows the results from the control group, which were trained using a standard random sampler. For each configuration, the table lists the estimated mean ± the standard deviation of the normal distribution, which describe the normal distribution of potential performance of such a model on the testing set.

In order to better visualize some of the normal distributions described in the results tables, a diagram like Figure 9 shows a selection of normal distributions, usually from the combination of one complexity function and ordering, but varying the number of splits. The y-axis describes the probability density, while on the x-axis is the RMSE expected on the Testing set. Again the distribution from the control group is included as a reference.

Finally, some loss plots show the progression of some select models during training. In the title of each diagram is the complexity function mentioned, as well as, the RMSE achieved on the testing set by that particular model. All the models shown in the loss diagrams were trained with 3 splits in ascending order. Except the top left corner, which is the control, trained with a random sampler. The blue line represents the training loss, while the orange line indicates the validation loss. The training epoch is on the x-axis.

As mentioned in the description of statistical analysis, a 95% confidence interval was calculated for each of the values. These intervals can be found in the appendix A.2.

4.1 Results on the HLM Dataset

HLM Dataset					
Order	Splits	AtomBondCount	Fsp3	Spacialscore	control
Asc	2	0.415 ± 0.010	0.410 ± 0.018	0.404 ± 0.021	0.386 ± 0.011
	3	0.440 ± 0.018	0.410 ± 0.024	0.404 ± 0.015	
	4	0.446 ± 0.023	0.405 ± 0.028	0.413 ± 0.016	
Desc	2	0.409 ± 0.013	0.410 ± 0.019	0.410 ± 0.018	
	3	0.405 ± 0.019	0.434 ± 0.021	0.414 ± 0.020	
	4	0.413 ± 0.010	0.413 ± 0.015	0.430 ± 0.020	

Table 4: Means and standard deviations from HLM experiments

In the case of human liver microsomal stability, shown in 4, all of the models trained with a curriculum learning approach performed slightly worse than the control group. Each of the models trained with either the fraction of sp3 hybrid carbons or the spacialscore show very little difference in their mean and mostly overlap when taking their standard deviation into account. With some outliers being at 3 splits in descending order for fsp3 and 4 splits in descending order for spacialscore.

When looking at models trained with the number of atoms and bonds as the complexity function, there seems to be some increase in the expected RMSE when the samples are presented in ascending order with higher number of splits. However, there is no such continuous increase visible when the samples were presented in descending order.

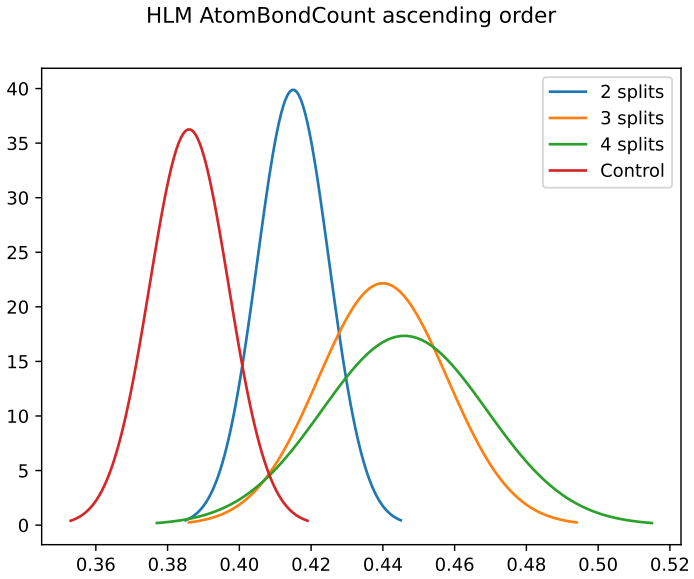


Figure 9: HLM model distributions with samples sorted by AtomBondCount in ascending order

Figure 9 shows the expected performances of models trained with the AtomBondCount function sorting in ascending order. The control condition is clearly the best, but, interestingly, training with 2 splits performs consistently better than training with 3 or 4 splits.

In each of the configurations shown in Figure 10, the validation loss increases in the first few epochs, after which it quickly declines to starting levels. In most cases, the validation loss seems to stagnate while the training loss continues to improve. The final RMSE on the testing set does not reflect the high level of the validation loss, but rather aligns more closely with the final training loss. This could possibly be due to the relatively small validation sets not properly reflecting the distribution of values seen in the training set.

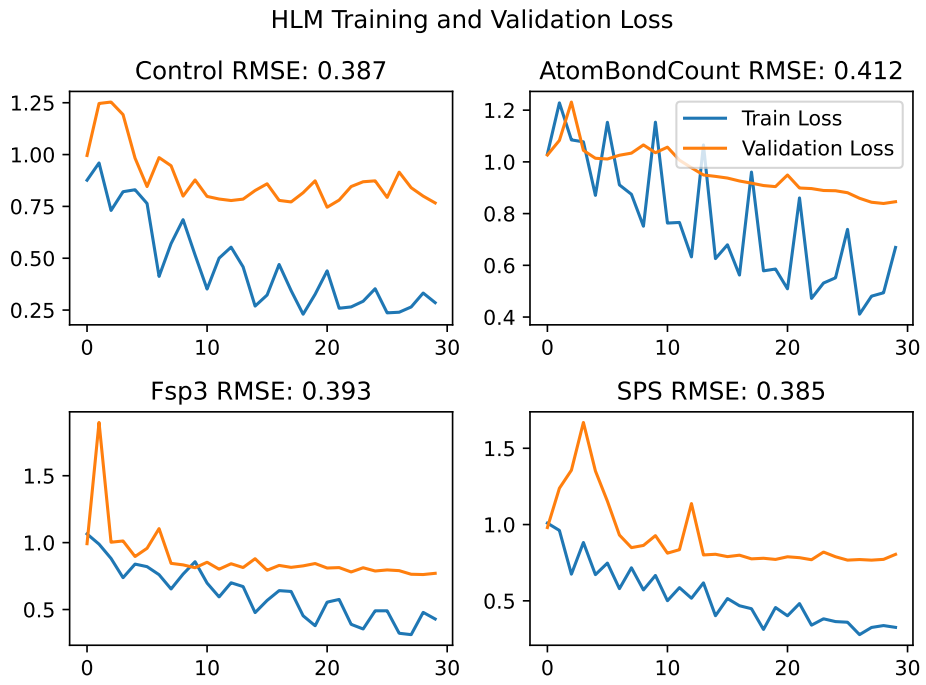


Figure 10: Loss during training

hPPB Dataset					
Order	Splits	AtomBondCount	Fsp3	Spacialscore	control
Asc	2	0.396 ± 0.013	0.399 ± 0.012	0.403 ± 0.030	0.392 ± 0.019
	3	0.400 ± 0.015	0.417 ± 0.044	0.405 ± 0.020	
	4	0.402 ± 0.011	0.413 ± 0.021	0.415 ± 0.022	
Desc	2	0.386 ± 0.015	0.412 ± 0.015	0.403 ± 0.018	
	3	0.389 ± 0.010	0.418 ± 0.018	0.426 ± 0.037	
	4	0.397 ± 0.018	0.410 ± 0.021	0.411 ± 0.027	

Table 5: Means and stds from hPPB experiments

4.2 Results on the hPPB Dataset

Reviewing the results from the models trained on human plasma protein binding data, it is noticeable that any model trained with the AtomBondCount function seemed to perform seemingly identical to the control. Some configurations even showed slightly lower means, though when considering the standard deviation, the overall distributions mostly overlap. Models trained with Fsp3 or Spacialscore tend to have a higher expected RMSE. However, comparing the different number of splits or the ordering of the samples, no great differences are to be seen.

As shown in Figure 11, despite the small differences in means and standard deviations in the plasma protein binding trials, the distributions of model performances still mostly overlap.

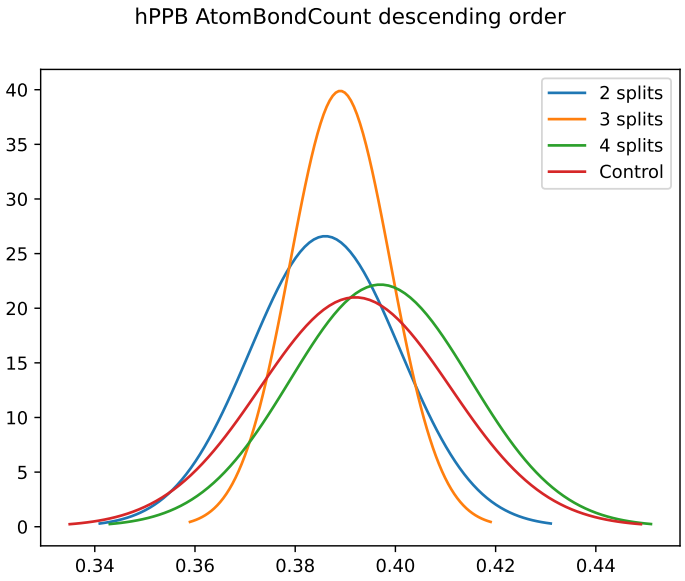


Figure 11: hPPB model distributions with 2 splits in descending order

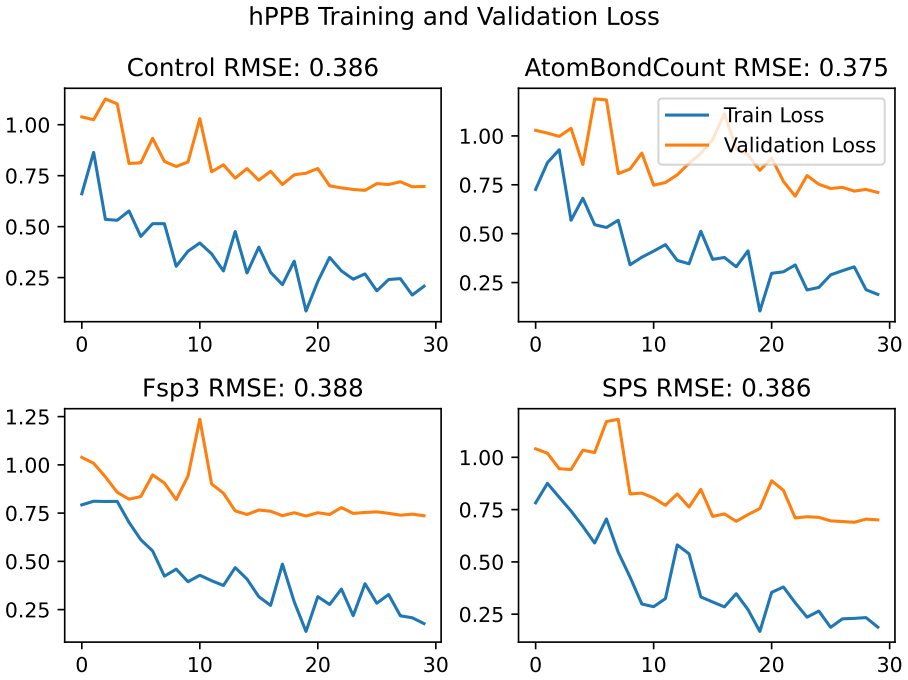


Figure 12: Loss during training

The loss progression visible in Figure 12 as well, shows a fairly big divergence between the training loss and the validation loss. While the training loss tends to improve, albeit with considerable fluctuations between epochs, the validation loss tends to stagnate at levels close to the starting value.

4.3 Results on the MDR1 ER Dataset

MDR1 ER Dataset					
Order	Splits	AtomBondCount	Fsp3	Spacialscore	control
Asc	2	0.407 ± 0.022	0.375 ± 0.043	0.364 ± 0.019	0.374 ± 0.038
	3	0.482 ± 0.060	0.382 ± 0.039	0.433 ± 0.060	
	4	0.552 ± 0.081	0.374 ± 0.022	0.440 ± 0.059	
Desc	2	0.413 ± 0.019	0.413 ± 0.032	0.417 ± 0.038	
	3	0.44 ± 0.020	0.535 ± 0.095	0.586 ± 0.128	
	4	0.445 ± 0.016	0.447 ± 0.048	0.588 ± 0.185	

Table 6: Means and standard deviations from MDR1 ER experiments

Predicting the permeability of molecules measured in the MDCK-MRD1 ER assays, there seems to be overall a higher standard deviation in many trials, compared to the other datasets. Making a clear distinction between the performance of different configurations more difficult. Training with samples in an ascending order of their fraction of sp3 hybridized carbons, there are barely any difference in either the mean or the standard deviation of the estimated distributions. When reversing the order, the expected RMSE of the models increases by a significant margin. A similar trend is observable when using the spacial score to sort the samples. With both the mean and standard deviation growing to the biggest values of any models trained on this dataset. In the case of sorting by the size of the molecules this seems to be reversed. Models performed worse when presented with an ascending order, the RMSE growing with the number of splits. Whereas the descending sequence seemed to not change much by the number of splits.

