University of Mohamed Boudiaf Msila

Department of Computer Science

Master Dissertation

Drug Design and Discovery using Artificial Intelligence

Youssouf Benyettou

Dr. LAMRIS Sayad

2018/2019

Abstract

With the rise of deep learning models and the successful result showing in different domains (such as Computer vision and Natural language processing) researchers and laboratories of cheminformatics try to apply these techniques in drug design and discovery. recently, the application of Deep Learning in this area of research has made a good progress but it is in the early stage and we can't say that the results lead us to rational drug design, which mean designing new drugs without *in vivo* and human trials. in this master thesis project, we applied Deep Neural network (Deep learning) on drug design and discovery dataset to predict the toxicity of molecules, after the evaluation and comparison of results we found that the big problem is the small amount of data compared with used model.

Table of Contents

Abstract
Table of figures IV
List of tables V
List of Abbreviations V
Introduction 7
Overview 7
Methodology 7
Thesis structure 8
Chapter 01 10
Drug discovery and Computer Aided Drug Design 10
Introduction 10
1. An overview of drug discovery and development 10
1.1. Drug discovery process. 10
2. Methodology 11
2.1. Quantitative structures-activity relationship (QSAR) 11
Conclusion 14
Chapter 02 16
Machine Learning and Deep Learning 16
Introduction 16
1. Type of Machine Learning 16
1.1. Supervised Learning 16
1.2. Unsupervised 17
1.3. Semi supervised learning 17
1.4. Reinforcement Learning 17
2. Building blocks of Machine learning Algorithms (supervised) 18
3. Mathematical formulation of Learning model: Linear regression as an example. 19
4. Neural Networks and Deep Learning 21
4.1. Concepts and mathematic formalism of Neural Network 21
CONCLUSION 23
Chapter 03 25
Implementing Deep Neural Network models on Tox21 Dataset. 25
1. Problem statement 25
2. Toxicology in the 21st Century (Tox21) program 26
1. Goals of the Tox21 Program 26
2. Tox21 dataset challenge 26
3. Development environment Programing language, Libraries and frameworks 27
4. Tools used calculate the descriptors of molecules 29

5. Implementing Deep learning model on Tox21 datasets 30
6. Results and Discussion 38
7. Discussion 39
Conclusion 41
Glossary 43
Index 44
Bibliography 45

Table of figures

Figure 1 Drug Discovery and Design Process 11
Figure 2 Molecular descriptors used in Quantitative structure Relationship (QSAR) [4] 12
Figure 3 Type of Machine Learning 18
Figure 4 Linear regression model for 1D 20
Figure 5 A Deep Neural network illustration , generated by https://playground.tensorflow.org/ 21
Figure 6 Graph of a fully connected neural network 21
Figure 7 Single Artificial Neuron 22
Figure 8: Molecular graph generated from Smile formula using AlvDesc 0.1 26
Figure 9 Package installation in Google CO Lab averment 30
Figure 10 Import Packages, Libraries and Frameworks 30
Figure 11 Install the PyDrive wrapper and import data from Google Drive 31
Figure 12 Import Tox21 dataset from the official web site 31
Figure 13 Loading the data from the SDF file 32
Figure 14 Exporting the data to Excel file 32
Figure 15 Loading the new Excel file and preparing the data 32
Figure 16 Descriptors data loading and preparation for the computation phase 33
Figure 17 Standardisation of Data 33
Figure 18 Some data cleansing and Validation set split 34
Figure 19 The shape of the data sets (train, validation, test) 34
Figure 20 Our Deep Neural Network Model (Layers, Neurons, Regularization, Loss function, Optimizer, traing configuration – epochs, batch size…) and showing the results of the training and and evaluation. 35
Figure 21 AUR_ROC Curve 37
Figure 22 Training and validation Loss curves 37
Figure 23 Accuracy Curve 38
Figure 24 Comparison of our results with MoleculeNet Benchmark 38

List of tables

Table 1 Common chemical descriptors for QSAR Analysis [5] 12
Table 2 Summary of prediction types 19
Table 3 Activation functions formula and their curves. 22
Table 4 Loss functions used in Neural Network 22
Table 5 Details of Drug design and Discovery Datasets[12] 25
Table 6 the first five raw of Tox21 Datasets, the red columns represents the twelve tasks of toxicology. 28
Table 7 our Last DNN Model configuration summary 35
Table 8 Comparison between our models results and the MoleculeNet Benchmark 38

List of Abbreviations

- **CADD**: Computer Aided Drug Design and Discovery
- **QSAR**: Quantitative Structure Activity Relationship
- **HTS**: High Throughput screening.
- **SMILES**: simplified molecular-input line-entry system.
- **ML**: Machine Learning.
- **DL**: Deep Learning.
- **SVM**: Support Vector Machine.
- **CNN**: Convolutional Neural Network.
- **RNN**: Recurrent Neural Network.
- **GAN**: Generative Adversarial Network.
- **TF**: TensorFlow.
- **AUC:** Area under curve.
- **ROC:** Receiver operating characteristic curve.
- **TPR:** True Positive Rate.
- **TPR:** False Positive Rate.

Introduction

Overview

The drug design and discovery have complicated phases and development processes. The discovery and development of new drugs can take up to 15 years and it cost billions of dollars, it starts by the idea of finding a new drug to treat new diseases where the lab researchers at universities or industries try to understand the symptoms and the disease mechanism.

in this phase there are many steps and process to follow in order to develop new chemical compounds that can interact with the targets, we will talk about it in detail in chapter 1. After the first phase of identifying the lead compounds, it comes preclinical phase which consists of testing these delivered candidates in two steps *in vitro* and *in vivo* before passing to human trials. this phase is the most important because it tries to find answers to the questions: **which drug can we try it on humans? And how this drug will affect the human body, is it toxic or not?** And many other questions.

When the candidate's compounds pass the preclinical tests, they enter another phase: the clinical test on humans which consists of three experimental steps or phases. In the first phase they try the new drugs on less than hundred volunteers' participants in a duration of a month to determine the compatibility of the human body with this drug and to find the maximum tolerated dose. In the second phase the number of volunteers increases to hundreds as well as the duration of the study, it can be months or years, the researcher here focus on the efficacy of and longer-term safety in disease population, they use randomized trials.e.g. Single blinded and double blinded experiments where the patients take a drug and they don't know if it is trial of not and in double blinded even the researchers don't have idea about the drug they are using. The trial phases are the difficult step in drug design because the volunteers risk their lives to test these drugs without any guarantee that made the scientific communities establish many rules and regulations in biomedical research. these problems triggered the researchers mind to find a solution to simulate these risky trials. With the development of statistical learning algorithms and their successful application results in different research area, the idea of rational drug design started where they found a method of modeling and designing new chemical compounds to predict their activities.

Methodology

Quantitative Structure Activity Relationship (QSAR) is the methodology that tries to predict the relationship between chemical compounds. i.e. predict the activity/propriety of given molecules as correlation function. the molecular descriptors are the mean key in this equation, they defined as independent variables which are used like inputs of prediction function of Molecular bioactivity.

QSAR methodology allows the researcher to apply the statistical learning models on the descriptors data i.e. the calculated descriptors can be used as a dataset and from this data set the Machine learning or Deep learning models can be applied.

In this project we have used different techniques (Mordred and Rdkit Library, AlvaDesc 0.1 software) to calculate the descriptors of given molecules dataset (Tox21) in the purpose of applying a deep Neural network models, evaluating and comparing the results with MoleculNet Benchmark.

Thesis structure

This thesis report is structured into three chapters. The first Chapter explains the Drug Discovery and design process, the Computational Aided Drug Design methods, ending by describing QSAR technique.

The second chapter focuses on the Machine Learning and Deep Learning Models, the chapter is designed to make the reader understand how the models built from scratch, we started by the type of machine learning after that we defined some concepts (type of prediction, type of learning models) before giving the mathematical formulation of the fundamental Algorithms, after that we talked about the Deep Neural Networks, in this subsection we focused on some definition and we made illustrations to give the reader some insight about Neural networks.

The third chapter is a report of implementing Deep Neural Network on dataset of Drug toxicity prediction(Tox21), this chapter started by the problem statement of this project and the solution that we suggest to solve it, followed by some definitions of tools and technologies, programing languages, libraries and frameworks used in this projects, after that we tried to explain the project code block by block, at the end, we presented the results and the comparison with the benchmark, followed by a discussion of the problems like overfitting, small amount of data and how to calculate more descriptors, data cleansing, future work and suggestion.

Chapter 01

Drug discovery and Computer Aided Drug Design

Introduction

In this chapter, we will focus on drug discovery and development, first of all, we will talk about the interaction of molecule in our body and how these interactions lead us to understand other functions of the body system, After that, we will explain the process of drug discovery and development. The second section, we will end the chapter with the QSAR methodology used in rational drug design.

1. An overview of drug discovery and development

Our body has complicated molecular reactions which perform different functions such as metabolism, defenses against bacteria and viruses or sending signals to different body part. The molecules in the cells of our body interact and affect each other to generate the protein or gene expression. Signaling pathway is the process of interaction/reaction of molecules. Any mistake happed during the signaling pathway lead to molecular disorders result for example mutations, cancer or diabetes. The drug molecules interact with some molecules that involved in the pathway to affect the disordered results, and sometimes restituting normal activities. Over the past decade scientist worked on finding new treatment for the new disease until they reach the modern Drug Discovery and Development technique in which has changed the way of treating disease and testing new drugs.

2. Drug discovery process

In this era, the drug discovery and development investment cost has passed $2.6 billion [1], and bringing a drug to the market takes more than 7 years, to identify a new drug it needs more than 100000 compounds as candidate, preclinical testing on hundreds of animals, and clinical trials on thousands of volunteer's patients. Only 1 of 10 compounds successes the clinical trials which means that the success rate is 0.001% of tested compounds [2].

