1 2 9 0

UNIVERSIDADE D
COIMBRA

Ana Catarina Dias Machado

END-TO-END TRANSFORMER-BASED APPROACH FOR OPTIMIZED *DE NOVO* DRUG DESIGN

Dissertation in the context of the Master in Biomedical Engineering, Specialization in Clinic Informatics and Bioinformatic advised by Prof. Dr. Joel P. Arrais and Nelson R. C. Monteiro, presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

Faculty of Sciences and Technology

Department of Informatics Engineering

End-to-End Transformer-based Approach for Optimized De Novo Drug Design

Ana Catarina Dias Machado

Dissertation in the context of the Master in Biomedical Engineering, Specialization in Clinic Informatics and Bioinformatic advised by
Prof. Dr. Joel P. Arrais and Nelson R. C. Monteiro,
presented to the
Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

1 2 9 0

UNIVERSIDADE D
COIMBRA

This work was developed in collaboration with:

Center for Informatics and Systems of the University of Coimbra

.CISUC

Centre for Informatics and Systems of the University of Coimbra

Agradecimentos

Em primeiro lugar, gostaria de agradecer ao Professor Joel P. Arrais, pelo seu voto de confiança, disponibilidade e orientação. Os seus conselhos e desafios foram importantes no desenvolvimento do trabalho, assim como na minha evolução pessoal, a superar os receios e ver mais além. Obrigada por me ter convidado a integrar o LARN, que apresenta um ambiente de trabalho e entreajuda exemplares para a evolução da ciência.

Agradeço ao Nelson R. C. Monteiro, pela sua constante disponibilidade, dedicação e ajuda prestada. Obrigada por todos os comentários, sugestões e revisões fundamentais ao longo da realização do projeto e que tanto contribuíram para o meu enriquecimento intelectual.

Um agradecimento à professora Maryam Abbasi pelas suas sugestões e conselhos que contribuíram para o melhoramento deste trabalho. Agradeço também ao Tiago O. Pereira, ao Tiago C. Pereira e à Filipa Carvalho pela disponibilidade facultada, e ao Daniel Martins pela sua boa disposição.

Agradeço profundamente aos meus pais, Cristina e Walter, e aos meus irmãos, Ana Íris e o João Valter, pelo amor incondicional, pela dedicação e conselhos sábios, por acreditarem em mim e darem-me forças para ultrapassar as adversidades. Muito obrigada pelo vosso apoio infinito e motivação para atingir os meus objetivos. Agradeço também ao João Jordão pela paciência e disponibilidade.

Ao Hugo Santiago, agradeço o amor, a companhia, paciência e apoio incansável ao longo destes cinco anos. Obrigada por seres a pessoa incrível que és e por estares ao meu lado em todos os momentos, sempre pronto a ajudar.

O meu obrigada aos meus tios Ana Maria e José Luís, à minha prima Leonor e aos meus primos-tios São e Nuno pela preocupação e importante apoio ao longo da minha vida académica.

Aos meus amigos de Coimbra, obrigada por terem tornado o meu percurso académico ainda mais especial, com tão bons momentos. Beatriz Costa, Maria Melo e Rogério Medeiros obrigada pela vossa amizade, pelas memórias e pelas conversas motivadoras.

E a todas as pessoas que possa não ter referido, o meu obrigada.

Dedicatória

Dedico este trabalho aos meus pais, por tudo o que fizeram e fazem por mim, por poder contar sempre com o seu apoio e carinho. Obrigada.

Financing

This research has been funded by the Portuguese Research Agency FCT, through D4 - Deep Drug Discovery and Deployment (CENTRO-01-0145-FEDER029266). This work is funded by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020.

Resumo

Os processos tradicionais de descoberta de fármacos são demorados, de elevado custo e com uma reduzida taxa de aceitação entre as agências de regulamentação. Recentemente, abordagens *in silico* combinadas com metodologias de aprendizagem profunda têm atraído cada vez mais atenção como abordagem de desenho *de novo* medicamentos, levando à descoberta de pequenos compostos com interesse farmacológico num tempo eficiente.

Apesar dos recentes avanços computacionais na geração de moléculas para alvos biológicos, as propriedades físico-químicas essenciais, como a lipofilicidade, a permeabilidade e o peso molecular, são geralmente otimizadas individualmente. Por conseguinte, os resultados ficam comprometidos, devido à eficiência farmacológica ser influenciada por uma variedade de fatores simultâneos. Neste projeto, propomos uma arquitetura baseada em Transformadores para gerar novas moléculas com propriedades farmacológicas desejadas e atividade de ligação relevante contra um alvo com interesse biológico. A arquitetura combina um Gerador de Transformador-Descodificador para produzir novos compostos válidos, um Preditor de Transformador-Codificador para estimar a atividade de ligação e um ciclo de Feedback baseado num algoritmo de otimização multiobjetivo para otimizar o gerador de acordo com as propriedades desejadas.

Os resultados demonstraram a eficácia da estratégia proposta para gerar compostos químicos novos e sintetizáveis. O Gerador baseado em Transformador superou os modelos do estado da arte na métrica novidade. A otimização do modelo não enviesado resultou em 99.79% de moléculas válidas geradas com uma taxa de conformidade de 99.36% com a regra dos Cinco de Lipinski e uma elevada afinidade de ligação ao recetor de adenosina A2A. Os resultados obtidos demonstraram a capacidade do modelo em selecionar componentes críticos no espaço químico para melhorar o interesse biológico e propriedades farmacológicas das moléculas.

Palavras-Chave

Descoberta de Fármacos, Desenho de Fármacos, Aprendizagem Profunda, Transformadores, Otimização Multiobjetivo.

Abstract

Traditional drug discovery processes are time-consuming, expensive, and have a low acceptance rate among regulatory agencies. In recent years, *in silico* approaches combined with deep learning methodologies have attracted increasing attention to address the *de novo* drug design challenge, leading to the discovery of small compounds with pharmacological interest in an efficient amount of time.

In spite of the recent computational advances for generating molecules for biological targets, crucial physicochemical properties, such as lipophilicity, permeability, and molecular weight, are usually individually optimized. On that account, the results are compromised since the pharmacological efficacy is influenced by a variety of simultaneous factors. In this study, we propose a Transformer-based architecture to generate new molecules with desirable pharmacological properties and relevant binding activity against a target with biological interest. The architecture combines a Transformer-Decoder Generator to produce valid new compounds, a Transformer-Encoder Predictor to estimate the binding activity, and a Feedback Loop based on a multi-objective optimization algorithm to optimize the generator according to the desired properties.

The results demonstrate the effectiveness of the proposed framework to generate novel and synthesizable chemical compounds. The Transformer-based Generator outperformed state-of-the-art baselines in the novelty metric. The optimization of the unbiased model resulted in 99.79% generated valid molecules with a 99.36% compliance rate with Lipinski's Rule of Five and a high binding affinity to the A2A adenosine receptor. Overall, these results demonstrated the model's capacity to select critical components in the chemical space in order to improve the biological interest and pharmacological properties of the molecules.

Keywords

Drug Discovery, Drug Design, Deep Learning, Transformers, Multi-Objective Optimization.

Contents

List of Tables

List of Figures

Abbreviations

AA2AR	Adenosine A2A Receptor
AAE	Adversarial Autoencoder
Adam	Adaptive Moment Estimation
ADME	Absorption, Distribution, Metabolism and Excretion
ADMET	Absorption, Distribution, Metabolism, Excretion and Toxicity
ANN	Artificial Neural Network
BERT	Bidirectional Encoder Representations from Transformers
CCC	Concordance Correlation Coefficient
DL	Deep Learning
FCNN	Fully Connected Neural Network
FDA	Food and Drug Administration
FG	Functional Group
GAN	Generative Adversarial Networks
IC50	Half-Maximal Inhibitory Concentration
Kd	Equilibrium Dissociation Constant

Ki	Inhibition Constant
KNN	K-Nearest Neighbors
LBDD	Ligand-Based Drug Design
LN	Layer Normalization
LSTM	Long Short-Term Memory
MHSA	Multi-Head Self-Attention
ML	Machine Learning
MLM	Masked Learning Modeling
MOP	Multi-Objective Optimization Problem
MSE	Mean Squared Error
MW	Molecular Weight
NLP	Natural Language Processing
ORGAN	Objective-Reinforced Generative Adversarial Networks
pIC50	Negative Logarithm of the Half-Maximal Inhibitory Concentration
PSA	Polar Surface Area
PWFFN	Position-Wise Feed-Forward Network
QED	Quantitative Estimate of Drug-Likeness
QSAR	Quantitative Structure-Activity Relationship
QSPR	Quantitative Structure-Property Relationship
RAdam	Rectified Adaptive Moment Estimation
RANC	Reinforced Adversarial Neural Computer
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network

SAS	Synthetic Accessibility Score
SBDD	Structure-Based Drug Design
SMILES	Simplified Molecular Input Line Entry System
SVM	Support Vector Machine
SVR	Support Vector Regression
Tanh	Hyperbolic tangent
TPSA	Topological Polar Surface Area
VAE	Variational Autoencoder

1

Introduction

1.1 Context

Drugs

Drugs play an important role in the overall health of the human race, enhancing the quality of life through the treatment, prevention, and management of diseases and other clinical conditions. The initial archaic drugs were based on mixtures of certain plants, however, given the recent advances in technology, numerous possible sources of drugs are now available, such as chemically synthesized, synthesized with the extraction of active elements from natural substances, and originated by organisms such as fungus [1].

There are currently a variety of medications available, including analgesics for pain alleviation, antibiotics for treating bacterial infections, and antivirals for curing viral infections [2]. Most drugs' therapeutic effects result from interactions between pharmaceuticals and protein structures called receptors. Depending on the medication's action modes, their interactions induce diverse biological responses. Hence, drugs can be classified as receptor agonists or antagonists. An agonist is a molecule that causes a functional response upon binding to a receptor, whereas an antagonist inhibits or decreases a physiological reaction [3].

Drug Discovery and Design

The dug discovery process comprises several steps, including target identification, target validation, lead identification, lead optimization, pre-clinical research, clinical trials, and regulatory approval [4].

In target identification, studies are performed to discover targets (proteins or genes) that are directly or indirectly associated with a disease, whereas in target validation, tests are conducted to evaluate the therapeutic effect and regulation of the discovered targets [4, 5].

Lead identification corresponds to the identification of molecules potentially active on desired targets. On the other hand, lead optimization aims to improve the efficiency, selectivity, and absorption, distribution, metabolism and excretion properties (ADME) of the lead compounds [4].

Pre-clinical *in vivo* and *in vitro* investigations are conducted prior to human application to determine absorption, distribution, metabolism, excretion and toxicity properties (ADMET), dosage, and efficacy [4, 6].

Clinical trials are conducted on humans and consist of three phases. In phase I, the drug's safety and dosage are tested on a small group of diseased and healthy people. In phase II, the drug is administered to a larger group of patients with the disease to assess its efficacy. In the final phase, a larger sample of patients is used to conduct a more reliable statistical analysis of the drug's effectiveness and adverse effects [7].

Finally, a document with the results of the drug's tests is delivered to a regulatory agency, such as the Food and Drug Administration (FDA) in the United States, which decides whether or not to approve the drug. Pharmaceutical companies continue to conduct studies on drugs that are already on the market in order to check for side effects and dosage modifications [5, 7].

Traditional drug discovery approaches include screening a large set of molecules to identify a viable drug, but their percentage of success is low [8]. Thus, the advancement of computational methods has revolutionized the identification of targets and the design of novel compounds.

The drug design procedure corresponds to the design of molecules with a structure capable of binding to the biological target and possessing properties that enable them to reach the site of action and confer a therapeutic effect [9]. Computer-aided drug design reduces the overall amount of time and cost necessary associated with the lead identication and lead optimization stages. These *in silico* methods can be categorized as either Structure-Based Drug Design (SBDD) or Ligand-Based Drug Design (LBDD) [8]. SBDD uses structural information from a specific biological target to create new molecules. The three-dimensional structure can be determined by techniques such as nuclear magnetic resonance or X-ray crystallography. On the other hand, LBDD does not require structural information from targets since it employs predictive *in silico* approaches to assess the affinity of compounds to receptors by learning from the known affinities of similar molecules [8, 10].

De novo drug design is a computational methodology to generate new compounds from blocks without a starting structure, i.e., without any preexisting connections between the blocks. Furthermore, it can take into account or discard the information of the target's binding sites, i.e., it can be of the SBDD or LBDD type, respectively. *De novo* drug design provides a vast opportunity for the discovery of drugs with novel or enhanced medicinal benefits, in a more extensive chemical space [10].

1.2 Motivation

The rising prevalence of chronic diseases combined with an aging population has increased the pressure on the pharmaceutical industry and the healthcare system to perform more efficiently [11]. Furthermore, advanced technologies need to be used to improve the drug discovery process in order to effectively prevent and treat new diseases and potential future epidemics with fewer financial and time resources.

Discovering and developing a new drug and putting it on the market is time-consuming, taking up to 15 years to achieve a potential drug [12], and expensive, with estimates putting the cost of drug development at $2.6 billion [13]. Furthermore, this procedure has a low chance of approval since the drugs undergo extensive testing to determine whether they are safe and efficient for public human consumption. In recent years, Machine Learning (ML) and Deep Learning (DL) have made significant contributions to drug discovery by identifying drug-target interactions and generating novel compounds based on large amounts of available researched data.

A potential drug should have an effective affinity for the biological target in order to reduce the risk of unintended side effects as well as physicochemical and toxicity properties that allow it to reach the target [14]. Characteristics that combined play a key role in the success of a drug, such as lipophilicity, Molecular Weight (MW), and Polar Surface Area (PSA), are usually considered individually. Since ignoring one of the drug-like properties could make a drug less effective, this could be seen as a multi-objective optimization problem where the goal is to optimize the most important therapeutic properties. Using large biochemical databases and combining them with computational approaches containing multi-objective optimization algorithms may lead to promising results in *de novo* drug design.

Functional groups (FGs) are sets of atoms that significantly influence the characteristics of the molecules they integrate. Similar FGs often have identical attributes [15], therefore, these atoms may be considered as a source of relevant information to be used in computional models, especially for predicting molecular properties.

1.3 Objectives

The main purpose of this dissertation is to develop an end-to-end transformer-based methodology capable of generating molecules using the Simplified Molecular Input Line Entry System (SMILES) notation for a specific biological target and with desired physicochemical properties. The objectives to be achieved are the following:

1. Implementation and pre-training of a Transformer-Encoder Predictor using Masked-Learning Modeling (MLM) in order to learn the chemical context present in the SMILES.
2. Fine-tuning of a predictor to estimate the binding activity of the compounds towards a relevant biological target.
3. Evaluate and compare the predictive performance of the Transformer-Predictor with other machine learning models.
4. Development of the Transformer-Decoder Generator deep neural network and comparison of its effectiveness with other deep learning models.
5. Adjustment of the generator using a multi-objective optimization approach to generate drug-like compounds.
6. Exploration of alternative molecular property combinations in the multi-objective optimization algorithm to optimize the generator.

1.4 Workflow

Considering the increasing challenges of the health system, exploring computational drug design models is crucial to efficiently produce drug-like and active compounds for relevant targets. In this study, transformer-based models are employed for both the generation of drugs in the SMILES format, and estimation of compounds' binding activity to a biological target. Therefore, these deep neural networks learn short and long-term dependencies between units in a molecular sequence and apply this knowledge in the respective model task. Moreover, the generative model is fine-tuned in a feedback loop that shifts the generator toward the desired properties. The feedback loop integrates a multi-objective optimization algorithm to sort and choose the best molecules created to be used as input data. The production of drug-like, target-selective, and synthesizable molecules accelerates lead identification and lead optimization phases, reducing the number of experimental attempts. Furthermore, it achieves a more extensive chemical space of potential drugs than traditional methods.

1.5 Document Structure

This document is organized into seven chapters. Chapter 2, Computational Techniques in Drug Discovery, provides an overview and discussion of the principal computational strategies used in drug discovery. It describes the most commonly considered molecular representations and physicochemical properties. In addition, it mentions several studies related to the estimation of therapeutic properties and design of lead compounds. The Chapter 3, Models, presents the architectures of the deep neural networks used to build the proposed framework. Chapter 4, Methods, describes the methodologies employed in each submodel of the main framework and the data preprocessing used. Chapter 5, Experimental Setup, specifies the datasets employed along with the selected hyperparameters and the machine learning approaches used. Chapter 6 presents the results achieved by each model and its analysis as well as the limitations of the framework. In Chapter 7,the main conclusions of this study are provided as well as potential future research suggestions.

2

Computational Techniques in Drug Discovery

Drug discovery is a complex and time-consuming process with numerous steps. The identification of potential molecules that are pharmacological active for a given relevant biological target remains one of the major challenges in drug discovery, especially due to the fact that conducting bioassays is not feasible for the vast chemical space, compromising the effectiveness of these traditional approaches [4]. Therefore, several computational approaches have been proposed to increase the efficacy and capability of drug research. These methods use different molecular representations as input and take into consideration multiple pharmacological properties for prediction and generation tasks. This thesis explores two techniques: prediction of binding activity and *de novo* drug design.

2.1 Molecular Representations

In order for the chemoinformatic algorithms to detect patterns and make predictions, it is crucial to select proper molecular nomenclatures to represent the chemical compounds. On that account, multiple chemical representations have been proposed and used over time, such as molecular graphs, SMILES, and molecular descriptors. Thus, a proper representation is crucial for the efficiency of chemoinformatic algorithms.

Molecular Graphs

Graphs define the atoms and bonds of chemical structures as nodes and edges, respectively. However, matrix representations are necessary for computers to understand atomic configurations. Therefore, the adjacency matrix, a binary matrix that relates nodes and edges, represents the atomic connections. Furthermore, nodes and edges are represented by the node feature matrix and edge feature matrix, respectively, where the columns contain chemical characteristics, such as atom and bond types [16].

Simplified Molecular Input Line Entry System

SMILES strings adopt characters to describe the graphical molecular structure of the compounds. Atoms are represented by their chemical symbols, where the first letter is capitalized for non-aromatized atoms and lowercase for aromatized atoms. Furthermore, the default is the hydrogen atom omission. The chemical single, double, and triple bonds are denoted by the symbols -, =, and #, respectively, and the branches are expressed by parenthesis. Moreover, the opening and closing of rings in cyclic structures are indicated by numbers in the respective atoms [17]. It is a non-unique notation, as multiple SMILES strings can represent the same molecule. Nonetheless, several tools, such as the RDKit library [18], canonicalize SMILES to standardize potential inconsistencies [16].

Molecular Descriptors

Molecular descriptors describe molecules based on their physicochemical and structural characteristics. They are determined with software tools (e.g., Dragon [19], RDKit [18]). Chemical descriptors are divided into structural keys and fingerprints. Structural keys use bits to represent the presence or absence of predefined chemical structures. Fingerprints are a type of representation widely used in Quantitative Structure-Activity Relationship (QSAR) machine learning, which considers the physical, chemical, and structural properties of compounds [16]. Extended Connectivity Fingerprints is a popular circular fingerprint that uses a variant of the Morgan algorithm to identify structures through neighboring circular atoms [20].

2.2 Molecular Properties

Appropriate physicochemical and biological properties are fundamental to a compound's therapeutic efficacy. Therefore, property identification in the early stages of drug discovery is crucial to the efficiency of the investigation process, both from a drug approval and financial perspective.

Binding Affinity

Binding affinity quantifies the interaction of molecules with particular biological targets. This property is influenced by intermolecular forces such as hydrogen bonding, Van der Waals and hydrophobic forces, and electrostatic interactions [21]. Moreover, the environment in which the interaction occurs also affects biological interaction. After a ligand-receptor pair is formed the conformation of the target is modified.

Binding affinity can be assessed using parameters such as equilibrium dissociation constant (Kd), inhibition constant (Ki), and half-maximal inhibitory concentration (IC50). Therefore, a high value of any of these metrics means a weak interaction. Kd is the proportion between the dissociation and association ratio constants of the chemical reaction caused by the biological binding at equilibrium. Ki represents the dissociation caused by the interaction of an inhibitor to a biomolecule, which reduces the target's activity [22]. IC50 is the drug concentration required to inhibit 50% of a receptor's activity and is typically determined by bioassay. In drug discovery, IC50 is the most frequent method for measuring the potency and efficacy of a potential antagonist drug.

IC50 has a wide range of possible values, from nanomolar to micromolar. Therefore it is converted to the negative logarithm of the half-maximal inhibitory concentration (pIC50) [23]. Thus, molecules with a high pIC50 exhibit a strong binding affinity to their target.

Lipophilicity

The lipophilicity of a molecule describes its affinity for lipophilic regions. It is one of the most crucial factors to consider in drug discovery, as it influences multiple characteristics of drugs, including membrane permeability, solubility, toxicity, and biological activity [24]. Lipophilicity is commonly defined by the logarithm of the partitional coefficient between octanol and water, i.e., LogP. Several methods can be used for calculating LogP. Wildman et al. (1999) provided a technique for determining this metric based on the LogP contributions of the constituent atoms of the molecules [25].

Molecular Weight

A large number of approved drugs are small molecules, i.e., compounds with a low molecular weight. These molecules are an important key in drug discovery since their reduced weight makes them more likely to be absorbed by the organism and orally bioavailable [26]. Hence, small compounds can reach the specific biological site of action more effectively.