Figures 13 and 14 show the outlier distributions from the MDR1 ER trials and visualize the impact that the greater standard deviation has on the reliability of the training results.

The training loss of some of the models trained on the MDR1-ER data, visible in Figure 15 experiences some very high jumps between each epoch, making it difficult to recognize a clear trend. In the control experiment the validation loss is fairly unstable over the first 10 epochs, but then approaches 0.9, while the trainig loss improves relatively consistently.

4.4 Results on the Solubility Dataset

In the case of the smaller solubility dataset, there seem to be, overall, very little differences between control and virtually any of the other CL configurations. The most notable difference is when comparing to training with FSP3 function and splitting the training set into four distinct subsets, as visible in 16. Here the models tended to perform slightly worse, with a slightly smaller standard deviation relative to the control, though still a fairly large overlap with the standard deviation also being slightly greater. Greater differences are visible in the standard deviation of the models, with fsp3 and spacial

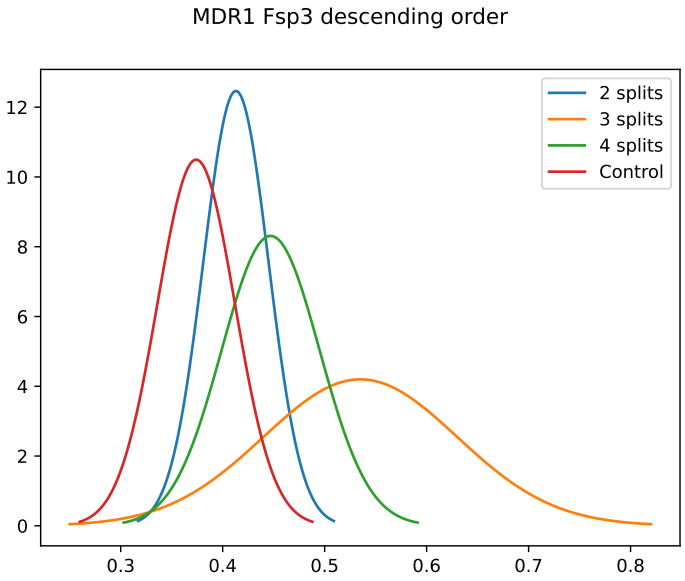


Figure 13: MDR1 ER model distributions with 2 splits in ascending order

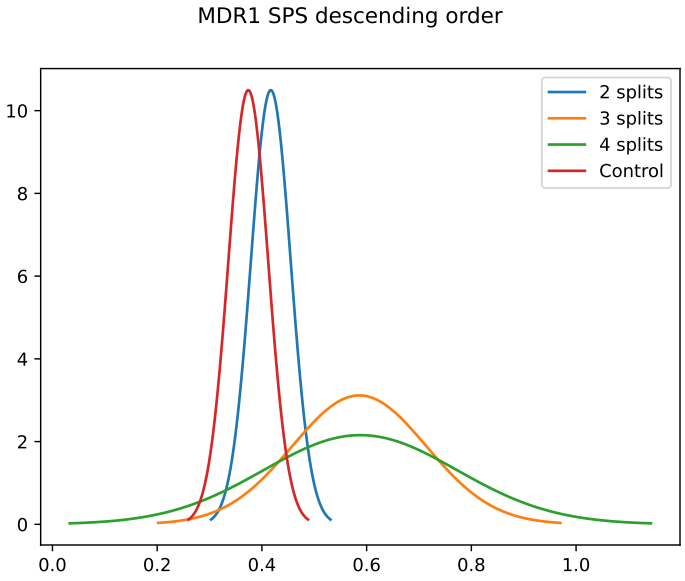


Figure 14: MDR1 ER model distributions with 2 splits in ascending order

score leading to more diverse results.

Across all of the models shown in Figure 17, the training loss fluctuates significantly while seemingly overall improving slightly. The validation loss in comparison is much more stable, but again does not seem to change much during training.

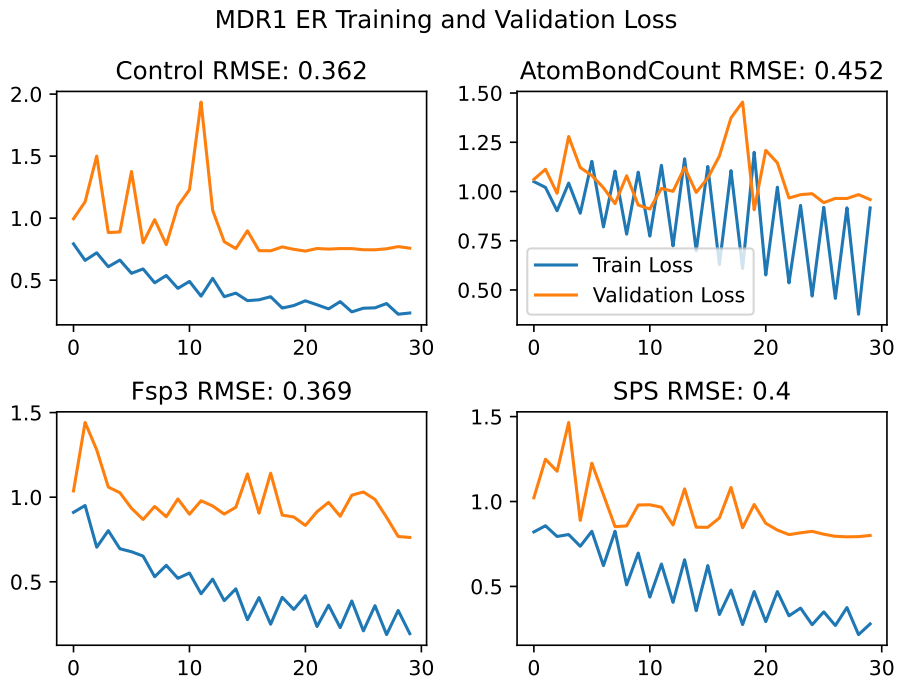


Figure 15: Loss during training

Solubility Dataset					
Order	Splits	AtomBondCount	Fsp3	Spacialscore	control
Asc	2	0.416 ± 0.015	0.403 ± 0.016	0.396 ± 0.016	0.399 ± 0.016
	3	0.419 ± 0.019	0.408 ± 0.022	0.402 ± 0.028	
	4	0.417 ± 0.011	0.427 ± 0.023	0.410 ± 0.023	
Desc	2	0.406 ± 0.010	0.405 ± 0.010	0.394 ± 0.014	
	3	0.402 ± 0.011	0.418 ± 0.018	0.416 ± 0.021	
	4	0.411 ± 0.013	0.433 ± 0.020	0.407 ± 0.014	

Table 7: Means and stds from Solubility experiments

4.5 Results on the TDC Solubility Dataset

The TDC solubility dataset was by far the biggest dataset, with around three times as many samples than the others. Again, sorting by the fraction of sp3 hybridized carbons had very little impact on the final evaluation, with most of the configurations overlapping noticeably with the control group. Here, only the combination of training with an ascending order and splitting into four subsets is an outlier. When sorting according to spacialscore values the models tended to perform worse, increasing in achieved test RMSE with a higher number of splits, though, whether they were presented in ascending or descending order had no clear effect. Finally, sorting according to the size of the molecules did have a noticeable negative impact on their performance on the test set, especially in the case of starting with smaller molecules and increasing in size during training, visible in Figure 18. There is no real

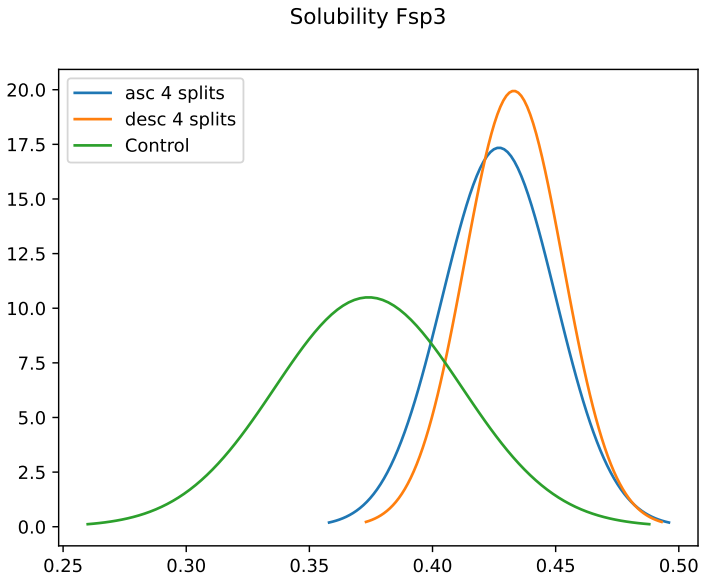


Figure 16: Solubility model distributions with 4 splits in ascending order

TDC Solubility Dataset					
Order	Splits	AtomBondCount	Fsp3	Spacialscore	Control
Asc	2	1.062 ± 0.059	0.788 ± 0.035	0.896 ± 0.038	0.782 ± 0.029
	3	1.18 ± 0.070	0.806 ± 0.020	0.976 ± 0.044	
	4	1.236 ± 0.055	0.847 ± 0.035	0.999 ± 0.049	
Desc	2	0.951 ± 0.068	0.793 ± 0.031	0.826 ± 0.033	
	3	0.989 ± 0.076	0.815 ± 0.036	0.933 ± 0.074	
	4	1.16 ± 0.064	0.811 ± 0.029	0.937 ± 0.052	

Table 8: Means and standard deviations from TDC Solubility experiments

overlap between the models trained with the curriculum sampler and the control group. Additionally, there is a fairly clear trend that increasing the number of splits, thereby making the ordering more granular, makes the models perform worse on the testing set.

It is noticeable in Figure 19 that the RMSE of the models achieved on the testing set are all higher than the validation loss during training. Interestingly, both the training and the validation loss are a lot bigger in the case of using AtomBondCount as the complexity function. In each other case the training loss reached about 0.2, with the validation loss stagnating around 0.6. Even though the final RMSE on the testing set show greater differences.

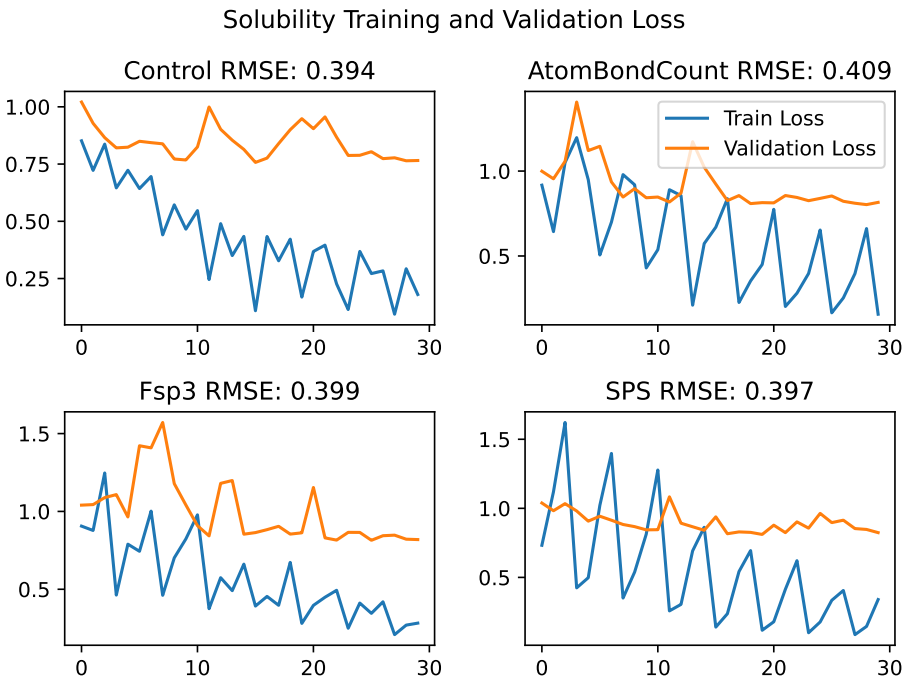


Figure 17: Loss during training

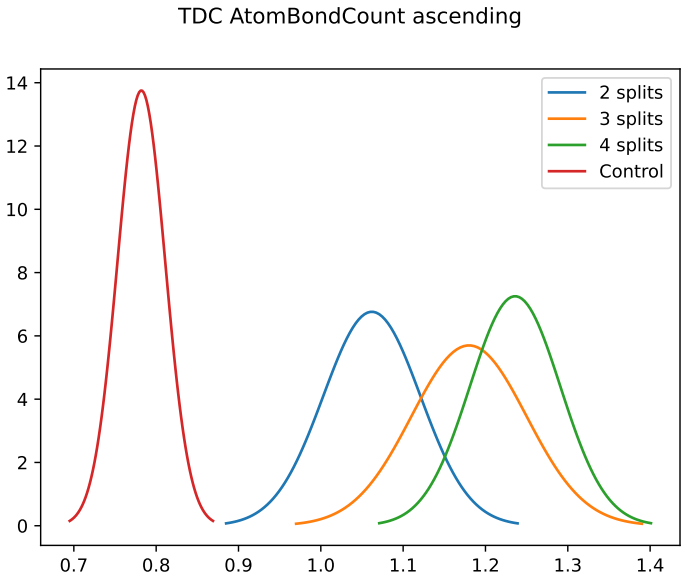


Figure 18: TDC model distributions with 4 splits in descending order

4.6 Pearson Correlation Coefficient

A Pearson correlation coefficient was calculated for each combination of complexity function and data set, to review potential correlations between the activity of molecules on their ADME endpoints and their respective complexity values, shown in Table 9.