The pipeline of drug discovery and development resume the whole process from the early identification of drug to the marketing. This pipeline has two major stages: Drug discovery (**Preclinical phase**) and drug development (**Clinical phase**). In drug discovery phase, there are three steps: **target identification**, **lead discovery**, and **lead optimization**. The first step, target identification, many experiments dedicated to the identification of a biological target, i.e. identifying the compound that have the ability to fix the activity of bio-target by performing ***in silico*** screens, next generation screening and structural analysis. The second step, **lead discovery**, is to identify a lead compound (molecule), which will exhibit drug-like property.

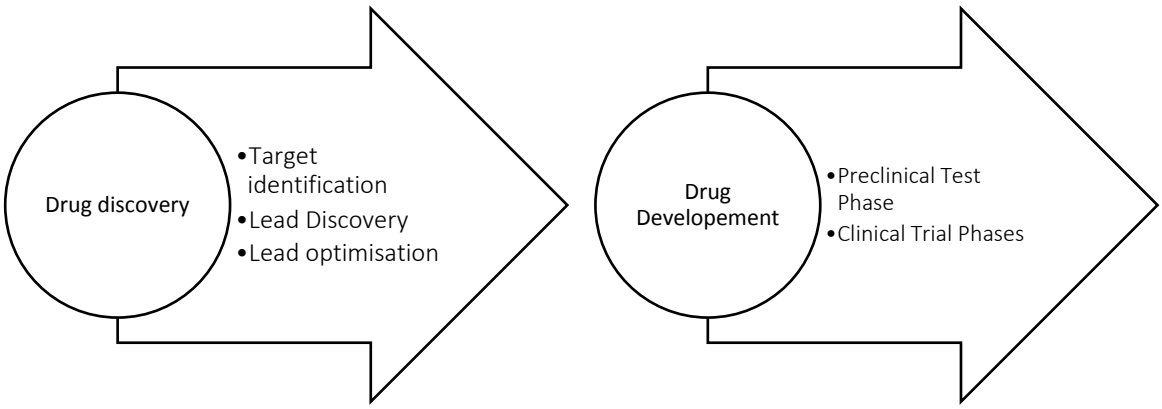


Figure 1 Drug Discovery and Design Process

The third step, the **lead optimization** is to modify and improve the identified molecules in previous step in order to take the **lead compounds**.

3. Methodology

In this section we will focus on Quantitative structures-activity relationship (QSAR), the chemical descriptors calculation and how these methods work with Statistical learning algorithm (Machine learning and Deep learning).

3.1.Quantitative structures-activity relationship (QSAR)

The development of techniques in chemistry tolerate the modification of features of molecules which is used in many fields of natural science especially in *in silico* phase. Quantitative Structure-Activity Relationship (QSAR) modeling is a technique that allows the discovery of new information on compounds covering many scientific aspects, these techniques can determine the reliable relations between variations in the values of calculated descriptors and the biological activity for a series of chemical molecules [3]. It allows the development of mathematical correlation between chemical features and the biological response. This technique is based on a mathematical algorithm which provides assumptions to build a predictive correlation model.

3.1.1. Molecular descriptors calculation

Chemical descriptors are numerical features extracted from chemical structures for molecular data mining, compound diversity analysis and compound activity prediction, QSAR analysis [4, 5]. There

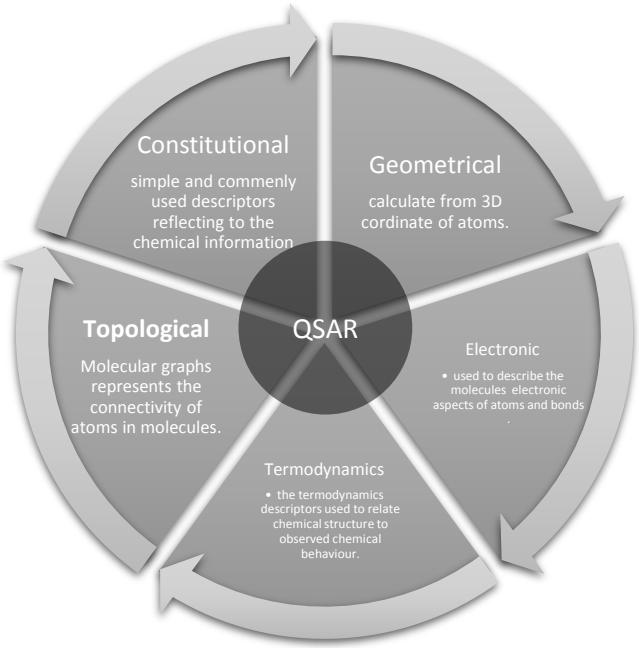


Figure 2 Molecular descriptors used in Quantitative structure Relationship (QSAR) [4]

are many categories of descriptors like Topological descriptors, electronic descriptors , geometrical … etc. see Figure 2 and they have different dimensions see **Error! Reference source not found.**. The mathematical representation of molecular descriptors allows building statistical learning model.

Chemical descriptors	Based on
Theoretical descriptors	
0D	Molecular formula
1D	Chemical graph
2D	Structural topology
3D	Structural geometry
4D	Chemical conformation

Table 1 Common chemical descriptors for QSAR Analysis [5]

3.1.2. Definition and formalism of QSAR/QSPR

QSAR modeling on a set of structurally related chemicals refers to the development of a mathematical correlation between a chemical response and quantitative chemical attributes defining the features of the analyzed molecules [6]. Other study accomplished a mathematical formalism between chemical behavior. The mathematical formalism of QSAR represented as follow:

$$Biologicalactivity = f(Chemical\ attributes) \quad 3.1)$$

Chemical attributes represent the features that define the behavior manifestation, i.e. they are the fundamental information of the chemicals which control the response under study [6]. These attributes are quantitative chemical information derived using empirical analysis or mathematical algorithms that diagnose chemistry of molecules. The behavioral manifestation of any chemical species is explained by its physicochemical properties. The chemical attributes in equation 3.1) can be described by the information derived from the chemical structure and physicochemical property. The experimental technique leads us to this expression:

$$Response = f(chemicalstructure, physicochemicalproperty) \quad 3.2)$$

The mathematical equation of QSAR:

$$Y = a_0 + a_1X_1 + a_2X_2 + a_3X_3 + \ldots + a_nX_n \quad (3.3)$$

- Y is the dependent variable that represents the chemical response (activity, toxicity, property) .
- $X_0, X_1, X_3, \ldots, X_n$ are the independent variables which denote the different features, physiochemical properties in the form of numerical quantities or descriptors.
- $a_0, a_1, a_2, \ldots, a_n$ the contribution of individual descriptors to the response.

3.1.3. QSAR process and model development.

Building a QSAR model need two necessary steps:

1- Description of molecular structure.
2- Multivariance analysis for correlating molecular descriptors with the observed activities/properties.

Between the two steps there are essential process should be applied on the data Figure 3 , these processes are the same process of Data preprocessing and preparation phase in any data analysis project.

- **Data preprocessing**
- **Data cleaning**
- **Data normalization**
- **Data standardization**

We will explain those process in Chapter 0 with the code applied on the Tox21 data.

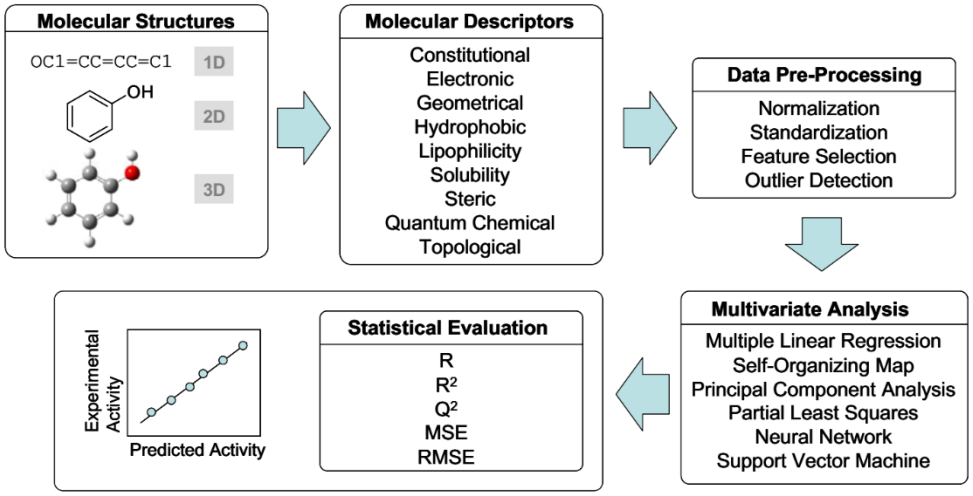


Figure 3 Schematic overview of the QSAR process [42]

Conclusion

In this chapter we tried to explain the drug discovery and design process and methodology (QSAR) with simple language in order to understand it from the reader, discovering a different scientific area is not easy and take a lot of time to understand, the key words of cheminformatics is different than the usual terms of computer science, that is why we tried to simplify the sentences as much as possible because we think that the readers of this dissertation are not chemists or specialists, in fact, there are many concepts we didn't talk about because we found that they don't have a direct relation with the project .

Chapter 02

Machine Learning and Deep Learning

Introduction

In this chapter, we will define Machine Learning and Deep Learning, the mathematic modeling of ML problem, some other concepts and finally we will focus on Deep neural networks (**DNN**).

Machine learning is a subfield of Artificial Intelligence which is referred to building algorithms or model related to a dataset, this data may be a collection of natural phenomena observation or hand-designed and sometimes generated by another Algorithm.

We can define it also in two steps:

1- Collecting dataset.
2- Building a statistical model for this data set

Other definitions:

> *"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed."*
>
> ***Arthur Samuel, 1959***

> *"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."*
>
> ***Tom Mitchell, 1997***

1. Type of Machine Learning

What do we mean by machine learning type?

In fact, we can't say that we have machine learning type, because there are many concepts in this area of research, some authors see the type of machine learning in the type of problems that these models try to solve, others statisticians see that the type of **ML** is related by the data distribution and representation, some other experts define the type of ML by its tasks type. In this section we will define the most common type of ML mentioned in literature [7].

