Interpretation of Compound Fragments via Attentive Recursive Tree

by

Nural Özel

M.S., Computational Science and Engineering, Boğaziçi University, 2023

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computational Science and Engineering
Boğaziçi University

2023

Interpretation of Compound Fragments via Attentive Recursive Tree

APPROVED BY:

Prof. .Kutlu Ülgen. . . .
(Thesis Supervisor)

Assoc.. Prof.. Arzucan. Özgür
(Thesis Co-supervisor)

DATE OF APPROVAL: 01.06.2023

ACKNOWLEDGEMENTS

This work is supported by Scientific and Technological Research Council of Turkey (TÜBİTAK) (Project No: 119E133).

First and foremost, I would like to express my infinite gratitude and appreciation to my supervisors Prof. Kutlu Ülgen and Assoc. Prof. Arzucan Özgür for rekindling the passion within me for knowledge, learning and most importantly, the curiosity.

Also, I express my sincere gratitude to Asu Büşra Temizer and Taha Khoulani who are doctoral candidates at the Department of Pharmaceutical Chemistry of Istanbul University for all of their assistance within the "Interpretability" and the "Discussion" chapters of this work.

Last but not least, my dear family, I am very aware those endeavors of yours to make easier this steep and thorned road of mine for me, as much as you can. I thank and owe you for every single thing that I have, I will have, and even my very self until the very end.

−..... − ...

− − −. − −.. − −.. − ..... − ...

....−

. − −. − .

− − .....

ABSTRACT

Interpretation of Compound Fragments via Attentive Recursive Tree

The discovery of new drug-like chemicals with desired properties is a challenging and costly process in the pharmaceutical industry. To facilitate this process in the preclinical phase, many different neural network models have been proposed for different tasks (e.g., drug-target affinity prediction, molecular property prediction, target-specific molecule generation). Despite producing successful results, they usually lack interpretability. To comprehend the significance of each fragment in the relevant compounds, we employed the Attention Recursive Tree (AR-Tree) model. Thanks to its task-specific attention mechanism, AR-Tree highlights the significant fragments of compounds by positioning them closer to the root of the tree structure. In this way, the identified significant fragments can be used to design new compounds with desired properties in future research. We experimented with five different classification and four different regression tasks of the MoleculeNet as benchmark tasks. The results of the experiments show that the proposed architecture succeeded in finding chemically meaningful fragments for the corresponding tasks.

ÖZET

Dikkatli Özçağrılı Ağaç ile Kimyasal Fragmanlarının Anlamlandırılması

İstenen özelliklere sahip yeni ilaç benzeri kimyasalların keşfi, ilaç endüstrisinde zorlu ve maliyetli bir süreçtir. Klinik öncesi aşamada bu süreci kolaylaştırmak adına, farklı görevler için birçok farklı sinir ağı modeli önerilmiştir (örneğin, ilaç-hedef afinite tahmini, moleküler özellik tahmini, hedefe özel molekül üretimi). Başarılı sonuçlar üretmelerine rağmen, bu modeller genellikle yorumlanabilirlikten yoksundurlar. İlgili bileşiklerdeki her bir parçanın önemini belirleyebilmek için Dikkatli Özçağrılı Ağaç (AR-Tree) modelini kullandık. Göreve özgün dikkat mekanizması sayesinde AR-Tree, bileşiklerdeki önemli parçaları ağaç yapısında köke daha yakın yerleştirerek vurgular. Bu şekilde, belirlenen önemli fragmanlar gelecekteki araştırmalarda istenen özelliklere sahip yeni bileşikler tasarlamak için kullanılabilir. Denek görevleri olarak MoleculeNet'in beş farklı sınıflandırma ve dört farklı regresyon görevini denedik. Deneylerin sonuçları, önerilen mimarinin ilgili görevler için kimyasal olarak anlamlı parçalar bulmayı başardığını göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

β	Cleavage site of the target molecule
θ	Parameter set of the multi layer perceptron layers
σ	The sigmoid function
$\#$	Number
b	Beginning index of the sentence
$\mathbf{b_c}$	Bias matrix of the cell state of the bidirectional Tree-LSTM layer
$\mathbf{c}$	Cell state of the bidirectional Tree-LSTM layer
$\mathbf{c_i}$	Cell state of the bidirectional Tree-LSTM layer in the parent node
$\overrightarrow{\mathbf{c_i}}$	Cell state of the right-directional Tree-LSTM layer
$\overleftarrow{\mathbf{c_i}}$	Cell state of the left-directional Tree-LSTM layer
$\mathbf{d_{max}}$	Distance between the fartest leaf node from the root node and the root in the tree structure
D	$n-octanol/water$ distribution coefficient
D_x	Dimension of the word embedding layer
DS	Dataset
e	Ending index of the sentence
$\overline{\mathbf{e}}$	Average test loss of the fragment in the dataset
$\mathbf{e_k}$	Test loss of the tree structure
$\mathbf{f_i}$	The gate of the bidirectional Tree-LSTM layer in the parent node
$\mathbf{f_L}$	The gate of the bidirectional Tree-LSTM layer in the left child node
$\mathbf{f_R}$	The gate of the bidirectional Tree-LSTM layer in the right child node
FS	Fragment set
$\mathbf{g}$	The candidate vector in the bidirectional Tree-LSTM layer
$\mathbf{h}$	Hidden vector of the bidirectional Tree-LSTM layer

$\mathbf{h_i}$	Hidden vector of the bidirectional Tree-LSTM layer in the parent node
$\mathbf{h_L}$	Hidden vector of the bidirectional Tree-LSTM layer in the left child node
$\mathbf{h_R}$	Hidden vector of the bidirectional Tree-LSTM layer in the right child node
$\overleftarrow{\mathbf{h_{i-1}}}$	Hidden vector of the left-directional Tree-LSTM layer in the previous word's node
$\overrightarrow{\mathbf{h_{i-1}}}$	Hidden vector of the right-directional Tree-LSTM layer in the previous word's node
$\overleftarrow{\mathbf{h_i}}$	Hidden vector of the left-directional Tree-LSTM layer
$\overrightarrow{\mathbf{h_i}}$	Hidden vector of the right-directional Tree-LSTM layer
$\overleftarrow{\mathbf{h_{i+1}}}$	Hidden vector of the left-directional Tree-LSTM layer in the next word's node
$\overrightarrow{\mathbf{h_{i+1}}}$	Hidden vector of the right-directional Tree-LSTM layer in the next word's node
$\mathbf{i}$	Word index of the sentence
$\mathbf{ig}$	The input gate of the bidirectional Tree-LSTM layer
$\mathbf{L}$	Number of data in the dataset
$\overleftarrow{\mathbf{LSTM}}$	Left-directional Tree-LSTM layer
$\overrightarrow{\mathbf{LSTM}}$	Right-directional Tree-LSTM layer
$\mathbf{m}$	Repeat count of the fragment in the dataset
$\mathbf{n}$	Repeat count of the fragment in the tree structure
N	Number of words in the sentence
$\mathbf{o}$	The output gate of the bidirectional Tree-LSTM layer
$O(n^3)$	Cubic time complexity
$\overline{\mathbf{P}}$	Average point of the fragment in the dataset
$\mathbf{P_i}$	Point of the node
$\mathbf{P_F}$	Final point of the fragment in the fragment set
$\mathbf{P^{mol}}$	Total point of the fragment in the tree structure
$\mathbf{P_j^{mol}}$	Repeat count of the fragment in the tree structure
R	Root node of the tree structure

S	Sentence
t	Parent node
$t.left$	Right child node of the parent node
$t.right$	Left child node of the parent node
$tanh$	The hyperbolic tangent function
T	Tree structure of the sentence
$\mathbf{W_c}$	Weight matrix of the cell state of the bidirectional Tree-LSTM layer
$\mathbf{x}$	Unnormalized point of the fragment in the fragment set
$\mathbf{x_i}$	Word embedding vector
$\mathbf{x_{max}}$	Maximum point in the fragment set
$\mathbf{x_{min}}$	Minimum point in the fragment set
$\hat{\mathbf{x}}$	Normalized point of the fragment in the fragment set

LIST OF ACRONYMS/ABBREVIATIONS

3D	Three-Dimensional
Aβ	β-Amyloid
ADR	Adverse Drug Reaction
ADR	Adverse Drug Reaction
AR-Tree	Attentive Recursive Tree
Asp	Aspartic acid
BACE	β-site Amyloid precursor protein Cleaving Enzyme
BACE1	β-site Amyloid precursor protein Cleaving Enzyme 1
BBB	Blood-Brain Barrier
BBBP	Blood-Brain Barrier Penetration
BCE	Binary Cross-Entropy
CID	Chemical Identification Number of PubChem database
Clf	Classification
ClinTox	Clinical Trial Toxicity
CNN	Convolutional Neural Network
CNS	Central Nervous System
COX	Cyclo-oxygenase
CSV	Comma-Separated Values
CT_TOX	Clinical Toxicity
CYK	Cocke–Younger–Kasam chart parser
DFD	Dynamic Fragment Dictionary
DL	Deep Learning
DNN	Dense Neural Network
ESOL	Estimated Solubility
FDA	Food and Drug Administration
FreeSolv	Free Solvation
H-bond	Hydrogen bond
HEA	Hydroxyethylamine

LRP	Layerwise Relevance Propagation
ML	Machine Learning
MLP	Multi-Layer Perceptron
nrn	Neuron
NSAID	Non-Steroidal Anti-Inflammatory Drug
pH	Potential Hydrogen
PRC-AUC	Area Under the Precision-Recall Curve
Reg	Regression
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMSE	Root-Mean-Square Error
ROC-AUC	Area Under the Receiver Operating Characteristic Curve
SAR	Structure-Activity Relationship
SIDER	Side Effect Resource
SMILES	Simplified Molecular Input Line Entry System
SOTA	State-of-the-Art
SR_p53	p53 Stress-Response pathway activation
STG	Straight-Through Gumbel-Softmax estimator
STR	Structure-Toxicity Relationship
TBL	Tree-Bidirectional-LSTM
TF-IDF	Term Frequency - Inverse Document Frequency
Tox21	Toxicology in the 21st century
Tree-LSTM	Tree-structured Long Short-Term Memory
Tree-RNN	Tree-structured Recurrent Neural Network

1. INTRODUCTION

Determining the physicochemical and functional properties of novel compounds (e.g., blood-brain barrier penetration, toxicity, and lipophilicity) is an important step in the discovery of new drugs. However, this is time-consuming and costly as it requires complicated experiments in the laboratory. Therefore, computational methods (e.g., deep learning (DL)-based) have been developed to reduce the time and cost involved in these processes [1]. One of the most important parts of DL algorithms is learning molecular representations that should contain the most distinctive features in a compact form [2, 3]. However, the interpretability of the models is weak despite their impressive performance on known compounds, because it is very difficult to understand how the models make predictions and what types of patterns are captured by the model because the models work like a black box.

In this work, we use an interpretation strategy with pre-trained Attentive Recursive Tree (AR-Tree) [4] to learn general compound representations. We then fine-tune the model to tailor these representations to benchmark tasks. As a Tree-structured Long Short-Term Memory (Tree-LSTM)-based sentence embedding model, AR-Tree was developed to learn task-specific sentence embeddings considering the importance of fragments to the task at hand. It emphasizes the important fragments by placing them closer to the root of the tree, thanks to its task-specific attention mechanism for parsing sentences. In addition, owing to construction of a latent tree based on the importance of the fragments, it is also possible to interpret the embeddings created by the model. In other words, for each task, it is easy to identify which fragments are necessary for each activity by looking at their placements in the tree structure. This is an important aspect since understanding the crucial components of compounds can help experts make better predictions of physicochemical and functional features.

2. LITERATURE SURVEY

In the literature, some approaches to calculate the feature importances have been proposed with the intent of interpretability. These approaches mentioned below subsections alter certain inputs or neurons while observing the effects on subsequent neurons in the network.

2.1. Backpropagation-based Methods

DeepLIFT [5] is an algorithm that rates the significance of inputs for a specific outcome. It is distinctive in two ways. First, it defines the important question in terms of variations from "reference" state, where the "reference" is selected based on the current issue. Using a difference variable from reference enables DeepLIFT to transmit an importance signal even when the gradient is zero and prevents artifacts brought on by gradient discontinuities, in contrast to other gradient-based algorithms. Second, DeepLIFT may identify dependencies that other techniques have overlooked by optionally taking into account separately the impacts of positive and negative contributions at nonlinearities. DeepLIFT scores can be effectively produced in a single backward pass once a prediction has been made since they are calculated using a backpropagation-like approach, making it efficient.

2.2. Forward Propagation-based Methods

[6] displayed the change in the activations of subsequent layers by obscuring various portions of an input picture. Using "in-silico mutagenesis" [7], virtual mutations were introduced at specific locations in a genomic sequence, and their effects on the result were measured.

[8] is based on an instance-specific method by [9] so called the prediction difference analysis. The proposed methodology by [8] is a method that similar to [6], but

the difference is both removing information from the image and evaluating the effect of this. For explaining classification decisions made by deep neural networks, the method is used to produce a saliency map for each $(instance, node)$ pair that highlights the parts (features) of the input that constitute most evidence for or against the activation of the given (internal or output) node.

[10, 11] used a technique for plotting the weights of a linear classifier or their p-values (as determined by permutation testing) [12,13] to visualize feature importances. These are independent of the input picture, and reading these weights in general may be deceptive, according to [14] and [15].

2.3. Layer-wise Relevance Propagation-based Methods

[16] suggested Layerwise Relevance Propagation (LRP) as a method for distributing significance ratings. [17, 18] demonstrated that the LRP rules for rectified linear units (ReLU) networks were equal within a scaling factor to an element-wise product between the saliency maps of [19] and the input ($gradient * input$) in the absence of adjustments to address numerical stability. The saturation issue or the thresholding phenomenon are still unaddressed, despite the fact that $gradient * input$ is often superior than gradients alone since it makes use of the input's sign and intensity.

2.4. Deconvolutional Network-based Methods

[19] suggested computing an image's saliency map utilizing the gradient of the output with reference to pixels of the input picture, while doing image classification tasks. Except for how they handled the nonlinearity at ReLU, the authors demonstrated that this was similar to deconvolutional networks [6]. When backpropagating importance using gradients, if the input to the ReLU during the forward pass is negative, the gradient flowing into the ReLU during the backward pass is zeroed out. The importance signal entering a ReLU during the backward pass of deconvolutional networks, in contrast, is zeroed out if and only if it is negative, regardless of the sign of

the input into the ReLU during the forward pass. The importance signal at a ReLU is zeroed out if either the input to the ReLU during the forward pass or the importance signal during the backward pass is negative, according to guided backpropagation [20].

With the exception that gradients that become negative during the backward run are eliminated at ReLUs, guided backpropagation may be compared to calculating gradients. Both guided backpropagation and deconvolutional networks may be unable to identify inputs that have a negative impact on the output due to the zeroing out of negative gradients.

2.5. Gradient-based Localization Methods

[19] showed that, first, the numerical optimization of the input picture may be used to produce intelligible visualisations of CNN classification models [21]. The final fully-connected classification layer's optimal neuron should be maximized to display the class of interest since, unlike [21], the net is trained in a supervised way (to identify the neuron in charge of each class in the unsupervised instance, [22] needed to consult a different collection of annotated picture data).

Second, using a single backpropagation run through a classification Convolutional Neural Network (CNN), [19] developed a technique for calculating the spatial support of a particular class in an image (image-specific class saliency map). Weakly supervised object localization may be done using these saliency maps.

Grad-CAM [23] creates a coarse-grained feature-importance map by classifying the final convolutional layer's feature maps according to the gradients of each class with respect to each feature map, and then using the weighted activations of the feature maps to determine which inputs are most crucial. The authors suggested conducting an elementwise product between the scores acquired from Grad-CAM and the scores received from guided backpropagation, known as Guided Grad-CAM, to get more finely grained feature significance. Yet since negative gradients are zeroed out during backpropaga-

tion, this approach inherits the drawbacks of guided backpropagation. Moreover, it is unique to convolutional neural networks.

[24] integrated the gradients as the inputs are scaled up from a beginning value (such as all zeros) to their present value, instead of calculating the gradients simply at the input's current value. Nevertheless, numerically deriving high-quality integrals adds processing complexity. This fixes the saturation and thresholding issues. Moreover, this strategy may still provide false findings.

2.6. Latent Tree-based Methods

[25] use a general shift-reduce parser, whose training depends on ground-truth parsing trees, to construct trees and combine semantics. Combining latent tree learning with Tree-structured Recurrent Neural Networks (Tree-RNN) has been shown to be a successful strategy for sentence embedding since it simultaneously optimizes the sentence compositions and a task-specific target. For instance, [26] train a shift-reduce parser using reinforcement learning (RL) without using any ground truth.

[27] replace the shift-reduce parser with a Cocke–Younger–Kasami chart parser (CYK) [28–30] and completely differentiate it using the softmax annealing method. Nevertheless, since the chart parser needs $O(n^3)$ time and space complexity, their approach has problems with both time and space. According to the easy-first parsing technique proposed by [31], each pair of neighboring nodes is scored using a query vector, and the best pair is greedily combined into one parent node at each stage. They allow end-to-end training by computing parent embedding in a hard categorical gating approach using the Straight-Through Gumbel-Softmax estimator (STG) [32]. Using various datasets, [33] compare the aforementioned models and show that [31] perform the best.

2.7. Attention-based Methods

Inter-attention [34,35], which needs a pair of sentences to attend with each other, and intra-attention [36, 37], which just requires the phrase, may be categorized as attention-based approaches. The latter is more flexible than the former. [38] use graphical models rather than recursive trees to include structural distributions into attention networks. Notice that current latent tree-based models treat all input words identically as leaf nodes and neglect the fact that various words contribute to sentence semantics to differing degrees, despite the fact that this is the basic driver of the attention process.

3. METHODOLOGY

3.1. Molecular Property Benchmark Tasks

We experimented with five different classification and four different regression datasets from MoleculeNet [39] as benchmark tasks. According to [39]; BBBP, ClinTox, SIDER and Tox21 datasets are categorized as physiology, BACE dataset is categorized as biophysics and ESOL, FreeSolv and Lipophilicity datasets are categorized as physical chemistry. Number of molecules and number of tasks of all the datasets can be seen in **Appendix 3.1** all together. Also, homogeneities of both the classification and regression datasets' distributions can be found in **Appendix A**

Table 3.1. Informations about datasets.

Dataset Name	# of Molecules	# of Tasks	Task Type
BACE	1513	1	clf, reg
BBBP	2039	1	clf
ClinTox	1478	2	clf
SIDER	1427	27	clf
Tox21	7831	12	clf
ESOL	1128	1	reg
FreeSolv	642	1	reg
Lipophilicity	4200	1	reg

clf and reg indicate classification and regression, respectively.

3.1.1. BACE Task

β-site Amyloid precursor protein Cleaving Enzyme (BACE) dataset contains compounds that inhibitors of human β-site amyloid precursor protein cleaving enzyme 1 (BACE1). The dataset contains regression (the half maximal inhibitory concentration (IC50)) and classification binding labels of the compounds. Both the regression and the classification datasets consist of 1513 same compounds.

3.1.2. BBBP Task

Blood-Brain Barrier Penetration (BBBP) dataset contains compounds and their classification labels based on their ability to penetrate the blood-brain barrier (BBB). The blood-brain barrier, a membrane that separates circulating blood from brain extracellular fluid, inhibits the majority of medications, hormones, and neurotransmitters. As a result, the penetration of the barrier has long been a problem in the development of medications that target the central nervous system. The dataset consists of 2039 compounds.

3.1.3. ClinTox Task

Clinical Trial Toxicity (ClinTox) dataset contains compounds that have not been approved and approved by the Food and Drug Administration (FDA) due to toxicity. The dataset contains 2 classification tasks. In the experiments, the clinical toxicity (CT_TOX) task [39] was used. The dataset consists of 1478 compounds.

3.1.4. SIDER Task

The Side Effect Resource (SIDER) dataset contains compounds that both are marketed and their adverse drug reactions (ADR). The dataset contains 27 classification tasks. In the experiments, the hepatobiliary disorders task [39] was used. The dataset consists of 1427 compounds.

3.1.5. Tox21 Task

Toxicology in the 21st Century (Tox21) dataset contains information on the toxicity of compounds according to different toxicity criteria. The dataset contains 12 classification tasks. In the experiments, the p53 stress-response pathway activation (SR_p53) [39] task was used to determine the toxicity labels of the compounds. The dataset consists of 7831 compounds.

3.1.6. ESOL Task

Estimated SOLubility (ESOL) dataset contains solubility abilities of the compounds. The dataset consists of 1128 compounds.

3.1.7. FreeSolv Task

The Free Solvation (FreeSolv) dataset contains experimental and calculated hydration free energies of compounds in water. The obtained numbers were from molecular dynamics simulations of alchemical free energy calculations. The dataset consists of 642 compounds.

3.1.8. Lipophilicity Task

Lipophilicity dataset contains experimental results of $n - octanol/water$ distribution coefficient ($\log D$ at pH 7.4) of compounds which affects both membrane permeability and solubility. The dataset consists of 4200 compounds.

3.2. Tokenization of SMILES Representations

We model compounds as documents derived from a chemical language by using the representations of the Simplified Molecular Input Line Entry System (SMILES) [40].

Algorithm from [41,42] is used for the character-based segmentation of the SMILES representations. With this algoritm; every atom except the ones in salt structures within the molecules, every covalent bond except single bonds, every salt structure within the molecules and every molecular branching are represented as different molecular fragments (tokens). Then, by repeating this procedure through all the benchmark tasks, a fragment dictionary is obtained. By assigning integers to every fragment in this dictionary which includes 194 distinct fragments, SMILES representations are encoded

into a vector that contains only integers for the purpose of use to build tree structure later. An example for both positioning of tokens in tree structure that is explained in detail in **Section 3.4** and tokenization can be seen in **Figure 3.1**.

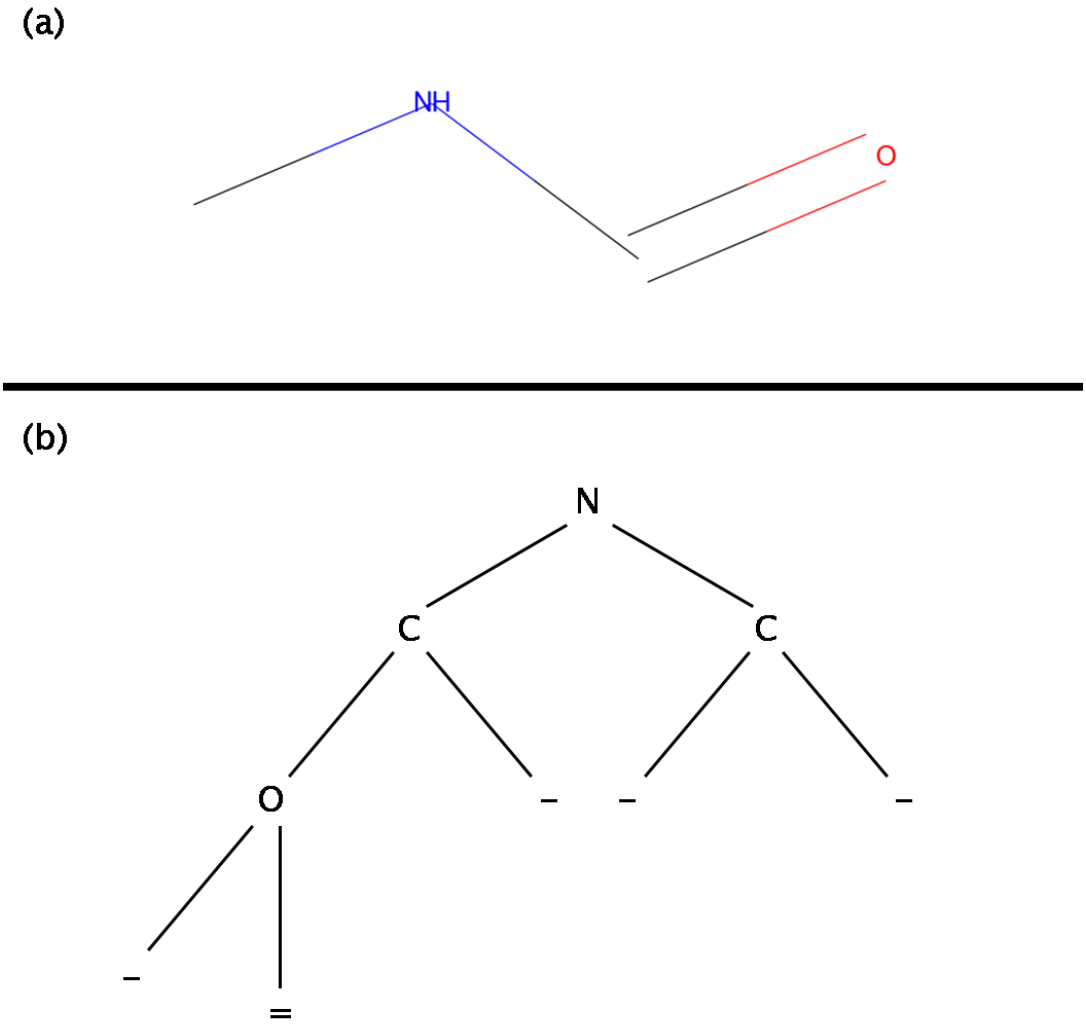
3.3. Experimental Setup

Python3 software [43] is used for all the programming related jobs and PyTorch library [44] is used for all the artificial intelligence related jobs in this work. First of all, the mentioned datasets are downloaded as their "scaffold" splits via the DeepChem [45] library. To read and process the datasets those are saved in comma-separated values (CSV) format, Pandas library is used [46].

The scripts of AR-Tree model is obtained from [4] and modified to meet the requirements of the changes due to input difference as it is not a normal sentence any more but a SMILES representation. There are two sub-model of the AR-Tree model which are the versions of it with single-input and double-input. In the context of this work, the only inputs are SMILE representations, the version with single-input is chosen. AR-Tree model also give an choice opportunity between RL and STG and between these two, STG is preferred due to some inconsistencies while summing the main loss with the RL loss to calculate the total loss.

During the training sessions, different hyperparameter combinations are tested to find the best one of them. Firstly, to find the hyperparameters that affect the benchmark results, all hyperparameters are tested individually with different values. Then, the hyperparameters that affect the benchmark results are combined with each other and the originated combinations are tested. Lastly, the best combination among them is used for the model that is trained from scratch again for one last time to obtain the best possible checkpoints of the model. Number of maximum epochs is chosen as 500 and early-stopping is used for all the trainings. Structure of the model is explained in detail in the next section (**Section 3.4**). An abstraction of the model can be seen in **Figure 3.2**.

Figure 3.1. Tree Representation of a Molecule.



2D-structured chemical representation and tree-structured representation of the molecule as known as N-methylformamide are shown in **Figure 3.1a and b**, respectively. Note that, SMILES representation of the molecule is **CNC = 0**. - indicates that there is no node.

The "TRUBA-akya-cuda" server that is provided by TÜBİTAK mentioned in **Acknowledgements** is used for the training sessions of the model. The mentioned server has 4 x NVIDIA Tesla V100 GPUs with 16 GB VRAM and 2 x 20-core Intel Xeon Scalable 6148 processors as specifications in brief. But it is necessary to note that -as different from the others- while values lesser than 1500 for the number of neurons in Tree-LSTM layers affect the length of the training time almost negligible, values greater than 1500 increase it relatively exponentially.

The best model checkpoints of the tasks obtained during training session are used for the evaluation of the models' benchmarks. The binary cross-entropy (BCE) loss function is employed as the training objective for the classification tasks. To evaluate the performance of the model, we used the Area Under the Receiver Operating Characteristic Curve (ROC-AUC) as the metric. The root-mean-square error (RMSE) loss function is employed as both the training objective and the performance metric for the regression tasks. The obtained results are analyzed in detail in **Chapter 4**.

Same with the evaluation of the models' benchmarks, also for the formation and scoring of fragments which are explained in detail in **Section 3.5 and 3.6**, the best model checkpoints of the tasks obtained during training session are used. PubChemPy library [47] is used to identify with Chemical Identification Number (CID) of PubChem database [48] numbers and RDKit library [49] is used to visualize the top 20 fragments among all the scored fragments. Finally, the obtained fragments had examined whether they are chemically meaningful in **Chapter 5**.

Table 3.2. Tried hyperparameters and their values.

Hyperparameter	Values
optimizer	adadelta, adam, adagrad
dropout ratio	0.1, 0.3, 0.5, 0.7, 0.8, 0.9
# of DNN layers	1, 2, 3, 5, 7, 10
# of nrns in DNN	8, 16, 32, 64, 128, 256, 512, 1024, 2048, 3072, 4096
learning rate	1e-3, 1e-4, 1e-5
# of nrns in TBL	50, 100, 200, 300, 500, 1000, 1500
use of batchnorm	1, 0
rank of input	w, h
# of epochs	500

nrns, DNN and TBL indicate neurons, Dense Neural Network and Tree-Bidirectional-LSTM, respectively.

