Jambura Journal of Biomathematics

RESEARCH ARTICLE

Non-Linear Function Approximation for Predicting Binding Affinity of PPAR-Targeting Antidiabetic Compounds from Molecular Descriptors

La Ode Aman[1,✉], **Widy Susanti Abdulkadir**[1], **Dizky Ramadani Putri Papeo**[1], **Ariani Hutuba**[1], **Teti Sutriyati Tuloli**[1], **Mohamad Adam Mustapa**[1], **Yuszda K Salimi**[2], **Hamsidar Hasan**[1], **Arfan**[3], **Aiyi Asnawi**[4]

[1]Department of Pharmacy, Faculty of Sports and Health, Universitas Negeri Gorontalo, Gorontalo, Indonesia

[2]Department of Chemistry, Faculty of Matematics dan Natural Sciences, Universitas Negeri Gorontalo, Gorontalo, Indonesia

[3]Faculty of Pharmacy, Universitas Halu Oleo, Kendari, Southeast Sulawesi, Indonesia

[4]Faculty of Pharmacy, Universitas Bhakti Kencana, Bandung, West Java, Indonesia

✉Corresponding author. Email: *laode_aman@ung.ac.id*

Abstract. *Diabetes mellitus remains a major global health challenge, necessitating the development of more effective therapeutic agents. The PPAR family plays a crucial role in regulating glucose and lipid metabolism, making it an important target for antidiabetic drug discovery. However, the identification of potent PPAR-targeting compounds is often limited by the high cost and time-consuming nature of experimental approaches. This study aims to develop a non-linear function approximation model to predict docking-derived binding affinity of antidiabetic compounds targeting PPAR using 2D molecular descriptors. A dataset of 3,764 small molecules with IC50 values was curated from the ChEMBL database, followed by data preprocessing to remove duplicates and incomplete entries. Molecular docking simulations were performed using AutoDock Vina to obtain binding affinity scores (kcal/mol), which were used as the target variable. Subsequently, 2D molecular descriptors were calculated from SMILES representations to capture key structural and physicochemical properties of the compounds. These descriptors were used as input features for a Multi-Layer Perceptron (MLP) regression model to approximate the complex non-linear relationship between molecular structure and binding affinity. The model achieved R^2 values of 0.853 for the training set and 0.632 for the test set, indicating moderate predictive performance and acceptable generalizability. Overall, this approach demonstrates the potential of machine learning as a cost-effective and scalable tool to support early-stage discovery of antidiabetic compounds targeting the PPAR family.*

Check for updates

Article Identity
Jambura J. Biomath.
Volume 7 Issue 1 Pages 152 – 175
March 2026, E-ISSN 2723-0317

Article History
Received 12 July 2025
Revised 6 March 2026
Accepted 18 March 2026
Published 30 March 2026

Keywords
PPAR, Molecular docking, Binding affinity prediction, Molecular descriptors, Machine learning

Editorial office: Department of Mathematics, Universitas Negeri Gorontalo, Jln. Prof. Dr. Ing. B. J. Habibie, Bone Bolango 96554, Indonesia

To Cite this Article: Aman L et al.. Non-Linear Function Approximation for Predicting Binding Affinity of PPAR-Targeting Antidiabetic Compounds from Molecular Descriptors. Jambura Journal of Biomathematics. 2026;7(1):152-175. doi:10.37905/jjbm.v7i1.18

1. Introduction

Diabetes mellitus remains one of the most pressing global health challenges, with its prevalence escalating across both developed and developing countries. According to the International Diabetes Federation, approximately 537 million adults were living with diabetes in 2021, a figure projected to increase significantly in the coming decades [1]. This chronic metabolic disorder, characterized by persistent hyperglycemia due to insulin resistance and/or inadequate insulin secretion, is associated with severe complications, including cardiovascular disease, renal dysfunction, and increased mortality [2, 3]. The economic burden of diabetes is equally alarming, with the global cost of diabetes estimated at USD 966 billion in 2021 [4, 5]. These challenges underscore the urgent need for more effective and accessible antidiabetic therapies.

Among the molecular targets for type 2 diabetes (T2D), peroxisome proliferator-activated receptors (PPARs) have garnered significant attention due to their central role in regulating lipid metabolism, glucose homeostasis, and insulin sensitivity [6]. PPARs, comprising three isoforms—PPAR-α, PPAR-$\frac{\beta}{\delta}$, and PPAR-γ—are nuclear transcription factors that modulate gene expression involved in metabolic pathways [7]. Agonists of PPAR-γ, such as thiazolidinediones, have been clinically employed to enhance insulin sensitivity; meanwhile, PPAR-δ has been implicated in glucose uptake and energy balance, highlighting the therapeutic potential of targeting these receptors [8–10]. However, despite their promise, the discovery of novel PPAR modulators remains challenging due to the complexity of receptor-ligand interactions and the isoform-specific responses they elicit [11, 12].

Traditional drug discovery approaches, particularly experimental binding affinity assays, are time-consuming, resource-intensive, and limited in scalability [13]. To overcome these barriers, artificial intelligence (AI), specifically machine learning (ML) and deep learning (DL), has emerged as a transformative tool for virtual screening, molecular docking, and quantitative structure–activity relationship (QSAR) modeling [14–16]. AI models have demonstrated substantial success in predicting binding affinities, optimizing lead compounds, and accelerating the early stages of drug discovery [17, 18]. Recent studies have leveraged deep learning architectures to refine scoring functions in protein-ligand docking and to guide de novo molecular generation, achieving improved accuracy and efficiency [19–21]. This direction is also reflected in recent studies published in Jambura Journal of Biomathematics, where machine learning approaches were applied to biomedical prediction problems [22, 23].

Despite these advancements, AI-based binding affinity prediction for PPAR-targeting antidiabetic compounds faces several limitations. First, the scarcity of high-quality, diverse datasets for PPAR-ligand systems constrains model generalizability and robustness [24, 25]. Second, many existing models focus narrowly on structural features while neglecting other physicochemical properties essential for accurate binding affinity prediction [26]. Third, most AI models lack explicit incorporation of the mathematical underpinnings of molecular interaction modeling, such as non-linear function approximation or regression formulations tailored for bioactivity prediction [27, 28]. Furthermore, the absence of dedicated computational frameworks specifically designed for PPAR-ligand systems limits their applicability to antidiabetic drug discovery.

To address these gaps, this study proposes a non-linear function approximation model implemented using a Multi-Layer Perceptron (MLP) to predict the binding affinity of antidiabetic compounds targeting PPARs. The MLP architecture, consisting of multiple interconnected layers and non-linear activation functions, has been widely recognized for its ability to capture complex relationships between molecular descriptors and biological activities [29, 30]. In this framework, the binding affinity prediction task is formulated as an approximation of an unknown non-linear function that maps molecular descriptors to continuous binding affinity values, a concept central to modern AI-driven drug discovery efforts [31, 32].

Mathematically, the relationship between the molecular descriptor vector and the predicted

binding affinity can be expressed as:

$$\hat{y} = f(x;\theta), \tag{1}$$

where x represents the vector of molecular descriptors derived from the chemical structure of the ligand, $\hat{y}$ is the predicted binding affinity value, and θ denotes the model parameters, including weights and biases, optimized during training.

The goal of the model is to approximate the true, but unknown, mapping function f by minimizing the difference between predicted and actual binding affinity values using a suitable loss function. This mathematical formulation ensures that the predictive process is not merely empirical but grounded in a formal function approximation paradigm, enhancing both interpretability and robustness [33].

The integration of this AI-driven mathematical framework holds the potential to revolutionize antidiabetic drug discovery by streamlining lead identification, reducing reliance on costly experimental methods, and facilitating precision medicine approaches [34]. Ultimately, the adoption of computational models grounded in rigorous mathematical principles can accelerate therapeutic development and improve clinical outcomes for diabetes patients.

The main contributions of this study are as follows:

1. A curated, multi-stage dataset pipeline — spanning 15,161 ChEMBL compounds down to 2,180 potent ligands — combined with molecular docking simulations against PPARδ (PDB: 7VWG) using AutoDock Vina, providing a reproducible benchmark dataset for PPAR-targeted binding affinity prediction.
2. A non-linear function approximation model based on a Multi-Layer Perceptron (MLP) trained on 210 two-dimensional (2D) molecular descriptors computed with RDKit, directly targeting docking-derived binding affinity $\left(\frac{\text{kcal}}{\text{mol}}\right)$ as the prediction endpoint.
3. A systematic benchmarking of the proposed MLP against linear baseline models (Linear Regression and Ridge Regression), demonstrating that non-linear architecture is essential for capturing the complex structure–activity relationships inherent in PPAR-ligand systems.
4. A formal mathematical formulation of the binding affinity prediction task as a non-linear function approximation problem, providing a principled framework that enhances model interpretability and reproducibility.

