INTERPRETATION OF COMPOUND FRAGMENTS VIA ATTENTIVE RECURSIVE TREE

by

Nural Özel

B.S., Mechanical Engineering, Istanbul Technical University, 2019

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)	.....
Assoc. Prof. Burak Alakent	.....
Prof. Fikret Gürgen	.....
Assoc. Prof. Özge Kürkçüoğlu Levitas	.....

DATE OF APPROVAL: 29.05.2023

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 heartfelt 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, both are doctoral candidates at the Department of Pharmaceutical Chemistry of Istanbul University, for all of their assistance within the "Interpretation" and the "Discussion" chapters of this work.

Last but not least, my dear family, I am well aware of those endeavors of yours to make easier this steep and thorny road of mine for me, as much as you can. I thank and owe you for everything that I have, 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

DİKKATLİ ÖZÇAĞRILI AĞAÇ İLE KİMYASAL FRAGMANLARININ ANLAMLANDIRILMASI

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

b	Beginning index of the sentence
$\mathbf{b_c}$	Bias matrix of the cell state of the Tree-LSTM layer
$\mathbf{c}$	Cell state of the Tree-LSTM layer
$\mathbf{c_i}$	Cell state of the Tree-LSTM layer in the parent node
$\overrightarrow{\mathbf{c_i}}$	Cell state of the right Tree-LSTM layer
$\overleftarrow{\mathbf{c_i}}$	Cell state of the left Tree-LSTM layer
$\mathbf{d_{max}}$	Distance between the fartest leaf node from the root
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 Tree-LSTM layer in the parent node
$\mathbf{f_L}$	The gate of the Tree-LSTM layer in the left child node
$\mathbf{f_R}$	The gate of the Tree-LSTM layer in the right child node
FS	Fragment set
$\mathbf{g}$	The candidate vector in the Tree-LSTM layer
$\mathbf{h}$	Hidden vector of the Tree-LSTM layer
$\mathbf{h_i}$	Hidden vector of the Tree-LSTM layer in the parent node
$\mathbf{h_L}$	Hidden vector of the Tree-LSTM layer in the left child node
$\mathbf{h_R}$	Hidden vector of the Tree-LSTM layer in the right child node
$\overleftarrow{\mathbf{h_{i-1}}}$	Hidden vector of the previous left Tree-LSTM layer
$\overrightarrow{\mathbf{h_{i-1}}}$	Hidden vector of the previous right Tree-LSTM layer
$\overleftarrow{\mathbf{h_i}}$	Hidden vector of the left Tree-LSTM layer
$\overrightarrow{\mathbf{h_i}}$	Hidden vector of the right Tree-LSTM layer
$\overleftarrow{\mathbf{h_{i+1}}}$	Hidden vector of the next left Tree-LSTM layer
$\overrightarrow{\mathbf{h_{i+1}}}$	Hidden vector of the next right Tree-LSTM layer

$\mathbf{i}$	Word index of the sentence
$\mathbf{ig}$	The input gate of the Tree-LSTM layer
$\mathbf{L}$	Number of data in the dataset
$\overleftarrow{\mathbf{LSTM}}$	Left Tree-LSTM layer
$\overrightarrow{\mathbf{LSTM}}$	Right 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 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 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
β	Cleavage site of the target molecule
θ	Parameter set of the multi layer perceptron layers

σ	The sigmoid function

LIST OF ACRONYMS/ABBREVIATIONS

3D	Three-Dimensional
Aβ	β-Amyloid
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	Protein 53 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 Attentive Recursive Tree (AR-Tree) [4] to learn general compound representations. 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 interpretation. 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

Zeiler and Fergus [6] displayed that 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.

The method used by Zintgraf et al. [8] is based on an instance-specific method by Robnik-Šikonja and Kononenko [9] so called the prediction difference analysis. The

proposed methodology by Zintgraf et al. [8] is a method that similar to Zeiler and Fergus [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.

Klöppel et al. and Ecker et al. [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 Gaonkar and Davatzikos [14] and Haufe et al. [15].

2.3. Layer-wise Relevance Propagation-based Methods

Bach et al. [16] suggested that 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

Simonyan et al. [19] suggested that 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

Simonyan et al. [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 the method used by Erhan et al. [21], the net is trained in a supervised way (to identify the neuron in charge of each class in the unsupervised instance, ImageNet [22] needed to consult a different collection of annotated picture data).

Second, using a single backpropagation run through a classification Convolutional Neural Network (CNN), Simonyan et al. [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 backpropagation, this approach inherits the drawbacks of guided backpropagation. Moreover, it is unique to convolutional neural networks.

In [24], the gradients are integrated 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

Selvaraju et al. [25] used 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, Yogatama et al. [26] train a shift-reduce parser using reinforcement learning (RL) without using any ground truth.

Maillard et al. [27] replaced 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 Choi et al. [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, Williams et al. [33] compare the aforementioned models and show that Choi et al. [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

As benchmark tasks, we used five different classification and four different regression datasets from MoleculeNet [39]. According to MoleculeNet [39], the physiology datasets are BBBP, ClinTox, SIDER, and Tox21, while the BACE dataset is classified as biophysics. Physical chemistry datasets include ESOL and Lipophilicity, as well as FreeSolv. The total number of molecules and tasks for all datasets can be found in Appendix 3.1. Appendix C also contains homogeneities in the distributions of the classification and regression datasets.

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

3.1.1. BACE Task

β-site amyloid precursor protein cleaving enzyme (BACE) dataset contains compounds that are 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 obtained through 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

Using the Simplified Molecular Input Line Entry System (SMILES) representations, we model compounds as documents derived from a chemical language [40]. For character-based segmentation of SMILES representations, the algorithm proposed by Schwaller et al. [41] is used. Every atom except those in salt structures within molecules, every covalent bond except single bonds, every salt structure within molecules, and every molecular branching are represented as different molecular fragments (tokens) in this algorithm.

The fragment dictionary is then obtained by repeating this procedure through all of the benchmark tasks. SMILES representations are encoded into a vector that contains only integers by assigning integers to each fragment in this dictionary, which in-

cludes 194 distinct fragments, for later use in building tree structures. Figures 3.1a and b show a 2D-structured chemical representation and a tree-structured representation of the molecule known as N-methylformamide. It should be noted that the molecule's SMILES representation is **CNC** = **O** and "-" denotes the absence of a node.

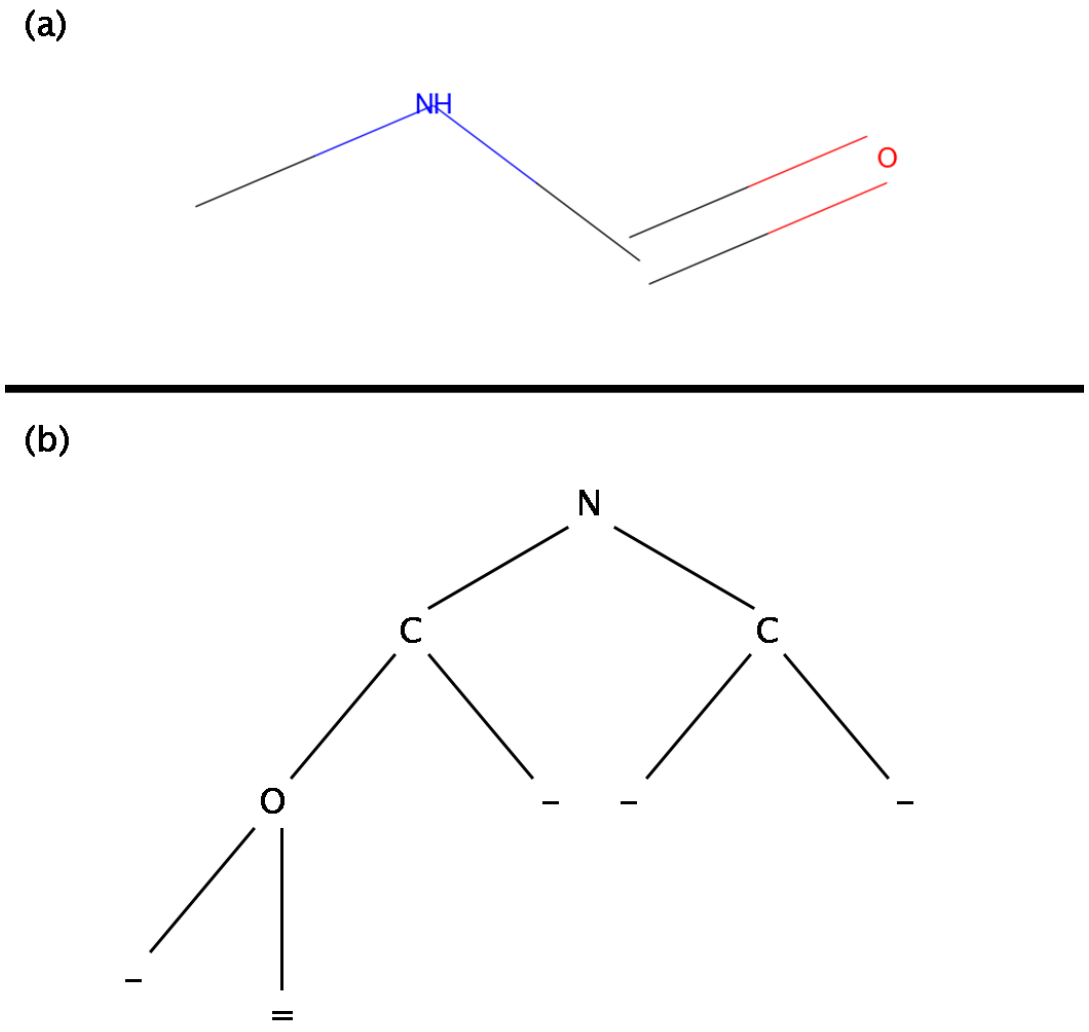

Figure 3.1. Tree representation of a molecule.

3.3. Experimental Setup

Python3 software [42] is used for all programming tasks, and the PyTorch library [43] is used for all artificial intelligence tasks in this work. First, the aforementioned datasets are downloaded as "scaffold" splits using the DeepChem [44] library. Pandas library [45] is used to read and process datasets saved in comma-separated values (CSV) format.

The AR-Tree model scripts were obtained from the work of Shi et al. [4] and modified to meet the requirements of the changes due to input differences, as it is no longer a normal sentence but a SMILES representation. The AR-Tree model is divided into two sub-models: single-input and double-input versions. Because the only inputs in this work are SMILE representations, the single-input version is chosen. The AR-Tree model also provides a choice between RL and STG, with STG being preferred due to some inconsistencies when summing the main loss with the RL loss to calculate the total loss.

Various hyperparameter combinations are tested during training sessions to determine the best one. To begin, all hyperparameters are tested individually with different values to determine which ones affect the benchmark results. The hyperparameters that affect the benchmark results are then combined, and the resulting combinations are tested. Finally, the best combination of them is used for the model, which is trained from scratch one more time to obtain the best possible checkpoints. The maximum number of epochs is set to 500, and all trainings use early-stopping. The model's structure is explained in detail in the following section (Section 3.4). Figure A.1 depicts an abstraction of the model.

The "TRUBA-akya-cuda" server provided by TÜBİTAK, as mentioned in the Acknowledgements, is used for the model's training sessions. In summary, the server has 4 x NVIDIA Tesla V100 GPUs with 16 GB VRAM and 2 x 20-core Intel Xeon Scalable 6148 processors. But it is necessary to note that, unlike the others, values

less than 1500 for the number of neurons in Tree-LSTM layers have a negligible effect on the length of the training time, while values greater than 1500 increase it relatively exponentially.

The best model checkpoints obtained during the training session are used to evaluate the models' benchmarks. As the training objective for the classification tasks, the binary cross-entropy (BCE) loss function is used. We used the Area Under the Receiver Operating Characteristic Curve (ROC-AUC) as a metric to assess the model's performance. For the regression tasks, the root-mean-square error (RMSE) loss function is used as both the training objective and the performance metric. The obtained results are thoroughly examined in Chapter 4.

As with the evaluation of the models' benchmarks, the best model checkpoints obtained during the training session are used for the formation and scoring of fragments, as explained in detail in Sections 3.5 and 3.6. The PubChemPy library [46] is used to identify Chemical Identification Number (CID) numbers from the PubChem database [47], and the RDKit library [48] is used to visualize the top 20 fragments out of all the scored fragments. Finally, in Chapter 5, the obtained fragments are tested to see if they are chemically meaningful.

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

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 A.1.

3.4.1. Top-Down AR-Tree Formation

A bidirectional LSTM basically concatenation of a right and a left-directional LSTMs is used to process the input phrase and produce a context-sensitive hidden vector for each word. The word hidden state ($\overrightarrow{\mathbf{h_i}}$) and the word cell state ($\overrightarrow{\mathbf{c_i}}$) of the right-directional LSTM are expressed as

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

where $\overrightarrow{\mathbf{h_{i-1}}}$, $\overrightarrow{\mathbf{c_{i-1}}}$, $\mathbf{x_i}$, and $\overrightarrow{\mathbf{LSTM}}$ indicate the previous word hidden state of the right-directional LSTM layer, the previous word cell state of the right-directional LSTM layer, the word embedding vector, and the right-directional LSTM layer, respectively. The word hidden state ($\overleftarrow{\mathbf{h_i}}$) and the word cell state ($\overleftarrow{\mathbf{c_i}}$) of the left-directional LSTM are expressed as

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

where $\overleftarrow{\mathbf{h_{i+1}}}$, $\overleftarrow{\mathbf{c_{i+1}}}$, $\mathbf{x_i}$, and $\overleftarrow{\mathbf{LSTM}}$ indicate the next word hidden state of the left-directional LSTM layer, the next word cell state of the left-directional LSTM layer, the word embedding vector, and the left-directional LSTM layer, respectively. The hidden state of the bidirectional LSTM ($\mathbf{h_i}$) is expressed as

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

The cell state of the bidirectional LSTM ($\mathbf{c_i}$) is expressed as

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

$\mathbf{h}$ 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 and this scoring function is expressed as

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

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.

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

Figure 3.2. Pseudo-code of the model architecture.

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 3.2. 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 [49, 50] is utilized as the composition function to calculate the parent representation from its children and corresponding word in a bottom-up fashion in Figure 3.3. 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.

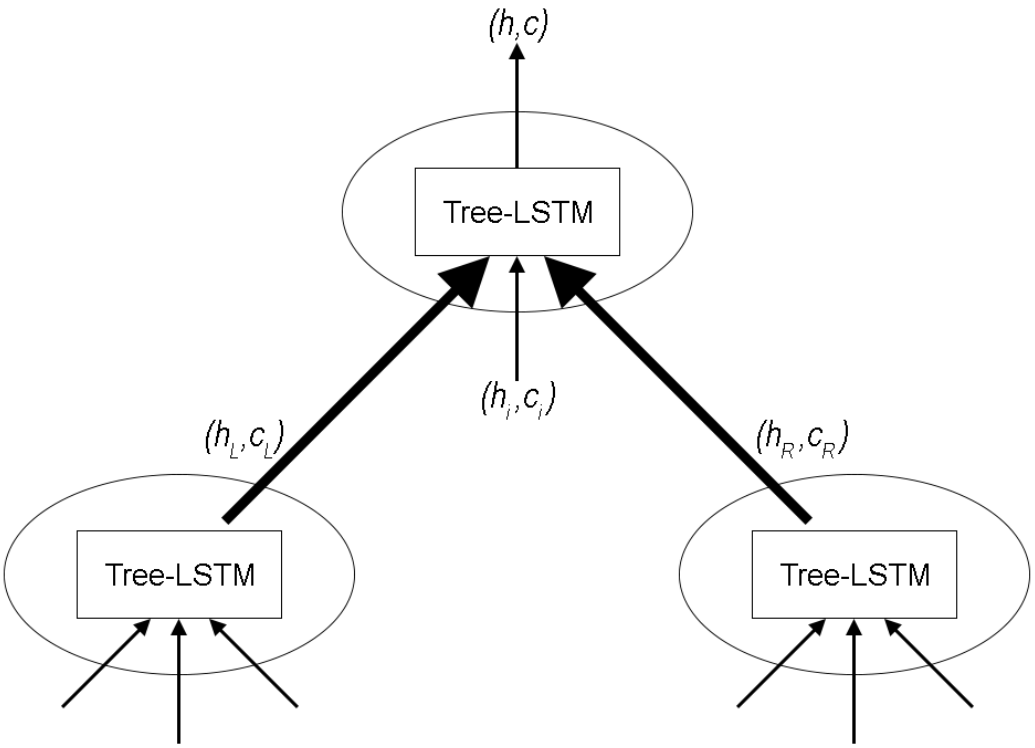

Figure 3.3. Bottom-up embedding.

The whole Tree-LSTM composition function is expressed as

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

where $\mathbf{ig}$, $\mathbf{f_L}$, $\mathbf{f_R}$, $\mathbf{f_i}$, $\mathbf{o}$, $\mathbf{g}$, 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. The cell state of the parent node ($\mathbf{c}$)is expressed as

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

where $\mathbf{c_L}$, $\mathbf{c_R}$, and $\mathbf{c_i}$ indicate the cell state of the left child node, the cell state of the right child node, and the cell state of the current word, respectively. The hidden state of the parent node ($\mathbf{h}$) is expressed as

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

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. $\mathbf{ig}$ determines the extent to which the cell state will be updated when adding new information. $\mathbf{o}$ converts the embedded representation obtained from the cell state into an output representation. $\mathbf{g}$ 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

To begin, all possible subtree formations (fragments) should be discovered in all molecule trees using the corresponding datasets. It is critical to find relatively large-sized fragments rather than small fragments (those with only 2 or 3 atoms except hydrogen atoms). Because larger fragments can be more interpretable and chemically meaningful.

The only way to accomplish this is to form the fragments from leaf to root or bottom-up. Although the root is more discriminative than the other nodes, atoms of the subtree structures formed from the root to the leafs cannot be found side by side in SMILES representations of the corresponding molecules, as shown in Figure 3.4's tree. This tree is identical to the tree in Figure 3.5a and represents the sentence $\mathbf{C_2DBA_2EA_1C_3FHC_1G}$ or, in this case, the SMILES representation. Unfortunately, only the subtree shown in the blue ellipse can form a valid chemical fragment; the others are not. So, in essence, these structures are chemically invalid because the atoms of the chemical fragments presented as nodes are not connected to one another. This also applies to subtree structures formed between leafs and roots. As a result, forming fragments from leaf to root is the only way to obtain chemically valid subtree structures (fragments). Figure 3.5 explains the formation procedure in detail. Subindexes are used to demonstrate that the same tokens can exist at different nodes. A dynamic fragment dictionary (DFD) is created using the obtained fragments, which is specifically dependent on the corresponding dataset. Finally, remaining fragments in the DND are scored using the scoring procedure described in the following section (Section 3.6).

3.6. Scoring Procedure for Chemical Fragments

To evaluate the model's interpretability, all of the fragments (subtrees) found in the previous section (Section 3.5) should be searched in all of the molecules in the

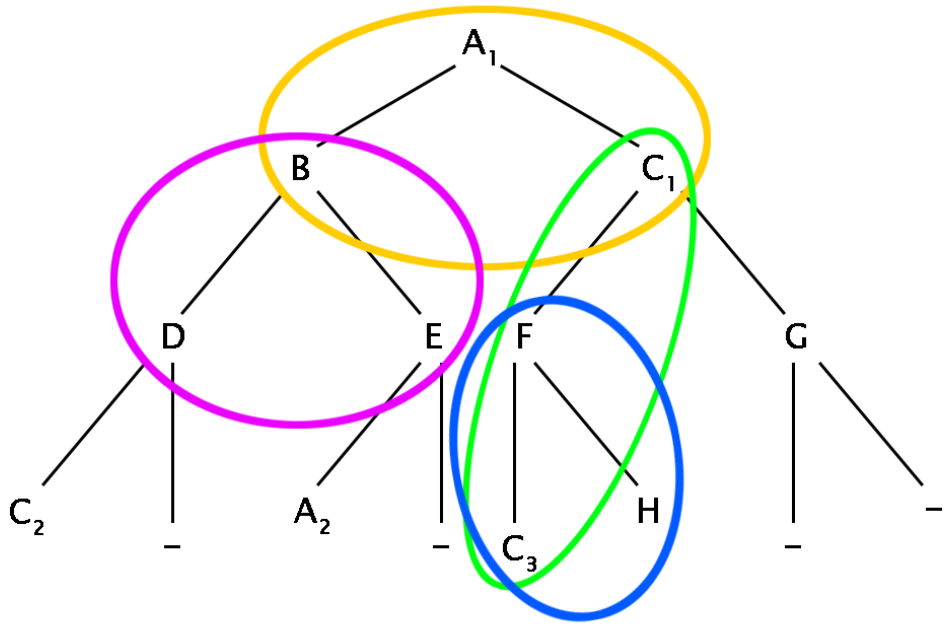

Figure 3.4. Different subtree locations on a tree.

dataset DS and rated using a scoring procedure.