3.4. Attentive Recursive Tree Model

An input sentence S of N words is represented as $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$, where $\mathbf{x_i}$ is a word embedding vector in the D_x-dimensions. An *Attentive Recursive Tree* (AR-Tree) is constructed basicly as a binary tree for each phrase, with R and T standing for the root and the tree's itself, respectively. Each node $t \in T$ has two children marked by $t.left \in T$ and $t.right \in T$ (nil for missing cases) and one word denoted by $t.index$ ($t.index = i$ means the i-th word of input sentence). The $in-order$ traversal of T corresponding to S (i.e., the index of each node in the left subtree of t must be smaller than $t.index$) is ensured to preserve the crucial sequential information. The AR-Tree's most notable characteristic is that words with more task-specific information are located closer to the root.

In order to accomplish the property, a scoring function that evaluates the relative significance of words and top-down recursively selects the word with the highest score is created. A modified Tree-LSTM is used to embed the nodes bottom-up, or from leaf to root, in order to produce the sentence embedding. The downstream tasks use the

resulting sentence embedding. Abstract of the model can be seen in **Figure 3.2**.

3.4.1. Top-Down AR-Tree Formation

A bidirectional LSTM is used to process the input phrase and produce a context-sensitive hidden vector for each word:

$$\overrightarrow{\mathbf{h_i}}, \overrightarrow{\mathbf{c_i}} = \overrightarrow{\mathbf{LSTM}}(\mathbf{x_i}, \overrightarrow{\mathbf{h_{i-1}}}, \overrightarrow{\mathbf{c_{i-1}}}) \tag{3.1}$$

$$\overleftarrow{\mathbf{h_i}}, \overleftarrow{\mathbf{c_i}} = \overleftarrow{\mathbf{LSTM}}(\mathbf{x_i}, \overleftarrow{\mathbf{h_{i+1}}}\overleftarrow{\mathbf{c_{i+1}}}), \tag{3.2}$$

$$\mathbf{h_i} = [\overrightarrow{\mathbf{h_i}}; \overleftarrow{\mathbf{h_i}}] \tag{3.3}$$

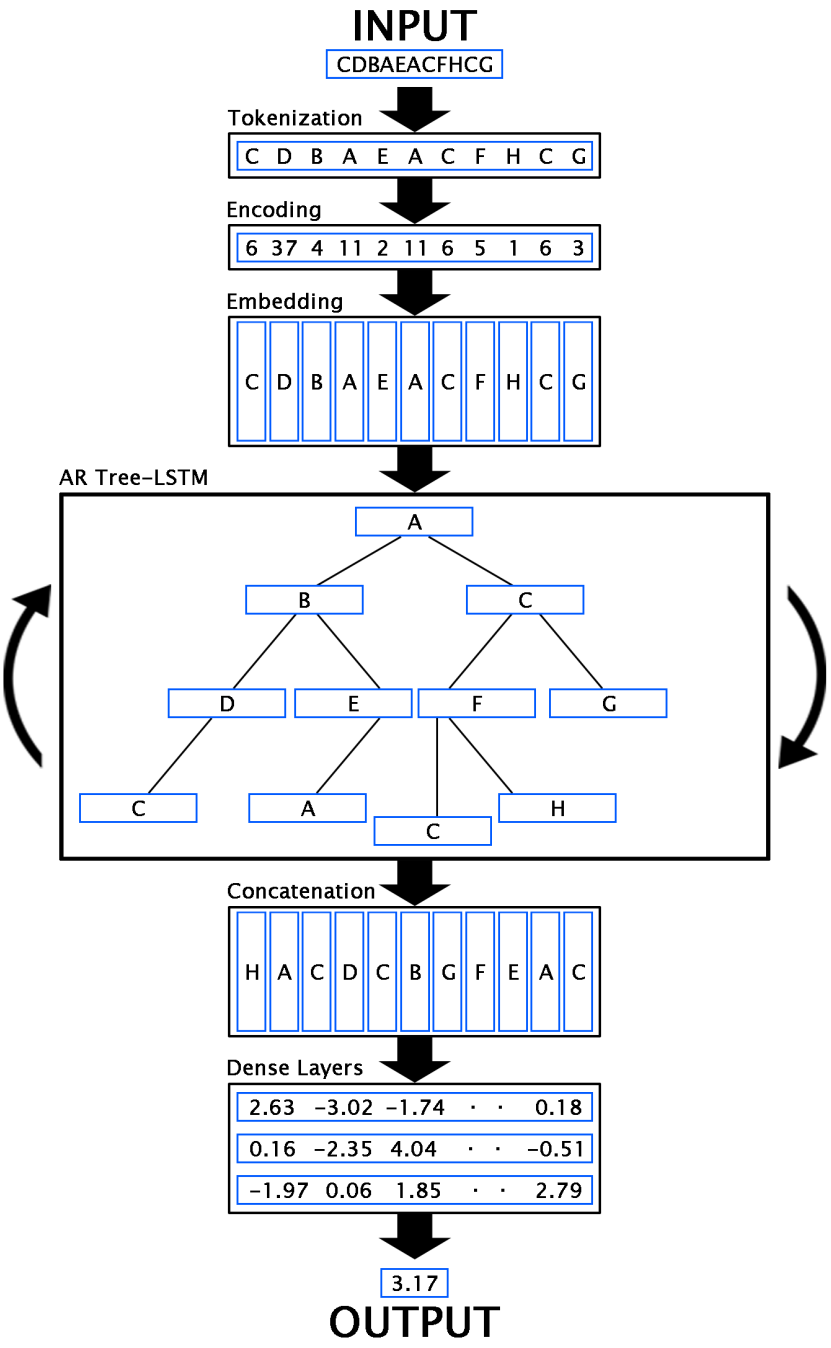
$$\mathbf{c_i} = [\overrightarrow{\mathbf{c_i}}; \overleftarrow{\mathbf{c_i}}] \tag{3.4}$$

where $\mathbf{h}$ and $\mathbf{c}$ indicate the hidden states and the cell states, respectively. $\mathbf{h}$ is used to score and leave $S = \{\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_N}\}$ alone. A trainable scoring function is created based on these context-aware word embeddings to account for the significance of each word:

$$Score(\mathbf{h_i}) = \mathbf{MLP}(\mathbf{h_i}; \theta) \tag{3.5}$$

where MLP is any multi-layer perceptron that has been parameterized by θ. A 2-layer MLP with 128 hidden units and ReLU activation are employed in particular. Traditional Term Frequency - Inverse Document Frequency (TF-IDF) is a straightforward and obvious way to express the value of words, but it is not intended for certain jobs. It will serve as the starting point.

Figure 3.2. Abstract of Attentive Recursive Tree.



AR indicates Attentive Recursive. Blue rectangulars represent layers or vectors as realistic as possible. **h** vectors of LSTM layers are concatenated in the "Concatenation" box and LSTMs are bidirectional. Output can be a classification or regression task value, here regression is prefered.

Figure 3.3. Pseudo-code of Attentive Recursive Tree Architecture.

Input: Sentence hidden vectors $S = \{\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_N}\}$, beginning index b and ending index e

Output: root node $R_{S[b:e]}$ of sequence $S[b:e]$

procedure BUILD(S, b, e)

$R \leftarrow nil$

if $e = b$ **then**

$R \leftarrow$ new Node

$R.index \leftarrow b$

$R.left, R.right \leftarrow nil, nil$

else if $e > b$ **then**

$R \leftarrow$ new Node

$R.index \leftarrow argmax_{i=b}^{e} Score(\mathbf{h_i})$

$R.left \leftarrow$ BUILD$(S, b, R.index - 1)$

$R.right \leftarrow$ BUILD$(S, R.index + 1, e)$

end if

return R

end procedure

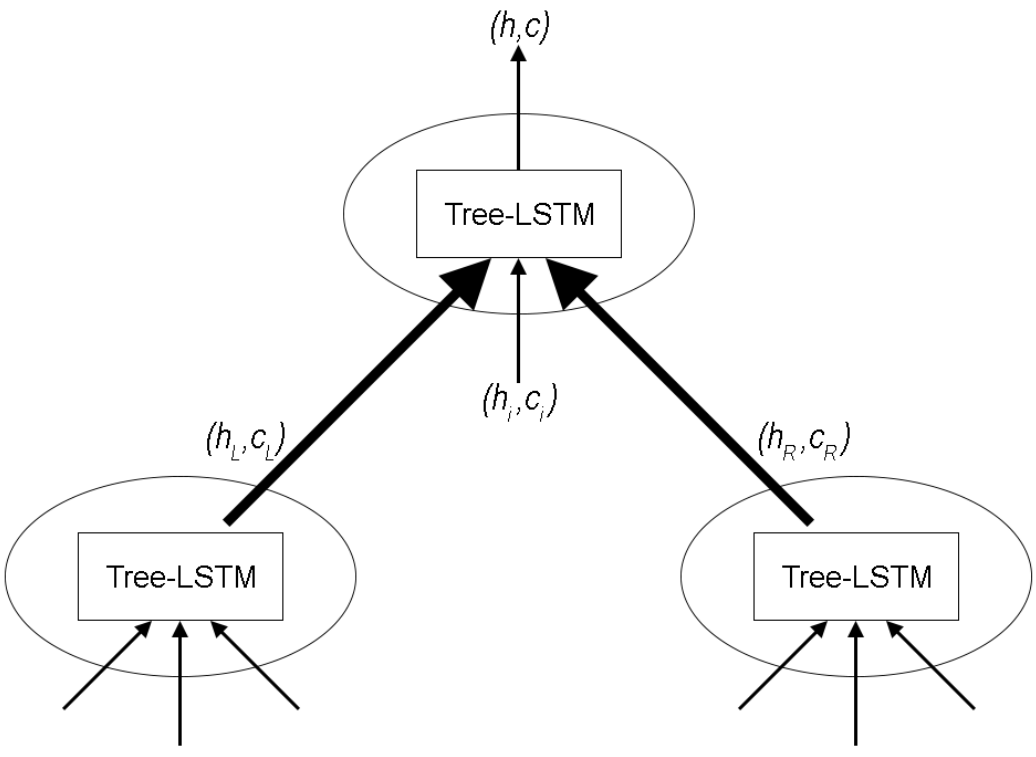
To build the AR-Tree, a recursive top-down attention-first method is used. Given an input phrase S and the scores for each word, The word with the highest score is chosen as the root R. Then, using recursion, the two subsequences that come before and after the R is used to get the two offspring of the root. The general algorithm for creating the AR-Tree for the sequence $S[b:e] = \{\mathbf{h_b}, \mathbf{h_{b+1}}, \ldots, \mathbf{h_e}\}$ is provided in **Figure 3.3**. By invoking $R =$ BUILD$(S, 1, N)$, the whole sentence's AR-Tree and T can be gotten by traversing all of the nodes. Each node in the parsed AR-Tree is the most insightful among its rooted subtree. Because of the fact that any additional data

is not utilized in the creation, AR-Tree is applicable to any tasks requiring sentence embedding.

3.4.2. Bottom-Up Tree-LSTM Embedding

After building the AR-Tree, Tree-LSTM [50, 51] is utilized as the composition function to calculate the parent representation from its children and corresponding word in a bottom-up fashion in **Figure 3.4**. Tree-LSTM inserts cell state into Tree-RNNs to promote improved information flow. Tree-LSTM units may use both the sequential and the structural information to compose semantics since the original word sequence is maintained throughout the in-order traversal of the AR-Tree.

Figure 3.4. Bottom-Up Embedding.



The following describes the whole Tree-LSTM composition function in the model:

$$\begin{bmatrix} \mathbf{ig} \\ \mathbf{f_L} \\ \mathbf{f_R} \\ \mathbf{f_i} \\ \mathbf{o} \\ \mathbf{g} \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} \left(\mathbf{W_c} \begin{bmatrix} \mathbf{h_L} \\ \mathbf{h_R} \\ \mathbf{h_i} \end{bmatrix} + \mathbf{b_c} \right) \tag{3.6}$$

$$\mathbf{c} = (\mathbf{f_L} \odot \mathbf{c_L}) + (\mathbf{f_R} \odot \mathbf{c_R}) + (\mathbf{f_i} \odot \mathbf{c_i}) + (\mathbf{i} \odot \mathbf{g}) \tag{3.7}$$

$$\mathbf{h} = \mathbf{o} \odot tanh(\mathbf{c}) \tag{3.8}$$

ig, $\mathbf{f_L}$, $\mathbf{f_R}$, $\mathbf{f_i}$, **o**, **g**, σ and *tanh* indicate the input gate, the gate of the left child node, the gate of the right child node, the gate of the parent node, the output gate, the candidate vector, the sigmoid function and the hyperbolic tangent function, respectively.

While $\mathbf{f_L}$ and $\mathbf{f_R}$ gates control the cell state of the left and right child nodes, $\mathbf{f_i}$ is responsible for adding new information to the current node's cell state. **ig** determines the extent to which the cell state will be updated when adding new information. **o** converts the embedded representation obtained from the cell state into an output representation. **g** is used to update the cell state. σ squeezes values between 0 and 1 which is an activation function commonly used in neural networks. And *tanh* squeezes values between -1 and 1 which is also another activation function commonly used in

neural networks.

To create the node embedding $(\mathbf{h}, \mathbf{c})$, the Tree-LSTM unit combines the semantics of the current word $(\mathbf{h_i}, \mathbf{c_i})$, the right child $(\mathbf{h_R}, \mathbf{c_R})$, and the left child $(\mathbf{h_L}, \mathbf{c_L})$. Zeros are substituted for the missing inputs for nodes that lack certain inputs, such as leaf nodes or nodes with just one child.

Finally, the phrase S is fed onto tasks farther down the line using the embedding $\mathbf{h}$ of the R. Because they are closer to the R and their semantics is naturally highlighted, the sentence embedding will concentrate on those informative terms.

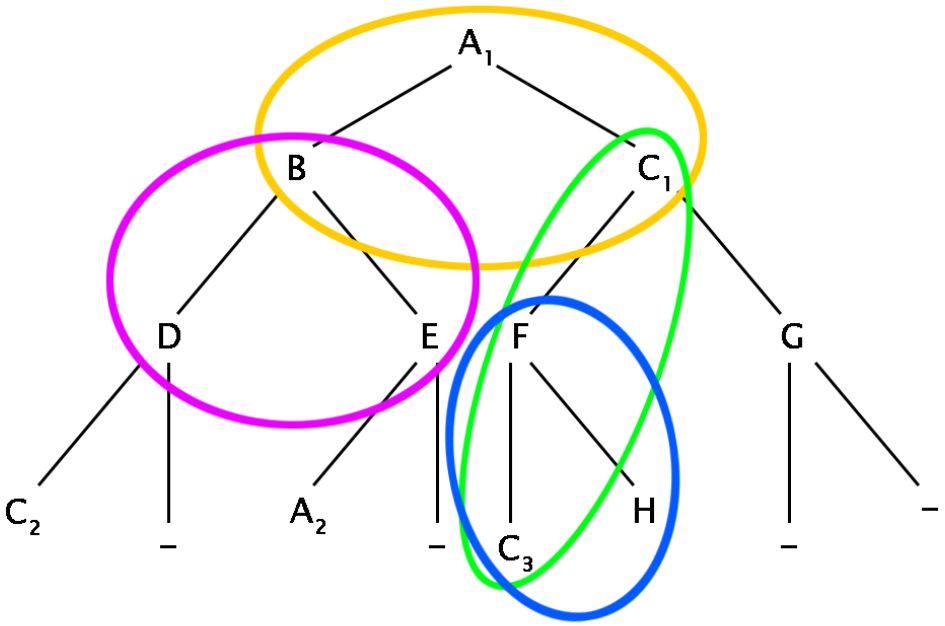
3.5. Creating Dynamic Fragment Dictionary

Firstly, all the possible subtree formations (fragments) should be found in all molecule trees through the corresponding datasets. It is important to find relatively large-sized fragments rather than small fragments (i.e., the ones that consist only 2 or 3 atoms except hydrogen atoms). Because larger-sized fragments are more interpretable and can be more chemically meaningful.

The only possible way of this is to form the fragments from leafs to root or bottom-up. Although root is more discriminative than other nodes, atoms of the subtree structures those are formed from root to leafs cannot be found side-by-side in SMILES representations of the corresponding molecules as it can be seen in **Figure 3.5**. So basicly, this structures are chemically not valid since the atoms of the chemical fragments which are presented as nodes are not connected to each other. That applies to the subtree structures those are formed between leafs and root as well. That's why, the only way to obtain chemically valid subtree structures (fragments) is to form fragments from leafs to root. The formation procedure is explained digestedly in **Figure 3.6**.

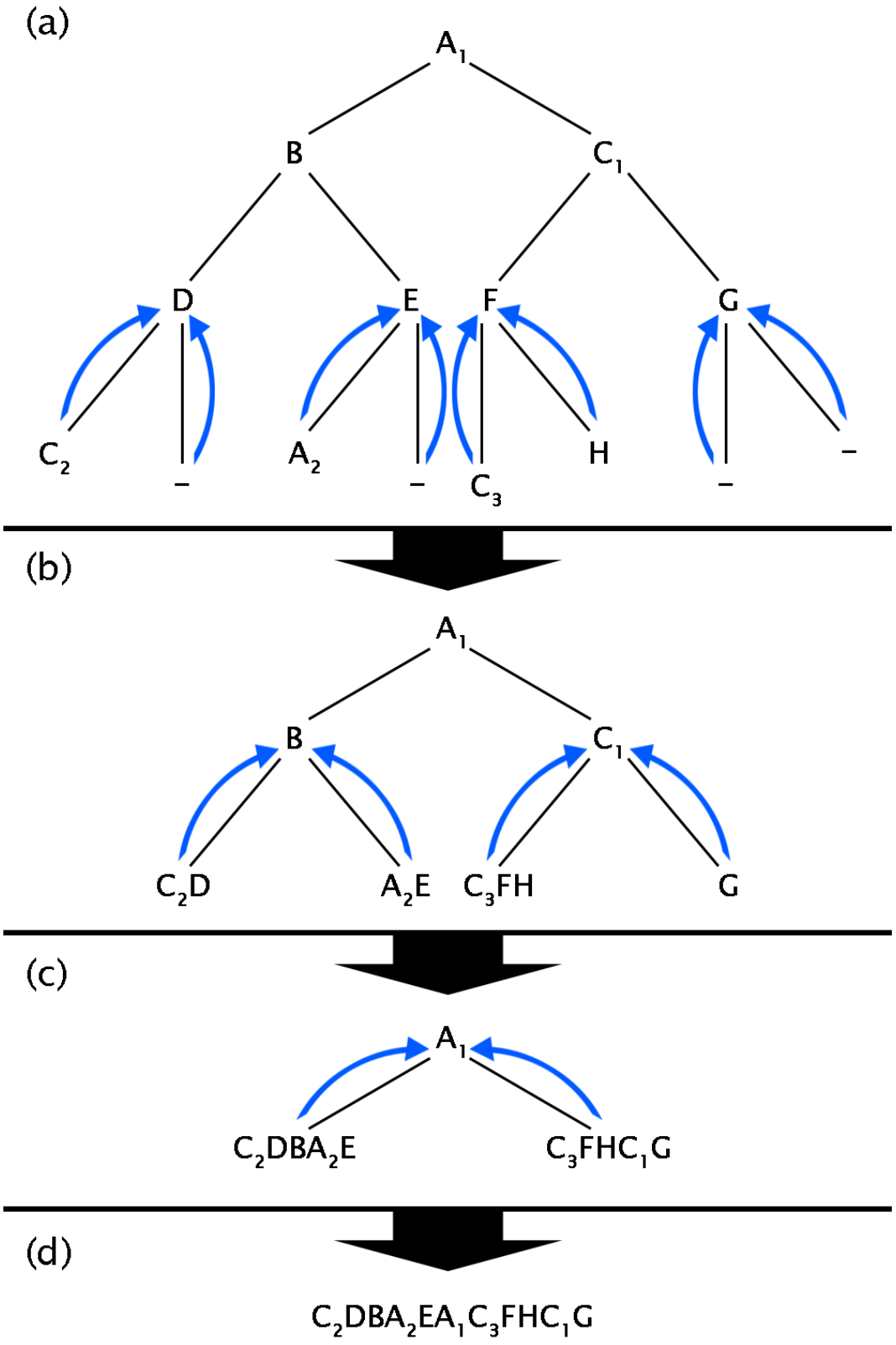
With obtained fragments, a dynamic fragment dictionary (DFD) is constituted which is specifically dependant to the its corresponding dataset. Then, some of the

Figure 3.5. Different Subtree Locations on a Tree.



This tree is exact same with the tree in **Figure 3.6a** and represents $\mathbf{C_2DBA_2EA_1C_3FHC_1G}$ sentence or in this case SMILES representation. Only the subtree that is showed in the blue ellipse can form a valid chemical fragment, others are not chemically valid, unfortunately.

Figure 3.6. Fragment Formation Procedure.



Subindexes are used to show that there can be same tokens at different nodes.

fragments are eliminated whether meet the chosen criterias. Also, after the first criteria, all the found fragments are canonicalized to obtain a better standardized representation for fragments. Criterias are shown below, respectively (**Figure 3.7**):

Figure 3.7. Fragment Elimination Criterias.

1 - The fragment must be pass the validation check that is created by using RDKit library. It is basicly a sanitizability test that checks whether there are open-rings, branches, illegal atom types etc. in the fragment. So that the fragment can be assumed both syntactically and chemically reasonable.

2 - While neglecting hydrogen atoms and assuming salt structures as one atom, total number of atoms in the fragment must be greater than or equal to 3. So that, more interpretable and chemically meaningful fragments can be focused for investigation.

3 -

For classification tasks,

Task label of the chemical that contains the fragment must be equal to 1. So, only the fragments those in the chemicals that have positive relation with its corresponding task can be focused to investigation.

For regression tasks,

Task label of the chemical that contains the fragment must be greater than or equal to the average value of the label set. So that, only the fragments those in the chemicals that have a positive relation with its corresponding task can be focused for investigation.

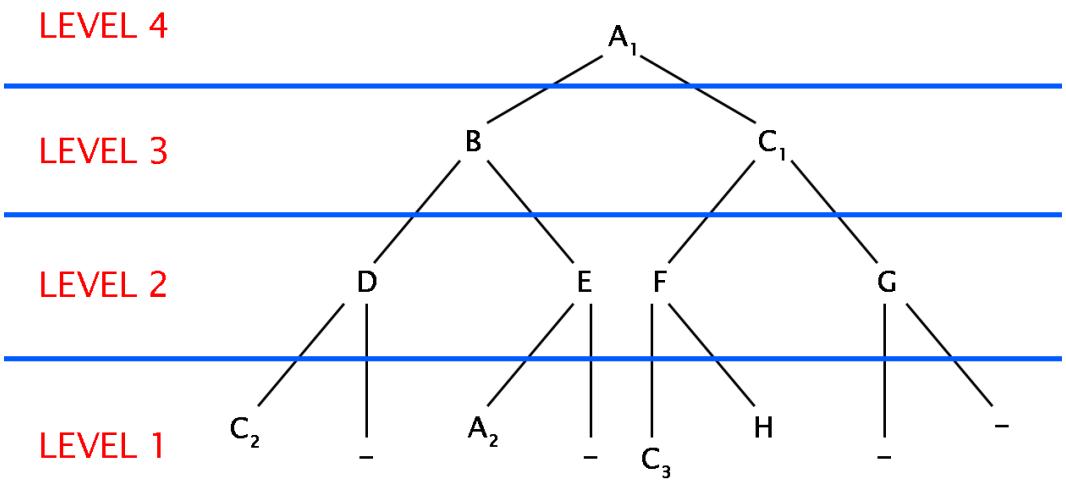
And finally, remained fragments in the DND are scored by using the scoring procedure in the next section (**Section 3.6**).

3.6. Scoring Procedure for Chemical Fragments

All the fragments (subtrees) those are found in the previous section (**Section 3.5**) should be searched in all the molecules in the dataset DS and rated using a scoring procedure in order to evaluate the model's interpretability.

In the scoring procedure, each leaf is first assigned to 1 point and each node is assigned to a point increasing by 1 point node-by-node from the leaves to the R in the molecule tree T as it can be seen in **Figure 3.8** where "level" indicates point level. In this way, the R receives the maximum point in the T.

Figure 3.8. Point Levels of a Tree for Scoring.



Then, each node's points ($\mathbf{P_i}$) is divided by the distance between the fartest leaf from the R and the R in the T ($\mathbf{d_{max}}$). So that all of the outcomes fall between 0 and 1.

$$\mathbf{P^{mol}} = \sum_{i=1}^{n} \frac{\mathbf{P_i}}{\mathbf{d_{max}}} \tag{3.9}$$

where $\mathbf{P^{mol}}$ and $\mathbf{n}$ indicate total score of the fragment F in the T and the total

repeat count of the F in the T, respectively.

Not only nodes' positions of the F in the T, but also the total repeat count of the F in the DS and the test loss of the T in the DS ($\mathbf{e_k}$) should be considered while scoring the fragments. Among the compounds in the DS, scores ($\mathbf{P^{mol}_j}$) of the F are added up. By adding up, the fragments that is more frequent have higher total score. After summation, the total score of the F in the DS is divided by $\mathbf{m}$. Thus, once more, all of the outcomes fall inside the range of 0 and 1.

$$\overline{\mathbf{P}} = \frac{1}{m} \sum_{j=1}^{m} \mathbf{P^{mol}_j} \tag{3.10}$$

where $\overline{\mathbf{P}}$ and $\mathbf{m}$ indicate the average score of the F in the DS and the total repeat count of the F in the DS, respectively. For including the $\mathbf{e_k}$ parameter also to general equation, **Equation 3.11** is employed and the average test loss of the F in the dataset DS ($\overline{\mathbf{e}}$) is calculated.

$$\overline{\mathbf{e}} = \frac{1}{L} \sum_{k=1}^{L} \mathbf{e_k} \tag{3.11}$$

where $\mathbf{L}$ indicates size of the DS.

After scoring all the fragments in the DND, all fragment scores' set FS are normalized. Normalization is done by using min-max normalization formula [52] which scales values in a set into between 0 and 1 (**Equation 3.12**).

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mathbf{x_{min}}}{\mathbf{x_{max}} - \mathbf{x_{min}}} \tag{3.12}$$

where $\mathbf{x}$, $\hat{\mathbf{x}}$, $\mathbf{x_{max}}$ and $\mathbf{x_{min}}$ indicate the unnormalized FS value, the normalized FS value, the maximum value of the FS and the minimum value of the FS.

$$\mathbf{P_F} = normalize(\overline{\mathbf{P}}\overline{\mathbf{e}}) \tag{3.13}$$

where $\mathbf{P_F}$ indicates final score of the F in the FS.

Finally, all fragments were sorted in descending order and the top 20 fragments were examined in detail in **Chapter 5**.

4. BENCHMARK RESULTS

The ROC-AUC and RMSE scores of our model and the state-of-the-art models (SOTA) [53–55] are shown in **Table 4.2 and 4.4**. The results show that our model outperformed the SOTA models in ClinTox task and achieved moderate scores in the other benchmark tasks. All the tried and the best hyperparameters are shown in **Table 3.2, 4.1 and 4.3**. Also, training and validation curves of the tasks can be found in **Appendix B**.

4.1. Results of Classification Tasks

Table 4.1. Best hyperparameters for classification tasks.

Hyperparameter	BACE Clf	BBBP	ClinTox	SIDER	Tox21
optimizer	adadelta	adadelta	adadelta	adagrad	adadelta
dropout ratio	0.3	0.3	0.5	0.9	0.5
# of DNN layers	1	5	5	1	3
# of nrns in DNN	6144	6144	2048	2048	2048
learning rate	1e-3	1e-3	1e-3	1e-3	1e-3
# of nrns in TBL	300	300	1500	500	100
use of batchnorm	1	1	0	0	1
rank of input	h	h	w	w	h
# of epochs	200	200	400	150	100

nrns, DNN and TBL indicate neurons, Dense Neural Network and Tree-Bidirectional-LSTM, respectively.

As it can be seen in **Table 4.1**, learning rate is same for all the classification tasks, respectively 0.001 and character-based. The optimizer is mostly “adadelta” except the SIDER task whose best optimizer is “adagrad”.

Table 4.2. Score table of classification tasks.