The remainder of this paper is organized as follows. Section 2 presents the MLP model architecture and its mathematical formulation. Section 3 describes the computational infrastructure used in this study. Section 4 details the data collection and preprocessing pipeline. Section 5 describes the model training and evaluation procedure. Section 6 presents the experimental results, including docking outcomes, descriptor computation, and model performance. Section 7 provides a critical discussion of the findings, and Section 8 concludes the paper with a summary and directions for future work.

2. Model Architecture

The predictive model developed in this study is based on a feedforward neural network (FNN), also known as a Multi-Layer Perceptron (MLP), specifically designed to predict the binding affinity of antidiabetic compounds targeting the Peroxisome Proliferator-Activated Receptor (PPAR) family. The input to the model consists of molecular descriptor vectors derived from the chemical structures of the ligands.

The network architecture comprises three fully connected hidden layers to capture the complex and non-linear interactions among the molecular descriptors. The first hidden layer consists of 128 neurons, while the second and third hidden layers each consist of 64 neurons. To introduce non-linearity and enhance the network's learning capacity, the Rectified Linear Unit (ReLU) activation function is applied to each hidden layer [35]. Mathematically, the output of each hidden layer is

defined as:

$$h^{(l)} = \mathrm{ReLU}\left(W^{(l)}h^{(l-1)} + b^{(l)}\right), \tag{2}$$

where $h^{(l)}$ represents the output of the l-th layer, $W^{(l)}$ and $b^{(l)}$ denote the weight matrix and bias vector of the l-th layer, respectively, and $h^{(l-1)}$ is the output of the preceding layer, with $h^{(0)}$ corresponding to the input molecular descriptor vector.

The ReLU activation function is formally defined as:

$$\mathrm{ReLU}(z) = \max(0, z). \tag{3}$$

The final output layer consists of a single neuron without an activation function, producing a continuous numerical value representing the predicted binding affinity. This design is suitable for regression tasks [29].

To train the model, the Mean Squared Error (MSE) loss function is employed, which is defined as:

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{4}$$

where y_i is the true binding affinity value, $\hat{y}_i$ is the predicted value from the model, and n is the number of training samples.

The optimization of the model parameters, including weights and biases, is conducted using the Adam optimizer [36], which updates parameters based on the gradients of the loss function:

$$W^{(l)} = W^{(l)} - \eta\frac{\partial L}{\partial W^{(l)}},$$
$$b^{(l)} = b^{(l)} - \eta\frac{\partial L}{\partial b^{(l)}},$$

where η denotes the learning rate.

Recent studies have demonstrated the effectiveness of deep learning, particularly MLPs, in modeling complex relationships between molecular structure and bioactivity, often surpassing traditional methods [31, 37]. This architecture was carefully selected to ensure the model's ability to learn and generalize from molecular descriptor data, thereby enabling accurate prediction of binding affinities for potential antidiabetic compounds targeting the PPAR family.

3. Computational Infrastructure

To ensure efficient execution of this study, a combination of hardware and software tools was utilized for data preparation, molecular docking simulations, and deep learning model development.

3.1. Hardware configuration

The majority of computational tasks, excluding molecular docking simulations, were executed on a Hewlett-Packard HP Z840 workstation. This system, equipped with a 40-core Intel® Xeon® E5-2650 v3 processor, 32.0 GiB of RAM, and an NVIDIA GeForce RTX™ 3070 GPU, provided the processing power required for demanding tasks such as descriptor generation and neural network training. Its 3.0 TB storage capacity ensured efficient data handling and storage throughout the project. These specifications were instrumental in managing large datasets and performing computationally intensive operations.

Molecular docking simulations, which required high computational resources, were conducted on the Fugaku Supercomputer. This facility provided the necessary performance for ligand–receptor binding affinity calculations, enabling the study to obtain reliable docking results.

3.2. Software environment

The computational setup for this study was based on the Ubuntu 24.10 (64-bit) operating system, supported by the Linux kernel version 6.11.0-9-generic. The GNOME 47 desktop environment with the Wayland windowing system provided a stable and efficient user interface. This configuration was chosen to ensure compatibility and optimal performance with the required scientific software tools.

3.3. Key computational tools

Python 3.11 served as the central programming platform, supplemented by specialized libraries to streamline specific tasks. RDKit [38] was used for cheminformatics operations, including molecular fingerprint generation and SMILES processing. Open Babel [39] facilitated molecular file conversions to ensure interoperability. TensorFlow [40] was employed for developing and training deep learning models, while Scikit-learn [41] supported preprocessing and performance evaluation. Pandas [42] enabled efficient data manipulation.

Molecular docking simulations were carried out using AutoDock Vina [43] on the Fugaku Supercomputer, which allowed precise prediction of binding affinities. Data visualization was performed with Matplotlib [44] and Seaborn, which provided detailed and aesthetically pleasing graphical representations of results.

4. Data Collection

4.1. Macromolecule and inhibitor identification

Protein targets were retrieved from the ChEMBL database using the keyword "PPAR," corresponding to the Peroxisome Proliferator-Activated Receptor family. This search identified key isoforms, including PPARα, PPARγ, and PPARδ, which play crucial roles in regulating lipid and glucose metabolism, making them significant targets for antidiabetic therapies.

Small molecule inhibitors were subsequently extracted by filtering for compounds with IC_{50} values, ensuring that the dataset included bioactivity data necessary for accurate modeling. Compounds lacking IC_{50} data or showing insufficient potency (IC_{50} above a defined threshold) were excluded to focus on the most promising inhibitors. The Python scripts used to perform these queries and extractions are provided in Appendices A and B.

4.2. Data preparation and cleaning

A meticulous data-cleaning process was employed to enhance the quality and reliability of the dataset. Duplicate entries, identified by identical `molecule_chembl_id` values, were consolidated by retaining the record with the lowest IC_{50} value, which indicates higher potency. Rows containing missing or invalid IC_{50} values, such as NaN or zero, were removed to prevent inaccuracies in downstream analyses. Additionally, columns irrelevant to the computational workflows, such as `pref_name` and `search_term`, were eliminated to streamline the dataset.

These steps produced a high-quality dataset, optimized for molecular descriptor generation, docking simulations, and machine learning model development. Detailed descriptions of the data-cleaning procedures are included in Appendix C.

4.3. Binding affinity calculation

Binding affinities were predicted using AutoDock Vina, a widely adopted molecular docking tool[43], building on the established methodologies of molecular docking [45]. This involved careful preparation of both the macromolecule (receptor) and the small molecules (ligands) to ensure accurate and biologically relevant simulations.

The 3D structure of the PPARδ protein used in this study was obtained from crystallographic data reported by Oyama et al (PDB ID: 7VWG) [46]. Prior to docking, the protein structure was

refined to ensure its stability and functional integrity. Water molecules, ions, and other non-essential elements were removed to prevent interference with ligand binding. The binding site was identified and defined using a grid box centered at coordinates X = 15.884, Y = 1.890, Z = 39.237, with dimensions X = 40, Y = 40, and Z = 40. This configuration ensured that the docking simulations were focused on the receptor's active site, capturing biologically meaningful interactions.

To validate the binding site, a re-docking procedure was conducted with the co-crystallized ligand from PDB ID 7VWG. This step verified the accuracy of the docking protocol, ensuring reliable predictions for subsequent simulations.

AutoDock Vina was configured to achieve high accuracy in predicting ligand-receptor interactions. The grid spacing was set to 0.375 to provide a fine resolution for the search space. The exhaustiveness parameter was adjusted to 32 to thoroughly explore potential binding conformations, balancing precision and computational efficiency. To optimize performance, the simulations were executed using 32 CPU threads.

The Vina scoring function was employed to estimate the binding affinities, ranking the ligands based on their predicted interaction strength with the PPARδ receptor. This systematic approach ensured reliable predictions, providing a solid foundation for evaluating the potential of candidate antidiabetic compounds.

The molecular docking of all ligands against the target PPAR receptor was performed in an automated manner using a Python-based workflow. The complete code used for ligand preparation, docking execution, and extraction of binding affinity results is provided in Appendix D.

The most favorable docking pose targeting the PPAR receptor yielded a binding affinity of -10.69 $\frac{\text{kcal}}{\text{mol}}$. The results demonstrated a high degree of agreement between the docked ligand pose and its native conformation, affirming the reliability of the docking procedure. Figure 1 illustrates the superimposition of the docked ligand (cyan carbon) and the native ligand (pale green carbon) within the PPAR binding site.