Polar Surface Area

The polar surface area of molecules influences the passive transport of drugs into membranes. PSA is calculated based on the summation of the surface areas of a molecule's polar atoms. However, its determination is time-consuming and complex, requiring 3D structure and surface determination. Therefore, Ertl et al. (2000) developed a technique known as Topological Polar Surface Area (TPSA). TPSA is a faster method for determining polar surface area using the PSA tabulated values of the molecules' fragments [27].

Drug-Like Molecules and Lipinski's Rule of Five

In drug discovery, the pharmacokinetic properties of molecules play a crucial role in their selection as a potential drug and their therapeutic efficacy, as these properties relate to the drug's absorption, distribution, metabolism, and excretion. Drug-like molecules are compounds that satisfy criteria based on the characteristics of approved drugs [28].

Several approaches can be used to anticipate the drug-like properties of molecules [29]. The popular Rule of Five method, developed by Lipinski et al. (1997) [30], defines restrictions in physicochemical properties that may be associated with non-bioavailability. The Rule of Five states that compounds with more than one of the following characteristics are more likely to have low permeability and poor absorption: MW greater than 500Da, LogP higher than 5, more than 10 H-bond acceptors, i.e., the sum of oxygen and nitrogen atoms, and the number of H-bond donors greater than 5, i.e., the sum of O-H and N-H bonds [30]. There are numerous variants of Lipinski's Rule. Veber et al. (2002) [31] proposed that for a drug to be orally bioavailable, it should also possess a PSA not higher than 140 $\mathring{A}^2$, rotatable bonds less than or equal to 10, and the sum of H-donors and H-acceptors fewer than or equal to 12.

However, the Rule of Five has limitations, such as the fact that certain types of orally active drugs, such as antibiotics, do not comply with the regulated qualities since they probably have a structure that naturally enables them to cross the membrane via transporter substrates [30]. Furthermore, focusing solely on orally administered drugs limits the discovery of potentially effective pharmaceuticals.

Quantitative Estimate of Drug-Likeness (QED)

Bioavailability rules, such as Lipinski's Rule of Five, indicate only whether or not a molecule is likely to be drug-like but do not provide numerical values. The Quantitative Estimate of Druglikeness was developed to quantify drug-likeness based on physicochemical properties using continuous numbers between 0 (all undesirable properties) and 1 (all attractive characteristics). QED proves to be more realistic than binary approaches since it considers tolerable molecules, even if they have one undesirable property and other attributes near the ideal limits [32].

Synthetic Accessibility Score (SAS)

In *de novo* drug design, one of the factors to be considered is the ease of synthesizing compounds in the laboratory. The Synthetic Accessibility Score, developed by Ertl et al. (2009), [33] is one of the most widely used methods for assessing the value of synthetic accessibility. It is calculated based on the structure of the fragments that constitute the molecules and on penalty scores, which indicate the presence of complex elements. The SAS values range between 1 (easily synthesized) and 10 (extremely difficult to produce) [33].

2.3 Prediction of Physicochemical Properties and Biological Activity

Predictive models relate molecular characteristics to pharmacological or biological activity. They are based on the principle that identical structural properties (physicochemical and configurational) exhibit identical activities in Quantitative Structure-Activity Relationship models and the same property in Quantitative Structure-Property Relationship (QSPR) models [34].

These methods are appropriate for optimizing the molecules for a specific target so that when administered to a patient, they bind to the element with sufficient affinity to avoid side effects. A biological target is a chemical structure that affects the health of an organism directly or indirectly and, when combined with another chemical structure, such as a drug, induces a biological reaction. The most promising computational methods are those that employ Machine Learning and Deep Learning.

2.3.1 Machine Learning Approaches

Machine learning is composed of methods that enable computers to acquire knowledge from available data. [35, 36] It consists of several types of learning, including unsupervised, supervised, semi-supervised, and reinforcement. Unsupervised learning utilizes relationships between data to group them without supervision, i.e., without knowing the labels. Supervised learning methods predict the labels based on the learned relationships between the training features and the training outputs. Semi-supervised learning is a combination of the two types of learning described previously, as it includes both labeled and unlabeled data [36]. Reinforcement learning determines which actions are more beneficial to the environment through a trial-and-error search for reward maximization [37].

In predicting molecular properties, supervised learning is the principal technique. It is divided into two types of problems: classification, which forecasts a discrete value, such as whether a molecule is active or inactive in a target, and regression, which predicts a continuous numerical value of a molecular property [38].

Moreover, in this context, the most popular machine learning models are Random Forest (RF) [39], Support Vector Machine (SVM) [40–42], and K-Nearest Neighbors (KNN) [43,44]. RF [45] employs a set of decision trees, and the output is determined based on all individual predictions. KNN [46] returns the prediction based on the values of the K nearest neighbors according to a similarity criterion. SVM [47] selects the hyperplane with the maximum margins to separate the classes, and for non-linear separation problems, it employs kernel functions and soft-margin. The models focus on the totality of the provided descriptors for prediction, although some of it is redundant and may hinder performance. This requires the use of feature selection techniques to determine the most relevant characteristics [44].

Xiao et al. (2002) [43] investigated the utilization of the KNN algorithm in the context of QSAR by developing the variable selection K nearest neighbor quantitative structure-activity relationship, KNN QSAR. This method combines feature selection with the prediction of compounds' anticancer potential.

Svetnik et al. (2003) [39] used the random forest method in the domain of quantitative structure-activity relationships to predict binding affinity in classification and regression tasks. Besides the comparatively high predictive efficiency, the model is able to determine the relevance of the features used by attending to a significant decline in the performance of important descriptors when noise is introduced to them.

Zhou et al. (2006) [42] proposed the boosting Support Vector Regression (SVR) method, which predicts the toxicity and inhibitory activity of two chemical substances based on their structural properties. The training uses a series of SVRs, where the data used in each model are predominantly the elements incorrectly predicted by the previous SVR model. Therefore, the final output is the weighted average of individual predictions. The adoption of boosting suggests an increased generalization capability comparable to a single SVR.

2.3.2 Deep Learning Approaches

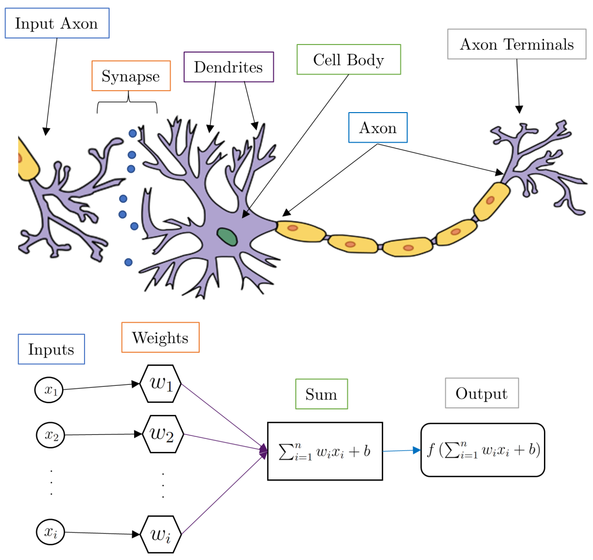


Figure 2.1: Comparison of the structure of a biological neuron with an artificial neuron. Image adapted from "`https://commons.wikimedia.org/wiki/File:Derived_Neuron_schema_with_no_labels.svg`"

Artificial Neural Network (ANN) is a branch of machine learning inspired by biological neural networks. It was created to perform tasks that were previously only possible for humans.

Biological neural networks are composed of massive interconnections of neurons, which are the basic unit of the brain and nervous system. The network's information exchange is triggered by signals and occurs through synapses, releasing neurotransmitters. This process transports information to different types of cells, promoting the execution of actions and learning ability [48].

ANN consists of a network of artificial neurons that simulate biological neurons and are organized into an input layer, one or more hidden layers, and an output layer. The input layer contains the elements supplied to the neural network without transformation, whereas the hidden layers receive the output of the input layer and transform based on the problem. In addition, the output layer employs an activation function to the weighted sum of the outputs of the final hidden layer [48, 49].

Each artificial neuron (Figure 2.1) contains an input, a bias, weights, an activation function, and an output. Input corresponds to the data transmitted to the neuron. Bias shifts the activation function to fit the data better. Weights adjust the relevance of the input and bias. The activation function transforms the weighted sum based on the context of the problem. The output of each neuron is the result obtained from the activation function and is given by:

$$f\left(\sum_{i=1}^{N} W_i X_i + b\right) \tag{2.1}$$

considering f the activation function, W the weight, X the input, and b the bias.

The ANNs acquire knowledge by updating the weights based on the prediction error between the true value and the predicted value, which is determined by a loss function. Using an optimizer that calculates the gradient of the loss function via backpropagation, the weights are modified [49]. This is an iterative process, known as the training phase. However, during the testing phase, where the architecture's generality is evaluated, the weights remain static. One of the advantages of ANN is the ability to select features without any previously employed technique, due to the different importance given by the architecture's weights [50].

Deep learning is composed of deeper neural networks, i.e., with multiple hidden layers. The number of hidden layers determines the degree of depth. DL is becoming increasingly promising, achieving state-of-the-art performance in a wide variety of areas, including Natural Language Processing (NLP) and object recognition, and drug discovery is no exception [50]. Furthermore, as data became more accessible, several QSAR and QSPR deep learning models outperformed popular machine-learning approaches [51], demonstrating the ability to discover more complex patterns and relationships between molecular features.

Dahl et al. (2014) [52] utilized a multi-tasking feedforward neural network method to simultaneously predict multiple biological targets' activity using the same set of molecular descriptors generated by a fingerprinting tool. They assumed that various pharmacological properties were interrelated. Duvenaud et al. (2015) [53] proposed an architecture based on graph convolutional neural networks for descriptors extraction, followed by a Fully Connected Neural Network (FCNN) for predicting several properties of compounds. Thus, the model produced fingerprints from the graph model that received as input the molecules in graph format.

Popova et al. (2018) [54] developed an LSTM-based neural network predictor for estimating several physicochemical and bioactivity properties.The model represented the molecules in SMILES format. Furthermore, its architecture demonstrated the ability to identify relationships between SMILES string elements without using numerical feature vectors.

Wang et al. (2019) proposed the SMILES-BERT model, an adaptation of Bidirectional Encoder Representations from Transformers (BERT) for predicting molecular properties from the SMILES representation [55]. Implemented by Devlin et al., BERT [56] is a framework based on the Transformer Encoder from the Transformer Model [57] and comprises two different stages: a) pre-training with Masked Learning Model and Next Sentence Prediction to create a model with a general understanding of the data language, and b) fine-tuning that leverages the pre-training parameters for the downstream task. Due to molecular individualism, the pre-training phase of SMILES-BERT does not consider the Next Sentence Prediction. Furthermore, the fine-tuning phase employs molecular characteristic forecasting. The obtained results demonstrate flexible pre-training with a large dataset of unlabeled data for pharmacologic property prediction.

2.4 Molecular Generation

Generative modeling has gained prominence in diverse technological fields with the evolution of DL, employing deep neural networks with increasing ability to solve more complex challenges. These models learn the distribution of training data in order to generate new elements with comparable distributions [58]. In drug design, generative models have revolutionized the generation of molecules with potentially desirable therapeutic properties in a more efficient and cost-effective form.

Recurrent Neural Networks (RNNs) were integrated into several DL models in the drug development field, inspired by the analogy of natural language processing problems with the generation of molecules in SMILES format. Seger et al. (2018) [59] developed a model for generating molecules active toward a target based on RNNs. Firstly, the architecture with three Long Short-Term Memory layers (LSTM) was pre-trained for a larger dataset to acquire a general knowledge of SMILES rules, then it was fine-tuned via transfer learning by retraining with a smaller dataset of active molecules for a specific target. Fine-tuning was also retrained cyclically using the best molecules selected by a predictor to optimize the model. Other studies, such as Gupta et al. (2018) [60], also applied transfer learning to an RNN architecture to produce molecules with the same properties as the training dataset for fine-tuning.

Popova et al. (2018) proposed an RNN model, the ReLeaSE (Reinforcement Learning for Structural Evolution) method, which includes a Stack-RNN generator for molecular SMILES sequence synthesis and an LSTM predictor for assessing its properties, such as binding affinity to a target and LogP. First, the two models are trained separately, then joined by reinforcement learning. Hence, the predictor evaluates the new molecules and gives a numerical reward based on their physicochemical properties, and the generator learns to maximize the rewards, becoming optimized [54]. Furthermore, Yasonik (2020) employed an evolutionary multi-objective algorithm on the RNN for drug design. By transferring learning, the model was retrained with the best molecules chosen by the non-dominated sorting algorithm based on the optimization of the characteristics present in an extension of the Rule of Five. He showed that his model could be effective both in optimizing the compounds' properties and in generating different molecules with the same distribution as the training dataset [61].

Gomez-Bombarelli et al. (2018) designed molecules based on the Variational Autoencoder architecture (VAE). The VAE integrated an RNN encoder to convert the SMILES notation into a continuous representation (the latent vector), a predictor to evaluate the properties of molecules in the latent space, and an RNN decoder to perform the reverse task, i.e., transforming continuous values into SMILES strings. The purpose of this study was to produce compounds with enhanced characteristics; hence, the training process focused on the optimization of generation based on the predictor's estimations of the molecules' properties in continuous space [62].

Generative Adversarial Autoencoder is a modified version of VAE that includes an additional block corresponding to the discriminator, whose function is to distinguish whether an element is a part of the latent space or a prior distribution. Polykovskiy et al. (2018) applied Adversarial Autoencoder (AAE) to produce molecules with particular properties [63]. Jin et al. (2018) proposed the Junction Tree Variational Autoencoder Model, JT-VAE, an application of VAE for developing molecular graphs. This model built junction trees, which correspond to subgraph valid structures, and combined them to produce molecules. Therefore, validity was maintained across the entire procedure [64].

Several studies in the design of novel chemical compounds have used Generative Adversarial Networks (GANs), an architecture that includes a generator to produce realistic and high-quality representations to fool the discriminator, which in turn has the function of distinguishing between real and generated representations. Objective-Reinforced Generative Adversarial Networks (ORGAN) [65] and Reinforced Adversarial Neural Computer (RANC) [66] are based on the combination of GAN architecture with reinforcement learning. The training proceeded to maximize rewards both for the discriminator's prediction ability and the generated molecules' drug-like qualities. ORGAN optimized QED, synthetic accessibility, and LogP properties, whereas RANC enhanced LogP, TPSA, and QED. Prykhodko et al. (2019) developed LatentGAN, a model that incorporates an autoencoder and a GAN to generate compounds with identical drug-like properties to the training dataset. The main innovation of this model compared to ORGAN and RANC is that its generator and discriminator used the autoencoder's latent vector as input instead of being trained directly with SMILES [67].

LigGPT, developed by Bagal et al. (2021), is a generative model based on the transformer-decoder that generates molecules based on conditions. Its input consists of SMILES strings concatenated with the criteria represented by a vector, which specifies the desired characteristics of the compounds, including synthetic accessibility score, LogP, TPSA, QED and chemical structure. During training, the datasets required a variety of attributes so that the model could learn to associate them with the conditions [68]. This model demonstrated the versatility and potential of the Transformers in drug design by yielding promising results.

3

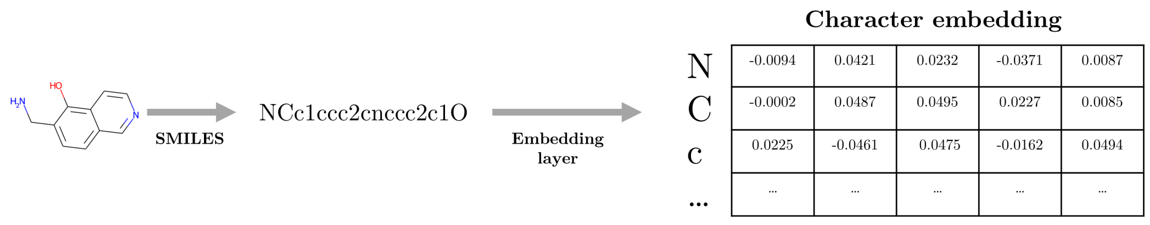
Models

3.1 Embedding Layer

The molecules are represented in the SMILES notation, however, in order for the deep learning models to understand raw characters, it is necessary to convert these characters into numerical values. On that account, each SMILES string was transformed into a set of non-negative unique integers, i.e., categorical variables without order or relationship (Section 4.2). Higher categorical values, however, can have more influence than the others in the learning stage, resulting in inaccurate performances and hindering the training process. Thus, encoding methods to normalize the importance of each categorical value need to be explored and employed. There are many types of encoding, e.g., one-hot encoding turns each variable into a normalized binary vector, where '1' is assigned to the corresponding integer and '0' to the rest of the elements in the vector. However, this type of approach is inefficient in the case of large sets of integers, thus, it was necessary to apply an embedding layer that has the same purpose, and additionally, the new vector provides more information and can have a fixed reduction length.

The embedding layer converts each categorical data into a floating vector of a specified dimension, mapping each element into an embedding geometric space, where the similarity between sequence elements is represented [69]. The learnable embedding matrix is randomly initialized during training and updated through backpropagation at each learning step.

Figure 3.1 depicts the output of the embedding layer applied to the molecule Edaravone (C10H10N2O).



N	-0.0094	0.0421	0.0232	-0.0371	0.0087
C	-0.0002	0.0487	0.0495	0.0227	0.0085
c	0.0225	-0.0461	0.0475	-0.0162	0.0494
...	...	...	...	...	...

Figure 3.1: A five-dimensional embedding layer applied to the molecule Edaravone.

3.2 Sinusoidal Positional Encoding

The localization of each element in the sentence is crucial for context and meaning, where a change in the order could result in a different interpretation. Considering that SMILES is a type of chemical language, it is crucial for the deep learning models to have information about the relative or absolute position of each element in the string.

Contrarily to the LSTM architecture, Transformer-based architectures do not contain hidden states that keep the information about the relative position of each element. Therefore, it is necessary to consider methods that provide this information, which can be static or learned. In this case, the sinusoidal positional encoding, which is a static strategy, was employed. The sinusoidal positional encoding proposed by Vaswani et al. [57] is based on sine and cosine functions of different frequencies, resulting in a unique encoding for each position of the sequence, i.e., the resulting values are fixed and unique, and the sequence length does not influence these values since it is based on relative positions.

The positional encoding for the kth token of the sequence and ith position of the embedding vector can be given by:

$$\begin{aligned} PE(k, 2i) &= \sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right) \\ PE(k, 2i+1) &= \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right) \end{aligned} \tag{3.1}$$

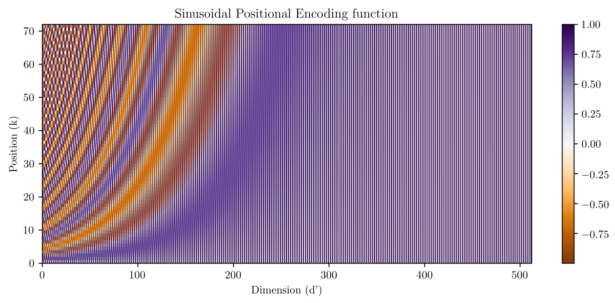
where $k \in [0,L-1]$ is the position of the token in the sentence (L is the length of the sentence), $i \in [0,\frac{d}{2}-1]$ is the position on the embedding vector, d is the dimension of the encoding input, 10000 is a pre-defined constant value that can be changed e.g. for longer sequences, and $PE \in \mathbb{R}^{Lxd}$.

This embedding is based on relative positions, which can be demonstrated by linearly transforming $PE(k,d')$ $(d' = \{2i, 2i+1\})$ with a rotation matrix $M \in \mathbb{R}^{2x2}$, resulting in the identification of any $PE(k+x,d')$. The mathematical justification that expression 3.2 is valid, i.e., that the position in the sentence (k) does not affect the matrix M (3.3) is provided in appendix A (A.2) [70].

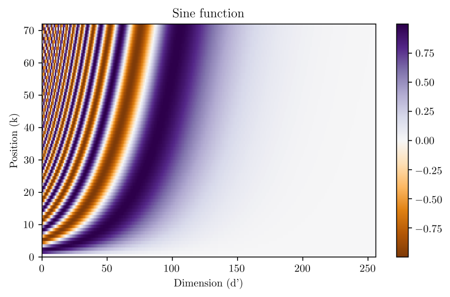
$$PE(k+x,d') = MPE(k,d') \tag{3.2}$$

with

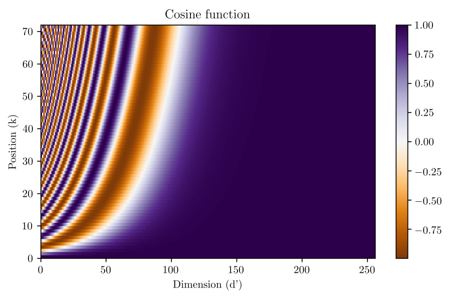
$$M(i) = \begin{bmatrix} \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) & \sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ -\sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) & \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \end{bmatrix} \tag{3.3}$$



(a)



(b)



(c)

Figure 3.2: Positional Encoding Heatmaps for a sentence with a length of 72 and an encoding dimension of 512. a) Sinusoidal positional encoding. b) Sine function. c) Cosine function.