Model	BACE Clf	BBBP	ClinTox	SIDER	Tox21
AR-Tree	76.2	64.9	99.7	57.7	63.2
RF	86.7	71.4	71.3	68.4	76.9
SVM	86.2	72.9	66.9	68.2	81.8
GCN	71.6	71.8	62.5	53.6	70.9
GIN	70.1	65.8	58.0	57.3	74.0
SchNet	76.6	84.8	71.5	53.9	77.2
MGCN	73.4	85.0	63.4	55.2	70.7
D-MPNN	85.3	71.2	90.5	63.2	68.9
[56]	85.9	70.8	78.9	65.2	78.7
N-Gram	87.6	91.2	85.5	63.2	76.9
MolCLR$_{\mathbf{GCN}}$	78.8	73.8	86.7	66.9	74.7
MolCLR$_{\mathbf{GIN}}$	89.0	73.6	93.2	68.0	79.8
ChemBERTa-1	-	64.3	73.3	-	72.8
ChemBERTa-2	79.9	74.2	60.1	-	83.4
MoLFormer-XL	88.2	93.7	94.8	69.0	84.7

ROC-AUC is used as the performance metric and BCE is used as the loss function. The first row is our model's results, the results for the both version of ChemBERTa retrieved from [54], the results in the last row are retrieved from [55] and the rest are retrieved from [53]. Also, - indicates there is no result there in related paper. Note, the models that are in last 5 rows are pretrained models.

4.2. Results of Regression Tasks

Table 4.3. Best hyperparameters for regression tasks.

Hyperparameter	BACE Reg	ESOL	FreeSolv	Lipophilicity
optimizer	adagrad	adagrad	adagrad	adam
dropout ratio	0.8	0.3	0.3	0.3
# of DNN layers	1	3	1	1
# of nrns in DNN	3072	128	1024	512
learning rate	1e-3	1e-3	1e-3	1e-3
# of nrns in TBL	1500	50	1500	500
use of batchnorm	1	0	1	1
rank of input	w	h	h	h
# of epochs	100	200	100	150

nrns, DNN and TBL indicate neurons, Dense Neural Network and Tree-Bidirectional-LSTM, respectively.

As it can be seen in **Table 4.3**, learning rate is same for all the regression tasks, respectively 0.001 and character-based. The optimizer is mostly “adagrad” except the Lipophilicity task whose best optimizer is “adam”.

Table 4.4. Score table of regression tasks.

Model	BACE Reg	ESOL	FreeSolv	Lipophilicity
AR-Tree	1.02	0.45	0.77	0.71
RF	1.32*	1.07	2.03	0.88
SVM	-	1.50	3.14	0.82
GCN	1.65*	1.43	2.87	0.85
GIN	-	1.45	2.76	0.85
SchNet	-	1.05	3.22	0.91
MGCN	-	1.27	3.35	1.11
D-MPNN	2.25*	0.98	2.18	0.65
[56]	-	1.22	2.83	0.74
N-Gram	-	1.10	2.51	0.88
MolCLR$_{\mathbf{GCN}}$	-	1.16	2.39	0.78
MolCLR$_{\mathbf{GIN}}$	-	1.11	2.20	0.65
ChemBERTa-1	-	-	-	-
ChemBERTa-2	1.36	0.86	-	0.74
MoLFormer-XL	-	0.28	0.23	0.53

RMSE is used as both the performance metric and the loss function. The first row is our model's results, the results for the both version of ChemBERTa and the cells with * sign are retrieved from [54], the results in the last row are retrieved from [55] and the rest are retrieved from [53]. Also, - indicates there is no result there in related paper. Note, the models that are in last 5 rows are pretrained models.

5. INTERPRETABILITY

The obtained fragments from the model have been analyzed within the scope of their respective tasks. If the fragments were considered as significant fragments based on pharmaceutical chemistry perspective and supported by literature, they were referred to as "meaningful fragments". During the interpretation of the tasks, literature-based SAR, STR or physicochemical properties were taken into consideration to figure out whether the model provided truly meaningful fragments.

Firstly, SAR is the relationship between the three-dimensional (3D) structures of molecules and their biological activities. Based on the assumption that structurally similar compounds have similar physical and biological properties, minor changes are made to the structure of the lead molecule and their effects on biological activity are evaluated. Secondly, STR is the relationship between the structures of chemical compounds or chemical groups in chemical molecules and toxicity profiles. Thirdly, physicochemical properties are related to parameters such as water or lipid partition coefficient, polarity, reactivity, acidity or basicity etc.

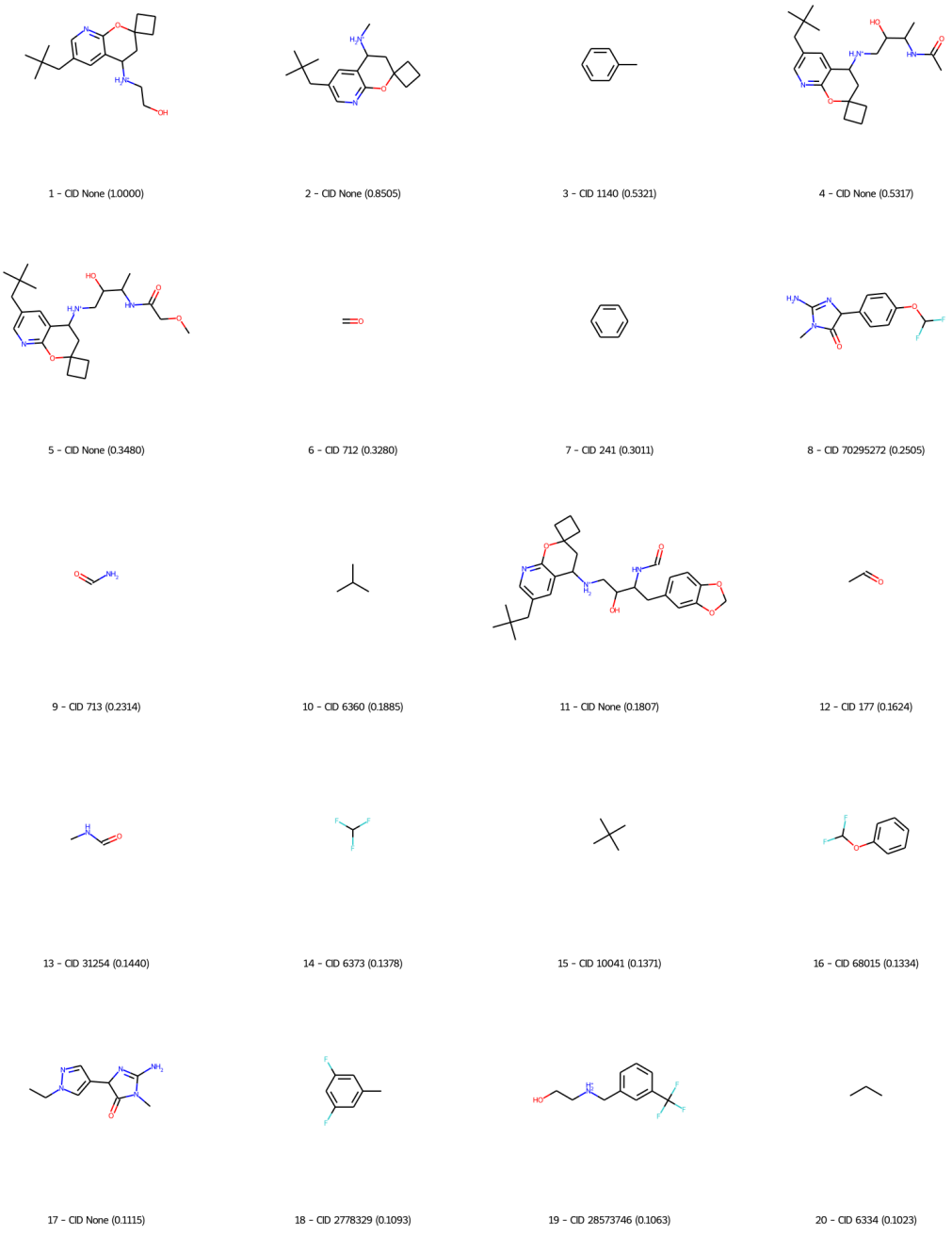
The BACE and BBBP tasks were analyzed using SAR studies available in the literature. The ClinTox and Tox21 tasks were examined using STR studies from the literature. The ESOL and Lipophilicity tasks, were investigated by taking into account the physicochemical properties of the molecules. The fragments, the names of the corresponding compounds, and the results of the analyses for different tasks are presented in the following subsections. According to the model, the colored fragments were determined to be the most meaningful fragments for the corresponding tasks.

The mentioned fragments can be seen in the images those are in its corresponding subsection below. Fragments in the images were sorted in descending order from left to right according to their fragment scores. Best fragment at the top left and the worst one at the bottom right. Also, detailed information about the fragments can be found

in **Appendix C**.

5.1. BACE Classification Analysis

Figure 5.1. Top 20 Fragments for BACE Classification task.



BACE1 is a protease enzyme connected to the production of β-amyloid (Aβ) peptides, which are thought to be the root of Alzheimer's disease. As a result, one of

the key approaches for creating drugs to treat Alzheimer's disease is to inhibit BACE1. The catalytic dyad of aspartic acid (Asp) residues which are Asp32 and Asp228 found in the active site of BACE1 are necessary for the enzyme's function. Therefore, the catalytic dyad is typically interacted with by inhibitors that bind to the catalytic site of BACE1. Chemical groups that can form electrostatic and hydrogen bonds with the enzyme, as well as polar and charged groups that can bind to the enzyme's hydrophobic pockets, are all crucial for binding to BACE1 [57].

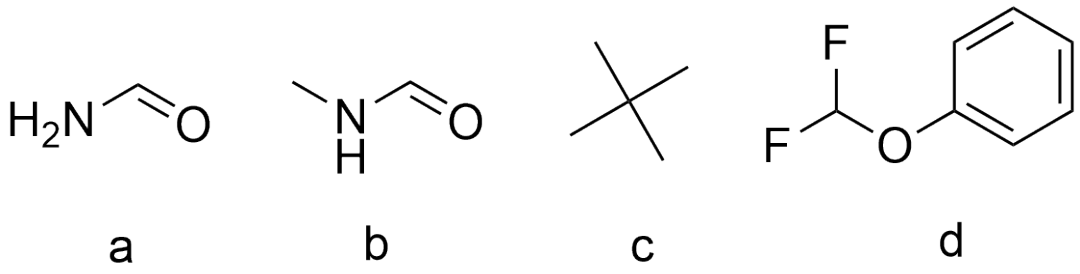
In the **Figure 5.1**, the structures ranked 1, 2, 4, 5 and 11 share the same structural core, 2-spirocyclobutyl-6-neopentyl-8-azachromane, which has been demonstrated in **Figure 5.2**, respectively with letters. The hydroxyethylamine (HEA) scaffold, which is present in all of these studies and is depicted in blue in **Figure 5.2**, is a feature that unites them and is thought to be crucial for the formation of hydrogen bonds (H-bonds) with the aspartic catalytic dyad of BACE1. Fragments a, c, d and e all display the HEA scaffold indicated in blue, whereas fragment b only carries the methylamino portion of the HEA moiety. It is also reported in all three of these works that the azachromane (shown in yellow in **Figure 5.2**) and the spirocyclobutane (shown in pink in **Figure 5.2**) rings occupying the S1′ subpocket along with the neopentyl group (shown in green in **Figure 5.2**) filling the S2′ subpocket, all increase the binding affinity to BACE1 and therefore are essential structural components of BACE1 inhibitors. It is encouraging that the algorithm was able to keep these structures as significant fragments and give them high scores by giving them higher significance ranks because it perfectly matches the results of these studies.

Additionally, it should be noted that the algorithm extracted a positive charge from the nitrogen atom in the HEA group, which, according to the literature, can be attributed to the fact that this amine forms an H-bond with the catalytic aspartic residues in BACE1. The protonation of this secondary amine group in physiological pH is an alternative explanation for this.

Additionally, according to [58], the oral bioavailability and metabolic stability of

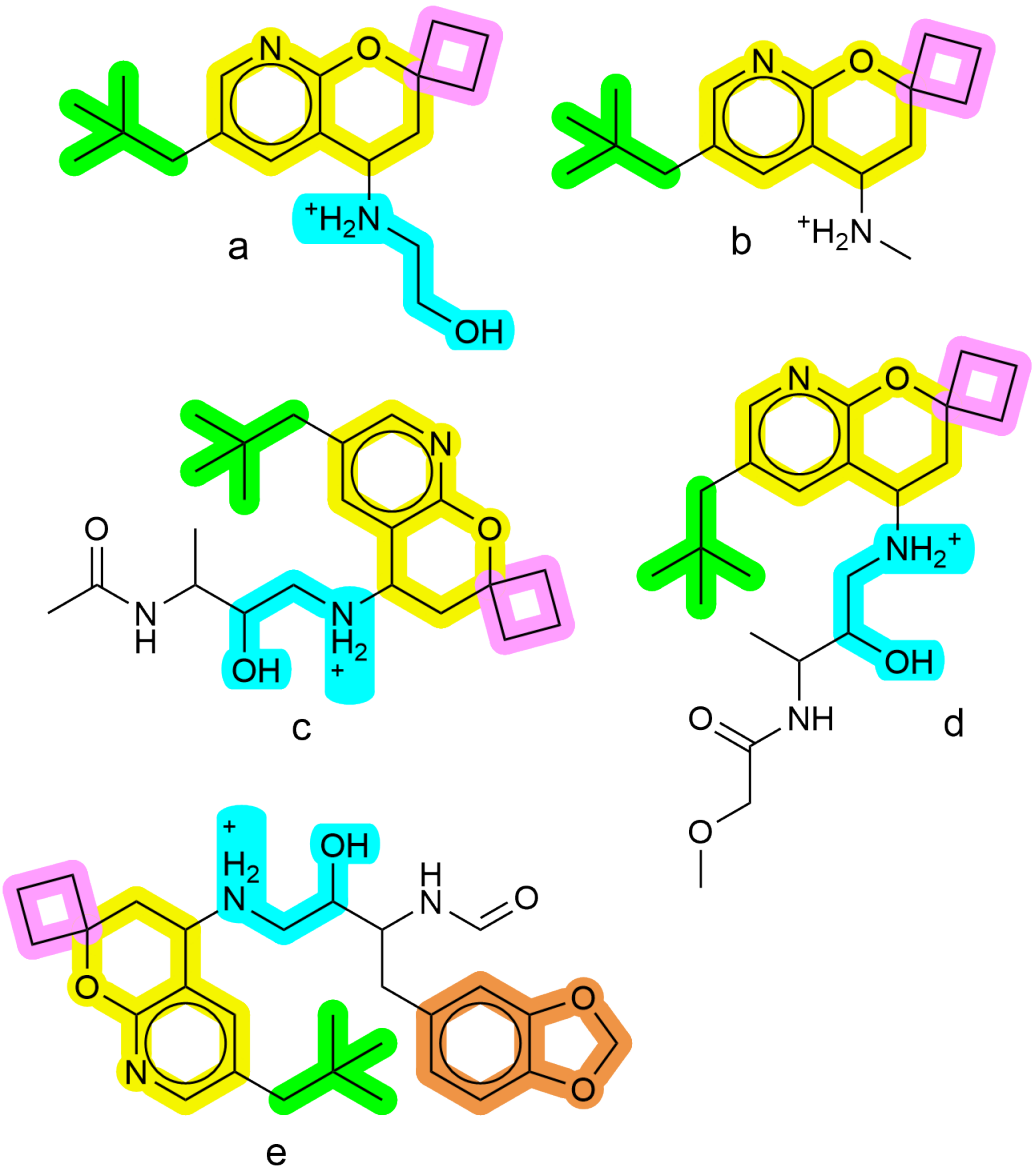
the previously developed compounds were improved by the introduction of the benzodioxolane group (shown in orange in **Figure 5.2e**). The algorithm was successful in extracting this fragment, which has characteristics that are appropriate for rational drug design.

Figure 5.2. BACE Classification Analysis Example 1.



There are several fragments that contains aminohydantoinsand and iminohydantoin tautomers in **Figure 5.1**. This group of fragments is present in many BACE1 inhibitors that have been reported in the literature, either in their imidazole-4-one (aminohydantoin) form (shown in blue in **Figure 5.3a and c**) [59, 60] or in their respective iminohydantoin tautomeric form (shown in yellow in **Figure 5.3b**) [61]. Tautomerism is a chemical phenomenon in which two structural isomers easily interconvert to one another due to the movement of a hydrogen atom within the molecule. **Figure 5.3** shows the schematic for this imine-amine tautomerism. The scoring system placed fragments a and c, respectively in ranks 8 and 17 in **Figure 5.1**. The catalytic residues Asp32 and Asp228 at the active site of BACE1 form an H-bond complex with the amidine moiety of the iminohydantoin compounds (shown in orange in **Figure 5.2b**). According to the aforementioned studies, analogs with either aminohydantoin or iminohydantoin moieties have been shown to have higher oral bioavailability and BACE1 selectivity. They have also been shown to be more brain-penetrable.

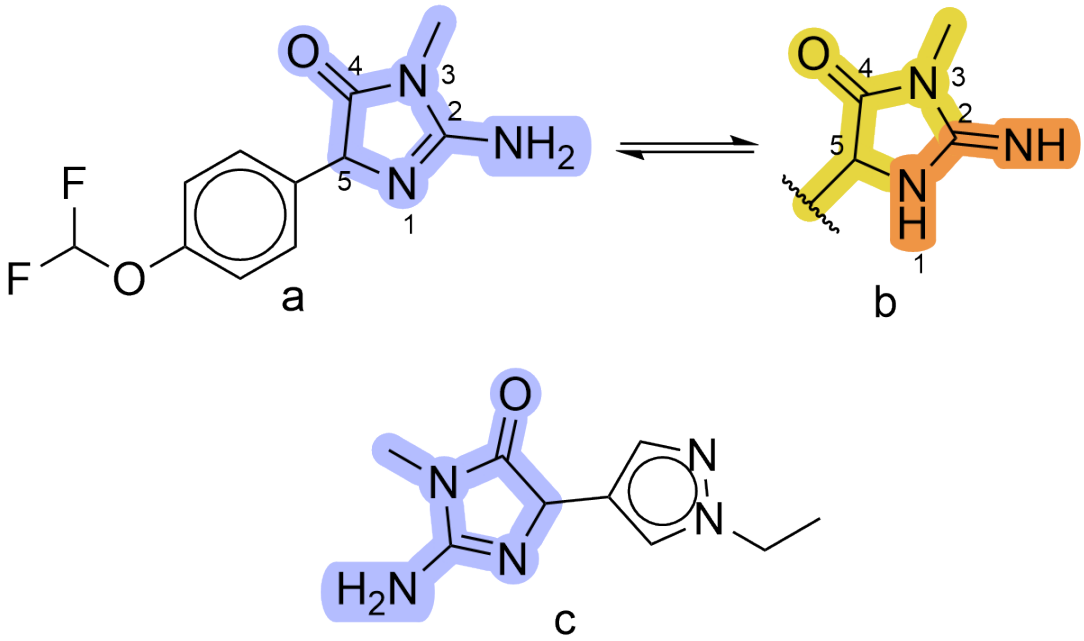
Figure 5.3. BACE Classification Analysis Example 2.



As seen in **Figure 5.4**, the algorithm also identified fragments a, b, c and d as significant language units, with each having ranks of 9, 13, 15 and 16 in **Figure 5.1**. It is possible to link fragments a and b to the terminal acetamide group shown in **Figure 5.2**. According to [62], this acetamide group was created by shortening a longer amide group in order to maximize the molecular weight, the effectiveness of binding, and the permeability of previously synthesized undesirable compounds to the central nervous system (CNS). The algorithm also extracted fragment c, which represents the neopentyl group shown in **Figure 5.2**, separately as an important moiety. In [59],

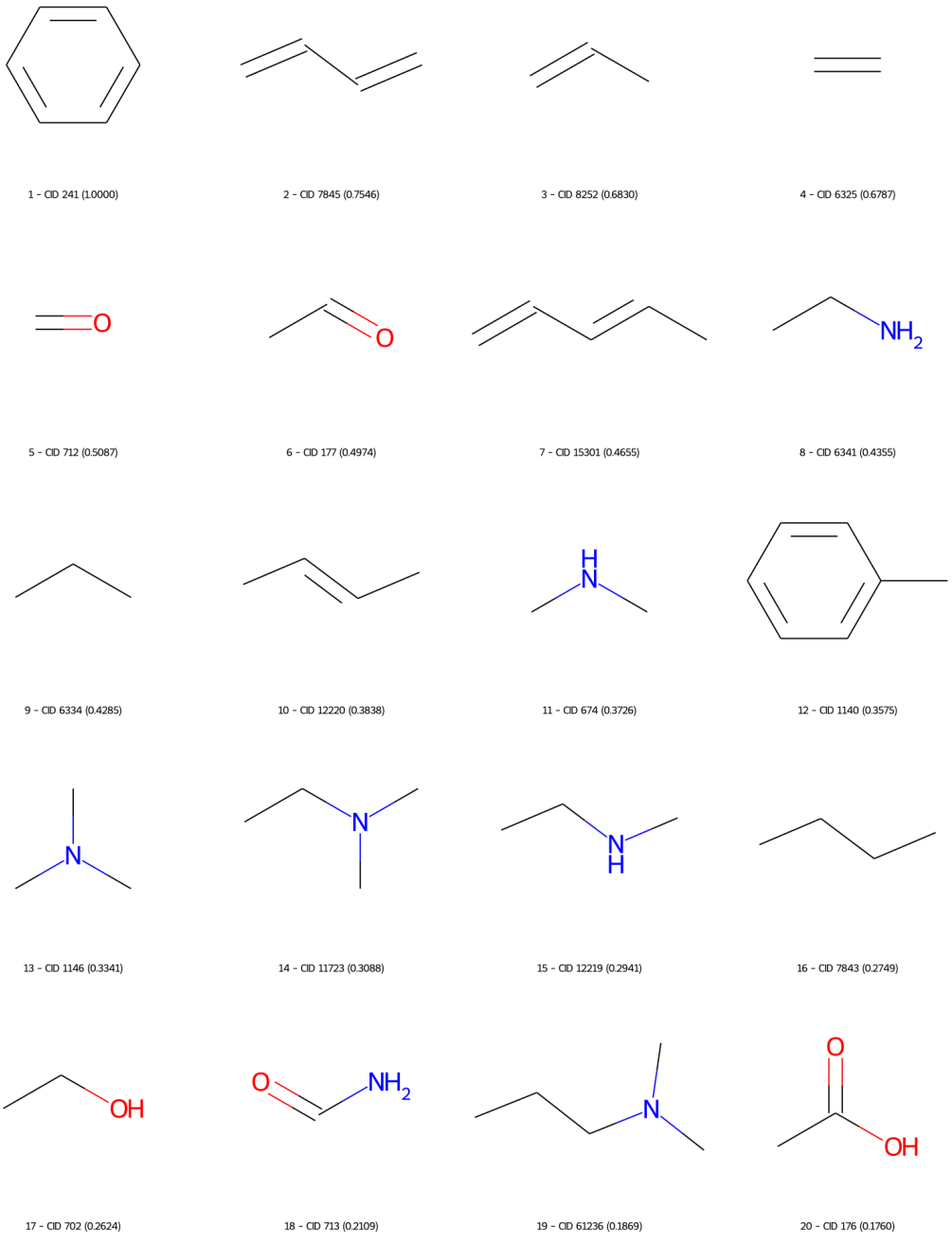
fragment d designating the difluoromethoxy phenyl moiety is claimed to have been added to aminohydantoin derivatives **Figure 5.3a** to provide more potent compounds. This moiety is found to be located in BACE1's S2′ region.

Figure 5.4. BACE Classification Analysis Example 3.



5.2. BBBP Analysis

Figure 5.5. Top 20 Fragments for BBBP task.



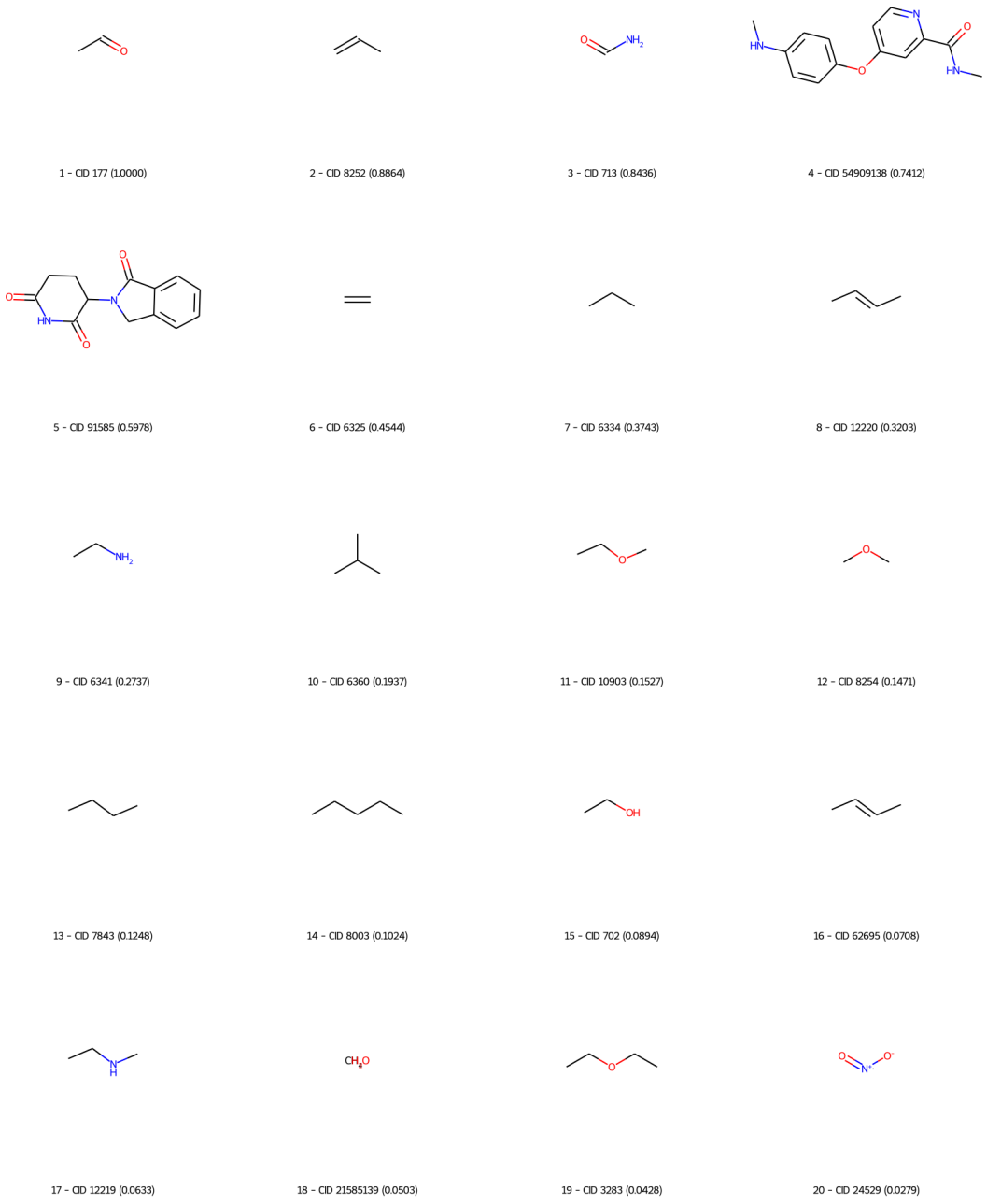
The BBBP dataset contains the chemicals that are able to cross the BBB. The algorithm tends to extract smaller chemical groups as fragments from this task that can be seen in **Figure 5.5**, which may not necessarily be directly related to the larger molecule's BBB penetrability from which the fragment was extracted. In other words,

these particular smaller fragments might not directly improve or worsen the molecule's BBB permeability, as in some cases, that particular fragment might decrease lipophilicity and increase BBBP, whereas that same fragment might increase lipophilicity and BBBP when found on a different molecular skeleton. It would be sensible to investigate the lipophilicity of each molecule containing a specific fragment according to that molecule only and refrain from making general judgments based on the fragments because the given fragments are not large enough chemical moieties to assign accurate qualities to a specific fragment in terms of being able to change the main molecule's BBBP.

However, based on fundamental chemical knowledge, ranks 1 and 12 (benzene and toluene moieties) are known to increase the lipophilicity of molecules. The same is true for tertiary amines (ranks 13, 14 and 19), as the decrease in the number of hydrogen atoms bound to the nitrogen atom of an amine would result in a decrease in that amine's ability to form H-bonds, which in turn would result in a decrease in lipophilicity, putting tertiary amines above secondary amines and above primary amines in the hierarchy of lipophilicity. This particular instance deviates from the usual order, with rank 15 (a secondary amine) coming after rank 14 (a tertiary amine). This is as a result of the scoring mechanism used here, which is not based on chemical rules and features of each dataset but solely on methodological approaches intended to make the algorithm extract the most repeated meaningful words. Furthermore, fragments ranks 17, 18 and 20 are undesirable fragments when designing molecules that cross the BBB because, due to their capacity to form H-bonds, they typically decrease a molecule's lipophilicity rather than increasing it [63].