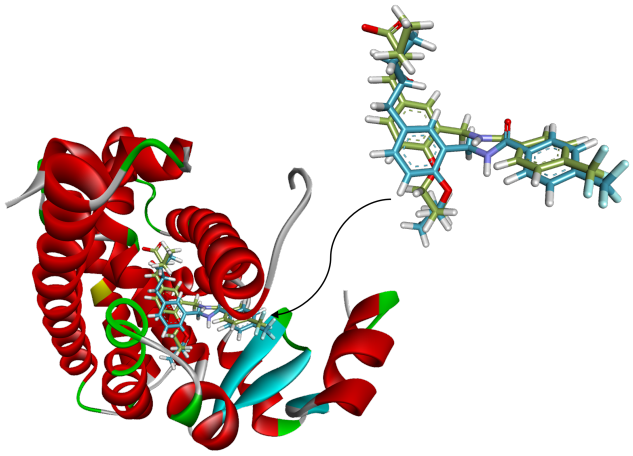

Figure 1. Superimposition of the docked ligand (cyan) and the native conformation (pale green) within the PPARδ active site.

4.4. Molecular descriptor calculation

To numerically represent the chemical and structural features of the ligands, 2D molecular descriptors were calculated using the RDKit library. These descriptors encompassed topological, geometric, and electronic properties, enabling comprehensive molecular characterization.

Selected descriptors were tailored to capture the relevant molecular attributes associated with ligand-receptor interactions. Each molecule's descriptor values were formatted as feature vectors,

forming the input dataset for subsequent model training. Details of descriptor generation and the corresponding Python code are outlined in Appendix E.

5. Model Training and Evaluation

The molecular descriptors served as input features (X), while the corresponding binding affinity values (y) were used as the target variable. For model training and validation, the dataset was partitioned into two subsets: 80% for training and 20% for testing. To enhance numerical stability and improve model convergence, the binding affinity values were scaled prior to training.

The model was trained using the Adam optimizer, employing the Mean Squared Error (MSE) as the loss function. Training was conducted over 50 epochs with a batch size of 32. To monitor performance and prevent overfitting, a validation set comprising 20% of the training data was used during the training process.

Model evaluation was carried out by computing standard regression performance metrics, including the mean squared error (MSE), which measures the average of the squares of the prediction errors; the mean absolute error (MAE), which represents the average absolute difference between predicted and true values; and the coefficient of determination (R^2), which quantifies the proportion of variance in the target variable explained by the model.

These metrics were calculated for both the training and testing subsets to assess model accuracy, error magnitude, and generalizability. A complete implementation of the workflow, including Python scripts for data preprocessing, model training, and evaluation, is provided in Appendix F.

6. Results

The search for macromolecular targets using the keyword "PPAR" identified proteins associated with the Peroxisome Proliferator-Activated Receptor (PPAR) family. Each target was categorized by its respective `target_chembl_id` and the associated preferred name (`pref_name`), representing individual receptors within the PPAR family.

Table 1 summarizes the macromolecular targets identified from the "PPAR" keyword search, highlighting not only the primary PPAR receptor isoforms but also several interacting proteins such as coactivators and corepressors, which play essential roles in modulating PPAR-mediated transcriptional activity.

Table 1. Macromolecular targets associated with the PPAR family identified through the "PPAR" keyword search.

target_chembl_id	pref_name
CHEMBL3559683	Peroxisome proliferator-activated receptor
CHEMBL239	Peroxisome proliferator-activated receptor alpha
CHEMBL3979	Peroxisome proliferator-activated receptor delta
CHEMBL235	Peroxisome proliferator-activated receptor gamma
CHEMBL6116	Peroxisome proliferator-activated receptor gamma coactivator 1-alpha
CHEMBL2095162	Peroxisome proliferator-activated receptor gamma/Nuclear receptor coactivator 1
CHEMBL2095163	Peroxisome proliferator-activated receptor gamma/Nuclear receptor coactivator 2
CHEMBL2095161	Peroxisome proliferator-activated receptor gamma/Nuclear receptor coactivator 3
CHEMBL2096976	Peroxisome proliferator-activated receptor gamma/Nuclear receptor corepressor 2

6.1. Identifying small molecules as PPAR modulators

The subsequent phase focused on identifying small molecule modulators targeting the PPAR receptor family. This involved querying the ChEMBL database using the `target_chembl_id` identifiers retrieved from the macromolecule search. The dataset was curated to include only compounds with available IC_{50} values, ensuring the inclusion of bioactivity data relevant for evaluating modulator potency.

A total of 15,161 small molecules were initially retrieved as potential modulators of the PPAR receptor family. After removing entries with missing or invalid IC_{50} values, 9,947 ligands with valid IC_{50} records were retained. Further filtering based on a potency threshold reduced this to 3,764 compounds, which formed the working dataset for subsequent steps. The IC_{50} values, recorded in the `standard_value` column, served as the key metric for determining compound efficacy, with lower IC_{50} values indicating stronger modulatory activity. These values are expressed in nanomolar (nM) units.

A representative subset of the dataset is presented in Table 2, showcasing selected modulators along with their IC_{50} values and supplementary details. The `pref_name` column highlights the specific PPAR receptor targeted by each compound, while the `canonical_smiles` column contains the SMILES notation of each compound's molecular structure, enabling further computational analyses.

Table 2. Sample of small molecule modulators targeting the PPAR receptor family, including IC_{50} values and SMILES representations.

molecule_chembl_id	IC_{50} (nM)	canonical_smiles
CHEMBL327767	10000	CCCc1cc(Oc2ccc(CC(C)C)cc2)ccc1OCCCOc1cccc(C2SC(=O)NC2=O)c1
CHEMBL94496	2100	CCCc1cc(Oc2ccc(C(C)C)cc2)ccc1OCCCOc1cccc(C2SC(=O)NC2=O)c1
CHEMBL420441	100	CCCc1cc(Oc2ccc(Cl)cc2)ccc1OCCCOc1cccc(C2SC(=O)NC2=O)c1
CHEMBL121	50000	CN(CCOc1ccc(CC2SC(=O)NC2=O)cc1)c1ccccn1
CHEMBL330191	5000	CCCc1cc(Oc2ccccc2)ccc1OCCCCOc1cccc(C2SC(=O)NC2=O)c1
CHEMBL300629	50000	CCCc1cc(Oc2ccccc2)ccc1OCCCOc1ccc(C2SC(=O)NC2=O)cc1
CHEMBL328615	162	CCCc1cc(Oc2ccc(Cl)c(C)c2)ccc1OCCCOc1cccc(C2SC(=O)NC2=O)c1

The complete dataset, comprising 3,764 small molecules with associated IC_{50} values, was prepared for subsequent steps, including data preprocessing, molecular docking simulations, and the construction of machine learning models for binding affinity prediction. The detailed Python implementation for this process is provided in Appendix B.

6.2. Data preprocessing

The dataset from Step 2, initially containing 3,764 small molecules, underwent a comprehensive cleaning process to ensure its suitability for further analysis. Duplicate entries with identical `molecule_chembl_id` were identified and removed. From the duplicates, only the entries with the lowest `standard_value`, indicating the most potent inhibitors, were retained to preserve the relevance and accuracy of the dataset.

Additionally, entries with missing or invalid values in the `standard_value` column, such as NaN or zero, were discarded. These incomplete or erroneous data points could not contribute meaningful insights for binding affinity predictions, and their exclusion improved the overall quality of the dataset.

Columns not required for molecular docking or machine learning tasks, such as `pref_name` and `search_term`, were also removed to streamline the dataset for the computational workflows.

After these cleaning steps, the dataset was refined to 2,191 unique small molecules, formatted and prepared for molecular docking and the creation of predictive deep learning models. The Python code used for this data cleaning process is provided in Appendix C.

6.3. Binding affinity calculation using AutoDock Vina

The binding affinities of the 2,191 unique small molecules retained after preprocessing (Step 3) were predicted using molecular docking with AutoDock Vina. The process began with preparing each ligand from the cleaned dataset, using its `canonical_smiles` string. These SMILES strings were converted into the PDBQT format required for docking simulations in AutoDock Vina. This conversion was performed using RDKit and Open Babel, with the corresponding Python code provided in Appendix D.

Following ligand preparation, molecular docking simulations were carried out using AutoDock Vina. Each ligand was docked with a receptor protein representing the PPAR target. The output from

these simulations provided binding affinity predictions, which were extracted from the "REMARK VINA RESULT" section of the output file, indicating the optimal docking score.

The predicted binding affinities, measured in $\frac{\text{kcal}}{\text{mol}}$, reflect the strength of the interaction between each ligand and its respective receptor. These results were compiled into a new dataset and saved in CSV format for subsequent analysis. The dataset includes key columns such as `molecule_chembl_id` (unique identifier for each ligand), `canonical_smiles` (SMILES string for the ligand's structure), `standard_value` (IC_{50} value in nM), and `binding_affinity` $\left(\text{the predicted binding affinity in } \frac{\text{kcal}}{\text{mol}} \text{ from the docking simulations}\right)$.

An example of the docking results is shown in Table 3, where several small molecules are listed along with their IC_{50} values and predicted binding affinities. The receptor used in the docking simulations was a PPAR protein model in PDBQT format, and the final docking results were stored in the output file `ppar_docking.csv`.