1.1.Supervised Learning

In Supervised learning each example in the dataset has a label, we can define it like that:

$$\{(x_i, y_i)\}_i^N \quad 1.1)$$

x_i : is the feature vector in the dataset.

y_i : is the label given to this example, it can be a number or vector or another complex structure.

Supervised learning algorithm goal is building a learning model, that takes as input a set of examples X and deduce the outputs, which will be compared with labels Y.

The most used algorithm in supervised learning:

- K-nearest neighbors.
- Linear regression.
- Logistic regression.
- Support vector machine.
- Random forest.
- Neural networks.

1.2.Unsupervised

In unsupervised learning, we don't have labels for the examples in the dataset, the algorithm learns without help. There are many classes of algorithms in unsupervised learning, the most used are:

- Clustering Algorithms
- Dimensionality reduction and visualization algorithms.
- Association Rule learning algorithms

Unsupervised algorithm takes as input the feature vector X and produce an output vector; the value of this vector can be used to solve the given problem.

1.3.Semi supervised learning

In semi-supervised learning the dataset contains labeled and unlabeled data, usually, the amount of unlabeled data is higher than the labeled data. The semi-supervised learning algorithm goal is the same as supervised learning. The role of unlabeled data here to enhance the learning algorithm to produce a good model.

1.4.Reinforcement Learning

In reinforcement learning, the learning algorithm can observe the environment by take the state of the environment as vector of features. It can perform actions in each state, these actions give the algorithm rewards (negative or positive). The algorithm here learns by itself the best **Policy**, it is like

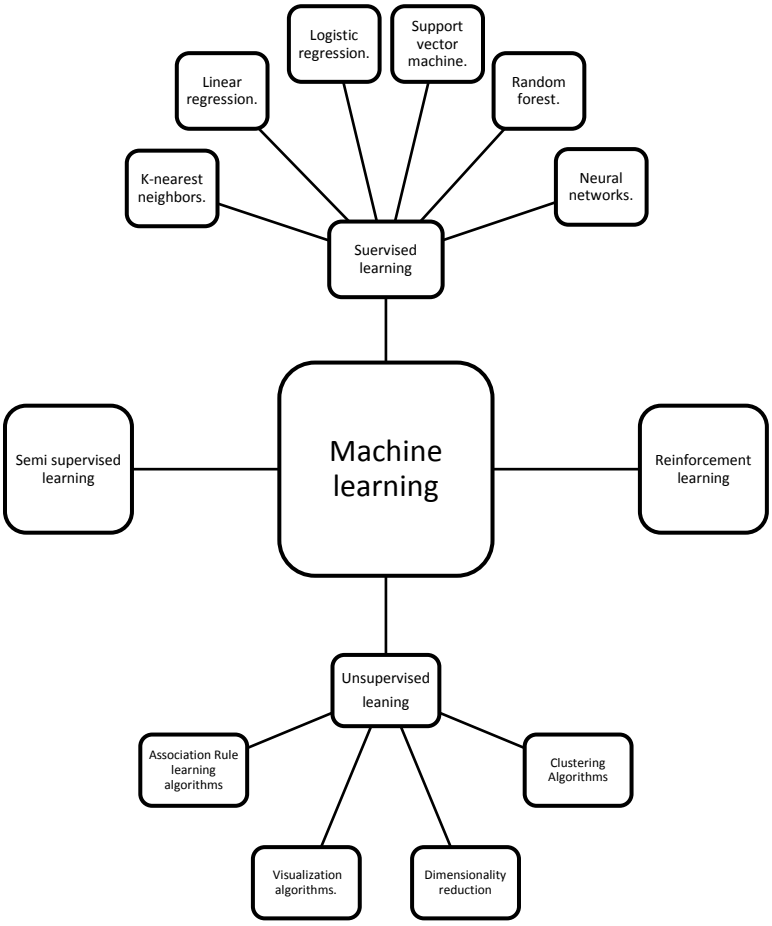


Figure 4 Type of Machine Learning

the model in Supervised learning, take as input a feature vector (state of environment) and give as output the optimal actions which maximize the reward.

2. Building blocks of Machine learning Algorithms (supervised)

In this section and for the rest of the chapter we will focus on the fundamental algorithms of machine learning and Deep neural networks in supervised learning only, we will define the mathematical model of learning problem using linear regression as an example, then we will pass to deep neural networks models, and before we start defining these algorithms, we need to explain some concepts.

Types of Prediction

We have two types of prediction in ML, these types are relied on the nature of outcomes (set of labels **Y**) data: **continuous** or **classes**, the prediction model called **regression** model when it deals with continuous data, and the data contains classes needs a **classifier** model [7].

	Regression	Classifier
Outcomes	Continuous	Class
Examples	Linear regression	Logistic regression, SVM

Table 2 Summary of prediction types

Type of learning models

the type of learning model can be defined by the estimation used to solve the given problem (data). We briefly define these two types below:

Discriminative Models

The discriminative model learns directly $P(y|x)$, and the result of learned is a Decision boundary. Example of this model: Regression, SVM [8].

Generative model

The generative model estimates $P(x|y)$ to deduce $P(y|x)$, the result given by this model is a probability distribution of Data, e.g. Generative adversarial Network (GAN), Naïve Bayes [8].

3. Mathematical formulation of Learning model: Linear regression as an example.

Linear regression algorithm is the popular regression algorithm, because everyone studied machine learning should have started with it. It is an algorithm that learns a linear combination of features of input example.

Problem statement

The data represented as a collection of labeled examples:

$$\left\{\left(\boldsymbol{x}_{\boldsymbol{i}}^{\boldsymbol{j}}, \boldsymbol{y}_{\boldsymbol{i}}\right)\right\}_{\boldsymbol{i}=\overline{\boldsymbol{1},\boldsymbol{N}};\, \boldsymbol{j}=\overline{\boldsymbol{1},\boldsymbol{D}}} \tag{2.1}$$

N is the size of the collection.

$\boldsymbol{x_i}$: is a D-dimensional feature vector of example $i = \overline{1, N}$, in the dataset.

$\boldsymbol{y_i}$: is the label given to this example or the real valued target.

In linear regression, the goal is to build a model $f_{\mathbf{w},b}(\mathbf{x})$ as linear combinations of features:

$$f\boldsymbol{w}, b(\boldsymbol{x}) = \boldsymbol{wx} + b \tag{2.2}$$

$\boldsymbol{W}$: D-dimensional vector of parameters

$\boldsymbol{b}$: bias, a real number

The model is used to predict $\boldsymbol{y}$ of a given $\boldsymbol{x}$, the objective is to find the two optimal parameters (w^*, b^*) to define the model of the most accurate prediction.

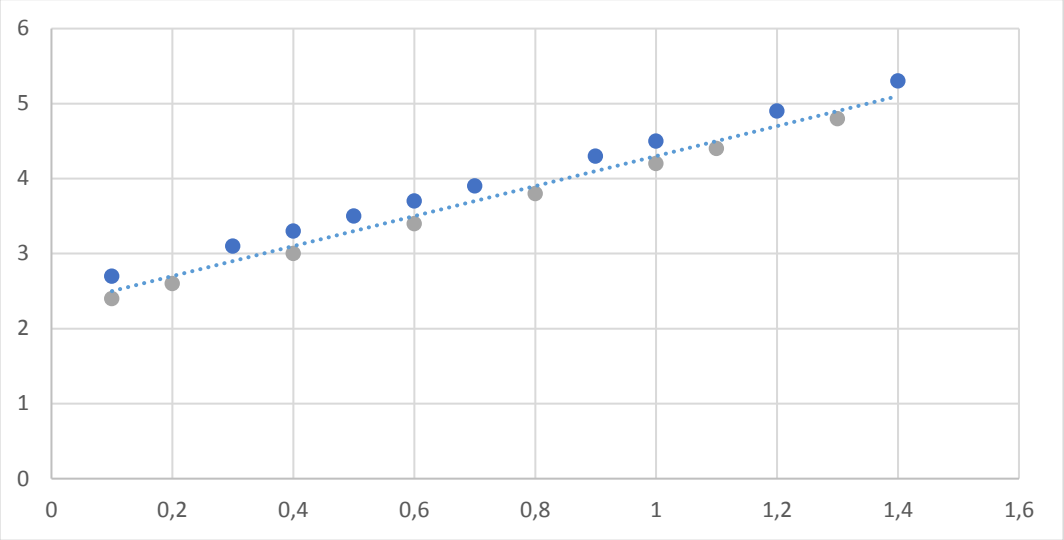


Figure 5 Linear regression model for 1D

Geometrically, we want to draw a line (2D) or hyperplane D > 2 close to all training example as much as possible. see Figure 5.

In machine learning algorithm our goal is to minimize the **cost function** which is the same as the **objective function** in the optimization problem, these function dependent on other function: the **loss function.**

Loss function

The loss function is a function that take the hypotheses $\hat{\boldsymbol{y}} = f\boldsymbol{w},_b(\boldsymbol{x})$ and the real data value $\boldsymbol{y}$ as input and give us the deference between them as output. For example, we use **Squared error loss function** in linear regression

$$\mathcal{L}(y, \hat{y}) = \left(\hat{y} - y_i\right)^2 \tag{2.3}$$

Cost function

The cost function $\mathcal{J}$ is used to evaluate the performance of a model, and is defined with the loss function $\mathcal{L}$ as follows:

$$\mathcal{J}_{w,b} = \frac{1}{N} \sum_{i=1\ldots N} \mathcal{L}(y, \hat{y}) \tag{2.4}$$

4. Neural Networks and Deep Learning

Neural networks are a class of models that are built with layers. Commonly used types of neural networks include convolutional and recurrent neural networks [9].