5.3. ClinTox Analysis

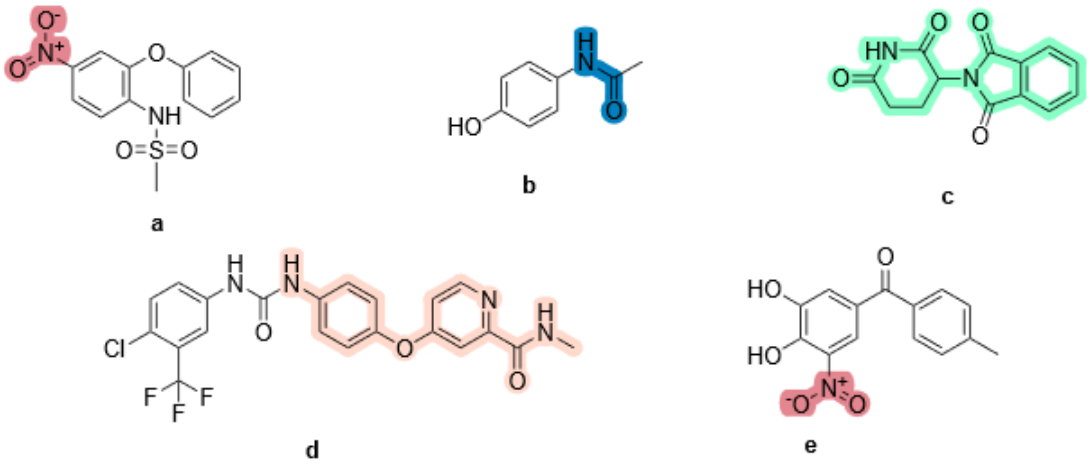
Figure 5.6. Top 20 Fragments for ClinTox task.



Drugs may fail to pass clinical trials due to toxicity. One of the reasons for this situation is the toxic metabolites formed after the biotransformation of drugs. There are some functional groups that may contribute to the formation of toxic metabolites. For example, nitro groups, aromatic amines, polyhalogenated groups etc. Some of the

fragments found by the algorithm are represented by coloring on the drug molecules in **Figure 5.7**.

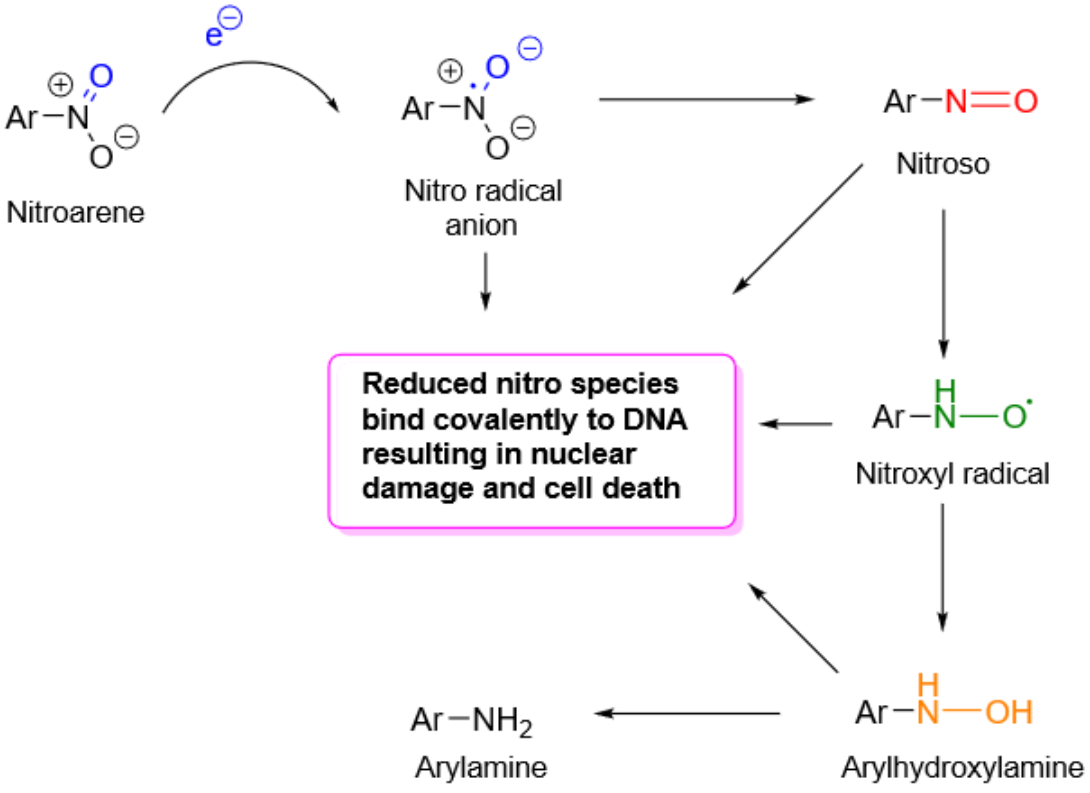
Figure 5.7. ClinTox Analysis Example 1.



2D representation of nimesulide (a), acetaminophen (b), thalidomide (c), sorafenib (d), and tolcapone (e).

The fragment in rank 20 in **Figure 5.6**, whose SMILES representation is **O** = [**N**+][**O**−], represents the nitro group when bound to a molecule. Although the nitro group is present in many active molecules, it has toxicity issues and is often categorised as a structural pharmacophore and/or toxicophore group. In general, many studies in the literature on molecules containing nitro group demonstrate toxicity problems such as carcinogenicity, hepatotoxicity, mutagenicity, and bone marrow suppression [64]. Nitro radical anion, nitrozo derivative, nitroxyl radical, hydroxylamine and primary amine derivatives are formed in the biotransformation process by reduction of the aromatic nitro group [65]. Studies have revealed that intermediate products are responsible for toxicity. Especially hydroxylamine derivatives are responsible for methemoglobinemia, while other intermediates have shown mutagenicity and carcinogenicity [66]. **Figure 5.8** shows the mechanism of reductive biotransformation of aromatic nitro compounds.

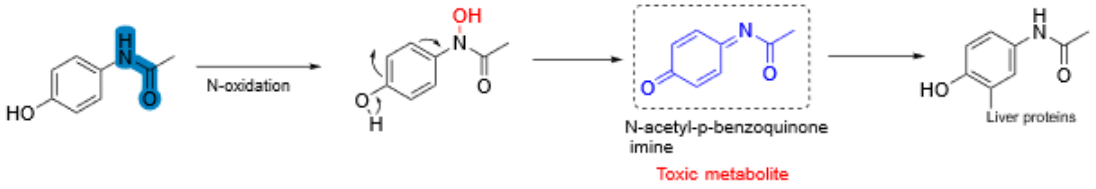
Figure 5.8. Reductive Biotransformation Mechanism of Aromatic Nitro Compounds.



Nimesulide (4-nitro-2-phenoxymethane-sulfo-anilide), shown in **Figure 5.7a**, is a selective cyclo-oxygenase (COX)-2 inhibitor nonsteroidal anti-inflammatory drug used in the treatment of various inflammatory and pain conditions. The aromatic nitro group in nimesulide undergoes reductive metabolism and shows hepatotoxic effect. Nimesulide bioactivation in the liver produces reactive intermediates as shown in **Figure 5.8**, binds to macromolecules in the body and causes oxidoreductive stress [67].

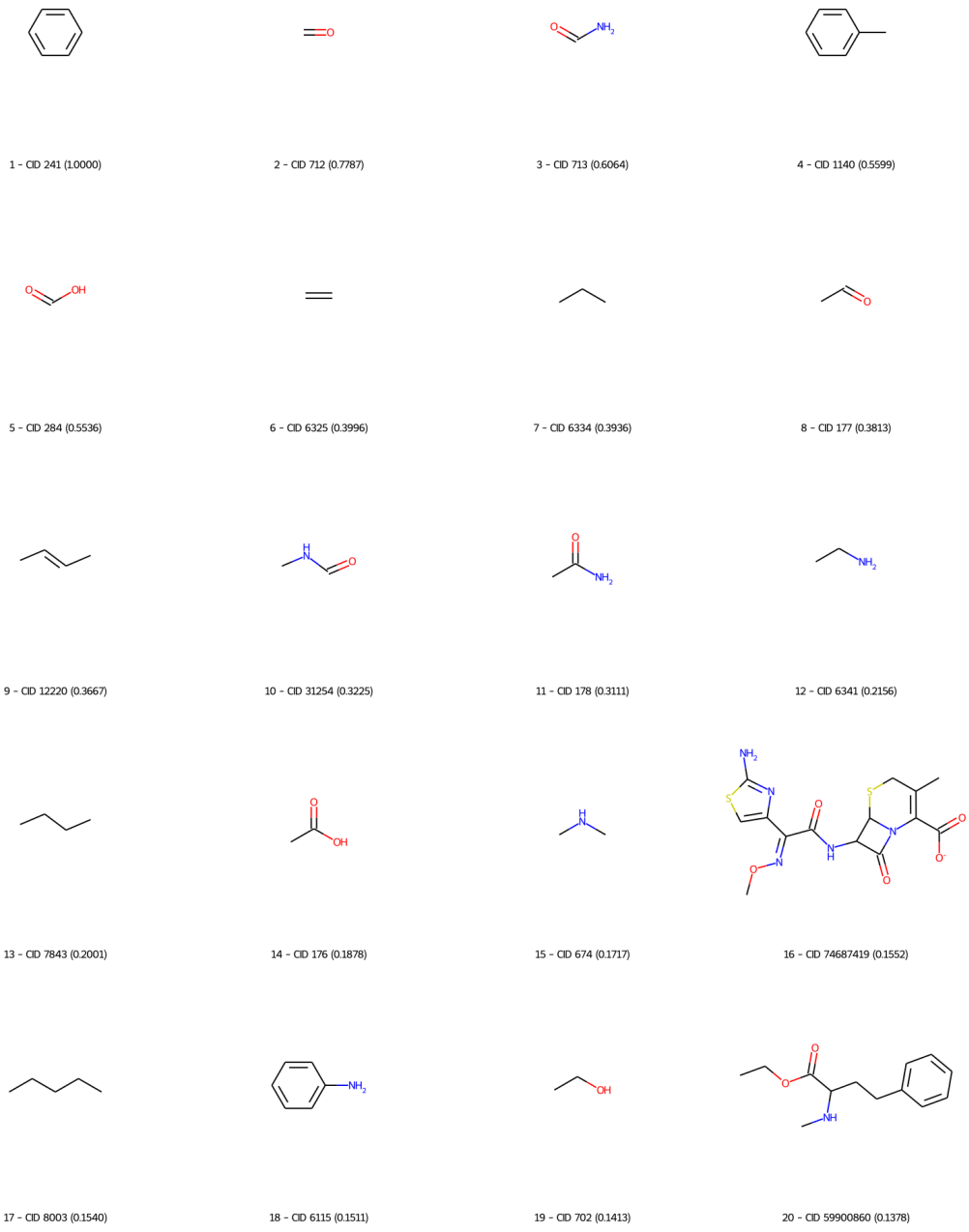
Fragment in rank 3 (**NC = O**) found by the algorithm represents the formamide structure. Intermediates formed as a result of N-oxidation of aromatic amides show carcinogenic and cytotoxic effects by covalent bonding with biological macromolecules as in aromatic amines. Acetaminophen compound is an analgesic drug molecule. The N-acetylaminoquinone derivative formed from the N-hydroxyacetaminophen intermediate product formed during biotransformation is responsible for the hepatotoxic activity of the compound [68]. The reaction mechanism is given in **Figure 5.9**.

Figure 5.9. ClinTox Analysis Example 2.



5.4. SIDER Analysis

Figure 5.10. Top 20 Fragments for SIDER task.

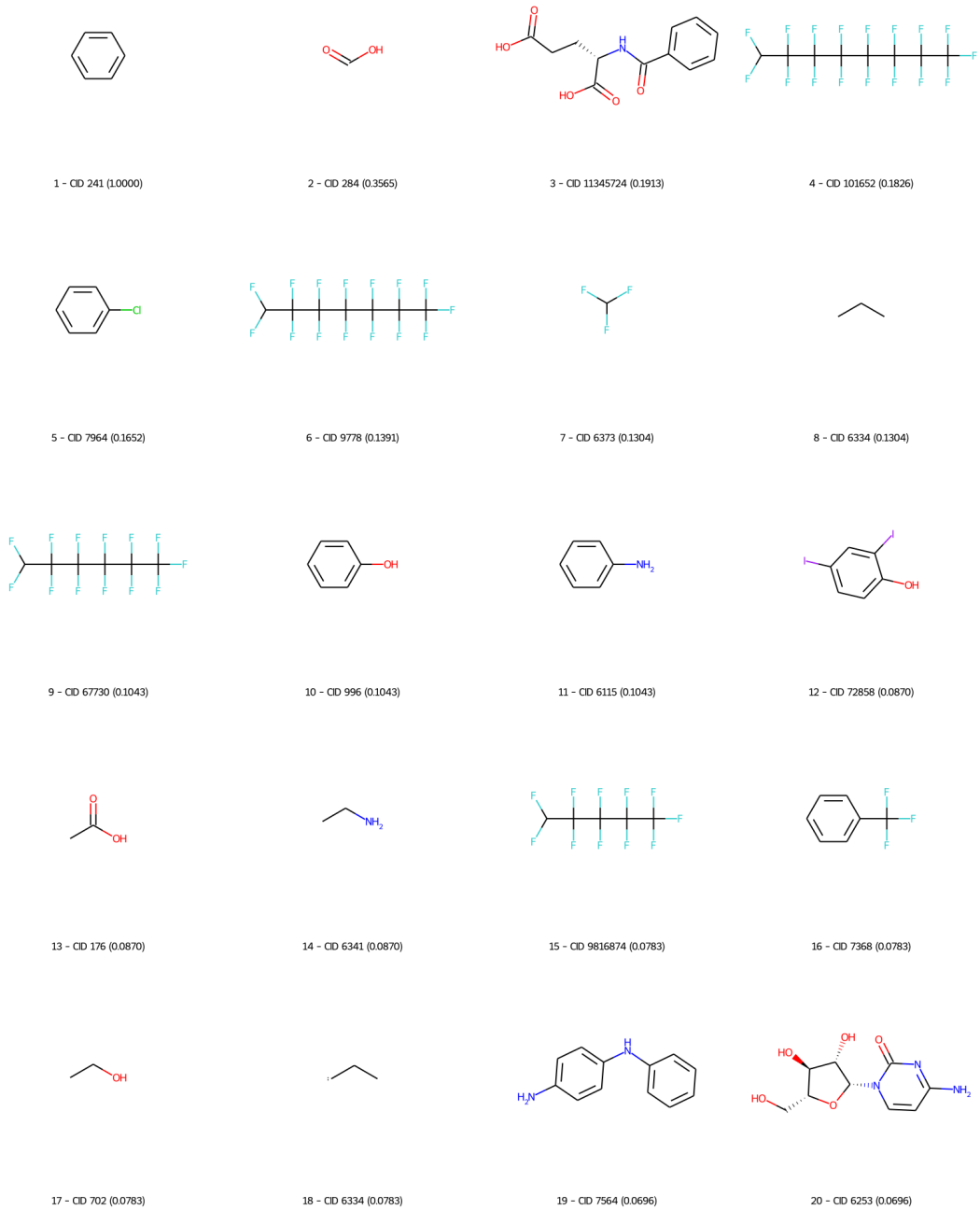


SIDER task encompasses a dataset based on adverse effects, which introduces a wide range of parameters for interpretability. For instance, a drug may have been misused, given the wrong dosage, or had interactions with other medications that led to a reported adverse effect. That's why, no interpretation has been performed for this

dataset due to the potential unreliability of analysis methods used to interpret these fragments.

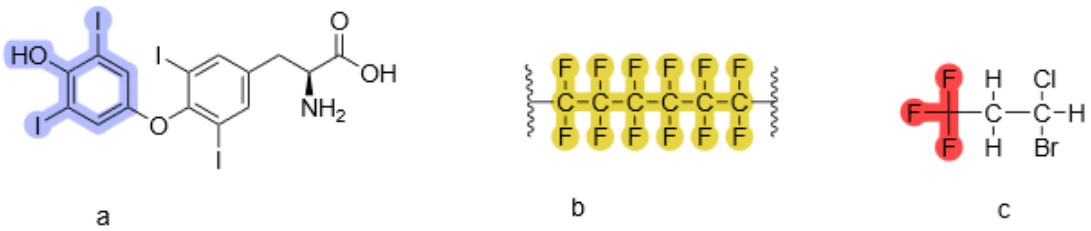
5.5. Tox21 Analysis

Figure 5.11. Top 20 Fragments for Tox21 task.



The contribution of polyhalogen groups to toxicity was mentioned in the ClinTox dataset analysis (**Section 5.3**). Organic halogenated compounds should be used very carefully. They have a high potential to cause toxicity by accumulating in adipose tissue and cause cancer. Halogens facilitate passage through the blood brain barrier because they increase lipophilicity [69]. Thyroxine, chloramphenicol, teflon and halothane organic halogenated compounds are examples of halogenated compounds with high toxicity. Eight of the top 20 fragments found by the algorithm are halogenated compounds. It can be said that the high amount of fluorine halogen found in fragments 4, 6, 9, and 15 will show toxicity similar to the teflon compound in **Figure 5.12**. In addition, fragments 11 and 19 containing aromatic amine and amide structure may show toxicity through the mechanism shown in **Figure 5.9**.

Figure 5.12. Tox21 Analysis Example 1.



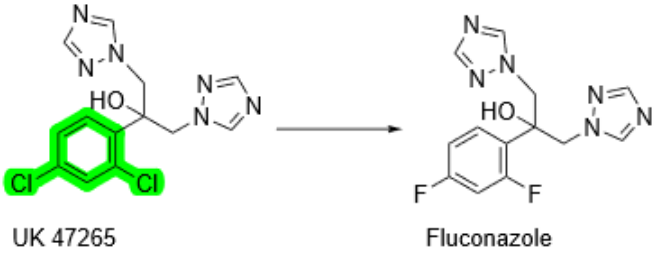
2D representation of thyroxine (a), teflon (b), and halothane (c). The fragments found by the algorithm are marked on the compounds in the figure.

Studies have shown that non-steroidal anti-inflammatory drugs (NSAID) with acidic structure can covalently bind to hepatic proteins. NSAIDs containing carboxylic acid structure form electrophilic intermediates which are acyl glucronides and bind to nucleophilic amino acids. Although not proven in vivo, this covalent binding mechanism to these macromolecules has been suggested to play a role in the hepatic toxicity of many NSAIDs [70]. Fragments 2, 3 and 13 obtained by the algorithm successfully found the carboxylic acid structure.

In one study, the halogen substituents of the antifungal drug UK 47265 were modified to find a compound that is less toxic to the liver. Here, 1,3-dichlorobenzene containing the chlorobenzene fragment in fragment 5 was removed and replaced by 1,3-

difluorobenzene structure and the less toxic fluconazole compound was obtained [71,72]. 2D representation of UK47265 shown in **Figure 5.13**.

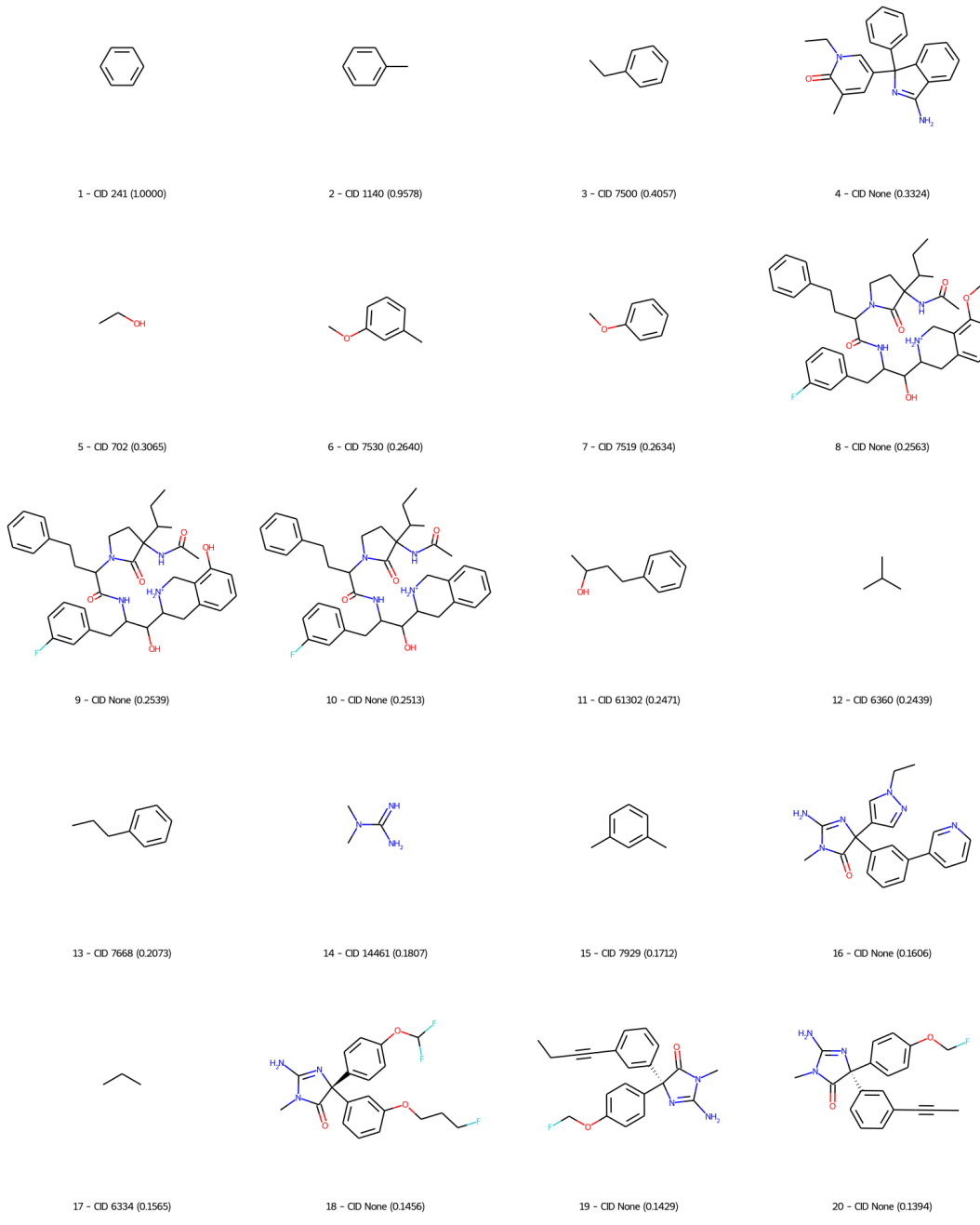
Figure 5.13. Tox21 Analysis Example 2.



2D representation of UK47265 and fluconazole. Toxicity was reduced by the diversification of aromatic substituents. The **Clc1ccccc1** fragment found by the algorithm is the subfragment of the 1,3-dichlorobenzene structure marked in green on the molecule.

5.6. BACE Regression Analysis

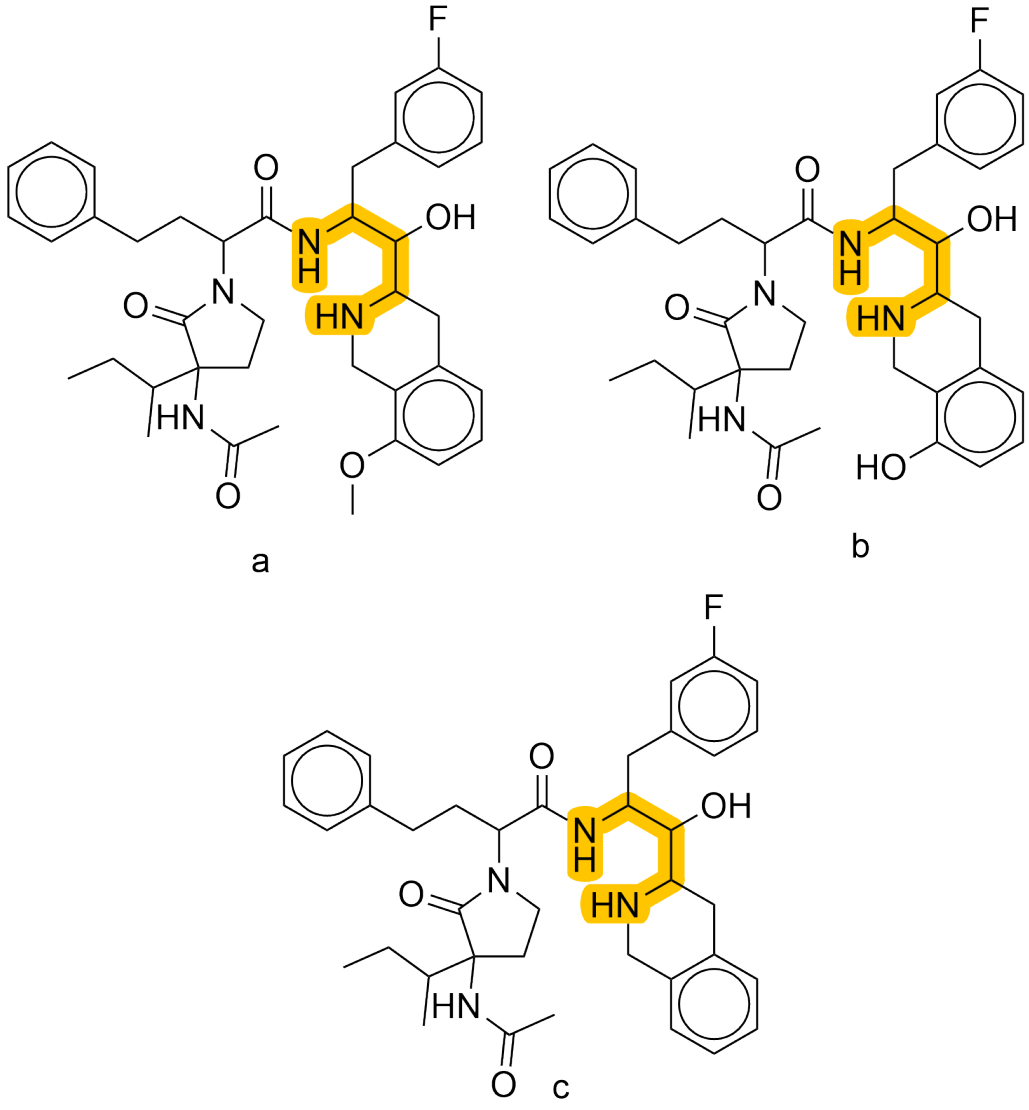
Figure 5.14. Top 20 Fragments for BACE Regression task.



The fragment in rank 4 in **Figure 5.14** is included in [73], who has developed fresh chemical substances with BACE-inhibiting properties. A series of diaminopropane (marked in orange in **Figure 5.15**) analogs have been introduced as potential BACE1 inhibitors in [74], from which a patent has also been issued in [75]. These derivatives'

extra large rings allow them to fit perfectly in the enzyme's ligand-binding regions due to their extra bulk. The BACE Regression task scoring system has assigned the fragments a, b, and c in **Figure 5.15** the rankings 8, 9 and 10; respectively in **Figure 5.14**.