Table 3. Example of docking results showing small molecules, IC_{50} values, and predicted binding affinities.

molecule_chembl_id	IC_{50} (nM)	Binding Affinity $\left(\frac{\text{kcal}}{\text{mol}}\right)$
CHEMBL3678131	0.06	-10.208
CHEMBL3695832	0.13	-11.550
CHEMBL5191837	0.16	-8.040
CHEMBL5207130	0.17	-8.425
CHEMBL3678134	0.20	-10.001
CHEMBL3678128	0.20	-10.260
CHEMBL5187164	0.22	-8.456
CHEMBL5179281	0.23	-7.749
CHEMBL5206512	0.25	-8.428
CHEMBL2088421	0.28	-9.021

6.4. 2D descriptor calculation

In this step, 2D molecular descriptors were calculated for each small molecule in the dataset, including the ligands and their binding affinities obtained from the docking simulations in Step 4. These 2D descriptors offer a numerical representation of the molecular structure, capturing various structural and physicochemical properties, making them ideal for input into machine learning models.

The calculation process began by deriving the 2D descriptors for each ligand. These descriptors were based on the SMILES notation of each molecule, which was processed to obtain values representing various molecular features, such as molecular weight, lipophilicity, and polarity. A variety of 2D descriptors were computed, covering topological, geometric, and physicochemical properties of the molecules.

A total of 210 2D molecular descriptors were computed per ligand using RDKit [38], covering topological indices (e.g., BalabanJ, BertzCT, Chi series), physicochemical properties (e.g., MolWt, MolLogP, TPSA, qed), electronic properties (e.g., EState indices, partial charges), and fragment-based descriptors (e.g., fr_amide, fr_ether, fr_benzene). No descriptors were removed due to missing values, as all 210 features were complete across the 2,180-ligand dataset. All features were subsequently standardized using `StandardScaler` prior to model training, yielding a final input dimensionality of 210.

6.5. Model training and evaluation

In this study, a deep learning model was developed to predict the binding affinity of ligands targeting the PPAR receptor family, based on their 2D descriptors. The model's performance was

assessed using several key metrics, including Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2) values, for both the training and test datasets.

To focus on potent inhibitors, the dataset was filtered to include only ligands with docking-derived binding affinity values $\leq$ -5 $\frac{\text{kcal}}{\text{mol}}$, yielding 2,180 ligands for model development. The molecular structures of these ligands were represented by their 2D descriptors, derived from the SMILES notation of each molecule. These descriptors capture important molecular properties and structural features relevant to predicting binding affinity.

Before training, the binding affinity values were normalized using the `StandardScaler` to improve model convergence. The 2,180-ligand dataset was then divided into training and test sets, with 80% (1,744 samples) allocated for training and 20% (436 samples) reserved for testing. A deep learning model was built using the `TensorFlow Keras` library, designed to predict the binding affinity values of ligands as a regression task. The model architecture began with an input layer designed to accept 2D descriptor values, representing the ligands' structural features. This was followed by three fully connected hidden layers, containing 128, 64, and 64 neurons, respectively, with the ReLU activation function applied to introduce nonlinearity and enhance the model's learning capacity. The output layer consisted of a single neuron, responsible for predicting the binding affinity value.

The model was compiled using the Adam optimizer and the Mean Squared Error (MSE) loss function, which is suitable for regression tasks. Training was conducted over 50 epochs with a batch size of 32, and 20% of the dataset was reserved for validation to monitor the model's performance and prevent overfitting. The model's performance evaluation showed its effectiveness in predicting binding affinities. The Mean Squared Error (MSE) was 0.385, while the Mean Absolute Error (MAE) was 0.450. The R-squared value for the training set was 0.853, indicating a strong fit, while the test set showed a value of 0.632, demonstrating good generalization. These results highlight the model's ability to capture the underlying patterns in the data.

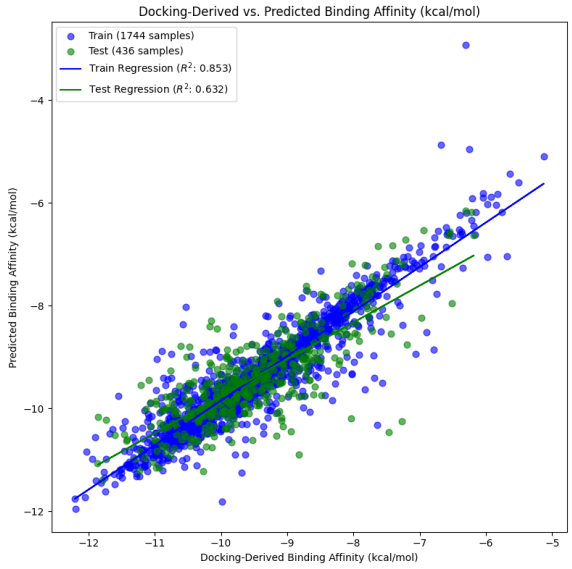

Figure 2. Docking-Derived vs. Predicted Binding Affinity $\left(\frac{\text{kcal}}{\text{mol}}\right)$ for PPAR Inhibitors.

For a visual assessment of the model's performance, a scatter plot comparing the true binding affinity values with the predicted binding affinities was generated. The plot included regression lines for both the training and test sets, further illustrating the model's ability to predict binding affinities accurately. After training, the model was saved in the H5 format (`ppar_binding_affinity_model.h5`) for future use, including potential deployment in further studies or clinical applications. A comparison

plot of docking-derived binding affinity versus predicted binding affinity for the PPAR inhibitors was also generated and saved as `ppar_2d_descriptors_plot.png` (Figure 2). The plot demonstrates reasonable agreement between the true docking-derived and predicted values, especially for the training set, suggesting the model captures general binding affinity trends for PPAR inhibitors.

Table 4. Comparison of predictive performance across models on the test set.

Model	MSE	MAE	R^2 (test)
Linear Regression	0.561	0.498	0.464
Ridge Regression	0.611	0.501	0.416
MLP (proposed)	**0.385**	**0.450**	**0.632**

7. Discussion

This study aimed to develop a deep learning model to predict the docking-derived binding affinity of ligands targeting the PPAR receptor family, based on 2D molecular descriptors derived from SMILES notation. Ligand data were sourced from the ChEMBL database, which provided IC_{50} values used for compound curation and selection. These ligands were then subjected to molecular docking using AutoDock Vina to obtain binding affinity scores $\left(\frac{\text{kcal}}{\text{mol}}\right)$, which served as the target variable for model training. This distinction ensures that the predicted endpoint is clearly defined as a computationally derived quantity, not a direct experimental measurement.

7.1. Biological significance of PPAR as a target

Peroxisome proliferator-activated receptors (PPARs) are nuclear receptors that play a pivotal role in the regulation of glucose and lipid metabolism, making them promising therapeutic targets for metabolic disorders, including type 2 diabetes and dyslipidemia [6, 47]. The ability to accurately predict the binding affinity of ligands for PPAR subtypes can facilitate the discovery of novel antidiabetic compounds, reducing the need for costly and time-consuming experimental screening.

7.2. Strengths of the dataset

The ChEMBL database is a widely trusted resource known for its comprehensive and high-quality bioactivity data, making it an excellent source for ligand information. The compounds chosen for this study had strong experimental validation, providing confidence in their interactions with the PPAR receptor. This enhanced the credibility of the predictions by grounding the results in reliable bioactivity data.

7.3. Challenges and data cleaning

While the ChEMBL dataset provided a solid foundation, significant preprocessing was required due to missing or invalid data for some ligands. After removing entries with missing IC_{50} values, 9,947 ligands were retained. Further filtering by potency threshold and deduplication reduced this to 3,764 and then 2,191 unique molecules, respectively. Following molecular docking with AutoDock Vina and filtering for binding affinity $\leq$ -5 $\frac{\text{kcal}}{\text{mol}}$, 2,180 ligands constituted the final dataset used for model training.

Although this cleaning step reduced the sample size, it was essential to prioritize data quality to avoid introducing bias or inaccuracies into the model. Despite the reduction, the resulting dataset maintained higher reliability for training predictive models.

7.4. 2D descriptors as molecular representations

In this study, 2D molecular descriptors were employed to represent the ligands' molecular structures. Unlike more complex representations such as 3D conformations or graph-based molecular

fingerprints, 2D descriptors offer a simplified yet effective means to capture key molecular features [34]. These descriptors are widely used in cheminformatics, facilitating molecular similarity searches and feature extraction for machine learning models.

The conversion of SMILES representations into 2D descriptors enabled the generation of a compact and informative feature set for model training. Although 2D descriptors efficiently capture structural information, they may not fully account for conformational flexibility or 3D interactions essential for molecular recognition.

7.5. Molecular docking to predict binding affinities

To predict the binding affinities of the ligands for the PPAR receptor, molecular docking simulations using AutoDock Vina were conducted [43]. This tool is widely used in computational chemistry due to its efficiency and accuracy in predicting ligand-receptor interactions. The docking results provided predicted binding affinities, which were essential for training the predictive model.