In the scoring procedure, each leaf is assigned 1 point, and each node is assigned 1 point node-by-node from the leaves to the R in the molecule tree T, as shown in Figure 3.6, where "level" indicates point level. As a result, the R receives the maximum point in the T. The points ($\mathbf{P_i}$) of each node are then divided by the distance ($\mathbf{d_{max}}$) between the farthest leaf from the R and the R in the T. As a result, all of the outcomes are between 0 and 1. The total score of the fragment F in the T ($\mathbf{P^{mol}}$) is expressed as

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

where $\mathbf{n}$ represents the total number of repeats of the F in the T. While scoring the fragments, not only the node 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 taken into account. Scores ($\mathbf{P^{mol}_j}$) of the F are added up among the compounds in the DS. The more frequent fragments have a higher total score when added together. The total

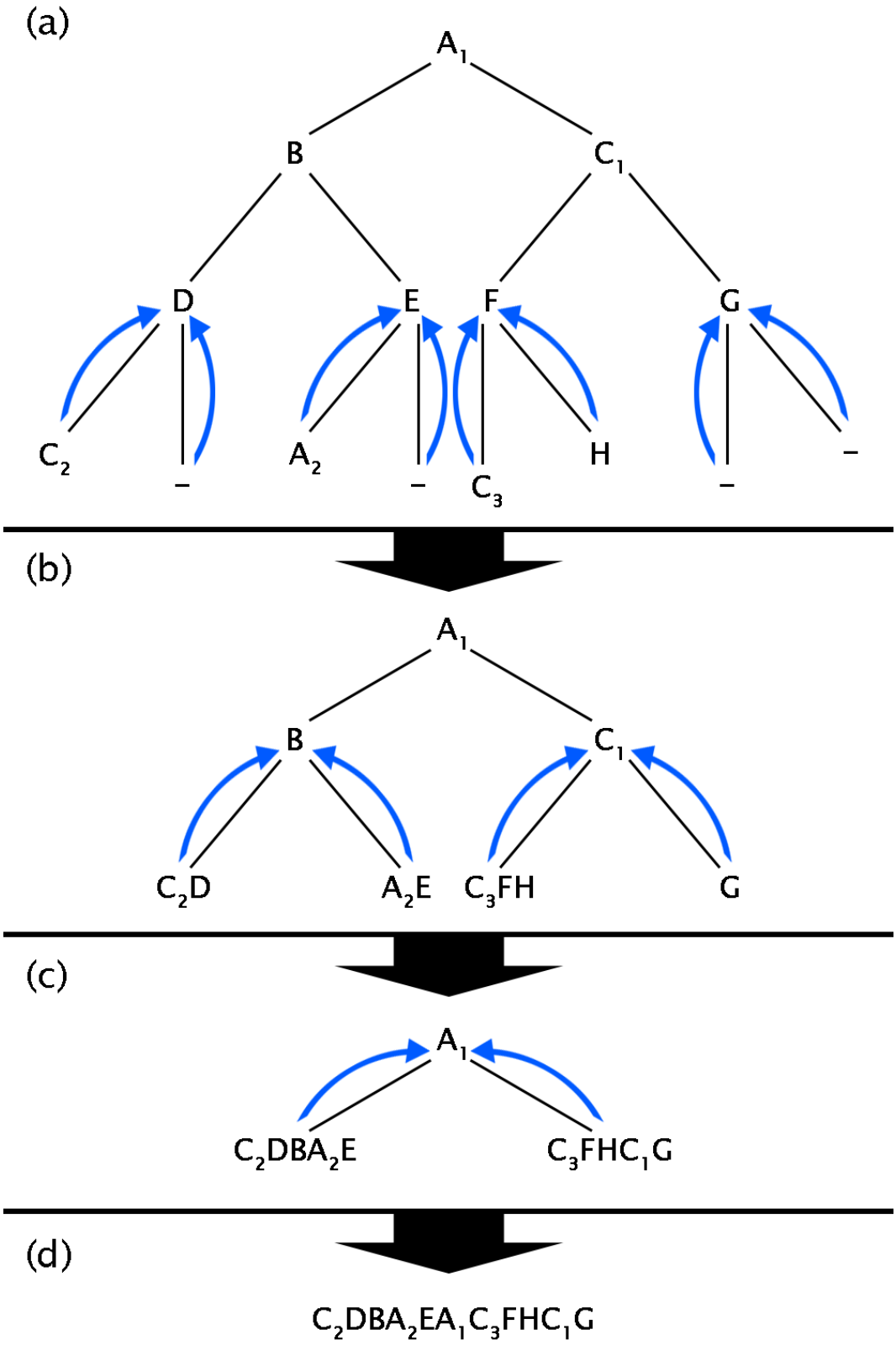

Figure 3.5. Fragment formation procedure.

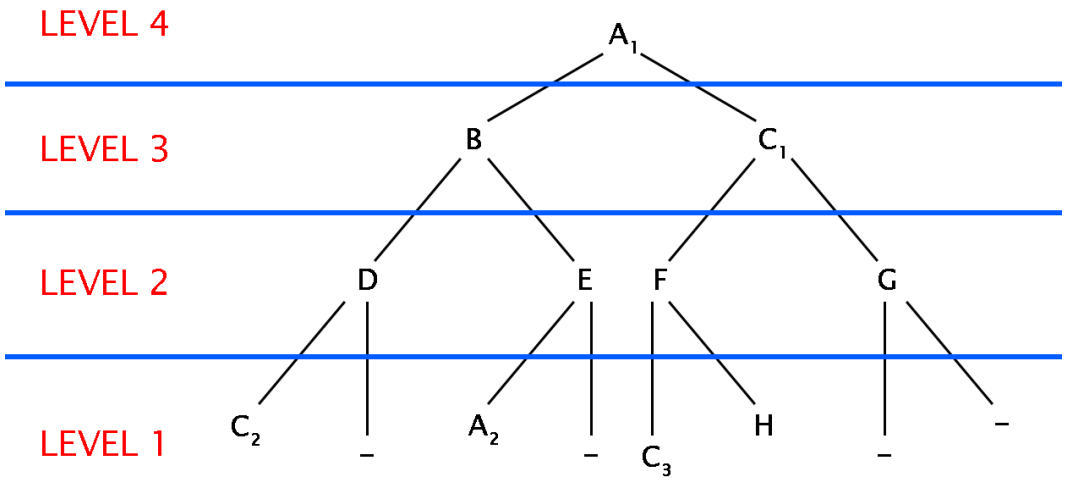

Figure 3.6. Point levels of a tree for scoring.

score of the F in the DS is divided by $\mathbf{m}$ after summarization. As a result, once again, all of the outcomes fall between 0 and 1. The F's average score in the DS ($\overline{\mathbf{P}}$) is expressed as

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

where $\mathbf{m}$ represents the F's total repeat count in the DS. The average test loss of the F in the dataset DS ($\overline{\mathbf{e}}$) is expressed as

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

where $\mathbf{L}$ denotes the DS's size. Equation 3.11 is used to include the $\mathbf{e_k}$ parameter in the general equation. After scoring all of the fragments in the DND, the set FS of all fragment scores is normalized. The min-max normalization formula [51] is expressed as

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

where $\mathbf{x}$, $\hat{\mathbf{x}}$, $\mathbf{x_{max}}$, and $\mathbf{x_{min}}$ denote the unnormalized FS value, the normalized FS value, the maximum FS value, and the minimum FS value. Equation 3.12 is used for normalization, which scales values in a set between 0 and 1. The final score of the F

in the FS ($\mathbf{P_F}$) is expressed as

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

Finally, all fragments were sorted in descending order. Because fragment scores after the first 20 are significantly lower, it was decided to only analyze the top 20 fragments across all datasets (Chapter 5).

4. BENCHMARK RESULTS

Tables 4.2 and 4.4 show the ROC-AUC and RMSE scores of our model and the state-of-the-art models (SOTA) [52–54]. Our model outperformed the SOTA models in the ClinTox task and achieved moderate scores in the other benchmark tasks, according to the results. Tables 3.2, 4.1 and 4.3 show all of the tested parameters as well as the best hyperparameters. Appendix D also contains training and validation curves for the tasks.

4.1. Results of Classification Tasks

As shown in Table 4.1, the learning rate is the same for all classification tasks, 0.001. Except for the SIDER task, where the best optimizer is "adagrad", the optimizer is mostly "adadelta."

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

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
[55]	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

4.2. Results of Regression Tasks

As shown in Table 4.3, the learning rate is the same for all regression tasks, 0.001. Except for the Lipophilicity task, where the best optimizer is “adam”, the optimizer is mostly “adagrad.”

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

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
[55]	-	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

5. INTERPRETATION

The model fragments obtained have been analyzed within the context of their respective tasks. The fragments were referred to as "meaningful fragments" if they were deemed significant by pharmaceutical chemistry and supported by literature. To determine whether the model provided truly meaningful fragments, literature-based Structure-Activity Relationship (SAR), Structure-Toxicity Relationship (STR), or physicochemical properties were considered during task interpretation.

To begin, SAR is the relationship between molecules' three-dimensional (3D) structures and their biological activities. Minor changes to the structure of the lead molecule are made and their effects on biological activity are evaluated based on the assumption that structurally similar compounds have similar physical and biological properties. Second, STR refers to the relationship that exists between the structures of chemical compounds or chemical groups in chemical molecules and their toxicity profiles. Third, physicochemical properties are linked to parameters like water or lipid partition coefficient, polarity, reactivity, acidity or basicity, and so on.

SAR studies from the literature were used to analyze the BACE and BBBP tasks. STR studies from the literature were used to examine the ClinTox and Tox21 tasks. The ESOL and Lipophilicity tasks were investigated by taking the molecules' physicochemical properties into account. The following subsections present the fragments, the names of the corresponding compounds, and the results of the analyses for various tasks. The model determined that the colored fragments were the most meaningful fragments for the corresponding tasks.

The fragments mentioned above can be seen in the images in Appendix B. The image fragments were sorted in descending order from left to right based on their fragment scores. The best fragment is at the top left, while the worst is at the bottom right. Appendix E contains additional information about the fragments.

5.1. BACE Classification Analysis

BACE1 is a protease enzyme involved in the formation of β-amyloid (Aβ) peptides, which are thought to be the cause of Alzheimer's disease. As a result, inhibiting BACE1 is one of the key approaches for developing drugs to treat Alzheimer's disease. In the active site of BACE1, the catalytic dyad of aspartic acid (Asp) residues Asp32 and Asp228 is required for the enzyme's function. As a result, inhibitors that bind to the catalytic site of BACE1 typically interact with the catalytic dyad. Chemical groups capable of forming electrostatic and hydrogen bonds with the enzyme, as well as polar and charged groups capable of binding to the enzyme's hydrophobic pockets, are all required for BACE1 binding [56].

The structures ranked 1, 2, 4, 5, and 11 in Figure B.1 share the same structural core, 2-spirocyclobutyl-6-neopentyl-8-azachromane, as shown in Figure 5.1, respectively with letters. The hydroxyethylamine (HEA) scaffold, which is present in all of these studies and is depicted in blue in Figure 5.1, is a feature that connects them and is thought to be important for the formation of hydrogen bonds (H-bonds) with the aspartic catalytic dyad of BACE1. Fragments a, c, d, and e all have the blue HEA scaffold, whereas fragment b only has the methylamino portion of the HEA moiety. The azachromane (shown in yellow in Figure 5.1) and spirocyclobutane (shown in pink in Figure 5.1) rings occupying the S1$'$ subpocket, as well as the neopentyl group (shown in green in Figure 5.1) filling the S2$'$ subpocket, all increase the binding affinity to BACE1 and are thus 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 assigning them higher significance ranks because the results of these studies perfectly match.

It should also be noted that the algorithm extracted a positive charge from the nitrogen atom in the HEA group, which is attributed to the fact that this amine forms an H-bond with the catalytic aspartic residues in BACE1. Another possibility is that this secondary amine group is protonated in physiological pH.

Furthermore, the introduction of the benzodioxolane group (shown in orange in Figure 5.1e) improved the oral bioavailability and metabolic stability of the previously developed compounds, according to Kaller et al. [57]. The algorithm was successful in extracting this fragment, which has properties suitable for rational drug design.

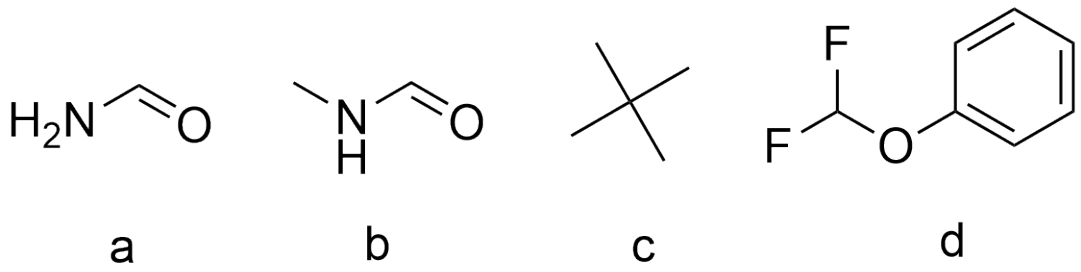

Figure 5.1. BACE classification analysis example 1.

Figure B.1 shows several fragments containing aminohydantoinsand and iminohydantoin tautomers. Many BACE1 inhibitors reported in the literature contain this group of fragments, either in their imidazole-4-one (aminohydantoin) form (shown in blue in Figures 5.2a and c) [58, 59] or in their respective iminohydantoin tautomeric form (shown in yellow in **Figure 5.2b**) [60]. Tautomerism is a chemical phenomenon that occurs when two structural isomers easily interconvert due to the movement of a hydrogen atom within the molecule. The schematic for this imine-amine tautomerism is shown in Figure 5.2. In Figure B.1, fragments a and c were ranked 8 and 17, respectively, by the scoring system. At the active site of BACE1, the catalytic residues Asp32 and Asp228 form an H-bond complex with the amidine moiety of the iminohydantoin compounds (shown in orange in Figure 5.1b). According to the studies, analogs with either aminohydantoin or iminohydantoin moieties have higher oral bioavailability and BACE1 selectivity.

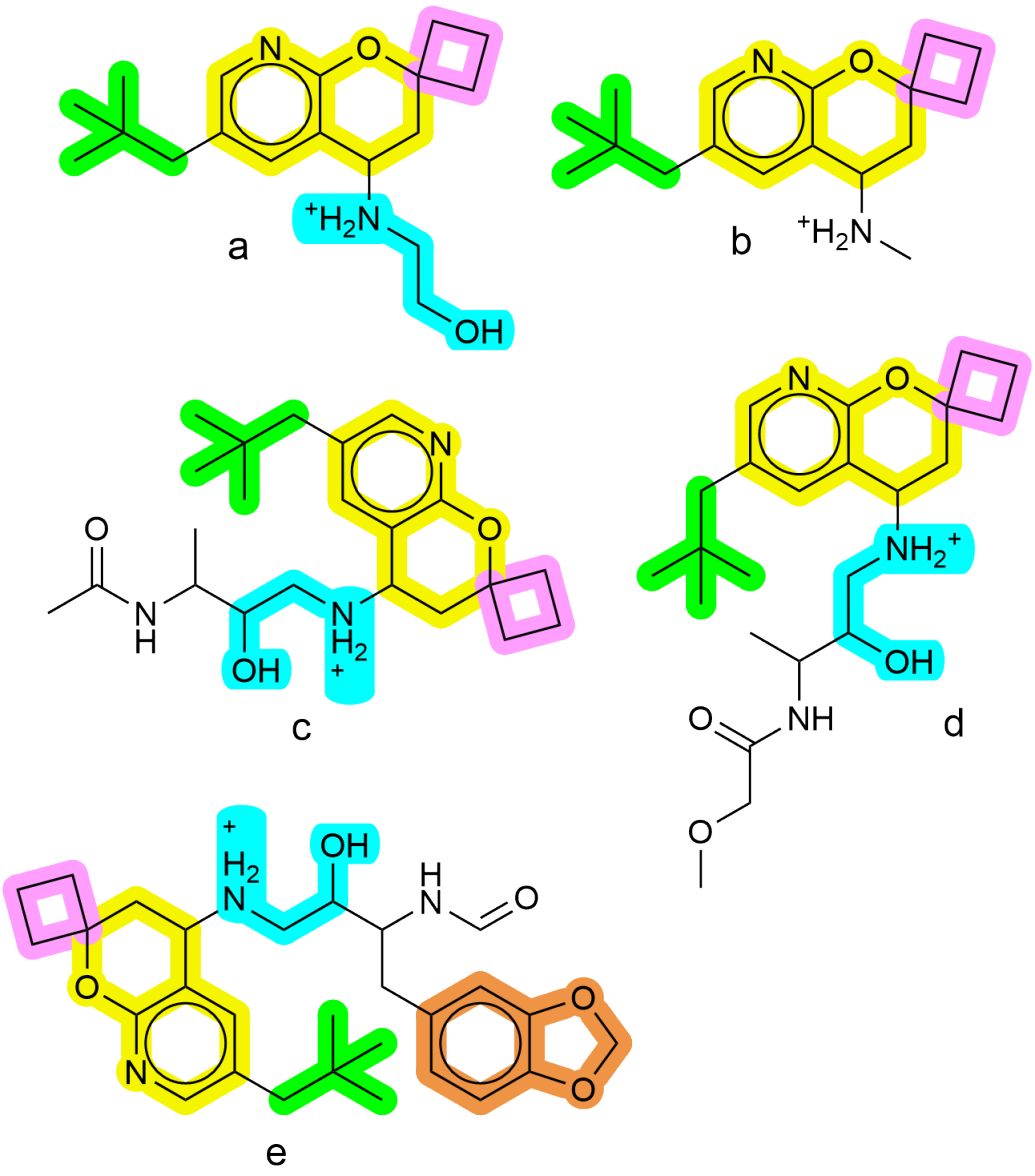

Figure 5.2. BACE classification analysis example 2.

The algorithm also identified fragments a, b, c, and d as significant language units, as shown in Figure 5.3, with ranks of 9, 13, 15, and 16 in Figure B.1. The fragments a and b can be linked to the terminal acetamide group shown in Figure 5.1. This acetamide group was created by shortening a longer amide group in order to maximize the molecular weight, binding effectiveness, and permeability of previously synthesized undesirable compounds to the central nervous system (CNS), according to Weiss et al. [61]. The algorithm also extracted fragment c as an important moiety,

which represents the neopentyl group shown in Figure 5.1. Malamas et al. [58] claims that the difluoromethoxy phenyl moiety of fragment d was added to aminohydantoin derivatives Figure 5.2a to provide more potent compounds. This moiety was discovered in the S2′ region of BACE1.

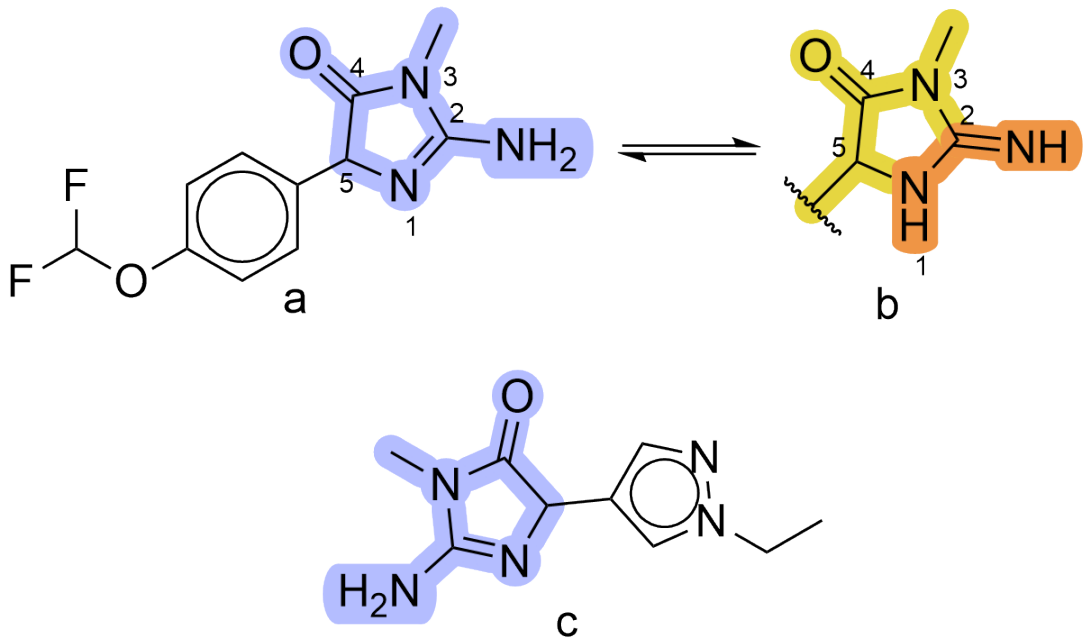

Figure 5.3. BACE classification analysis example 3.

5.2. BBBP Analysis

The BBBP dataset includes chemicals that can cross the BBB. As shown in Figure B.2, the algorithm tends to extract smaller chemical groups as fragments from this task, which may or may not be directly related to the larger molecule's BBB penetrability from which the fragment was extracted. In other words, these smaller fragments may not directly improve or worsen the molecule's BBB permeability, because in some cases, that fragment may decrease lipophilicity and increase BBBP, whereas that same fragment may increase lipophilicity and BBBP when found on a different molecular skeleton. 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, it would be reasonable 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.