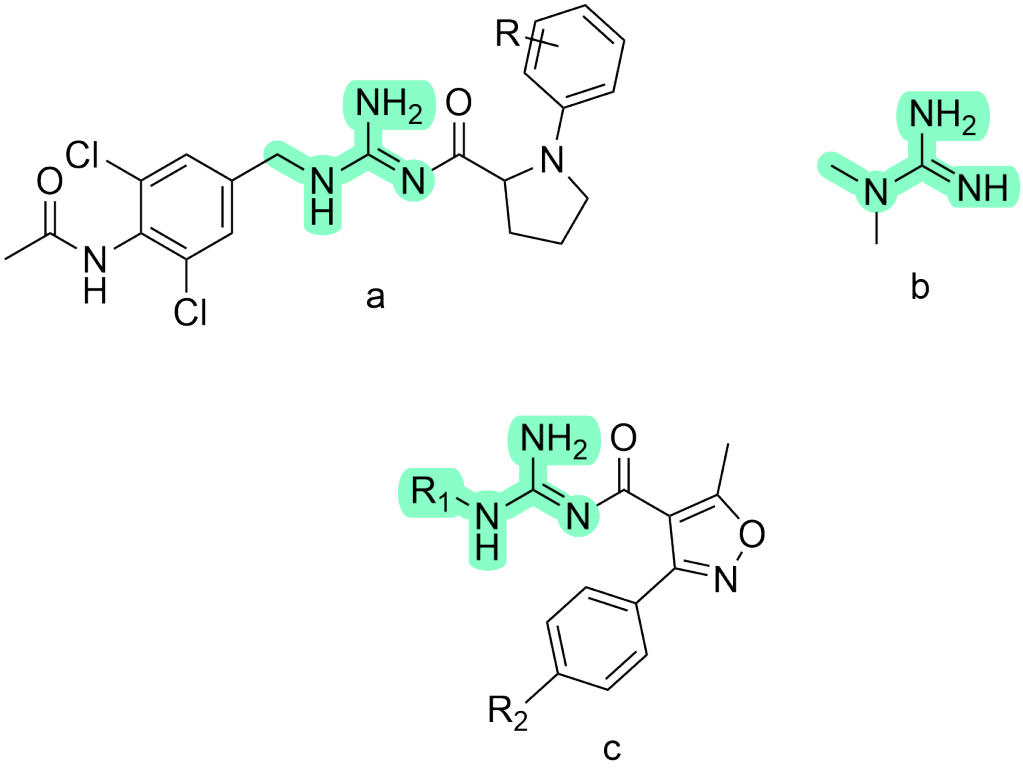
Figure 5.15. BACE Regression Analysis Example 1.



The algorithm ranked the 1,1-dimethylguanidine fragment **Figure 5.16b** at position 14 out of the top 20 fragments for the BACE Regression task. The acyclic guanidine-based BACE1 inhibitors reported in these studies contain a guanidine base bound to one alkyl chain (shown in mint green in **Figure 5.16**), not two alkyl chains as seen in the fragment extracted by the algorithm (**Figure 5.16b**), which has two

alkyl side chains (dimethyl); despite this, a large corpus of acyclic guanidine-based BACE1 inhibitors exist. The significance of this group in the development of BACE1 inhibitors is highlighted by studies published in [76] (**Figure 5.16a**) and [77] (**Figure 5.16c**). Furthermore, as stated before, cyclic guanidines (aminohydantoins, iminohydantoins) [59–61] are also a common structural core among BACE1 inhibitory compounds, and the fragment $\mathbf{CN(C)C(=N)N}$ extracted in this case by the algorithm may also be pointing to the cyclic guanidine moieties, specifically iminohydantoins. Overall, the algorithm's success in locating significant language units (fragments) can be seen in its discovery of and highlighting of the guanidine moiety as a key fragment, which is frequently used as a common motif across BACE1 inhibitors.

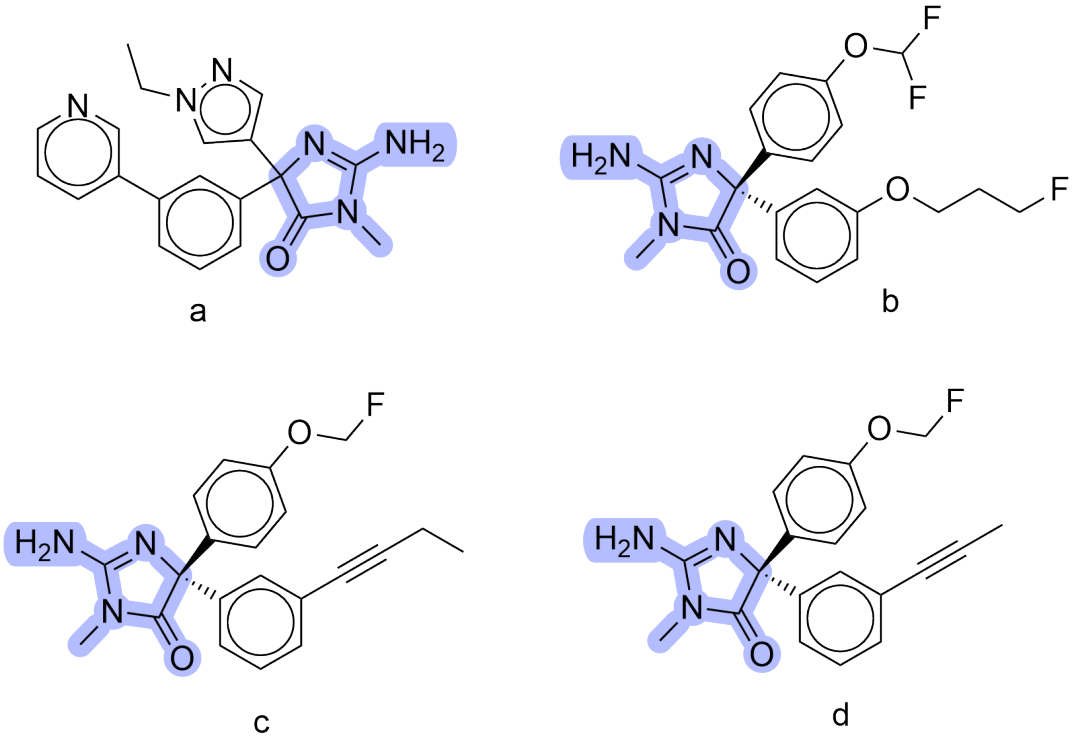
Figure 5.16. BACE Regression Analysis Example 2.



Similar to the BACE Classification task, the algorithm was also successful in removing fragments from the BACE Regression task that contain an aminohydantoin moiety (**Figure 5.17**, where the aminohydantoin core is depicted in purple). This time, among the top 20 fragments of the BACE Regression task, the algorithm scored fragments a, b, c and d with the corresponding ranks of 16, 18, 19 and 20. These

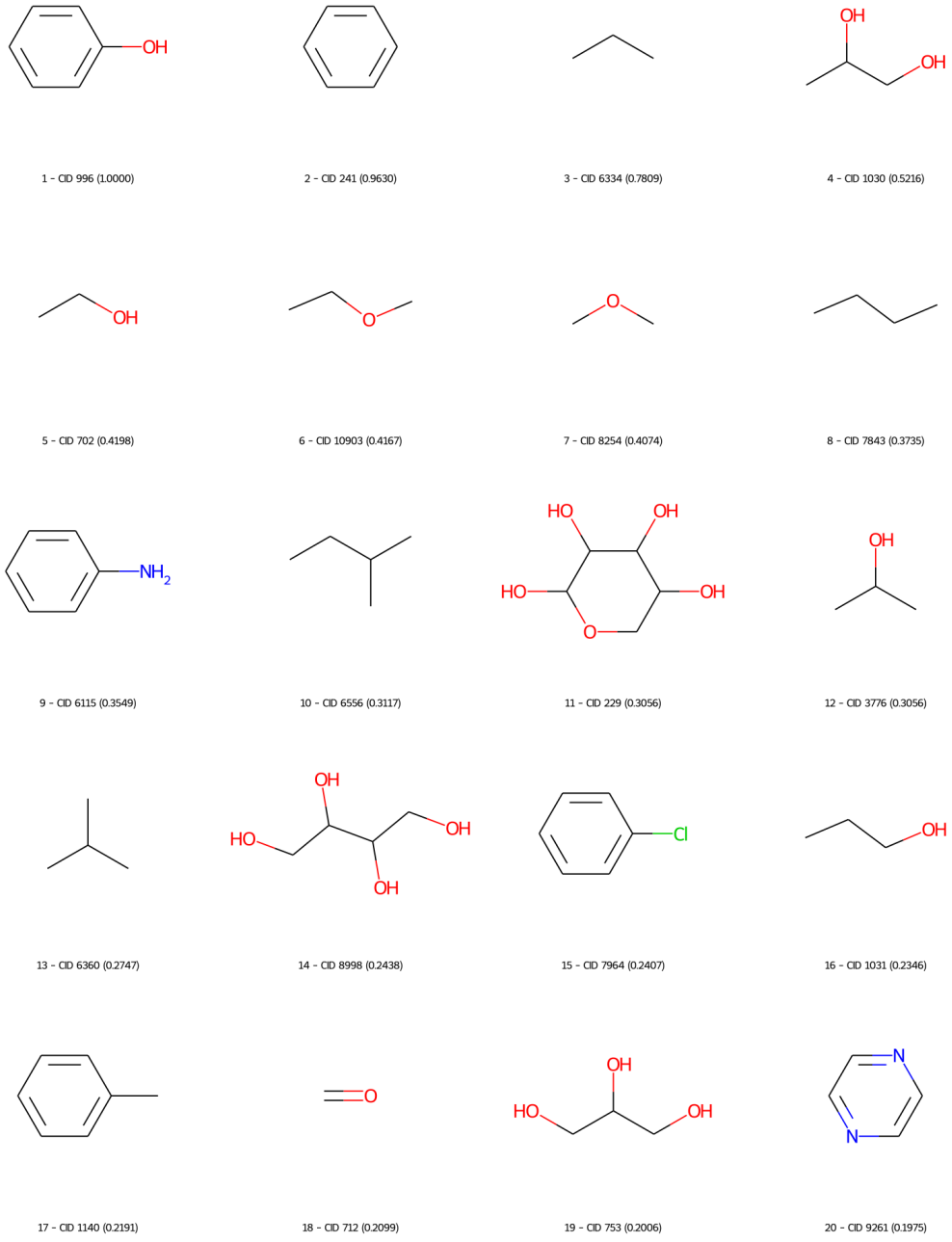
fragments, which can be found in [59, 60].

Figure 5.17. BACE Regression Analysis Example 3.



5.7. ESOL Analysis

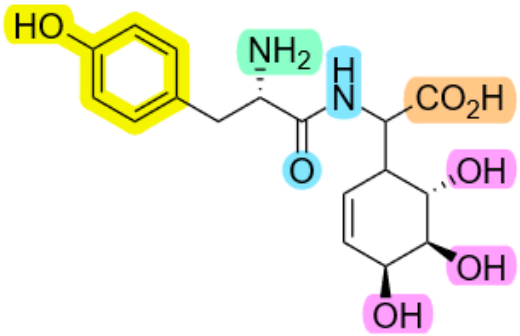
Figure 5.18. Top 20 Fragments for ESOL task.



Generally, the factors affecting the solubility of a compound in water depend on the polarity of the molecule and the presence of functional groups that can form hydrogen bonds with water molecules [78]. Increasing the polarity of a molecule by adding hydrophilic groups (hydroxyl, amino, carboxylic acid etc.) is one of the most

widely used methods in drug design and development studies. For example, polar hydroxyl group and polar heterocyclic rings were added to thioconazole compound which is used only in skin effects due to its high lipophilicity. In this way, fluconazole compound whose solubility was improved and which can be used against systemic infections was synthesised [79].

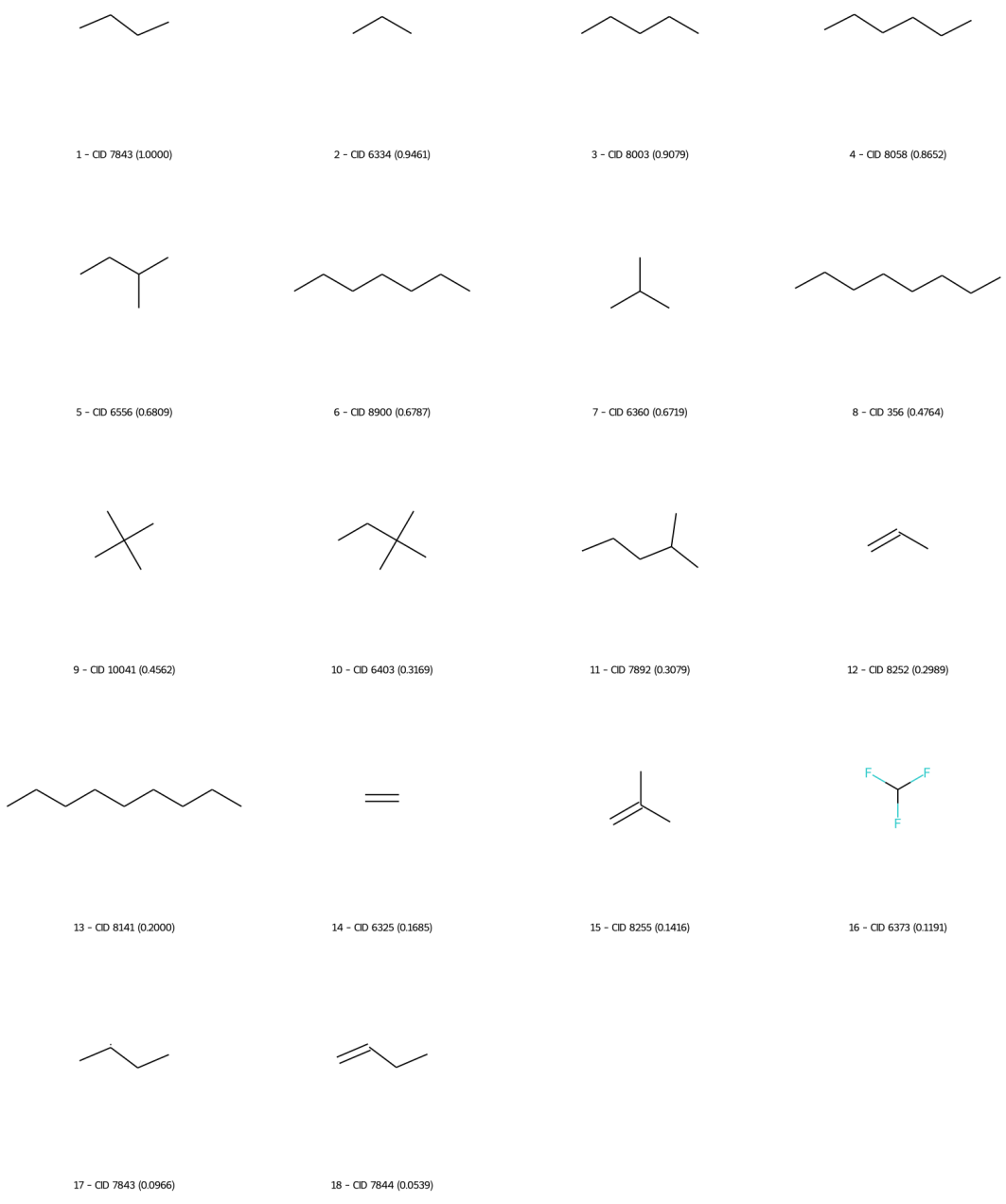
Figure 5.19. Functional Groups that Cause Excessive Polarity in a Drug.



The groups causing high polarity in an antibacterial drug are coloured in **Figure 5.19**. Out of the top 20 compounds found by the algorithm, 7 of them contain hydroxyl (OH) group and 1 contains phenol group (represented by pink and yellow colour respectively in **Figure 5.19**). Fragments 6, 7, 9 and 18 contain ether ($R-O-R$), ether ($R-OR$) amino (NH_2) and carbonyl ($C=O$) structures, respectively. Fragment 20 found by the algorithm represents the pyrazine ring. Pyrazine ring freely soluble in water.

5.8. FreeSolv Analysis

Figure 5.20. Top 20* Fragments for FreeSolv task.



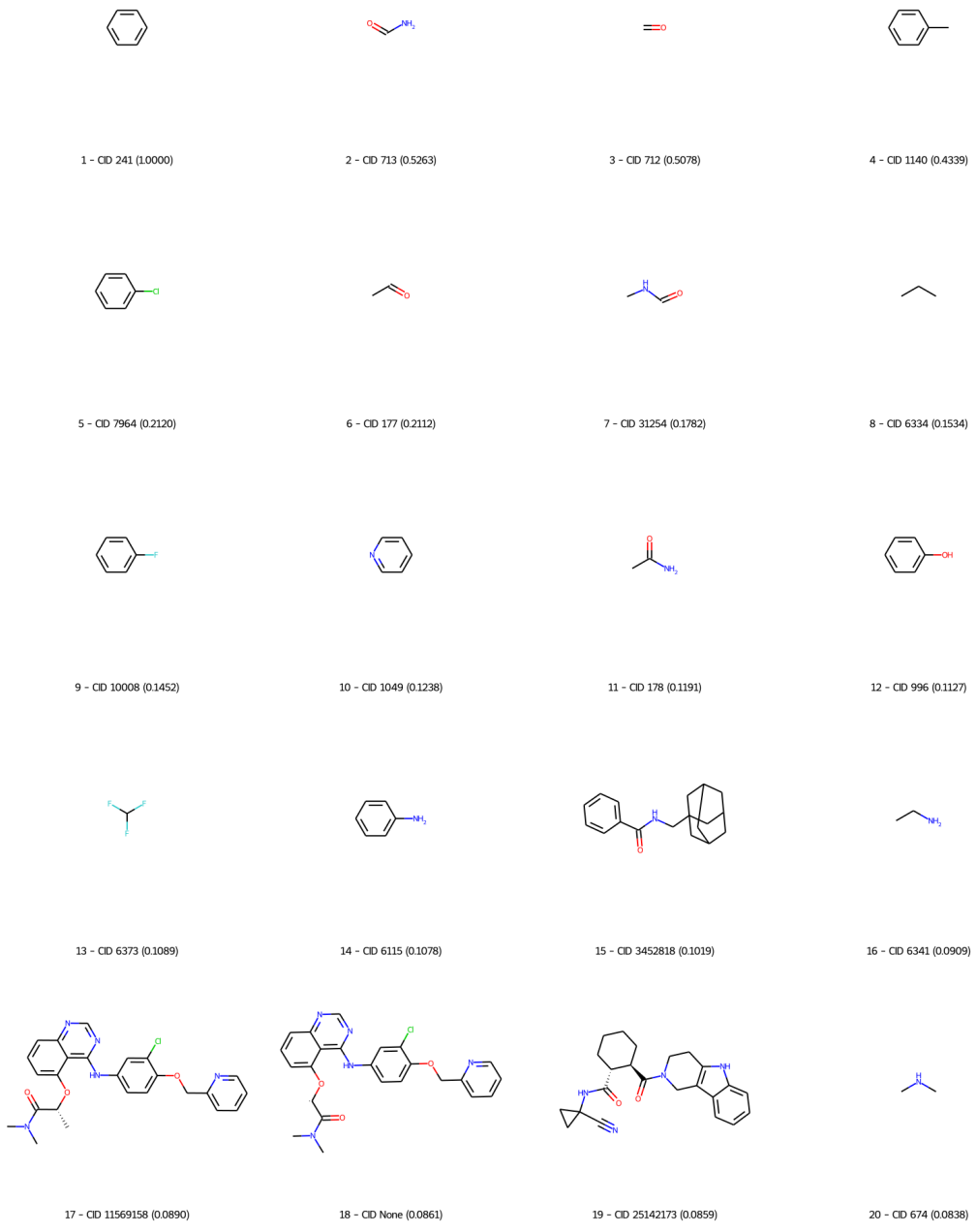
* Only 18 fragments could be found due to the task's small sized dataset.

The fragments found for the FreeSolv task by the model do not represent interpretable parts, unfortunately. The obtained fragments primarily correspond to com-

mon simple aliphatic groups found in most molecules, including ethane, propane etc. That's why, no interpretation has been performed for this dataset due to the potential unreliability of analysis methods used to interpret these fragments.

5.9. Lipophilicity Analysis

Figure 5.21. Top 20 Fragments for Lipophilicity task.



Similar to the BBBP task, it cannot be said that the extracted fragments in the Lipophilicity task directly contribute to an increase in the overall lipophilicity of the compound from which they were extracted. However, the phenyl moieties present in fragments ranked 1, 4, 5 and 9 generally increase the lipophilicity of chemical compounds. Another well-known property of the trifluoromethyl fragment, which is ranked at position 13, is its ability to make molecules more lipophilic [13]. The bulky group adamantyl, which is present in fragment 15, is expected to increase lipophilicity. Similar to the BBBP task, the scoring system is not intended to rank the fragments according to their physicochemical differences, but rather to aid the method in finding the best fragment. Fragments among the ranks 17–19 also contain bulky hydrophobic structures, which amplify the lipophilic features of the molecule.

6. DISCUSSION

6.1. Affinity Between Benchmark Results and Expectations

There are 4 different main titles in [39] for its datasets which are quantum mechanics, physical chemistry, biophysics, physiology. Due to its input incompatibility (3D-coordinates), datasets of the quantum mechanics could not be used in this work. In the context of the remained tasks' complexity, it is acceptable to assume that learning the physical chemistry tasks (i.e., ESOL, FreeSolv, Lipophilicity) easier than the rest since the tasks include only the information about chemicals' themselves not any external variable's information.

On the other hand, it can be seen that the biophysics tasks (e.g., BACE) are more complex since now the tasks include not only the information about chemicals' themselves but also at least one external variable's information (e.g., proteins and interactions with proteins).

But the most complex one among them is clearly the physiology tasks (e.g., BBBP, Tox21, SIDER) those are also regression tasks, since now there are much more information due to consecutive systematic interactions in cell-scale. Overall, from this point of view, it is expected that physical chemistry tasks' scores should be better than biophysics tasks' scores and also biophysics tasks' scores should be better than physiology tasks' scores.

And indeed, the obtained benchmark results are parallel with the expectations mentioned above. For the physical chemistry tasks, the model is only in top 3 ranks among all the SOTA models. For the biophysics tasks, the only tested task is BACE and it is the only classification task that has a "relatively" good rank which is 4th to last among all the SOTA models. And lastly, for the physiology tasks, the model is unfortunately either last or second last except ClinTox task.

6.2. Comparison of Regression and Classification Tasks

The model definitely much better for regression tasks than classification tasks as it is only in top 3 ranks among all the SOTA models. One of the possible explanations of this situation is related with the target values' continuity. In regression tasks, the target variable is a continuous variable, such as a numerical value, and the goal is to predict this value as accurately as possible. In contrast, in classification tasks, the target variable is a discrete variable, such as a category or a binary label, and the goal is to predict the correct category or label for each input instance. Because regression tasks involve continuous variables, the target values can take on a wide range of possible values, which can make the problem easier to predict accurately. In contrast, in classification tasks, the number of possible target values is often limited, which can make it more difficult to achieve high accuracy [80, 81].

Another possible explanation is the fact that classification tasks may have imbalanced datasets, where some categories or labels have much fewer examples than others, which can further complicate the task [82, 83] and as it can be seen in the figures those in **Appendix A**, most of the classification datasets seem heterogeneous.

6.3. Found Chemical Fragments

Some fragments that in the same top 20 fragments looks like gradually growing branches(e.g., fragments in rank 4, 6, 9 and 15 in the top 20 fragments for Tox21 task). Since the fragments' branching size affect their score, bigger branches were found in better positions in top 20 fragments for corresponding tasks.

Also, some fragments can be found more than one in top 20 fragments for different tasks. The reason of this related with the fact that some small molecules (e.g., benzene, ethene, propane) repeat very frequently in most of the datasets. Since the fragments' frequency affect their score, these small molecules can be seen in most of the top 20 fragments for different tasks. Also, again, some fragments are found during

researches those are not registered in the PubChem database, so called “None” in the top 20 fragments, yet there are patents for these fragments in the literature which are mentioned in **Chapter 5**.

It is necessary to note that for example, in case of the Tox21 task, the fragment in rank 7 is more toxic than the fragment in ranked 4. Therefore, the scoring procedure cannot precisely rank them in order. Nevertheless, “importance” is a relative term, so it has been considered an acceptable result for the scoring procedure to at least highlight certain chemically meaningful fragments rather than having a fully accurate ranking among the fragments themselves.

7. CONCLUSION

We here offer an interpretation strategy for compounds and their fragments using the AR-Tree model. Thanks to the task-specific attention mechanism of the model, it can highlight the important fragments for the corresponding tasks. The results of the experiments show that our approach outperforms the SOTA models for ClinTox and BACE Regression tasks. In terms of interpretability, the model and scoring strategy succeeded in finding chemically meaningful fragments in the compounds for BACE Classification, BACE Regression, ClinTox, Tox21, ESOL and Lipophilicity tasks.

Determining the most significant fragments for the given tasks is an important feature because when predicting physicochemical and functional properties, recognizing the important parts of compounds can give insights to experts for experimental studies. Our approach allows to construct novel compounds with desired features more precisely and hopefully, by this way, individuals suffering from various incurable diseases may find the necessary compounds.

REFERENCES

1. Kim, J., S. Park, D. Min and W. Kim, "Comprehensive survey of recent drug discovery using deep learning", *International Journal of Molecular Sciences*, Vol. 22, No. 18, p. 9983, 2021.

2. Huang, B. and O. A. Von Lilienfeld, "Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity", *The Journal of Chemical Physics*, Vol. 145, No. 16, p. 161102, 2016.

3. David, L., A. Thakkar, R. Mercado and O. Engkvist, "Molecular representations in AI-driven drug discovery: a review and practical guide", *Journal of Cheminformatics*, Vol. 12, No. 1, pp. 1–22, 2020.

4. Shi, J., L. Hou, J. Li, Z. Liu and H. Zhang, "Learning to Embed Sentences Using Attentive Recursive Trees", , 2018.

5. Shrikumar, A., P. Greenside and A. Kundaje, "Learning Important Features Through Propagating Activation Differences", , 2019.

6. Zeiler, M. D. and R. Fergus, "Visualizing and Understanding Convolutional Networks", , 2013.

7. Zhou, J. and O. Troyanskaya, "Predicting effects of noncoding variants with deep learning-based sequence model", *Nature methods*, Vol. 12, 08 2015.

8. Zintgraf, L. M., T. S. Cohen, T. Adel and M. Welling, "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis", , 2017.

9. Robnik-Šikonja, M. and I. Kononenko, "Explaining Classifications For Individual Instances", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 20, No. 5, pp. 589–600, 2008.

10. Klöppel, S., C. Stonnington, C. Chu, B. Draganski, R. Scahill, J. Rohrer, N. Fox, C. Jack, J. Ashburner and R. Frackowiak, "Automatic classification of MR scans in Alzheimer's Disease", *Brain : a journal of neurology*, Vol. 131, pp. 681–9, 04 2008.

11. Ecker, C., A. Marquand, J. Mourão-Miranda, P. Johnston, E. Daly, M. Brammer, S. Maltezos, C. Murphy, D. Robertson, S. Williams and D. Murphy, "Describing the Brain in Autism in Five Dimensions-Magnetic Resonance Imaging-Assisted Diagnosis of Autism Spectrum Disorder Using a Multiparameter Classification Approach", *The Journal of neuroscience : the official journal of the Society for Neuroscience*, Vol. 30, pp. 10612–23, 08 2010.

12. Mourão-Miranda, J., A. Bokde, C. Born, H. Hampel and M. Stetter, "Classifying brain states and determining the discriminating activation patterns: support vector machine on functional MRI data", *NeuroImage*, Vol. 28, pp. 980–95, 01 2006.

13. Wang, Z., A. Childress, D. Wang and J. Detre, "Support Vector Machine Learning-based fMRI Data Group Analysis", *NeuroImage*, Vol. 36, pp. 1139–51, 08 2007.

14. Gaonkar, B. and C. Davatzikos, "Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification", *NeuroImage*, Vol. 78, pp. 270–283, 2013, `https://www.sciencedirect.com/science/article/pii/S1053811913003169`.

15. Haufe, S., F. Meinecke, K. Görgen, S. Dähne, J.-D. Haynes, B. Blankertz and F. Biessmann, "On the interpretation of weight vectors of linear models in multivariate neuroimaging", *NeuroImage*, Vol. 87, 11 2013.

16. Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller and W. Samek, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation", *PLoS ONE*, Vol. 10, p. e0130140, 07 2015.

17. Shrikumar, A., P. Greenside, A. Shcherbina and A. Kundaje, "Not Just a Black Box: Learning Important Features Through Propagating Activation Differences", , 2017.

18. Kindermans, P.-J., K. Schütt, K.-R. Müller and S. Dähne, "Investigating the influence of noise and distractors on the interpretation of neural networks", , 2016.

19. Simonyan, K., A. Vedaldi and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", , 2014.

20. Springenberg, J. T., A. Dosovitskiy, T. Brox and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net", , 2015.

21. Erhan, D., Y. Bengio, A. Courville and P. Vincent, "Visualizing Higher-Layer Features of a Deep Network", *Technical Report, Univeristé de Montréal*, , No. 1341, 06 2009.

22. Krizhevsky, A., I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Commun. ACM*, Vol. 60, No. 6, p. 84–90, 05 2017, https://doi.org/10.1145/3065386.

23. Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization", *International Journal of Computer Vision*, Vol. 128, No. 2, pp. 336–359, 10 2019, https://doi.org/10.1007%2Fs11263-019-01228-7.

24. Sundararajan, M., A. Taly and Q. Yan, "Gradients of Counterfactuals", , 2016.

25. Bowman, S. R., J. Gauthier, A. Rastogi, R. Gupta, C. D. Manning and C. Potts, "A Fast Unified Model for Parsing and Sentence Understanding", , 2016.