Figure 3.2 shows a sinusoidal positional encoding matrix (given by 3.1), where it is possible to observe that increasing the dimension (d') leads to less variation of the heatmap, i.e., the positional encoding has more influence on smaller d'.

3.3 Transformer-Encoder

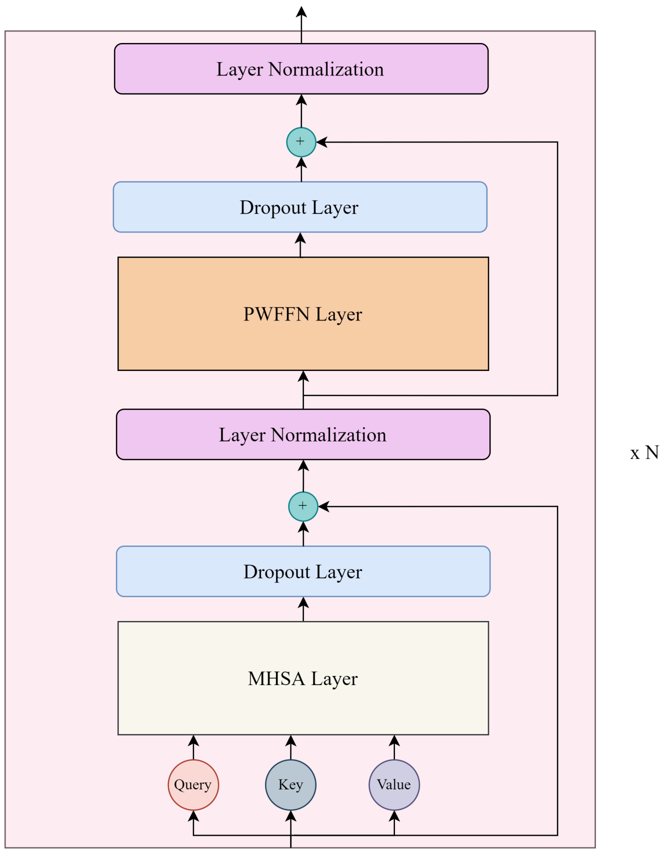


Figure 3.3: Transformer Encoder architecture, with N encoder blocks.

The Transformer-Encoder neural network transforms the input sequence into a representation vector that reflects the interdependence of its elements through self-attention. In this work, it was used to learn the chemical context of SMILES strings and their short and long-term context dependencies for the prediction of a drug's biological target affinity.

The Transformer-Encoder architecture is composed of a stack of N identical encoder blocks, where each block contains two different layers: Multi-Head Self-Attention (MHSA) and Position-Wise Feed-Forward Network (PWFFN). Furthermore, residual connections are applied after each block to preserve the information of the old token representation, followed by Layer Normalization (LN) to aid in the learning task of the model. Additionally, dropout layers were added before the residual connections of each MHSA and PWFFN, and after each dense layer of the PWFFN, to avoid model overfitting. Considering x^1 and x^2, the outputs of the MHSA layer and the PWFFN layer, respectively, the output of the ith encoder block can be expressed as:

$$\begin{aligned} x^1 &= LN(Dropout(MHSA(x^2_{i-1})) + x^2_{i-1}) \\ x^2 &= LN(Dropout(PWFFN(x^1_i)) + x^1_i) \end{aligned} \tag{3.4}$$

, where $i \in \{1,...,N\}$ (x_0 represents the input of the initial block obtained through embedding), $x^1 \in \mathbb{R}^{Lxd}$ and $x^2 \in \mathbb{R}^{Lxd}$ (L is the number of elements and d the dimension of x_0). The Transformer-Encoder architecture is illustrated in Figure 3.3.

3.3.1 Multi-Head Self-Attention

The MHSA applies self-attention multiple times in parallel to the input sequence in order to learn the inter-dependencies between the elements of the input sequence, resulting in a contextual and more robust representation of each token of the sequence.

This layer receives the input in the format of three parameters—Query (Q), Key (K), and Value (V) — all of which are derived from the same input sequence (self-attention). The query is used to search for self-related elements,the keys are the query-related elements that are used to calculate attention, and the values are a representation of the keys. The queries, keys, and values are linearly projected and split over several heads of attention. Multi-heads can learn different representations of attention with fewer encoders compared to single-heads, improving the learning process of the architecture and the stability of the training [71].

Each head employs scaled-dot product attention to transform a query and a set of key-value pairs into an output,which is expressed as the weighted sum of the values. The attention weights applied to each Value are computed by employing a softmax function to the $\sqrt{d_K}$ divided dot product between the queries and keys. Furthermore, a masking matrix is also considered to hide specific elements so that the model does not consider them. This masking matrix is obtained by assigning an extremely negative value (close to minus infinite) to the positions of the padding tokens since in the softmax $-\infty$ tends to 0. The MHSA output is the linear projection of the concatenation of all heads' outputs.

$$Attent(QW_i^Q, KW_i^K, VW_i^V) = Softmax\left(\frac{QW_i^Q(KW_i^K)^T}{\sqrt{d_K}} + Mask\right)VW_i^V \quad (3.5)$$

$$\begin{aligned} MHSA(Q,K,V) = Concat[&Attent(QW_1^Q, KW_1^K, VW_1^V), \ldots, \\ &Attent(QW_h^Q, KW_h^K, VW_h^V)]W^0 \end{aligned} \quad (3.6)$$

, where $Q,K,V \in \mathbb{R}^{Nxd}$. $W_i^Q \in \mathbb{R}^{dxd_Q}$, $W_i^K \in \mathbb{R}^{dxd_K}$ and $W_i^V \in \mathbb{R}^{dxd_V}$ are the projection matrices of the Query, Key and Value respectively. $W^0 \in \mathbb{R}^{hxd_vxd}$ is the final projection matrix, and h is the number of head attentions. $Mask \in \mathbb{R}^{NxN}$, and $d_K = d_Q = d_V = \frac{d}{h}$. The Multi-Head Self-Attention architecture is shown in Figure 3.4.

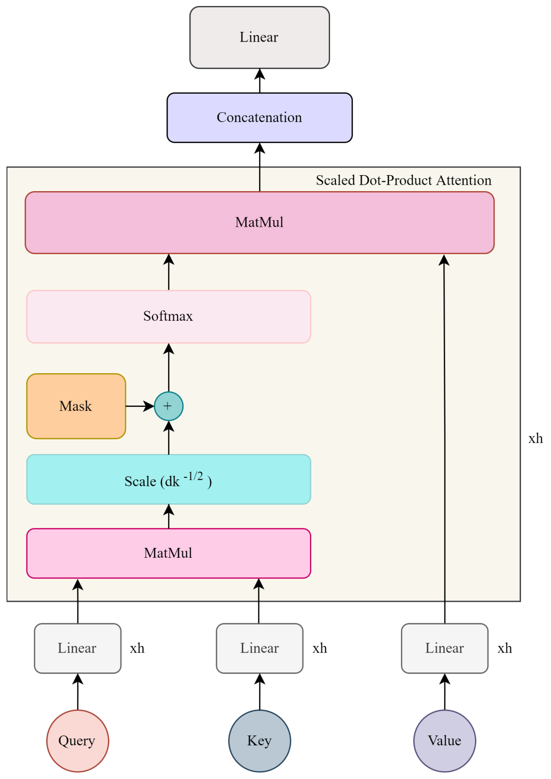


Figure 3.4: Multi-Head Self-Attention architecture, with h head attention.

3.3.2 Dropout Layer

Deep neural networks use the dropout method between their layers to prevent overfitting since neurons may overly adapt to each other during training, leading to increased noise. Dropout is a regularization technique that promotes the learning independence of each unit and increases the model generalization. During the training stage, this method deactivated a percentage (p) of randomly selected neurons and their connections, whereas, in testing, the weights are multiplied by the training forgetting rate (p). Figure 3.5 shows the dropout method applied to a FCNN.

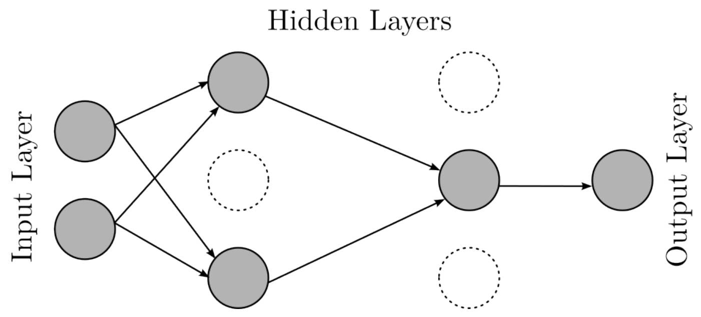


Figure 3.5: Dropout method applied to a FCNN.

3.3.3 Layer Normalization

Training deep neural networks is time-consuming due to the vast number of parameters and gradients updates. This can be improved using techniques such as layer normalization. LN [72] prevents an accentuated disparity in the values of the neurons in each layer's input by distributing the values to a mean of approximately 0 and a standard deviation of 1.

The mean (μ_i) and standard deviation (σ_i) of the outputs of neurons connected to layer i are given by :

$$\begin{aligned} \mu_i &= \frac{1}{H_i}\sum_{j=1}^{H} x_{ij} \\ \sigma_i &= \sqrt{\frac{1}{H_i}\sum_{j=1}^{H}(x_{ij}-\mu_i)^2} \end{aligned} \tag{3.7}$$

where x_{ij} is the inputs of neuron with $j \in \{1,..H_i\}$ of layer i, and H is the number of hidden units.

These two statistical measures are used to normalize x_{ij} according to expression 3.8, where σ_i^2 represents the variance and ϵ is a stability factor. This transformation is applied to every neuron in the layer and is independent of the batch size. In addition, this mechanism performs identically for training and testing.

$$x'_{ij} = \frac{x_{ij} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} \tag{3.8}$$

3.3.4 Position-Wise Fully Connected Feed-Forward Network

FCNNs have a simple architecture and are identical to the traditional neural networks, where all neurons are interlinked and each individual output is determined by applying an activation function to the weighted sum of the outputs of all previous neurons. The activation function contextualizes the data for a specific problem. The information in this architecture flows in one direction, from the input layers to the end of the networks, i.e., they do not employ feedback mechanisms [73]

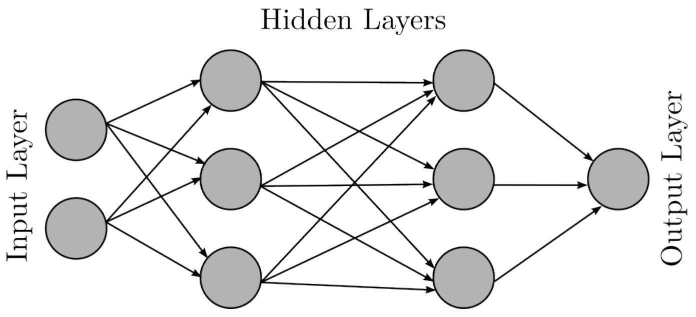


Figure 3.6: Fully Connected Feed-Forward Network Architecture.

The PWFFN aims to create a more robust representation of the output of MHSA layer by employing two dense layers separated by a dropout layer. These layers are applied to the last dimension of the MHSA output, position-wise, with the first dense layer projecting it to a higher dimension and the second returning it to the original last dimension. Besides, this neural network enhances the learning ability of the Transformer-Encoder.

Considering $x \in \mathbb{R}^{Nxd}$ the FCNN input, f the activation function of the first dense layer with dimension d_{ff}, $W_2 \in \mathbb{R}^{d_{ff}xd}$ and $b_2 \in \mathbb{R}^d$, PWFFN is expressed as:

$$PWFFN(x) = f(x)W_2 + b_2 \tag{3.9}$$

3.4 Transformer-Decoder

The Transformer-Decoder uses self-attention to generate new output sequences token by token. This architecture was initially proposed in the NLP domain in order to translate sentences. However, in the context of this study, it is used to generate valid and diverse molecules by learning the context and syntax of SMILES.

The Transformer-Decoder is composed of a stack of identical blocks, where each block contains an MHSA and a PWFFN, similar to the Transformer-Encoder. However, instead of the input sequence attending to all of its elements, it only takes into consideration the elements at the previous time step (similar to the LSTM architecture) using a look-ahead mask, i.e., this architecture is auto-regressive at each step. This look-ahead mask is employed to mask future tokens (right-sided) using an upper triangle matrix, where the diagonal values are not considered and the remaining elements are set to close to minus infinite. Therefore, the transformer decoder mask considers a look-ahead and a padding mask, which is illustrated in Figure 3.7. Figure 3.8 illustrates the Transformer-Decoder architecture.

$$\overset{\text{Padding Mask}}{\begin{array}{|c|c|c|c|}\hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & -\infty \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & -\infty & -\infty \\ \hline\end{array}} + \overset{\text{Look-ahead Mask}}{\begin{array}{|c|c|c|c|}\hline 0 & -\infty & -\infty & -\infty \\ \hline 0 & 0 & -\infty & -\infty \\ \hline 0 & 0 & 0 & -\infty \\ \hline 0 & 0 & 0 & 0 \\ \hline\end{array}} = \overset{\text{Decoder Mask}}{\begin{array}{|c|c|c|c|}\hline 0 & -\infty & -\infty & -\infty \\ \hline 0 & 0 & -\infty & -\infty \\ \hline 0 & 0 & 0 & -\infty \\ \hline 0 & 0 & -\infty & -\infty \\ \hline\end{array}}$$

Figure 3.7: Transformer-Decoder Masking, where a look-a-head mask is combined with a padding mask.

3.5 Multi-Objective Optimization

Multi-Objective Optimization Problem (MOP) is a field with numerous applications. MOP is responsible for identifying a set of variables that simultaneously optimizes two or more objective functions, i.e., determining which variables best achieve all the desired goals.

A solution integrates the Pareto optimal set if it is not worse than the others and is strictly better in at least one function [74]. Considering a minimization problem with vectors $\vec{a} = (a_1, a_2,..,a_q)^T$ and $\vec{b} = (b_1, b_2,..,b_q)^T \in \mathbb{R}^q$ and N objective functions

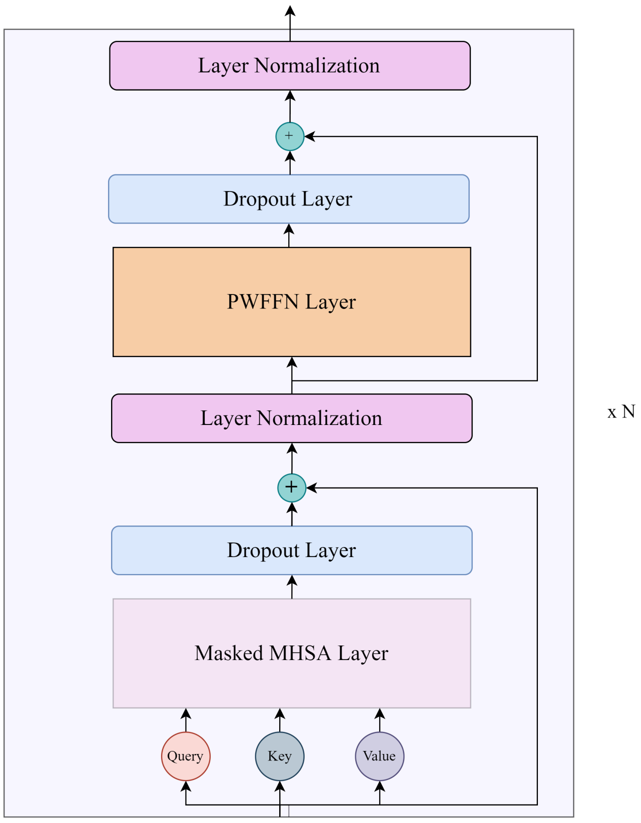


Figure 3.8: Transformer-Decoder architecture, with N decoder blocks.

$(f_1,...f_N)$:

$$f_i(\vec{a}) \leq f_i(\vec{b}), \forall i \in \{1,...N\} \wedge \exists i' \in \{1,...N\} : f_{i'}(\vec{a}) < f_{i'}(\vec{b}) \tag{3.10}$$

this means $\vec{a}$ dominants $\vec{b}$ ($\vec{a} \preceq \vec{b}$). Hence, $\vec{a}$ is Pareto optimal if no solution dominates it ($\vec{a}$ is non-dominated). Consequently, it does not improve the performance of an objective function without affecting the performance of at least one of the others. The set of all non-dominated solutions constitutes the Pareto optimal set (p^*), and the image of each ($f_i(p^*)$ with $i \in \{1,..N\}$) is on the Pareto front.

Non-Dominated Sorting is a type of multi-objective optimization that puts elements into different Pareto Fronts based on how dominant they are. This method was applied in our framework to optimize the transformer-decoder generator to produce SMILES molecules with the best physicochemical and biological properties. This was accomplished by sorting the new chemical compounds into Pareto Fronts based on their properties (the objective functions), followed by a feedback loop where the unsatisfactory generator input compounds were replaced iteratively with the preferable new ones.

The non-dominated sorting algorithm is implemented using external and internal loops. The external loop stores the solutions in different Pareto Fronts, whereas the internal loops determine whether an element is dominant in the space solution. If an element dominates at least one, the loop breaks and the program then looks for the next solution. Furthermore, when the internal cycle finishes, it adds the non-dominant solution to the current Pareto Front, which is no longer used to verify the dominance.

Considering a set of s molecules, from which two properties are extracted and added in the non-dominated sorting method. This yields K Pareto Fronts (F_1,...,F_K), where F_1 contains the dominated s_1 elements in s and F_2 contains the dominated molecules in space $s - s_1$, thus F_K are the non-dominated. This process is repeated until all the elements are distributed across K Pareto Fronts. Figure 3.9 shows the non-dominated sorting with K equal to three, and $f1$ and $f2$ being the properties of the molecules to be optimized.

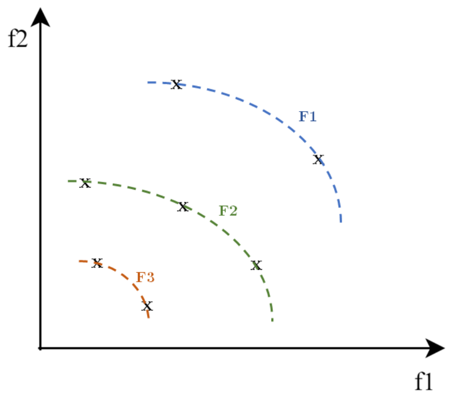


Figure 3.9: Ordering of solutions by dominance, considering two objective functions, $f1$ and $f2$

4

Methods

4.1 Framework Overview

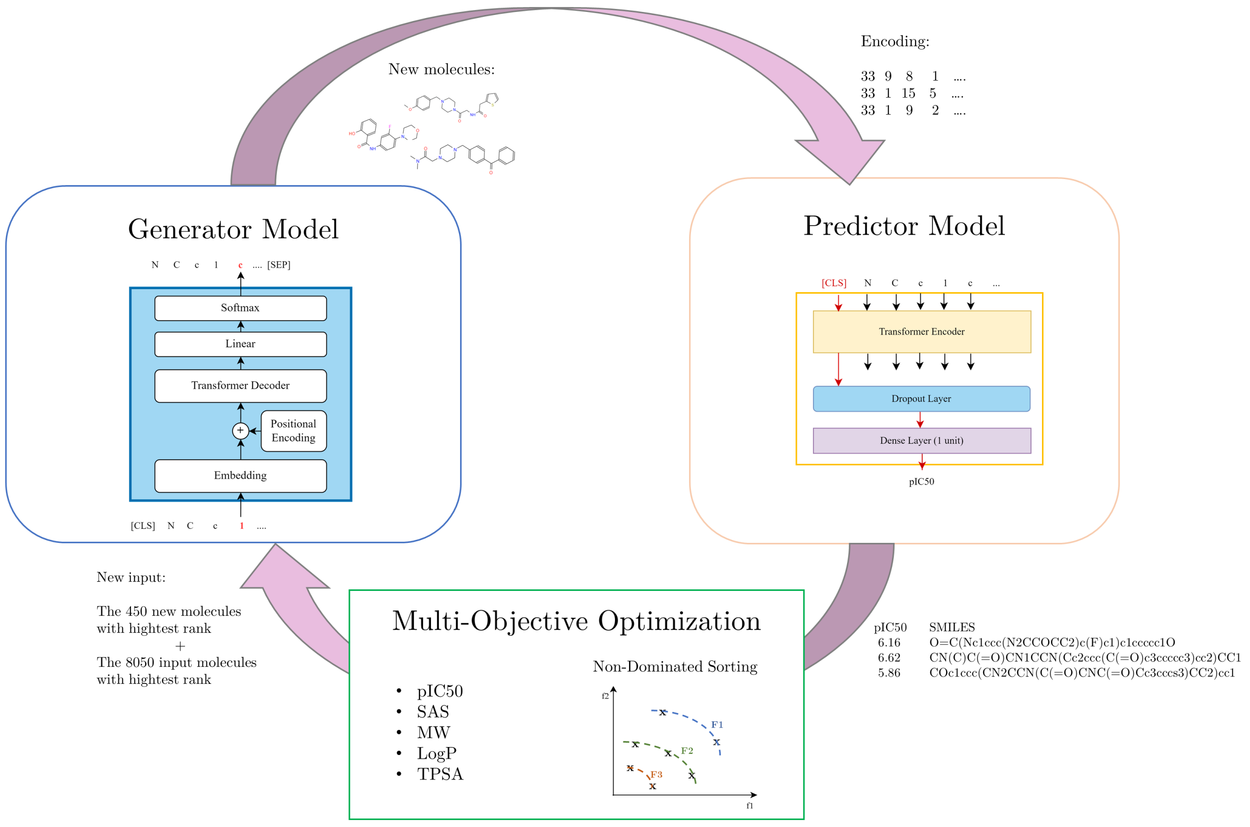


Figure 4.1: End-to-End Transformer-based Framework for Optimized *De Novo* Drug Design

The main objective of this research is the development of molecules with optimized biological and pharmaceutical properties.