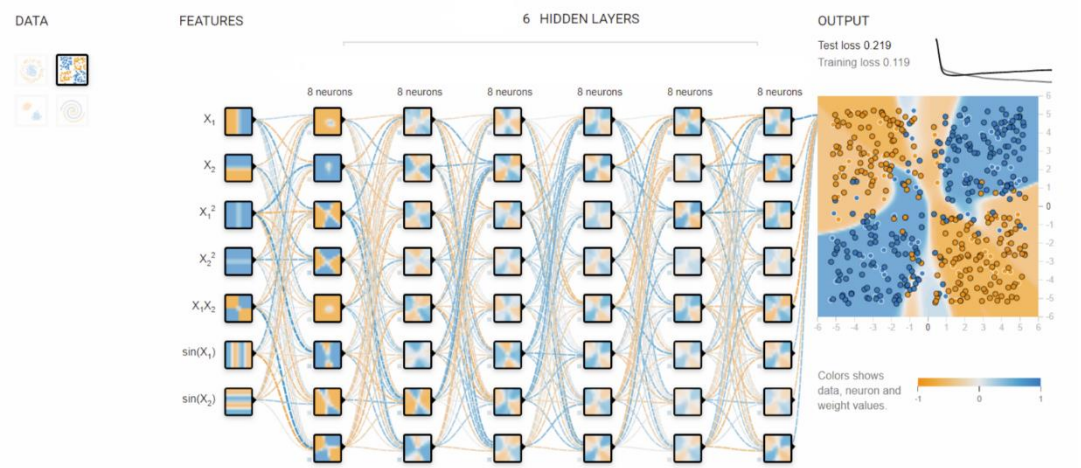


Figure 6 A Deep Neural network illustration , generated by https://playground.tensorflow.org/

4.1.Concepts and mathematic formalism of Neural Network

Neural network is a nested mathematical function,

$$y = f(\mathrm{x}) = f_3\left(f_2\left((f_1(x)\right)\right) \tag{4.1}$$

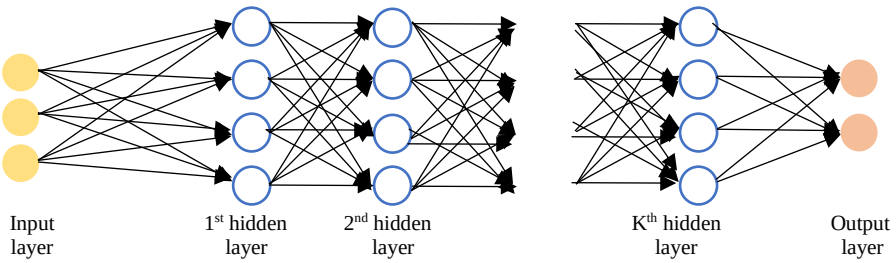


Figure 7 Graph of a fully connected neural network

By noting i the i^{th} layer of the network and j the j^{th} hidden unit of the layer, we have:

$$z_j^{[i]} = w_j^{[i]^T} x + b_j^{[i]}$$

where we note w, b, z the weight, bias and output respectively

4.1.1. Hidden unit and Neurons

The hidden layers are the layers come after the input layer of neural network; it contains the hidden units (artificial neurons).

Type of neuron and activation function

The artificial neuron is a unit (function) has receive inputs from the previous layers units and produce an output to the next layer units (in the output layer gives a prediction or classification).

Activation functions are used at the end of a hidden unit to introduce non-linear complexities to the model.

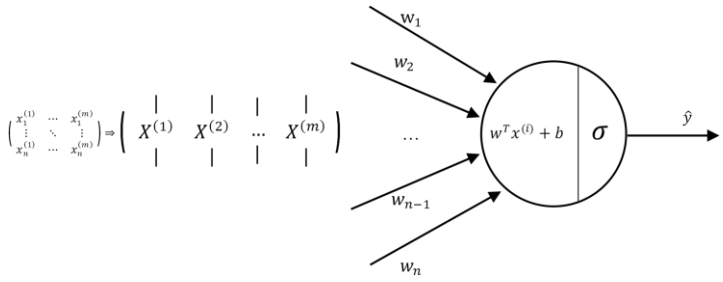


Figure 8 Single Artificial Neuron

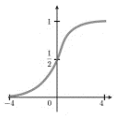
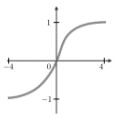
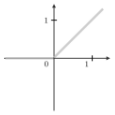
Sigmoid	Tanh	ReLu
$g(z) = \frac{1}{1 + e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = max(z, 0)$
		

Table 3 Activation functions formula and their curves.

LOSS FUNCTIONS

Loss function is the function that measure the prediction of the neural network. In the table below we give an example of some used function in neural network.

Least squared	**Logistic**	**Hinge**	**Cross-entropy**
$\frac{1}{2}(y - z)^2$	$log(1 + e^{-yz})$	$max(0,1 - yz)$	$-y\, log(z) + (1 - y)\, log(1 - z)$
Linear regression	Logistic regression	SVM	Neural Network

Table 4 Loss functions used in Neural Network

CONCLUSION

In this chapter we tried to explain the main concept of machine learning algorithms and modeling, we started by some definitions, the type of problems, prediction and type of learning after that we tried to explain the building blocks of Machine learning model, finally we explained the concept of neural network and we made some illustrations, tables and equations all this concept we adapted from CS229 [10]and CS230 [11] , Stanford university course by Andrew Ng.

The neural network concepts and mathematical formulation, Optimizers: (SGD, Adam, RMSProp…), regularization techniques (L2 regularization, Dropout), type of neural networks, …etc. all these concepts we didn't mention in this dissertation because they are complicated and needs a lot of time to rewrite and review.

Chapter 03

Implementing Deep Neural Network models on Tox21 Dataset.

In this chapter, we will talk in detail about the project that we worked on, at the beginning we will explain the general problem of molecular data sets, and then we will pick Tox21 data set which we used in our project to build the deep neural network model, we start with an overview of the Toxicology in the 21-century program after that we talked about the Tox21 challenge data sets. The next section we will find the implementation of this project, we tried to explain each block of code from the collection of data until the results comparison and discussion.

1. Problem statement

Generally, the molecular dataset are supervised datasets tested on a range of different properties, there are many categories considered as subdivisions of these properties like: Physiology, physical chemistry, biophysics and physiology.

the datasets of drug discovery and design are collection of compounds, we find that the total number of compounds in the existed open datasets are more than 700000 compounds (examples) [12], we often find these compounds represented by SMILE format, in order to make the data understood by the algorithms we calculate the descriptors of this compounds by different tools such as open source libraries like rdkit or software like Dragon 7.0 or alvaDesc 0.1. these datasets are different and we can classify them by their tasks (subset of the subdivision, as we mentioned in the first paragraph), distribution of Data and the type of problem: regression or classification, see Table 5 Figure 25, all these factors help us to build the best models that fit this data and make an accurate prediction. In this project we worked on Tox21 [13] Dataset to apply different neural networks model types and compare it with the benchmark of **MoeculeNet** .

Category	Dataset	Data Type	Numbers of Tasks	Task Type	Numbers of Compounds
Quantum Mechanics	QM7	SMILES, 3D coordinates	1	Regression	7160
	QM7b	3D coordinates	14	Regression	7210
	QM8	SMILES, 3D coordinates	12	Regression	21786
	QM8	SMILES, 3D coordinates	12	Regression	133885
Physical Chemistry	ESOL	SMILES	1	Regression	1128
	FreeSolv	SMILES	1	Regression	642
	Lipophilicity	SMILES	1	Regression	4200
Biophysics	PCBA	SMILES	128	Classification	437929
	MUV	SMILES	17	Classification	93087
	HIV	SMILES	1	Classification	41127
	PDBbind	SMILES, 3D coordinates	1	Regression	11908
	BACE	SMILES	1	Classification	1513
Physiology	BBBP	SMILES	1	Classification	2039
	Tox21	SMILES	12	Classification	7831
	ToxCast	SMILES	617	Classification	8575
	SIDER	SMILES	27	Classification	1427
	ClinTox	SMILES	2	Classification	1478

Table 5 Details of Drug design and Discovery Datasets [12]

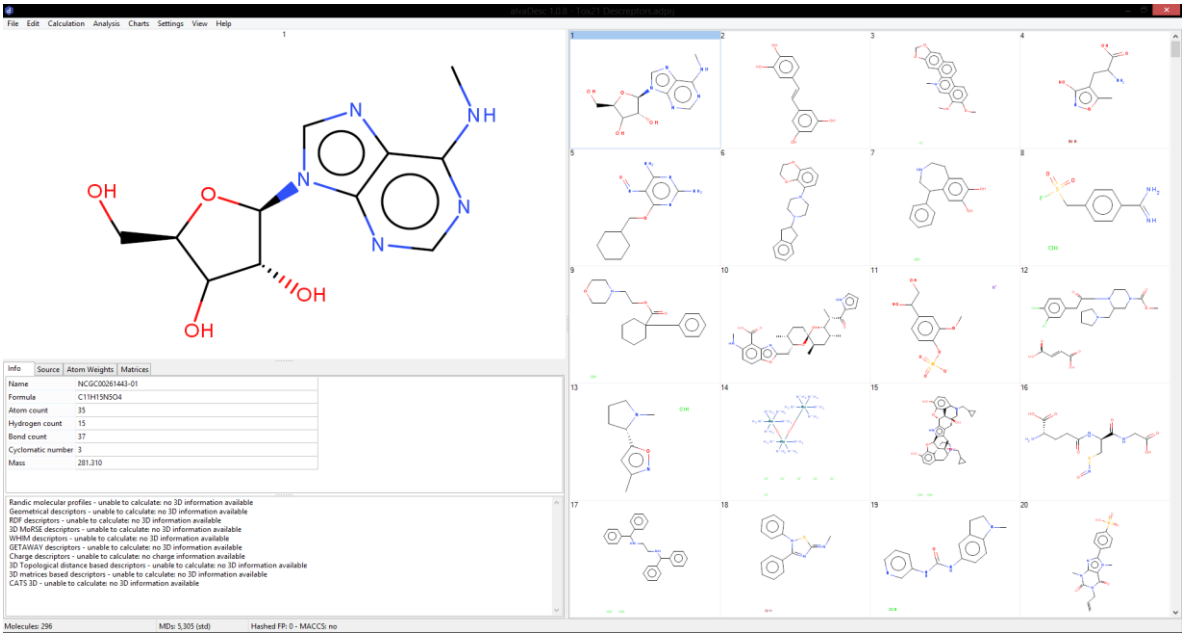


Figure 9: Molecular graph generated from Smile formula using AlvDesc 0.1

2. Toxicology in the 21st Century (Tox21) program

In this subsection we will define the **Tox21** program and mention the general goals, finally we will explain in details the Tox21 data challenge which is part of Toxicology in 21st Century and the dataset used in our project.