However, rank 1 and 12 (benzene and toluene moieties) are known to increase the lipophilicity of molecules based on fundamental chemical knowledge. The same is true for tertiary amines (ranks 13, 14, and 19), as a decrease in the number of hydrogen atoms bound to an amine's nitrogen atom results in a decrease in that amine's ability to form H-bonds, which results in a decrease in lipophilicity, putting tertiary amines above secondary amines and above primary amines in the lipophilicity hierarchy. In this case, rank 15 (a secondary amine) comes after rank 14 (a tertiary amine), which is not the usual order. This is due to the scoring mechanism used here, which is solely based on methodological approaches intended to make the algorithm extract the most repeated meaningful words rather than chemical rules and features of each dataset. Furthermore, fragments 17, 18, and 20 are undesirable when designing molecules that cross the BBB because, due to their ability to form H-bonds, they typically decrease rather than increase a molecule's lipophilicity [62].

5.3. ClinTox Analysis

Toxicity may cause drugs to fail clinical trials. Toxic metabolites formed after drug biotransformation are one of the causes of this situation. Some functional groups may play a role in the formation of toxic metabolites. Nitro groups, aromatic amines, polyhalogenated groups, and so on. Some of the fragments discovered by the algorithm (nimesulide, acetaminophen, thalidomide, sorafenib, and tolcapone) are depicted in Figure 5.4 by coloring on the drug molecules.

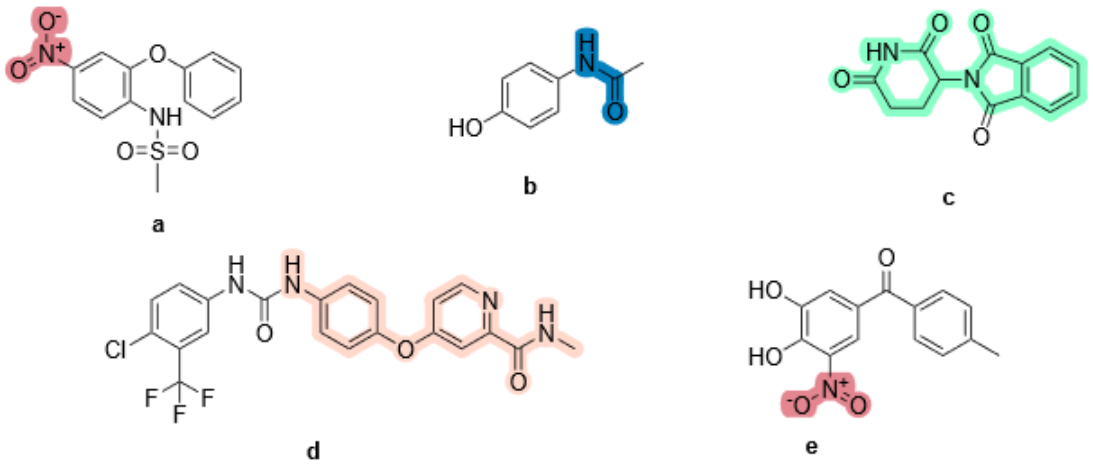

Figure 5.4. ClinTox analysis example 1.

When bound to a molecule, the fragment in rank 20 in Figure B.3, whose SMILES representation is $\mathbf{O} = [\mathbf{N}+][\mathbf{O}-]$, represents the nitro group. Although the nitro group is found in many active molecules, it is toxic and is frequently classified as a structural pharmacophore and/or toxicophore group. Many studies on molecules containing nitro groups in the literature show toxicity issues such as carcinogenicity, hepatotoxicity, mutagenicity, and bone marrow suppression [63]. By reducing the aromatic nitro group, nitro radical anion, nitrozo derivative, nitroxyl radical, hydroxylamine, and primary amine derivatives are formed [64]. Toxicity has been linked to intermediate products, according to research. Methemoglobinemia is caused by hydroxylamine derivatives in particular, while other intermediates have been shown to be mutagenic and carcinogenic [65]. Figure 5.5 depicts the mechanism of aromatic nitro compound reductive biotransformation.

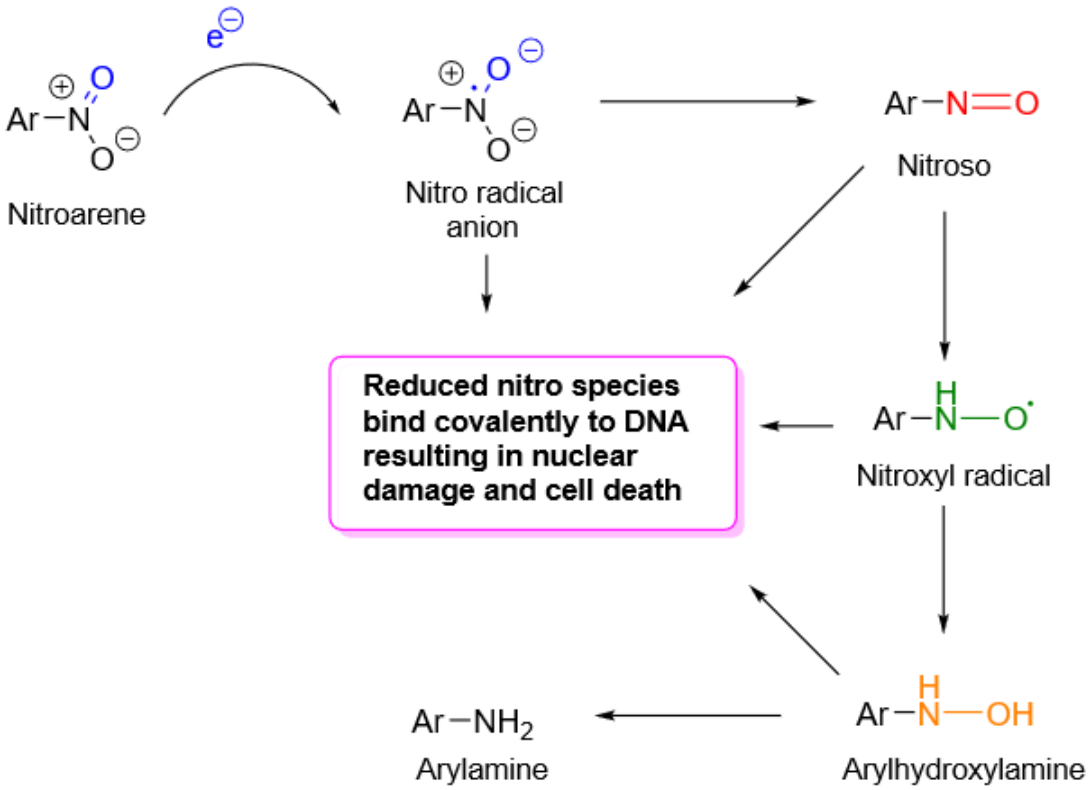

Figure 5.5. Reductive biotransformation mechanism of aromatic nitro compounds.

Nimesulide (4-nitro-2-phenoxymethane-sulfo-anilide), depicted in Figure 5.4a, is a nonsteroidal anti-inflammatory drug with a selective cyclo-oxygenase (COX)-2 inhibitor that is used to treat a variety of inflammatory and pain conditions. Nimesulide's aromatic nitro group undergoes reductive metabolism and has a hepatotoxic effect. As shown in Figure 5.5, nimesulide bioactivation in the liver generates reactive intermediates that bind to macromolecules in the body and cause oxidoreductive stress [66]. The formamide structure is represented by a fragment in rank 3 (**NC = O**) discovered by the algorithm. Intermediates formed as a result of aromatic amide N-oxidation exhibit carcinogenic and cytotoxic effects via covalent bonding with biological macromolecules, as do aromatic amines. An analgesic drug molecule is acetaminophen compound. The hepatotoxic activity of the compound is due to the N-acetylaminoquinone derivative formed from the N-hydroxyacetaminophen intermediate product formed during biotransformation [67]. Figure 5.6 depicts the reaction mechanism.

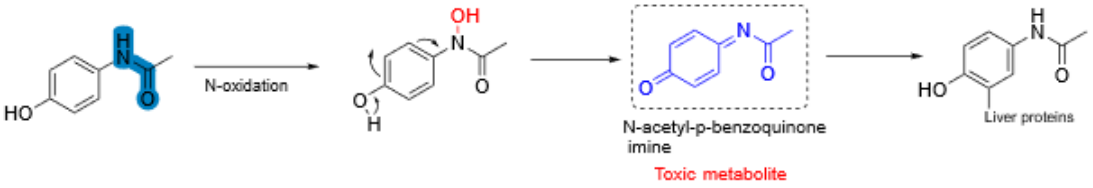

Figure 5.6. ClinTox analysis example 2.

5.4. SIDER Analysis

SIDER task includes a dataset based on adverse effects, which introduces a wide range of interpretation parameters. For example, a medication may have been abused, given the incorrect dosage, or had interactions with other medications that resulted in a reported adverse effect. As a result of the potential unreliability of the analysis methods used to interpret these fragments, no interpretation has been performed for this dataset.

5.5. Tox21 Analysis

The ClinTox dataset analysis (Section 5.3) mentioned the contribution of polyhalogen groups to toxicity. Organic halogenated compounds should be used with extreme caution. They have a high potential for toxicity when they accumulate in adipose tissue and cause cancer. Because they increase lipophilicity, halogens facilitate passage through the blood-brain barrier [68]. Organic halogenated compounds with high toxicity include thyroxine (Figure 5.7a), chloramphenicol, teflon (Figure 5.7b), and halothane (Figure 5.7c). The algorithm discovered eight halogenated compounds among the top 20 fragments. The high amount of fluorine halogen found in fragments 4, 6, 9, and 15 is said to have toxicity similar to the teflon compound. Furthermore, fragments 11 and 19 with aromatic amine and amide structures may be toxic via the mechanism depicted in Figure 5.6.

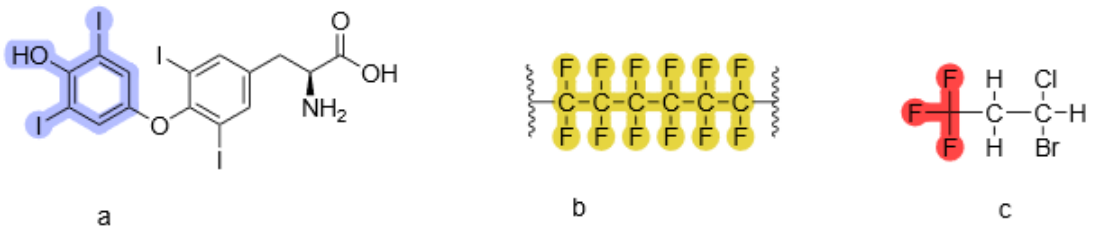

Figure 5.7. Tox21 analysis example 1.

Nonsteroidal anti-inflammatory drugs (NSAIDs) with an acidic structure have been shown in studies to covalently bind to hepatic proteins. NSAIDs with a carboxylic acid structure produce electrophilic intermediates called acyl glucronides, which bind to nucleophilic amino acids. Although not proven in vivo, this covalent binding mechanism to these macromolecules has been proposed to play a role in many NSAIDs' hepatic toxicity [69]. The algorithm successfully found the carboxylic acid structure in fragments 2, 3, and 13.

In one study, the halogen substituents of the antifungal drug UK 47265 were changed to create a less toxic compound for the liver. In this case, the chlorobenzene fragment in fragment 5 was removed and replaced by the 1,3-dichlorobenzene structure, yielding the less toxic fluconazole compound [70, 71]. Figure 5.8 depicts a 2D representation of UK47265 and fluconazole. The diversification of aromatic substituents reduced toxicity. The algorithm discovered the **Clc1ccccc1** fragment, which is a subfragment of the 1,3-dichlorobenzene structure shown in green on the molecule.

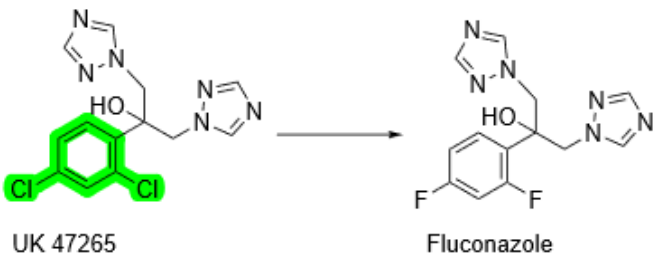

Figure 5.8. Tox21 analysis example 2.

5.6. BACE Regression Analysis

The fragment in rank 4 of Figure B.6 is from the work of Abdel-Magid [72], who has created new chemical substances with BACE-inhibitory properties. In the works of Thompson et al. and Thompson et al. [73, 74], a series of diaminopropane (marked in orange in Figure 5.9) analogs were introduced as potential BACE1 inhibitors, and a patent was also issued. Because of their extra bulk, the extra large rings of these derivatives fit perfectly in the enzyme's ligand-binding regions. The BACE Regression task scoring system assigned the fragments a, b, and c in Figure 5.9 the rankings 8, 9, and 10 in Figure B.6, respectively.

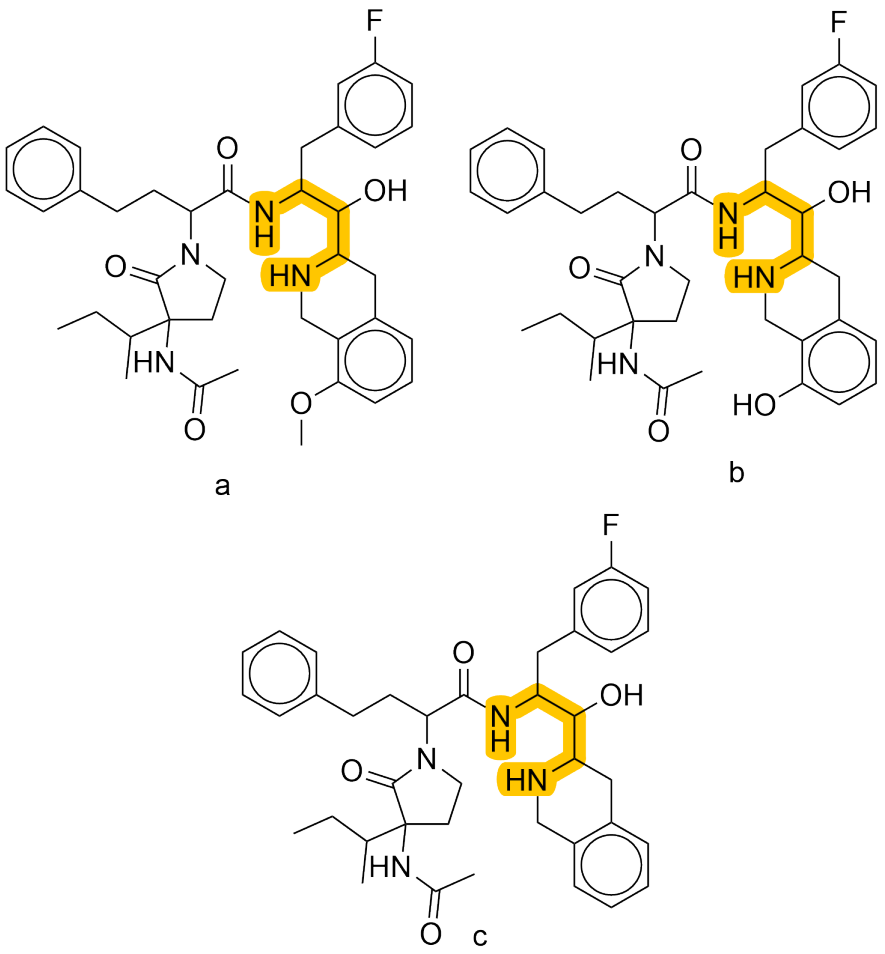

Figure 5.9. BACE regression analysis example 1.

The 1,1-dimethylguanidine fragment Figure 5.10b was ranked 14th out of the top 20 fragments for the BACE Regression task by the algorithm. Despite the fact that 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 5.10), rather than two alkyl chains as seen in the algorithm-extracted fragment (Figure 5.10b), which has two alkyl side chains (dimethyl), a large corpus of acyclic guanidine-based BACE1 inhibitors exist. Studies published in the works of Boy et al. [75] (Figure 5.10a) and Gerritz et al. [76] (Figure 5.10c) highlight the importance of this group in the development of BACE1 inhibitors. Furthermore, as previously stated, cyclic guanidines (aminohydantoins, iminohydantoins) [58–60] are a common structural core among BACE1 inhibitory compounds, and the algorithm-extracted fragment $\mathbf{CN(C)C(=N)N}$ may also be pointing to cyclic guanidine moieties, specifically iminohydantoins. The algorithm's overall success in locating significant language units (fragments) can be seen in its discovery and highlighting of the guanidine moiety as a key fragment, which is frequently used as a common motif across BACE1 inhibitors.

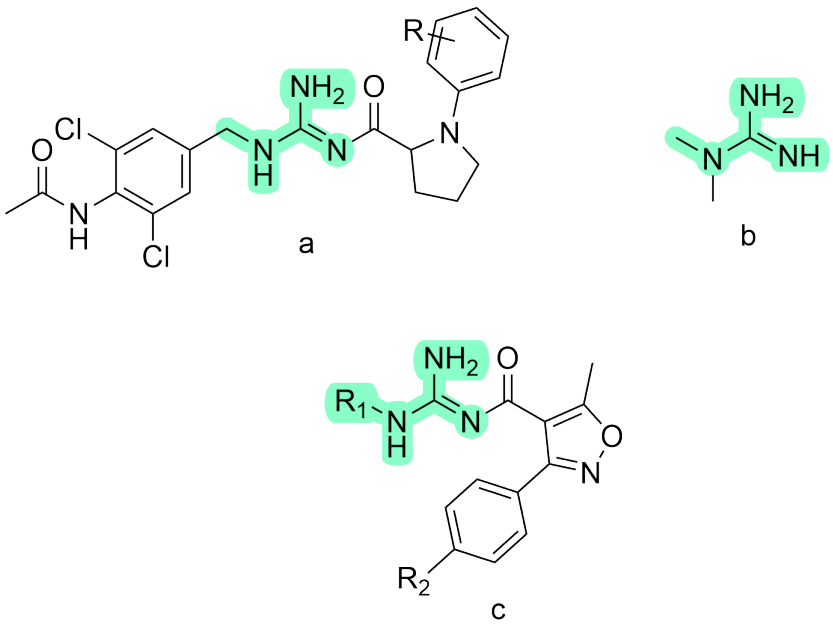

Figure 5.10. BACE regression analysis example 2.

The algorithm was also successful in removing fragments from the BACE Regression task that contained an aminohydantoin moiety (Figure 5.11, where the aminohydantoin core is depicted in purple). This time, the algorithm scored fragments a, b, c, and d with ranks of 16, 18, 19, and 20 among the top 20 fragments of the BACE Regression task. These are the fragments found in the works of Malamas et al. and Malamas et al. [58, 59].

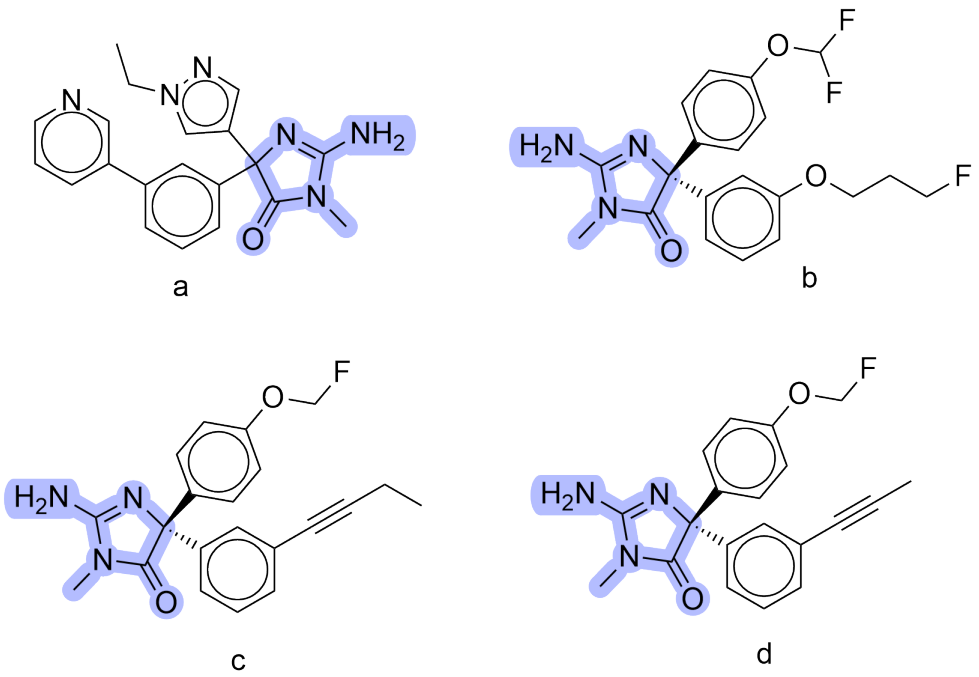

Figure 5.11. BACE regression analysis example 3.

5.7. ESOL Analysis

In general, the factors influencing a compound's solubility in water are the polarity of the molecule and the presence of functional groups capable of forming hydrogen bonds with water molecules [77]. One of the most widely used methods in drug design and development studies is increasing the polarity of a molecule by adding hydrophilic groups (hydroxyl, amino, carboxylic acid, etc.). Polar hydroxyl groups and polar heterocyclic rings, for example, were added to thioconazole compound, which is only used in skin effects due to its high lipophilicity. In this manner, a fluconazole compound with

improved solubility that can be used against systemic infections was synthesized [71].