The proposed framework contains a feedback loop combining a transformer decoder generator to create chemical compounds, a transformer encoder predictor to evaluate their biological affinity (pIC50) to a certain target with pharmacological interest, and a multi-objective optimization algorithm to sort the drugs based on their properties: biological affinity, SAS, MW, LogP, and TPSA. This end-to-end transformer-based framework is illustrated in Figure 4.1.

Initially, the models are trained from scratch separately, then incorporated into the feedback loop to optimize the generative model for the desired pharmaceutical standards. Furthermore, the training of the predictor is similar to the one used in the BERT architecture [56], comprising two phases: pre-training or MLM, where the model gains knowledge about the molecule representation and its context, and fine-tuning, which leverages the pre-training parameters for the downstream task.

4.1.1 Predictor

Pre-Training

In Masked Learning Modeling, the model acquires molecular context by learning to identify the elements of the input sentence that are masked based on the unaltered sentence units.

The architecture used in pre-train includes both an Embedding Layer and a Positional Encoding to provide a vector representation of the similarity and relative position of the elements. Furthermore, the Transformer-Encoder converts this vector to a new representation including the token's relevance, also its output is projected to a dimension equal to the size of the SMILES token dictionary via a dense linear layer. The Softmax function then transforms the dense output into a normalized probability distribution, indicating the likelihood of each token in the vocabulary having been replaced with the mask token. The pre-training stage is illustrated in Figure 4.2.

Masked Learning Modeling randomly selects 7.5% of the SMILES functional groups and another 7.5% of the remaining elements, excluding the [CLS], [STEP], and [PAD] tokens, to be masked, or 15% if the number of elements belonging to the molecule's functional groups is less than a threshold of 6. In those selected tokens, 80% are replaced with a [MASK] token, 10% are replaced with another randomly selected valid SMILES token and 10% are kept unchanged.

These percentage values are the defaults proposed by BERT [56] in order to not hinder model knowledge and optimize performance on subsequent tasks. Furthermore,

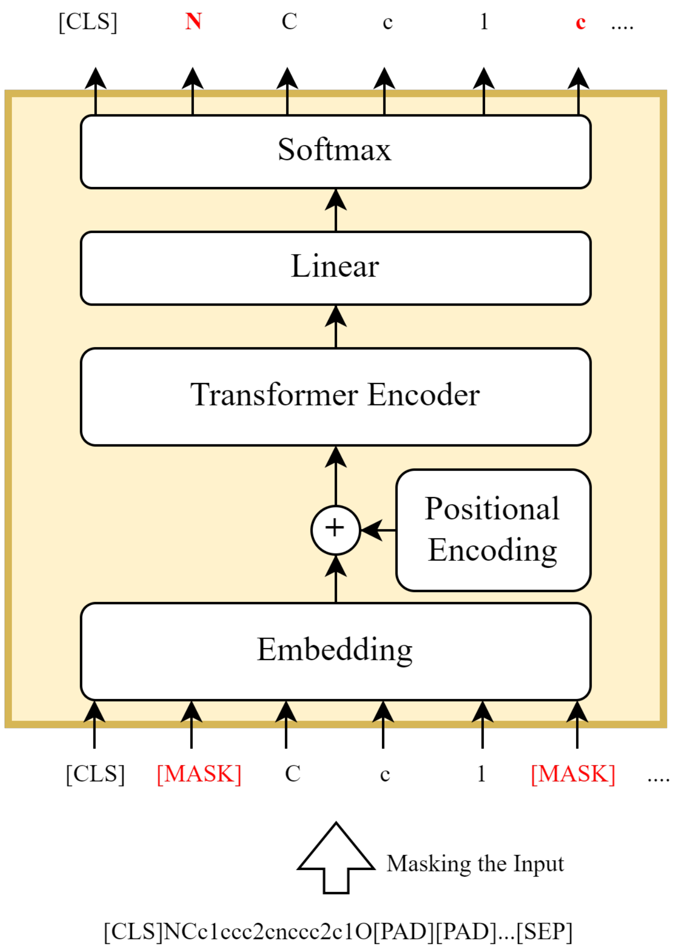


Figure 4.2: Masked-Learning Modeling based on Transformer-Encoder. Prediction of N and c tokens that were hidden by masking tokens.

the masking approach does not consider only the mask token since the [MASK] token does not appear in the fine-tuning, which could result in inconsistencies in the model during the fine-tuning. Moreover, the random selection of masked elements increases the model's capacity for learning and decreases the risk of overfitting.

In this study, functional group elements are incorporated into the traditional pre-training of the predictor for the potential addition of extra chemical information. FG is a group of atoms that give molecules similar reactivity and characteristics (e.g. polarity, solubility, structure) [75]. Initially, only the atoms that compose functional groups (15%) were selected for masking. Due to the lack of diversity of these atoms, the model was incapable of achieving generalized learning. Therefore, the strategy was also applied to the non-function group atoms (7.5%/7.5%). Furthermore, the molecules used for training contained diverse amounts of atoms belonging to functional groups. Therefore, to prevent compounds with fewer FGs constituents from having a lower percentage of masked components, a threshold of 6 FGs tokens was defined.

During training, the model learns to predict hidden tokens based on others that remain unchanged. Although this learning strategy requires a larger dataset in order to properly learn the context of the language, it only needs unlabeled data, i.e., without a target vector, which is highly available. During testing, the masks are applied identically to training, and the model's capability to predict the tokens is scored.

Fine-Tuning

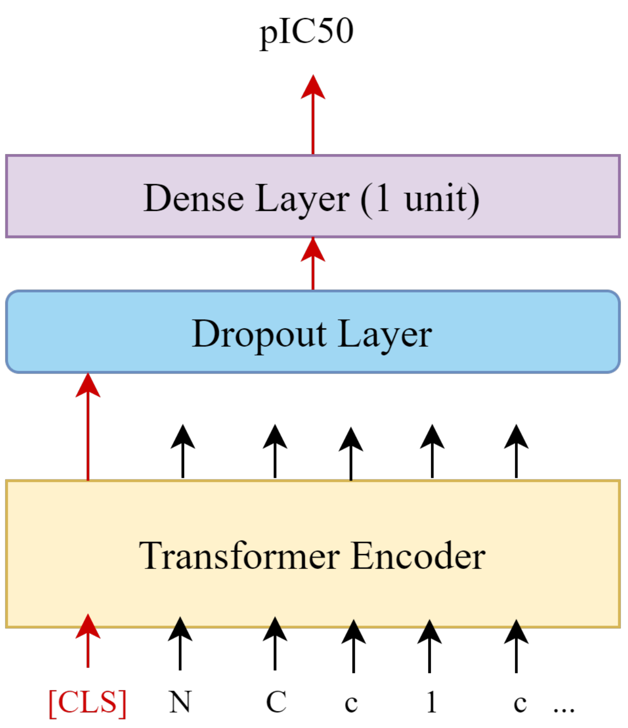


Figure 4.3: The Predictor Network to forecast the binding affinity of compounds that interact with a biological target.

In the fine-tuning phase, the predictor model's parameters are initialized with the ones learned during the MLM stage, and the model is retrained to predict the affinity of a particular biological target using a different labeled dataset. The pre-trained model already contains generalized chemical information that is not dependent on the potentially limited size of the labeled dataset. Hence, the training procedure is simplified and speeded up.

The fine-tuning incorporates the embedding and positional encoding and the transformer-encoder model of the MLM. However, it only considers the model output of the start token [CLS], also known as the classification token, since it is a robust representation that takes into account all the short and long-term dependencies amongst the tokens of the sentence. The average of all elements' output is not assumed because the end and padding elements are not informative to the classification task and are only used to facilitate model implementation. The architecture then integrates a dropout layer to reduce the risk of poor generalization, followed by a dense linear layer with one unit to provide the predicted number. All of the parameters that comprise the predictor model are updated during training. Figure 4.3 demonstrates the architecture of this classifier model.

4.1.2 Generator

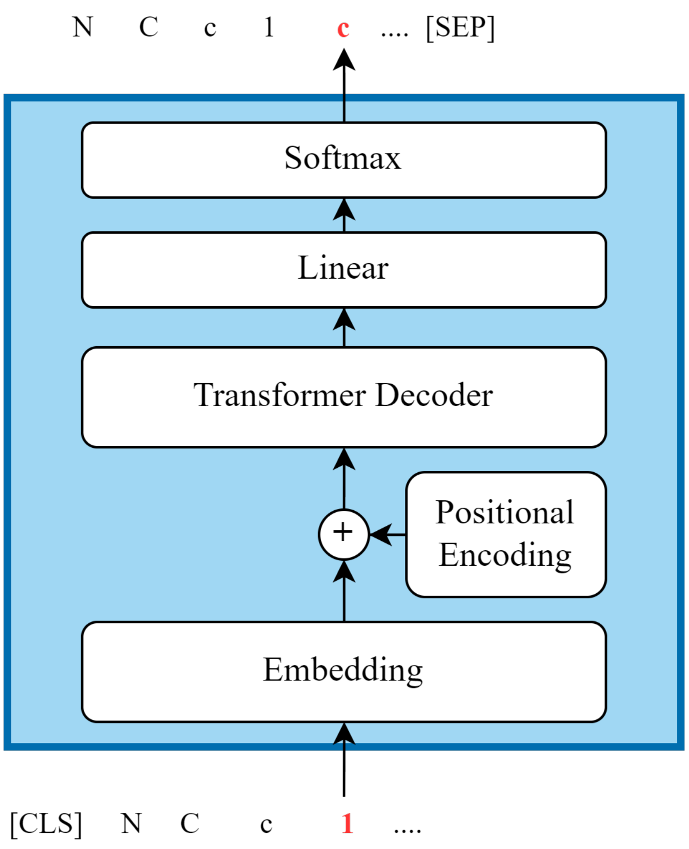


Figure 4.4: Generator architecture based on a transformer-decoder backbone.

The unconditional generator's purpose is to learn the chemical syntax required to generate neutral compounds, that is, compounds with no specific properties.

Its architecture combines an Embedding Layer with a Positional Encoding, analogous to that used in MLM (section 4.1.1). Then, a Transformer-Decoder is incorporated, which outputs a self-attention representation containing the relevance and dependencies of the elements to predict the next sequence entity. The output of the Transformer-Decoder is sent to a dense linear layer and a Softmax activation function, which projects it to a dimension equal to the size of the SMILES vocabulary and turns it into token probabilities, respectively. The generator neural network is shown in Figure 4.4.

The model is auto-regressive, i.e., for the generation of molecules, each token is predicted based on previous predictions [76]. Therefore, the model's input sequence is shifted to the left, and the output sequence is shifted to the right, i.e., the input sequence begins with the start token ([CLS]) and ends with the last molecule's token, whereas the output sequence starts with the first molecule's token and terminates with the end token ([SEP]). For example, for the Edaravone molecule, the model input is [CLS]NCc1ccc2cnccc2c1O, and the output NCc1ccc2cnccc2c1O[SEP].

During the training, the Transformer-Decoder applies the *Teacher-Forcing* method, in which the generation process is based on the previous token of the original sequence and is not influenced by the model prediction performance [77]. Figure 4.5 depicts this methodology. The input data is presented all at once on the Transformer-Decoder, however, due to the look-ahead mask in the MHSA (subsection 3.3.1), it is possible to employ *Teacher-Forcing*.

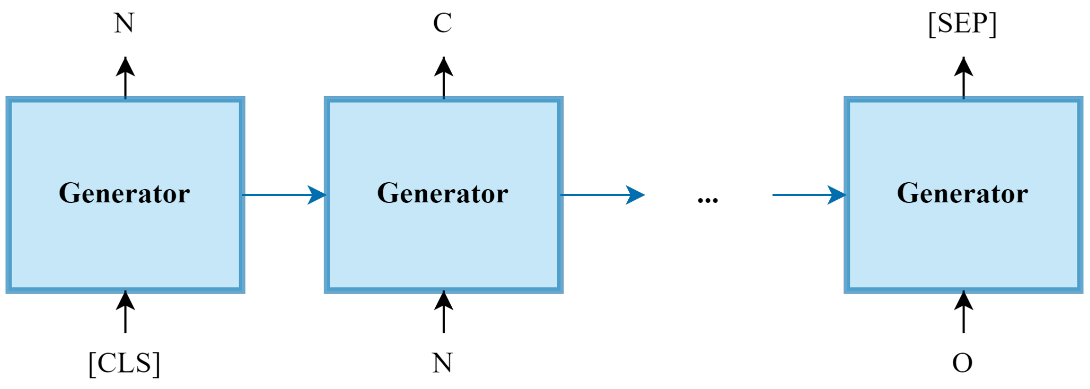


Figure 4.5: The training process for the SMILES molecule *NCc1ccc2cnccc2c1O* using the *Teacher-Forcing* method.

During the generation process, the model synthesizes novel compounds token by

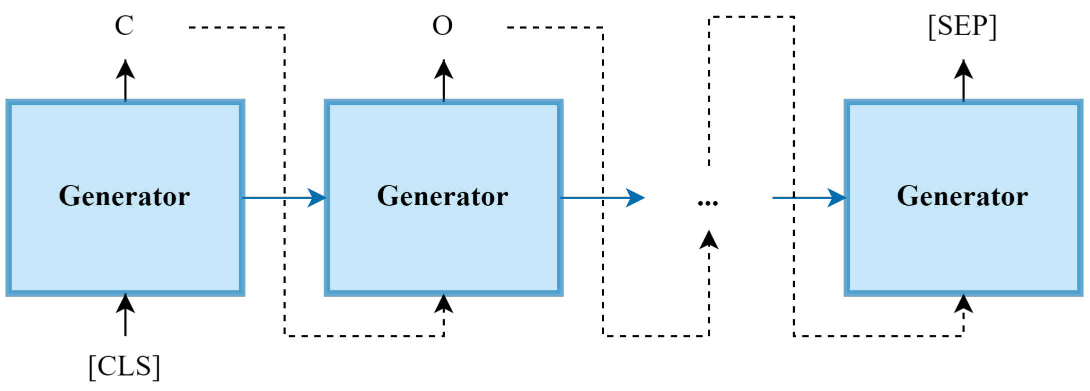


Figure 4.6: The generation of SMILES strings token by token using a trained model.

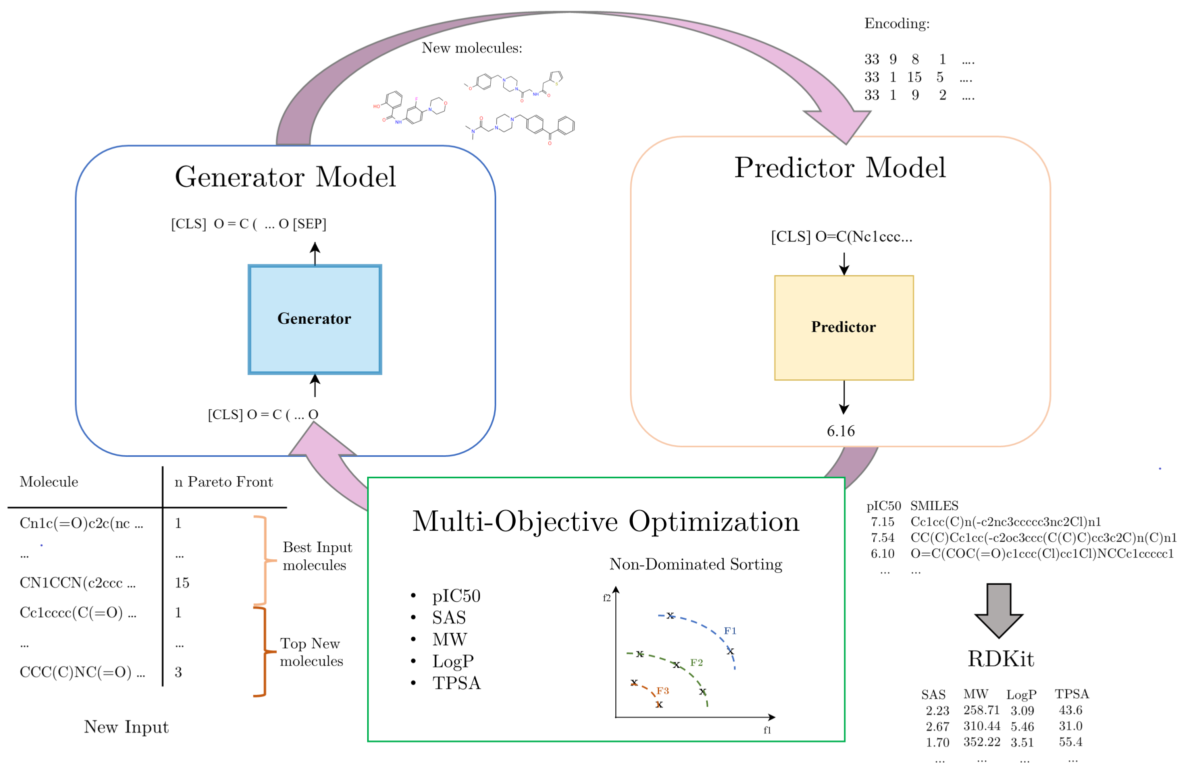
token, with the predicted one being included in the next input. Furthermore, the softmax function has a distinction: the inclusion of the sample temperature parameter (T), which provides a random factor to produce diverse drugs. Employing low-temperature parameters results in more confident predictions, on the other hand, high-temperature parameters cause more variable predictions [78], resulting in a greater diversity of produced compounds.

$$Softmax(x) = \frac{e^{\frac{x_i}{T}}}{\sum_{j=1}^{n} e^{\frac{x_j}{T}}} \tag{4.1}$$

4.1.3 Optimized Generator

The optimized generator is the Transformer-Decoder-based generator enhanced to design drugs with the desired characteristics, specifically biological activity and physicochemical properties that allow the drug to reach its target.

The framework training is composed of a feedback loop that includes a generator, a predictor, and a multi-objective optimization algorithm. First, the Transformer-Decoder generator is retrained to learn the syntax of the input molecules. Subsequently, the generator designs molecules, and the Transformer-Encoder predictor determines their affinity for the relevant pharmacological target. Furthermore, the RDKit library [18] is employed to determine the characteristics of compounds to be optimized: SAS, LogP, MW, and TPSA. The reason for their choice is given in section 4.3.3. Then, the multi-objective optimization algorithm receives the drugs and the values of their respective five properties (the objective functions to optimize).



Molecule	n Pareto Front
Cn1c(=O)c2c(nc ...	1
...	...
CN1CCN(c2ccc ...	15
Cc1cccc(C(=O) ...	1
...	...
CCC(C)NC(=O) ...	3

- pIC50
- SAS
- MW
- LogP
- TPSA

pIC50	SMILES
7.15	Cc1cc(C)n(-c2nc3ccccc3nc2Cl)n1
7.54	CC(C)Cc1cc(-c2oc3ccc(C(C)C)cc3c2C)n(C)n1
6.10	O=C(COC(=O)c1ccc(Cl)cc1Cl)NCCc1ccccc1
...	...

SAS	MW	LogP	TPSA
2.23	258.71	3.09	43.6
2.67	310.44	5.46	31.0
1.70	352.22	3.51	55.4
...	...	...	...

Figure 4.7: The multi-objective optimization framework for the generator. Optimization of SMILES generation based on several characteristics using a feedback loop that uses the best newly generated molecules as input.

The molecules are sorted into Pareto Fronts such that the preferable ones have the lowest values of SAS, LogP, MW, and TPSA, and the highest pIC50 (to inhibit the target). The best molecules, i.e., the ones on the top Pareto Fronts of the optimization algorithm, are added to the input replacing previous input compounds with less desirable properties. This method has a limited number of input substitutions per iteration to bias the model.

In the process of generating molecules, the Softmax function with a temperature parameter was used, analogous to Section 4.1.2. Hence, the generation was not based solely on the softmax's maximum probability, since it would result in repetitive compounds.

During the testing phase, only the Transformer-Decoder generator is used, without the feedback loop or other components, since the framework is a methodology used to improve the generator's parameters.

This strategy considered for optimizing the generator has the benefit of not requiring a specific dataset with the desired property distributions of the molecule, which could be extremely scarce. Moreover, it could optimize other characteristics in a relatively efficient training period.

4.2 Data Preparation

4.2.1 Data Preprocessing

The molecules used as input for the models are all represented in the SMILES notation as sequences of characters [17]. After removing duplicate strings in the dataset, each element of the sequence was used as a feature. Tokenization was necessary to convert the chemical structure into a set of word pieces, known as tokens, based on dictionary tokens containing the existing atoms, bonds, branches, and rings.