Although molecular docking is valuable for affinity prediction, it is subject to limitations. The accuracy of docking results may be influenced by factors such as receptor structure quality, docking parameters, and the precision of the defined binding site. While receptor preparation was carefully performed in this study, real-world applications may encounter challenges in maintaining docking accuracy [46].

7.6. Model development and evaluation

The core objective of this study was to develop a deep learning model to predict the docking-derived binding affinity $\left(\frac{\text{kcal}}{\text{mol}}\right)$ of PPAR inhibitors. The model was trained using 2D descriptors as input features and normalized docking-derived binding affinity scores as the target variable. The deep learning architecture consisted of three hidden layers, each employing the ReLU activation function to capture complex relationships between molecular features and binding affinity [35].

Model performance was evaluated using key metrics, including R-squared (R^2), Mean Squared Error (MSE), and Mean Absolute Error (MAE), providing insight into both accuracy and reliability of the predictions.

7.7. Performance analysis and comparison with baseline models

The model achieved an R^2 value of 0.853 on the training set and 0.632 on the test set. The gap of 0.221 between training and test R^2 indicates mild overfitting, which is expected given the high input dimensionality (210 features) relative to the training set size (1,744 samples). This behavior is consistent with findings in similar QSAR/docking-score prediction studies, where deep neural networks trained on 2D descriptors tend to memorize training patterns to some degree [31]. Importantly, the test R^2 of 0.632 still demonstrates meaningful generalization, suggesting that the learned non-linear mapping captures genuine structure–activity trends rather than random noise.

To further contextualize these results, the MLP was compared against two linear baseline models — Linear Regression and Ridge Regression — trained on the same dataset under identical conditions (see Table 4). The MLP achieved a test R^2 of 0.632, substantially higher than Linear Regression (0.464) and Ridge Regression (0.416), while also yielding lower MSE (0.385 vs. 0.561 and 0.611) and MAE (0.450 vs. 0.498 and 0.501). This consistent superiority across all metrics confirms that the non-linear function approximation capability of the MLP is essential for modeling the complex, non-linear relationship between 2D molecular descriptors and docking-derived binding affinity in PPAR-ligand systems [31, 37].

The Mean Squared Error (MSE) of 0.385 $\frac{\text{kcal}^2}{\text{mol}^2}$ reflects residual prediction error, with sensitivity to outliers arising from structurally atypical ligands in the dataset. The Mean Absolute Error (MAE)

of 0.450 $\frac{\text{kcal}}{\text{mol}}$ is particularly informative: in the context of computational drug discovery, prediction errors below 1.0 $\frac{\text{kcal}}{\text{mol}}$ are generally considered acceptable for early-stage virtual screening, and errors below 0.5 $\frac{\text{kcal}}{\text{mol}}$ indicate good practical utility [48]. The MAE of 0.450 $\frac{\text{kcal}}{\text{mol}}$ therefore falls within a range that is meaningful for prioritizing candidate ligands, even if it is insufficient for precise free-energy ranking.

The high R^2 value for the training set, combined with the lower test R^2, is consistent with mild overfitting rather than a fundamental modeling failure. In contrast, the linear baseline models (Linear Regression $R^2 = 0.464$; Ridge $R^2 = 0.416$) show no such gap — their limited capacity prevents overfitting but also prevents them from capturing the non-linear relationships inherent in the data. This trade-off underscores the necessity of the MLP architecture for this task, while also pointing to the need for regularization strategies (e.g., dropout, early stopping) in future iterations to close the generalization gap.

7.8. Implications, limitations, and future directions

The findings of this study demonstrate that deep learning models based on 2D molecular descriptors can effectively approximate binding affinity predictions, offering a scalable and cost-efficient computational screening tool. However, limitations remain, particularly in handling complex molecular interactions and generalizing to novel chemical scaffolds.

Future research could explore the integration of graph-based molecular representations [34, 49], transfer learning approaches, and ensemble methods to enhance predictive accuracy. Furthermore, experimental validation of high-confidence predictions is essential to bridge computational insights with real-world drug discovery applications [33].

Overall, the application of AI-driven models holds significant promise for accelerating the early phases of drug discovery, including hit identification and lead optimization for PPAR-related therapeutic targets.

8. Conclusion

This study developed a deep learning model to predict the docking-derived binding affinity $\left(\frac{\text{kcal}}{\text{mol}}\right)$ of ligands targeting PPAR receptors using 2D molecular descriptors derived from SMILES notation. The dataset curation followed a clear pipeline: 15,161 compounds were initially retrieved from ChEMBL, reduced to 9,947 after removing invalid IC_{50} entries, further filtered to 3,764 by potency threshold, deduplicated to 2,191 unique molecules, docked using AutoDock Vina, and finally filtered to 2,180 ligands $\left(\text{binding affinity} \leq \text{-5 } \frac{\text{kcal}}{\text{mol}}\right)$ for model training. The model was trained on a curated dataset of ligands whose IC_{50} values from the ChEMBL database were used for selection and preprocessing. Molecular docking simulations using AutoDock Vina then provided the binding affinity scores $\left(\frac{\text{kcal}}{\text{mol}}\right)$ as the quantitative target variable for model training. The model achieved an R^2 of 0.853 on the training set and 0.632 on the test set, demonstrating good predictive performance and generalizability. Benchmarking against linear baseline models (Linear Regression: $R^2 = 0.464$; Ridge Regression: $R^2 = 0.416$) confirmed that the non-linear MLP architecture is essential for capturing the complex structure–activity relationships in this dataset. While some prediction errors remain, the approach shows promising applicability for ligand-based virtual screening in antidiabetic drug discovery. Future work will focus on optimizing molecular descriptor selection and addressing outliers to enhance accuracy and expand applicability in antidiabetic drug discovery.

Supplementary Information

Author Contributions. **La Ode Aman:** Conceptualization, Methodology, Software, Writing - Original Draft. **Widy Susanti Abdulkadir:** Resources, Investigation, Writing - Review & Editing. **Dizky Ramadani Putri Papeo:** Data Curation, Validation, Writing - Review & Editing. **Ariani Hutuba:** Visualization & Investigation. **Teti Sutriyati Tuloli:** Supervision, Project Administration, Writing - Review & Editing. **Mohamad Adam Mustapa:** Data Curation, Formal Analysis, Writing - Review & Editing. **Yuszda K Salimi:** Formal Analysis, Validation, Writing - Review & Editing. **Hamsidar Hasan:** Supervision, Visualization, Writing - Review & Editing5. **Arfan: Software, Validation, Writing - Review & Editing. Aiyi Asnawi:** Supervision, Visualization, Writing - Review & Editing.

Acknowledgements. We would like to express our sincere gratitude to the Fugaku Supercomputer facility for providing the essential computational resources that enabled the molecular docking simulations conducted in this study. The high-performance computing capabilities of Fugaku were crucial in efficiently calculating the binding affinities for ligands targeting the PPAR receptor family. This research would not have been possible without their generous support. We also extend our thanks to the ChEMBL database for supplying the valuable data that formed the cornerstone of this work.

Funding. This research was funded by the Non-Tax State Revenue (PNBP) of Universitas Negeri Gorontalo under contract number 632/UN47.D1/PT.01.03/2025, dated May 27, 2025 and research implementation decree number 981/UN47/HK.02/2025, dated May 27, 2025.

Conflict of interest. The authors declare no conflict of interest.

Data availability. The datasets generated and analyzed during this study are available from the corresponding author upon reasonable request. The processed dataset including 2D descriptors and binding affinities has been archived in the file ppar_docking.csv.

Abbreviations.

PPAR	:	Peroxisome Proliferator-Activated Receptor
MLP	:	Multi-Layer Perceptron
QSAR	:	Quantitative Structure-Activity Relationship
AI	:	Artificial Intelligence
ML	:	Machine Learning
DL	:	Deep Learning
IC50	:	Half-maximal inhibitory concentration
MSE	:	Mean Squared Error
MAE	:	Mean Absolute Error
R2	:	Coefficient of Determination
SMILES	:	Simplified Molecular Input Line Entry System
PDB	:	Protein Data Bank
RDKit	:	Open-source cheminformatics toolkit
nM	:	Nanomolar
$\frac{\text{kcal}}{\text{mol}}$	:	Kilocalories per mole

References

[1] Ahmed I, El Turk S, Al Ghaferi A, Samad YA, Butt H. Nanocomposite Hydrogel-Based Optical Fiber Probe for Continuous Glucose Sensing. Small Science. 2024 feb;4(2). doi:10.1002/smsc.202300189.

[2] Ebere Ezinwanne Ilodibe, CU Nwankwo. Nurses' competencies and resource availability in the care of diabetes mellitus patient attending primary health care Centres in Anambra State. GSC Advanced Research and Reviews. 2023 feb;14(2):007-21. doi:10.30574/gscarr.2023.14.2.0038.