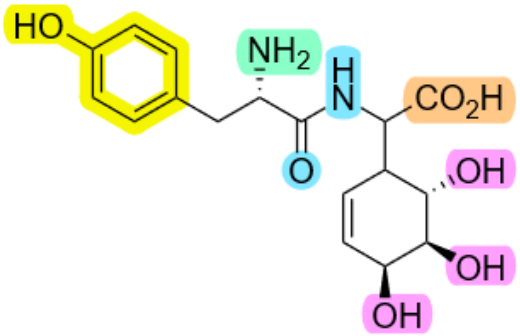

Figure 5.12. Functional groups that cause excessive polarity in a drug.

The groups responsible for high polarity in an antibacterial drug are depicted in Figure 5.12. Seven of the top twenty compounds discovered by the algorithm contain hydroxyl (OH) groups, while one contains phenol groups (represented by pink and yellow colors, respectively, in Figure 5.12). Ether ($R-O-R$), ether ($R-OR$), amino (NH_2), and carbonyl ($C=O$) structures are found in fragments 6, 7, 9, and 18. The pyrazine ring is represented by fragment 20 discovered by the algorithm. The pyrazine ring is water soluble.

5.8. FreeSolv Analysis

Only 18 fragments were found due to the task's limited dataset; unfortunately, these fragments do not represent interpretable parts. The fragments obtained primarily correspond to common simple aliphatic groups found in most molecules, such as ethane and propane. As a result of the potential unreliability of the analysis methods used to interpret these fragments, no interpretation has been performed for this dataset.

5.9. Lipophilicity Analysis

Similarly to the BBBP task, the extracted fragments in the Lipophilicity task cannot be said to directly contribute to an increase in the overall lipophilicity of the compound from which they were extracted. However, the phenyl moieties present in

fragments 1, 4, 5, and 9 increase the lipophilicity of chemical compounds in general. Another well-known property of the trifluoromethyl fragment, which is ranked 13th, is its ability to increase the lipophilicity of molecules [62]. The presence of the bulky group adamantyl in fragment 15 is expected to increase lipophilicity. The scoring system, like the BBBP task, is not intended to rank the fragments based on their physicochemical differences, but rather to aid the method in finding the best fragment. Fragments in ranks 17-19 also have bulky hydrophobic structures, which amplify the molecule's lipophilic properties.

6. DISCUSSION

6.1. Affinity Between Benchmark Results and Expectations

MoleculeNet [39] has four main titles for its datasets: quantum mechanics, physical chemistry, biophysics, and physiology. Quantum mechanics datasets could not be used in this work due to input incompatibility (3D-coordinates). In the context of the remaining tasks' complexity, it is reasonable to assume that learning the physical chemistry tasks (i.e., ESOL, FreeSolv, Lipophilicity) is easier than the rest because the tasks include only information about the chemicals themselves and no information about any external variables.

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

But the most complex are undoubtedly the physiology tasks (e.g., BBBP, Tox21, SIDER), which are also regression tasks because there is now much more information due to consecutive systematic interactions at the cell-scale. Overall, it is expected that physical chemistry task scores will be higher than biophysics task scores, and that biophysics task scores will be higher than physiology task scores.

And, indeed, the obtained benchmark results are consistent with the previously stated expectations. The model ranks only in the top three of all SOTA models for physical chemistry tasks. The only tested task for biophysics is BACE, and it is the only classification task with a relatively good rank, which is 4th to last among all SOTA models. Finally, with the exception of the ClinTox task, the model is either last or second last in the physiology tasks.

6.2. Comparison of Regression and Classification Tasks

The model is clearly superior for regression tasks than classification tasks, as it ranks only third among all SOTA models. One possible explanation for this situation is the continuity of the target values. The target variable in regression tasks is a continuous variable, such as a numerical value, and the goal is to predict this value as accurately as possible. In comparison, the target variable in classification tasks 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 have a wide range of values, making the problem easier to predict accurately. In contrast, the number of possible target values in classification tasks is frequently limited, making it more difficult to achieve high accuracy [78, 79].

Another possible explanation is that classification tasks may have unbalanced datasets, in which some categories or labels have far fewer examples than others, further complicating the task [80, 81], and as shown in the figures in Appendix C, most classification datasets appear heterogeneous.

6.3. Found Chemical Fragments

Some fragments in the same top 20 fragments appear to be gradually growing branches (for example, fragments ranked 4, 6, 9, and 15 in the top 20 fragments for the Tox21 task). Because the branching size of fragments affects their score, larger branches were found in better positions in the top 20 fragments for corresponding tasks.

Furthermore, some fragments appear more than once in the top 20 fragments for different tasks. The reason for this is that some small molecules (for example, benzene, ethene, and propane) repeat very frequently in most datasets. Because the frequency of fragments affects their score, these small molecules can be found in the majority

of the top 20 fragments for various tasks. Also, during research, some fragments are discovered that are not registered in the PubChem database, so-called "None" in the top 20 fragments, despite the fact that there are patents for these fragments in the literature, as discussed in Chapter 5.

It is worth noting that, in the Tox21 task, the fragment in rank 7 is more toxic than the fragment in rank 4. As a result, the scoring procedure cannot precisely rank them. Nonetheless, because "importance" is a relative term, it has been deemed 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

Using the AR-Tree model, we present an interpretation strategy for compounds and their fragments. It can highlight the important fragments for the corresponding tasks thanks to the model's task-specific attention mechanism. Experiment results show that our approach outperforms SOTA models for ClinTox and BACE Regression tasks. In terms of interpretability, the model and scoring strategy were successful in identifying chemically meaningful fragments in compounds for the BACE Classification, BACE Regression, ClinTox, Tox21, ESOL, and Lipophilicity tasks. Determining the most important fragments for the given tasks is an important feature because recognizing the important parts of compounds can provide experts with insights in experimental studies for the discovery of new medicines. Because our method allows us to determine the important fragments of molecules, we hope that the model will be useful to people suffering from various incurable diseases.

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", *Proceedings of the Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, Vol. 33, pp. 6991–6998, Honolulu, USA, 2019.

5. Shrikumar, A., P. Greenside and A. Kundaje, "Learning Important Features Through Propagating Activation Differences", *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70, pp. 3145–3153, Sydney, Australia, 2017.

6. Zeiler, M. D. and R. Fergus, "Visualizing and Understanding Convolutional Networks", *Computer Vision - ECCV*, Vol. 8689, pp. 818–833, Heidelberg, Germany, 2014.

7. Zhou, J. and O. Troyanskaya, "Predicting Effects of Non-Coding Variants with Deep Learning-Based Sequence Model", *Nature Methods*, Vol. 12, No. 10, p. 931–934, 2015.

8. Zintgraf, L. M., T. S. Cohen, T. Adel and M. Welling, "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis", *International Conference on Learning Representations*, Toulon, France, 2017.

9. Robnik-Šikonja, M. and I. Kononenko, "Explaining Classifications For Individual Instances", *Institute of Electrical and Electronics Engineers 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, No. 3, pp. 681–689, 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*, Vol. 30, No. 32, pp. 10612–10623, 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, No. 4, pp. 980–995, 2005.

13. Wang, Z., A. Childress, D. Wang and J. Detre, "Support Vector Machine Learning-Based fMRI Data Group Analysis", *NeuroImage*, Vol. 36, No. 4, p. 1139–1151, 2007.

14. Gaonkar, B. and C. Davatzikos, "Analytic Estimation of Statistical Significance Maps for Support Vector Machine-Based Multivariate Image Analysis and Classification", *NeuroImage*, Vol. 78, No. 9, pp. 270–283, 2013.

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, No. 10, pp. 96–110, 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", *Public Library of Science One*, Vol. 10, No. 7, pp. 1–46, 2015.

17. Shrikumar, A., P. Greenside, A. Shcherbina and A. Kundaje, "Not Just a Black Box: Learning Important Features Through Propagating Activation Differences", *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70, pp. 3145–3153, Sydney, Australia, 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", arXiv 1611.07270, 2016.

19. Simonyan, K., A. Vedaldi and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", *Workshop at International Conference on Learning Representations*, Alberta, Canada, 2014.

20. Springenberg, J. T., A. Dosovitskiy, T. Brox and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net", *International Conference on Learning Representations*, San Diego, USA, 2015.

21. Erhan, D., Y. Bengio, A. Courville and P. Vincent, *Visualizing Higher-Layer Features of a Deep Network*, Tech. Rep. 1341, University of Montreal, 2009.

22. Krizhevsky, A., I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Communications of the Association for Computing Machinery*, Vol. 60, No. 6, p. 84–90, 2017.

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, 2019.

24. Sundararajan, M., A. Taly and Q. Yan, "Gradients of Counterfactuals", arXiv 1611.02639, 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", *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 1, p. 1466–1477, Berlin, Germany, 2016.

26. Yogatama, D., P. Blunsom, C. Dyer, E. Grefenstette and W. Ling, "Learning to Compose Words into Sentences with Reinforcement Learning", *International Conference on Learning Representations*, Toulon, France, 2017.

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, 2019.

28. Cocke, J., *Programming Languages and Their Compilers: Preliminary Notes*, New York University, New York, USA, 1969.

29. Younger, D. H., "Recognition and Parsing of Context-Free Languages in Time n^3", *Information and Control*, Vol. 10, No. 2, pp. 189–208, 1967.

30. Kasami, T., *An Efficient Recognition and Syntax-Analysis Algorithm for Context-Free Languages*, Tech. Rep. AFCRL-65-758, Air Force Cambridge Research Laboratory, 1965.

31. Choi, J., K. M. Yoo and S. goo Lee, "Learning to Compose Task-Specific Tree Structures", *Proceedings of the Thirty-Second Association for the Advancement of*

Artificial Intelligence Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth Association for the Advancement of Artificial Intelligence Symposium on Educational Advances in Artificial Intelligence, p. 5094–5101, New Orleans, USA, 2018.

32. Jang, E., S. Gu and B. Poole, "Categorical Reparameterization with Gumbel-Softmax", *International Conference on Learning Representations*, Toulon, France, 2017.

33. Williams, A., A. Drozdov and S. Bowman, "Do Latent Tree Learning Models Identify Meaningful Structure in Sentences?", *Transactions of the Association for Computational Linguistics*, Vol. 6, No. 19, p. 253–267, 2018.

34. Santos, C. D., M. Tan, B. Xiang and B. Zhou, "Attentive Pooling Networks", arXiv 1602.03609, 2016.

35. Munkhdalai, T. and H. Yu, "Neural Tree Indexers for Text Understanding", *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, Vol. 1, pp. 11–21, Valencia, Spain, 2017.

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

38. Kim, Y., C. Denton, L. Hoang and A. M. Rush, "Structured Attention Networks", *International Conference on Learning Representations*, Toulon, France, 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", *Chemical Science*, Vol. 9, No. 2, pp. 513–530, 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", *American Chemical Society Central Science*, Vol. 5, No. 9, pp. 1572–1583, 2019.

42. Van Rossum, G. and F. L. Drake, *Python 3 Reference Manual*, 2009, `https://www.python.org`, accessed on 15 August, 2023.

43. 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*, Curran Associates, New York, USA, 2019.

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

45. 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, USA, 2010.

46. Swain, M., R. Sjögren, zachcp, Y. Hsiao, L. Lazzaro and B. Dahlgren, "PubChemPy: A Way to Interact with PubChem in Python", `https://pubchempy.readthedocs.io`, 2017, accessed on August 15, 2023.

47. 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, 2022.

48. Landrum, G., "RDKit: Open-Source Cheminformatics Software", `https://www.rdkit.org`, 2016, accessed on August 15, 2023.

49. Zhu, X., P. Sobhani and H. Guo, "Long Short-Term Memory Over Tree Structures", arXiv 1503.04881, 2015.

50. Tai, K. S., R. Socher and C. D. Manning, "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks", *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1, p. 1556–1566, Beijing, China, 2015.

51. Patro, S. G. K. and K. K. Sahu, "Normalization: A Preprocessing Stage", *International Advanced Research Journal in Science, Engineering and Technology*, Vol. 2, No. 3, 2015.

52. 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.

53. Ahmad, W., E. Simon, S. Chithrananda, G. Grand and B. Ramsundar, "ChemBERTa-2: Towards Chemical Foundation Models", arXiv 2209.01712, 2022.

54. 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.

55. Hu, W., B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande and J. Leskovec, "Strategies for Pretraining Graph Neural Networks", *International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

56. 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.

57. 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 β-Secretase Inhibitors That Demonstrate Robust Reduction of Central β-Amyloid", *Journal of Medicinal Chemistry*, Vol. 55, No. 21, pp. 9009–9024, 2012.

58. 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 β-Secretase (BACE1) Inhibitors with Enhanced Brain Permeability", *Bioorganic & Medicinal Chemistry Letters*, Vol. 20, No. 22, pp. 6597–6605, 2010.

59. 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.

60. 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.

61. 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 β-Secretase", *American Chemical Society Medicinal Chemistry Letters*, Vol. 3, No. 11, pp. 886–891, 2012.

62. Landry, M. L. and J. J. Crawford, "LogD Contributions of Substituents Commonly Used in Medicinal Chemistry", *American Chemical Society Medicinal Chemistry Letters*, Vol. 11, No. 1, pp. 72–76, 2020.

63. 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.

64. 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.

65. Gross, P. and R. P. Smith, "Biologic Activity of Hydroxylamine: A Review", *Chemical Rubber Company Critical Reviews in Toxicology*, Vol. 14, No. 1, pp. 87–99, 1985.

66. 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, No. 55, pp. 469–475, 2018.

67. 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.

68. 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.

69. Boelsterli, U. A., "Mechanisms of NSAID-Induced Hepatotoxicity: Focus on Nimesulide", *Drug Safety*, Vol. 25, No. 9, pp. 633–648, 2002.

70. 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.

71. 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. 3, pp. 267–271, 1990.

72. Abdel-Magid, A. F., "BACE Inhibitors: Potential Treatment of Alzheimer's Disease, Dementia, and Related Neurodegenerative Disorders (B): 3-Amino-4-Fluoro-1H-Isoindol Derivatives", *American Chemical Society Medicinal Chemistry Letters*, Vol. 3, No. 11, pp. 869–870, 2012.

73. 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.

74. Thompson, L. A., K. M. Boy, J. Shi, J. E. Macor, A. C. Good and L. R. Marcin, "Substituted Tetrahydroisoquinolines as β-Secretase Inhibitors", U.S. Patent 7902218, March 8, 2011, `https://patents.google.com/patent/US7902218B2`, accessed on August 15, 2023.

75. 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 β-Secretase (BACE)", *Bioorganic & Medicinal Chemistry Letters*, Vol. 25, No. 22, pp. 5040–5047, 2015.

76. 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 β-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.

77. 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. 3–26, 2001.

78. Hastie, T., R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Second Edition, Springer, New York, USA, 2009.

79. Kuhn, M. and K. Johnson, *Applied Predictive Modeling*, Springer, New York, USA, 2013.

80. Bishop, C., *Pattern Recognition and Machine Learning*, Springer, New York, USA,

2006.

81. Geron, A., *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, Second Edition, O'Reilly Media, Sebastopol, USA, 2019.

APPENDIX A: ABSTRACT OF MODEL

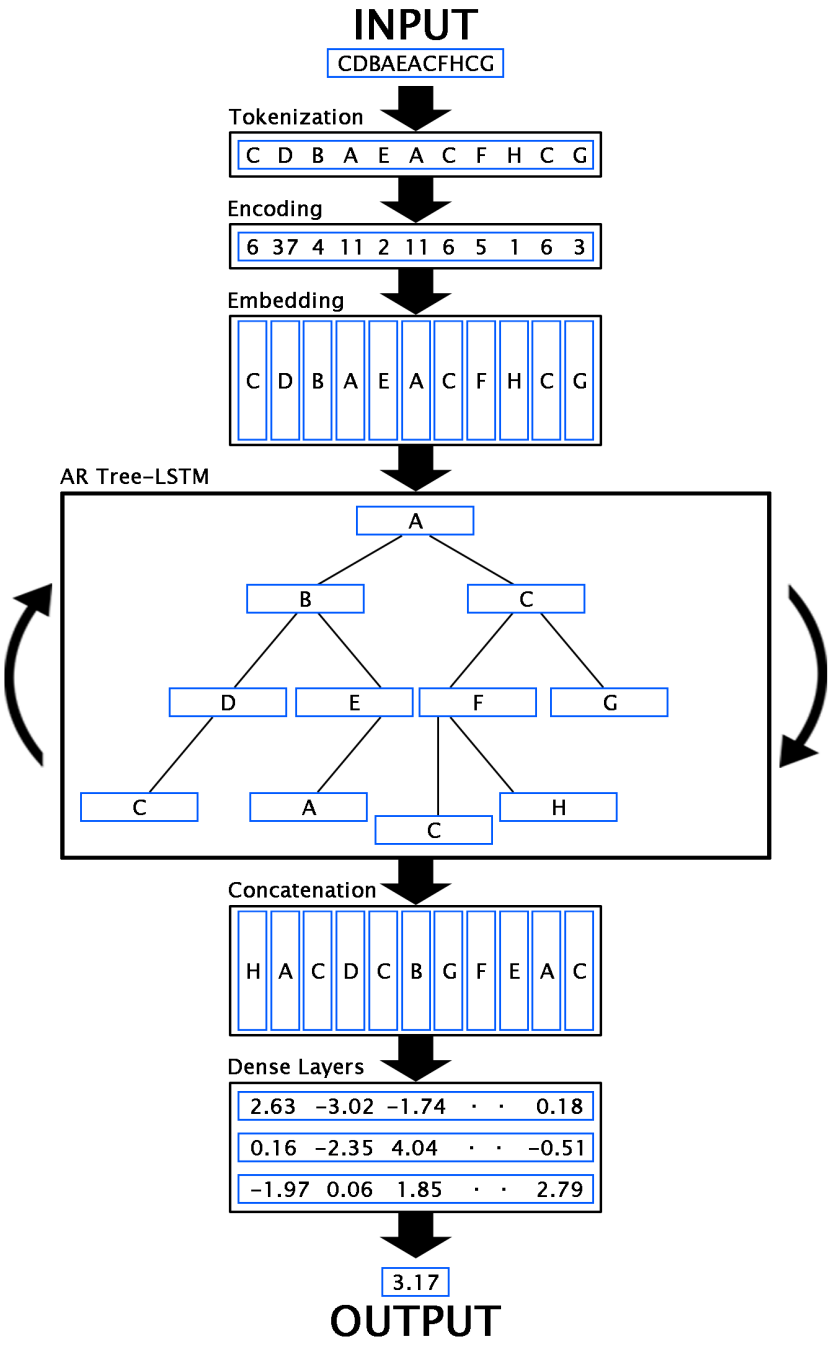

Figure A.1. Abstract of the model.

APPENDIX B: TOP 20 FRAGMENTS

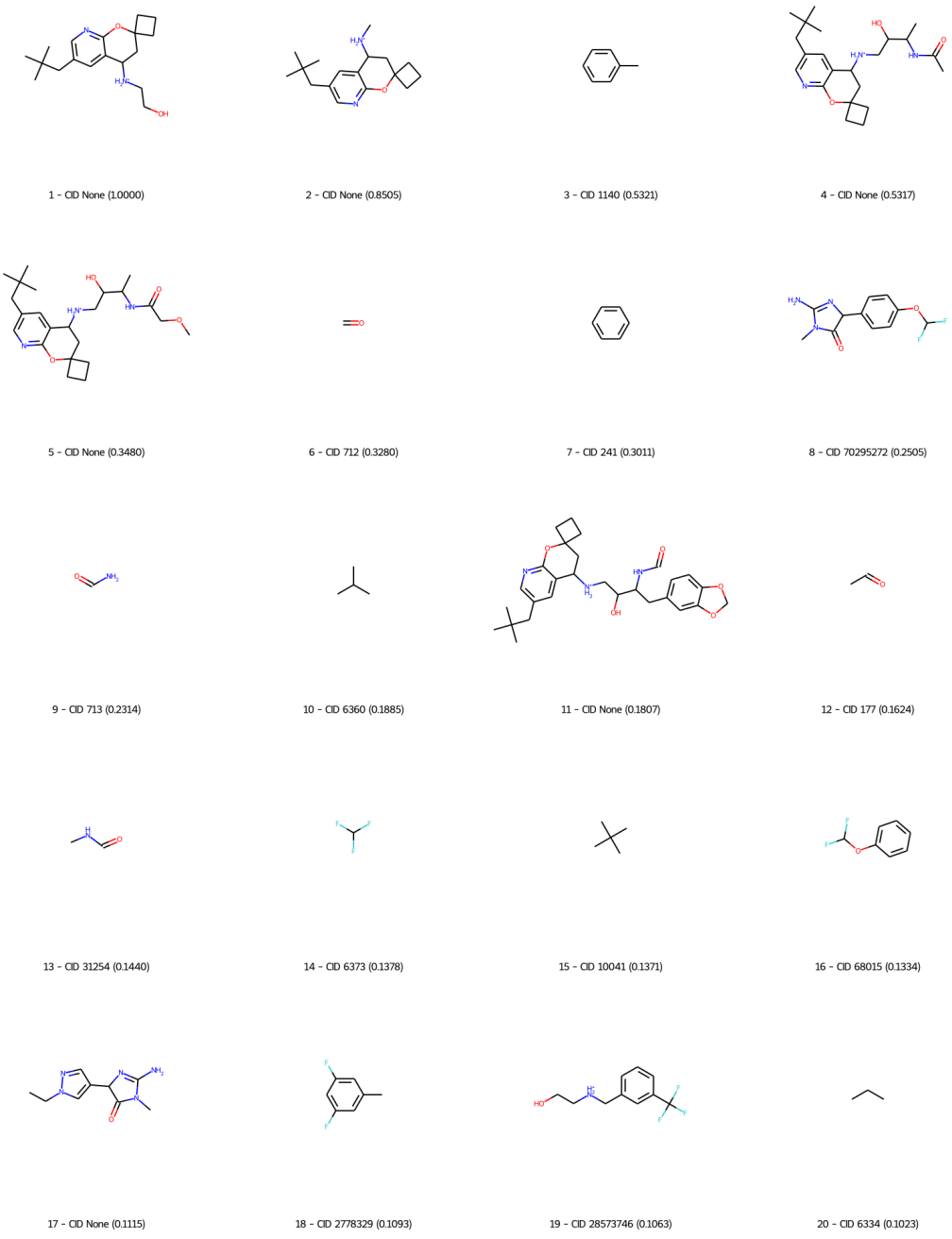

Figure B.1. Top 20 fragments for BACE classification task.