26. Yogatama, D., P. Blunsom, C. Dyer, E. Grefenstette and W. Ling, "Learning to Compose Words into Sentences with Reinforcement Learning", , 2016.

27. Maillard, J., S. Clark and D. Yogatama, "Jointly learning sentence embeddings and syntax with unsupervised Tree-LSTMs", *Natural Language Engineering*, Vol. 25, No. 4, pp. 433–449, 07 2019, https://doi.org/10.1017%2Fs1351324919000184.

28. Cocke, J., "Programming languages and their compilers: Preliminary notes", , 1969.

29. Younger, D. H., "Recognition and Parsing of Context-Free Languages in Time n^3", *Inf. Control.*, Vol. 10, pp. 189–208, 1967.

30. Kasami, T., "An Efficient Recognition and Syntax-Analysis Algorithm for Context-Free Languages", , 1965.

31. Choi, J., K. M. Yoo and S. goo Lee, "Learning to Compose Task-Specific Tree Structures", , 2017.

32. Jang, E., S. Gu and B. Poole, "Categorical Reparameterization with Gumbel-Softmax", , 2017.

33. "Learning to parse from a semantic objective: It works. Is it syntax?", .

34. dos Santos, C., M. Tan, B. Xiang and B. Zhou, "Attentive Pooling Networks", , 2016.

35. Munkhdalai, T. and H. Yu, "Neural Tree Indexers for Text Understanding", , 2017.

36. Arora, S., Y. Liang and T. Ma, "A Simple but Tough-to-Beat Baseline for Sentence Embeddings", *International Conference on Learning Representations*, 2017.

37. Lin, Z., M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou and Y. Bengio, "A Structured Self-attentive Sentence Embedding", , 2017.

38. Kim, Y., C. Denton, L. Hoang and A. M. Rush, "Structured Attention Networks",

, 2017.

39. Wu, Z., 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.

40. Weininger, D., "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules", *Journal of chemical information and computer sciences*, Vol. 28, No. 1, pp. 31–36, 1988.

41. Schwaller, P., T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas and A. A. Lee, "Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction", *ACS central science*, Vol. 5, No. 9, pp. 1572–1583, 2019.

42. Klein, G., Y. Kim, Y. Deng, J. Senellart and A. M. Rush, "OpenNMT: Open-Source Toolkit for Neural Machine Translation", *Proc. ACL*, 2017, https://doi.org/10.18653/v1/P17-4012.

43. Van Rossum, G. and F. L. Drake, *Python 3 Reference Manual*, CreateSpace, Scotts Valley, CA, 2009.

44. Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library", *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

45. Ramsundar, B., P. Eastman, P. Walters, V. Pande, K. Leswing and Z. Wu, *Deep Learning for the Life Sciences*, O'Reilly Media, 2019.

46. McKinney, W. *et al.*, "Data structures for statistical computing in python", *Proceedings of the 9th Python in Science Conference*, Vol. 445, pp. 51–56, Austin, TX,

2010.

47. Swain, M., R. Sjögren, zachcp, Y. Hsiao, L. Lazzaro and B. Dahlgren, "PubChemPy: A way to interact with PubChem in Python", , 04 2017, https://github.com/mcs07/PubChemPy.

48. Kim, S., J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang and E. E. Bolton, "PubChem 2023 update", *Nucleic Acids Research*, Vol. 51, No. D1, pp. D1373–D1380, 10 2022, https://doi.org/10.1093/nar/gkac956.

49. Landrum, G., "RDKit documentation", *Release*, Vol. 1, No. 1-79, p. 4, 2013.

50. Zhu, X., P. Sobhani and H. Guo, "Long Short-Term Memory Over Tree Structures", , 2015.

51. Tai, K. S., R. Socher and C. D. Manning, "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks", , 2015.

52. Patro, S. G. K. and K. K. Sahu, "Normalization: A Preprocessing Stage", , 2015.

53. Wang, Y., J. Wang, Z. Cao and A. Barati Farimani, "Molecular contrastive learning of representations via graph neural networks", *Nature Machine Intelligence*, Vol. 4, No. 3, pp. 279–287, 2022.

54. Ahmad, W., E. Simon, S. Chithrananda, G. Grand and B. Ramsundar, "ChemBERTa-2: Towards Chemical Foundation Models", , 2022.

55. Ross, J., B. Belgodere, V. Chenthamarakshan, I. Padhi, Y. Mroueh and P. Das, "Large-scale chemical language representations capture molecular structure and properties", *Nature Machine Intelligence*, Vol. 4, No. 12, pp. 1256–1264, 2022.

56. Hu, W., B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande and J. Leskovec, "Strate-

gies for pre-training graph neural networks", *arXiv preprint arXiv:1905.12265*, 2019.

57. Moussa-Pacha, N. M., S. M. Abdin, H. A. Omar, H. Alniss and T. H. Al-Tel, "BACE1 inhibitors: Current status and future directions in treating Alzheimer's disease", *Medicinal Research Reviews*, Vol. 40, No. 1, pp. 339–384, 2020, https://onlinelibrary.wiley.com/doi/abs/10.1002/med.21622.

58. Weiss, M. M., T. Williamson, S. Babu-Khan, M. D. Bartberger, J. Brown, K. Chen, Y. Cheng, M. Citron, M. D. Croghan, T. A. Dineen, J. Esmay, R. F. Graceffa, S. S. Harried, D. Hickman, S. A. Hitchcock, D. B. Horne, H. Huang, R. Imbeah-Ampiah, T. Judd, M. R. Kaller, C. R. Kreiman, D. S. La, V. Li, P. Lopez, S. Louie, H. Monenschein, T. T. Nguyen, L. D. Pennington, C. Rattan, T. San Miguel, E. Sickmier, R. C. Wahl, P. H. Wen, S. Wood, Q. Xue, B. H. Yang, V. F. Patel and W. Zhong, "Design and Preparation of a Potent Series of Hydroxyethylamine Containing -Secretase Inhibitors That Demonstrate Robust Reduction of Central -Amyloid", *Journal of Medicinal Chemistry*, Vol. 55, No. 21, pp. 9009–9024, 2012, https://doi.org/10.1021/jm300119p.

59. Malamas, M. S., A. Robichaud, J. Erdei, D. Quagliato, W. Solvibile, P. Zhou, K. Morris, J. Turner, E. Wagner, K. Fan, A. Olland, S. Jacobsen, P. Reinhart, D. Riddell and M. Pangalos, "Design and synthesis of aminohydantoins as potent and selective human -secretase (BACE1) inhibitors with enhanced brain permeability", *Bioorganic Medicinal Chemistry Letters*, Vol. 20, No. 22, pp. 6597–6605, 2010, https://www.sciencedirect.com/science/article/pii/S0960894X1001320X.

60. Malamas, M. S., J. Erdei, I. Gunawan, K. Barnes, Y. Hui, M. Johnson, A. Robichaud, P. Zhou, Y. Yan, W. Solvibile, J. Turner, K. Y. Fan, R. Chopra, J. Bard and M. N. Pangalos, "New pyrazolyl and thienyl aminohydantoins as potent BACE1 inhibitors: Exploring the S2′ region", *Bioorganic Medicinal Chemistry Letters*, Vol. 21, No. 18, pp. 5164–5170, 2011,

https://www.sciencedirect.com/science/article/pii/S0960894X11009954.

61. Cumming, J. N., E. M. Smith, L. Wang, J. Misiaszek, J. Durkin, J. Pan, U. Iserloh, Y. Wu, Z. Zhu, C. Strickland, J. Voigt, X. Chen, M. E. Kennedy, R. Kuvelkar, L. A. Hyde, K. Cox, L. Favreau, M. F. Czarniecki, W. J. Greenlee, B. A. McKittrick, E. M. Parker and A. W. Stamford, "Structure based design of iminohydantoin BACE1 inhibitors: Identification of an orally available, centrally active BACE1 inhibitor", *Bioorganic Medicinal Chemistry Letters*, Vol. 22, No. 7, pp. 2444–2449, 2012, https://www.sciencedirect.com/science/article/pii/S0960894X12001965.

62. Kaller, M. R., S. S. Harried, B. Albrecht, P. Amarante, S. Babu-Khan, M. D. Bartberger, J. Brown, R. Brown, K. Chen, Y. Cheng, M. Citron, M. D. Croghan, R. Graceffa, D. Hickman, T. Judd, C. Kriemen, D. La, V. Li, P. Lopez, Y. Luo, C. Masse, H. Monenschein, T. Nguyen, L. D. Pennington, T. S. Miguel, E. A. Sickmier, R. C. Wahl, M. M. Weiss, P. H. Wen, T. Williamson, S. Wood, M. Xue, B. Yang, J. Zhang, V. Patel, W. Zhong and S. Hitchcock, "A Potent and Orally Efficacious, Hydroxyethylamine-Based Inhibitor of -Secretase", *ACS Medicinal Chemistry Letters*, Vol. 3, No. 11, pp. 886–891, 2012, https://doi.org/10.1021/ml3000148.

63. Landry, M. L. and J. J. Crawford, "LogD Contributions of Substituents Commonly Used in Medicinal Chemistry", *ACS Medicinal Chemistry Letters*, Vol. 11, No. 1, pp. 72–76, 2020, https://doi.org/10.1021/acsmedchemlett.9b00489.

64. Nepali, K., H.-Y. Lee and J.-P. Liou, "Nitro-group-containing drugs", *Journal of medicinal chemistry*, Vol. 62, No. 6, pp. 2851–2893, 2018.

65. Kovacic, P. and R. Somanathan, "Nitroaromatic compounds: Environmental toxicity, carcinogenicity, mutagenicity, therapy and mechanism", *Journal of Applied Toxicology*, Vol. 34, No. 8, pp. 810–824, 2014.

66. Gross, P. and R. P. Smith, "Biologic activity of hydroxylamine: a review", *CRC critical reviews in toxicology*, Vol. 14, No. 1, pp. 87–99, 1985.

67. Nunes, J. H., D. H. Nakahata, W. R. Lustri, P. P. Corbi and R. E. de Paiva, "The nitro-reduced metabolite of nimesulide: crystal structure, spectroscopic characterization, ESI-QTOF mass spectrometric analysis and antibacterial evaluation", *Journal of Molecular Structure*, Vol. 1157, pp. 469–475, 2018.

68. James, L. P., P. R. Mayeux and J. A. Hinson, "Acetaminophen-induced hepatotoxicity", *Drug metabolism and disposition*, Vol. 31, No. 12, pp. 1499–1506, 2003.

69. Brouwer, A., U. G. Ahlborg, M. Van den Berg, L. S. Birnbaum, E. R. Boersma, B. Bosveld, M. S. Denison, L. E. Gray, L. Hagmar, E. Holene *et al.*, "Functional aspects of developmental toxicity of polyhalogenated aromatic hydrocarbons in experimental animals and human infants", *European Journal of Pharmacology: Environmental Toxicology and Pharmacology*, Vol. 293, No. 1, pp. 1–40, 1995.

70. Boelsterli, U. A., "Mechanisms of NSAID-induced hepatotoxicity: focus on nimesulide", *Drug safety*, Vol. 25, pp. 633–648, 2002.

71. Richardson, K., K. Cooper, M. Marriott, M. Tarbit, P. Troke and P. Whittle, "Design and evaluation of a systemically active agent, fluconazole", *Annals of the New York Academy of Sciences*, Vol. 544, No. 1, pp. 4–11, 1988.

72. Richardson, K., K. Cooper, M. Marriott, M. Tarbit, F. Troke and P. Whittle, "Discovery of fluconazole, a novel antifungal agent", *Reviews of infectious diseases*, Vol. 12, No. Supplement_3, pp. S267–S271, 1990.

73. "BACE Inhibitors: Potential Treatment of Alzheimer's Disease, Dementia, and Related Neurodegenerative Disorders (B): 3-Amino-4-fluoro-1H-isoindol Derivatives", *ACS Medicinal Chemistry Letters*, Vol. 3, No. 11, pp. 869–870, 2012, https://doi.org/10.1021/ml300317n.

74. Thompson, L. A., J. Shi, C. P. Decicco, A. J. Tebben, R. E. Olson, K. M. Boy, J. M. Guernon, A. C. Good, A. Liauw, C. Zheng, R. A. Copeland, A. P. Combs, G. L. Trainor, D. M. Camac, J. K. Muckelbauer, K. A. Lentz, J. E. Grace, C. R. Burton, J. H. Toyn, D. M. Barten, J. Marcinkeviciene, J. E. Meredith, C. F. Albright and J. E. Macor, "Synthesis and in vivo evaluation of cyclic diaminopropane BACE-1 inhibitors", *Bioorganic Medicinal Chemistry Letters*, Vol. 21, No. 22, pp. 6909–6915, 2011, https://www.sciencedirect.com/science/article/pii/S0960894X11009073.

75. Thompson, L. A., K. M. Boy, J. Shi, J. E. Macor, A. C. Good and L. R. Marcin, "Substituted tetrahydroisoquinolines as -secretase inhibitors", , U.S. Patent 7902218, 2011, https://patents.google.com/patent/US7902218B2.

76. Boy, K. M., J. M. Guernon, Y.-J. Wu, Y. Zhang, J. Shi, W. Zhai, S. Zhu, S. W. Gerritz, J. H. Toyn, J. E. Meredith, D. M. Barten, C. R. Burton, C. F. Albright, A. C. Good, J. E. Grace, K. A. Lentz, R. E. Olson, J. E. Macor and L. A. Thompson, "Macrocyclic prolinyl acyl guanidines as inhibitors of -secretase (BACE)", *Bioorganic Medicinal Chemistry Letters*, Vol. 25, No. 22, pp. 5040–5047, 2015, https://www.sciencedirect.com/science/article/pii/S0960894X15301402.

77. Gerritz, S. W., W. Zhai, S. Shi, S. Zhu, J. H. Toyn, J. E. J. Meredith, L. G. Iben, C. R. Burton, C. F. Albright, A. C. Good, A. J. Tebben, J. K. Muckelbauer, D. M. Camac, W. Metzler, L. S. Cook, R. Padmanabha, K. A. Lentz, M. J. Sofia, M. A. Poss, J. E. Macor and L. A. I. Thompson, "Acyl Guanidine Inhibitors of -Secretase (BACE-1): Optimization of a Micromolar Hit to a Nanomolar Lead via Iterative Solid- and Solution-Phase Library Synthesis", *Journal of Medicinal Chemistry*, Vol. 55, No. 21, pp. 9208–9223, 2012, https://doi.org/10.1021/jm300931y.

78. Lipinski, C. A., F. Lombardo, B. W. Dominy and P. J. Feeney, "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings", *Advanced drug delivery reviews*, Vol. 23, No. 1-3, pp. 3–25, 1997.

79. Richardson, K., K. Cooper, M. Marriott, M. Tarbit, F. Troke and P. Whittle, "Discovery of fluconazole, a novel antifungal agent", *Reviews of infectious diseases*, Vol. 12, No. Supplement_3, pp. S267–S271, 1990.

80. Hastie, T., R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*, 02 2009.

81. Kuhn, M. and K. Johnson, *Applied Predictive Modeling*, 01 2013.

82. Bishop, C., *Pattern Recognition and Machine Learning*, Vol. 16, pp. 140–155, 01 2006.

83. Geron, A., *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, O'Reilly Media, Inc., 2nd edn., 2019.

APPENDIX A: Homogeneity of the Tasks

Figure A.1. Homogeneities of Classification Tasks on Heatmap.

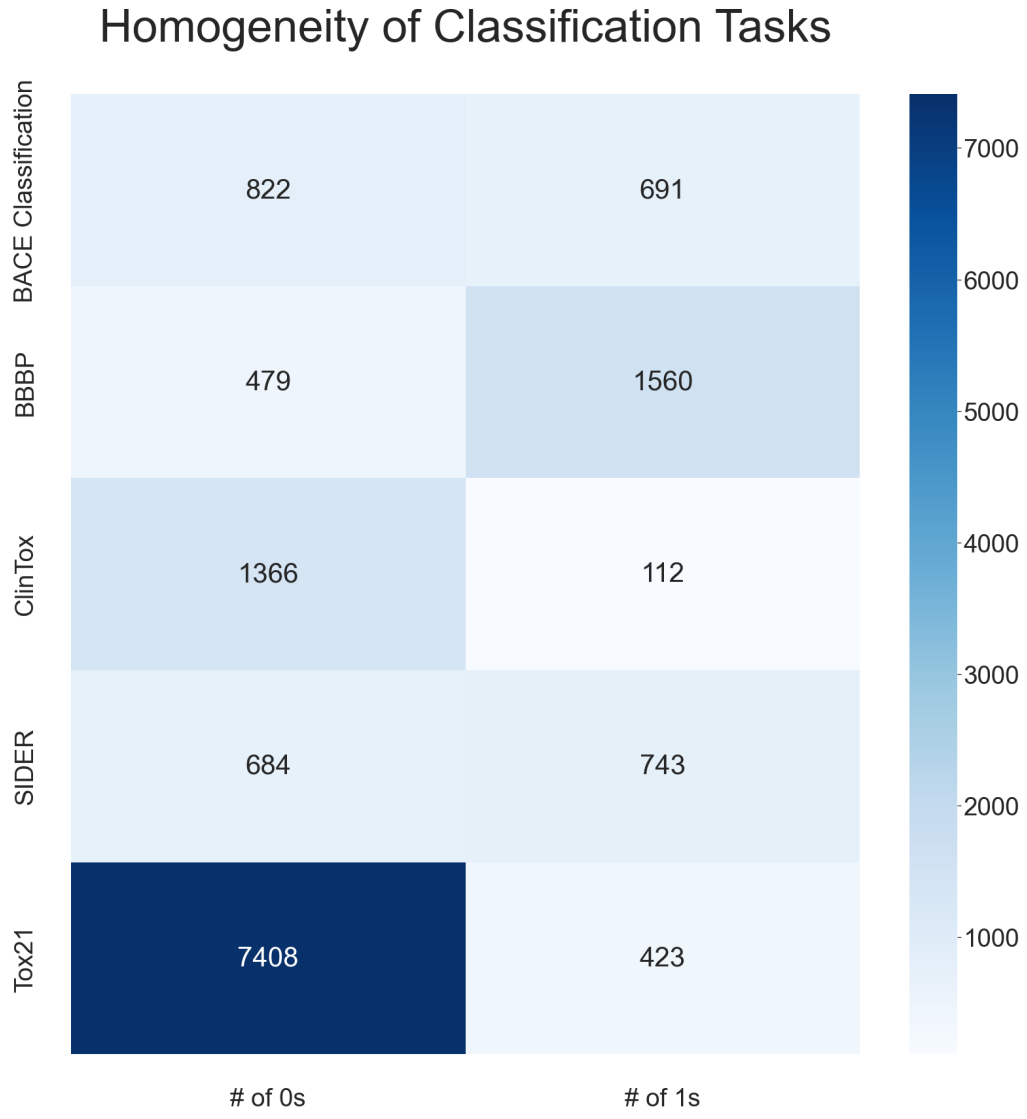
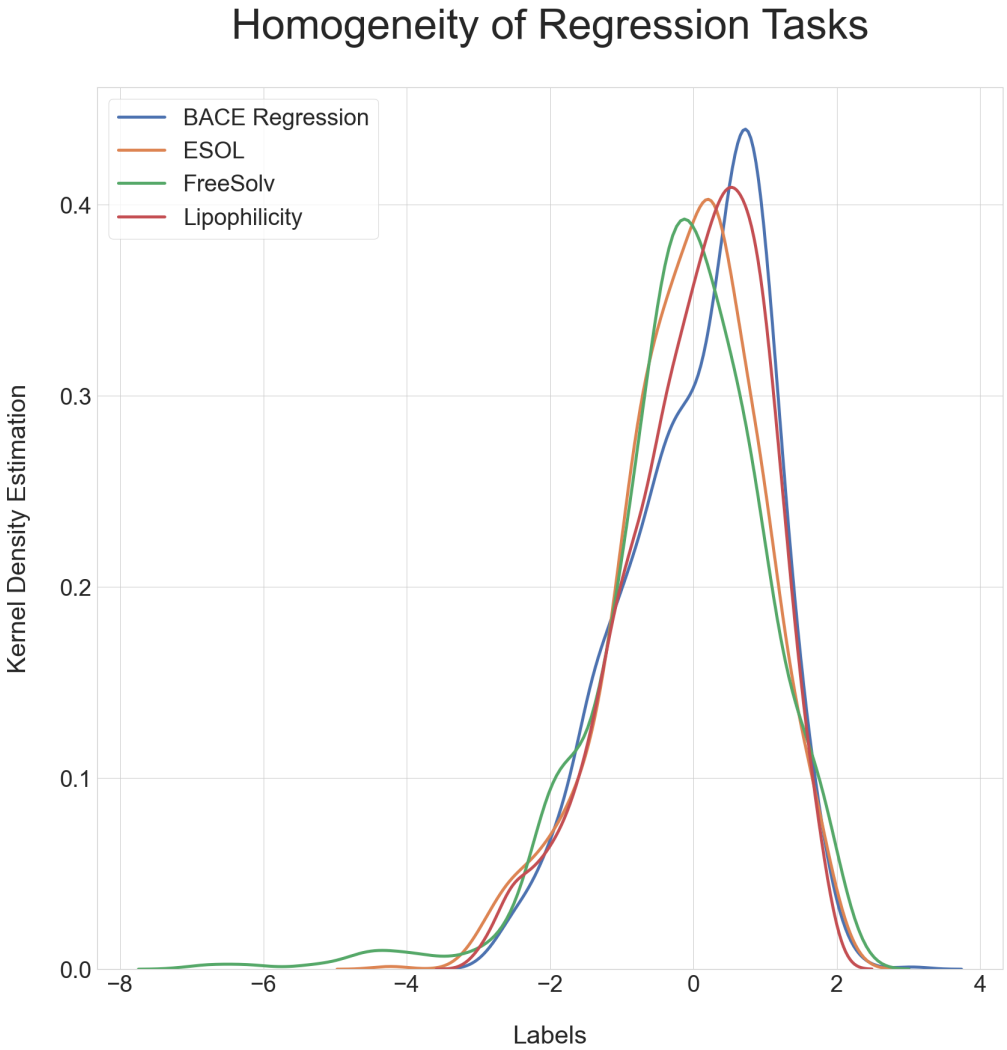


Figure A.2. Homogeneities of Regression Tasks on KDE Plot.



APPENDIX B: Scores vs Epoch Curves of the Tasks

Figure B.1. BACE Classification - Scores vs Epochs Curves 1.

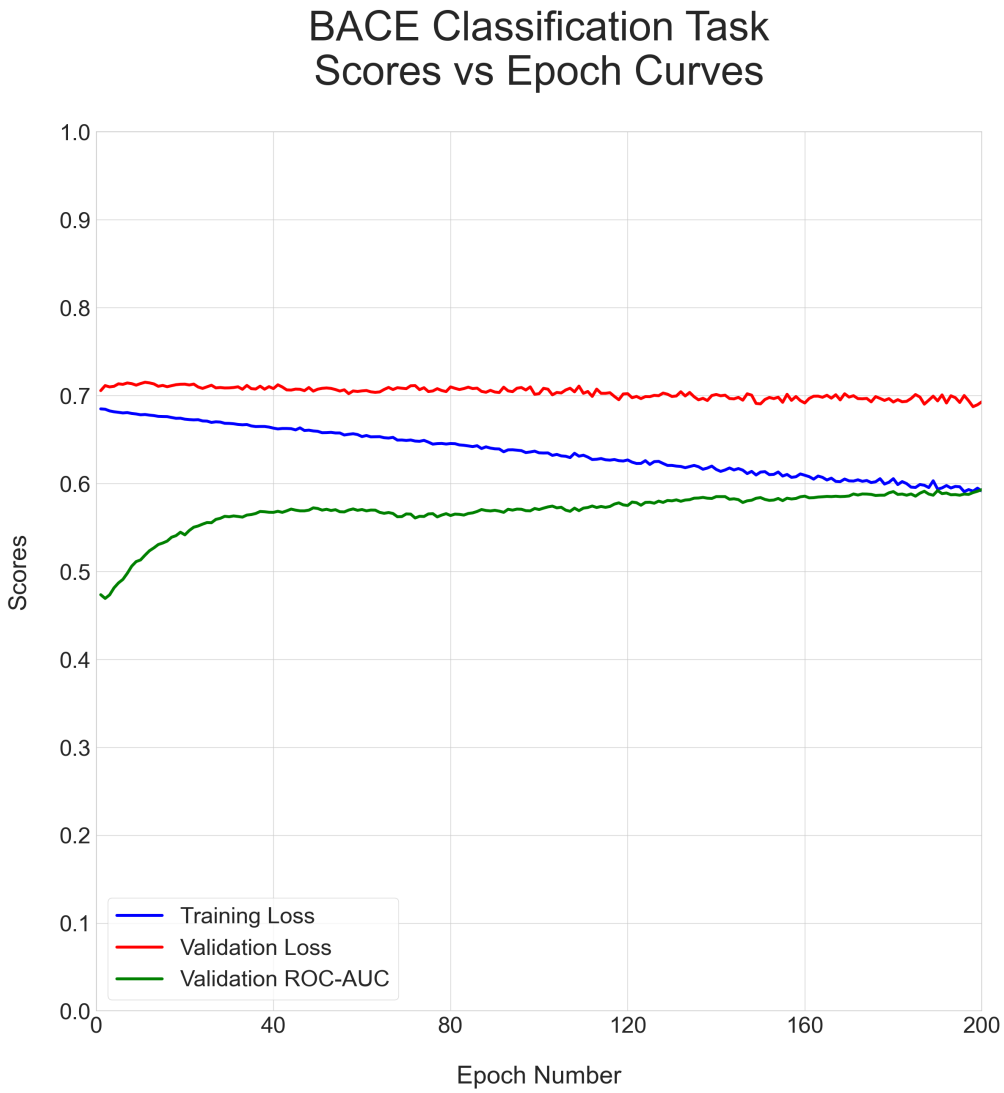


Figure B.2. BBBP - Scores vs Epochs Curves 1.

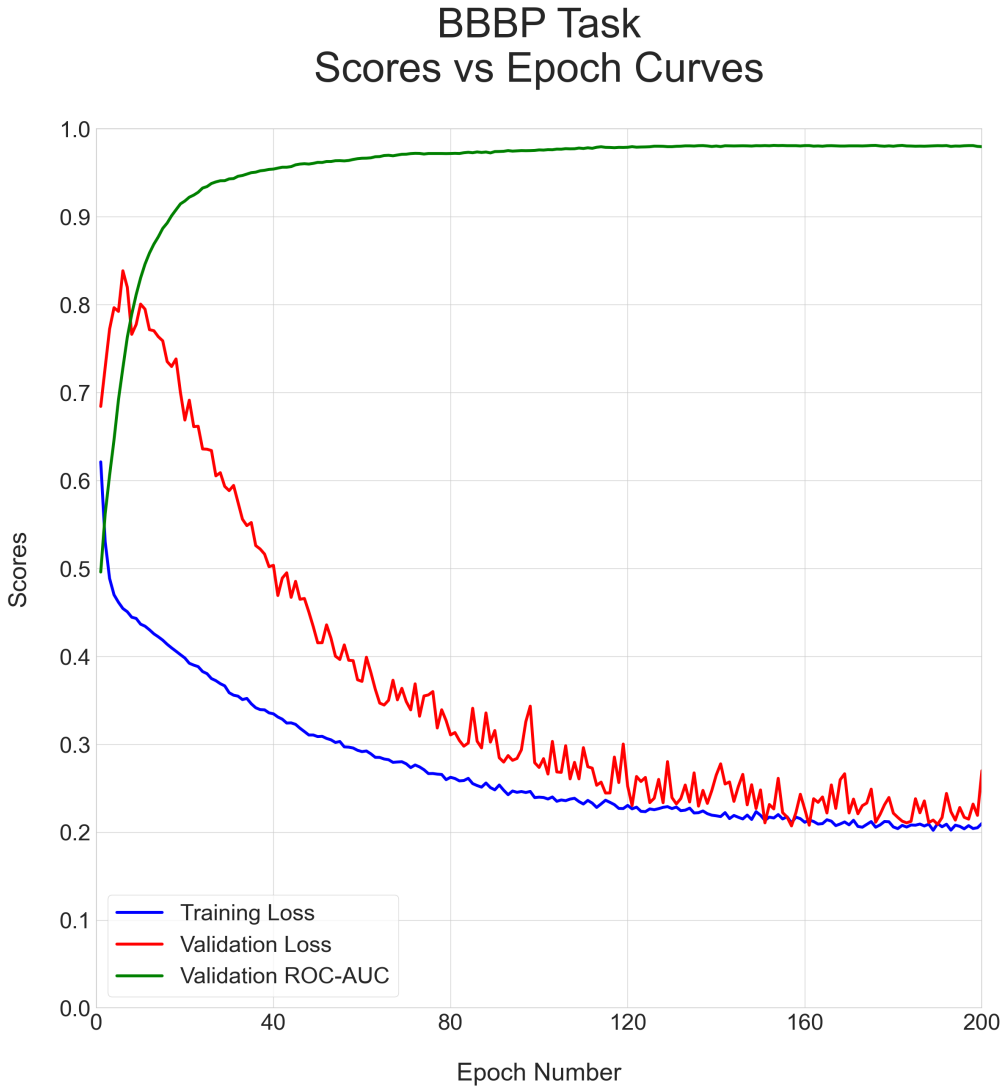


Figure B.3. ClinTox - Scores vs Epochs Curves 1.