The Toxicology in the 21st Century program, or Tox21, is a unique collaboration between several federal agencies to develop new ways to rapidly test whether chemicals may affect human health [14].

1. Goals of the Tox21 Program

Tox21 lunched to research, develop, evaluate and create new test methods to predict the toxicology of substances and how they will affect the human being.

2. Tox21 dataset challenge

Tox21 dataset has more than 10000 chemical compounds, with 12 different toxic effects given to the participants as of these labels compounds to build predictive models and look for structure activity relationship [13]. The data challenge organizers prepared the training set and the test set to evaluate the results of participants, but after that they released that the participants could split the data randomly to improve the results of their models in the final evaluation. The chemical compounds data were given in SMILE formula, it can be represented as an undirected graph, the nodes and edges represent atoms and bonds respectively, we can generate the molecular graph and calculate the descriptors of chemical compounds data given as smile formula, see **Figure 9**. To understand more the data, we made an illustration of the first four line of the tox21 dataset see Table 6, this table contains all the information before the cleaning process.

Tox21 aims to:

- Prioritize substances for further in-depth toxicological evaluation
- Identify mechanisms of action for further investigation (e.g., toxicity-associated and disease-associated pathways).
- Develop models that better predict how substances will affect biological responses (predictive toxicology).
- Employ testing methods using human cells (in vitro approaches).
- Reduce time, effort, and costs associated with testing.
- Contribute to the reduction, refinement, and replacement of animals used in toxicity testing.

3. Development environment Programing language, Libraries and frameworks

In this project we have used many tools, frameworks, libraries and new software, we will define all of them briefly.

Google Colaboratory (CO Lab)

CO lab is a Jupyter notebook that run on the cloud, it's related with Google drive and GCP, Its give the developers a free GPU with 12 GB of ram and 358 GB temporary Disk storage. The notebook runs on Linux OS, and we can install Libraries, add packages to the environment, and it is completely free [15].

Python

It is a high-level language with an easy syntax, recently becomes the most used programming language in machine learning, deep learning, data science, and artificial intelligence projects. The high level and ease use of frameworks like Scikitlearn, Tensorflow, Pytorch, and libraries like Numpy, Pandas, rdkit and, Mordred in Python made us decide to use it in this project, ach one of these tools has a function, for example: Numpy deal with mathematical calculation and linear algebra, Pandas is used in preprocessing phase which consist on cleaning, displaying, changing the type of data, …etc.

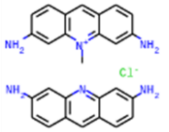
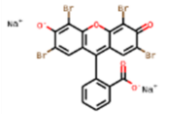
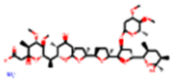
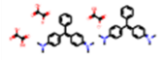
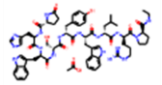
Scikitlearn

Scikit-learn [16] is an open source machine learning library for python, released in first February 2010. Built on top of SciPy library, integrate the state-of-the-art of machine learning algorithms [17].

TensorFlow

TensorFlow [18] is an open source library developed by researchers and engineers of google brain team, designed for numerical computations of data flow graphs. The general purpose of **TF** is to work on machine learning and deep learning research projects [19].

Table 6 the first five raw of Tox21 Datasets, the red columns represents the twelve tasks of toxicology.

DSSTox_CID	F	Formula	ID	NR-AR	NR-AR-LBD	NR-AhR	NR-Aromatase	NR-ER	NR-ER-LBD	NR-PPAR-gamma	ROMol	SR-ARE	SR-ATAD5	SR-HSE	SR-MMP	SR-p53
25848	468.9806 (35.4535+224.2805+209.2465)	C27H25ClN6	NCGC00178831-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN		NaN	NaN	0	NaN	NaN
5234	691.8542 (645.8757+22.9892+22.9892)	C20H6Br4Na2O5	NCGC00166114-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN		NaN	NaN	0	NaN	NaN
28909	934.1584 (916.1205+18.0379)	C47H83NO17	NCGC00263563-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN		NaN	NaN	0	NaN	NaN
5513	927.0048 (329.4575+89.0275+89.0275+329.4575+90.0349)	C52H54N4O12	NCGC00013058-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN		NaN	NaN	1	NaN	NaN
26683	1342.5025 (1282.4505+60.0520)	C66H87N17O14	NCGC00167516-01	0	NaN	NaN	NaN	NaN	NaN	NaN		NaN	NaN	NaN	NaN	NaN

Keras

Keras is a high-level neural network API, run on top of TensorFlow and other machine learning libraries, designed to code less lines in deep learning projects, it makes it easy to build, test and evaluate models [20].

NumPy

NumPy is a scientific package designed for scientific computing with Python language, and it allows using code of some other languages like C/C++ and Fortran, it has N-dimensional array objects, broadcasting functions, and other complex mathematical functions [21].

Matplotlib

Matplotlib is a python library for 2D plotting, used to produce high quality figures, we can use it in all Python environments of developing, these figures can be saved in different format to make their use easy for scientific publications [22].

Pandas

Pandas is an open source Python library designed for data analysis and data structure ease of use. It helps Python users to work on data analysis workflow without thinking to change over other data analysis languages [23].

RDKit

RDKit is an open source toolkit for cheminformatics, can be used in many platform, and different programing languages, it has many functionalities like 2D and 3D molecular operations, descriptors generations for machine learning models…etc [24].

4. Tools used calculate the descriptors of molecules

Mordred

Mordred [25] is an open source descriptors calculation software, it can be used as web application or as Python package on the most used operating systems (Windows, Linux, macOS),with Mordred we can calculate 1825 descriptors (2D and 3D descriptors) [26].

AlvaDesc 0.1

AlvaDesc is a high-level tool which can calculate more than 5305 Molecular descriptors, and number of molecular fingerprints. It provides tools to explore Datasets [27]:

- molecule structure verification using PubChem services.
- molecule structure visualization and filtering.
- Principal Component Analysis (PCA) and correlation analysis.

5. Implementing Deep learning model on Tox21 datasets

This section contains the project source code and the explanation of each code block, it is the same as the Jupyter notebook of this project.

1. Package installation in Google CO Lab averment

in this cell we collect all the required package installation in our project, It is like batch installation in Linux averment, almost all of the packages are installed using "pip", some other using "Conda" and or imported from GitHub.

In [0]
```
#Installing scikit learn
!pip install scikit-learn
#installing scikit-multilearn
!pip install scikit-multilearn
#Installing rdkit
!wget -c https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh
!chmod +x Miniconda3-latest-Linux-x86_64.sh
!time bash ./Miniconda3-latest-Linux-x86_64.sh -b -f -p /usr/local
!time conda install -q -y -c conda-forge rdkit
# Molecular descriptor calculator package
# https://github.com/mordred-descriptor/mordred/blob/develop/README.rst
pip install 'mordred[full]'
#installing seaborn
!pip install --upgrade seaborn== 0.9.0
#installing SDF
!pip install SDF
#installing
!pip install wget
#add the new installed packages to the path
%matplotlib inline
import matplotlib.pyplot as plt
import sys
import os
sys.path.append('/usr/local/lib/python3.7/site-packages')
```

Figure 10 Package installation in Google CO Lab averment

2. Import Packages, Libraries and Frameworks

Here we start our code by importing all the libraries and frameworks that we need in this project.

In [1]
```
import tensorflow as tf
import numpy as np
import sklearn as sk
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import roc_curve, auc, roc_auc_score
from sklearn.model_selection import train_test_split
import seaborn as sns
import warnings
```

Figure 11 Import Packages, Libraries and Frameworks

3. Install the PyDrive wrapper and import data from Google Drive

we use PyDrive to import data or files from google drive in colab envirment.

In [3]
```
# Install the PyDrive wrapper & import libraries.
# This only needs to be done once per notebook.
!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
# Authenticate and create the PyDrive client.
# This only needs to be done once per notebook.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
# Download a file based on its file ID.
# A file ID looks like: laggVyWshwcyP6kEI-y_W3P8D26sz
#https://drive.google.com/open?id=1D3W1xHfsz543zyVYpS9HEtyElAt177r0
fileid1 = '1EfC---GqsIObD0ouhJnSVTzSCkUPHMSj'
fileid2= ' 1qfqpgnSJFjoOBfv_zpRi40Wf7PlzXFgo'
fileid2= ' 1pAZqQhRgzE4j59DnFZdrRRDOP91us7LD'
fileid2= ' 14puiaHdD_-VMTA7XTMkJS3AkpAaqQymU'
Downloaded1 = drive.CreateFile({'id' : fileid1 })
downloaded2 = drive.CreateFile({'id' : fileid2 })
Downloaded3 = drive.CreateFile({'id' : fileid3 })
downloaded4 = drive.CreateFile({'id' : fileid4 })
downloaded1.GetContentFile("tox21_descriptors_5k_train")
downloaded2.GetContentFile("tox21_descriptors_5k_test")
Downloaded3.GetContentFile("tox21_10k_data_all")
Downloaded4.GetContentFile("tox21_10k_challenge_test")
```

Figure 12 Install the PyDrive wrapper and import data from Google Drive

4. Data Preprocessing

Process used for Tox21 Data preparation

In data preparation phase we follow many rules to prepare the data which are necessary before start building the model, there are many processes but, in our case, we will use just some of them:

- Convert the data from raw data to numeric.
- Data cleaning.
- Data transformation.
- Standardization of data.
- Splitting the data to train and test sets.
  filling missing values.