Due to the fact that each SMILES string is unique for a given molecule, there are molecules with varying lengths and, consequently, a different number of tokens. A threshold was added to standardize the number of features to be considered. The cutoff was established based on the percentage of molecules with fewer or equal tokens than the limit (not their length size because an atom can be represented with more than one character). While the larger structures are discarded, padding tokens [PAD] are added to the smaller ones until they reach the threshold. This filtration is necessary because a discrepancy in the length of the token set could generate noise due to excessive padding.

In addition, two special tokens, [CLS] and [SEP], have been added to the beginning and end of the sequence, respectively. The [CLS] is used in the classification task to represent the context and interdependencies of the input, and it indicates the start of production in the generation task. Furthermore, the [SEP] denotes the termination of molecule synthesis, i.e., the generation process ends when the model predicts this token.

4.2.2 SMILES Encoding

Due to the fact the models use SMILES strings as input, it was vital to convert each character into a numerical value capable of being used by the models.

Each unique char token was converted into an integer, as an index, using SMILES

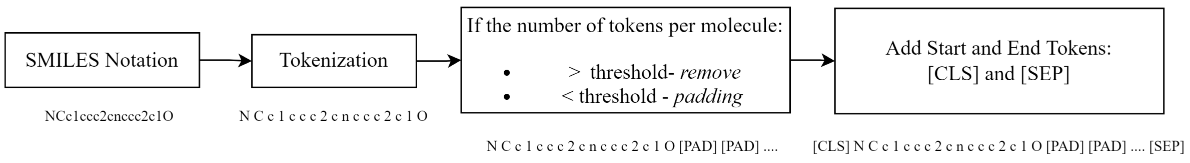

SMILES Notation
NCc1ccc2cnccc2c1O
Tokenization
N C c 1 c c c 2 c n c c c 2 c 1 O
If the number of tokens per molecule:
> threshold- *remove*
< threshold - *padding*
N C c 1 c c c 2 c n c c c 2 c 1 O [PAD] [PAD] ....
Add Start and End Tokens: [CLS] and [SEP]
[CLS] N C c 1 c c c 2 c n c c c 2 c 1 O [PAD] [PAD] .... [SEP]

Figure 4.8: The processing applied to drugs in SMILES strings based on tokenization, filtration by the number of tokens per sequence, and the addition of special tokens.

encoding, Figure 4.9. This is accomplished through the use of a SMILES char-integer dictionary (Table 4.1), which also includes the key-value pairs [CLS], [MASK], [PAD], and [SEP]. This encoding method is straightforward and preserves the structure and order of the chemical compounds.

Table 4.1: Char-Integer dictionary

Char Token	**Integer**
C	1
c	2
...	...
[MASK]	34

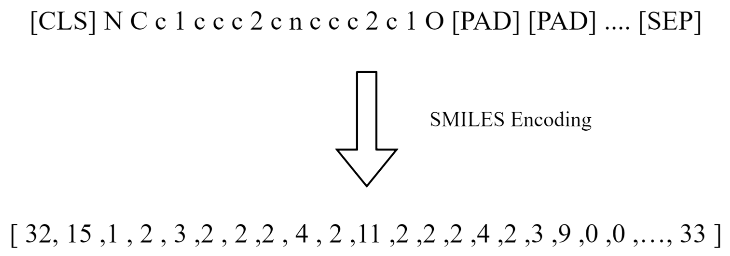


Figure 4.9: SMILES Encoding, to transform tokens into categorical values.

4.2.3 Additional Predictor data processing

Masked Learning Modeling

MLM was trained using molecules preprocessed with the methodology defined in the previous subsections and masked with the strategy described in 4.1.1, which covers up a percentage of functional groups and remaining tokens. Masking is achieved by replacing the SMILES token with the [MASK] char.

To determine the tokens that compose the FGs of each molecule (Figure 4.10), the function implemented by Hall et al. [79], which is based on the algorithm proposed by Ertl [80], was adapted.

This method identifies the FGs of a molecule based on the localization of heteroatoms and their environments, discarding other structural characteristics. Heteroatoms are non-carbon or hydrogen atoms that compose organic molecules. They affect the reactivity of the chemical compounds they are present in. Oxygen (O), nitrogen (N), sulfur (S), and phosphorus (P) are examples of elements [81]. The Ertl algorithm [80] is processed by marking the heteroatoms and halogens of the molecules as well as the following types of carbon atoms: those connected by non-aromatic double and triple bonds to heteroatoms or other carbon; those connected by single bonds to more than one oxygen, nitrogen, or sulfur (acetal carbons); and those integrated into the rings of aziridine, oxirane, and thiirane. Furthermore, all neighboring marked atoms are considered a single FG. In addition, the unmarked carbon connected to the FG is considered its environment. Therefore, it is a simple and automatic method, and the outcomes are compatible with the software tool Checkmol [82]. Additionally, it can be applied to determine complex FGs.

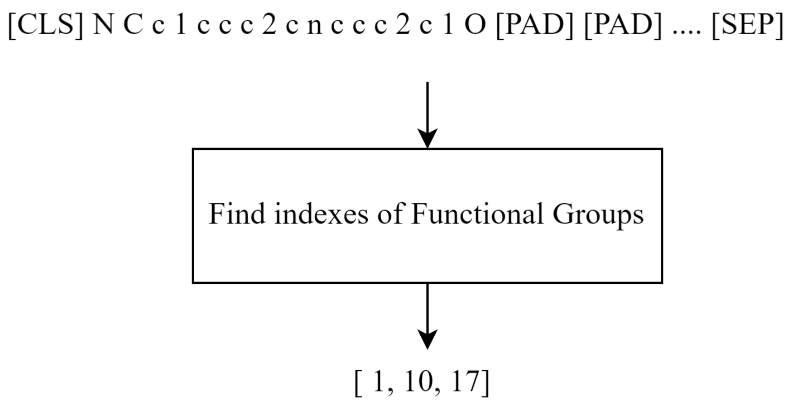

[CLS] N C c 1 c c c 2 c n c c c 2 c 1 O [PAD] [PAD] .... [SEP]
Find indexes of Functional Groups
[1, 10, 17]

Figure 4.10: Identification of the atoms that compose the functional groups of the chemical compound *NCc1ccc2cnccc2c1O*.

Fine-Tuning

Before training, the labels are normalized to compress the data so that the value disparity does not affect the model's performance. Hence, the model predicts a normalized biological affinity, taking into account the maximum and minimum values of the training set. Therefore, the testing's final output is the denormalization of the predictions.

Two types of normalization applied to the biological affinity data were employed: the Min-Max Scaler (4.2) and the Robust Scaler (4.3). The Robust Scaler is more robust to outliers than the Min-Max Scaler since it normalizes based on the quartile range, the minimum and maximum values (x_{min} and x_{max}, respectively) are replaced by the first quartile ($Q_1(x)$) and third quartile ($Q_3(x)$) of vector x

$$\frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{4.2}$$

$$\frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)} \tag{4.3}$$

4.3 Hyperparameter Optimization

4.3.1 Predictor hyperparameters

Masked Learning Modeling

For the model to achieve its best performance, the hyperparameters that affect the learning process should be configured as optimally as possible. Hyperparameter optimization is the process of determining the optimal set of parameters. Grid search with cross-validation is one of the most popular techniques, where the model is trained considering a grid of hyperparameters and the data is divided into training for the model's learning, validation to evaluate the performance of the selected parameters, and testing to evaluate the model's generalization.

In this stage, grid search with holdout cross-validation was applied instead of k-fold cross-validation since the MLM uses a large dataset and the model has a wide range of hyperparameters: *the dimension of the embedding vector, the PWFFN dimension projection, the number of heads, the number of encoder blocks, the optimizer* and *the learning rate*. Hence, the data is divided statically into training to train the model and testing to analyze the performance with new data, as opposed to running the training k times with different splits,which would require considerable resources.

The type of data partitioning used is adequate because the sets are independent, devoid of duplicated data, and the selection of tokens to be masked for prediction is random.

The *optimizer* is the algorithm used to update the training weights at a defined learning rate. The Transformer-encoder is a memory-intensive architecture, so the search space for the *number of heads* and *encoder blocks* is restricted by the capabilities of the computer used. The architecture requires extensive computational resources, however, it only needed 10 epochs in the grid search to converge to a local minimum.

The grid search with cross-validation was used in conjunction with early stopping and model checkpoint to find the optimal model. Early stopping terminates model training if the performance does not improve over a predetermined number of iterations, and the model checkpoint saves the model at the epoch with improved performance.

Fine-Tuning

The hyperparameter optimization approach in the Fine-Tuning stage was a grid search with 5-fold cross-validation. Although more computationally expensive, it is more suitable for small data sets, as it prevents instability between the split sets. Since the fine-tuning uses pre-trained weights, the training is faster and it is possible to apply this type of cross-validation. The dataset is randomly divided into training and testing sets. The training data is split into five subsets - the folds. Therefore, for each combination of parameters in the grid search, the model is trained five times, each time with four folds, and validated on the remaining one. The folds in training and validation are rotational.

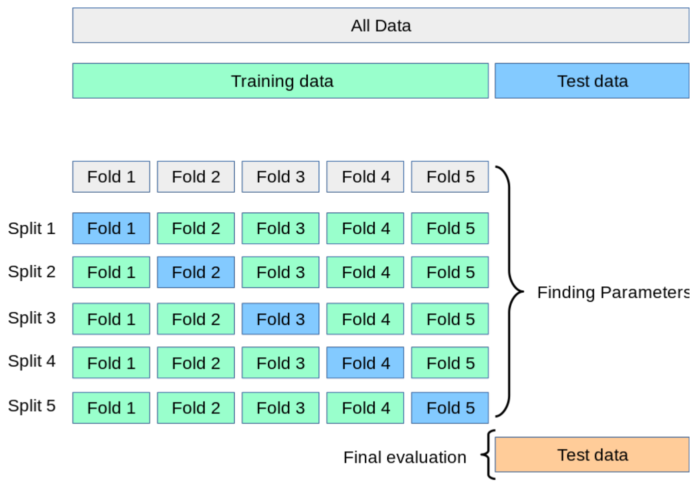


Figure 4.11: 5-fold cross-validation. Image from "`https://scikit-learn.org/stable/modules/cross_validation.html`".

The hyperparameters to be optimized include: *the dropout rate of the layer preceding the dense linear with 1 unit, the dimension and function of additional hidden layers, the batch size, number of epochs, optimizer function*, and *learning rate.* Moreover, the *number of epochs* corresponds to the number of iterations that the entire dataset was used to train the model, and within each epoch, the dataset was divided into multiple *batches* to reduce memory requirements.

Furthermore, similar to MLM, early stopping and the model checkpoint were used to determine the best model.

4.3.2 Generator Hyperparameters

Grid search was used with a large training set to determine the most suitable parameters for the generator. Since the objective of the model is to generate valid and diverse drugs, a testing set was unnecessary. Early stopping and model checkpoint were considered in the training stage. Furthermore, to select the best model, 500 molecules were created and scored using metrics that assess their validity, diversity, and novelty.

Due to the vast number of combinations in the search space, the temperature parameter of the softmax function in the generation task was manually tested with the same hyperparameters.

The hyperparameters of the transformer-decoder based generator to be optimized are *the dimension of the embedding vector, the PWFFN dimension projection, the number of heads, the number of decoder blocks, the optimizer*, and *the learning rate.* For the same reason stated in MLM, the number of grid search training epochs was ten.

4.3.3 Optimized Generator Hyperparameters

The best-optimized model is determined using both a grid search and manual methods. Since this is a generation task, only training data was used, as was the case with the unbiased generator. Early stopping and model checkpoints assist model training.

The *optimizer, learning rate, batch size*, and *number of epochs* were the hyperparameters optimized via grid search.

To optimize the compounds in the feedback loop, the following groups of chemical properties were explored manually: pIC50; pIC50 with SAS; pIC50 with SAS and LogP or MW or TPSA; and all five metrics (pIC50, SAS, LogP, MW, and TPSA). All property combinations considered the molecular binding affinity to a biological target, as represented by pIC50 since one of the primary goals of the work was to generate active compounds for a biomolecule. The SAS property is also considered in the majority of sets, as it is essential that novel molecules can be synthesized easily in the laboratory. The remaining selected properties, LogP, MW, and TPSA, are employed most frequently in computational models for drug design.

Also, the threshold of input molecules to be replaced in each epoch was manually adjusted. The evaluation was carried out by generating 500 molecules and analyzing their characteristics.

5

Experimental Setup

5.1 Datasets

The datasets used in the Unbiased Generator and all stages of the Predictor were collected from the ChEMBL database, which is an open-access and manually curated drug discovery database with a vast amount of compounds with drug-like characteristics (including their bioactivities for certain targets) [83].

The SMILES strings were fixed to the maximum length of 72 characters, which corresponds to almost 95% of the information present in the dataset. Molecules that included more than 72 elements were removed, those with less were padded to fit the criteria, and those that contained exactly the threshold value of elements were maintained.

In both the unbiased generator and the masked learning model, the dataset consists solely of preprocessed SMILES strings. Nonetheless, the extracted dataset for fine-tuning contains compounds and their biological affinities (pIC50) for the Adenosine A2A Receptor (AA2AR) with the ChEMBL ID of CHEMBL251.

The unbiased generator was also trained with the MOSES dataset [84] in order to compare its efficiency with state-of-the-art models. MOSES is a platform used as a benchmark for generative models. It includes training and test datasets, metrics, and implemented models to standardize the frameworks' outcomes, facilitating their comparison. The MOSES data is derived from ZINC Clean Leads [85], where charged atoms, atoms that are not H, C, N, S, O, F, Cl, or Br, and larger cycles were excluded. Moreover, all the molecules possess drug-like properties.

The training input of the optimized generator was comprised of valid and unique molecules produced by the unbiased generator,which were not optimized for any specific and relevant pharmacological property. The input sizes used were 8500 and

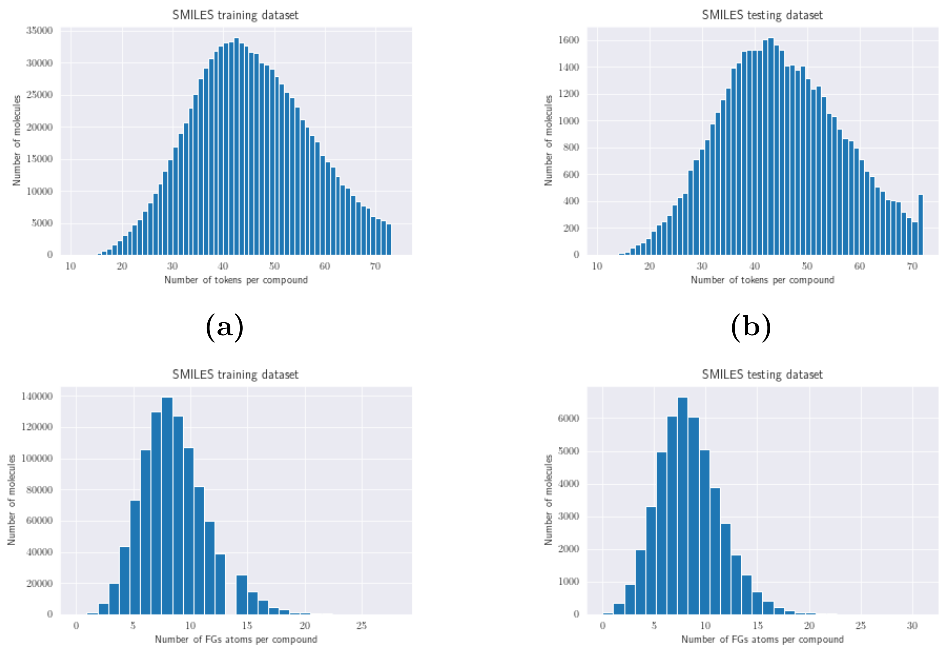
20000. Selecting which is preferable is a matter of trade-off based on metrics and attributes that are more essential to the problem.

Predictor - Masked Learning Modeling

Figure 5.1 illustrates the distributions of the training and testing sets in terms of the number of tokens and functional groups present in the SMILES strings. The distributions of the two sets are identical, and there is no duplication of molecules between them. Table 5.1 provides a summary of the number of molecules comprising each set.

Table 5.1: Distribution of data in training and testing in the predictor MLM stage.

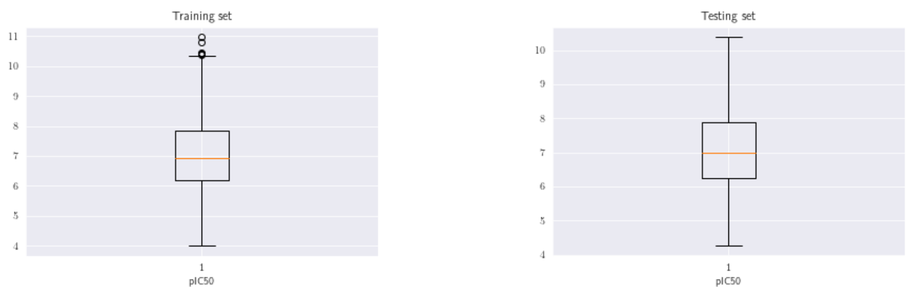
	Number of SMILES
Training	1000000
Testing	46964



(a) (b)

(c) (d)

Figure 5.1: Distribution of the number of tokens and functional groups present in the SMILES strings associated with the ChEMBL dataset [83]. (a) SMILES training dataset. (b) SMILES testing dataset. (c) SMILES FGs for training. (d) SMILES FGs for testing.



(a) (b)

Figure 5.2: Distribution of interquartile pIC50 values in the training and testing. (a) Training set. (b) Testing set.

Predictor - Fine-Tuning

For the fine-tuning of the predictor, the dataset collected from the ChEMBL included 4562 molecules (Table 5.2), and some SMILES were not present in the pre-training dataset. However, the pre-trained model had generalized learning, allowing it to use the context for the prediction task. Figure 5.2 depicts the distribution of non-normalized pIC50 values for Adenosine A2A Receptor in the training and test sets, with an equivalent interquartile range. Therefore, the best-trained model should theoretically also fit the test data.

Table 5.2: Distribution of data in training and testing in the predictor fine-tuning.

	Number of SMILES
Training	3877
Testing	685

The design of drugs capable of inhibiting the AA2AR is one of the primary goals of this research. This biological target is widely studied in drug discovery due to its potential therapeutic effects.

The Adenosine A2A Receptor is part of the Adenosine Receptors, which are G-protein coupled receptors. GPCRs are a family of membrane proteins responsible for transmitting extracellular signals into the cellular interior, resulting in physiological responses [86]. The development of AA2AR antagonist drugs focuses mainly on treating neurodegenerative diseases such as Parkinson's, Alzheimer's, and depression. Furthermore, AA2AR inhibition can also potentially be considered in cancer immunotherapy [87]. On the contrary, AA2AR agonist molecules are commonly used

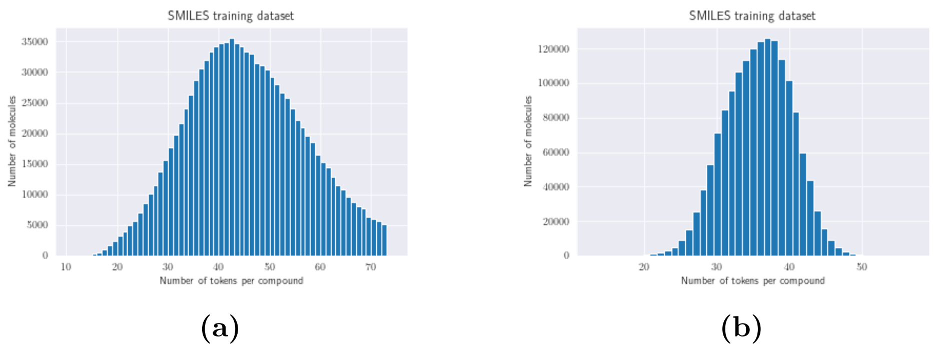


Figure 5.3: Distribution of tokens per sequence in the training with ChEMBL and MOSES datasets. (a) ChEMBL dataset. (b) MOSES dataset.

to treat inflammatory diseases such as asthma and to promote wound healing [87].

Generator

The generator was trained using the datasets from ChEMBL and MOSES. The first dataset contains chemical compounds without specified features, whereas the second dataset is more restricted since it has been considered in several state-of-the-art models to replicate the input distribution. Table 5.3 summarizes the number of drugs contained in each dataset.

Table 5.3: Datasets for unbiased generator training.

Dataset	Number of SMILES
ChEMBL	1046964
MOSES	1584663

5.2 Predictors

5.2.1 Transformer-Encoder Predictor

The neural network predictor is used to identify the affinity to inhibit the adenosine A2A receptor, moreover, its development is based on two stages: MLM and fine-tuning. As described in Section 4.3, their strategies for selecting the most suitable models with optimal parameters differ.

Grid search is employed in MLM to optimize the model based on a range of predefined values. The dropout rate of the transformer-encoder, the number of epochs, and the batch size of training, on the other hand, had fixed values. Similar to the BERT [56] and SMILES-BERT models [55], which were the referenced works in this stage, the dropout rate was 0.1. Even though grid search has a reduced number of epochs, the training of the best model has been extended to 50 epochs in order to improve performance. This restriction on iterations is coupled with early stopping for computational efficiency. Furthermore, the batch size was 256 because 32 and 64 values were insufficient for convergence due to the model's large input; 128 required more time to train, and computer memory was not available for larger batch sizes.