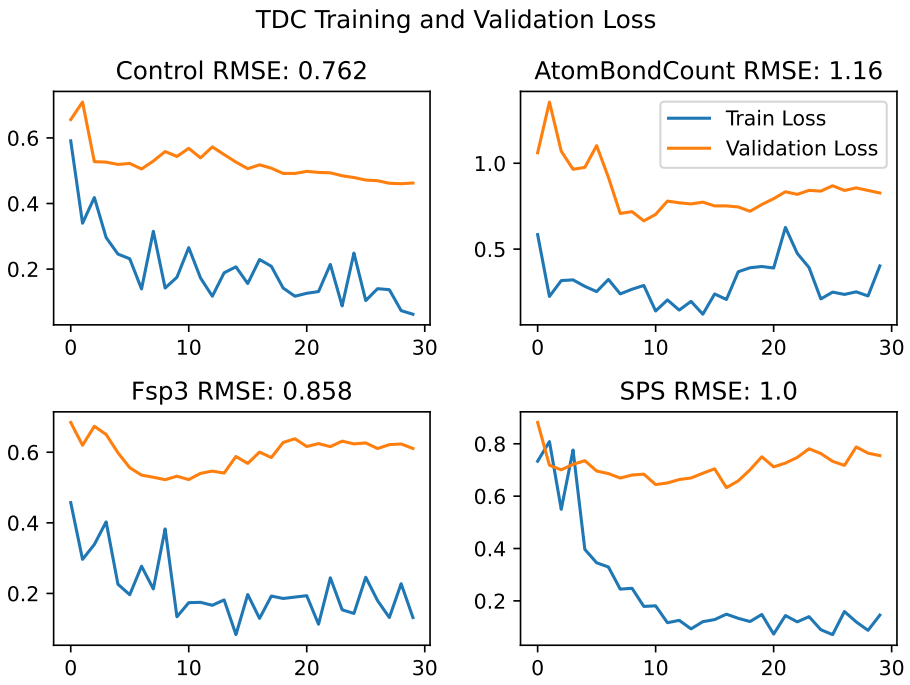


Figure 19: Loss during training

Correlation between activity and complexity			
	AtomBondCount	Fsp3	SpacialScore
HLM	0.323	insignificant	insignificant
hPPB	-0.121	0.257	0.223
MDR1 ER	0.479	0.190	0.216
Solubility	0.235	0.250	0.177
TDC	-0.373	0.035	-0.036

Table 9: Pearson correlation coefficients

5 Discussion and Conclusions

The findings of this thesis should provide some insights into the potential application of CL for molecular property prediction. This discussion will critically analyze the results and address the potential limitations of this technique, providing some directions of potential future research.

5.1 Interpretation of results

First of all, most of the experimental set-ups performed worse than the corresponding controls. Although, depending on the dataset, some significant differences can be observed between the trials. While the models trained on the HLM data, for example, performed fairly consistently across all split numbers and ordering, there are more noticeable differences in the distributions of models trained on MDR1 ER data, where the ascending ordering of molecular complexity achieved significantly higher

RMSE than the descending ordering. This aligns with the fact that the Pearson's correlation coefficient between the MDR1 ER activity and the AtomBondCount is significantly larger compared to the other complexity functions used in this study.

A similar trend is observable in the trials with the therapeutics commons dataset on Solubility. The strongest correlation here is again between the AtomBondCount and the aqueous solubility of the molecules. Again we observe a stronger difference in the distribution of model performances using this complexity function to the control trial, with a descending ordering of molecule complexity outperforming an ascending ordering in every number of splits. Further increasing the number of splits, and therefore the granularity of complexity of the samples presented to the models during training, made the models perform overall worse on the testing set.

Seeing as there are some quite significant differences between model performance depending on the configuration, the sequence in which samples are presented during training has a significant impact on the final performance of the model. The correlation between the value by which they are ordered and the value to be predicted might give some insight into the size of the impact this sorting has. Given that the cases in which there is a noticeable difference between the control and the curriculum configuration, is always a negative one, grouping the samples by complexity/activity could make the training process more vulnerable to getting stuck in some local minima, which are more likely to be avoided when batching more heterogeneous molecules, specially seeing as many of the datasets are not very evenly distributed, but rather have some small set of all possible values highly over represented.

Because of how opaque DL models are, it is not trivial to understand why the models tended to perform worse when trained with the CL sampler. One possible explanation is over-fitting. This implies that the model is learning dynamics in the training data that do not generalize correctly to testing data. One indication of this happening is when the validation loss plateaus or declines, while the training loss continues to improve.

Figure 20 shows the prediction error of a model trained with a normal random sampler and the error of a model trained with the curriculum sampler. The x-axis showing the complexity value was calculated according to the AtomBondCount function. On the y-axis is the prediction error of the models. Each point represents an individual samples from the testing set. While these are only single exemplary models, and it has been shown that their performance can vary significantly, there is an interesting trend in that the CL model seems to perform worse on smaller molecules.

5.2 Assumptions, limitations and future work

There are several assumptions being made in this thesis. One such assumption concerns the accuracy of the experimentally determined ADME endpoint activity of the data. There can be a significant

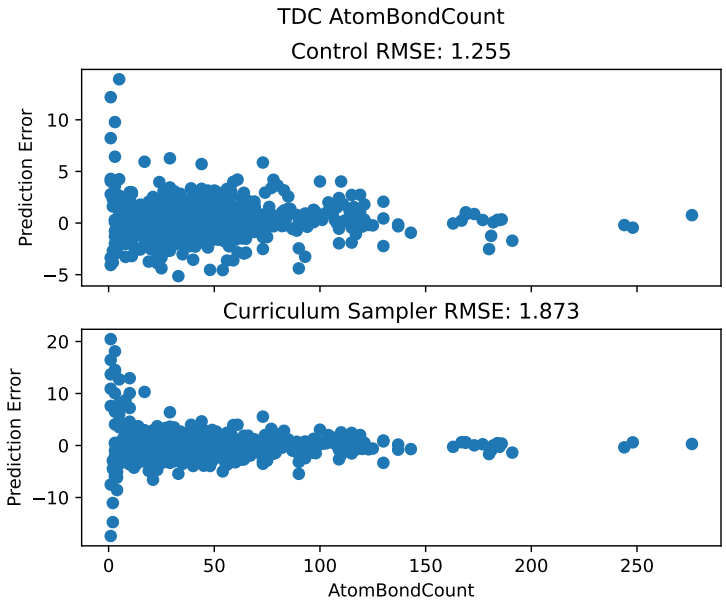


Figure 20: Prediction error of TDC models

variability in the measured behavior of molecules [34], which in turn impacts the prediction model and puts a limit on the achievable accuracy.

Another assumption is made on the existence of a good descriptor function, that predicts the activity of a molecule based on its structure. While neural networks have been shown to be universal approximators [35], the reality is that the more complex the function underlying the physical phenomenon is, the harder it is for the model to properly learn. The existence of activity cliffs in medical compounds [36] implies that the function that maps from the molecular structure to the activity is not perfectly smooth, increasing the likelihood for models to have high prediction errors on some individual compounds.

Finally, using molecular graph representations for the model assumes that the 2-dimensional structure of a molecule is sufficient to predict the behavior. Furthermore, it is a requirement of graph neural networks to be invariant to graph isomorphism. This conflicts with the concept of chirality in chemistry, in which molecules with an identical graph representation can take different 3-dimensional shapes, altering their behavior [37].

According to the results, there is no single trend throughout every data set that would give a good direction of what to try when confronted with a new problem. Rather, the specifics of how the different complexity functions influenced the training are unique to each dataset. This means that a function that might perform well on one dataset, might actually be counterproductive on another one.

Given the number of repetitions that have to be performed to be able to more clearly differentiate the effect, it might be wholly unfeasible in a production environment to first explore a variety of sorting functions, as they tend to be restricted in computation time.

One potential direction for further research is the exploration of the connection between some predictor value, like the correlation between complexity and activity, that gives an indication of whether a complexity function might perform well on a given dataset or not. This would allow to first evaluate a variety of complexity functions without having to invest the computational time to actually train models.

Another approach could be to construct some algorithm that is wholly independent of the dataset. By presenting the samples in order of the prediction error of model during the previous training epoch, for example. If such an approach could be shown to positively impact the learning of a model, or how it generalizes to unseen data, it could be applied without the consideration of what specific property is being predicted.

5.3 Ethical Social Impact, Sustainability and Diversity

While this work is mostly theoretical, the field of application raises some considerations on the quality of data and its extrapolation in practical use.

A predictor model is only able to generalize from structures found in the training data. In this case the behavior of potential drug candidates in *in vitro* experiments on relevant ADME benchmarks. If such a predictor model became good enough to be used to select promising compounds to focus on in further trials, any biases present in the training data could directly impact what kind of drugs might receive the limited resources of research departments.

There has been research on drugs acting differently in patients based on gender or race [38] [39], suggesting that one compound that has acceptable toxicity in one population, might be dangerous to another.

Taking the human plasma protein binding data as an example, the values were collected in a controlled experimental setting. These values in turn trained a model, that might predict a compound to be a potential candidate. But research suggest there might be a difference in the concentration of drug-binding proteins based on gender [40]. If this variance is not correctly accounted for in the data, models trained on it would be gender-biased. Which might lead to the drug having to be rejected during clinical trials, because of the bio-availability of the compound not lining up as expected.

5.4 Conclusion

Seeing as there has been not a single model that consistently performed better when trained with the curriculum sampler, the hypothesis that CL improves on the overall accuracy of models, can not be confirmed.

However, it has been shown that the order in which samples are presented in training can impact the models ability to generalizes to unseen examples. The questions of how to best sort the

data to benefit the learning remains open. Out of the three complexity functions used in this study, AtomBondCount showed the biggest correlation on some properties as well as influencing the most during training. But as the research into molecular complexity continues, it might provide better measures. So while CL has shown some potential, it has more so proven its risk. If not applied correctly, it can further exasperate issues such as model overfitting, thereby actively damaging the desired results.

Finally, it has been made clear that even with identical training parameters, there are random factors that can influence the success of a model. Which raises the relevance of repeating experiments and using some statistical validation, when studying new deep learning techniques. Otherwise it is not trivial to be certain whether there has actually been a significant effect, or result of some unknown variance.

References

[1] Izumi V Hinkson, Benjamin Madej, and Eric A Stahlberg. "Accelerating therapeutics for opportunities in medicine: a paradigm shift in drug discovery". In: *Frontiers in pharmacology* 11 (2020), p. 770.

[2] Tohru Takebe, Ryoka Imai, and Shunsuke Ono. "The current status of drug discovery and development as originated in United States academia: the influence of industrial and academic collaboration on drug discovery and development". In: *Clinical and translational science* 11.6 (2018), pp. 597–606.

[3] Connor W. Coley. "Defining and Exploring Chemical Spaces". In: *Trends in Chemistry* 3.2 (2021). Special Issue: Machine Learning for Molecules and Materials, pp. 133–145. ISSN: 2589-5974. DOI: https://doi.org/10.1016/j.trechm.2020.11.004. URL: https://www.sciencedirect.com/science/article/pii/S2589597420302884.

[4] Hongming Chen, Thierry Kogej, and Ola Engkvist. "Cheminformatics in Drug Discovery, an Industrial Perspective". In: *Molecular Informatics* 37.9-10 (2018), p. 1800041. DOI: https://doi.org/10.1002/minf.201800041. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/minf.201800041. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/minf.201800041.

[5] Yoshua Bengio et al. "Curriculum learning". In: *International Conference on Machine Learning*. 2009. URL: https://api.semanticscholar.org/CorpusID:873046.

[6] Petru Soviany et al. "Curriculum Learning: A Survey". In: *CoRR* abs/2101.10382 (2021). arXiv: 2101.10382. URL: https://arxiv.org/abs/2101.10382.

[7] Kevin Yang et al. "Analyzing Learned Molecular Representations for Property Prediction". In: *Journal of Chemical Information and Modeling* 59.8 (2019). PMID: 31361484, pp. 3370–3388. DOI: 10.1021/acs.jcim.9b00237. eprint: https://doi.org/10.1021/acs.jcim.9b00237. URL: https://doi.org/10.1021/acs.jcim.9b00237.