5.4.1. Import Tox21 dataset from the official web site

The original file of tox21 is a sdf file contains a table of 17 columns and 12K lines

In [4]
```
#extarct file to directory
import zipfile
import gzip
import wget
#tox21 21 dataset link
url='https://tripod.nih.gov/tox21/challenge/download?id=tox21_10k_data_allsdf&sec='
wget.download(url, '/content/')
zip_ref = zipfile.ZipFile("../content/tox21_10k_data_all.sdf.zip", 'r')
zip_ref.extractall("../content/")
zip_ref.close()
```

Figure 13 Import Tox21 dataset from the official web site

5.4.2. Loading the data from the SDF file

we use Pandas tools from rdkit library to load the data of sdf format to a Pandas Dataframe object.

In [5]
```
#Load the data from sdf file
path = "../content/tox21_10k_data_all.sdf"
tox21_10k_challenge= PandasTools.LoadSDF(path)
#Displaing the first four lines of the Data
tox21_10k_challenge.head()
```

Out[5] : See: Table 6

Figure 14 Loading the data from the SDF file

5.4.3. Exporting the data to Excel file

In [6] :
```
#Export the Data to an exel files
tox21_10k_challenge.to_excel('Tox21_10k_data_all.xlsx',columns= ('DSSTox_CID',
'FW','Formula', 'ID', 'NR-AR', 'NR-AR-LBD', 'NR-AhR', 'NR-Aromatase', 'NR-ER','NR-ER-LBD', 'NR-
PPAR-gamma', 'SR-ARE', 'SR-ATAD5', 'SR-HSE', 'SR-MMP','SR-p53'))

tox21_10k_challenge_test.to_excel(' Tox21_10k_challenge_test.xlsx', columns= ('DSSTox_CID',
'FW', 'Formula', 'ID', 'NR-AR', 'NR-AR-LBD', 'NR-AhR', 'NR-Aromatase', 'NR-ER', 'NR-ER-LBD',
'NR-PPAR-gamma', 'SR-ARE', 'SR-ATAD5', 'SR-HSE', 'SR-MMP',
'SR-p53'))
```

Figure 15 Exporting the data to Excel file

5.4.4. Loading the new Excel file and preparing the data

we put the data in a Pandas Dataframe after that we delete the columns that will not be used in the computing process, we keep only the columns of the 12 tasks which represent the labels of this dataset.

In [6]:
```
#loading Execl data of tox21 Train and test sets into Pandas data frame
tox21_10k_train=pd.read_excel('../content/Tox21_10k_data_all.xlsx')
tox21_10k_test=pd.read_excel('../content/Tox21_10k_challenge_test.xlsx')

# Read the Dataset from Exel files
xlxsf1=pd.ExcelFile('../content/tox21_descriptors_5k_train')
xlxsf2=pd.ExcelFile('../content/tox21_descriptors_5k_test')
tox21_desc_train= xlxsf1.parse(sheet_name='Sheet2')
tox21_desc_test= xlxsf2.parse(sheet_name='Sheet2')
```

Figure 16 Loading the new Excel file and preparing the data

5.4.5. Descriptors data loading and preparation for the computation phase

Descriptors data represents the examples (set of features X) that will be used in the predictive model.

In [7]:
```
#Preparing and cleansing  of data
## Droping  ids and smile formula columns
tox21_10k_train.drop(['Unnamed: 0','DSSTox_CID','FW','Formula','ID'],axis=1,
inplace=True)
tox21_10k_test.drop(['Unnamed: 0', 'DSSTox_CID','FW','Formula','ID'],axis=1,
inplace=True)


# Droping 'No.' and 'NAME' columns from Desc data (feature sets)
tox21_desc_train.drop(['No.','NAME','Uc'], axis=1, inplace=True)
tox21_desc_test.drop(['No.','NAME'], axis=1, inplace=True)


## fill Nan cells with zeros
tox21_10k_train=tox21_10k_train.fillna(0)
tox21_10k_test=tox21_10k_test.fillna(0)

tox21_desc_train.replace('Nan', 0 , inplace=True)
tox21_desc_test.replace('Nan', 0 ,inplace=True)

tox21_desc_train.fillna(0, inplace=True)
tox21_desc_test.fillna(0,inplace=True)

#convert the data to numeric
#changing the data type to float
tox21_10k_train_T = np.array(tox21_10k_train.values, dtype = np.float64)
tox21_10k_test_T = np.array(tox21_10k_test.values, dtype = np.float64)
tox21_desc_train_T = np.array(tox21_desc_train.values, dtype = np.float64)
tox21_desc_test_T = np.array(tox21_desc_test.values, dtype=np.float64)
```

Figure 17 Descriptors data loading and preparation for the computation phase

5.4.5. Standardization of Data

The data has been scalled using Scikit learn ***StandarScalar()*** function to standardize features by removing the mean and scaling to unit variance

The standard score of a sample x is calculated as:

$$z = (x - u) / s$$

where ***u*** *is the mean of the training samples or zero if* ***with_mean = False****, and* ***s*** *is the standard deviation of the training samples or one if* ***with_std=False*** [28].

In [8]:
```
#standardization: scaling the data
scaler = sk.preprocessing.StandardScaler()
scaler.fit(tox21_desc_train_T)
scaler.fit(tox21_desc_test_T)
# transforming the data
tox21_desc_train_S = scaler.transform(tox21_desc_train_T)
tox21_desc_test_S = scaler.transform(tox21_desc_test_T)
```

Figure 18 Standardisation of Data

Splitting the data

we have two splitting strategy:

- **Default split**

As we mention before in page 26, the organizers have prepared the data for the participants to evaluate them.

- **Random Split**

The data splitted randomly into two sets: training set and validation set, using Scikit-learn *train_test_split* module [29], we give the validation set 2.465 % of the training set.

In [9]:
```
x_train = tox21_desc_train_S
y_train = tox21_10k_train_T
x_test = tox21_desc_test_S
y_test = tox21_10k_test_T

# checking the shape of data before building the model
# reshape the x train data
x_train = np.delete(x_train, [11758,11759,11760, 11760, 11761, 11762, 11763,11764], axis=0)
x_test = np.delete(x_test , [295] , axis= 0 )

# Split  290/11764
x_train, y_train,x_val, y_val = train_test_split( x_train , y_train, test_size=0.02465)
# we put this two sets in one tuple to use it in tf.keras.model.fit
val_data= (x_val,y_val)
```

Figure 19 Some data cleansing and Validation set split

The shape of the data sets (train, validation, test)

In [10]:
```
print('x_train.shape =', x_train.shape)
print('y_train.shape =', y_train.shape)

print('x_val.shape =', x_val.shape)
print('y_val.shape =' , y_val.shape)

print('x_test.shape =', x_test.shape)
print('y_test.shape =' , y_test.shape)
```

Out [10]:
```
x_train.shape = (11468, 3874)
y_train.shape = (11468, 12)
x_val.shape = (290, 3874)
y_val.shape = (290, 12)
x_test.shape = (295, 3874)
y_test.shape = (295, 12)
```

Figure 20 The shape of the data sets (train, validation, test)

5.5. Building the Deep Neural Network model

5.5.1. Model Configuration (hyperparameters, Régularisation)

In this project we tried to find the best configuration for the deep neural network model to get better results and to reduce overfitting, we have built many models with different hyperparameters (#neurons, activation function, #number of layers….), optimizers, and we tried different technique of regularization. We made a summary of the last model that we developed, see **Table 7**.

Layer		#Neurons	Activation function	Regularization	Loss function	Optimizer
Hidden layers	Layer 1	500	ReLu	L2 regularization(0.0005)	Binary cross entropy	ADAM(0.0001)
				Dropout(0.5)		
	Layer 2	500	ReLu	L2 regularization(0.0005)		
				Dropout(0.5)		
	Layer 3	200	ReLu	L2 regularization(0.0005)		

				Dropout(0.5)		
Output Layer	12	Softmax		-		

Table 7 our Last DNN Model configuration summary

Loss function

We use binary cross-entropy (rather than categorical cross-entropy.) in neural network model for multi-label classification, however, the goal is to treat each output label as an independent Bernoulli distribution and we want to penalize each output node independently [30].

In [12]:

```
from keras import regularizers

model=tf.keras.models.Sequential( [
                                    tf.keras.layers.Dense(500,
                                            kernel_regularizer= tf.keras.regularizers.l2(0.001),
                                            activation = "relu" ),
                                    tf.keras.layers.Dropout(0.5),
                                    tf.keras.layers.Dense(500,
                                            kernel_regularizer= tf.keras.regularizers.l2(0.001),
                                            activation = "relu" ),
                                    tf.keras.layers.Dropout(0.5),
                                    tf.keras.layers.Dense(500,
                                            kernel_regularizer= tf.keras.regularizers.l2(0.001),
                                            activation = "relu" ),
                                    tf.keras.layers.Dropout(0.5),

                                    tf.keras.layers.Dense(12, activation = "softmax")])

#optimzers
adam = tf.keras.optimizers.Adam(lr=0.001,beta_1=0.9, beta_2=0.999)
adammax =tf.keras.optimizers.Adamax()
sgd=tf.keras.optimizers.SGD()

#losses
bce=tf.keras.losses.binary_crossentropy
mse=tf.keras.losses.mean_squared_error

#Compiling the model
model.compile( optimizer=adam , loss =bce, metrics=['binary_accuracy'])

# start the training

fit_log = model.fit(x_train, y_train, epochs=100 , batch_size= 128 , validation_split=0.25)

# start the training

test_loss = model.evaluate(x_test, y_test)
```

Out[12]

```
Train on 11468 samples, validate on 290 samples
Epoch 1/100
4404/4404 - 0s 120us/sample - loss: 1.7662 - acc: 0.7751 - val_loss: 1.5472 - val_acc: 0.9114
.
.
.
Epoch 100/100
4404/4404  - 0s 43us/sample - loss: 0.1464 - acc: 0.9630 - val_loss: 0.2992 - val_acc: 0.9208


Test on 1958 samples
1958/1958  - 0s 63us/sample - loss: 0.2286 - acc: 0.9428
```

Figure 21 Our Deep Neural Network Model (Layers, Neurons, Regularization, Loss function, Optimizer, traing configuration – epochs, batch size,) and showing the results of the training and and evaluation.