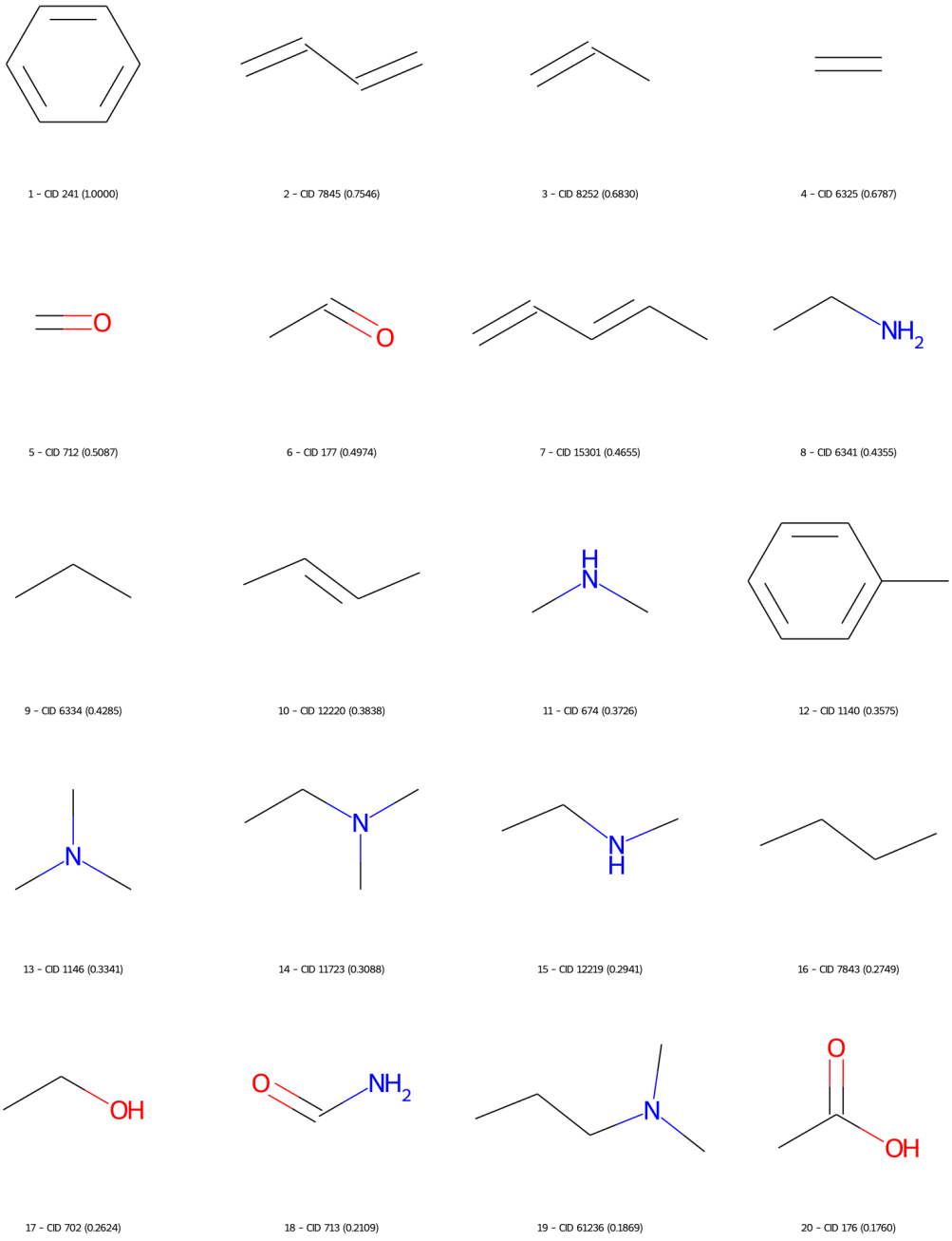

Figure B.2. Top 20 fragments for BBBP task.

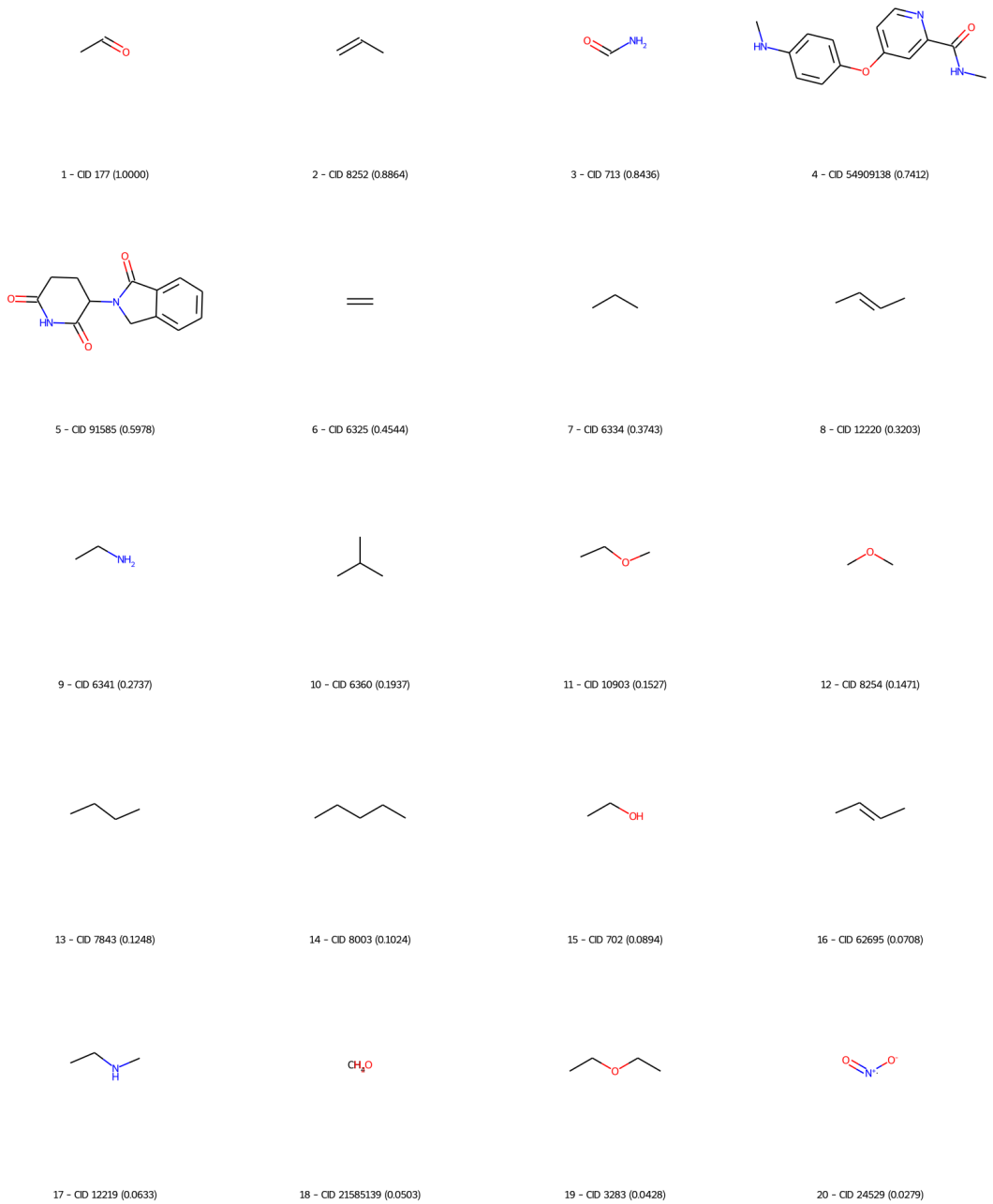

Figure B.3. Top 20 fragments for ClinTox task.

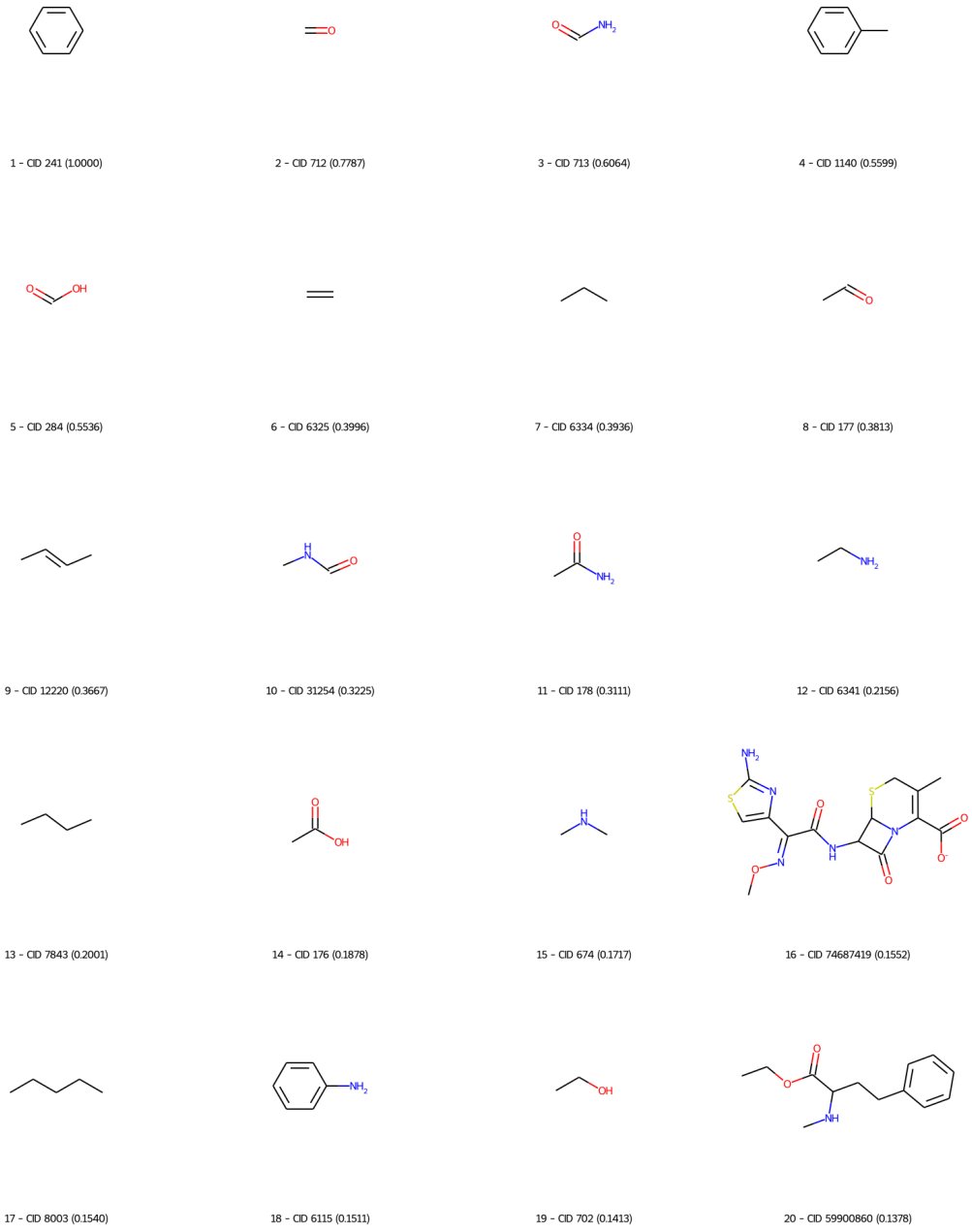

Figure B.4. Top 20 fragments for SIDER task.

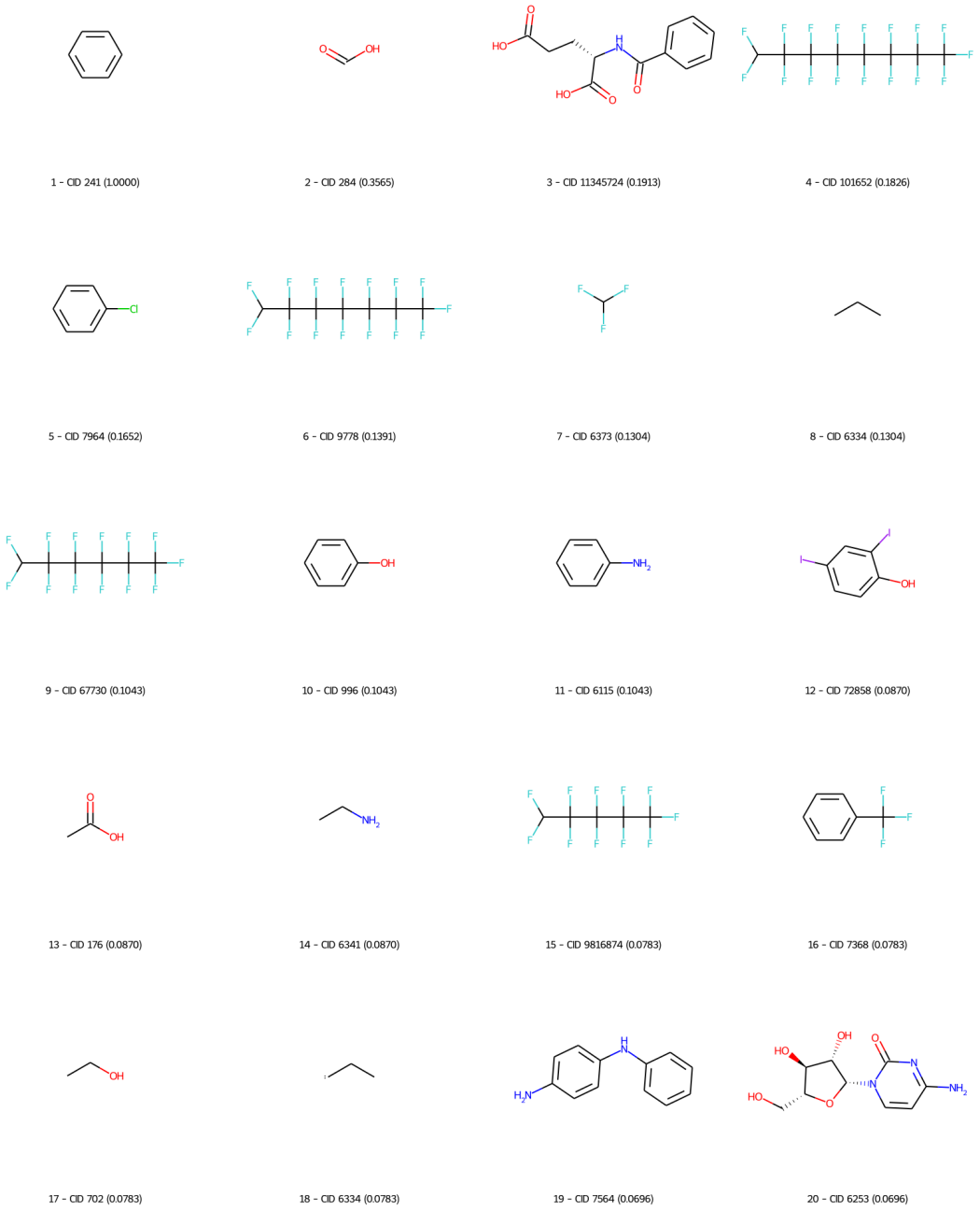

Figure B.5. Top 20 fragments for Tox21 task.

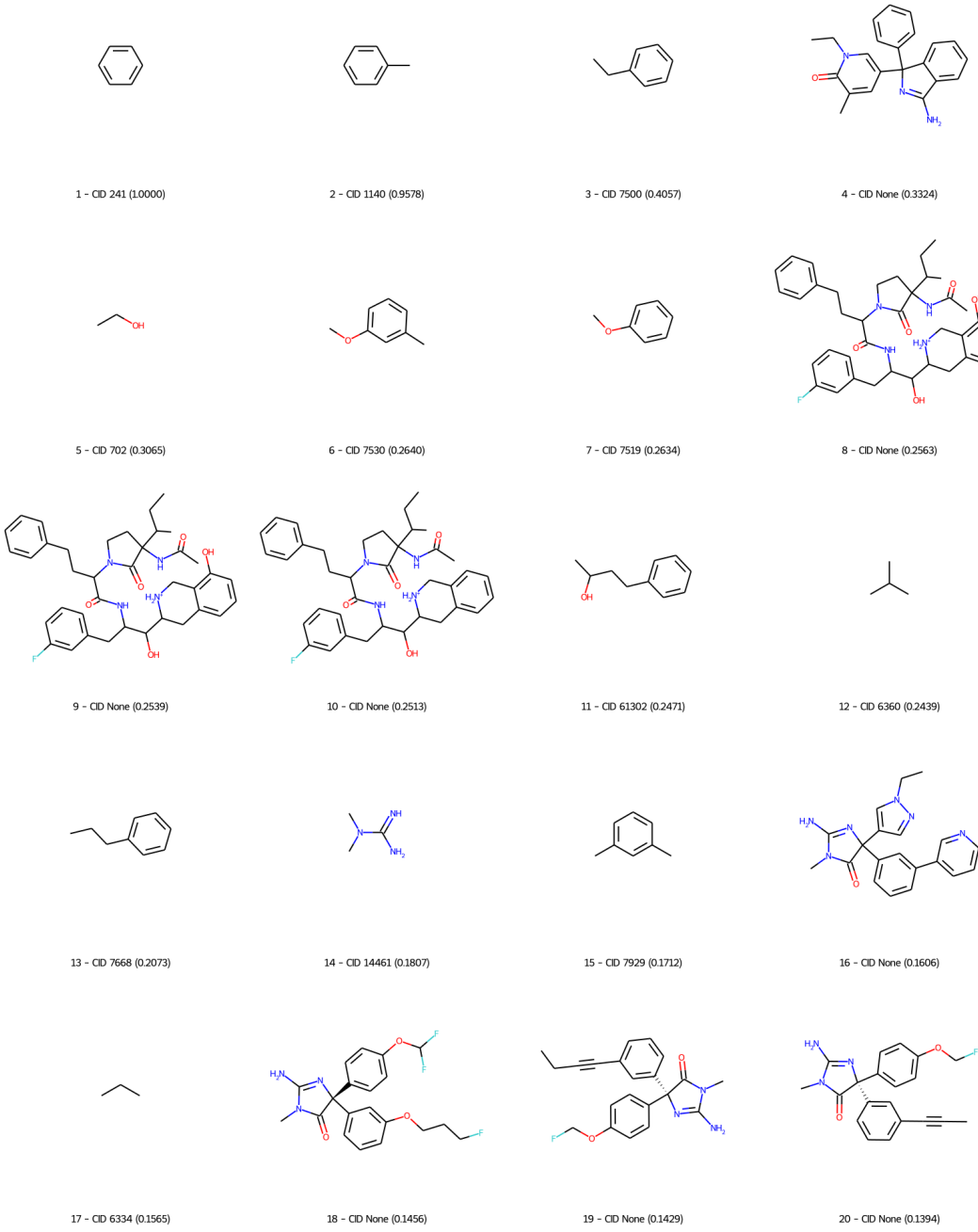

Figure B.6. Top 20 fragments for BACE regression task.