In fine-tuning, a grid search with 5-fold cross-validation was used to determine the optimal hyperparameter values. The insertion of hidden layers in the output of the transformer-encoder was also analyzed. With the information learned during the pre-training, adding more layers did not significantly improve the predictor's performance. Therefore, it was decided not to employ them and instead use just a dense output layer to turn into the regression task of predicting the pIC50 value.

The training of the models is directly affected by the activation function, optimizer algorithm, and loss function. The activation function, also known as the transfer function, maps the weighted sum of a neuron's inputs to an output value within the prediction problem's range of values. There are both linear and nonlinear activation functions. Rectified Linear Unit (ReLU), Sigmoid, and Hyperbolic tangent (Tanh) are examples of the latter category.

The linear function does not perform any transformation to the input, hence the output is equal to the input. This is a simple function, and due to the lack of back-propagation, it is incapable of learning complex predictions. The linear function output range of values extends from negative infinity to positive infinity [88]. Therefore, it is employed in the predictor, in the second dense layer of the PWFFN in the Transformer-Encoder architecture to project the vector to its original dimension, and in the output layer of the fine-tuning to predict the affinity without restrictions.

$$f(x) = x \tag{5.1}$$

ReLU is one of the most commonly used functions in deep learning neural networks [88]. It is computationally efficient and simple, identical to the linear function except for negative inputs that return zero, and it can learn complex patterns in data. Similar to the Transformer architecture [57], ReLU function was employed in the PWFFN's first layer.

$$f(x) = max(0,x) \tag{5.2}$$

Optimizers are algorithms responsible for adjusting the architecture's parameters based on minimizing the error, calculated by the loss function, between the predictions and the true values via gradient backpropagation (from the output layer to the input layers) [89]. The rate at which parameters are updated throughout each iteration in order to obtain the minimum error is known as the learning rate. When training models, there may be two opposing problems with the gradient that impede the learning process: the vanishing gradient, in which the gradient is so small that it varies the parameters to update insignificantly, affecting model training convergence; and the exploding gradient, in which the gradient increases excessively and causes the model to diverge [90]. Adaptive Moment Estimation (Adam) and Rectified Adaptive Moment Estimation (RAdam) were selected as the main optimizers given their capacity to obtain high performances.

The Adam optimizer algorithm estimates appropriate learning rates for each neural network weight in order to converge faster to the local minimum error. It combines the first and second-order moments (mean and uncentered variance, respectively) [91]. Additionally, weight decay regularization was applied in the Adam algorithm in order to reduce the weight values to minimize the risk of overfitting.

$$\theta^t = \theta^{t-1} - \alpha \frac{\frac{m_t}{1-\beta_1^t}}{\sqrt{\frac{v_t}{1-\beta_2^t}} + \epsilon} \tag{5.3}$$

, where θ^t represents the weights of a model at time step t, α is a defined learning rate, m and v are the first and second order moments, β_1 and β_2 are the exponential decay of the respective moments, and ϵ is a constant that prevents division by zero. In this study, the values of Adam's hyperparameters were equal to those established in SMILES-BERT [55].

RAdam [92] is an optimizer that attempts to enhance Adam by automating the learning rate adaptation. Adam and the majority of algorithms need to define the appropriate warmup parameter, which is based on a very low learning rate in the initial epochs to compensate for the large variance and limited data that exist in these training steps and can influence the location of the minimum prediction error. RAdam adjusts the learning rate dynamically by assessing variance divergence using a variance rectification term without requiring the specification of extra parameters.

The loss function is crucial to the training of deep learning models since it allows to compute the calculation of the prediction error at each iteration. Due to the SMILES input representation being a multi-integer vector, Sparse Categorical Cross Entropy was used in MLM. This loss function can be expressed as:

$$scce = -\frac{1}{n}\sum_{i=1}^{n}\Big[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)\Big] \tag{5.4}$$

, where n is the output size, y is the true label, and $\hat{y}$ is the predicted one.

Furthermore, the Mean Squared Error (MSE) was used as the loss function across the phase of fine-tuning since it is for a regression task.

$$mse = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.5}$$

Tables 5.4 and 5.5 summarize the hyperparameters selected to obtain the best MLM and pIC50 predictor models, respectively.

The performance of the transformer-encoder predictor was compared to the most popular machine learning models, KNN, RF, and SVM, using the same dataset containing compounds and their pIC50 for AA2AR, with 85% of the data used for training and 15% for testing. Also, to explore the effect of functional groups on masking, two predictors were trained using the same sets but two distinct approaches for masking in MLM: the one used in this study containing FGs and the one applied in SMILES-BERT.

All the models of the framework were implemented and trained on AMD Ryzen 9 3900X and GeForce RTX 3070 8GB with Python 3.9.6, Keras [93], and Tensorflow 2.5 [94], and on Google Colab.

Table 5.4: Optimized hyperparameters used in Masked Learning Modeling. *Early stopping with a patience of 5 and model Checkpoint were used in the training process.

Hyperparameters	Value
D_model (dimensional embedding vector)	512
Dff (feed forward projection size)	1024
Activation function of the PWFFN	ReLU
Number of Encoders	4
Number of heads	4
Epochs*	50
Batch Size	256
Dropout Rate	0.1
Optimizer	Rectified Adam
Learning Rate	0.001
β_1	0.9
β_2	0.999
Weight Decay	0.1
Loss Function	Sparse Categorical Cross Entropy

Table 5.5: Parameter settings for the AA2AR predictor. *Early stopping with a patience of 50 and model Checkpoint were used in the training process.

Hyperparameters	Value
Activation Function	Linear
Epochs*	200
Batch Size	32
Optimizer	Rectified Adam
Learning Rate	1e-5
Loss Function	Mean Squared Error

5.2.2 Machine Learning Predictors

K-Nearest Neighbors

KNN is a simple algorithm that assumes that elements with similar features belong to the same class or have identical values, depending on the problem type (classification or regression, respectively).

In this case study, the method begins by calculating the distance between the molecules, followed by the selection of the K nearest neighbors. The predicted binding affinity is the average of the K neighbor values. It is considered a lazylearning algorithm since data is stored and training occurs only when a predictable element is received.

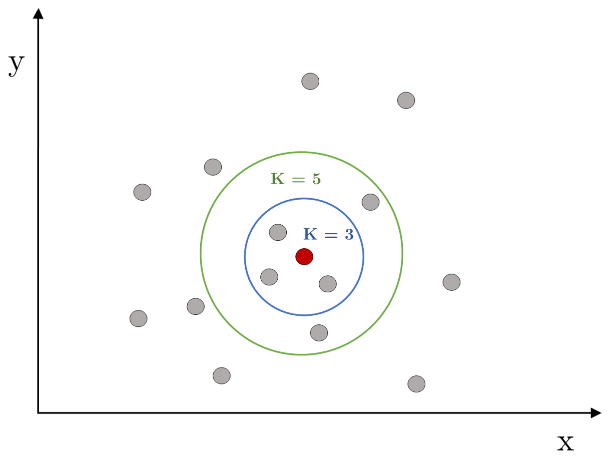


Figure 5.4: K-Nearest Neighbors.

The 5-fold cross-validation technique (Figure 4.11) was used to optimize the parameters *n_neighbors*, which refers to the number of neighbors to consider (odd numbers are preferred to avoid ties), and *metric*, which identifies the mathematical algorithm used to calculate the distances between compounds. All the employed machine-learning approaches were developed using Scikit-Learn [95].

Table 5.6: Hyperparameters of the KNN model.

Hyperparameter	**Value**
metric	manhattan
n_neighbors	3

Random Forest Regression

RF is a machine learning model efficient in classification and regression tasks. This approach employs an ensemble of uncorrelated decision trees, with the final output being the mean of each prediction in the case of regression or the most predicted class in the case of classification. Decision trees contain multiple nodes, and each node divides the dataset into subsets based on a splitting method (e.g., variance minimization for continuous data) applied to one feature. The process of data division continues until the predicted value is determined. Each decision tree was created using bootstrapping for randomly selected training data to ensure noncorrelation. However, the non-sampled data (out-of-the-bag) is used to evaluate the performance. Moreover, each tree considers a random assortment of features. Due to its ensemble learning and bootstrapping, the model is appropriate for different data types and produces precise results.

The optimized hyperparameters (Table 5.7) were identified using 5-fold cross-validation (Figure 4.11). The parameter *n_estimators* is the number of decision trees included in the model, *max_features* represents the number of features used in the split, and the other parameters are Scikit-learn defaults [95].

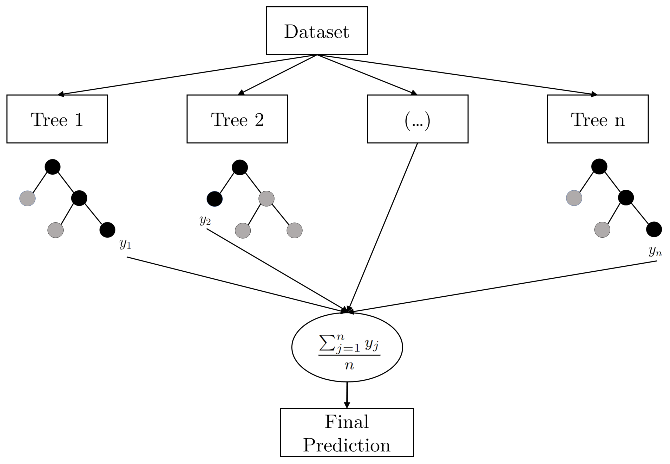


Figure 5.5: Random Forest Regression.

Table 5.7: Hyperparameters of the Random Forest Regression model.

Hyperparameter	**Value**
max_features	sqrt
n_estimators	500

Support Vector Regression

SVR is an adaptation of the SVM algorithm for predicting continuous values in regression tasks. The objective of SVM is to correctly classify the data by identifying an optimal hyperplane that separates the classes with maximum margins. It is applied to both linearly separable and non-linear data, with soft margins and kernel functions in the latter case. Soft-margins permit classes to be misclassified, i.e., data points that pass the hyperplane, within or outside the margins. Furthermore, Kernel functions map data to higher dimensions so that it can be separated linearly (e.g., Radial Basis Function, Polynomial).

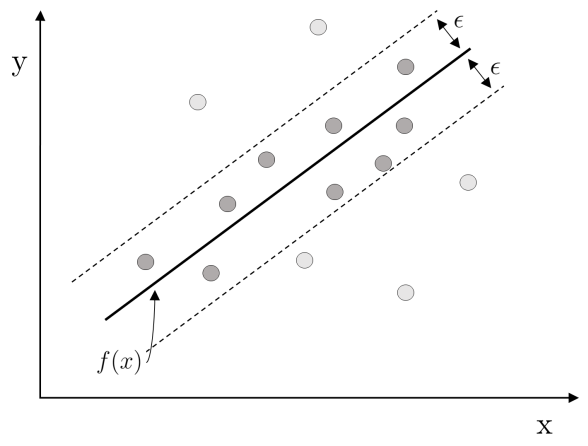


Figure 5.6: Support Vector Regression.

SVR is identical to SVM in terms of the hyperplane concept (regression function in SVR), Kernels, and soft-margin. However, it differs primarily by its objective, which is to minimize the amount of data out of the "tube". The regression line is in the middle of the "tube" whose width is set by the parameter epsilon (ϵ).

Table 5.8: Hyperparameters of the SVR model.

Hyperparameter	**Value**
C	1.0
metric	rbf
gamma	scale

Moreover, SVR is a flexible method whose performance is highly dependent on the parameters. The following parameters were achieved using the 5-fold cross-validation method: *C* (regularizes the margins), the *kernel* function, and *gamma* (the kernel function coefficient).

5.3 Transformer-Decoder Generator

The grid search methodology and the majority of training hyperparameters used by the generator were identical to Masking Learning model. Since the transformer-decoder and transformer-encoder behave similarly (the look-ahead mask is the primary architectural difference), the dropout rate, number of epochs (in grid search and training of the best model), and batch size have the same values as MLM. The optimized hyperparameters are presented in Table 5.9.

Table 5.9: Optimized hyperparameters used in the generator model. *Early stopping with a patience of 5 and model Checkpoint were considered.

Hyperparameters	Value
D_model (dimensional embedding vector)	512
Dff (feed forward projection size)	1024
Activation function of the PWFFN	ReLU
Number of Encoders	8
Number of heads	4
Epochs*	50
Batch Size	256
Dropout Rate	0.1
Optimizer	Rectified Adam
Learning Rate	0.001
β_1	0.9
β_2	0.999
Weight Decay	0.1
Loss Function	Sparse Categorical Cross Entropy
Temperature	0.9

5.4 Optimized Transformer-Decoder Generator

The hyperparameters of the best model to produce molecules with optimized desired properties are summarized in Table 5.10. Furthermore, the number of epochs is related to the number of molecules that are given as input, as well as those that are generated and replaced. After a certain epoch, the pharmacological properties of the generated molecules begin to deteriorate.

The efficiency of the optimization process is dependent on the molecules with the best properties. Consequently, the input substitution thresholds studied were significantly lower than the number of molecules generated in order to make a better selection of molecules with superior qualities. Moreover, a smaller number of molecules were substituted from the input in order to avoid instability in the generator, which could occur if a large proportion of compounds were replaced, causing difficulty in the model convergence. Furthermore, the batch size used was 128, due to an insufficiency of computational resources.

Table 5.10: Optimized hyperparameters used in optimized generator. *Early stopping with a patience of 20 and model Checkpoint were used in the training process.

Parameters	Value
Epochs*	60
Batch Size	128
Optimizer	Rectified Adam
Learning Rate	0.001
Loss Function	Sparse Categorical Cross Entropy
Number of Input Molecules	8500
Number of Generated Molecules	1200
Number of New Input	450
Temperature	0.9
Pharmacological properties to be optimized	pIC50, SAS, LogP, MW, TPSA

5.5 Evaluation Metrics

5.5.1 Predictor

Metrics permit the measurement of a model's capacity to perform a certain task. In this case, the objective of the metrics is to measure, during pre-training, the model's ability to predict the masked tokens from the context and, during fine-tuning, the model's efficiency in determining the affinity of medicines to AA2AR. Overall, the predictor metrics do make comparisons between predicted and true values. However, each one focuses on a certain aspect, and selecting the most appropriate one enables a more objective and reliable evaluation.

Since MLM is a classification problem, accuracy was used along with sparse categorical cross-entropy (5.4).

- Accuracy: proportion of correctly predicted classes.

$$acc = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions} \tag{5.6}$$

Accuracy is a metric that performs better for balanced data, as it can provide incorrectly good accuracy for imbalanced datasets if it correctly predicts solely the class with the most representations, for example. The accuracy of this work is calculated based on the hidden tokens, which were selected randomly. Furthermore, the sparse categorical cross entropy determined the difference between the predicted token and the original in tokens where the mask was applied.

The following regression metrics were applied to the fine-tuned predictor:

- Mean Squared Error (expression 5.5)
- Coefficient of Determination

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.7}$$

- Concordance Correlation Coefficient (CCC) [96]

$$CCC = \frac{2s_{y\hat{y}}}{s_y^2 + s_{\hat{y}}^2 + (\bar{y} - \bar{\hat{y}})^2} \tag{5.8}$$

  with

$$\begin{aligned} \bar{y} &= \frac{1}{n}\sum_{i=1}^{n} y_i \\ s_y^2 &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 \\ s_{y\hat{y}} &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}}) \end{aligned}$$

, where n is the number of samples, y is the true value, $\hat{y}$ is the predicted value and $\bar{\hat{y}}$ is its average. The MSE metric measures the prediction error of the regression. R^2 indicates how well the model fits the test dataset, although it does not provide enough information about the prediction performance. So, CCC was used to evaluate the outcomes' proximity to the optimal line $y_{true} = y_{predicted}$, taking precision and accuracy into account, i.e., how closely the predicted values are to the regression line and how far the regression line deviates from the optimal reference line, respectively.

5.5.2 Generator and Optimized Generator

There are an increasing number of computational algorithms in drug design, so it is necessary to use appropriate metrics to compare their performance. In this study, the following metrics were used to determine the quality of the molecules produced by the generator and the optimized generator:

- Validity: percentage of chemically valid generated compounds. Validity was determined using a RDKit [18] function that converts SMILES strings to molecules and checks the chemical syntax, bonds, and valence atoms.
- Novelty: percentage of valid unique molecules that are novel, i.e., not part of the training dataset. If this metric is too low, the model is overfitted.
- Uniqueness: percentage of valid molecules produced uniquely. If this measure is too low, the model repeatedly creates the same sequences. Unique@10000 is used in MOSES benchmarks to estimate the uniqueness of the first 10000 molecules.
- Internal diversity ($IntDiv_p$): diversity of unique and valid generated molecules (M). Since it uses the Tanimoto similarity (T_s), the SMILES strings need to be transformed into fingerprints. The value range between 0 and 1, and the higher it is, the more diverse the molecules are.

$$IntDiv_p = 1 - \sqrt[p]{\frac{1}{\mid M \mid^2} \sum_{m_1, m2 \in M} T_s(m_1, m_2)^p} \tag{5.9}$$

  with $p \in \{1,2\}$

For the optimized generator, the respective pharmacological characteristics were also determined: pIC50 of AA2AR, SAS, LogP, MW, TPSA, QED and the percentage of molecules that follow strictly Lipinski's Rule of Five [30]. The SAS, LogP, MW, TPSA properties were estimated using the RDKit library [18].

6

Results and Discussion

6.1 Predictor

The predictor plays a crucial role in the proposed framework for the design of novel AA2AR-active compounds. Hence, it is essential that the model effectively predicts the binding affinity of the molecules for the Adenosine A2A Receptor.

The Transformer-Encoder predictor was first pre-trained using MLM. At this stage, the objective is for the model to gain a general knowledge of the SMILES syntax by learning to identify the randomly masked tokens, which can belong to FGs or not. Table 6.1 summarizes the results obtained in the MLM phase in terms of the metrics described in Section 5.5.1.

Table 6.1: Pre-training prediction results in training and testing datasets.

Dataset	**Loss**	**Accuracy**
Training	0.1087	0.9612
Testing	0.1096	0.9611

There is no significant difference between the identified performance of the masked tokens in the training and testing datasets, with both sets presenting promising results with approximately 0.11 of loss and 0.96 of accuracy. The results show the model's ability to determine the elements that were masked using the context of unchanged tokens. Furthermore, the results demonstrated the method's generalization capability, as the testing set's evaluation outcome is similar to the training set's.

Then the model with the pre-trained MLM weights was shifted via fine-tuning to predict the affinity of molecules to the Adenosine A2A receptor. Min-Max Scaler and Robust Scaler were the two types of normalization applied to the pIC50 values of the dataset used for fine-tuning. The table 6.2 shows the results obtained for the fine-tuned predictor using the two types of normalization.

Table 6.2: Evaluation of predictor efficacy using different normalization techniques.

Scaling	**MSE**	R^2	**CCC**
Min-Max Scaler	0.012	0.589	0.801
Robust Scale	0.037	0.650	0.820

Normalization helps the model's convergence, with different types of normalization producing different results. The Robust Scaler outperformed the Min-Max Scaler in the R^2 and CCC metrics, with 0.65 and 0.82, respectively, demonstrating better precision and accuracy. The Min-Max Scaler has the lower MSE; however, this measure is heavily influenced by the distribution of pIC50 values. The difference in performance between the two types of normalization can be explained by the Robust Scaler (equation 4.3) being better suited to this dataset since it employs quartile measures that are less sensitive to outliers compared to the mean.

The framework's predictor forecast normalized values of pIC50 by Robust Scaler, as shown in Figure 6.1. However, the normalized values have no biological significance because they are dependent on the Q1 and Q3 values of the training data distribution. Thus, it is necessary to apply de-normalization to obtain the real pIC50 of AA2AR compounds.

To validate the Transformer-Encoder predictor's efficacy in estimating compounds' affinity for AA2AR, its performance was compared to the most popular ML models. Table 6.3 demonstrates the effectiveness of machine learning models based on the MSE, R^2, and CCC metrics with data normalized by RobustScaler.

Table 6.3: Predictive test data results for machine learning models.

ML Model	**MSE**	R^2	**CCC**
KNN	0.059	0.447	0.668
RF	0.045	0.579	0.712
SVR	0.059	0.446	0.618

The machine-learning approach that performed the best across all metrics was RF, which is typically the most robust for all the variety of data inputs and the least vulnerable to overfitting. Nevertheless, the proposed predictor outperformed Random

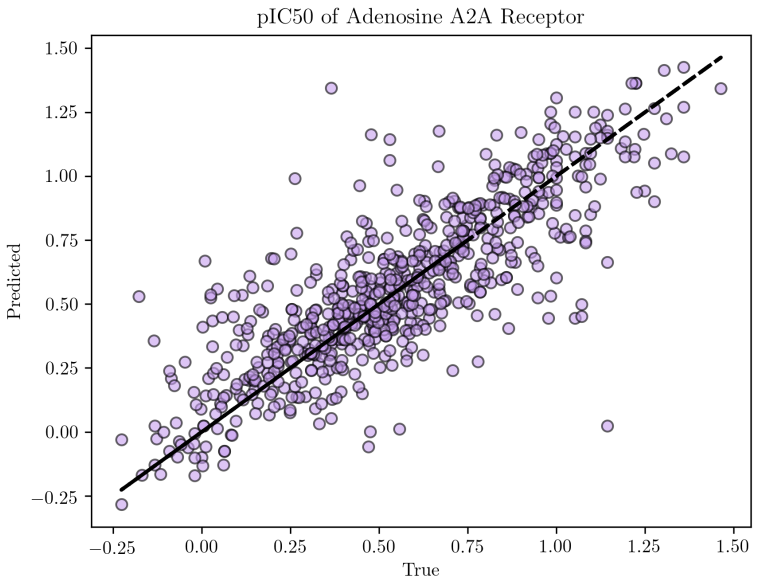


Figure 6.1: Predictions against true values for the binding affinity testing set [83], where the diagonal line is the reference (predicted = true value).