[3] Sahoo BR, Mohapatra G, Mohapatra J, Mohapatra N. Assessment of prevalence and pattern of comorbidities in hospitalized patients with uncontrolled hyperglycemia in Western Odisha. Multidisciplinary Science Journal. 2024 jul;6(3):2024015. doi:10.31893/MULTISCIENCE.2024015.

[4] IDF 11th. IDF Diabetes Atlas 11th Edition. In IDF Diabetes Atlas. vol. 11th editi. Brussels, Belgium: International Diabetes Federation; 2025.

[5] Haw JS, Galaviz KI, Straus AN, Kowalski AJ, Magee MJ, Weber MB, et al. Long-term sustainability of diabetes pre-

vention approaches: A systematic review and meta-analysis of randomized clinical trials. JAMA Internal Medicine. 2017;177(12):1808-17. doi:10.1001/jamainternmed.2017.6040.
[6] Hassan FU, Nadeem A, Li Z, Javed M, Liu Q, Azhar J, et al. Role of peroxisome proliferator-activated receptors (Ppars) in energy homeostasis of dairy animals: Exploiting their modulation through nutrigenomic interventions. International Journal of Molecular Sciences. 2021 nov;22(22):12463. doi:10.3390/ijms222212463.
[7] Setia M, Meena K, Madaan A, Srikanth N, Dhiman KS, Sastry J. In vitro Studies on Antidiabetic Potential of New Dosage Forms of AYUSH 82. Journal of Drug Research in Ayurvedic Sciences. 2017;2(1):1-9. doi:10.5005/jp-journals-10059-0001.
[8] Laganà AS, Vitale SG, Nigro A, Sofo V, Salmeri FM, Rossetti P, et al. Pleiotropic actions of peroxisome proliferator-activated receptors (PPARs) in dysregulated metabolic homeostasis, inflammation and cancer: Current evidence and future perspectives. International Journal of Molecular Sciences. 2016;17(7):999. doi:10.3390/ijms17070999.
[9] Zhao Y, Gao P, Sun F, Li Q, Chen J, Yu H, et al. Sodium Intake Regulates Glucose Homeostasis through the PPARδ/Adiponectin-Mediated SGLT2 Pathway. Cell Metabolism. 2016;23(4):699-711. doi:10.1016/j.cmet.2016.02.019.
[10] Peng L, Yang H, Ye Y, Ma Z, Kuhn C, Rahmeh M, et al. Role of peroxisome proliferator-activated receptors (Ppars) in trophoblast functions. International Journal of Molecular Sciences. 2021;22(1):1-13. doi:10.3390/ijms22010433.
[11] Przybycień P, Gąsior-Perczak D, Placha W. Cannabinoids and PPAR Ligands: The Future in Treatment of Polycystic Ovary Syndrome Women with Obesity and Reduced Fertility. Cells. 2022;11(16):2569. doi:10.3390/cells11162569.
[12] Holm LJ, Mønsted MØ, Haupt-Jorgensen M, Buschard K. PPARs and the development of type 1 diabetes. PPAR Research. 2020;2020:1-11. doi:10.1155/2020/6198628.
[13] Chen M, Lin W, Ye R, Yi J, Zhao Z. PPARβ/δ Agonist Alleviates Diabetic Osteoporosis via Regulating M1/M2 Macrophage Polarization. Frontiers in Cell and Developmental Biology. 2021;9. doi:10.3389/fcell.2021.753194.
[14] Singh S, Kumar R, Payra S, Singh SK. Artificial Intelligence and Machine Learning in Pharmacological Research: Bridging the Gap Between Data and Drug Discovery. Cureus. 2023. doi:10.7759/cureus.44359.
[15] Mariam Z, Niazi SK, Magoola M. Optimizing Lead Compounds: The Role of Artificial Intelligence in Drug Discovery. Preprints. 2024:1-9. doi:10.20944/preprints202404.0055.v1.
[16] Unogwu OJ, Ike ME, Joktan OO. Employing Artificial Intelligence Methods in Drug Development: A New Era in Medicine. Mesopotamian Journal of Artificial Intelligence in Healthcare. 2023;2023:52-6. doi:10.58496/MJAIH/2023/010.
[17] Vora LK, Gholap AD, Jetha K, Thakur RRS, Solanki HK, Chavda VP. Artificial Intelligence in Pharmaceutical Technology and Drug Delivery Design. Pharmaceutics. 2023;15(7):1916. doi:10.3390/pharmaceutics15071916.
[18] Meli R, Morris GM, Biggin PC. Scoring Functions for Protein-Ligand Binding Affinity Prediction Using Structure-based Deep Learning: A Review. Frontiers in Bioinformatics. 2022;2. doi:10.3389/fbinf.2022.885983.
[19] Shen C, Ding J, Wang Z, Cao D, Ding X, Hou T. From machine learning to deep learning: Advances in scoring functions for protein–ligand docking. Wiley Interdisciplinary Reviews: Computational Molecular Science. 2020;10(1):e1429. doi:10.1002/wcms.1429.
[20] Lee J, Yoon H, Lee YJ, Kim TY, Bahn G, Kim YH, et al. Drug–Target Interaction Deep Learning-Based Model Identifies the Flavonoid Troxerutin as a Candidate TRPV1 Antagonist. Applied Sciences (Switzerland). 2023;13(9):5617. doi:10.3390/app13095617.
[21] Shimizu Y, Ohta M, Ishida S, Terayama K, Osawa M, Honma T, et al. AI-driven molecular generation of not-patented pharmaceutical compounds using world open patent data. Journal of Cheminformatics. 2023;15(1). doi:10.1186/s13321-023-00791-z.
[22] Ridho M, Bustamam A, Adnan R. Reconstruction of the Phi-2 Method for Question-Answering Related to Diabetes Disease Using the MedAlpaca Dataset. Jambura Journal of Biomathematics (JJBM). 2025;6(3):183-7. doi:10.37905/jjbm.v6i3.30506.
[23] Adekunle TA, Ogundoyin IK, Akanbi CO. Machine Learning Model for Predicting the Temporal Lassa Fever Confirmed Cases in Nigeria. Jambura Journal of Biomathematics (JJBM). 2025;6(3):166-72. doi:10.37905/jjbm.v6i3.33831.
[24] Jiang X, Lu L, Li J, Jiang J, Zhang J, Zhou S, et al. Synthetically Feasible De Novo Molecular Design of Leads Based on a Reinforcement Learning Model: AI-Assisted Discovery of an Anti-IBD Lead Targeting CXCR4. Journal of Medicinal Chemistry. 2024;67(12):10057-75. doi:10.1021/acs.jmedchem.4c00184.
[25] Rafeeq MM, Sain ZM, Alturki NA, Alzamami A, Asiri SA, Mashraqi MM, et al. Computational Screening of Natural Compounds for the Discovery of Potential Aromatase Inhibitors: A Promising Therapy for Estrogen-Dependent Breast Cancer. Journal of Pharmaceutical Research International. 2021:72-8. doi:10.9734/jpri/2021/v33i32a31717.
[26] Stjernschantz E, Oostenbrink C. Improved ligand-protein binding affinity predictions using multiple binding modes. Biophysical Journal. 2010;98(11):2682-91. doi:10.1016/j.bpj.2010.02.034.
[27] Erdas-Cicek O, Atac AO, Gurkan-Alp AS, Buyukbingol E, Alpaslan FN. Three-Dimensional Analysis of Binding Sites for Predicting Binding Affinities in Drug Design. Journal of Chemical Information and Modeling. 2019;59(11):4654-62. doi:10.1021/acs.jcim.9b00206.
[28] Jiménez-Luna J, Pérez-Benito L, Martínez-Rosell G, Sciabola S, Torella R, Tresadern G, et al. DeltaDelta neural networks for lead optimization of small molecule potency. Chemical Science. 2019;10(47):10911-8.