[8] Yaowen Gu, Si Zheng, and Jiao Li. "CurrMG: A Curriculum Learning Approach for Graph Based Molecular Property Prediction". In: *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2021, pp. 2686–2693. DOI: 10.1109/BIBM52615.2021.9669478.

[9] Sonu S Singh. "Preclinical pharmacokinetics: an approach towards safer and efficacious drugs". In: *Current drug metabolism* 7.2 (2006), pp. 165–182.

[10] Alice Capecchi, Daniel Probst, and Jean-Louis Reymond. "One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome". In: *Journal of cheminformatics* 12 (2020), pp. 1–15.

[11] Shifa Zhong and Xiaohong Guan. "Count-Based Morgan Fingerprint: A More Efficient and Interpretable Molecular Representation in Developing Machine Learning-Based Predictive Regression Models for Water Contaminants' Activities and Properties". In: *Environmental Science & Technology* 57.46 (2023). PMID: 37406199, pp. 18193–18202. DOI: 10.1021/acs.est.3c02198. eprint: https://doi.org/10.1021/acs.est.3c02198. URL: https://doi.org/10.1021/acs.est.3c02198.

[12] David Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". In: *Journal of Chemical Information and Computer Sciences* 28.1 (1988), pp. 31–36. DOI: 10.1021/ci00057a005. eprint: https://doi.org/10.1021/ci00057a005. URL: https://doi.org/10.1021/ci00057a005.

[13] "molecular graph". In: (2019). DOI: doi:10.1351/goldbook.MT07069. URL: https://doi.org/10.1351/goldbook.MT07069.

[14] Oscar Méndez-Lucio and José L. Medina-Franco. "The many roles of molecular complexity in drug discovery". In: *Drug Discovery Today* 22.1 (2017), pp. 120–126. ISSN: 1359-6446. DOI: https://doi.org/10.1016/j.drudis.2016.08.009. URL: https://www.sciencedirect.com/science/article/pii/S1359644616302975.

[15] Adrian Krzyzanowski et al. "Spacial Score-A Comprehensive Topological Indicator for Small-Molecule Complexity". In: *Journal of Medicinal Chemistry* 66.18 (2023). PMID: 37651653, pp. 12739–12750. DOI: 10.1021/acs.jmedchem.3c00689. eprint: https://doi.org/10.1021/acs.jmedchem.3c00689. URL: https://doi.org/10.1021/acs.jmedchem.3c00689.

[16] Frank Lovering, Jack Bikker, and Christine Humblet. "Escape from Flatland: Increasing Saturation as an Approach to Improving Clinical Success". In: *Journal of Medicinal Chemistry* 52.21 (2009). PMID: 19827778, pp. 6752–6756. DOI: 10.1021/jm901241e. eprint: https://doi.org/10.1021/jm901241e. URL: https://doi.org/10.1021/jm901241e.

[17] Gabriele Corso et al. "Graph neural networks". In: *Nature Reviews Methods Primers* 4.1 (Mar. 2024), p. 17. ISSN: 2662-8449. DOI: 10.1038/s43586-024-00294-7. URL: https://doi.org/10.1038/s43586-024-00294-7.

[18] Justin Gilmer et al. *Neural Message Passing for Quantum Chemistry*. 2017. arXiv: 1704.01212 [cs.LG]. URL: https://arxiv.org/abs/1704.01212.

[19] Meenal V. Narkhede, Prashant P. Bartakke, and Mukul S. Sutaone. "A review on weight initialization strategies for neural networks". In: *Artificial Intelligence Review* 55.1 (Jan. 2022), pp. 291–322. ISSN: 1573-7462. DOI: 10.1007/s10462-021-10033-z. URL: https://doi.org/10.1007/s10462-021-10033-z.

[20] Boyuan Chen et al. "Towards training reproducible deep learning models". In: *Proceedings of the 44th International Conference on Software Engineering*. 2022, pp. 2202–2214.

[21] Shun-ichi Amari. "Backpropagation and stochastic gradient descent method". In: *Neurocomputing* 5.4-5 (1993), pp. 185–196.

[22] Lei Wu, Zhanxing Zhu, et al. "Towards understanding generalization of deep learning: Perspective of loss landscapes". In: *arXiv preprint arXiv:1706.10239* (2017).

[23] Esther Heid et al. "Chemprop: A Machine Learning Package for Chemical Property Prediction". In: *Journal of Chemical Information and Modeling* 64.1 (2024). PMID: 38147829, pp. 9–17. DOI: 10.1021/acs.jcim.3c01250. eprint: https://doi.org/10.1021/acs.jcim.3c01250. URL: https://doi.org/10.1021/acs.jcim.3c01250.

[24] Greg Landrum et al. *rdkit/rdkit: 2024_03_5 (Q1 2024) Release*. Version Release_2024_03_5. July 2024. DOI: 10.5281/zenodo.12782092. URL: https://doi.org/10.5281/zenodo.12782092.

[25] Allen Riddell, Ari Hartikainen, and Matthew Carter. *pystan (3.0.0)*. PyPI. Mar. 2021.

[26] Stan Development Team. *Stan Modeling Language Users Guide and Reference Manual, 2.35*. https://mc-stan.org. Mar. 2024.

[27] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[28] The pandas development team. *pandas-dev/pandas: Pandas*. Version v2.2.2. Apr. 2024. DOI: 10.5281/zenodo.10957263. URL: https://doi.org/10.5281/zenodo.10957263.

[29] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[30] Cheng Fang et al. "Prospective validation of machine learning algorithms for absorption, distribution, metabolism, and excretion prediction: An industrial perspective". In: *Journal of Chemical Information and Modeling* 63.11 (2023), pp. 3263–3274.

[31] Murat Cihan Sorkun, Abhishek Khetan, and Süleyman Er. "AqSolDB, a curated reference set of aqueous solubility and 2D descriptors for a diverse set of compounds". In: *Scientific data* 6.1 (2019), p. 143.

[32] Pablo A. M. Quiroga. "Biopharmaceutics Classification System". In: *The ADME Encyclopedia: A Comprehensive Guide on Biopharmacy and Pharmacokinetics*. Cham: Springer International Publishing, 2021, pp. 1–7. ISBN: 978-3-030-51519-5. DOI: 10.1007/978-3-030-51519-5_137-1. URL: https://doi.org/10.1007/978-3-030-51519-5_137-1.

[33] Shuhei Watanabe. *Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance*. 2023. arXiv: 2304.11127 [cs.LG]. URL: https://arxiv.org/abs/2304.11127.

[34] Mark C. Wenlock and Lars A. Carlsson. "How Experimental Errors Influence Drug Metabolism and Pharmacokinetic QSAR/QSPR Models". In: *Journal of Chemical Information and Modeling* 55.1 (2015). PMID: 25406036, pp. 125–134. DOI: 10.1021/ci500535s. eprint: https://doi.org/10.1021/ci500535s. URL: https://doi.org/10.1021/ci500535s.

[35] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[36] Dagmar Stumpfe, Huabin Hu, and Jürgen Bajorath. "Evolving Concept of Activity Cliffs". In: *ACS Omega* 4.11 (2019). PMID: 31528788, pp. 14360–14368. DOI: 10.1021/acsomega.9b02221. eprint: https://doi.org/10.1021/acsomega.9b02221. URL: https://doi.org/10.1021/acsomega.9b02221.

[37] Guo-Qiang Lin, Jian-Ge Zhang, and Jie-Fei Cheng. "Overview of Chirality and Chiral Drugs". In: *Chiral Drugs*. John Wiley & Sons, Ltd, 2011. Chap. 1, pp. 3–28. ISBN: 9781118075647. DOI: https://doi.org/10.1002/9781118075647.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118075647.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118075647.ch1.

[38] Franck Mauvais-Jarvis et al. "Sex- and gender-based pharmacological response to drugs". en. In: *Pharmacol. Rev.* 73.2 (Apr. 2021), pp. 730–762.

[39] A Ramamoorthy et al. "Racial/ethnic differences in drug disposition and response: review of recently approved drugs". en. In: *Clin. Pharmacol. Ther.* 97.3 (Mar. 2015), pp. 263–273.

[40] David E Amacher. "Female gender as a susceptibility factor for drug-induced liver injury". In: *Human & experimental toxicology* 33.9 (2014), pp. 928–939.

A Appendix

A.1 Code Availability

The code used in for the experiments is available at https://gitlab.com/JLagemann/curra-tfm

A.2 Results with confidence intervals

HLM

SPS complexity

order	n_splits	mean	std
asc	2	0.404 ± 0.00054	0.020 ± 0.00042
	3	0.404 ± 0.00044	0.015 ± 0.00027
	4	0.413 ± 0.00040	0.017 ± 0.00032
desc	2	0.411 ± 0.00046	0.018 ± 0.00037
	3	0.419 ± 0.00073	0.027 ± 0.00052
	4	0.430 ± 0.00056	0.020 ± 0.00042
control		0.386 ± 0.00025	0.011 ± 0.00022

HLM

Atom Bond count complexity

order	n_splits	mean	std
asc	2	0.415 ± 0.00029	0.010 ± 0.00021
	3	0.440 ± 0.00049	0.018 ± 0.00030
	4	0.446 ± 0.00053	0.022 ± 0.00044
desc	2	0.409 ± 0.00033	0.013 ± 0.00024
	3	0.401 ± 0.00026	0.010 ± 0.0002
	4	0.413 ± 0.00024	0.010 ± 0.00018
control		0.386 ± 0.00025	0.011 ± 0.00022

HLM

fsp3 complexity

order	n_splits	mean	std
asc	2	0.410 ± 0.00044	0.018 ± 0.00032
	3	0.410 ± 0.00054	0.025 ± 0.00045
	4	0.406 ± 0.00068	0.028 ± 0.00059
desc	2	0.410 ± 0.00050	0.019 ± 0.00038
	3	0.440 ± 0.00047	0.021 ± 0.00035
	4	0.413 ± 0.00040	0.0151 ± 0.00029
control		0.386 ± 0.00025	0.011 ± 0.00022

hPPB

SPS complexity

order	n_splits	mean	std
asc	2	0.403 ± 0.00081	0.030 ± 0.00062
	3	0.405 ± 0.00048	0.019 ± 0.00035
	4	0.415 ± 0.00062	0.022 ± 0.00045
desc	2	0.403 ± 0.00045	0.018 ± 0.00037
	3	0.423 ± 0.00076	0.025 ± 0.00045
	4	0.410 ± 0.00081	0.027 ± 0.00055
control		0.392 ± 0.00052	0.019 ± 0.00035

hPPB

Atom Bond count complexity

order	n_splits	mean	std
asc	2	0.396 ± 0.00036	0.013 ± 0.00028
	3	0.400 ± 0.00042	0.015 ± 0.00027
	4	0.402 ± 0.00029	0.011 ± 0.00039
desc	2	0.386 ± 0.000378	0.015 ± 0.00027
	3	0.393 ± 0.00055	0.018 ± 0.00036
	4	0.397 ± 0.00048	0.018 ± 0.00038
control		0.392 ± 0.00052	0.019 ± 0.00035

A.3 Complexity-Activity plots

hPPB

fsp3 complexity

order	n_splits	mean	std
asc	2	0.399 ± 0.00028	0.012 ± 0.00022
	3	0.417 ± 0.00108	0.044 ± 0.00083
	4	0.412 ± 0.00054	0.021 ± 0.00039
desc	2	0.411 ± 0.00041	0.015 ± 0.00032
	3	0.419 ± 0.00065	0.024 ± 0.00043
	4	0.409 ± 0.00055	0.021 ± 0.00041
control		0.392 ± 0.00052	0.019 ± 0.00035

MDR1 ER

SPS complexity

order	n_splits	mean	std
asc	2	0.363 ± 0.00052	0.019 ± 0.00037
	3	0.432 ± 0.00160	0.060 ± 0.00129
	4	0.441 ± 0.00153	0.059 ± 0.00116
desc	2	0.417 ± 0.00091	0.038 ± 0.00082
	3	0.547 ± 0.00311	0.101 ± 0.00188
	4	0.591 ± 0.00483	0.183 ± 0.00339
control		0.375 ± 0.00103	0.039 ± 0.00065

MDR1 ER

Atom Bond count complexity

order	n_splits	mean	std
asc	2	0.407 ± 0.00053	0.022 ± 0.00042
	3	0.482 ± 0.00153	0.061 ± 0.00121
	4	0.555 ± 0.00224	0.081 ± 0.00160
desc	2	0.413 ± 0.00048	0.019 ± 0.00036
	3	0.438 ± 0.00048	0.018 ± 0.00034
	4	0.444 ± 0.00039	0.016 ± 0.00028
control		0.375 ± 0.00103	0.039 ± 0.00065

MDR1 ER

fsp3 complexity

order	n_splits	mean	std
asc	2	0.375 ± 0.00107	0.043 ± 0.00076
	3	0.381 ± 0.00096	0.039 ± 0.00070
	4	0.375 ± 0.00055	0.022 ± 0.00048
desc	2	0.414 ± 0.00080	0.031 ± 0.00055
	3	0.535 ± 0.00290	0.111 ± 0.00247
	4	0.447 ± 0.00122	0.048 ± 0.00093
control		0.375 ± 0.00103	0.039 ± 0.00065