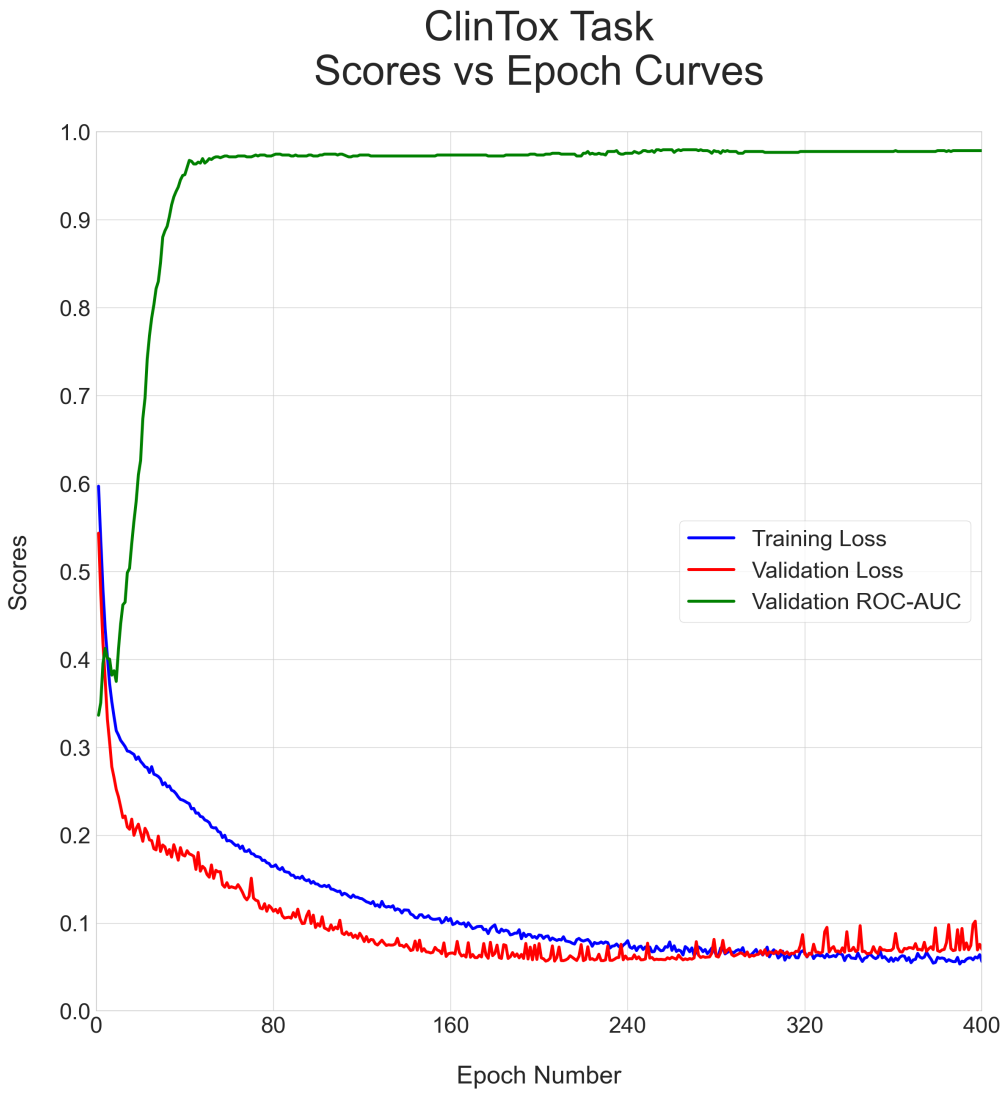


Figure B.4. SIDER - Scores vs Epochs Curves 1.

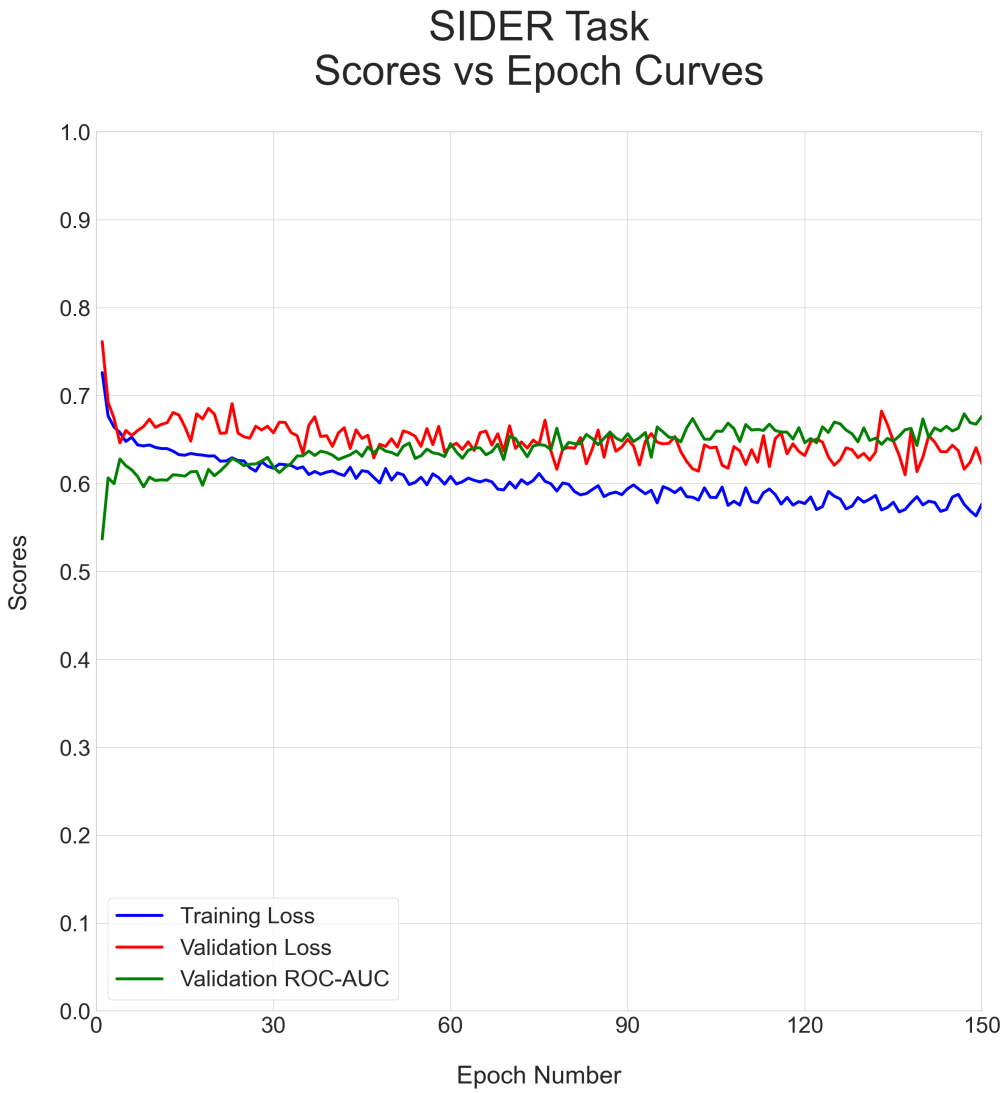


Figure B.5. Tox21 - Scores vs Epochs Curves 1.

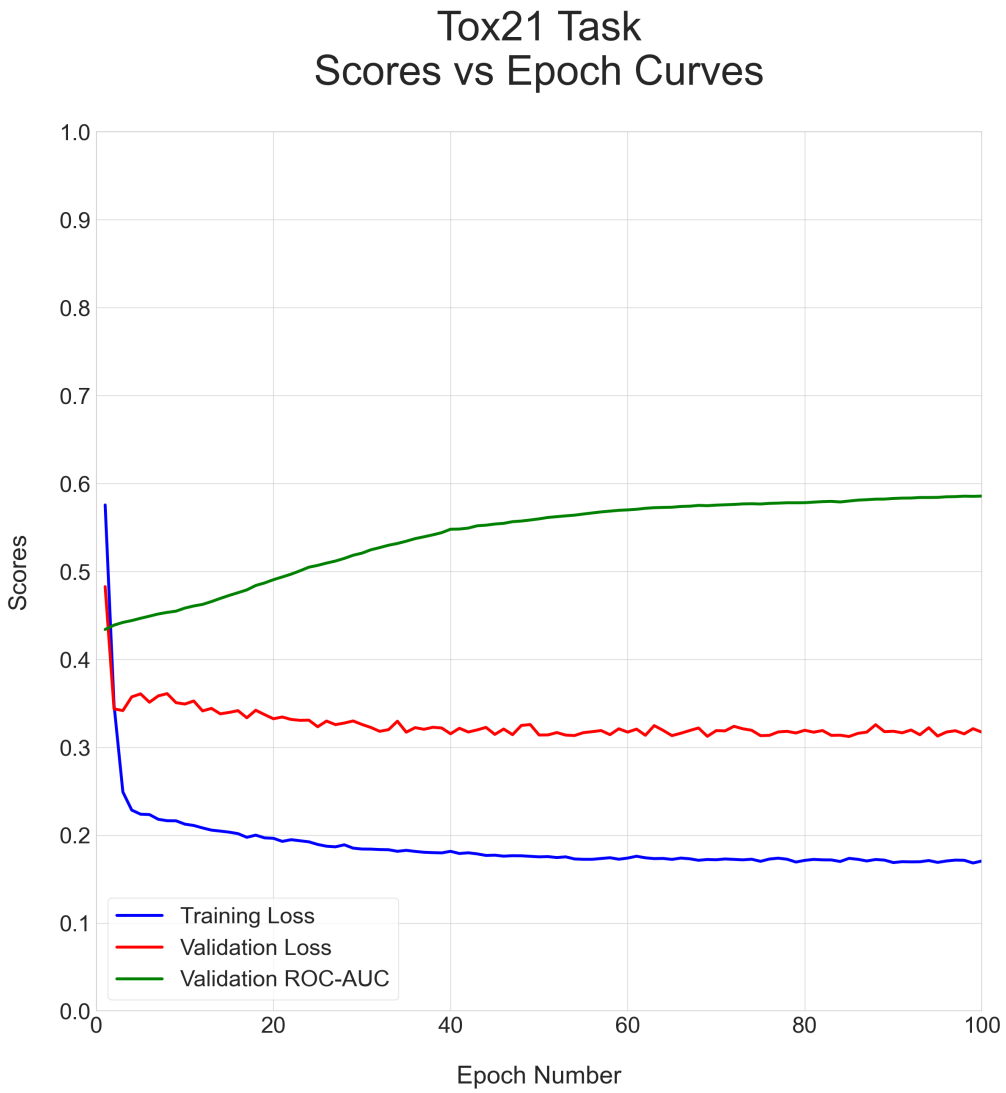


Figure B.6. BACE Classification - Scores vs Epochs Curves 2.

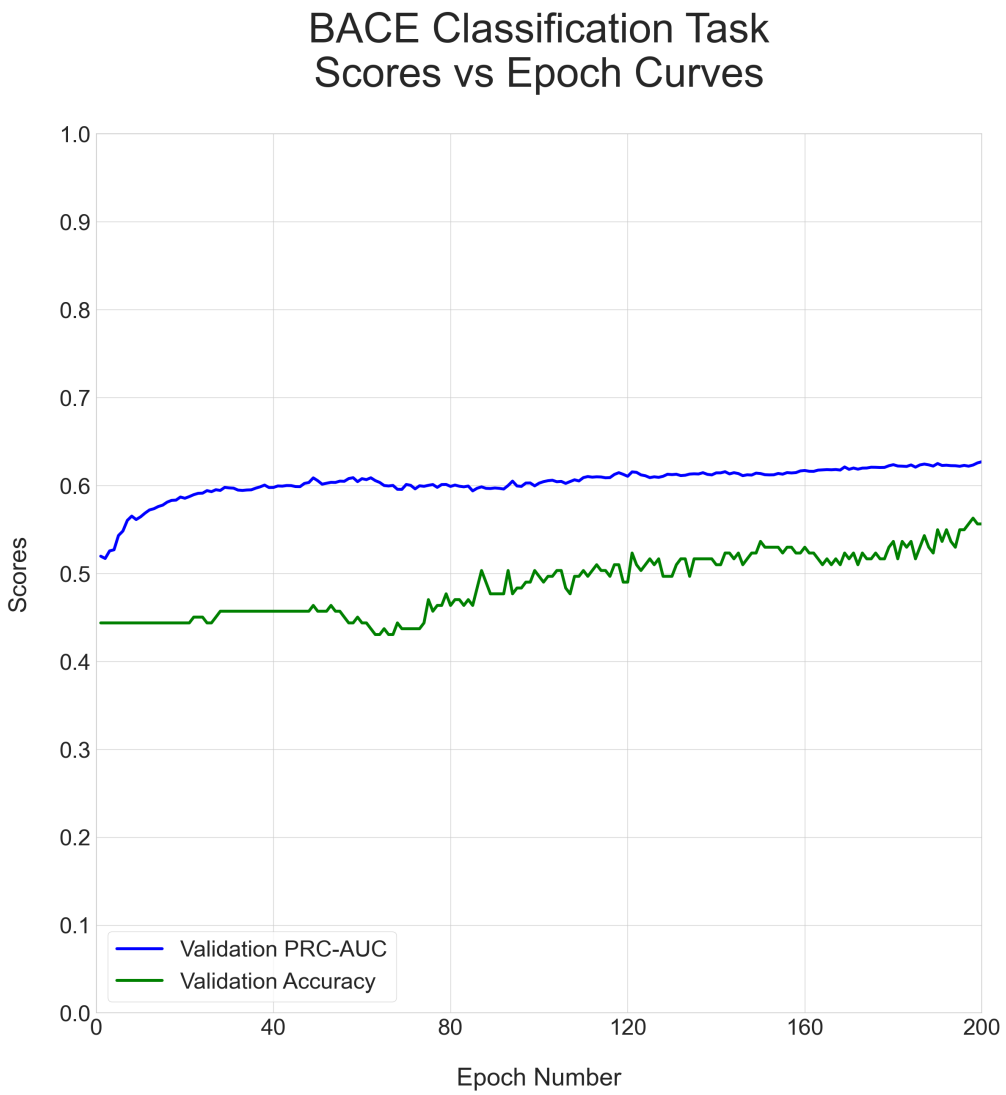


Figure B.7. BBBP - Scores vs Epochs Curves 2.

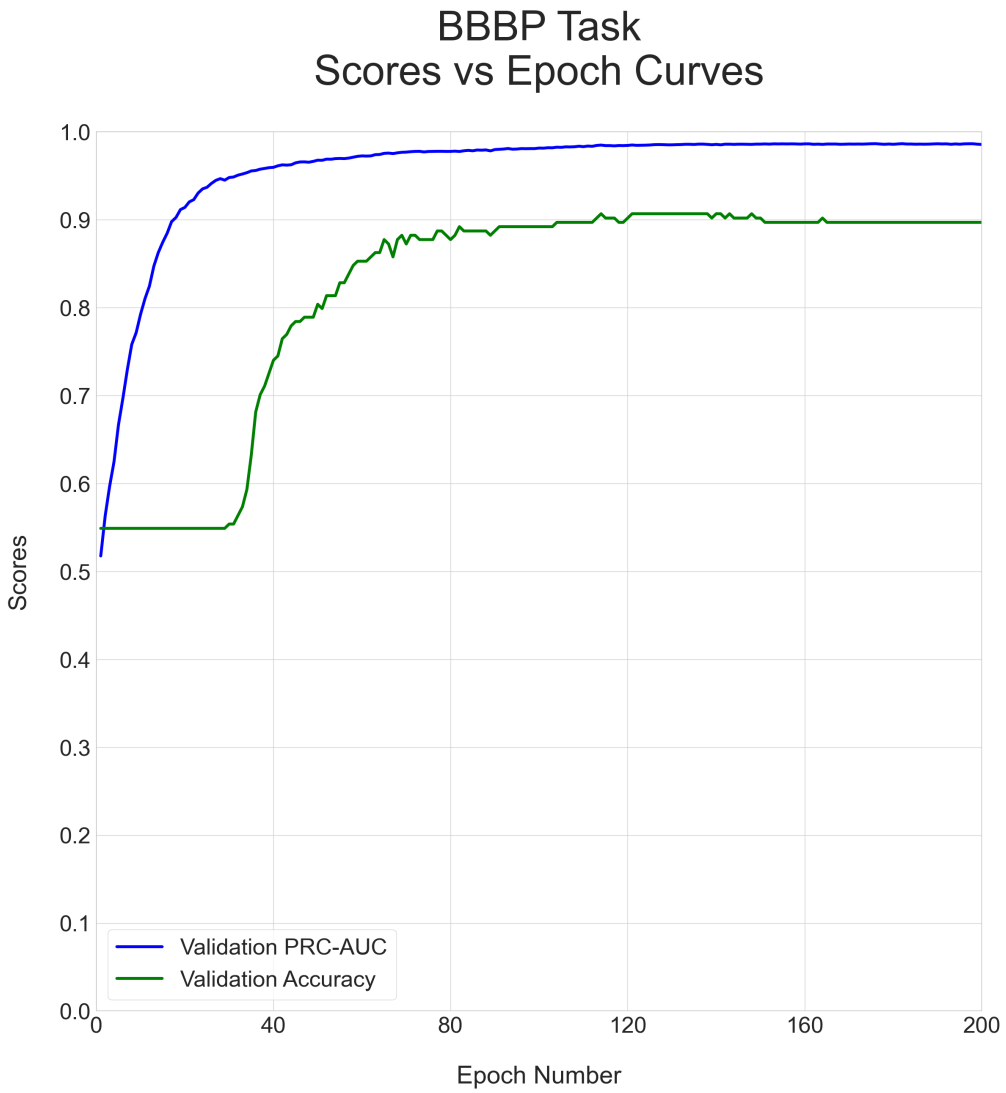


Figure B.8. ClinTox - Scores vs Epochs Curves 2.

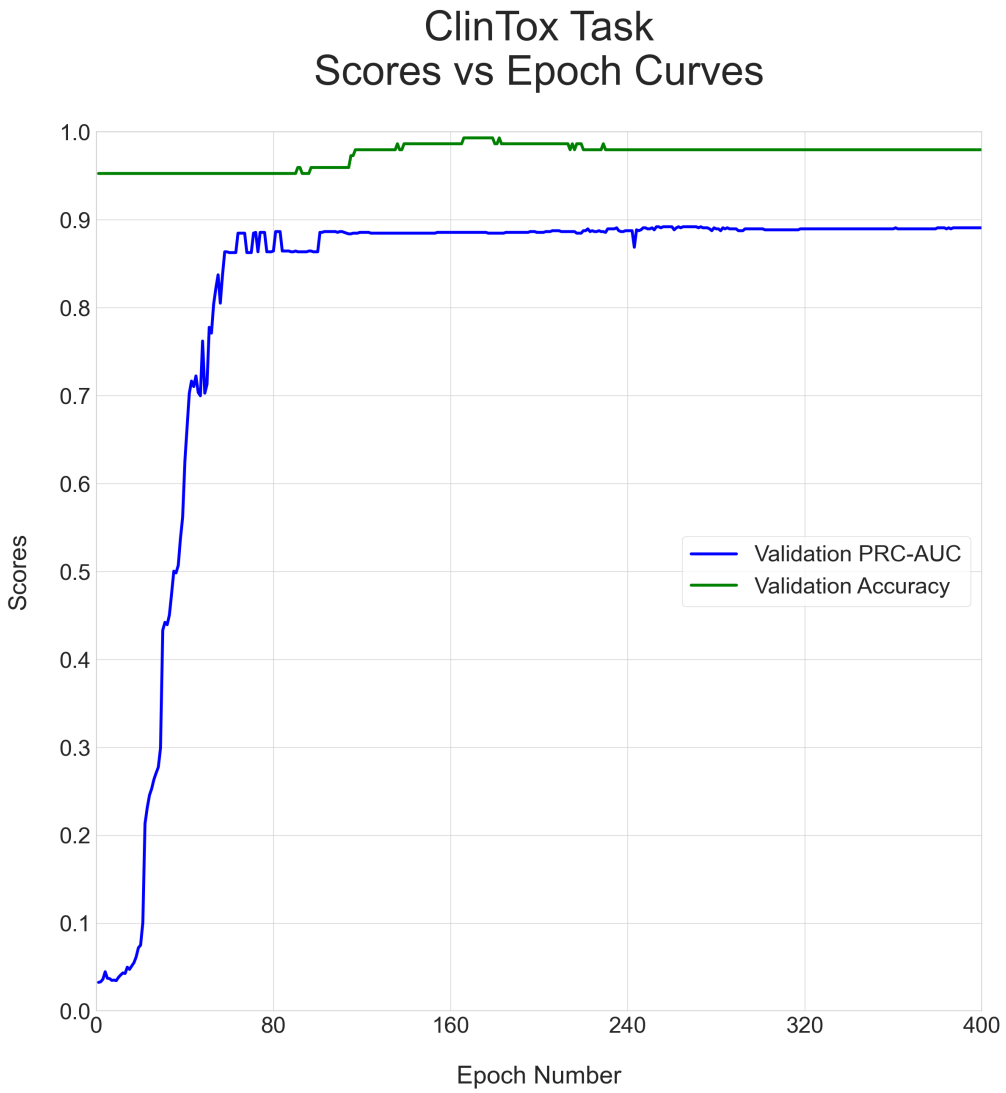


Figure B.9. SIDER - Scores vs Epochs Curves 2.

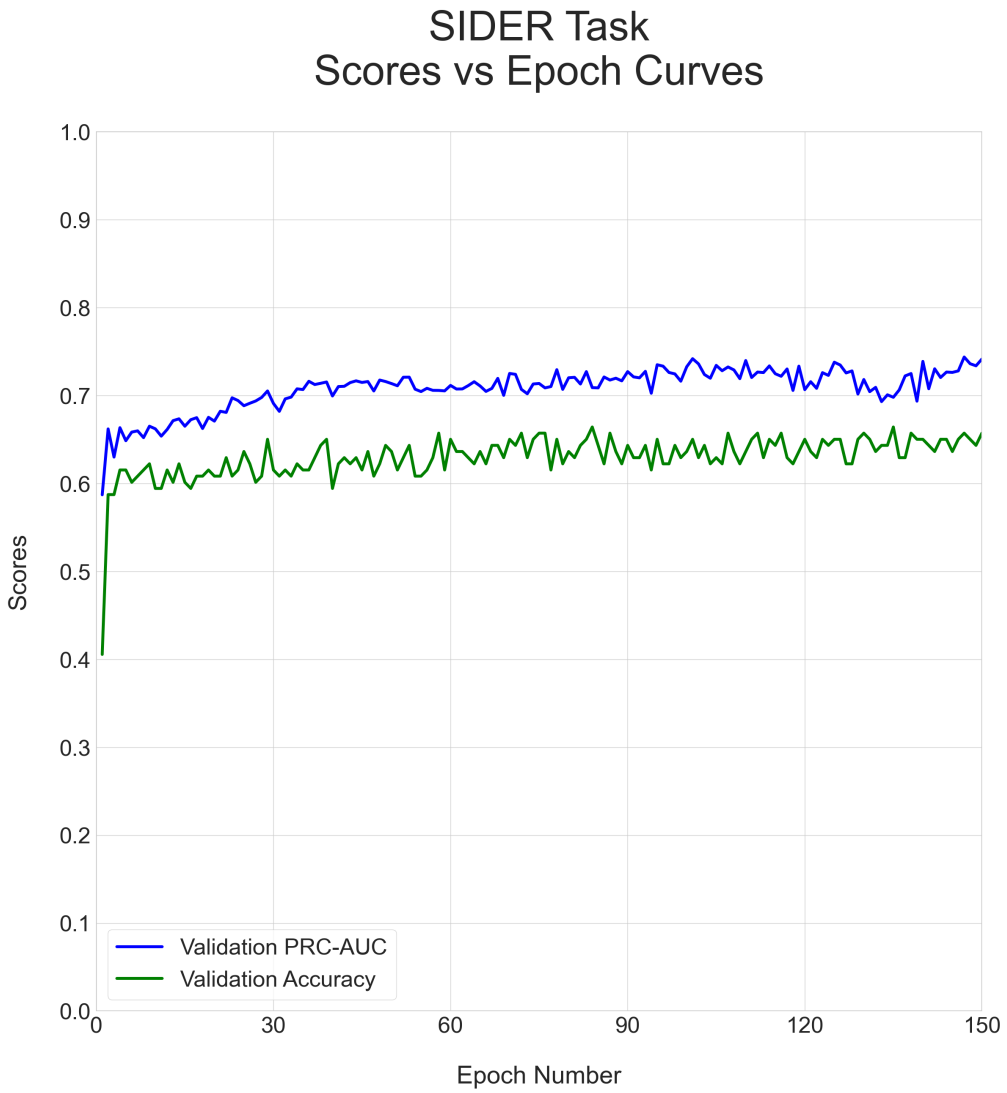


Figure B.10. Tox21 - Scores vs Epochs Curves 2.

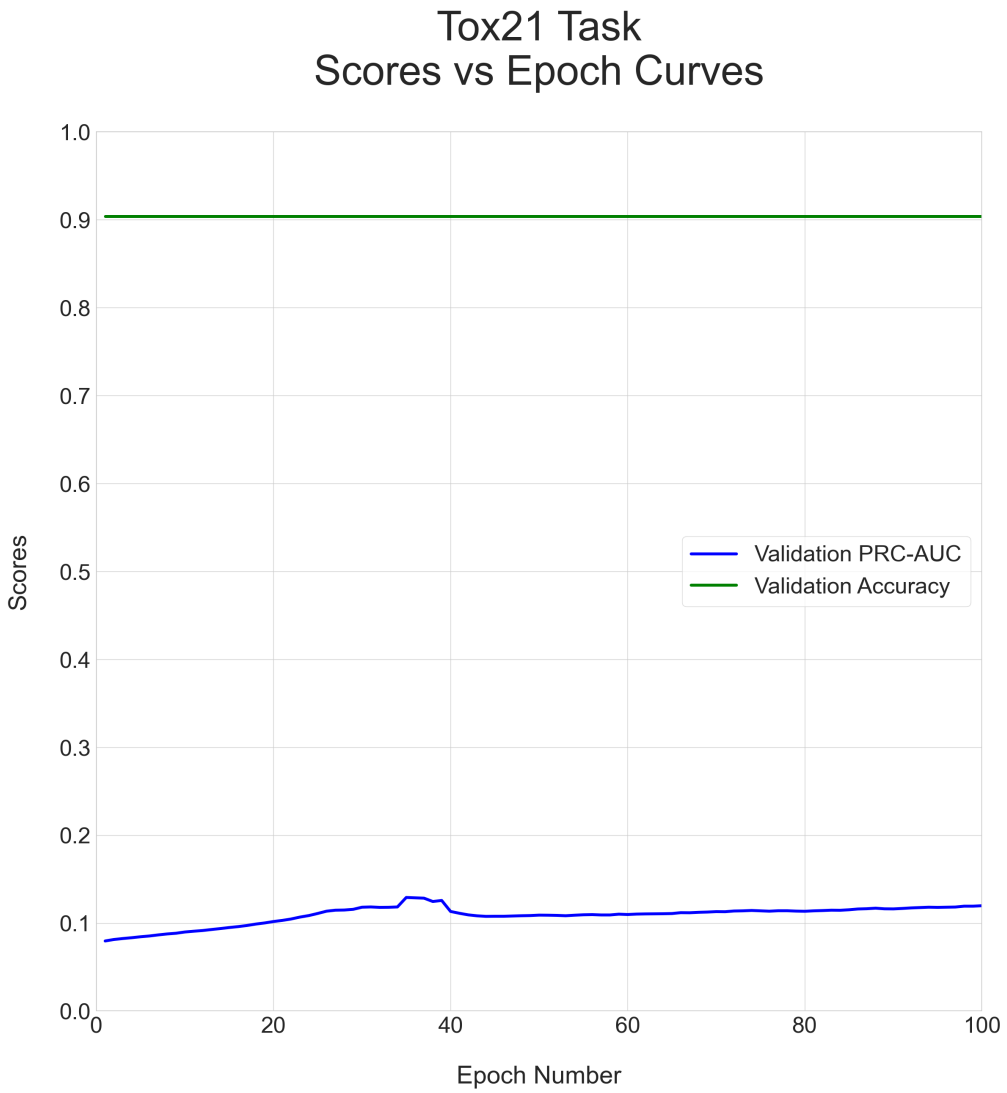


Figure B.11. BACE Regression - Scores vs Epochs Curves.