Forest in all three criteria: MSE (0.037), R^2 (0.650), and CCC (0.820). Thus, it demonstrates the effectiveness of the Transformer-Encoder in predicting the binding affinity for the specific target.

In this study, the MLM used differs from its conventional method by applying masks to a percentage of FGs atoms. In order to assess the usefulness of this innovation, two predictors were compared, one with the MLM used in BERT [56] and SMILES-BERT [55] and the other with the MLM employed in this study. The pre-training and fine-tuning results obtained by the models are summarized in Tables 6.4 and 6.5, respectively.

Table 6.4: Hidden token prediction results in training and testing datasets, considering two different masking approaches in MLM with 10 epoch training.

Masked FGs in MLM?	**Dataset**	**Loss**	**Accuracy**
Yes	Training	0.161	0.943
Yes	Testing	0.161	0.944
No	Training	0.149	0.948
No	Testing	0.146	0.949

The results of MLM show that the model that did not mask FGs (used in BERT [56] and SMILES-BERT [55]) appears to learn the SMILES semantics better than the

Table 6.5: Comparison of the Fine-tuning prediction results over the testing set based on two different MLM approaches.

Masked FGs in MLM?	**MSE**	R^2	**CCC**
Yes	0.0382	0.6413	0.8254
No	0.0429	0.5983	0.8106

one proposed in this work, which selects 7% of the tokens belonging to functional groups to mask and 7% of the remaining elements. Due to the limited diversity of functional group atoms, the proposed model learns context based on identical atoms. Nevertheless, the fine-tuned predictor used in the framework outperformed the other model in every metric: MSE (0.038 compared to 0.0429), R2 (0.64 compared to nearly 0.60), and CCC (0.83 compared to 0.81). The outcomes are consistent with the notion that information from the functional groups contributes to the prediction since they affect the properties of the compounds, and molecules with similar FGs usually exhibit identical properties.

Furthermore, the pre-trained predictor with 50 training epochs exhibits a slight improvement over the one with 10 training epochs in MSE and R^2 but is lower for CCC (MSE: 0.037, R^2: 0.65, CCC: 0.82 and MSE: 0.038, R^2: 0.64, CCC: 0.83, respectively).

Based on the obtained results, we can infer that the developed Transformer-Encoder predictor performed efficiently in the application of self-attention mechanisms for the identification of masked tokens. In addition, it was demonstrated that adding the information related to functional groups in the pre-training stage was advantageous for fine-tuning the model.

6.2 Generator

The unbiased generator is the basis of the proposed framework, as it is the model responsible for the development of novel chemical compounds. Therefore, it is essential that the generator be able to produce valid and distinct molecules.

In order to evaluate the performance of the unbiased generator trained with the ChEMBL dataset [83] that was used as a starting point in the framework, 10 000 molecules were generated. This number was chosen because it provided a more stable evaluation while complying with the computational resources. Table 6.6 shows the results of the evaluation metrics mentioned in section 5.5.2 for the 10 000 SMILES molecules produced by the unbiased generator.

Table 6.6: Evaluation of the efficiency of the unbiased generator trained on the ChEMBL dataset [83] relatively to the internal diversity, uniqueness, novelty, and validity.

IntDiv_1	**IntDiv_2**	**Novelty (%)**	**Unique (%)**	**Validity (%)**
0.871	0.863	95.96	99.85	93.52

The results indicate that the model is effective at generating valid and unique compounds (93.5% and almost 100%, respectively), with synthesis variability exceeding 0.86. Furthermore, it has a high degree of novelty, nearly 96% of the unique valid molecules were not present in the training set.

To compare the generation capacity of the Transformer-Decoder with the state-of-the-art generative models, 30000 molecules were generated from the model trained with the MOSES dataset [84]. This number of compounds produced is the minimum amount needed for an appropriate comparison between models, as suggested on the MOSES platform. Table 6.7 demonstrates the outcomes of the comparison between unconditional Transformer-Decoder and the baseline models implemented in the MOSES benchmark [84].

Table 6.7: Comparison of the effectiveness of multiple deep learning models trained on the MOSES dataset [84].

Generator	**IntDiv_1**	**IntDiv_2**	**Novelty (%)**	**Unique@10k (%)**	**Validity (%)**
Transformer-Decoder	0.855	0.849	**97.38**	99.92	91.15
CharRNN	0.856	0.850	84.19	99.94	97.48
AAE	0.856	0.850	79.31	99.73	93.68
VAE	0.856	0.850	69.49	99.84	97.67
JTN-VAE	0.855	0.849	91.43	**99.96**	**100.0**
LatentGAN	**0.857**	**0.850**	94.98	99.68	89.66

The proposed generator has the highest novelty metric value, exceeding 97%, indicating that the Transformer-Decoder is the least susceptible to overfitting when generating novel lead compounds.In other metrics except for validity, the proposed model's performance is comparable to that of models with better results. The validity is not so high; however, this generator was not trained to produce molecules with specific properties, unlike the other models.

The six deep neural networks exhibits similar internal diversity and uniqueness metric values, indicating that they all possess a comparable capacity to generate with reduced redundancy. In JTN-VAE, where molecules are represented by subgraphs, the validity is 100% because this metric is controlled during the model execution. The AAE, VAE, and CharRNN also have a higher validity measure than the Transformer-Decoder, but their performance in the novelty domain is relatively low when compared. Based on the results, the Transformer-Decoder and LatentGAN are the SMILES-based models with the most balanced performance across all five metrics. However, the Transformer-Decoder outperforms LatentGAN in validity, uniqueness, and novelty.

From the acquired results, the unbiased generator, which uses the knowledge of long-short term dependencies between the tokens of the SMILES strings, produces promising unconditional molecules. Moreover, the dataset used (ChEMBL and MOSES) has a visible effect on the performance of the model, evidenced by variations in all evaluation measures of the proposed generator (Tables 6.6 and 6.7). Nevertheless, the proposed model was efficient in both sets.

6.3 Optimized Generator

The generation of compounds with therapeutic properties is essential to the discovery of potential drugs. The primary focus of this study is the optimization of the generator, which needs to be capable of producing drug-like molecules.

Table 6.8 shows the results of training the model to optimize different properties with a training set of 8500 molecules created by the unbiased generator. Each set of 10 000 generated molecules is evaluated for validity, uniqueness, internal distance, novelty, mean value of biological affinity, and the number of molecules in agreement with Lipinski's Rule of Five [30].

Table 6.8: Performance results of the set of biased models trained on the 8500 molecules generated by the unbiased generator.

Type	IntDiv_1	IntDiv_2	Novelty (%)	Unique (%)	Validity (%)	Mean pIC50	% Rule of Five
Unbiased	0.871	0.863	95.76	99.86	93.46	5.86	77.36
pIC50	0.820	0.811	100.0	76.08	96.62	7.64	50.74
pIC50+SAS	0.780	0.771	99.93	82.70	99.37	7.27	69.19
pIC50+SAS+LogP	0.788	0.778	99.94	78.98	99.68	7.04	86.63
pIC50+SAS+MW	0.799	0.790	99.91	80.88	99.88	7.07	91.19
pIC50+SAS+TPSA	0.797	0.787	99.96	76.34	99.84	7.08	69.69
pIC50+SAS+LogP+MW+TPSA	0.826	0.817	99.28	75.41	99.79	6.81	99.36

In terms of novelty, validity, and pIC50, all the optimization approaches outperformed the unbiased generator. This can be explained by the feedback-loop optimization, which, during training in each iteration, replaces its input with the generated valid, unique, and new molecules with high affinity. The training dataset is smaller than the one used by the unbiased generator, allowing for more emphasis on enhancing the SMILES grammatical rules. However, the process of optimizing the Transformer-Decoder led to a reduction in uniqueness and diversity, probably as a result of the lower training data sample size.

On the biased models, pIC50+SAS+LogP+MW+TPSA has the highest percentage of molecules that follow Lipinski's Rule, with 99% of the molecules produced being potentially orally bioactive. The optimization of pIC50+SAS+MW obtained the second best value in this metric, followed by pIC50+SAS+LogP. Optimizing pIC50, pIC50+SAS and pIC50+SAS+TPSA produced molecules with inferior physicochemical properties than the unconditional generation. Therefore, molecular weight and LogP had a considerable influence on the Lipinski Rule violation in this study.

Increasing the number of attributes to be improved has a negative impact on the enhancement of pIC50. This is due to the fact that the multiobjective optimization algorithm gives equal weight to all characteristics.

The pIC50+SAS+LogP+MW+TPSA generator was selected as the best-biased model because of its superior ability to produce compounds with pharmacological properties. Even though its average pIC50 is not the highest, it significantly outperforms the unbiased. Figure 6.2 depicts the evolution of the biological affinity for AA2AR over several training epochs.

The ability of the optimization framework was analyzed by graphically comparing the distributions of 10000 molecules created by the unconditional generator and biased pIC50+SAS+LogP+MW+TPSA model. Figure 6.3 illustrates the comparisons. There is an increase in the affinity for AA2AR. Furthermore, virtually all physicochemical properties are within the desired range. Besides the drug-like effectiveness of the produced molecules, as demonstrated by the maximization of QED, they are also potentially synthesizable (SAS values lower than 5).

The optimized generator was also trained on a larger dataset, including 20000 molecules, to expand its variability since using a shortened dataset to optimize pharmacological qualities could be the cause of reduced diversity and uniqueness of biased molecules compared to unbiased. Table 6.9 summarizes the results obtained

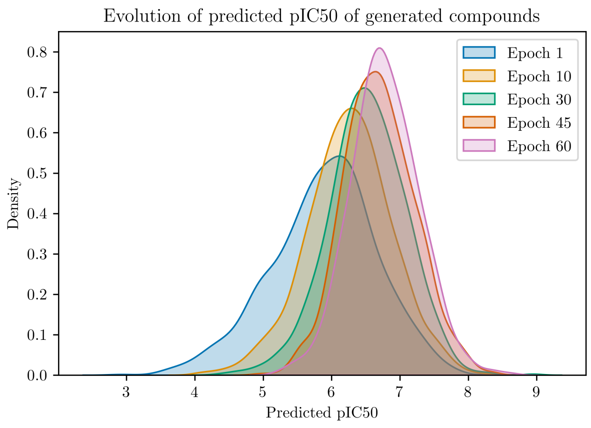


Figure 6.2: Distribution of the predicted pIC50 values in molecules synthesized by the optimized generator (pIC50+SAS+LogP+MW+TPSA) at multiple training epochs.

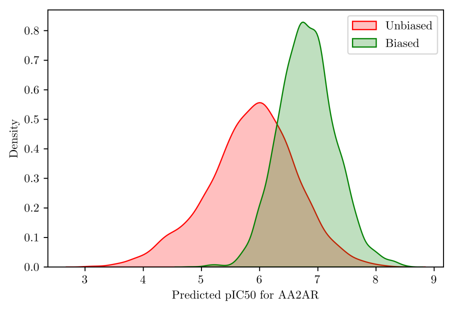
from the pIC50+SAS+LogP+MW+TPSA model with 20000 compounds.

Table 6.9: Evaluation of the optimized model trained with 20 000 molecules generated by the unconditional generator.

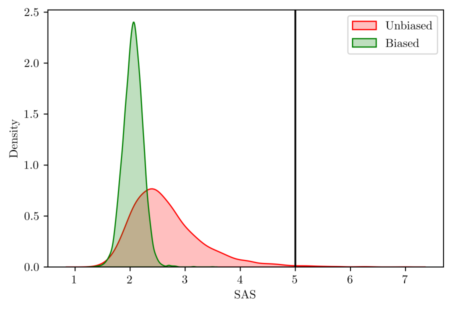
Type	IntDiv_1	IntDiv_2	Novelty (%)	Unique (%)	Validity (%)	Mean pIC50	% Rule of five
pIC50+SAS +LogP+MW +TPSA	0.841	0.830	99.47	89.22	99.78	6.42	96.37

The larger dataset improved the diversity and uniqueness of the generator (from 75% to 89% uniqueness). However, the drug-likeness and biological affinity values deteriorated slightly, which may have been caused by the inappropriate number of epochs or quantity of molecules to replace (which was the same proportion as used in the 8500 compounds).

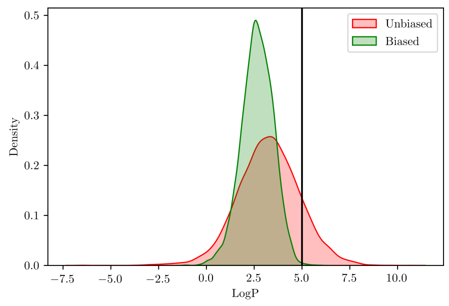
Based on the results, the effectively optimized generator pIC50+SAS+LogP+MW+TPSA trained with 8500 is shown to be potentially suitable for the generation of small molecules with pharmacokinetic parameters, such as absorption by the organism and cell membrane permeability, that allow drugs to be biologically active for the Adenosine A2A Receptor target. Figure 6.4 illustrates a few examples of these SMILES strings. Furthermore, there is a trade-off between obtaining a wider variety of molecules and their desired therapeutic characteristics.



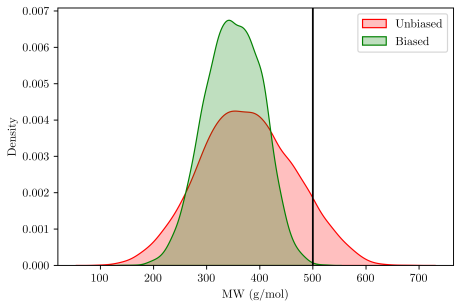
(a)



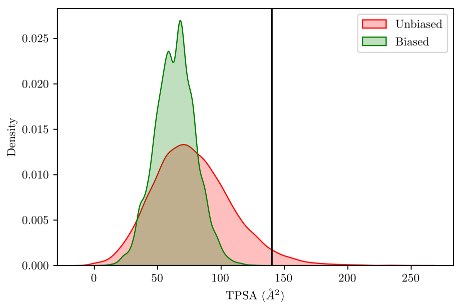
(b)



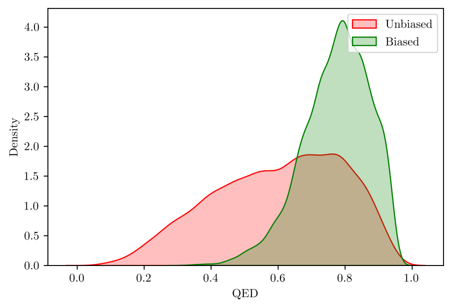
(c)



(d)



(e)



(f)

Figure 6.3: Comparison of the property distributions of the molecules generated by the unbiased model against the biased pIC50+SAS+LogP+MW+TPSA model. (a) Biological affinity of AA2AR. (b) Synthetic accessibility score. (c) Molecular lipophilicity. (d) Molecular weight. (e) Topological polar surface area. (f) Quantitative Estimate of Drug-Likeness. The vertical black lines correspond to the upper boundaries of the desired properties.

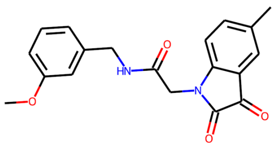
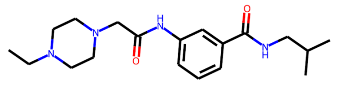
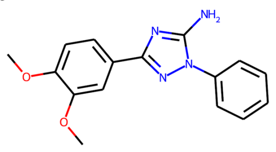
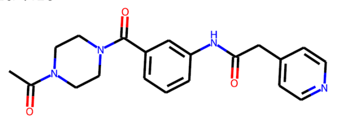
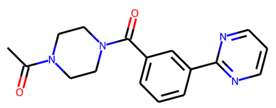
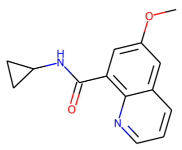
pIC50 for AA2AR: 7.5 SAS: 1.96 LogP: 1.85 MW: 338.36 TPSA: 75.71 QED: 0.84  COc1cccc(CNC(=O)CN2C(=O)C(=O)c3cc(C)ccc32)c1	pIC50 for AA2AR: 7.37 SAS: 1.86 LogP: 1.65 MW: 346.48 TPSA: 64.68 QED: 0.79  CCN1CCN(CC(=O)Nc2cccc(C(=O)NCC(C)C)c2)CC1
pIC50 for AA2AR: 7.78 SAS: 1.98 LogP: 2.53 MW: 296.33 TPSA: 75.19 QED: 0.80  COc1ccc(-c2nc(N)n(-c3ccccc3)n2)cc1OC	pIC50 for AA2AR: 7.16 SAS: 1.96 LogP: 1.57 MW: 366.42 TPSA: 82.61 QED: 0.89  CC(=O)N1CCN(C(=O)c2cccc(NC(=O)Cc3ccncc3)c2)CC1
pIC50 for AA2AR: 6.67 SAS: 1.97 LogP: 1.45 MW: 310.36 TPSA: 66.4 QED: 0.84  CC(=O)N1CCN(C(=O)c2cccc(-c3ncccn3)c2)CC1	pIC50 for AA2AR: 7.31 SAS: 1.94 LogP: 2.14 MW: 242.27 TPSA: 51.22 QED: 0.9  COc1cc(C(=O)NC2CC2)c2ncccc2c1

Figure 6.4: Examples of novel compounds created by the best optimizing generator, along with their biological affinity for AA2AR and physicochemical properties.

6.4 Framework Limitations

The models incorporated into the proposed framework achieved remarkable outcomes, despite having limitations.

This research focuses on *de novo* drug design using an optimization feedback loop with a generator and a predictor. In the preprocessing of the molecule data, the SMILES were inadvertently not standardized into a canonical format, which theoretically promotes semantic consistency by preventing the same structure from being represented by multiple SMILES. Nevertheless, after detecting this flaw, the models were also tested with data in the canonical form, and the models produced identical results, not being confused by the token context.

Large datasets were used in the initial phase of training transformer-based models from scratch because they needed to learn generalized SMILES dependencies. However, with more data available, all these deep neural networks could expand their knowledge, specifically the fine-tuning of the predictor and the optimization of the generator. As demonstrated by the results obtained, the amount of data and its representation influence the efficiency. Despite the limited set sizes, the models perform effectively.

The success of feedback loop generator optimization is highly dependent on the size of the training dataset, the number of training epochs, the number of molecules to be substituted, and the attributes to be enhanced. Despite the difficulty of selecting the appropriate parameters, this study reached a favorable balance. In terms of property enhancement, the optimization algorithm only minimizes or maximizes. Hence, it does not adjust the attributes for a particular range. Nevertheless, the goal was to create active molecules for a target by maximizing affinity and drug-likeness following the Rule of Five, i.e., by optimizing pharmacological properties to avoid exceeding the boundaries.

7

Conclusion

7.1 Final Remarks

In this study, an End-to-End Transformer-based Approach was proposed for the generation of molecules capable of acting on a specific biological target, and possessing appropriate physical and chemical properties for therapeutic efficacy. The strategy for optimizing the drug generation combines two transformer architectures and a multi-objective optimization algorithm in a feedback loop.

We started by developing the predictor model responsible for assessing the affinity of the drug-target complex. Its training was divided into two stages, pre-training and fine-tuning. In pre-training, the MLM technique was implemented with the masking innovation of focusing on functional groups. During fine-tuning, the model learned to predict the binding affinity of drugs for a biological target. The dataset used in fine-tuning required normalization, hence we analyzed the influence of two types of normalization on the prediction efficiency. A significant performance improvement was verified when using the robust outlier method, demonstrating the relevance of an adequate data representation. The performance of the Transformer-Encoder predictor was compared with the ML models KNN, RF, and SVR, and with the predictor with the same architecture but that applied conventional MLM. The proposed predictor approach obtained better results than machine-learning models. In addition, although the predictor's pre-training performance was inferior to the model with the standard MLM, its fine-tuning performance was superior across all metrics. The outcomes demonstrate the viability of the transformer architecture that considers short and long-term dependencies to create a vector representation. Moreover, the significance of performing a pre-train to gain chemical context is evidenced along with the informational potential provided by the functional groups.

The second step was to implement a transformer-decoder generator, which was used to generate molecules with no specific properties. The model was trained on two datasets, one of which aimed to compare its performance with the state-of-the-art generators. The results of the two datasets demonstrate that the generator can generate valid SMILES strings that are nearly 100% unique and almost all non-existent in the training set. The performance of the Transformer-Decoder compared to state-of-the-art models revealed coherence in all metrics and the highest capacity to produce novelty compounds, higher than 97%. The Transformer, which employs self-attention mechanisms, demonstrated the capability to learn SMILES syntax and generalization ability, and is less susceptible to overfitting due to its high novelty rate.