Solubility

SPS complexity

order	n_splits	mean	std
asc	2	0.395 ± 0.00042	0.016 ± 0.00032
	3	0.401 ± 0.00078	0.029 ± 0.00063
	4	0.409 ± 0.00065	0.023 ± 0.00041
desc	2	0.394 ± 0.00037	0.014 ± 0.00026
	3	0.413 ± 0.00060	0.023 ± 0.00047
	4	0.408 ± 0.00039	0.014 ± 0.00028
control		0.399 ± 0.00039	0.016 ± 0.00032

Solubility

Atom Bond count complexity

order	n_splits	mean	std
asc	2	0.415 ± 0.00037	0.015 ± 0.00024
	3	0.418 ± 0.00050	0.019 ± 0.00037
	4	0.417 ± 0.000025	0.010 ± 0.00018
desc	2	0.406 ± 0.00029	0.010 ± 0.00020
	3	0.406 ± 0.000298	0.012 ± 0.00022
	4	0.411 ± 0.00036	0.013 ± 0.000024
control		0.399 ± 0.00039	0.016 ± 0.00032

Solubility

fsp3 complexity

order	n_splits	mean	std
asc	2	0.403 ± 0.00040	0.016 ± 0.00033
	3	0.409 ± 0.00055	0.021 ± 0.00042
	4	0.427 ± 0.00062	0.023 ± 0.00038
desc	2	0.405 ± 0.00022	0.010 ± 0.00019
	3	0.422 ± 0.00054	0.023 ± 0.00041
	4	0.433 ± 0.00051	0.021 ± 0.00039
control		0.399 ± 0.00039	0.016 ± 0.00032

TDC Solubility

SPS complexity

order	n_splits	mean	std
asc	2	0.895 ± 0.00091	0.037 ± 0.00085
	3	0.976 ± 0.00115	0.044 ± 0.00080
	4	1.001 ± 0.00132	0.049 ± 0.00097
desc	2	0.827 ± 0.00088	0.034 ± 0.00074
	3	0.936 ± 0.00129	0.052 ± 0.00106
	4	0.936 ± 0.00131	0.052 ± 0.00105
control		0.782 ± 0.00085	0.029 ± 0.00049

TDC Solubility

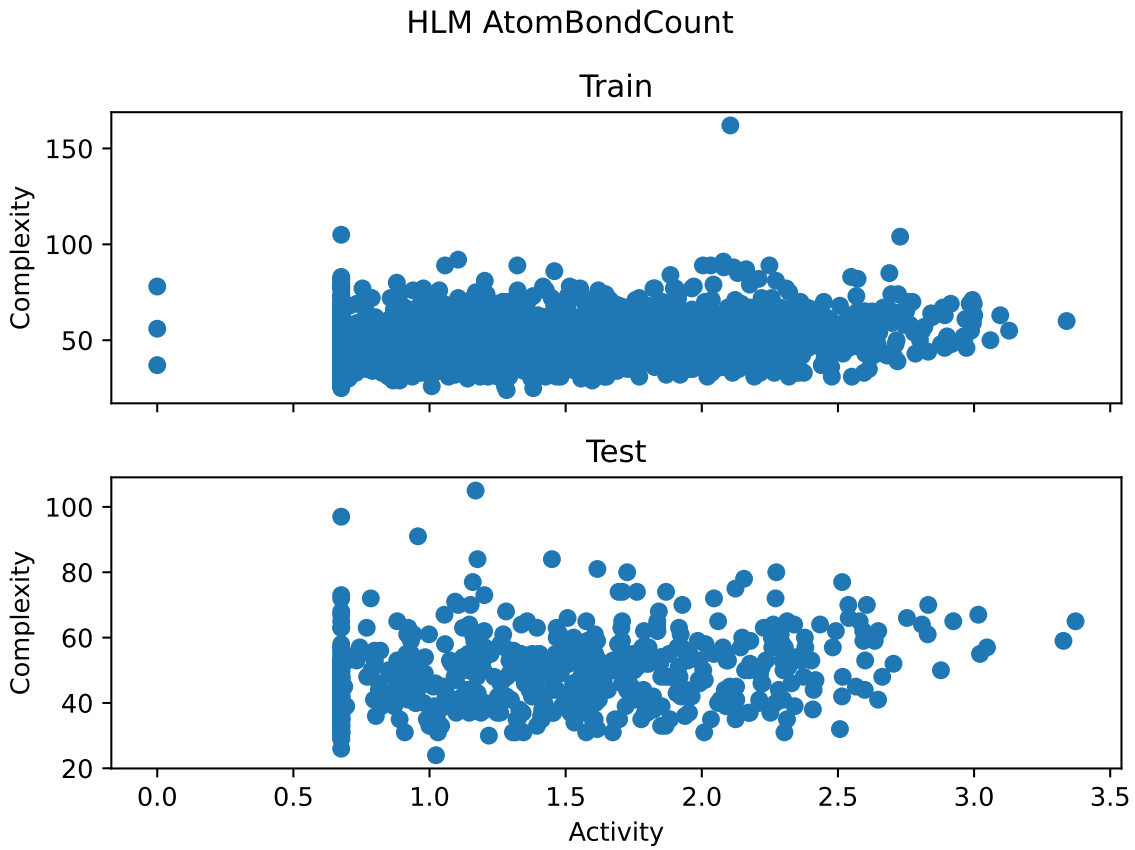
Atom Bond count complexity

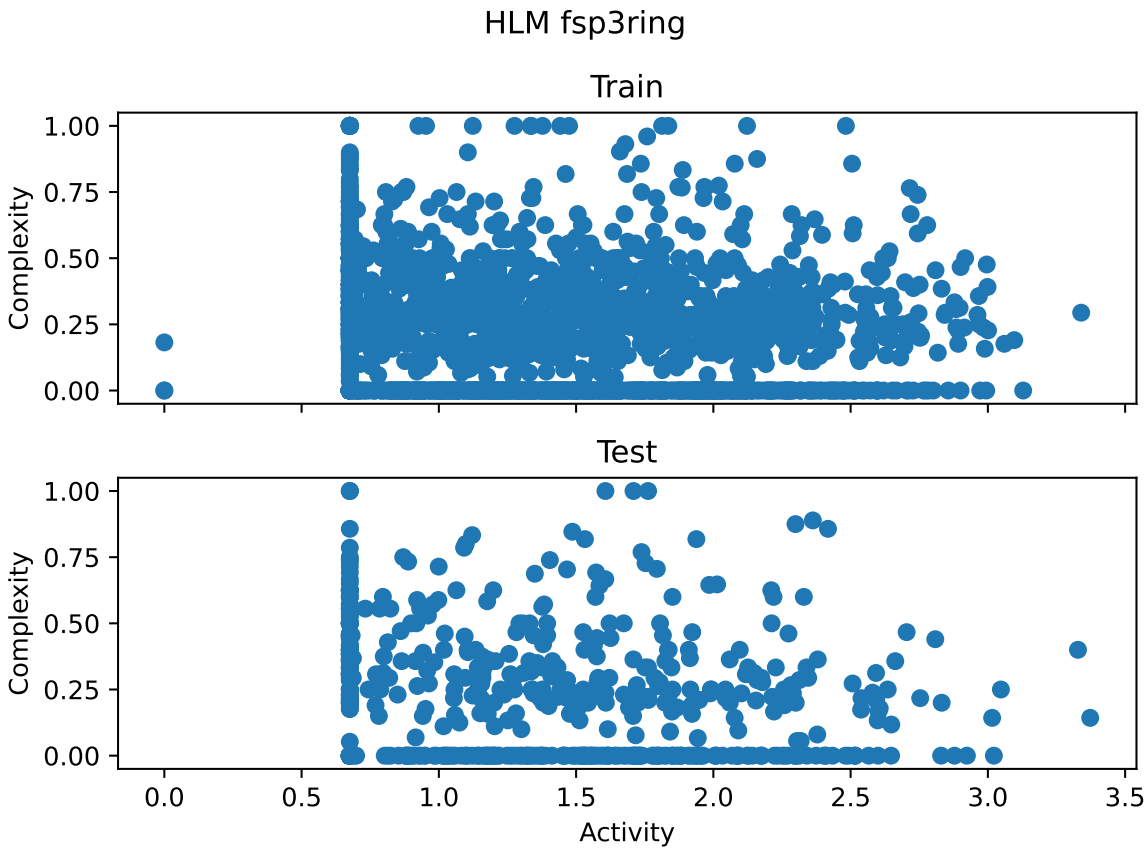
order	n_splits	mean	std
asc	2	1.062 ± 0.00171	0.060 ± 0.00115
	3	1.184 ± 0.00183	0.069 ± 0.00123
	4	1.234 ± 0.00161	0.056 ± 0.00117
desc	2	0.953 ± 0.00172	0.068 ± 0.00153
	3	1.005 ± 0.00247	0.093 ± 0.00203
	4	1.161 ± 0.00169	0.064 ± 0.00119
control		0.782 ± 0.00085	0.029 ± 0.00049