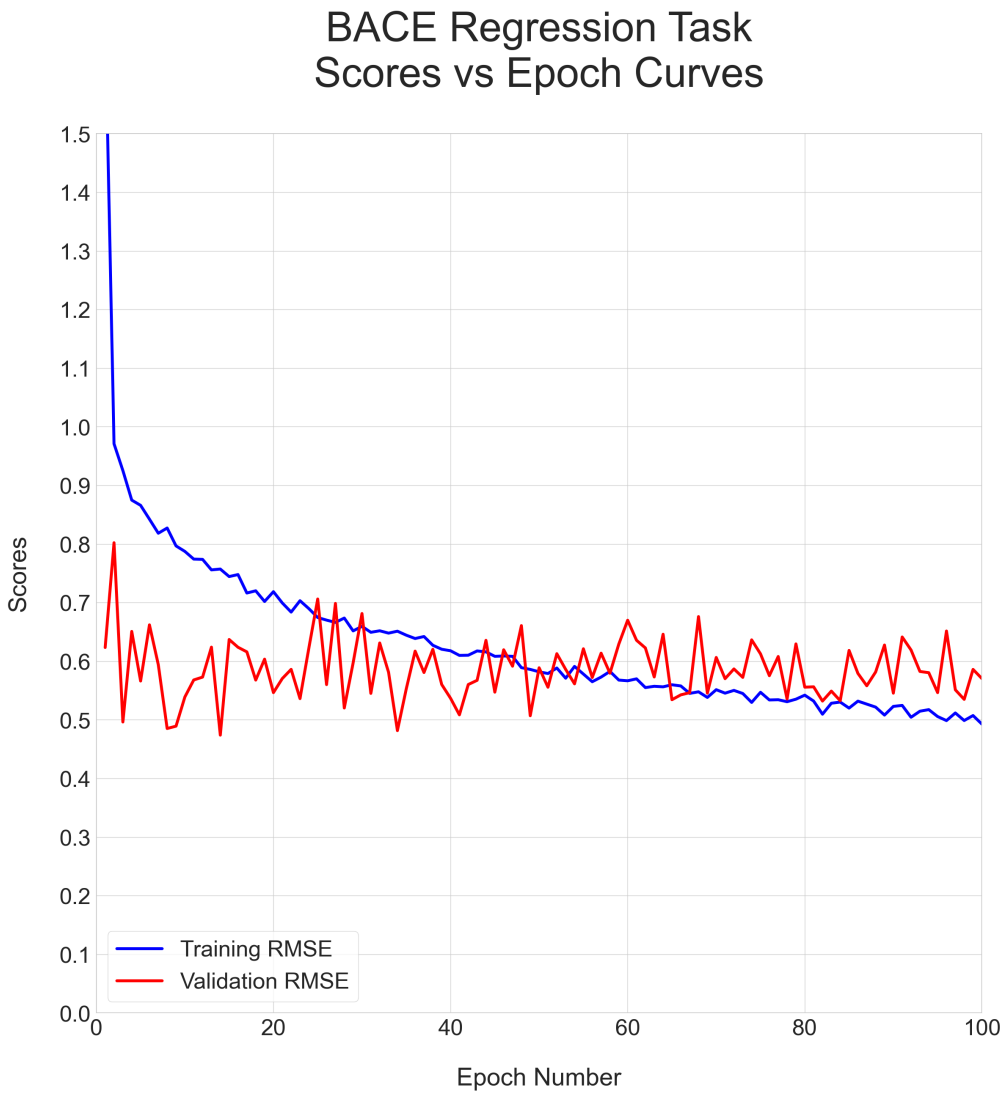


Figure B.12. ESOL - Scores vs Epochs Curves.

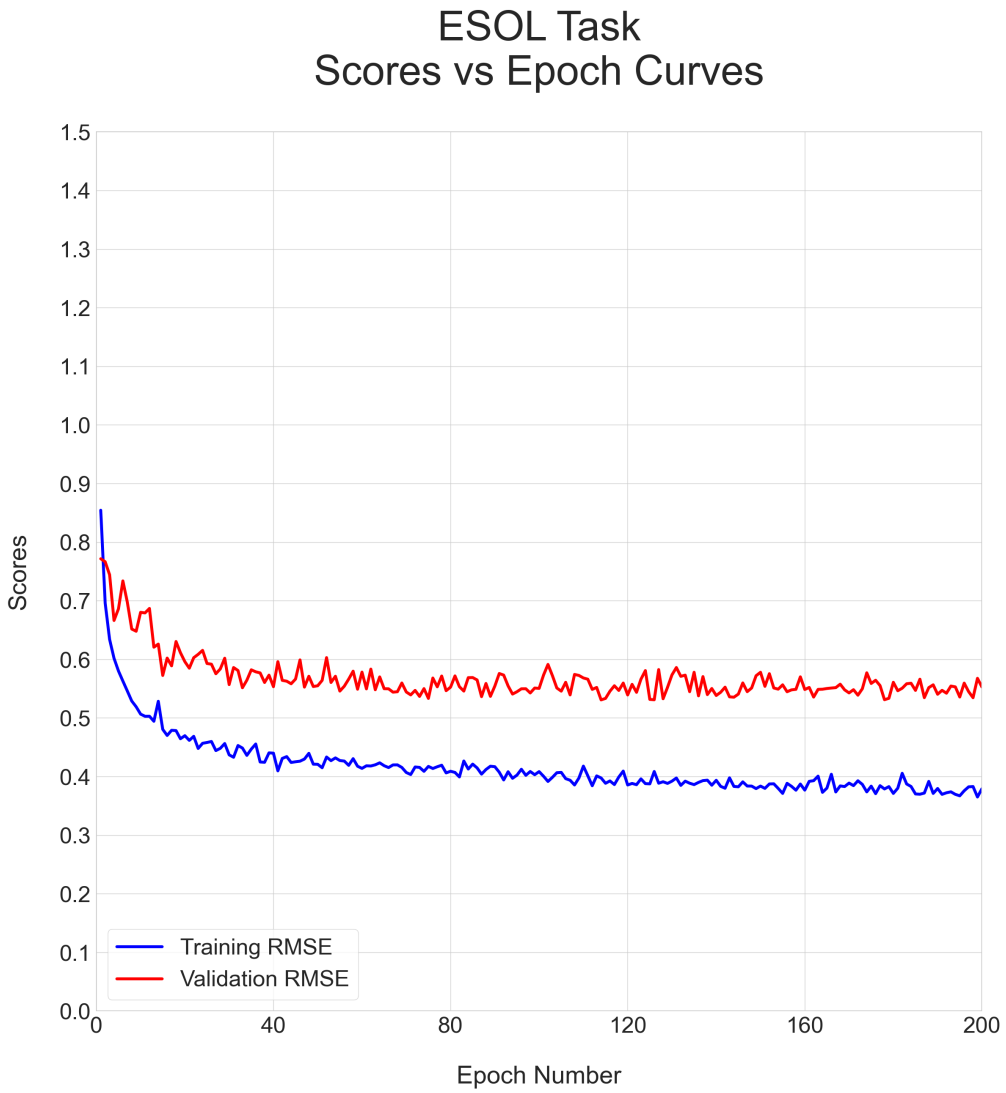


Figure B.13. FreeSolv - Scores vs Epochs Curves.

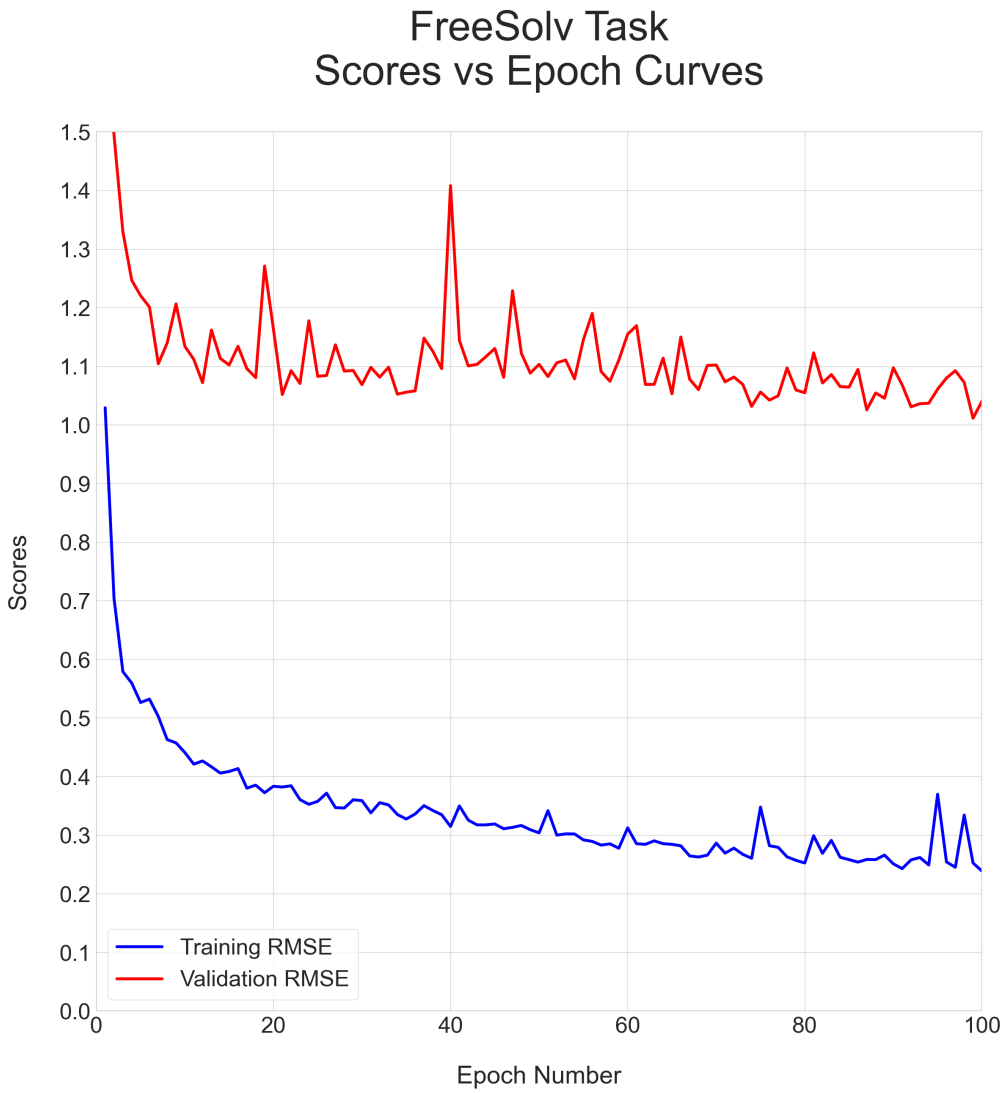
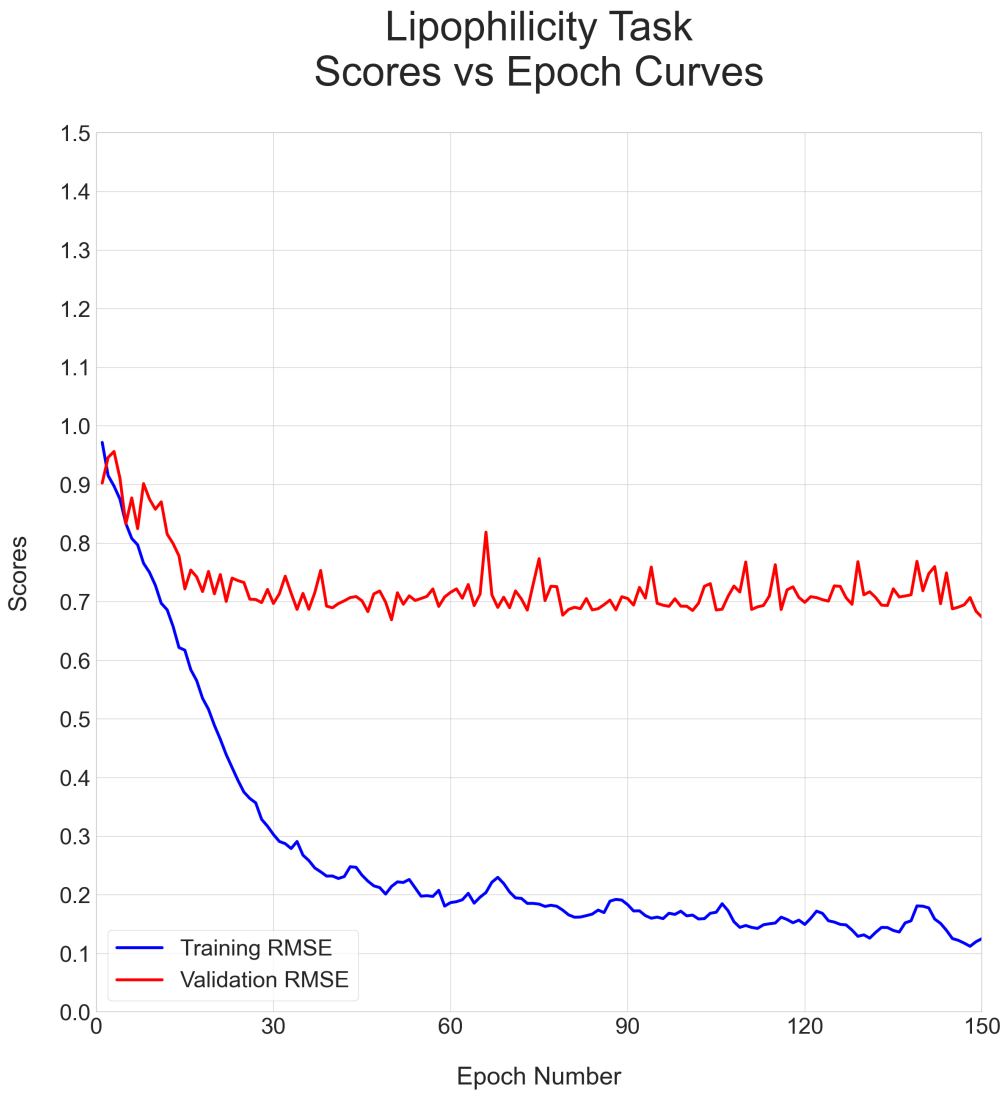


Figure B.14. Lipophilicity - Scores vs Epochs Curves.



APPENDIX C: Details of the Found Fragments

Respectively; the rank, the CID, the final point and the SMILES representation of the found fragments per task can be seen below sections in detail. Only 18 fragments could be found for FreeSolv task due to the task's small sized dataset.

C.1. Details of BACE Classification Top 20 Fragments

1 - CID None (1.0000) → **CC(C)(C)Cc1cnc2c(c1)C([NH2+]CCO)CC1(CCC1)O2**

2 - CID None (0.8505) → **C[NH2+]C1CC2(CCC2)Oc2ncc(CC(C)(C)C)cc21**

3 - CID 1140 (0.5321) → **Cc1ccccc1**

4 - CID None (0.5317) → **CC(=O)NC(C)C(O)C[NH2+]C1CC2(CCC2)Oc2ncc(CC(C)(C)C)cc21**

5 - CID None (0.3480) → **COCC(=O)NC(C)C(O)C[NH2+]C1CC2(CCC2)Oc2ncc(CC(C)(C)C)cc21**

6 - CID 712 (0.3280) → **C=O**

7 - CID 241 (0.3011) → **c1ccccc1**

8 - CID 70295272 (0.2505) → **CN1C(=O)[C](c2ccc(OC(F)F)cc2)N=C1N**

9 - CID 713 (0.2314) → **NC=O**

10 - CID 6360 (0.1885) → **CC(C)C**

11 - CID None (0.1807) → **CC(C)(C)Cc1cnc2c(c1)C([NH2+]CC(O)C(Cc1ccc3c(c1)OCO3)NC=O)CC1(CCC1)O2**

12 - CID 177 (0.1624) → **CC=O**

13 - CID 31254 (0.1440) → **CNC=O**

14 - CID 6373 (0.1378) → **FC(F)F**

15 - CID 10041 (0.1371) → **CC(C)(C)C**

16 - CID 68015 (0.1334) → **FC(F)Oc1ccccc1**

17 - CID None (0.1115) → **CCn1cc(C2N=C(N)N(C)C2=O)cn1**

18 - CID 2778329 (0.1093) → **Cc1cc(F)cc(F)c1**

19 - CID 28573746 (0.1063) → **OCC[NH2+]Cc1cccc(C(F)(F)F)c1**

20 - CID 6334 (0.1023) → **CCC**

C.2. Details of BBBP Top 20 Fragments

1 - CID 241 (1.0000) → **c1ccccc1**

2 - CID 7845 (0.7546) → **C = CC = C**

3 - CID 8252 (0.6830) → **C = CC**

4 - CID 6325 (0.6787) → **C = C**

5 - CID 712 (0.5087) → **C = O**

6 - CID 177 (0.4974) → **CC = O**

7 - CID 15301 (0.4655) → **C = CC = CC**

8 - CID 6341 (0.4355) → **CCN**

9 - CID 6334 (0.4285) → **CCC**

10 - CID 12220 (0.3838) → **CC = CC**

11 - CID 674 (0.3726) → **CNC**

12 - CID 1140 (0.3575) → **Cc1ccccc1**

13 - CID 1146 (0.3341) → **CN(C)C**

14 - CID 11723 (0.3088) → **CCN(C)C**

15 - CID 12219 (0.2941) → **CCNC**

16 - CID 7843 (0.2749) → **CCCC**

17 - CID 702 (0.2624) → **CCO**

18 - CID 713 (0.2109) → **NC = O**

19 - CID 61236 (0.1869) → **CCCN(C)C**

20 - CID 176 (0.1760) → **CC(= O)O**

C.3. Details of ClinTox Top 20 Fragments

1 - CID 177 (1.0000) → **CC = O**

2 - CID 8252 (0.8864) → **C = CC**

3 - CID 713 (0.8436) → **NC = O**

4 - CID 54909138 (0.7412) → **CNC(=O)c1cc(Oc2ccc(NC)cc2)ccn1**

5 - CID 91585 (0.5978) → **O=C1CCC(N2Cc3ccccc3C2=O)C(=O)N1**

6 - CID 6325 (0.4544) → **C=C**

7 - CID 6334 (0.3743) → **CCC**

8 - CID 12220 (0.3203) → **CC=CC**

9 - CID 6341 (0.2737) → **CCN**

10 - CID 6360 (0.1937) → **CC(C)C**

11 - CID 10903 (0.1527) → **CCOC**

12 - CID 8254 (0.1471) → **COC**

13 - CID 7843 (0.1248) → **CCCC**

14 - CID 8003 (0.1024) → **CCCCC**

15 - CID 702 (0.0894) → **CCO**

16 - CID 62695 (0.0708) → **C/C=C/C**

17 - CID 12219 (0.0633) → **CCNC**

18 - CID 21585139 (0.0503) → **C.O**

19 - CID 3283 (0.0428) → **CCOCC**

20 - CID 24529 (0.0279) → **O=[N+][O−]**

C.4. Details of SIDER Top 20 Fragments

1 - CID 241 (1.0000) → **c1ccccc1**

2 - CID 712 (0.7787) → **C=O**

3 - CID 713 (0.6064) → **NC=O**

4 - CID 1140 (0.5599) → **Cc1ccccc1**

5 - CID 284 (0.5536) → **O=CO**

6 - CID 6325 (0.3996) → **C=C**

7 - CID 6334 (0.3936) → **CCC**

8 - CID 177 (0.3813) → **CC=O**

9 - CID 12220 (0.3667) → **CC=CC**

10 - CID 31254 (0.3225) → **CNC=O**

11 - CID 178 (0.3111) → **CC(N)=O**

12 - CID 6341 (0.2156) → **CCN**

13 - CID 7843 (0.2001) → **CCCC**

14 - CID 176 (0.1878) → **CC(=O)O**

15 - CID 674 (0.1717) → **CNC**

16 - CID 74687419 (0.1552) → **CON=C(C(=O)NC1C(=O)N2C(C(=O)[O−])=C(C)CSC12)c1csc(N)n1**

17 - CID 8003 (0.1540) → **CCCCC**

18 - CID 6115 (0.1511) → **Nc1ccccc1**

19 - CID 702 (0.1413) → **CCO**

20 - CID 59900860 (0.1378) → **CCOC(=O)C(CCc1ccccc1)NC**

C.5. Details of Tox21 Top 20 Fragments

1 - CID 241 (1.0000) → **c1ccccc1**

2 - CID 284 (0.3565) → **O=CO**

3 - CID 11345724 (0.1913) → **O=C(O)CC[C@H](NC(=O)c1ccccc1)C(=O)O**

4 - CID 101652 (0.1826) → **FC(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)F**

5 - CID 7964 (0.1652) → **Clc1ccccc1**

6 - CID 9778 (0.1391) → **FC(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)F**

7 - CID 6373 (0.1304) → **FC(F)F**

8 - CID 6334 (0.1304) → **CCC**

9 - CID 67730 (0.1043) → **FC(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)F**

10 - CID 996 (0.1043) → **Oc1ccccc1**

11 - CID 6115 (0.1043) → **Nc1ccccc1**

12 - CID 72858 (0.0870) → **Oc1ccc(I)cc1I**

13 - CID 176 (0.0870) → **CC(=O)O**

14 - CID 6341 (0.0870) → **CCN**

15 - CID 9816874 (0.0783) → **FC(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)F**

16 - CID 7368 (0.0783) → **FC(F)(F)c1ccccc1**

17 - CID 702 (0.0783) → **CCO**

18 - CID 6334 (0.0783) → **[CH]CC**

19 - CID 7564 (0.0696) → **Nc1ccc(Nc2ccccc2)cc1**

20 - CID 6253 (0.0696) → **Nc1ccn([C@@H]2O[C@H](CO)[C@@H](O)[C@@H]2O)c(=O)n1**

C.6. Details of BACE Regression Top 20 Fragments

1 - CID 241 (1.0000) → **c1ccccc1**

2 - CID 1140 (0.9578) → **Cc1ccccc1**

3 - CID 7500 (0.4057) → **CCc1ccccc1**

4 - CID None (0.3324) → **CCn1cc(C2(c3ccccc3)N=C(N)c3ccccc32)cc(C)c1=O**

5 - CID 702 (0.3065) → **CCO**

6 - CID 7530 (0.2640) → **COc1cccc(C)c1**

7 - CID 7519 (0.2634) → **COc1ccccc1**

8 - CID None (0.2563) → **CCC(C)C1(NC(C)=O)CCN(C(CCc2ccccc2)C(=O)NC(Cc2cccc(F)c2)C(O)C2Cc3cccc(OC)c3C[NH2+]2)C1=O**

9 - CID None (0.2539) → **CCC(C)C1(NC(C)=O)CCN(C(CCc2ccccc2)C(=O)NC(Cc2cccc(F)c2)C(O)C2Cc3cccc(O)c3C[NH2+]2)C1=O**

10 - CID None (0.2513) → **CCC(C)C1(NC(C)=O)CCN(C(CCc2ccccc2)C(=O)NC(Cc2cccc(F)c2)C(O)C2Cc3ccccc3C[NH2+]2)C1=O**

11 - CID 61302 (0.2471) → **CC(O)CCc1ccccc1**

12 - CID 6360 (0.2439) → **CC(C)C**

13 - CID 7668 (0.2073) → **CCCc1ccccc1**

14 - CID 14461 (0.1807) → **CN(C)C(=N)N**

15 - CID 7929 (0.1712) → **Cc1cccc(C)c1**

16 - CID None (0.1606) → **CCn1cc(C2(c3cccc(−c4cccnc4)c3)N=C(N)N(C)C2=O)cn1**

17 - CID 6334 (0.1565) → **CCC**

18 - CID None (0.1456) → **CN1C(=O)[C@@](c2ccc(OC(F)F)cc2)(c2cccc(OCCCF)c2)N=C1N**

19 - CID None (0.1429) → **CCCCc1cccc([C@@]2(c3ccc(OCF)cc3)N=C(N)N(C)**

C2 = **O**)**c1**

20 - CID None (0.1394) → **CCCc1cccc**([**C@@**]**2**(**c3ccc**(**OCF**)**cc3**)**N** = **C**(**N**)**N**(**C**)**C2** = **O**)**c1**

C.7. Details of ESOL Top 20 Fragments

1 - CID 996 (1.0000) → **Oc1ccccc1**

2 - CID 241 (0.9630) → **c1ccccc1**

3 - CID 6334 (0.7809) → **CCC**

4 - CID 1030 (0.5216) → **CC**(**O**)**CO**

5 - CID 702 (0.4198) → **CCO**

6 - CID 10903 (0.4167) → **CCOC**

7 - CID 8254 (0.4074) → **COC**

8 - CID 7843 (0.3735) → **CCCC**

9 - CID 6115 (0.3549) → **Nc1ccccc1**

10 - CID 6556 (0.3117) → **CCC**(**C**)**C**

11 - CID 229 (0.3056) → **OC1COC**(**O**)**C**(**O**)**C1O**

12 - CID 3776 (0.3056) → **CC**(**C**)**O**

13 - CID 6360 (0.2747) → **CC**(**C**)**C**

14 - CID 8998 (0.2438) → **OCC**(**O**)**C**(**O**)**CO**

15 - CID 7964 (0.2407) → **Clc1ccccc1**

16 - CID 1031 (0.2346) → **CCCO**

17 - CID 1140 (0.2191) → **Cc1ccccc1**

18 - CID 712 (0.2099) → **C** = **O**

19 - CID 753 (0.2006) → **OCC**(**O**)**CO**

20 - CID 9261 (0.1975) → **c1cnccn1**

C.8. Details of FreeSolv Top 20 Fragments

1 - CID 7843 (1.0000) → **CCCC**

2 - CID 6334 (0.9461) → **CCC**

3 - CID 8003 (0.9079) → **CCCCC**

4 - CID 8058 (0.8652) → **CCCCCC**

5 - CID 6556 (0.6809) → **CCC(C)C**

6 - CID 8900 (0.6787) → **CCCCCCC**

7 - CID 6360 (0.6719) → **CC(C)C**

8 - CID 356 (0.4764) → **CCCCCCCC**

9 - CID 10041 (0.4562) → **CC(C)(C)C**

10 - CID 6403 (0.3169) → **CCC(C)(C)C**

11 - CID 7892 (0.3079) → **CCCC(C)C**

12 - CID 8252 (0.2989) → **C = CC**

13 - CID 8141 (0.2000) → **CCCCCCCCC**

14 - CID 6325 (0.1685) → **C = C**

15 - CID 8255 (0.1416) → **C = C(C)C**

16 - CID 6373 (0.1191) → **FC(F)F**

17 - CID 7843 (0.0966) → **C[CH]CC**

18 - CID 7844 (0.0539) → **C = CCC**

C.9. Details of Lipophilicity Top 20 Fragments

1 - CID 241 (1.0000) → **c1ccccc1**

2 - CID 713 (0.5263) → **NC = O**

3 - CID 712 (0.5078) → **C = O**

4 - CID 1140 (0.4339) → **Cc1ccccc1**

5 - CID 7964 (0.2120) → **Clc1ccccc1**

6 - CID 177 (0.2112) → **CC = O**

7 - CID 31254 (0.1782) → **CNC = O**

8 - CID 6334 (0.1534) → **CCC**

9 - CID 10008 (0.1452) → **Fc1ccccc1**

10 - CID 1049 (0.1238) → **c1ccncc1**

11 - CID 178 (0.1191) → **CC(N) = O**

12 - CID 996 (0.1127) → **Oc1ccccc1**

13 - CID 6373 (0.1089) → **FC(F)F**

14 - CID 6115 (0.1078) → **Nc1ccccc1**

15 - CID 3452818 (0.1019) → **O = C(NCC12CC3CC(CC(C3)C1)C2)c1ccccc1**

16 - CID 6341 (0.0909) → **CCN**

17 - CID 11569158 (0.0890) → **C[C@@H](Oc1cccc2ncnc(Nc3ccc(OCc4ccccn4)c(Cl)c3)c12)C(= O)N(C)C**

18 - CID None (0.0861) → **CN(C)C(= O)[CH]Oc1cccc2ncnc(Nc3ccc(OCc4cccn4)c(Cl)c3)c12**

19 - CID 25142173 (0.0859) → **NCC1(NC(= O)[C@@H]2CCCC[C@H]2C(= O)N2CCc3[nH]c4ccccc4c3C2)CC1**

20 - CID 674 (0.0838) → **CNC**