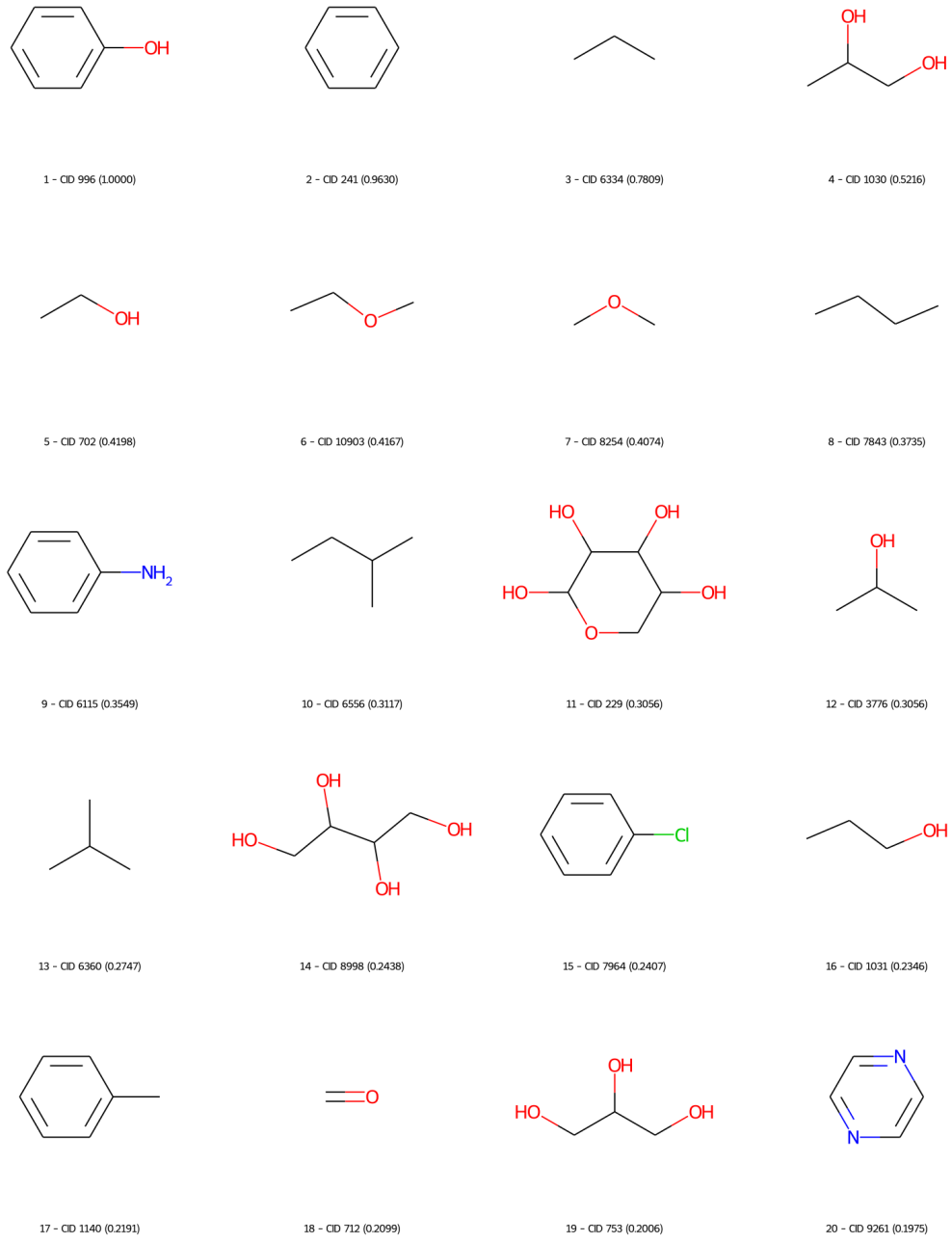

Figure B.7. Top 20 fragments for ESOL task.

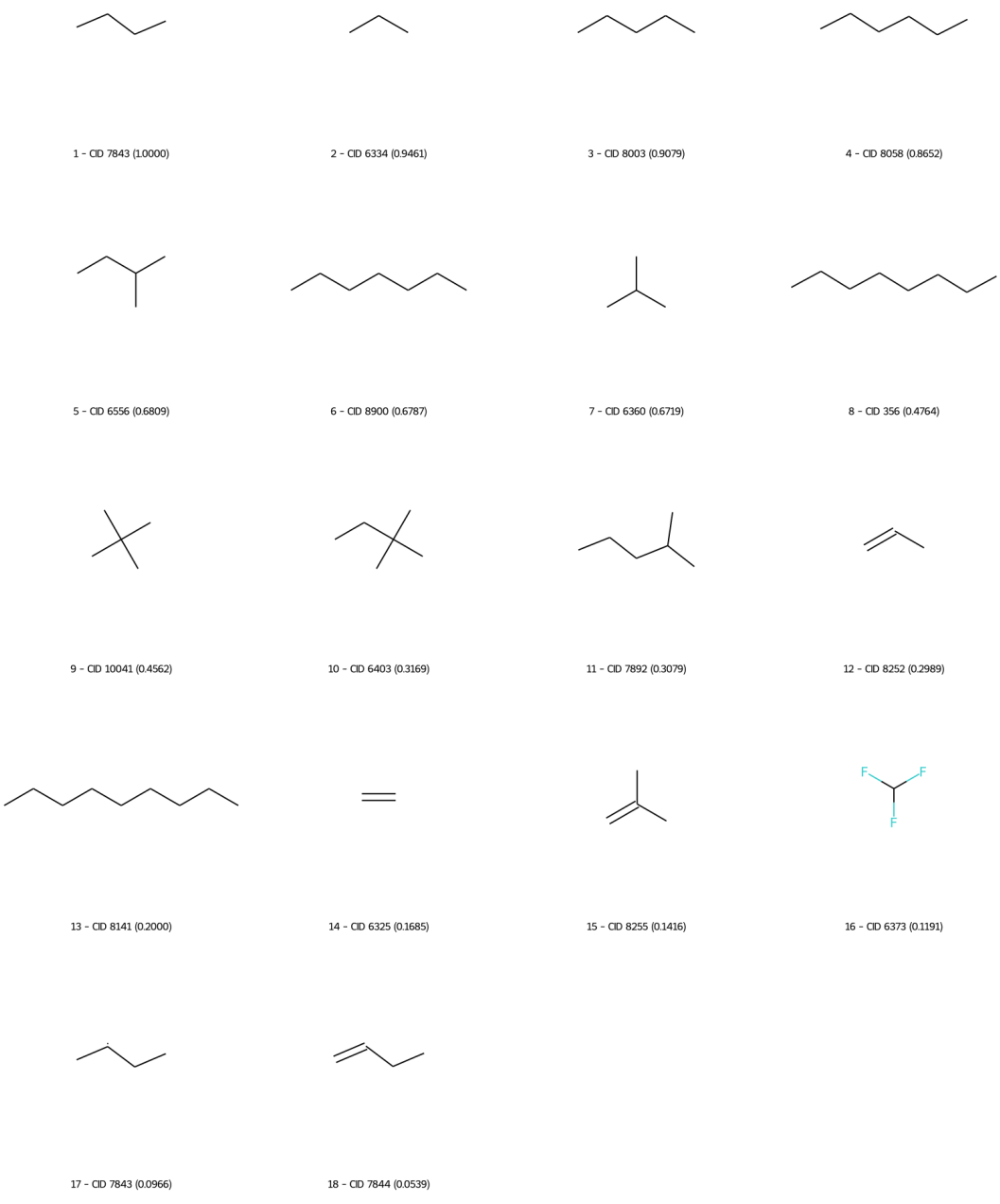

Figure B.8. Top 20 fragments for FreeSolv task.

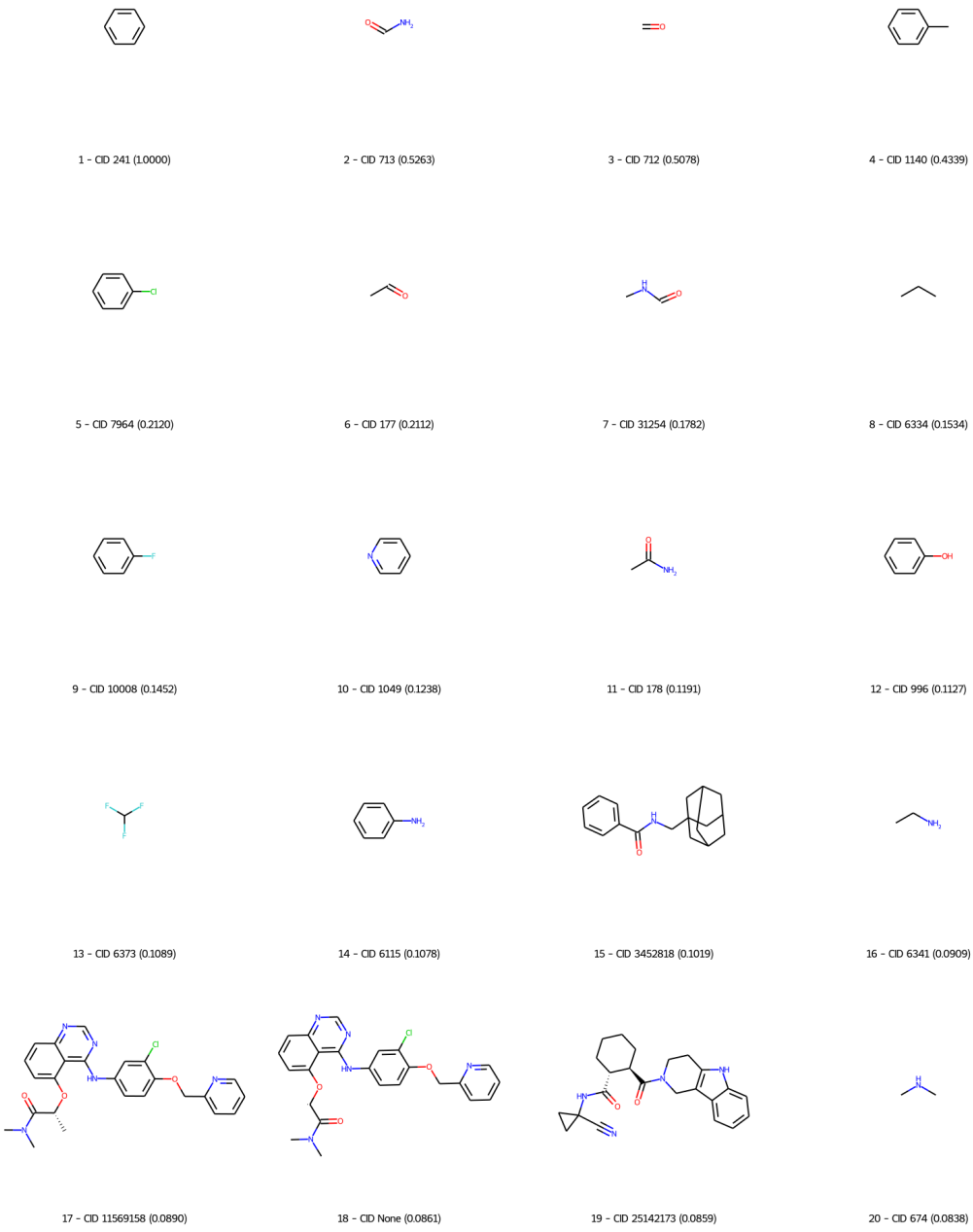

Figure B.9. Top 20 fragments for Lipophilicity task.

APPENDIX C: BALANCE OF TASKS

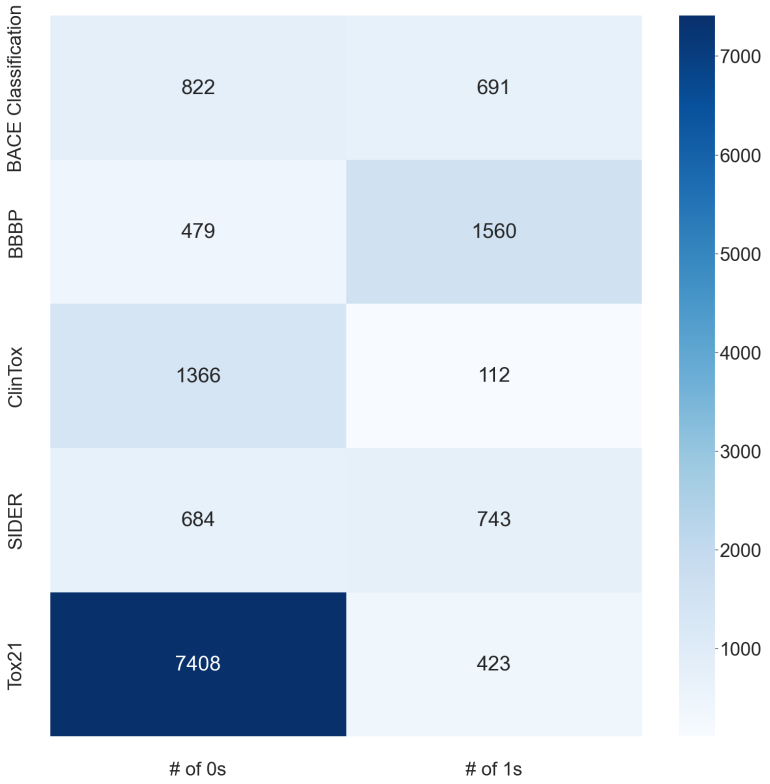

Figure C.1. Balance of classification tasks on heatmap.

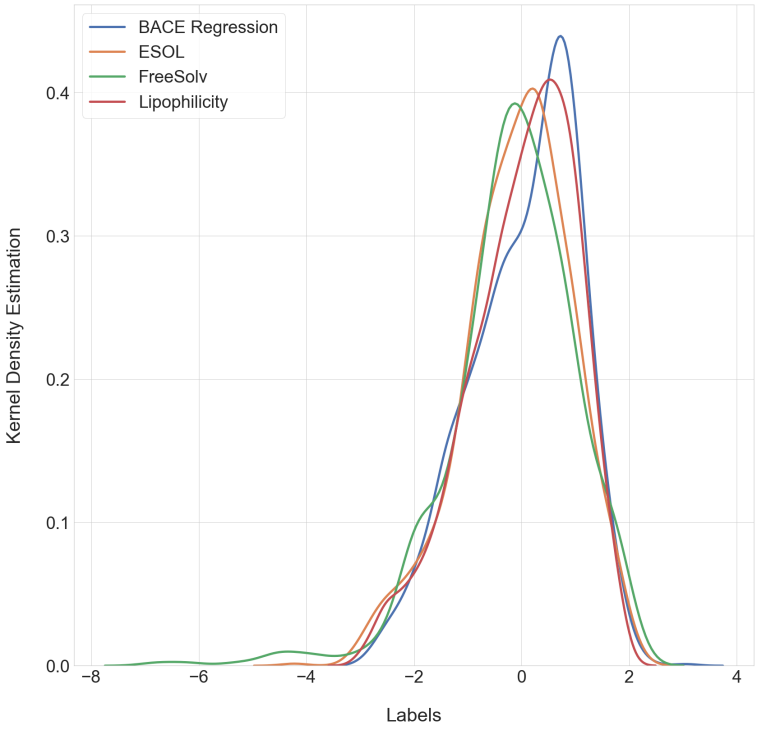

Figure C.2. Balance of regression tasks on KDE plot.

APPENDIX D: SCORES VS EPOCH CURVES OF TASKS

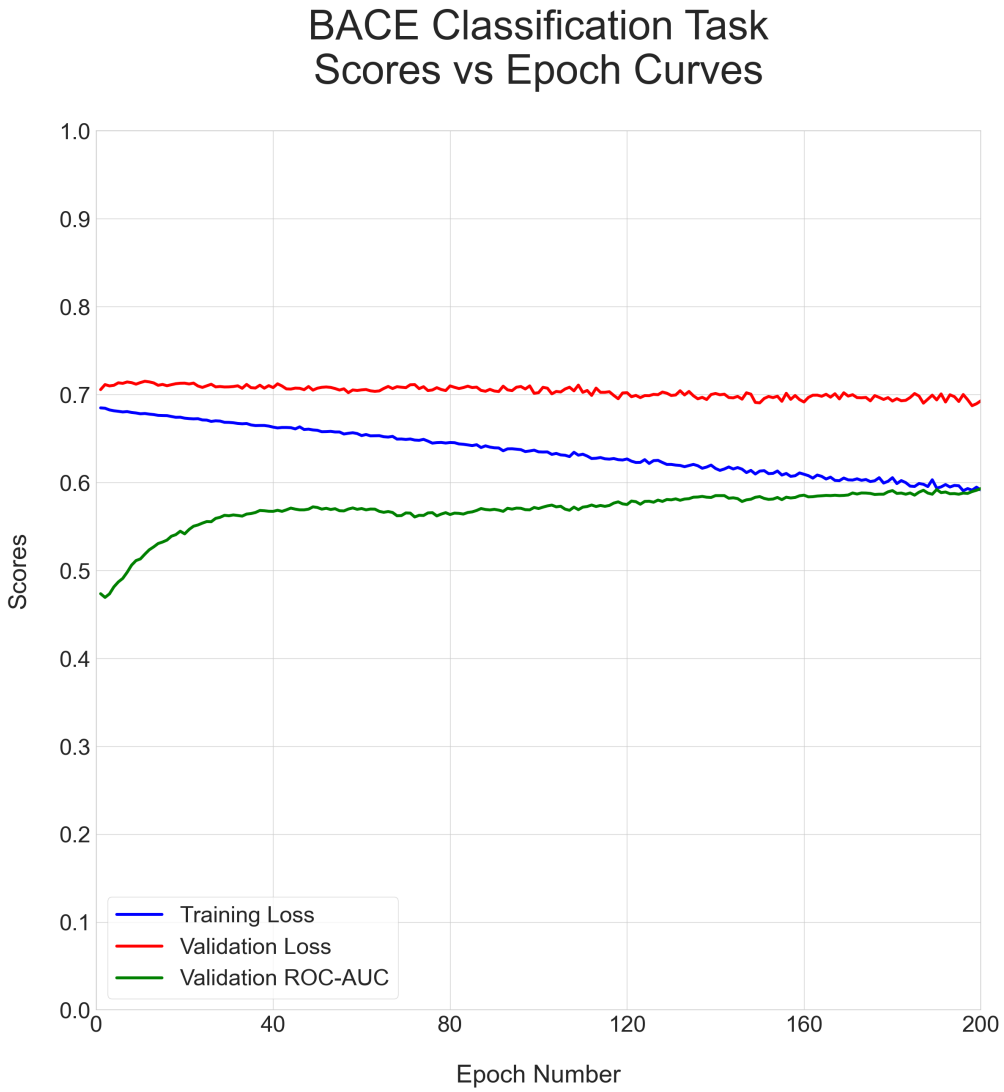

Figure D.1. BACE classification - scores vs epochs curves 1.

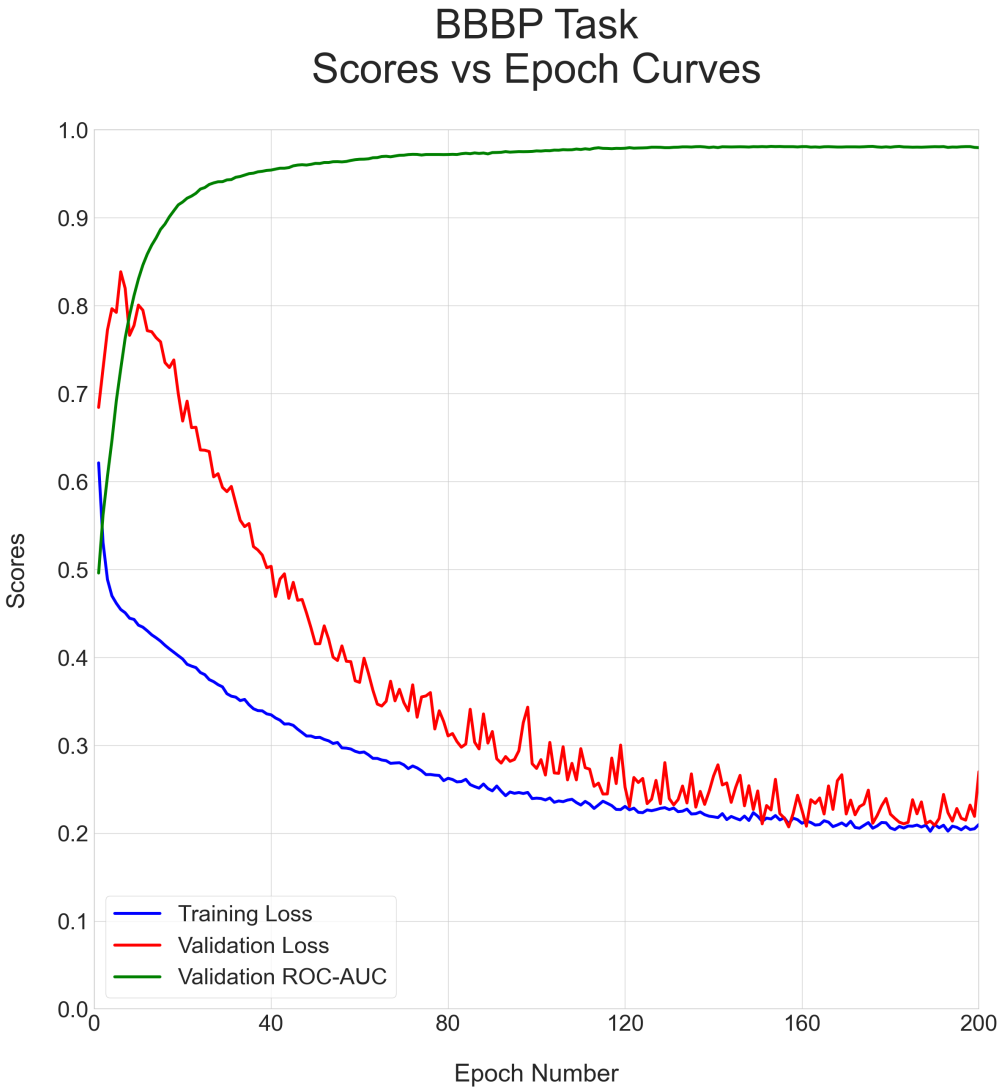

Figure D.2. BBBP - scores vs epochs curves 1.