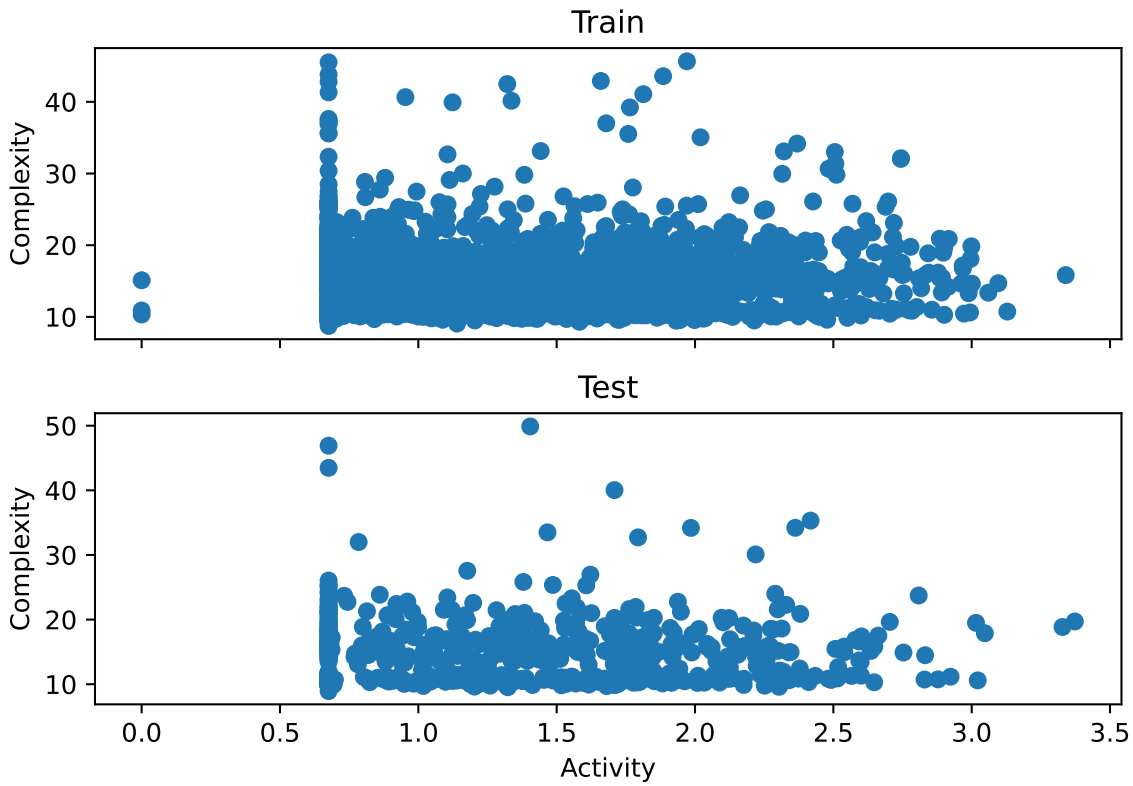
TDC Solubility

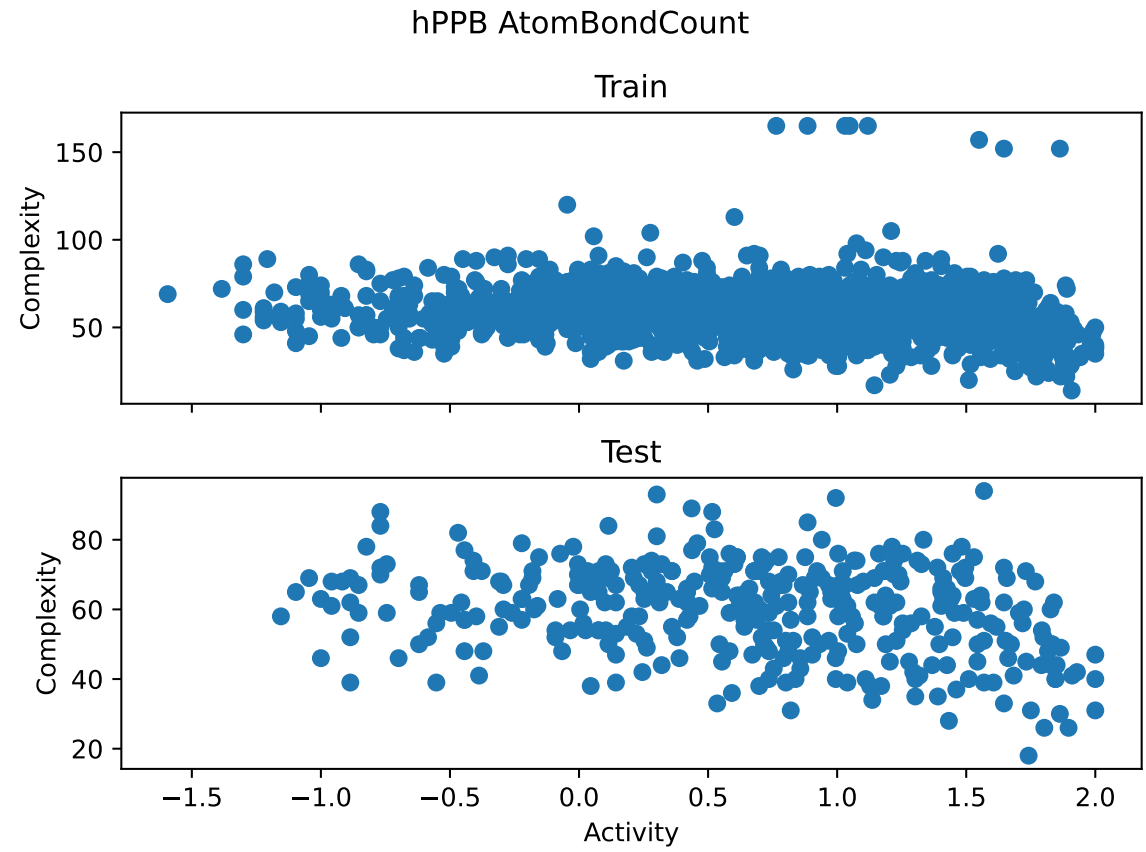
fsp3 complexity

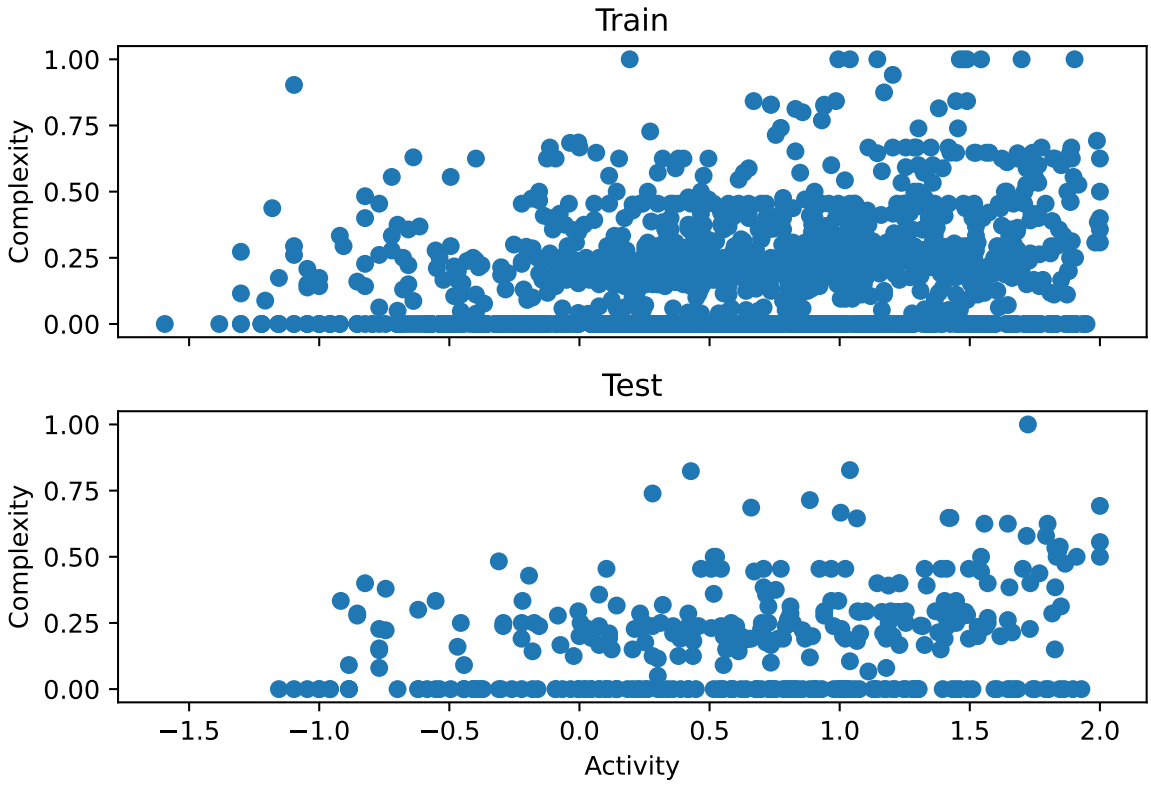
order	n_splits	mean	std
asc	2	0.788 ± 0.00085	0.036 ± 0.00065
	3	0.806 ± 0.00050	0.020 ± 0.00037
	4	0.846 ± 0.00100	0.036 ± 0.00067
desc	2	0.793 ± 0.000072	0.031 ± 0.00071
	3	0.798 ± 0.00060	0.027 ± 0.00054
	4	0.811 ± 0.00080	0.029 ± 0.00059
control		0.782 ± 0.00085	0.029 ± 0.00049

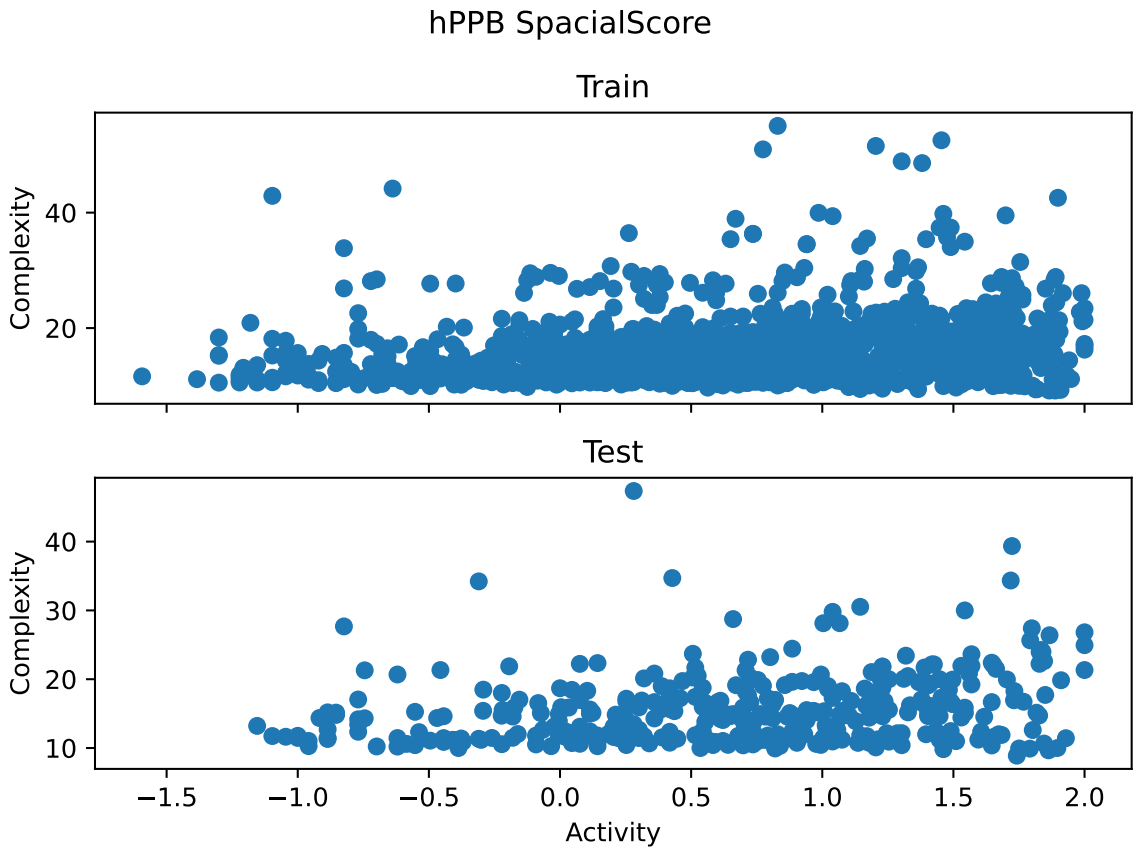


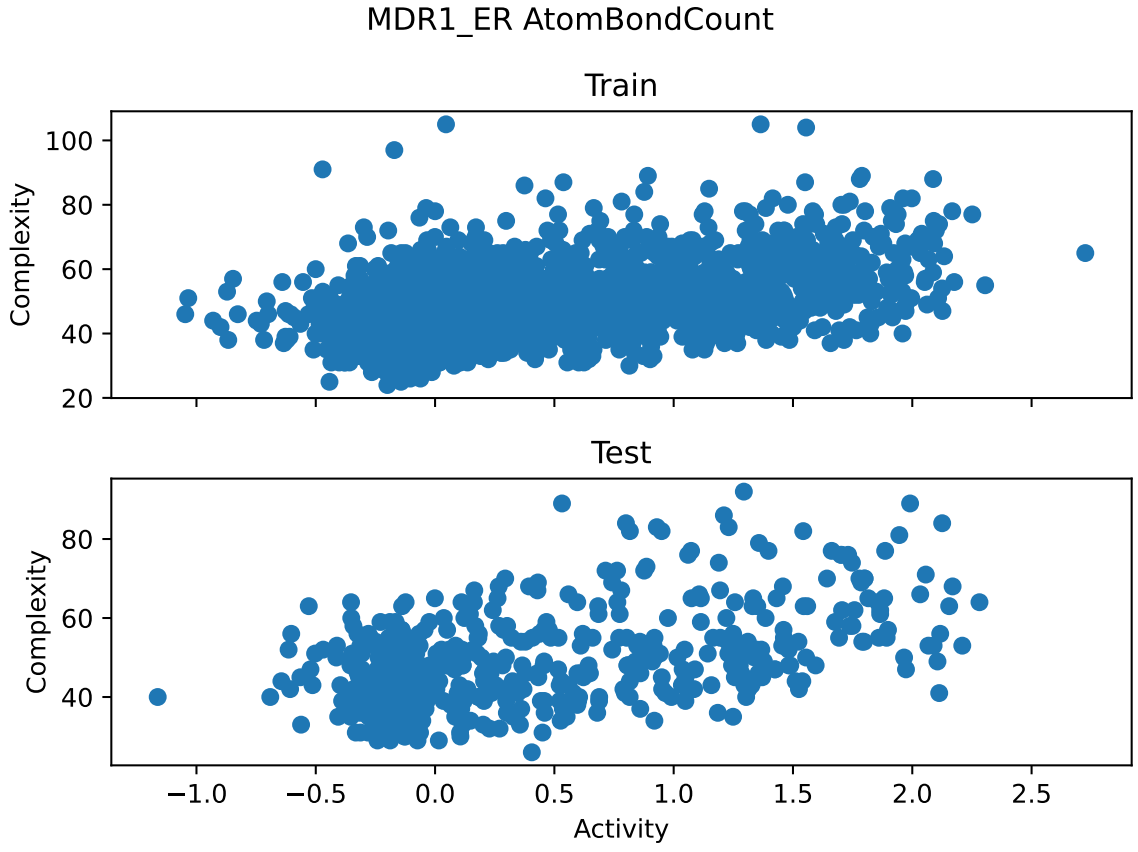

HLM fsp3ring
Train
Test
Complexity
Activity
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