5.5.2. Model evaluation

Model evaluation is the step that allow us to evaluate the performance of our model, we use the validation and test sets to see if the precision of prediction, there are many metrics can we use, the well-known metric that used often in deep learning project is the **accuracy**.

In this project we used **AUC** (**Area under curve**) **and ROC (**R**eceiver operating characteristic curve).** The reasons of choosing this metric are:

- ROC is great way to visualize the performance of binary classifier [31].
- AUC is a value that summaries the classifier performance.
- AUC_ROC is the only metric used in MoleculeNet Benchmark.

ROC - AUC metric

- **AUC**:

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.

- **ROC**

ROC curve is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate (TPR) is a synonym for recall and is therefore defined as follows:

$$TPR = \frac{TP}{TP + FN}$$

- False Positive Rate (FPR) is defined as follows:

$$FPR = \frac{FP}{FP + TN}$$

ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives [32].

AUC_ROC Curve

In [14]:
```
# plot
plt.figure(1)
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr_t, tpr_t, label='(auc_test area = {:.3f})'.format(auc_test))
plt.plot(fpr_val, tpr_val, label='(auc_val area = {:.3f})'.format(auc_val))
plt.plot(fpr_tr, tpr_tr, label='(auc_tr area = {:.3f})'.format(auc_tr))

plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve')
plt.legend(loc='best')
#plt.show()
plt.savefig("AUC.pdf")
```

Out[14]

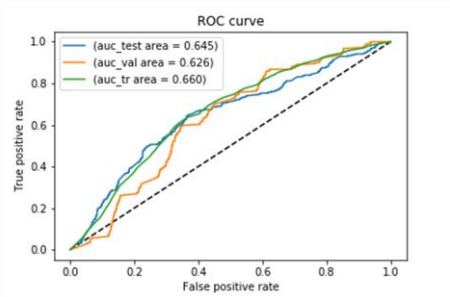


Figure 22 AUR_ROC Curve

Training and validation Loss curves

In [15]:
```
# plot
plt.xlabel('epochs')
plt.ylabel('Loss')
plt.title('Loss curve')
plt.ylim(0.,3)
plt.plot( fit_log.epoch, fit_log.history["loss"], label="Train loss")
plt.plot( fit_log.epoch, fit_log.history["val_loss"], label="Valid loss")
plt.legend()
```

Out[15]

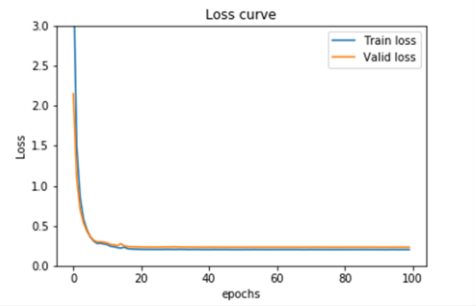


Figure 23 Training and validation Loss curves

Accuracy Curve

In [15]:
```
# plot
plt.xlabel('epochs')
plt.ylabel('Loss')
plt.title('Loss curve')
plt.ylim(0.,3)
plt.plot( fit_log.epoch, fit_log.history["loss"], label="Train loss")
plt.plot( fit_log.epoch, fit_log.history["val_loss"], label="Valid loss")
plt.legend()
```

Out[15]

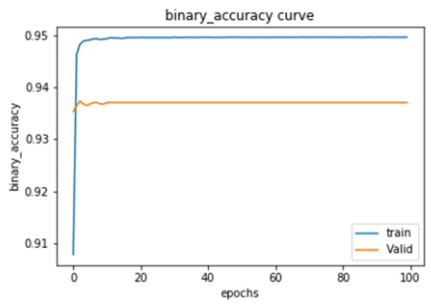


Figure 24 Accuracy Curve

6. Results and Discussion

We divided this section in two part, in the first one we tried to make a summary of our project results, we put the results in Table 8 with the benchmark of **MoleculNet** to see and compare between models, and then we made a visualization of this results in a clustered bar chart Figure 25.

1. Comparison of our results with the benchmark of Molecule Net

Model	Metric	Performance		
		Train	Validation	Test
Our model (1500)	AUC_ROC	0.931	0.820	0.834
	ACC	0.9534	0.920	0.9420
Our model (3874)	AUC_ROC	0,71	0,63	0,66
	ACC	0,95	0,91	0,92
Convolution Graph	AUC_ROC	0:905 ± 0:004	0:825 ± 0:013	0:829 ± 0:006
Wave	AUC_ROC	0:875 ± 0:004	0:828 ± 0:008	0:820 ± 0:010

Table 8 Comparison between our models results and the MoleculeNet Benchmark

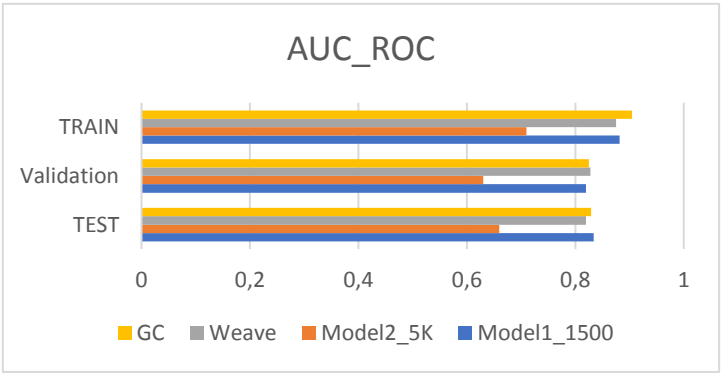


Figure 25 Comparison of our results with MoleculeNet Benchmark

7. Discussion

At the beginning, the model shows a bad result, after that we tried to identify the problems one by one, we found that we have small amount of data, missed values, and that cause overfitting. We discussed these problems in below section separately:

Problem of over fitting

The first model applicated on the dataset of 1500 (descriptors calculated using Mordred library) descriptors has a good result (AUC=0.834) and when we compare it with the benchmark, we can say it is approximately the same. see **Table 8** , but when we tried with the new calculated data (calculated using AlvDesc 0.1) we didn't get the same results, we kept adjusting the parameters of the DNN model to reduce the overfitting by applying the regularization techniques such as ***L2 regularization*** and ***Dropout.***

Data collection and calculation (Getting additional Data)

We didn't find a free Tools (Software or Libraries) to calculate more than 1500 descriptors, we tried to get the license of professional software like **Dragon 7**, unfortunately they don't have a license for student, after contacting the company they provide AlvDesc 0.1 license which is a new software in the test Phase (It's the next generation of **Dragon 7)**.

When we used it to calculate the descriptors, we found a large number of missed values and sometimes features without any values (Nan), this software made us focus more on data Cleansing and manually feature checking.

Future work and suggestions

We were looking for applying the Convolutional neural network and other type of neural networks on this data set, but we didn't find the sufficient time,

In the future if there are big molecular dataset (Millions simples) we suggest using complex neural network models like CNN, and for the students they will work on the same subject, I think working on comparative study at the same kind of data is good idea while the implementation of Deep neural network becomes more easy than before.

Conclusion

In this dissertation we discovered a new research area (cheminformatics) that use the mathematical modeling and the statistical learning algorithms. We started this dissertation with a chapter that explain the drug discovery and design process. The second chapter focuses on Machine learning and Deep neural network concepts and their mathematical formulation. The third chapter represent our work, in which we explained in details each step and tried to make the report of implementation comprehensive even for those who are beginners in deep learning.

In this project we tried to understand the molecular data sets before choosing one of them to try to get new results better than the results of the benchmark, first of all we found a problem of descriptors calculation, these descriptors represent the data sets that we need to feed to the neural network model. We started the project with 1500 descriptors (feature sets), we get the same results of MoleculeNet benchmark, but we think that the size of data is not enough, even the results of benchmark have the same problem of overfitting. We tried to get new calculation method, we brought the AlvaDesc 0.1 software which is on test phase (cost more than 900$) to calculate 5000 descriptors, after the cleansing process we found that we have only 3000 descriptors per example, after feeding the model with this new data we didn’t get the same results even the results on training set were not expected, in this short period of time (between the date of getting the license and the deadline of project delivery) we were not able to find the best solution we made a comparison in page 38 to show the deference between our results and the MoleculeNet benchmark.

We were aiming to use and apply Convolutional Neural Network on this project, we tried to add some convolution layers at the beginning but we stopped using them, because the CNN generally work well with high dimensions tensors like Image dataset, and our dataset is not big enough to get fed to a complex models like CNN.

Glossary

Assay: A procedure used by researchers to test or measure the activity of a chemical.

Biochemical: Pertaining to chemical substances and vital processes occurring in living organisms.

Biological pathway: Complex sequences of proteins and other molecules that, when activated, ultimately change some aspect of cell behavior. These pathways may alter cell behavior in an abnormal way, which can then lead to disease.

Compounds: atoms, molecules, substances.

Dataset: a collection of examples.

Example: One row of a dataset. An example contains one or more features and possibly a label. See also labeled example and unlabeled example.

High-throughput: automated assays capable of testing large numbers of chemicals in a short time frame.

***in silico*:** referring to an experiment conducted in a test tube. The silico component refers to silicon chips which were used for computing at the time when the term was coined.