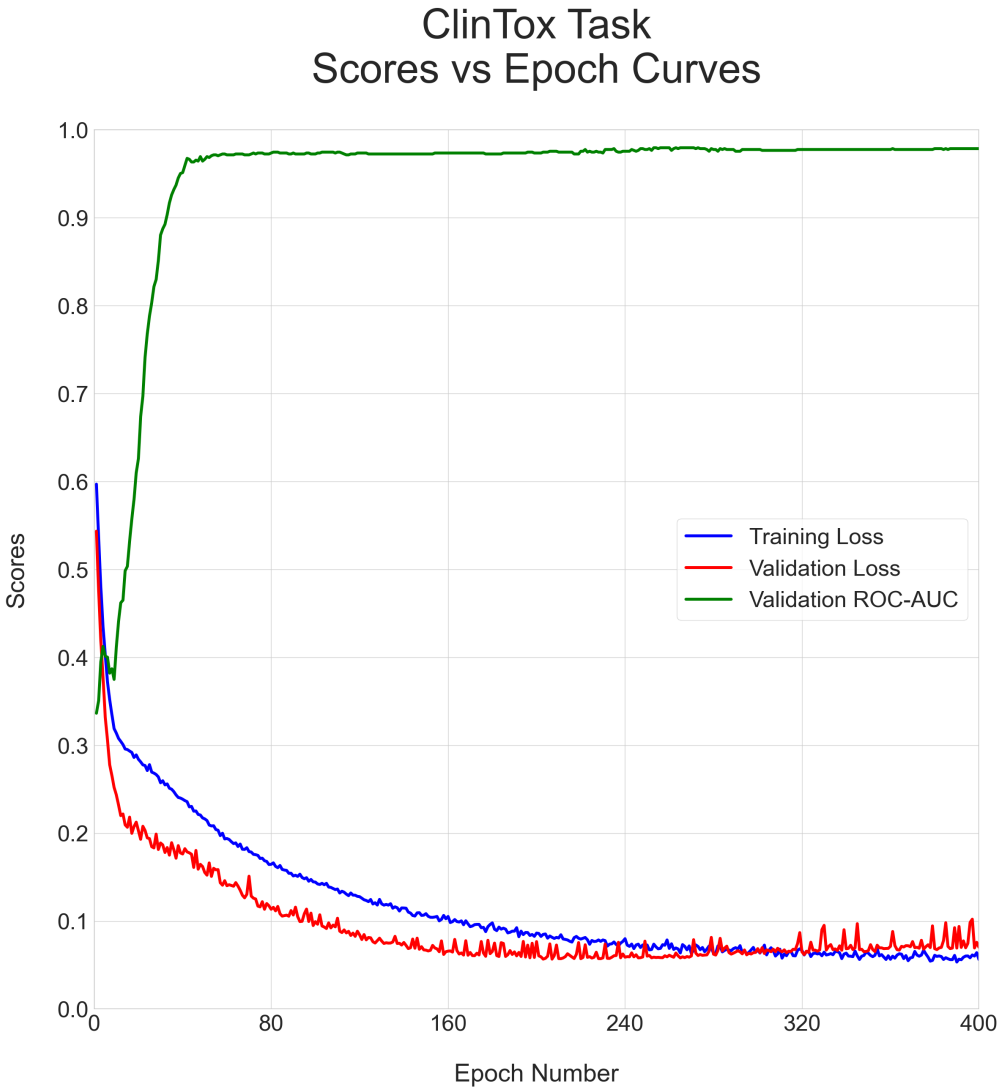

Figure D.3. ClinTox - scores vs epochs curves 1.

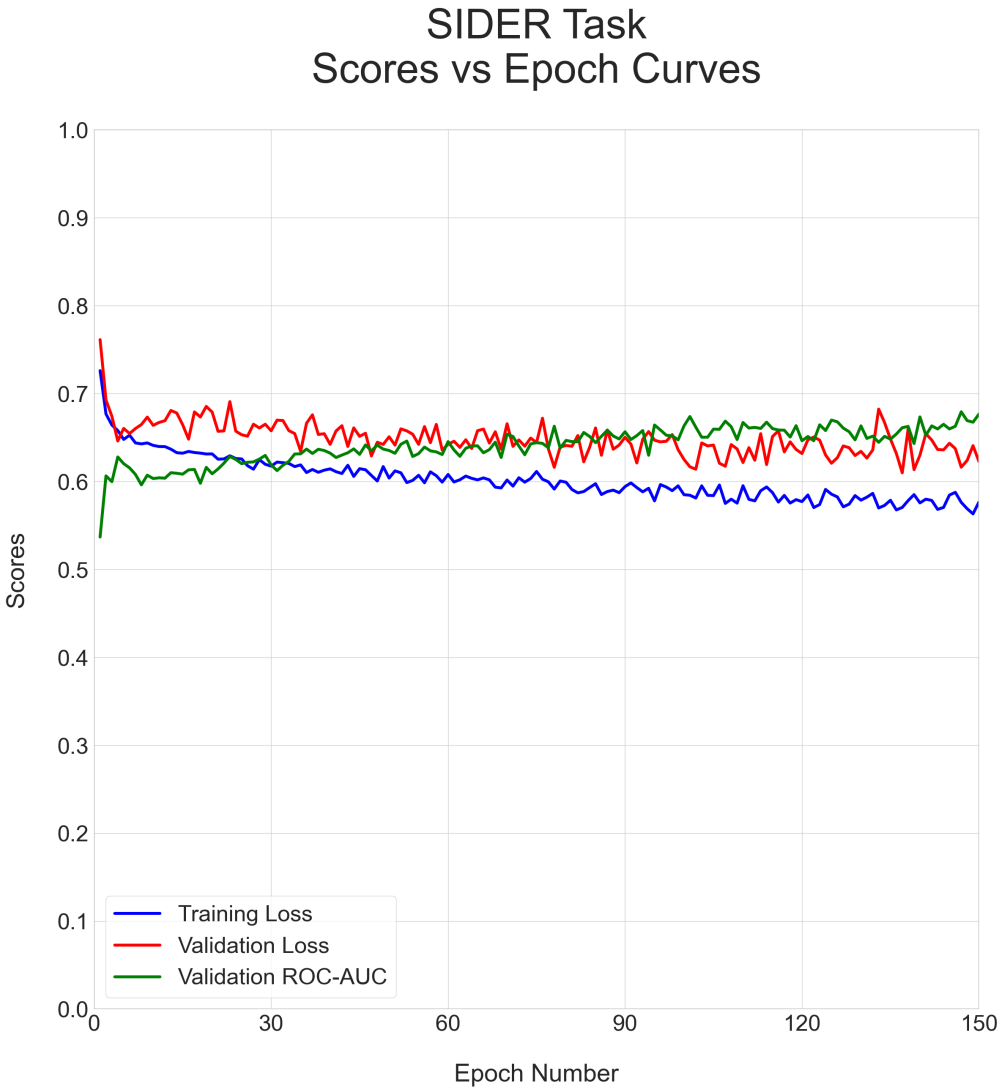

Figure D.4. SIDER - scores vs epochs curves 1.

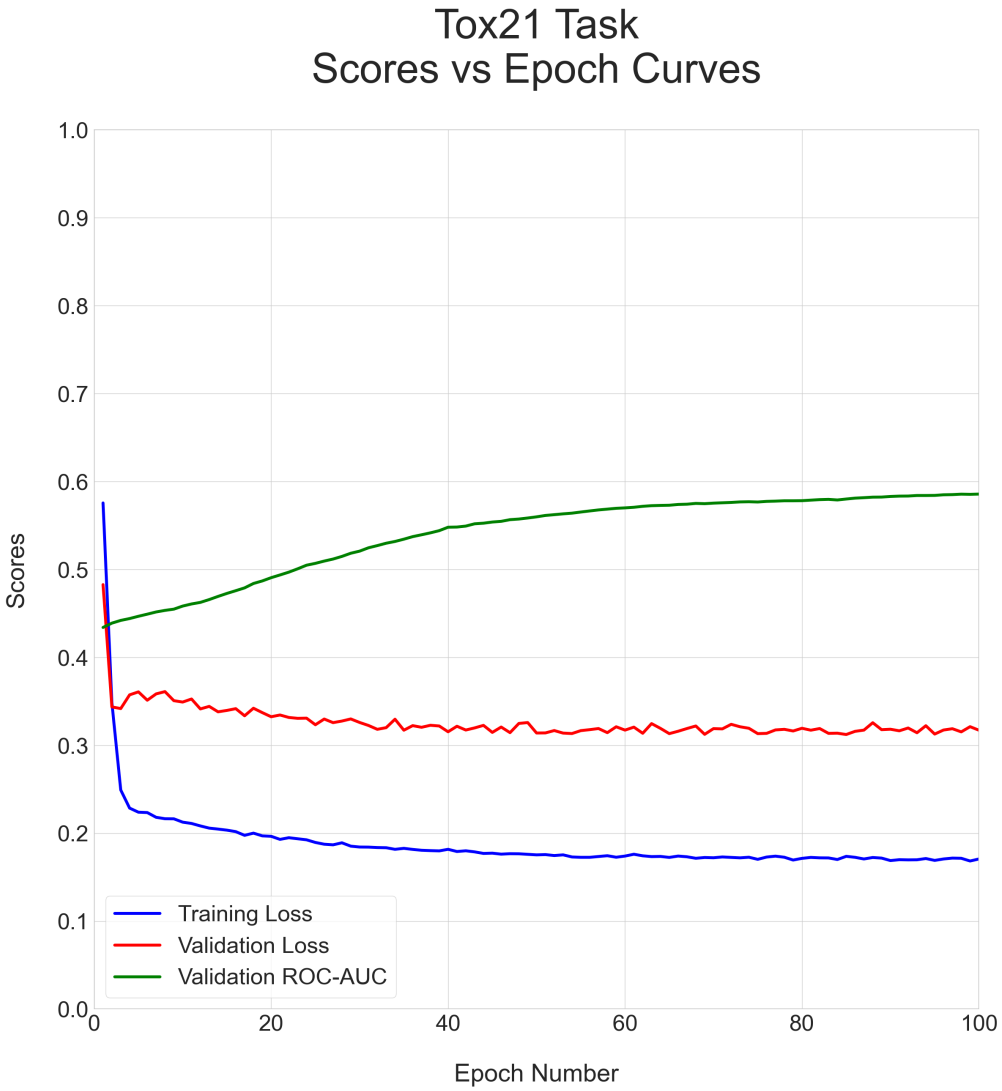

Figure D.5. Tox21 - scores vs epochs Curves 1.

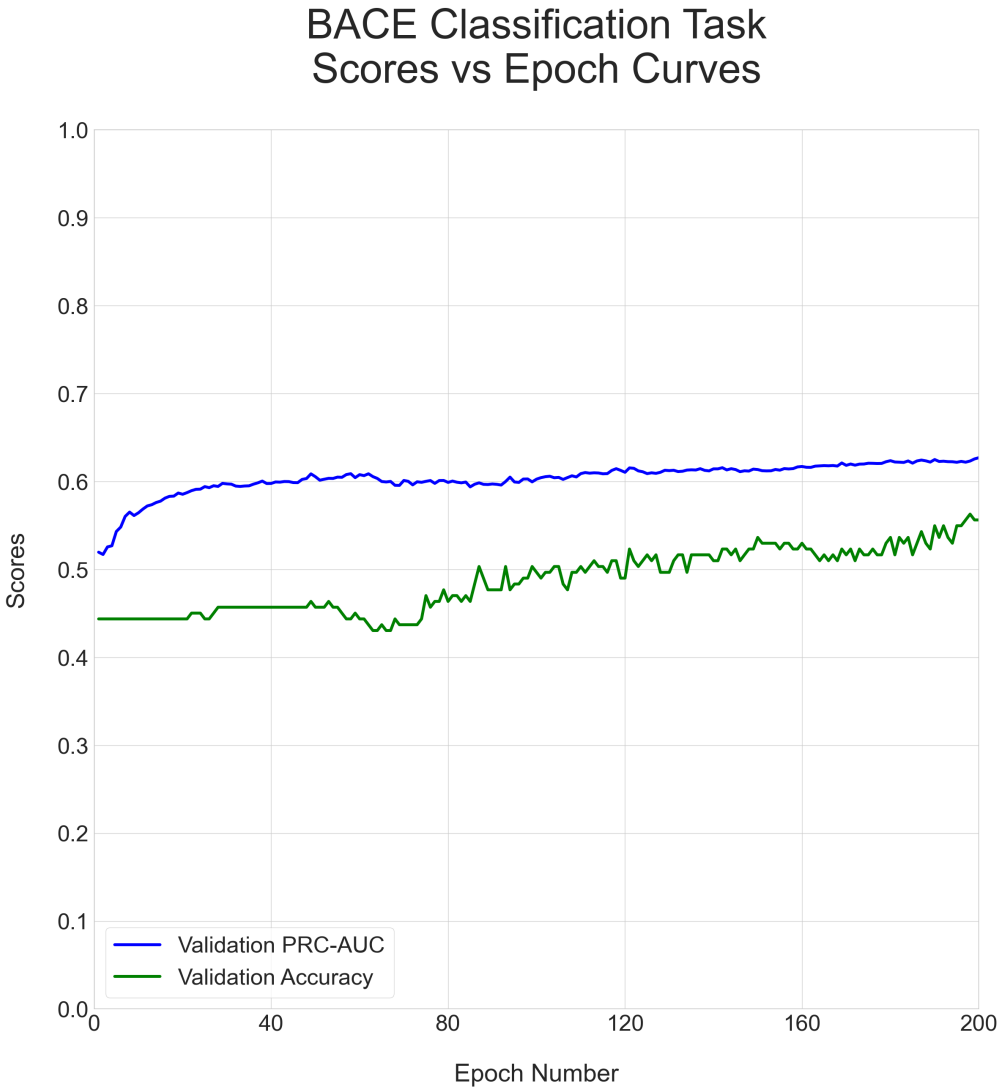

Figure D.6. BACE classification - scores vs epochs Curves 2.

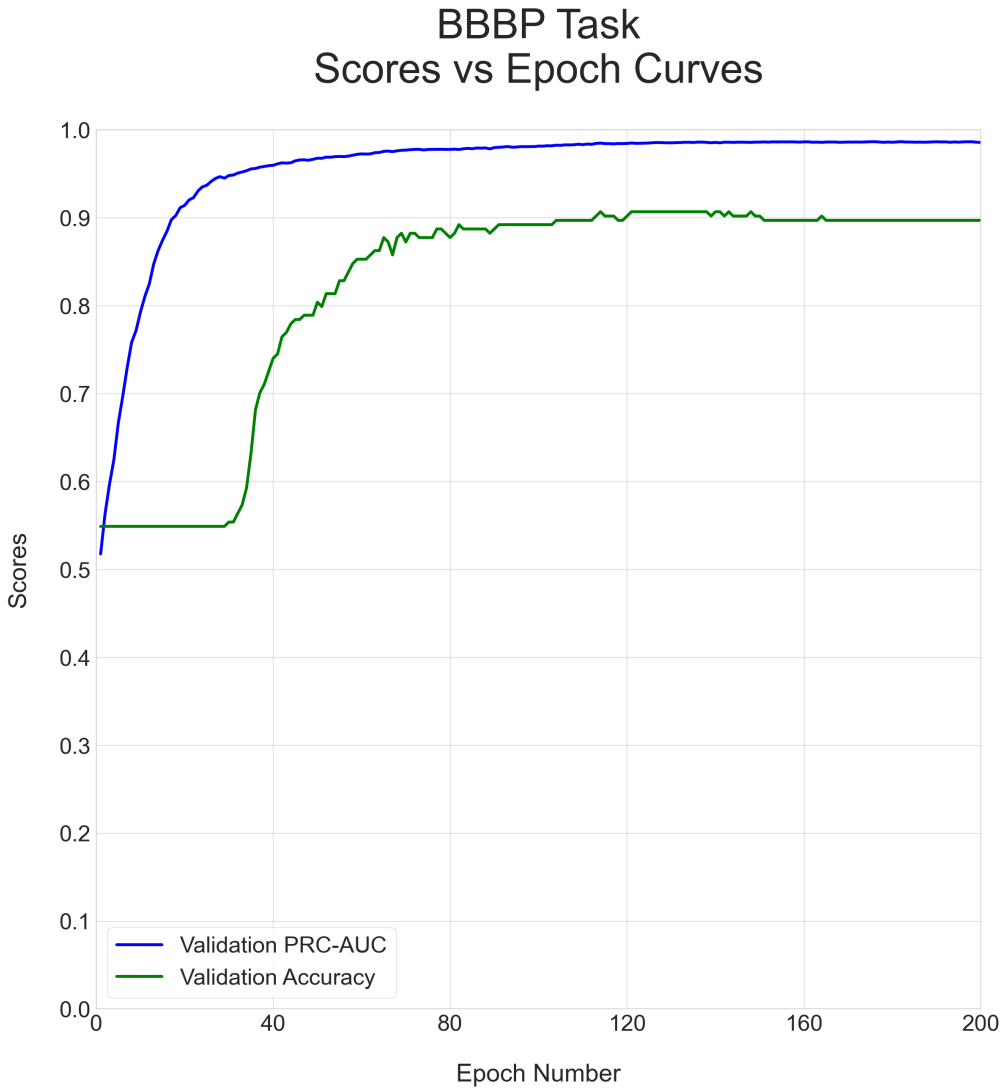

Figure D.7. BBBP - scores vs epochs curves 2.

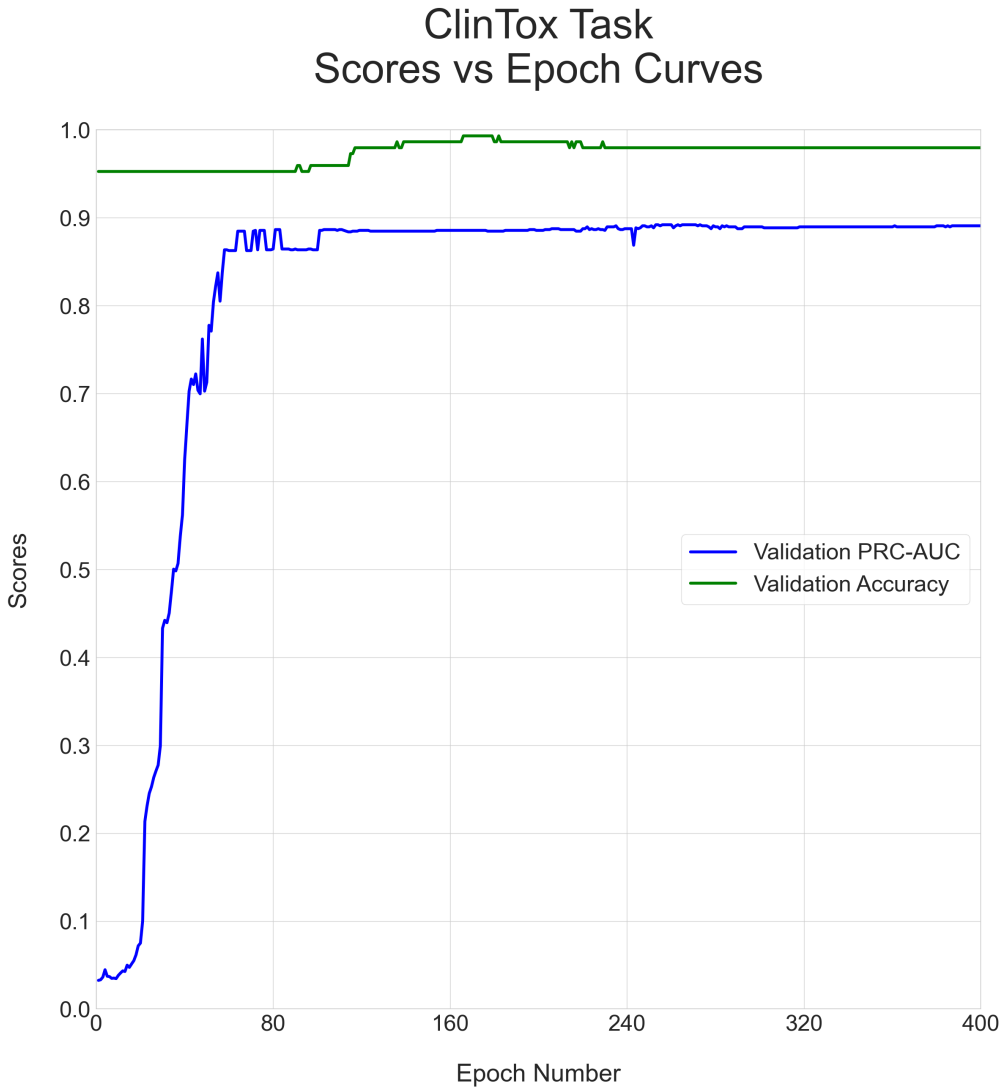

Figure D.8. ClinTox - scores vs epochs curves 2.