The final phase was to build the generator optimization framework to generate potential drug-like synthesizable compounds with higher activity for a target. It contains a transformer-decoder and a transformer-encoder that were previously trained. Furthermore, an optimization algorithm was used to rank the molecules based on their properties and provide the best chemical compounds to the generator in a feedback loop. Multiple models were trained with various combinations of characteristics to be optimized. The method with the best performance was the one that simultaneously improved pIC50, SAS, LogP, MW, and TPSA, containing 99.36% chemical entities that strictly follow Lipinski's rule of five, i.e. molecules that are likely to be orally bioactive. Moreover, its validity and novelty metrics values are nearly 100%, and the mean affinity to AA2AR measured by pIC50 is 6.81, compared to 5.81 for the non-optimized approach. Since diversity and uniqueness deteriorated throughout the process, we trained with a larger dataset, which improved these measures while decreasing the affinity and proportion of drug-like compounds. The experimental results reveal the framework's flexibility to bias the generator's output and the considerable quality of drug-likeness produced, which is highly dependent on the attributes analyzed in the multi-objective optimization algorithm.

The primary contribution of this dissertation was in the application of models based on Transformer, an interpretable architecture, to design drugs for a biological target while considering simultaneous multiple pharmacokinetic properties that influence pharmacological effectiveness. Furthermore, the proposed framework does not require a training set of SMILES with the desired properties since it can identify the most promising SMILES during training.

7.2 Future Work

The performance of the predictor and generator models is affected by the quantity of information provided. Hence, one of the following steps would be to use larger datasets, especially during the phase of fine-tuning, which had a limited amount of information for training. Therefore, providing additional information to the proposed transformer-based approaches could improve their capacity to learn the relationships between the constituents of each compound in order to acquire a better understanding of the SMILES syntax.

The results obtained by the framework demonstrate the significance of the properties selected to be optimized in drug design. It would be interesting if the multi-objective optimization algorithm included other biological targets and properties, such as toxicity, which is a relevant factor in determining the acceptability of potential drugs.

Molecules designed to bind to a specific biological target can also connect to other targets (off-targets), typically with a lower affinity. In some cases, however, this can result in adverse effects that lead to drug rejection in the preclinical and clinical phases [97]. The consideration of this issue in future research could be beneficial to the discovery of drug-like compounds.

This work used a multi-objective algorithm, although different non-dominated sorting techniques could be employed to explore other outcomes. Furthermore, the algorithm organized the molecules based on the minimization or maximization of pharmacologic characteristics. This approach is not always valid, such as in the case of drugs capable of crossing the blood-brain barrier, for which Hansch et. al. determined that the optimal LogP ranges between 1.5 and 2.7 [98]. In this circumstance, the objective functions for ordering molecules across the different Pareto fronts could have upper and lower limits based on the restriction values.

The predictor forecasts the activity of a molecular entity toward a single target. However, the predictor could be modified to estimate the binding affinity for multiple targets using multitasking fine-tuning. This method could be trained with only one pre-training, followed by a fine-tuning stage where the model is trained to predict various variables simultaneously using different losses. Since diseases may be affected by multiple factors, it would be advantageous to include a multitasking predictor into the framework for the production of potential drugs for multi-targets.

A

Appendix

A.1 Sinusoidal Positional Encoding Demonstration

$$\begin{aligned}\sin(\alpha+\beta) &= \sin\alpha\cos\beta + \cos\alpha\sin\beta \\ \cos(\alpha+\beta) &= \cos\alpha\cos\beta - \sin\alpha\sin\beta\end{aligned} \tag{A.1}$$

Considering the equations 3.1 and 3.2,

$$\begin{bmatrix}\sin\left(\frac{k+x}{10000^{\frac{2i}{d}}}\right) \\ \cos\left(\frac{k+x}{10000^{\frac{2i}{d}}}\right)\end{bmatrix} = \begin{bmatrix}m_1 & m_2 \\ m_3 & m_4\end{bmatrix}\begin{bmatrix}\sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right) \\ \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\end{bmatrix}$$

Using the trigonometric formulas from A.1,

$$\begin{aligned}
&\Leftrightarrow \begin{bmatrix}\sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) + \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) - \sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right)\end{bmatrix} \\
&= \begin{bmatrix}m_1 & m_2 \\ m_3 & m_4\end{bmatrix}\begin{bmatrix}\sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right) \\ \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\end{bmatrix} \\
&= \begin{bmatrix}m_1\sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right) + m_2\cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right) \\ m_3\sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right) + m_4\cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right)\end{bmatrix} \\
&\Leftrightarrow \begin{cases} m_1 = \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ m_2 = \sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ m_3 = -\sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ m_4 = \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right)\end{cases}
\end{aligned} \tag{A.2}$$

Thereby, the matrix M in 3.2 is defined by:

$$M(i) = \begin{bmatrix} \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) & \sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \\ -\sin\left(\frac{x}{10000^{\frac{2i}{d}}}\right) & \cos\left(\frac{x}{10000^{\frac{2i}{d}}}\right) \end{bmatrix} \tag{A.3}$$

Bibliography

[1] J. Drews, “Drug discovery: a historical perspective,” *science*, vol. 287, no. 5460, pp. 1960–1964, 2000.

[2] “General drug categories.” [Online]. Available: https://www.fda.gov/drugs/investigational-new-drug-ind-application/general-drug-categories

[3] R. M. Dick, “2 pharmacodynamics : The study of drug action.”

[4] V. S. Rao and K. Srinivas, “Modern drug discovery process: An in silico approach,” *Journal of bioinformatics and sequence analysis*, vol. 2, no. 5, pp. 89–94, 2011.

[5] A. Pandeyl, “Overview of all phases in drug development and discovery process.” [Online]. Available: https://www.nebiolab.com/drug-discovery-and-development-process/

[6] R. C. Mohs and N. H. Greig, “Drug discovery and development: Role of basic biological research,” *Alzheimer’s & Dementia: Translational Research & Clinical Interventions*, vol. 3, no. 4, pp. 651–657, 2017.

[7] “The drug development process.” [Online]. Available: https://www.fda.gov/patients/learn-about-drug-and-device-approvals/drug-development-process

[8] P. Aparoy, K. Kumar Reddy, and P. Reddanna, “Structure and ligand based drug design strategies in the development of novel 5-lox inhibitors,” *Current medicinal chemistry*, vol. 19, no. 22, pp. 3763–3778, 2012.

[9] S.-F. Zhou and W.-Z. Zhong, “Drug design and discovery: principles and applications,” p. 279, 2017.

[10] V. D. Mouchlis, A. Afantitis, A. Serra, M. Fratello, A. G. Papadiamantis, V. Aidinis, I. Lynch, D. Greco, and G. Melagraki, “Advances in de novo drug de-

sign: From conventional to machine learning methods," *International journal of molecular sciences*, vol. 22, no. 4, p. 1676, 2021.

[11] ShareVault, "The biggest challenges for the pharmaceutical industry in 2022." [Online]. Available: https://www.sharevault.com/blog/life-sciences/biggest-challenges-for-the-pharmaceutical-industry-in-2022

[12] H. Xue, J. Li, H. Xie, and Y. Wang, "Review of drug repositioning approaches and resources," *International journal of biological sciences*, vol. 14, no. 10, p. 1232, 2018.

[13] S. P. Leelananda and S. Lindert, "Computational methods in drug discovery," *Beilstein journal of organic chemistry*, vol. 12, no. 1, pp. 2694–2718, 2016.

[14] F. Mao, W. Ni, X. Xu, H. Wang, J. Wang, M. Ji, and J. Li, "Chemical structure-related drug-like criteria of global approved drugs," *Molecules*, vol. 21, no. 1, p. 75, 2016.

[15] S. Maslehat, S. Sardari, and M. G. Arjenaki, "Frequency and importance of six functional groups that play a role in drug discovery," *Biosciences Biotechnology Research Asia*, vol. 15, no. 3, pp. 541–548, 2018.

[16] L. David, 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.

[17] D. Weininger, "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.

[18] G. Landrum *et al.*, "RDKit: Open-source cheminformatics," 2021.

[19] K. Chemoinformatics, "Dragon 7.0." [Online]. Available: https://chm.kode-solutions.net/pf/dragon-7-0/

[20] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of chemical information and modeling*, vol. 50, no. 5, pp. 742–754, 2010.

[21] M. Panalytical, "Binding affinity." [Online]. Available: https://www.malvernpanalytical.com/en/products/measurement-type/binding-affinity

[22] C. Yung-Chi and W. H. Prusoff, "Relationship between the inhibition constant (ki) and the concentration of inhibitor which causes 50 per cent inhibition (i50)

of an enzymatic reaction," *Biochemical pharmacology*, vol. 22, no. 23, pp. 3099–3108, 1973.

[23] C. Selvaraj, S. K. Tripathi, K. K. Reddy, and S. K. Singh, "Tool development for prediction of pic 50 values from the ic 50 values-a pic 50 value calculator." *Current Trends in Biotechnology & Pharmacy*, vol. 5, no. 2, 2011.

[24] E. Rutkowska, K. Pajak, and K. Jóźwiak, "Lipophilicity–methods of determination and its role in medicinal chemistry." *Acta poloniae pharmaceutica*, vol. 70, no. 1, pp. 3–18, 2013.

[25] S. A. Wildman and G. M. Crippen, "Prediction of physicochemical parameters by atomic contributions," *Journal of chemical information and computer sciences*, vol. 39, no. 5, pp. 868–873, 1999.

[26] H. X. Ngo and S. Garneau-Tsodikova, "What are the drugs of the future?" *MedChemComm*, vol. 9, no. 5, pp. 757–758, 2018.

[27] P. Ertl, B. Rohde, and P. Selzer, "Fast calculation of molecular polar surface area as a sum of fragment-based contributions and its application to the prediction of drug transport properties," *Journal of medicinal chemistry*, vol. 43, no. 20, pp. 3714–3717, 2000.

[28] T. H. Keller, A. Pichota, and Z. Yin, "A practical view of 'druggability'," *Current opinion in chemical biology*, vol. 10, no. 4, pp. 357–361, 2006.

[29] M. P. Pollastri, "Overview on the rule of five," *Current protocols in pharmacology*, vol. 49, no. 1, pp. 9–12, 2010.

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

[31] D. F. Veber, S. R. Johnson, H.-Y. Cheng, B. R. Smith, K. W. Ward, and K. D. Kopple, "Molecular properties that influence the oral bioavailability of drug candidates," *Journal of medicinal chemistry*, vol. 45, no. 12, pp. 2615–2623, 2002.

[32] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins, "Quantifying the chemical beauty of drugs," *Nature chemistry*, vol. 4, no. 2, pp. 90–98, 2012.

[33] P. Ertl and A. Schuffenhauer, "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions," *Journal of cheminformatics*, vol. 1, no. 1, pp. 1–11, 2009.

[34] J. Verma, V. M. Khedkar, and E. C. Coutinho, "3d-qsar in drug design-a review," *Current topics in medicinal chemistry*, vol. 10, no. 1, pp. 95–115, 2010.

[35] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, "Machine learning, neural and statistical classification," 1994.

[36] R. Saravanan and P. Sujatha, "A state of art techniques on machine learning algorithms: a perspective of supervised learning approaches in data classification," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS).* IEEE, 2018, pp. 945–949.

[37] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[38] S. Yousefinejad and B. Hemmateenejad, "Chemometrics tools in qsar/qspr studies: A historical perspective," *Chemometrics and Intelligent Laboratory Systems*, vol. 149, pp. 177–204, 2015.

[39] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and qsar modeling," *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.

[40] V. V. Zernov, K. V. Balakin, A. A. Ivaschenko, N. P. Savchuk, and I. V. Pletnev, "Drug discovery using support vector machines. the case studies of drug-likeness, agrochemical-likeness, and enzyme inhibition predictions," *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 2048–2056, 2003.

[41] X. Yao, A. Panaye, J.-P. Doucet, R. Zhang, H. Chen, M. Liu, Z. Hu, and B. T. Fan, "Comparative study of qsar/qspr correlations using support vector machines, radial basis function neural networks, and multiple linear regression," *Journal of chemical information and computer sciences*, vol. 44, no. 4, pp. 1257–1266, 2004.

[42] Y.-P. Zhou, J.-H. Jiang, W.-Q. Lin, H.-Y. Zou, H.-L. Wu, G.-L. Shen, and R.-Q. Yu, "Boosting support vector regression in qsar studies of bioactivities

of chemical compounds," *European journal of pharmaceutical sciences*, vol. 28, no. 4, pp. 344–353, 2006.

[43] Z. Xiao, Y.-D. Xiao, J. Feng, A. Golbraikh, A. Tropsha, and K.-H. Lee, "Antitumor agents. 213. modeling of epipodophyllotoxin derivatives using variable selection k nearest neighbor qsar method," *Journal of medicinal chemistry*, vol. 45, no. 11, pp. 2294–2309, 2002.

[44] A. Tropsha and Z. Weifan, "Identification of the descriptor pharmacophores using variable selection qsar applications to database mining," *Current pharmaceutical design*, vol. 7, no. 7, pp. 599–612, 2001.

[45] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[46] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[47] V. Vapnik, *The nature of statistical learning theory.* Springer science & business media, 1999.

[48] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[49] G. Ognjanovski, "Everything you need to know about neural networks and backpropagation — machine learning easy and fun." [Online]. Available: https://towardsdatascience.com/everything-you-need-to-know-about-neural-networks-and-backpropagation-machine-learning-made-easy-e5285bc2be3a

[50] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[51] X. Li and D. Fourches, "Inductive transfer learning for molecular activity prediction: Next-gen qsar models with molpmofit," *Journal of Cheminformatics*, vol. 12, no. 1, pp. 1–15, 2020.

[52] G. E. Dahl, N. Jaitly, and R. Salakhutdinov, "Multi-task neural networks for qsar predictions," *arXiv preprint arXiv:1406.1231*, 2014.

[53] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," *Advances in neural information processing systems*, vol. 28, 2015.

[54] M. Popova, O. Isayev, and A. Tropsha, "Deep reinforcement learning for de novo drug design," *Science advances*, vol. 4, no. 7, p. eaap7885, 2018.

[55] S. Wang, Y. Guo, Y. Wang, H. Sun, and J. Huang, "Smiles-bert: large scale unsupervised pre-training for molecular property prediction," in *Proceedings of the 10th ACM international conference on bioinformatics, computational biology and health informatics*, 2019, pp. 429–436.

[56] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[58] A. Oussidi and A. Elhassouny, "Deep generative models: Survey," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*. IEEE, 2018, pp. 1–8.

[59] M. H. Segler, T. Kogej, C. Tyrchan, and M. P. Waller, "Generating focused molecule libraries for drug discovery with recurrent neural networks," *ACS central science*, vol. 4, no. 1, pp. 120–131, 2018.

[60] A. Gupta, A. T. Müller, B. J. Huisman, J. A. Fuchs, P. Schneider, and G. Schneider, "Generative recurrent networks for de novo drug design," *Molecular informatics*, vol. 37, no. 1-2, p. 1700111, 2018.

[61] J. Yasonik, "Multiobjective de novo drug design with recurrent neural networks and nondominated sorting," *Journal of Cheminformatics*, vol. 12, no. 1, pp. 1–9, 2020.

[62] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, "Automatic chemical design using a data-driven continuous representation of molecules," *ACS central science*, vol. 4, no. 2, pp. 268–276, 2018.

[63] D. Polykovskiy, A. Zhebrak, D. Vetrov, Y. Ivanenkov, V. Aladinskiy, P. Mamoshina, M. Bozdaganyan, A. Aliper, A. Zhavoronkov, and A. Kadurin,

"Entangled conditional adversarial autoencoder for de novo drug discovery," *Molecular pharmaceutics*, vol. 15, no. 10, pp. 4398–4405, 2018.

[64] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in *International conference on machine learning.* PMLR, 2018, pp. 2323–2332.

[65] G. L. Guimaraes, B. Sanchez-Lengeling, C. Outeiral, P. L. C. Farias, and A. Aspuru-Guzik, "Objective-reinforced generative adversarial networks (organ) for sequence generation models," *arXiv preprint arXiv:1705.10843*, 2017.

[66] E. Putin, A. Asadulaev, Y. Ivanenkov, V. Aladinskiy, B. Sanchez-Lengeling, A. Aspuru-Guzik, and A. Zhavoronkov, "Reinforced adversarial neural computer for de novo molecular design," *Journal of chemical information and modeling*, vol. 58, no. 6, pp. 1194–1204, 2018.

[67] O. Prykhodko, S. V. Johansson, P.-C. Kotsias, J. Arús-Pous, E. J. Bjerrum, O. Engkvist, and H. Chen, "A de novo molecular generation method using latent vector based generative adversarial network," *Journal of Cheminformatics*, vol. 11, no. 1, pp. 1–13, 2019.

[68] V. Bagal, R. Aggarwal, P. Vinod, and U. D. Priyakumar, "Liggpt: Molecular generation using a transformer-decoder model," 2021.

[69] T. B. Hashimoto, D. Alvarez-Melis, and T. S. Jaakkola, "Word embeddings as metric recovery in semantic spaces," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 273–286, 2016.

[70] A. Kazemnejad, "Transformer architecture: The positional encoding," *kazemnejad.com*, 2019. [Online]. Available: https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

[71] L. Liu, J. Liu, and J. Han, "Multi-head or single-head? an empirical comparison for transformer training," *arXiv preprint arXiv:2106.09650*, 2021.

[72] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[73] M. H. Sazli, "A brief review of feed-forward neural networks," *Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering*, vol. 50, no. 01, 2006.

[74] C. C. Coello, D. Van Veldhuizen, and G. Lamont, "Evolutionary algorithms for solving multi-objective problems kluwer academic publishers," *Norwell, MA, USA*, 2002.

[75] N. A. Afagh and A. K. Yudin, "Chemoselectivity and the curious reactivity preferences of functional groups," *Angewandte Chemie International Edition*, vol. 49, no. 2, pp. 262–310, 2010.

[76] N. Chen, S. Watanabe, J. Villalba, P. Żelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2020.

[77] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.

[78] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.

[79] R. Hall and G. Godin, "Ifg algorithm," 2018. [Online]. Available: https://github.com/rdkit/rdkit/tree/master/Contrib/IFG

[80] P. Ertl, "An algorithm to identify functional groups in organic molecules," *Journal of cheminformatics*, vol. 9, no. 1, pp. 1–7, 2017.

[81] A. Jacobs, *The effect of heteroatoms.* Cambridge University Press, 1997, p. 115–157.

[82] N. Haider, "Functionality pattern matching as an efficient complementary structure/reaction search tool: an open-source approach," *Molecules*, vol. 15, no. 8, pp. 5079–5092, 2010.

[83] D. Mendez, A. Gaulton, A. P. Bento, J. Chambers, M. De Veij, E. Félix, M. P. Magariños, J. F. Mosquera, P. Mutowo, M. Nowotka *et al.*, "Chembl: towards direct deposition of bioassay data," *Nucleic acids research*, vol. 47, no. D1, pp. D930–D940, 2019.

[84] D. Polykovskiy, A. Zhebrak, B. Sanchez-Lengeling, S. Golovanov, O. Tatanov, S. Belyaev, R. Kurbanov, A. Artamonov, V. Aladinskiy, M. Veselov *et al.*, "Molecular sets (moses): a benchmarking platform for molecular generation models," *Frontiers in pharmacology*, vol. 11, p. 565644, 2020.

[85] T. Sterling and J. J. Irwin, "Zinc 15–ligand discovery for everyone," *Journal of chemical information and modeling*, vol. 55, no. 11, pp. 2324–2337, 2015.

[86] J. Zhao, Y. Deng, Z. Jiang, and H. Qing, "G protein-coupled receptors (gpcrs) in alzheimer's disease: a focus on bace1 related gpcrs," *Frontiers in aging neuroscience*, vol. 8, p. 58, 2016.

[87] O. H. Al-Attraqchi, M. Attimarad, K. N. Venugopala, A. Nair, and N. H. Al-Attraqchi, "Adenosine a2a receptor as a potential drug target-current status and future perspectives," *Current Pharmaceutical Design*, vol. 25, no. 25, pp. 2716–2740, 2019.

[88] S. Sharma, "Activation functions in neural networks." [Online]. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

[89] S. Doshi, "Various optimization algorithms for training neural network." [Online]. Available: https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6

[90] K. Pykes, "The vanishing/exploding gradient problem in deep neural networks." [Online]. Available: https://towardsdatascience.com/the-vanishing-exploding-gradient-problem-in-deep-neural-networks-191358470c11

[91] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[92] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019.

[93] F. Chollet *et al.* (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[94] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous

systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[95] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[96] I. Lawrence and K. Lin, "A concordance correlation coefficient to evaluate reproducibility," *Biometrics*, pp. 255–268, 1989.

[97] M. S. Rao, R. Gupta, M. J. Liguori, M. Hu, X. Huang, S. R. Mantena, S. W. Mittelstadt, E. A. Blomme, and T. R. Van Vleet, "Novel computational approach to predict off-target interactions for small molecules," *Frontiers in big data*, vol. 2, p. 25, 2019.

[98] C. Hansch, S. D. Rockwell, P. Y. Jow, A. Leo, and E. E. Steller, "Substituent constants for correlation analysis," *Journal of medicinal chemistry*, vol. 20, no. 2, pp. 304–306, 1977.