***in vitro*:** Biological or chemical work conducted in culture dishes rather than in living animals.

***In vivo*:** research trials in the living organism

Index

A

Accuracy, 34
Algorithm, 13

B

benchmark, 22

C

cancer, 8
cell, 8
clinical, 8
Colaboratory, 24
compounds, 8
Curve, 33, 34, 42

D

diabetes, 8
Dragon 7.0, 22

J

Jupyter, 24

L

Logistic, 14, 16
loss, 17
Loss, 31, 32, 34

M

machine learning, 15, 16, 17, 24, 26
metabolism, 8
molecule, 8
Mordred, 24, 26
mutation, 8

N

notebook, 24
Numpy, 24

O

optimization, 8, 9

P

Pandas, 24
pathway, 8
precision, 33
preclinical, 8
Preclinical, 8
Preclinical phase, 8
Python, 24, 26
Pytorch, 24

R

Regression, 16, 22

S

SMILE, 22, 23
Split, 31
Standardization, 28, 30

T

TensorFlow, 5, 24, 26
Tox21, 22, 23, 27
toxic, 23
Toxicology, 23, 38

Bibliography

[1] J. DiMasi, H. Grabowski and R. Hansen, "Innovation in the Pharmaceutical industry: new estimates of R&D costs.," *Journal of Health Economics,* vol. 47, pp. 20-33, 2016.

[2] I. Kapetanovic , "Computer-aided drug discovery and development (CADDD): in silico-chemico-biological approach.," vol. 171, pp. 165 - 176, 2008.

[3] Danishuddin and A. U. Khan, "Descriptors and their selection methods in QSAR analysis: paradigm for drug design," *Drug Discovery Today,* vol. 21, no. 8, pp. 1292-1302, 2016.

[4] Khan, A.U, "Descriptors and their selection methods in QSAR analysis: paradigm for drug design," *Drug Discovery Today,* vol. 21, p. 1291–1302, 2016.

[5] L. Yu-Chen , E. R. Stefano , T. Wen and B. A. Russ , "Machine learning in chemoinformatics and drug discovery," *Drug Discovery Today,* vol. 32, pp. 1538-1546, 8 August 2018.

[6] R. Kunal , K. Supratik and N. D. Rudra , A Primer on QSAR/QSPR Modeling: Fundamental Concepts, New York: Springer-Verlag Inc, 2015, pp. 1-7.

[7] A. Afshine and A. Shervine, "Super VIP Cheatsheet: Machine Learning - CS 229-Machine Learnig," Stanford University., 2018.

[8] N. Andrew Y, M. I and Jordan, "On discriminative vs.generative classifiers : A comparision of logistic regression and naive Bayes.," in *NIPS*, New York.

[9] A. Ng, "CS229 Lecture notes - Deep Learning," Stanford University.

[10] "CS229 Machine Learning," [Online]. Available: http://cs229.stanford.edu/.

[11] "CS230 Deep Learning," [Online]. Available: https://cs230.stanford.edu/.

[12] E. N. Feinberg, B. Ramsundar, Z. Wu, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, "MoleculeNet: A Benchmark for Molecular," Pande Group, 2018.

[13] "Tox21 Data Challenge 2014," National Center for Advancing Translational Sciences (NCATS), 2014. [Online]. Available: https://tripod.nih.gov/tox21/challenge/data.jsp.

[14] "Tox21," [Online]. Available: https://ntp.niehs.nih.gov/results/tox21/.

[15] "Welcome to Colaboratory," Google, [Online]. Available: https://colab.research.google.com/notebooks/welcome.ipynb#scrollTo=-gE-Ez1qtyIA.

[16] "Scikit learn," [Online]. Available: https://scikit-learn.org.

[17] P. Fabian, V. Gaël, G. Alexandre, M. Vincent, B. Thirion, G. Olivier, B. Mathieu, P. Peter, R. Weiss, D. Vincent, V. Jake, P. Alexandre, C. David, B. Matthieu and P. Matthieu, "Scikit-learn: Machine Learning in Python," *JMLR*, vol. 12, pp. 2825-2830, 2011.

[18] "TensorFlow," Google, [Online]. Available: https://www.tensorflow.org/ .

[19] "GitHub Tensorflow," [Online]. Available: https://github.com/tensorflow/tensorflow.

[20] "keras," [Online]. Available: https://keras.io/.

[21] "Numpy officiel web site," [Online]. Available: https://www.numpy.org/ .

[22] "Matplotlib officiel web site," [Online]. Available: https://matplotlib.org/.

[23] "Pandas," [Online]. Available: https://pandas.pydata.org/.

[24] "Rdkit," [Online]. Available: http://www.rdkit.org/docs/Overview.html.

[25] "Mordred project GitHub page," [Online]. Available: https://github.com/mordred-descriptor/mordred.

[26] H. Moriwaki, Y. Tian, N. Kawashita and T. Takagi, "calculator, Mordred: A molecular descriptor," *Journal of Cheminformatics*, vol. 10, 2018.

[27] "Alvadesc officile web page," [Online]. Available: https://www.alvascience.com/alvadesc/ .

[28] "Standard Scaler - scikit-learn," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

[29] "Sklearn.model_selection.train_test_split," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html.

[30] "multi-label-classification-with-keras," [Online]. Available: https://www.pyimagesearch.com/2018/05/07/multi-label-classification-with-keras/.

[31] "Simple guide on how to generate roc plot for keras classifier," [Online]. Available: https://www.dlology.com/blog/simple-guide-on-how-to-generate-roc-plot-for-keras-classifier/.

[32] "Classification: ROC Curve and AUC," [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.

[33] C. Hongming , . E. Ola, W. Yinhai , O. Marcu and B. Thomas , "The rise of deep learning in drug discovery," *Drug Discovery Today,* vol. 23, pp. 1241-1250, June 2018.

[34] A. Burkov, The 100 page Machine learning book, Andriy Burkov, 2019.

[35] F. Chollet , Deeplearing with python, 1 ed., Manning Publications, 2017.

[36] N. Buduma and N. Locascio, Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms, 1 ed., O'Reilly, 2017.

[37] Consonni, V, Todeschini, R and eds, Handbook of Molecular Descriptors, Wiley-VCH, 2000.

[38] R. Kunal, K. Supratik and N. D. Rudra, Understanding the Basics of QSAR for Applications in Pharmaceutical Sciences and Risk Assessment, 1 ed., Academic Press, 2015.

[39] G. Aurélien , Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition, O'Reilly Media, Inc., 2019.

[40] E. N. Feinberg, B. Ramsundar, Z. Wu, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, "MoleculeNet: A Benchmark for Molecular," Pande, 2018.

[41] R. C Glem, A. Bender, C. Hasselgren, L. Carlsson, S. Boyer and J. Smith, "Circular fingerprints: Flexible molecular descriptors with applications from physical chemistry to ADME," *IDrugs : the investigational drugs journal,* vol. 9, pp. 199-204, 2006.

[42] P. Nutan and D. A. Gareja, "Cheminformatics," *Journal of Proteomics and Bioinformatics,* vol. 3, no. 8, pp. 249-252, 2010.

[43] "Alvascience," [Online]. Available: https://www.alvascience.com/alvadesc/.

[44] M. A. Nielsen, Neural Networks and Deep Learning., Determination Press, 2018.

[45] N. Chanin, I.-N.-A. Chartchalerm, N. Thanakorn and P. Virapong, "A PRACTICAL OVERVIEW OF QUANTITATIVE STRUCTURE-ACTIVITY RELATIONSHIP," *EXCLI Journal,* vol. 8, pp. 74-88, 2009.

ملخص

مع تطور نماذج التعلم العميق والنتائج المقنعة التي قدمتها في العديد من المجالات (مثل معالجة الصور الرقمية ومعالجة اللغات الطبيعية)، يحاول الباحثون ومختبرات الكيمياء الرقمية تطبيق هذه التقنيات في تصميم واكتشاف أدوية جديدة. في الآونة الأخيرة تم تطبيق خوارزمية التعلم العميق في هذا المجال و كانت نتائج البحث جيدة، لكنها نتائج اولية فقط ولا يمكننا القول أنها تؤدي إلى تصميم عقلاني للأدوية، و نعني بهذا تصميم أدوية جديدة من دون اختبارها على عينات بشرية، في هذا المشروع قمنا بتطبيق الشبكة العصبية العميقة على مجموعة البيانات الخاصة بتصميم الأدوية واكتشافها للتنبأ بسمومية الادوية، بعد تقييم النتائج و مقارنتها، وجدنا أن اكبر اشكال هو قلة البيانات مقارنة بالخوارزمية المستعملة .

Abstract

With the rise of deep learning models and the successful result showing in deferent domains (such as Computer vision and Natural language processing) researchers and laboratories of cheminformatics try to apply these techniques in drug design and discovery. recently, the application of Deep Learning in this area of research has made a good progress but it is in the early stage and we can't say that the results lead us to rational drug design, which mean designing new drugs without *in vivo* and human trials. in this master thesis project, we applied Deep Neural network (Deep learning) in drug design and discovery dataset to predict the toxicity of molecules. After the evaluation and comparison of results we found that the big problem is the small amount of data compared with used model.

Abstrait

Avec la montée en puissance des modèles d'apprentissage profond (Deep learning) et les résultats obtenus dans de nombreux domaines (tels que la vision par ordinateur et le traitement du langage naturel), les chercheurs et les laboratoires de Cheminformatics tentent d'appliquer ces techniques à la conception et à la découverte de médicaments. Récemment, l'application de Deep Learning dans ce domaine de recherche a bien progressé, mais nous n'en sommes qu'au début et nous ne pouvons pas dire que les résultats nous ont conduits à une conception rationnelle des médicaments, ce qui signifie la conception de nouveaux médicaments sans *in vivo* et les essais sur l'être humain. Dans ce travail, nous avons appliqué le réseau de Neurones profond (apprentissage profond) à la conception et à la découverte de médicaments. Après l'évaluation et la comparaison des résultats, nous avons constaté que le gros problème était la faible quantité de données comparant par les Modèles utilisés.