doi:10.1039/c9sc04606b.
[29] Lecun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436-44. doi:10.1038/nature14539.
[30] Chen H, Engkvist O, Wang Y, Olivecrona M, Blaschke T. The rise of deep learning in drug discovery. Drug Discovery Today. 2018;23(6):1241-50. doi:10.1016/j.drudis.2018.01.039.
[31] Lenselink EB, Ten Dijke N, Bongers B, Papadatos G, Van Vlijmen HWT, Kowalczyk W, et al. Beyond the hype: deep neural networks outperform established methods using a ChEMBL bioactivity benchmark set. Journal of Cheminformatics. 2017;9(1):45. doi:10.1186/s13321-017-0232-0.
[32] Wieder O, Kohlbacher S, Kuenemann M, Garon A, Ducrot P, Seidel T, et al. A compact review of molecular property prediction with graph neural networks. Drug Discovery Today: Technologies. 2020;37:1-12. doi:10.1016/j.ddtec.2020.11.009.
[33] Schneider G, Clark DE. Automated De Novo Drug Design: Are We Nearly There Yet? Angewandte Chemie - International Edition. 2019;58(32):10792-803. doi:10.1002/anie.201814681.
[34] Jiang D, Wu Z, Hsieh CY, Chen G, Liao B, Wang Z, et al. Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. Journal of Cheminformatics. 2021 feb;13(1):12. doi:10.1186/s13321-020-00479-8.
[35] Nair V, Hinton GE. Rectified linear units improve Restricted Boltzmann machines. In: ICML 2010 - Proceedings, 27th International Conference on Machine Learning; 2010. p. 807-14.
[36] Kingma DP, Ba JL. Adam: A method for stochastic optimization. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. 2015.
[37] Stokes JM, Yang K, Swanson K, Jin W, Cubillos-Ruiz A, Donghia NM, et al. A Deep Learning Approach to Antibiotic Discovery. Cell. 2020;180(4):688-702.e13. doi:10.1016/j.cell.2020.01.021.
[38] Landrum G. RDKit: Open-source cheminformatics. RDKit. 2013.
[39] O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR. Open Babel: An Open chemical toolbox. Journal of Cheminformatics. 2011;3(10):33. doi:10.1186/1758-2946-3-33.
[40] Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: A system for large-scale machine learning. Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016. 2016:265-83.
[41] Buitinck L, Louppe G, Blondel M, Pedregosa F, Mueller A, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12(85):2825-30.
[42] McKinney W. Data Structures for Statistical Computing in Python. In: Proceedings of the 9th Python in Science Conference; 2010. p. 56-61. doi:10.25080/majora-92bf1922-00a.
[43] Trott O, Olson AJ. AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of Computational Chemistry. 2010;31(2):455-61. doi:10.1002/jcc.21334.
[44] Hunter JD. Matplotlib: A 2D graphics environment. Computing in Science and Engineering. 2007;9(3):90-5. doi:10.1109/MCSE.2007.55.
[45] Morris GM, Ruth H, Lindstrom W, Sanner MF, Belew RK, Goodsell DS, et al. Software news and updates AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility. Journal of Computational Chemistry. 2009;30(16):2785-91. doi:10.1002/jcc.21256.
[46] Oyama T, Takiguchi K, Miyachi H. Crystal structures of the ligand-binding domain of human peroxisome proliferator-activated receptor δ in complexes with phenylpropanoic acid derivatives and a pyridine carboxylic acid derivative. Acta Crystallographica Section F: Structural Biology Communications. 2022 feb;78(2):81-7. doi:10.1107/S2053230X22000449.
[47] Tyagi P, Singh HM, Ghosh B, Biswas SK. Virtual screening for plant-based inhibitors targeting SARS-CoV papain-like protease. Journal of Biomolecular Structure and Dynamics. 2011;29(4):634-49. doi:10.1080/07391102.2011.10508572.
[48] Ain QU, Aleksandrova A, Roessler FD, Ballester PJ. Machine-learning scoring functions to improve structure-based binding affinity prediction and virtual screening. Wiley Interdisciplinary Reviews: Computational Molecular Science. 2015;5(6):405-24. doi:10.1002/wcms.1225.
[49] Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, et al. MoleculeNet: A benchmark for molecular machine learning. Chemical Science. 2018;9(2):513-30. doi:10.1039/c7sc02664a.

A. Target Search Python Code

```
import os
import pandas as pd
from chembl_webresource_client.new_client import new_client

def search_targets(query_list, job_dir):
    for search_term in query_list:
        target = new_client.target
```

```
        try:
            target_query = target.search(search_term)

            if target_query:
                targets = pd.DataFrame.from_dict(target_query)
                search_term_formatted = search_term.replace(" ", "-")
                selected_columns = ['target_chembl_id', 'pref_name']

                targets['search_term'] = search_term_formatted
                targets = targets[selected_columns + ['search_term']]

                os.makedirs(job_dir, exist_ok=True)
                output_file = os.path.join(job_dir, f'targets_{search_term_formatted}.csv')
                targets.to_csv(output_file, index=False)

                print(f"Results for '{search_term}' saved to {output_file}")
            else:
                print(f"No results found for '{search_term}'")

        except Exception as e:
            print(f"Error while searching for '{search_term}': {str(e)}")

def main():
    query_list = ['alpha_glucosidase']
    job_dir = os.getcwd()
    search_targets(query_list, job_dir)

if __name__ == "__main__":
    main()
```

B. Molecule Retrieval Python Code

```
import pandas as pd
import numpy as np
import os
from chembl_webresource_client.new_client import new_client
from rdkit import Chem
from tqdm import tqdm
from concurrent.futures import ProcessPoolExecutor
from glob import glob

# Function to validate SMILES
def validate_smiles(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        return Chem.MolToSmiles(mol)
    else:
        return None

# Function to combine multiple CSV files into one
def combine_csv_files(target, target_dir):
    csv_files = glob(os.path.join(target_dir, '*.csv'))

    if not csv_files:
        print("No CSV files found in the directory.")
        return None

    dfs = [pd.read_csv(file) for file in csv_files if os.path.getsize(file) > 0]

    if not dfs:
```

```
        print(f"No data in CSV files in {target_dir}.")
        return None

    combined_df = pd.concat(dfs, ignore_index=True)
    output_file = os.path.join(target_dir, f'{target}.csv')
    combined_df.to_csv(output_file, index=False)
    print(f"Data from {target_dir} saved to {output_file}")
    return combined_df

# Function to process individual ChEMBL ID
def process_chembl_id(chembl_id, target, target_dir, other, others_column):
    try:
        activity = new_client.activity
        res = activity.filter(target_chembl_id=chembl_id).filter(standard_type="IC50")
        df = pd.DataFrame.from_dict(res)

        if df.empty:
            return f"No data for {chembl_id}"

        mol_cid = df.get('molecule_chembl_id', [])
        smiles = df.get('canonical_smiles', [])
        std_value = df.get('standard_value', [])

        data = list(zip(mol_cid, smiles, std_value))
        df = pd.DataFrame(data, columns=['molecule_chembl_id', 'canonical_smiles',
        'standard_value'])

        df = df.dropna(subset=['molecule_chembl_id', 'canonical_smiles'])
        df['canonical_smiles'] = df['canonical_smiles'].apply(validate_smiles)
        df = df.drop_duplicates(subset=['molecule_chembl_id'], keep='first',
        ignore_index=True)

        for col in others_column:
            df[col] = other[col]

        df['search_term'] = target

        output_path = os.path.join(target_dir, f'{chembl_id}.csv')
        df.to_csv(output_path, index=False)

        return f"Processed {chembl_id}"

    except Exception as e:
        return f"Error processing {chembl_id}: {str(e)}"

# Main function to search molecules
def search_molecules(job_dir):
    np.random.seed(1)
    all_dataframes = []
    targets = []

    target_files = glob(os.path.join(job_dir, 'targets_*.csv'))

    for file_path in target_files:
        df_targets = pd.read_csv(file_path)
        chembl_ids = df_targets['target_chembl_id']
        other_cols = df_targets.columns.difference(['t']()
```

C. Data Cleaning Python Code

```
import pandas as pd

def clean_combined_data(input_file, output_file):
    df = pd.read_csv(input_file)

    if 'standard_value' in df.columns:
        df['standard_value'] = pd.to_numeric(df['standard_value'], errors='coerce')
        df = df[df['standard_value'].notna() & (df['standard_value'] != 0)]

    df = df.sort_values('standard_value').drop_duplicates(subset='molecule_chembl_id',
    keep='first')

    columns_to_drop = ['pref_name', 'search_term']
    df = df.drop(columns=[col for col in columns_to_drop if col in df.columns])

    df.to_csv(output_file, index=False)
    print(f"Cleaned data saved to: {output_file}")
    return output_file

if __name__ == "__main__":
    input_file = "ppar.csv"
    output_file = "ppar_cleaned.csv"
    clean_combined_data(input_file, output_file)
```