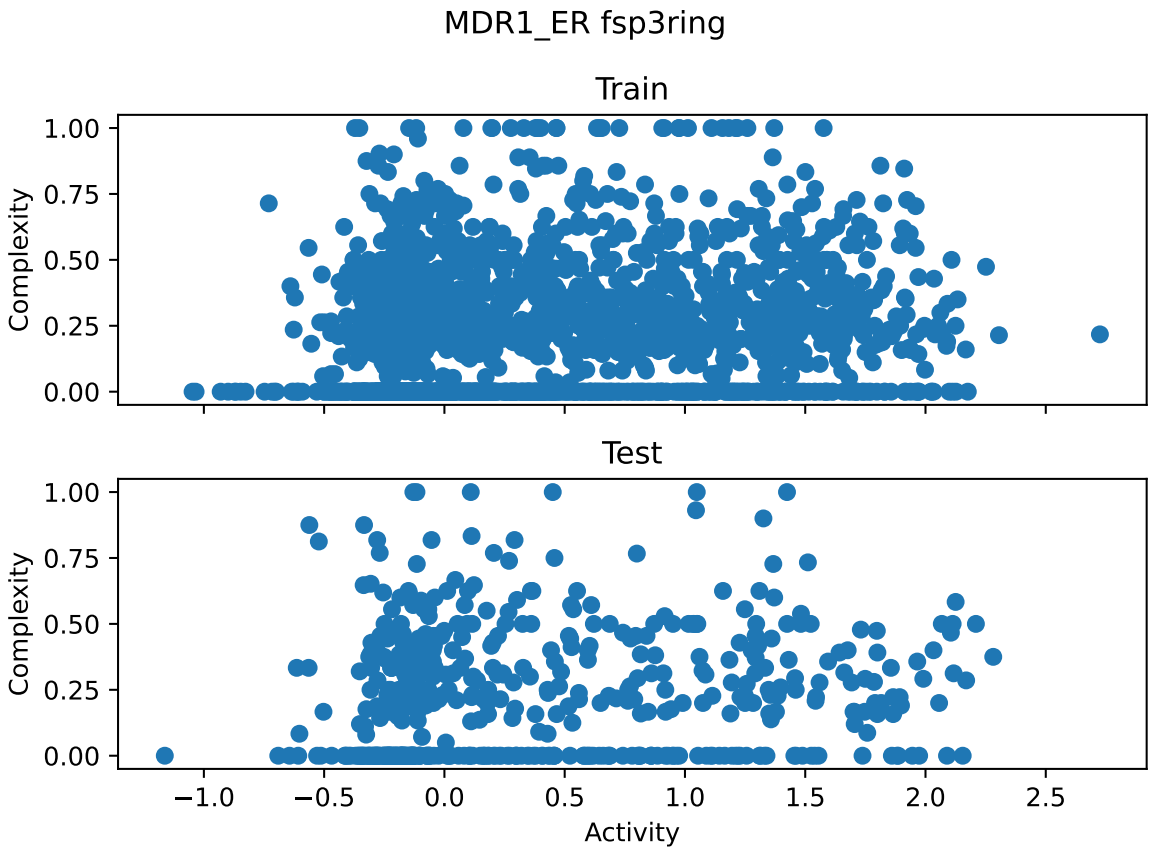

HLM SpacialScore
Train
Test
Complexity
Activity
40
30
20
10
50
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

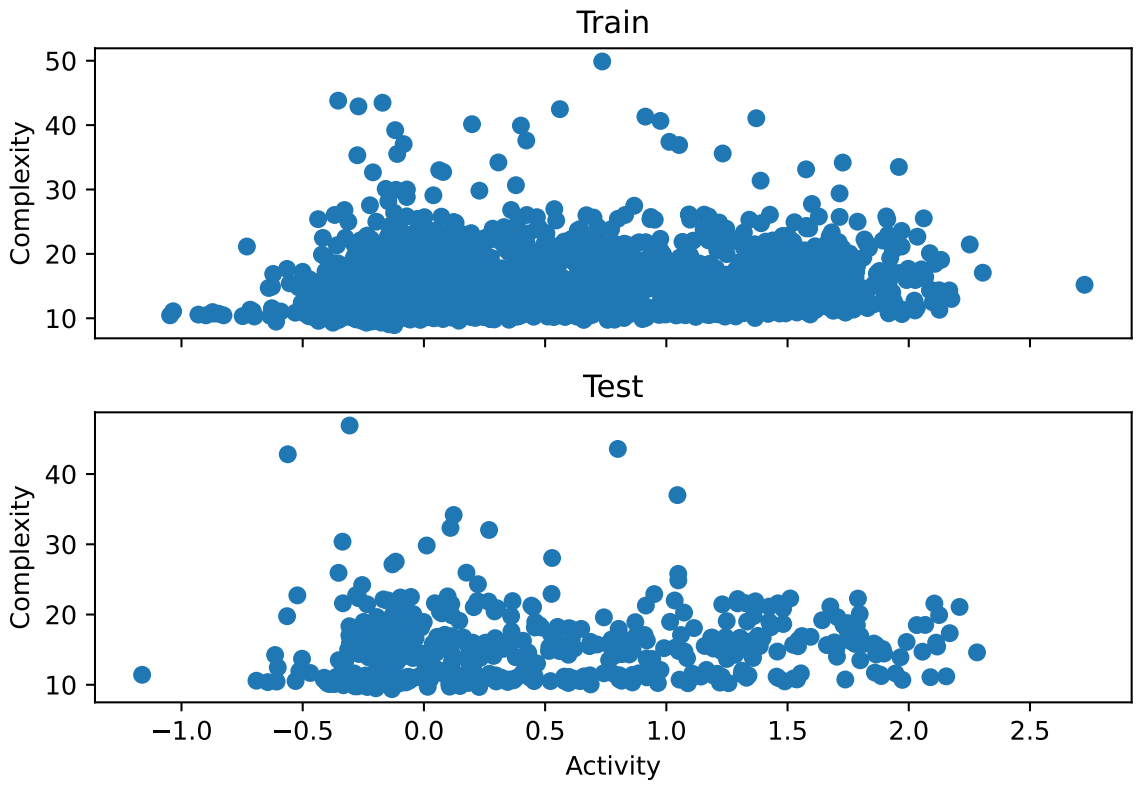

hPPB AtomBondCount
Train
Test
Complexity
Activity
150
100
50
80
60
40
20
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

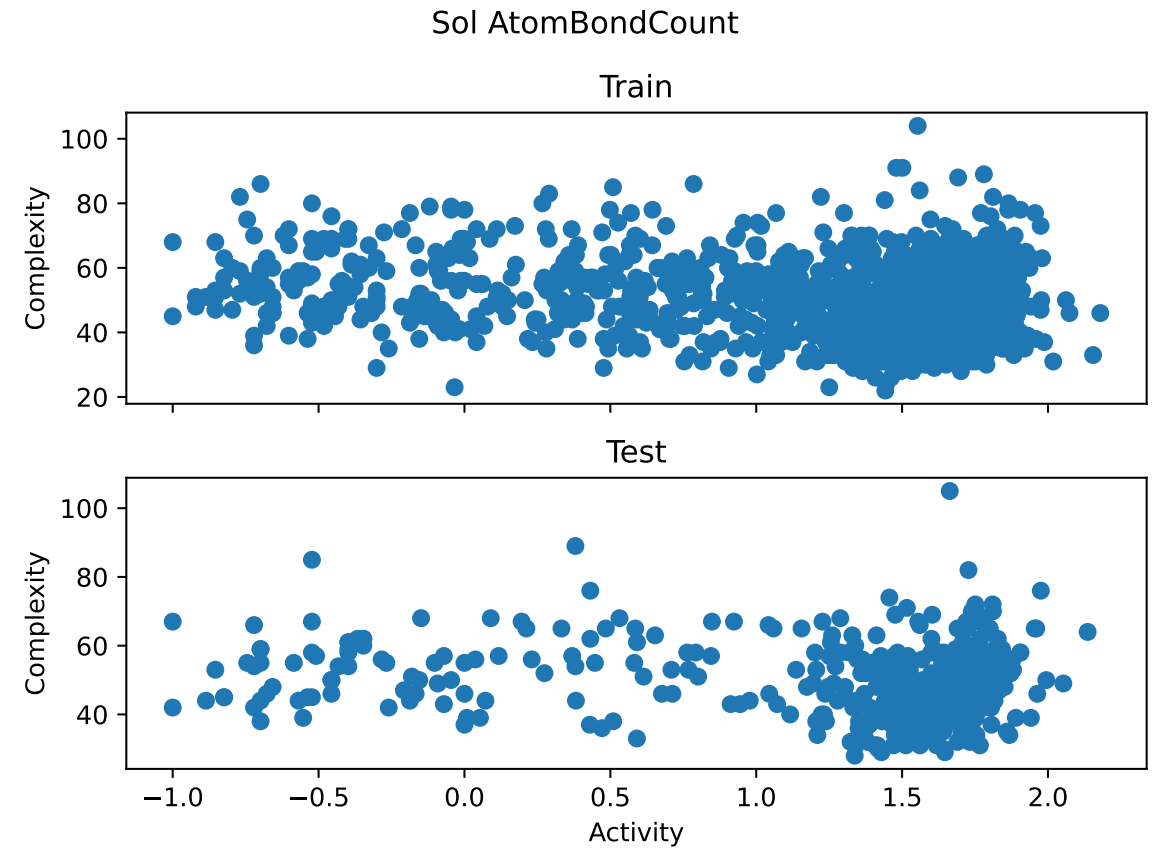

hPPB fsp3ring
Train
Test
Complexity
Activity
1.00
0.75
0.50
0.25
0.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

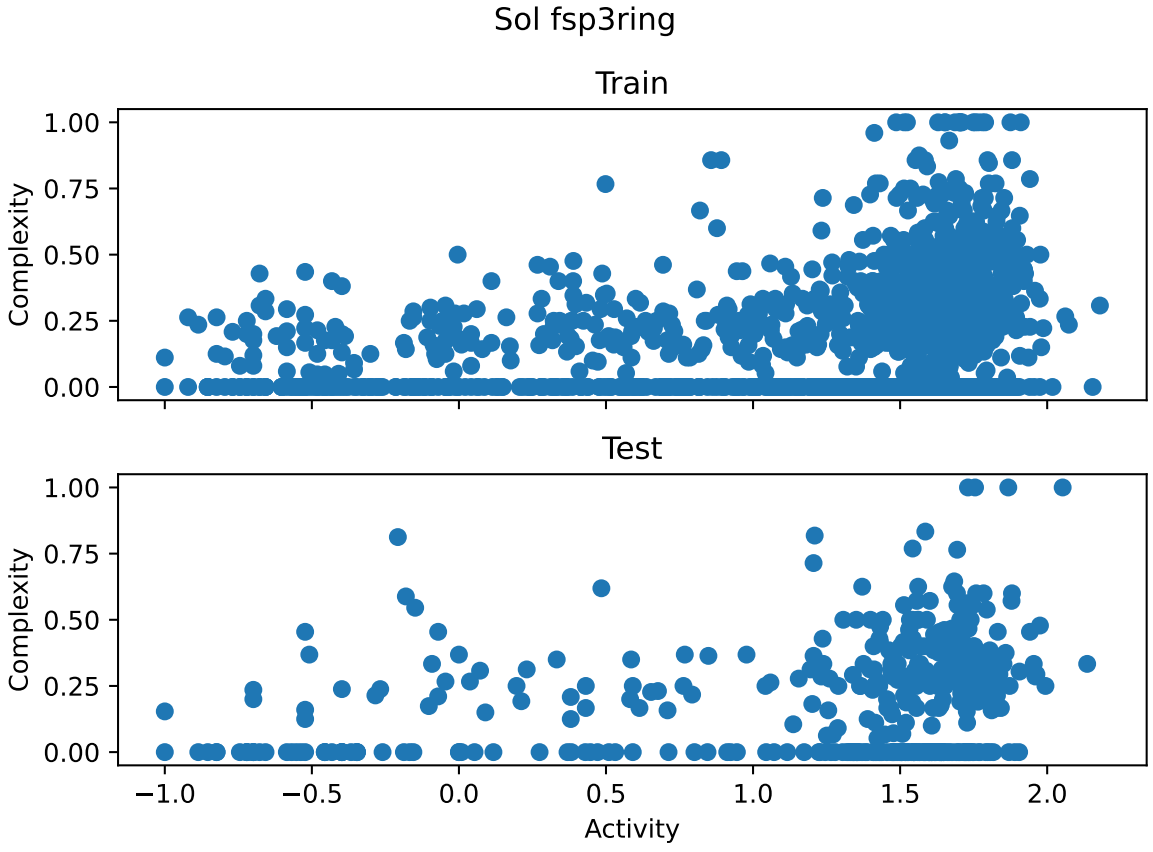

hPPB SpacialScore
Train
Complexity
40
20
Test
Complexity
40
30
20
10
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Activity