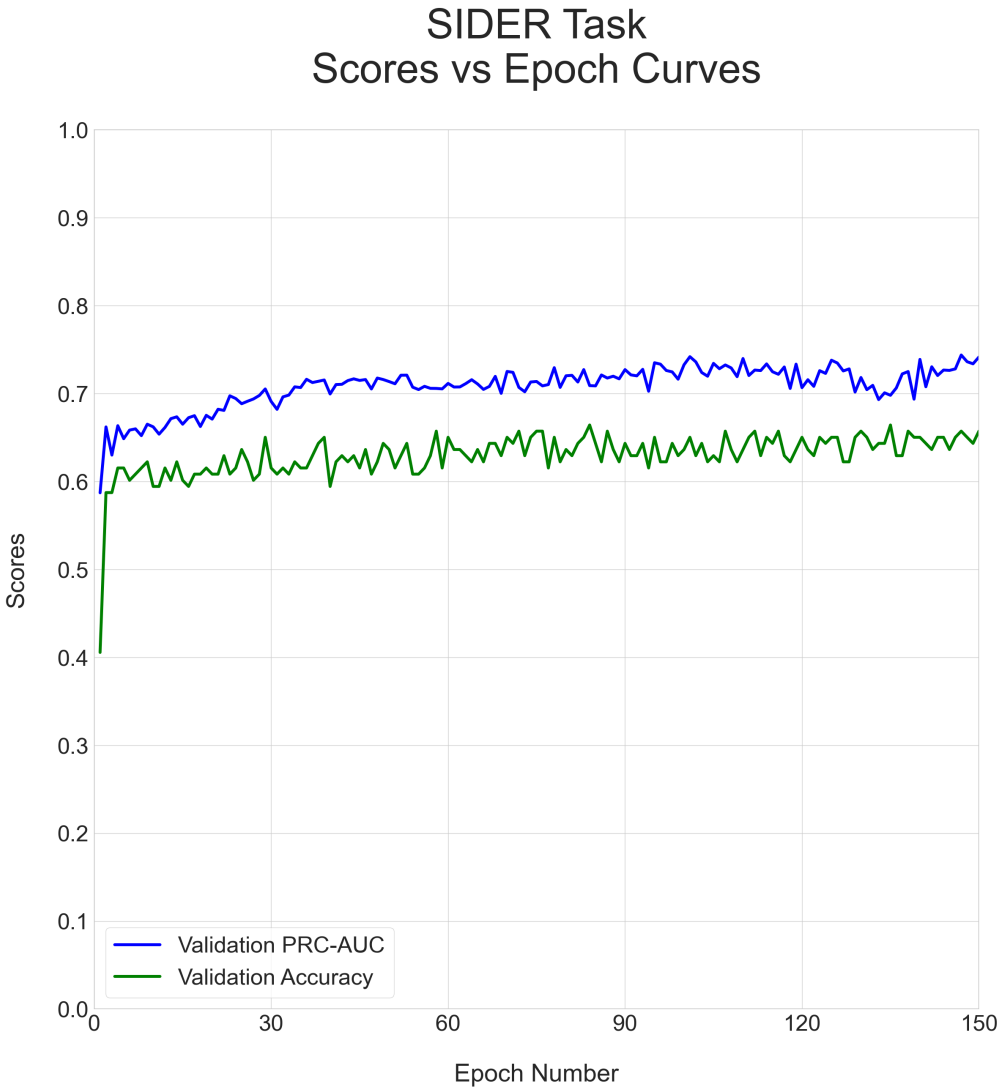

Figure D.9. SIDER - scores vs epochs curves 2.

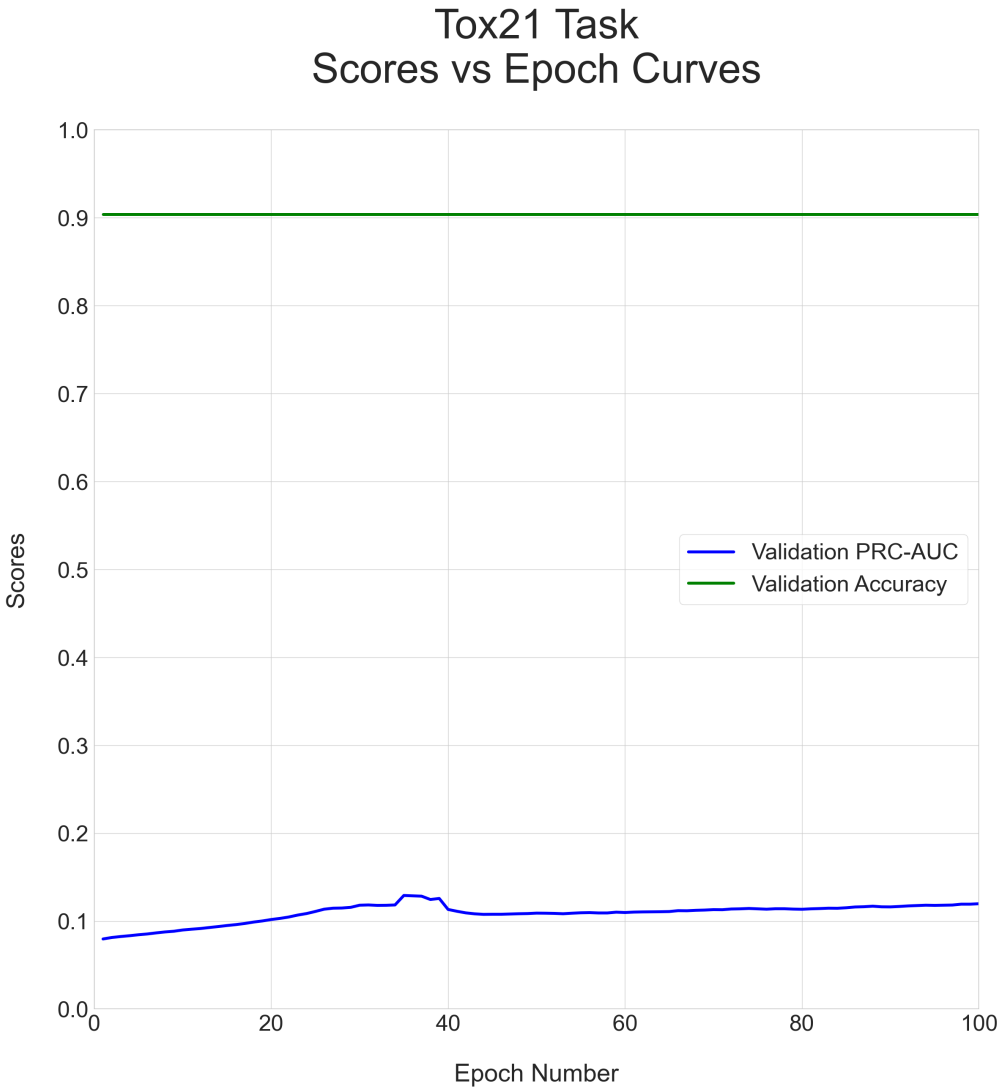

Figure D.10. Tox21 - scores vs epochs curves 2.

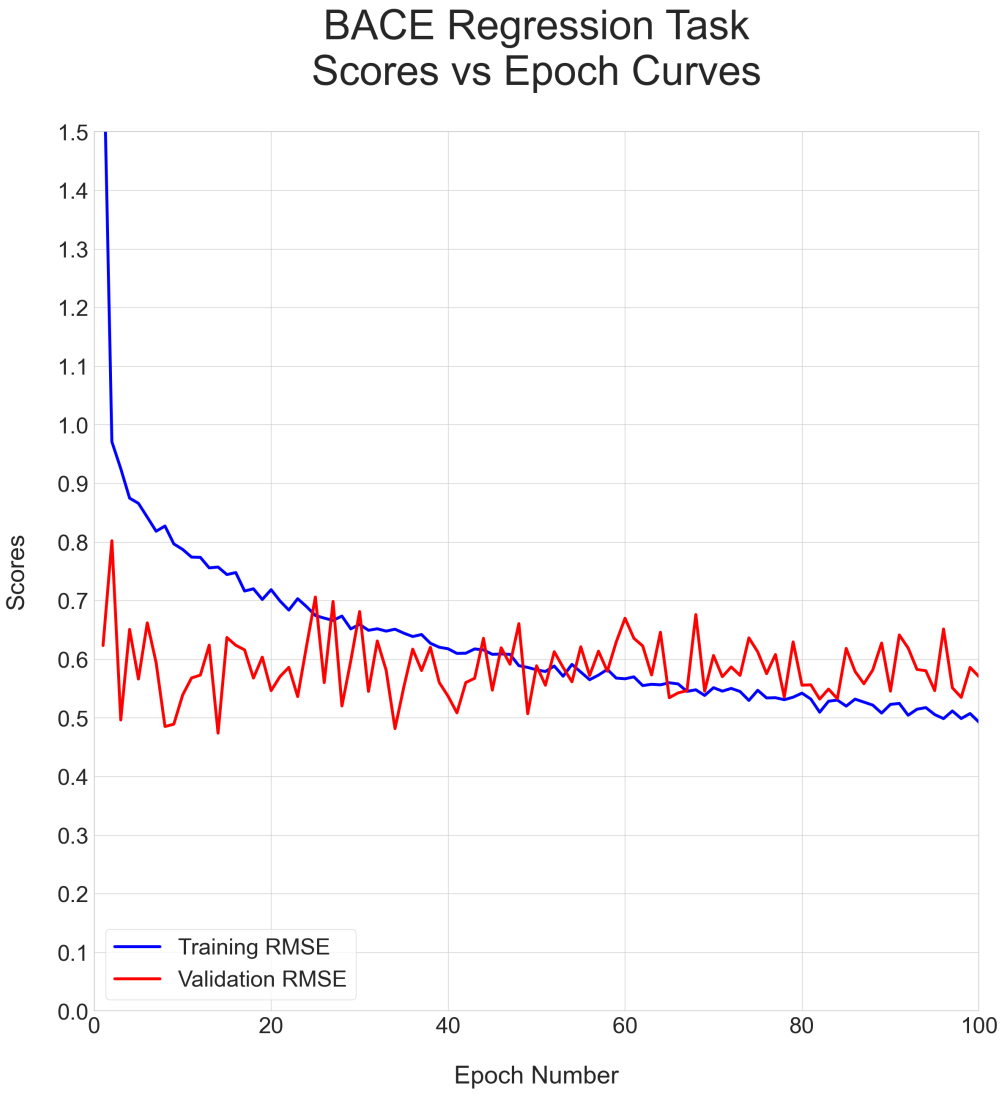

Figure D.11. BACE regression - scores vs epochs curves.

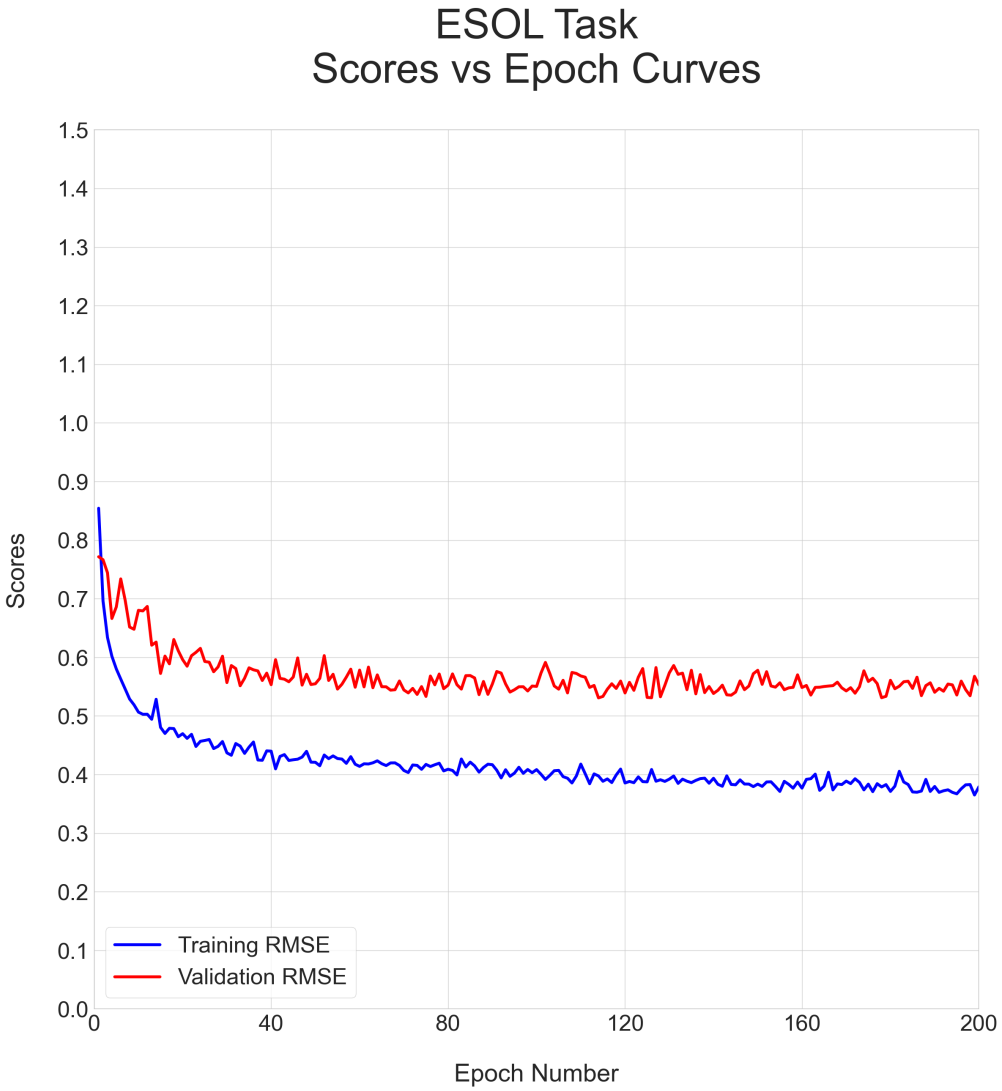

Figure D.12. ESOL - scores vs epochs curves.

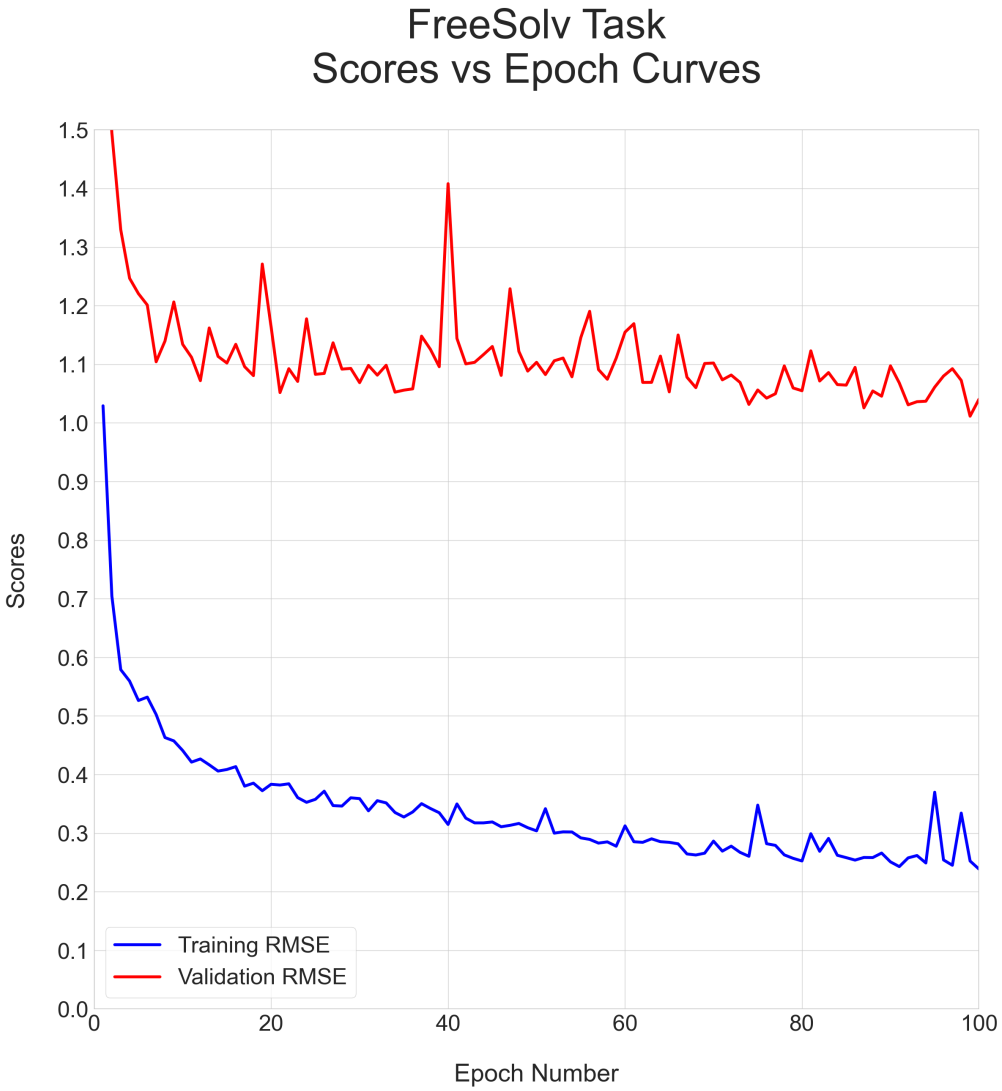

Figure D.13. FreeSolv - scores vs epochs curves.

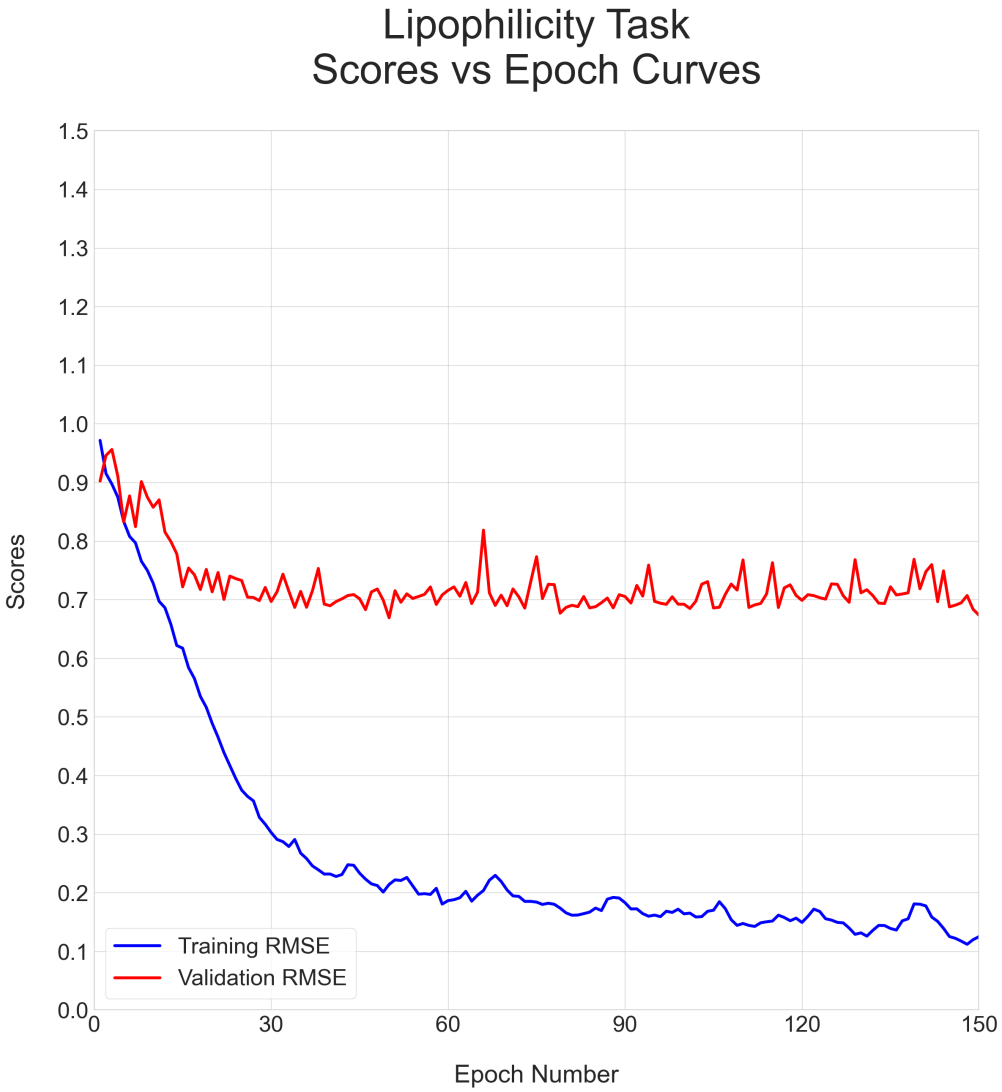

Figure D.14. Lipophilicity - scores vs epochs curves.

APPENDIX E: DETAILS OF 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.

E.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**

E.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**

E.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−]**

E.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**

E.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)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**

E.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) → **CCC#Cc1cccc**([**C@@**]**2**(**c3ccc**(**OCF**)**cc3**)**N** = **C**(**N**)**N**(**C**)**C2** = **O**)**c1**

20 - CID None (0.1394) → **CC#Cc1cccc**([**C@@**]**2**(**c3ccc**(**OCF**)**cc3**)**N** = **C**(**N**)**N**(**C**)**C2** = **O**)**c1**

E.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**

E.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**

E.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)c1cc ccc1 16 -** CID 6341 (0.0909) → **CCN**

17 - CID 11569158 (0.0890) → **C[C@@H](Oc1cccc2ncnc(Nc3ccc(OCc4cc ccn4)c(Cl)c3)c12)C(=O)N(C)C**

18 - CID None (0.0861) → **CN(C)C(=O)[CH]Oc1cccc2ncnc(Nc3ccc(OCc 4ccccn4)c(Cl)c3)c12**

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

20 - CID 674 (0.0838) → **CNC**