D. Automated Docking Python Code

```
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import AllChem
from subprocess import run, DEVNULL, CalledProcessError
from tqdm import tqdm
from concurrent.futures import ProcessPoolExecutor

def smiles_to_pdbqt(smiles, ligand_name):
    """
    Convert SMILES to PDBQT format using RDKit and Open Babel.
    """
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol is None:
            raise ValueError(f"Invalid SMILES: {smiles}")

        mol = Chem.AddHs(mol)
        AllChem.EmbedMolecule(mol, AllChem.ETKDG())
        AllChem.UFFOptimizeMolecule(mol)

        pdb_path = f"{ligand_name}.pdb"
        Chem.MolToPDBFile(mol, pdb_path)

        pdbqt_path = f"{ligand_name}.pdbqt"
        run(["obabel", pdb_path, "-O", pdbqt_path, "--gen3d"], stdout=DEVNULL,
        stderr=DEVNULL, check=True)

        os.remove(pdb_path)
        return pdbqt_path
    except (ValueError, CalledProcessError, Exception) as e:
        print(f"Error converting {ligand_name} to PDBQT: {e}")
        return None
```

```
def run_vina_docking(receptor, config_file, ligand_pdbqt, ligand_name):
    """
    Perform molecular docking using AutoDock Vina.
    """
    try:
        output_pdbqt = f"{ligand_name}_out.pdbqt"
        run([
            "vina",
            "--receptor", receptor,
            "--ligand", ligand_pdbqt,
            "--config", config_file,
            "--out", output_pdbqt
        ], stdout=DEVNULL, stderr=DEVNULL, check=True)
        return output_pdbqt
    except CalledProcessError as e:
        print(f"Vina docking failed for {ligand_name}: {e}")
        return None

def extract_best_binding_affinity(output_pdbqt):
    """
    Extract the best binding affinity value from Vina output file.
    """
    try:
        with open(output_pdbqt, 'r') as file:
            for line in file:
                if "REMARK VINA RESULT" in line:
                    return float(line.split()[3])
    except Exception as e:
        print(f"Error reading output file {output_pdbqt}: {e}")
    return None

def process_ligand(row, receptor, config_file):
    """
    Process a single ligand: convert to PDBQT, dock, extract binding affinity.
    """
    smiles = row['canonical_smiles']
    molecule_id = row['molecule_chembl_id']
    standard_value = row['standard_value']
    ligand_name = f"ligand_{molecule_id}"

    ligand_pdbqt = smiles_to_pdbqt(smiles, ligand_name)
    if not ligand_pdbqt:
        return molecule_id, smiles, standard_value, None

    output_pdbqt = run_vina_docking(receptor, config_file, ligand_pdbqt,
    ligand_name)
    best_affinity = None
    if output_pdbqt:
        best_affinity = extract_best_binding_affinity(output_pdbqt)
        os.remove(output_pdbqt)

    if os.path.exists(ligand_pdbqt):
        os.remove(ligand_pdbqt)

    return molecule_id, smiles, standard_value, best_affinity

def perform_docking(input_file, receptor, config_file, output_csv):
    """
    Perform docking for multiple ligands and save results to CSV.
```

```
    """
    if not os.path.exists(receptor):
        print(f"Receptor file not found: {receptor}")
        return

    if not os.path.exists(config_file):
        print(f"Config file not found: {config_file}")
        return

    input_df = pd.read_csv(input_file)
    required_columns = ['canonical_smiles', 'molecule_chembl_id',
    'standard_value']
    if not all(col in input_df.columns for col in required_columns):
        print("Required columns missing in input file.")
        return

    if not os.path.exists(output_csv):
        with open(output_csv, 'w') as f:
            f.write("molecule_chembl_id,canonical_smiles,standard_value,
            binding_affinity\n")

    rows = input_df.to_dict('records')
    with ProcessPoolExecutor(max_workers=1) as executor:
        futures = [executor.submit(process_ligand, row, receptor, config_file)
        for row in rows]

        for future in tqdm(futures, total=len(rows), desc="Docking ligands"):
            try:
                result = future.result()
                if result:
                    molecule_id, smiles, standard_value, best_affinity = result
                    with open(output_csv, 'a') as f:
                        f.write(f"{molecule_id},{smiles},{standard_value},
                        {best_affinity}\n")
            except Exception as e:
                print(f"Error during docking: {e}")

    print(f"Docking results saved to {output_csv}")

if __name__ == "__main__":
    input_file = "ppar_cleaned.csv"
    receptor = "ppar.pdbqt"
    config_file = "config.txt"
    output_csv = "ppar_docking.csv"
    perform_docking(input_file, receptor, config_file, output_csv)
```

E. 2D Molecular Descriptor Generation Code

```
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

# Load the dataset
df = pd.read_csv('ppar_docking.csv')

def get_2d_descriptors(mol, missing_val=None):

    descriptors_2d = {}

    if mol is not None:
        try:
```

```
            for desc_name, fn in Descriptors.descList:
                val = fn(mol)
                descriptors_2d[desc_name] = val
        except Exception:
            descriptors_2d = {desc_name: missing_val for desc_name,
            _ in Descriptors.descList}
    else:
        descriptors_2d = {desc_name: missing_val for desc_name,
        _ in Descriptors.descList}

    return pd.DataFrame([descriptors_2d])

def generate_2d_descriptors(smiles):

    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return get_2d_descriptors(mol)
    else:
        return None

# Generate descriptors for each molecule
df_descriptors = df['canonical_smiles'].apply(generate_2d_descriptors)

# Filter out None results and concatenate into a single DataFrame
descriptors_df = pd.concat([desc for desc in df_descriptors if desc is not
None], ignore_index=True)

# Combine original data with the descriptors
df_combined = pd.concat([df.reset_index(drop=True),
descriptors_df.reset_index(drop=True)], axis=1)

# Save the combined dataset
df_combined.to_csv('ppar_2d_descriptors.csv', index=False)

# Display first few rows
print(df_combined.head())
```

F. Deep Learning Model for Binding Affinity Prediction

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
import tensorflow as tf
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Load the dataset
df = pd.read_csv('ppar_2d_descriptors.csv')

# Filter rows where binding_affinity <= -5
df_filtered = df[df['binding_affinity'] <= -5]

# Drop non-feature columns
X = df_filtered.drop(columns=['molecule_chembl_id', 'canonical_smiles',
'standard_value', 'binding_affinity'])

# Handle missing values
X = X.dropna(axis=1)
```

```
# Extract target
y = df_filtered['binding_affinity'].values
y = np.nan_to_num(y)

# Check length
assert len(X) == len(y), f"Length mismatch: X({len(X)}), y({len(y)})"

# Normalize features and target
scaler_X = StandardScaler()
X_scaled = scaler_X.fit_transform(X)

scaler_y = StandardScaler()
y_scaled = scaler_y.fit_transform(y.reshape(-1, 1)).flatten()

# Split data
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled,
test_size=0.2, random_state=42)

print(f'Training samples: {len(X_train)}')
print(f'Test samples: {len(X_test)}')

# Build model
model = models.Sequential([
    layers.InputLayer(input_shape=(X_train.shape[1],)),
    layers.Dense(128, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(1)
])

model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mae'])

# Train model
history = model.fit(X_train, y_train, epochs=50, batch_size=32,
validation_split=0.2, verbose=1)

# Predict
y_pred_train_scaled = model.predict(X_train).flatten()
y_pred_test_scaled = model.predict(X_test).flatten()

# Inverse transform
y_train_orig = scaler_y.inverse_transform(y_train.reshape(-1, 1)).flatten()
y_test_orig = scaler_y.inverse_transform(y_test.reshape(-1, 1)).flatten()
y_pred_train = scaler_y.inverse_transform(y_pred_train_scaled.reshape(-1, 1))
.flatten()
y_pred_test = scaler_y.inverse_transform(y_pred_test_scaled.reshape(-1, 1))
.flatten()

# Metrics
r2_train = r2_score(y_train_orig, y_pred_train)
r2_test = r2_score(y_test_orig, y_pred_test)
mse = mean_squared_error(y_test_orig, y_pred_test)
mae = np.mean(np.abs(y_test_orig - y_pred_test))

print(f"Test MSE: {mse:.4f}")
print(f"Test MAE: {mae:.4f}")
print(f"Training R2: {r2_train:.3f}")
print(f"Test R2: {r2_test:.3f}")
```

```
# Save model
model.save('ppar_2d_descriptors_model.keras')

# Regression lines
train_lr = LinearRegression().fit(y_train_orig.reshape(-1, 1), y_pred_train)
test_lr = LinearRegression().fit(y_test_orig.reshape(-1, 1), y_pred_test)

train_line = train_lr.predict(y_train_orig.reshape(-1, 1))
test_line = test_lr.predict(y_test_orig.reshape(-1, 1))

# Plot
plt.figure(figsize=(8, 8))
plt.scatter(y_train_orig, y_pred_train, color='blue', alpha=0.6,
label=f'Train ({len(X_train)} samples)', s=40)
plt.scatter(y_test_orig, y_pred_test, color='green', alpha=0.6,
label=f'Test ({len(X_test)} samples)', s=40)
plt.plot(y_train_orig, train_line, color='blue', linestyle='-',
label=f'Train Regression ($R^2$: {r2_train:.3f})')
plt.plot(y_test_orig, test_line, color='green', linestyle='-',
label=f'Test Regression ($R^2$: {r2_test:.3f})')
plt.title('Docking-Derived vs. Predicted Binding Affinity (kcal/mol)')
plt.xlabel('Docking-Derived Binding Affinity (kcal/mol)')
plt.ylabel('Predicted Binding Affinity (kcal/mol)')
plt.legend()
plt.tight_layout()
plt.savefig('ppar_2d_descriptors_plot.png')
plt.show()
```