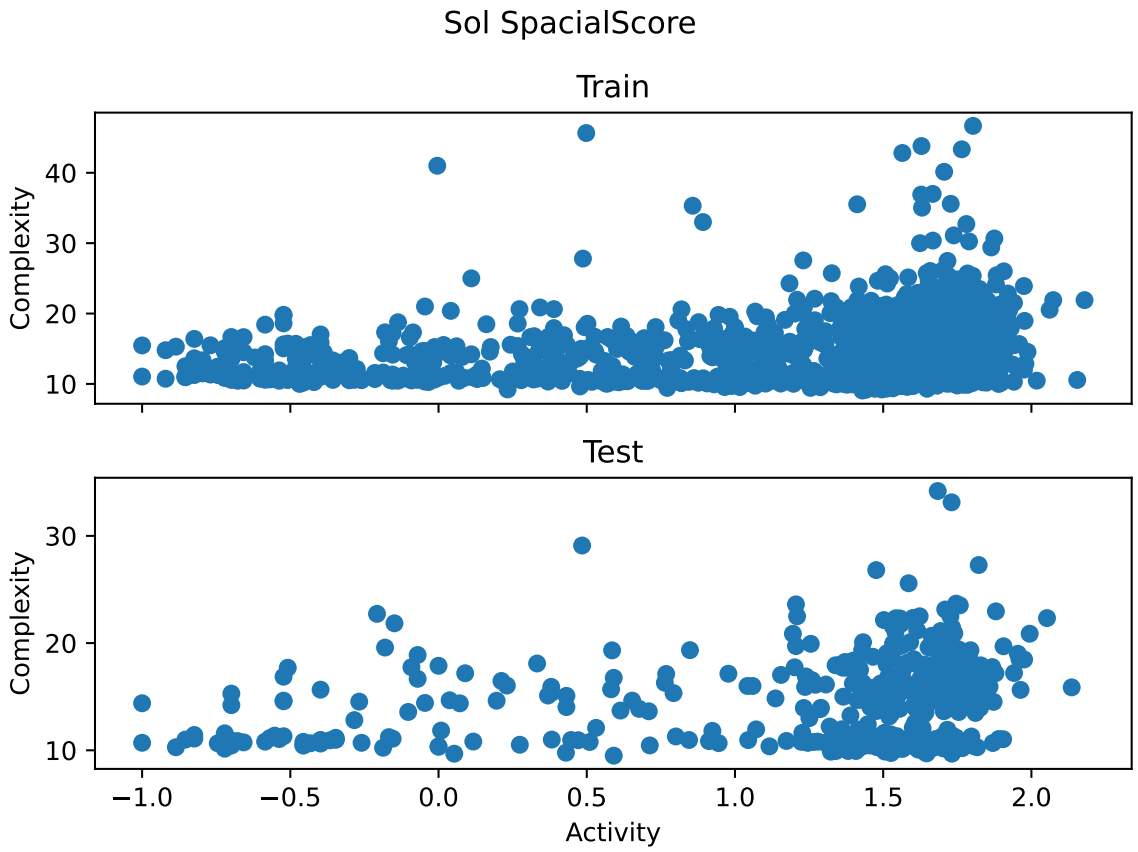

MDR1_ER AtomBondCount
Train
Complexity
100
80
60
40
20
Test
Complexity
80
60
40
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
Activity

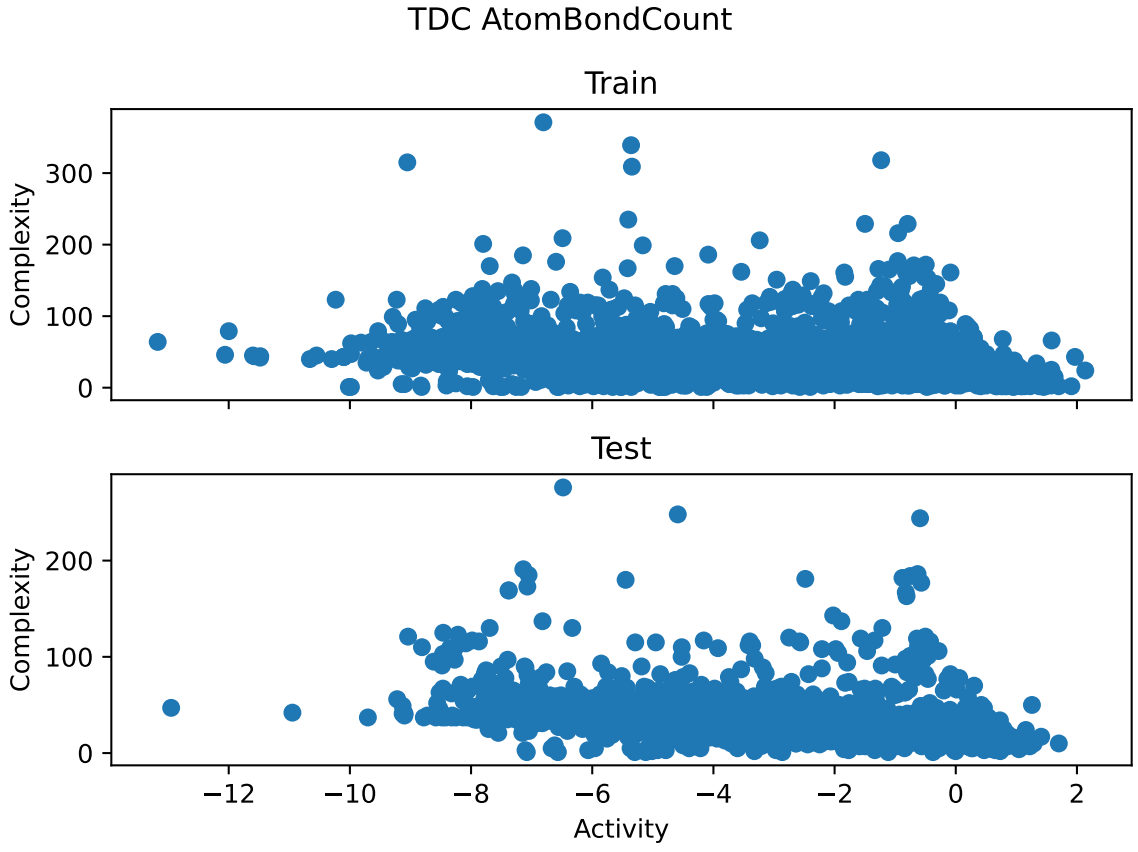

MDR1_ER fsp3ring
Train
Test
Complexity
Activity
1.00
0.75
0.50
0.25
0.00
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5

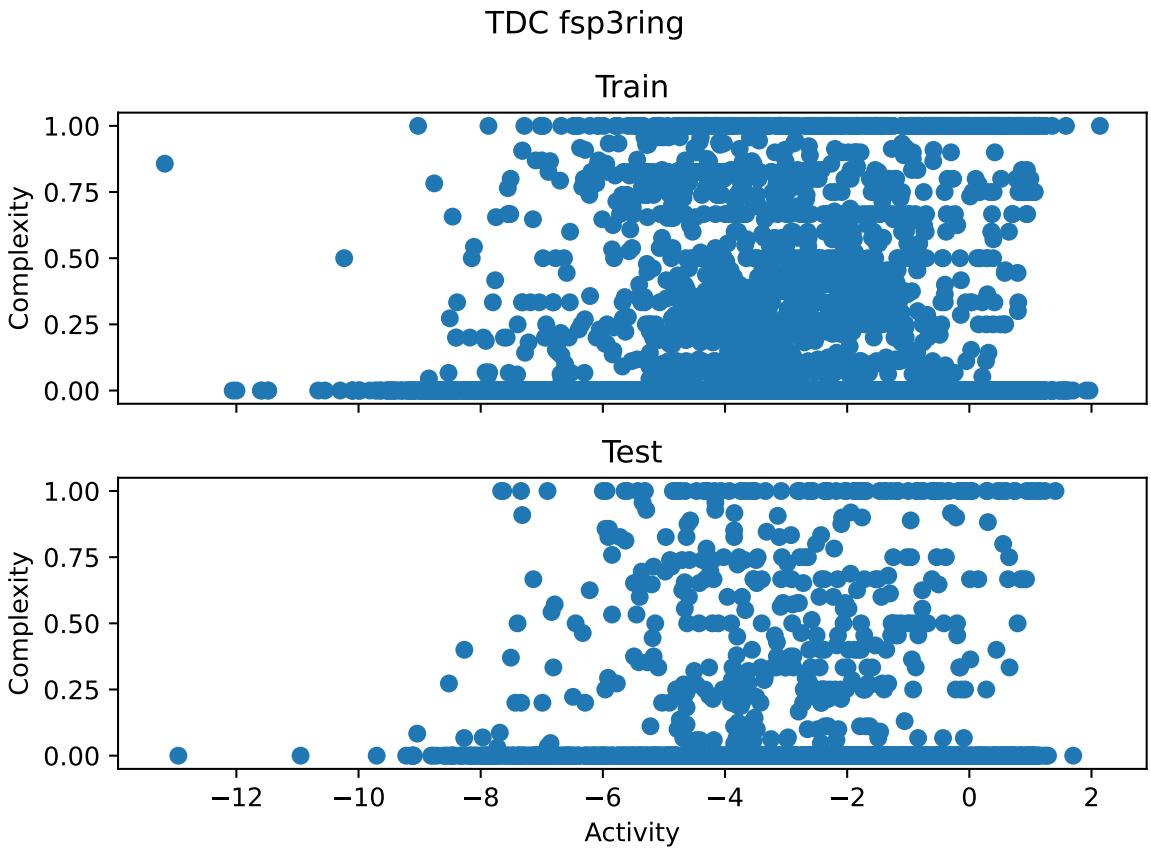

MDR1_ER SpacialScore
Train
Test
Complexity
Activity
50
40
30
20
10
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5

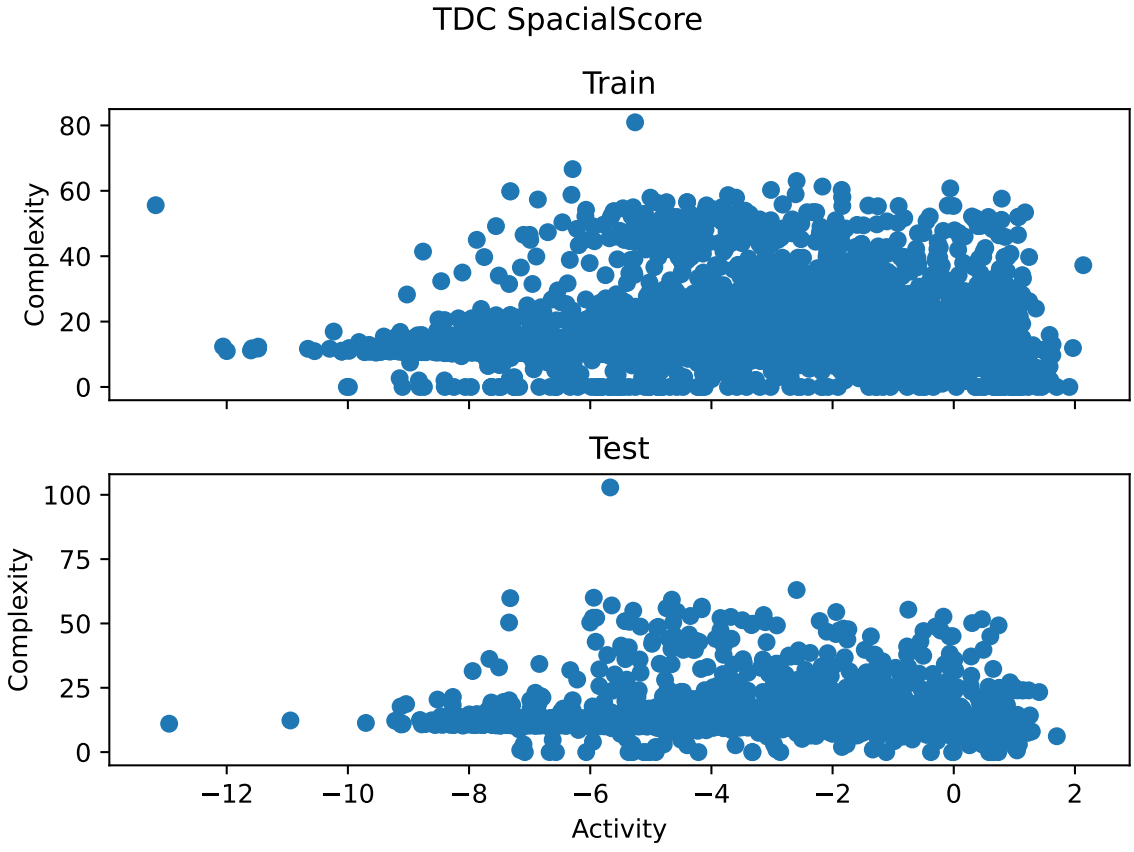

Sol AtomBondCount
Train
Test
Complexity
Activity
100
80
60
40
20
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


Sol fsp3ring
Train
Test
Complexity
Activity
1.00
0.75
0.50
0.25
0.00
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


Sol SpacialScore
Train
Complexity
40
30
20
10
Test
Complexity
30
20
10
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Activity


TDC AtomBondCount
Train
Complexity
300
200
100
0
Test
Complexity
200
100
0
−12
−10
−8
−6
−4
−2
0
2
Activity


TDC fsp3ring
Train
Test
Complexity
Activity
1.00
0.75
0.50
0.25
0.00
−12
−10
−8
−6
−4
−2
0
2


TDC SpacialScore
Train
Test
Complexity
Activity
80
60
40
20
0
100
75
50
25
0
−12
−10
−8
−6
−4
−2
0
2