Proposing a potentially highly active molecule against a target protein of the 2019 Novel Coronavirus

Enroll. Nos. - 18104050, 18104051, 18104064
Name of Students - Teghdeep Kapoor, Vardhika Jain, Tanya Pandhi
Name of Supervisor - Prof. Prashant Kaushik

December - 2021

Submitted in partial fulfillment of the Degree of
Bachelor of Technology
in
Information Technology

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

TABLE OF CONTENTS

DECLARATION

I/We hereby declare that this submission is my/our own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place:	Signature:
Date:	Name: Teghdeep Kapoor
	Enrollment No: 18104050

Signature:
Name: Vardhika Jain
Enrollment No: 18104051

Signature:
Name: Tanya Pandhi
Enrollment No: 18104064

CERTIFICATE

This is to certify that the work titled "**Proposing a potentially highly active molecule against a target protein of the 2019 Novel Coronavirus**" was submitted by **Teghdeep Kapoor , Vardhika Jain and Tanya Pandhi** in partial fulfilment for the award of degree of Information Technology and Computer Science & Engineering, B. Tech. of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor ……………………..

Name of Supervisor- Prashant Kaushik

Designation- Assistant Professor

Date- 13th December 2021

ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we want to give to our pre-final year minor project supervisor, Prof. Prashant Kaushik, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project and in making this detailed report.

Furthermore, we would also like to acknowledge with much appreciation the crucial role of the staff of Mr. Bharat Gupta, guiding the team in achieving the goals on time. We have to appreciate the guidance given by other supervisors as well as the panel, especially in our project presentation that has improved our presentation skills thanks to their comments and advice.

Signature of the Student:
Name of Student : Teghdeep Kapoor
Enrollment Number : 18104050
Date :

Signature of the Student:
Name of Student : Vardhika Jain
Enrollment Number : 18104051
Date :

Signature of the Student:
Name of Student : Tanya Pandhi
Enrollment Number : 18104064
Date :

SUMMARY

The spread of this virus in the human race has caused 3.35M deaths in the world as of May 2021 and has brought the economy to a standstill. It has also introduced several challenges worldwide. Scientists and Doctors are working day and night to find a drug that will be able to dilute the symptoms of COVID-19 in the patients. We have attempted to make a project which is based on an approach to inhibit the protease of coronavirus (Or any virus) by a covalent inhibitor (also called Ligand). We will be comparing the Inhibition score of Ligand and Protease to find out one of the best inhibitors. Our inspiration and the starting point is the published research (https://www.rcsb.org/structure/6LU7), In this research N3 ligand is the inhibitor for the coronavirus protease. We will be generating the new ligands, as well as predicting the current ligands used in SARS, MERS etc. Many biological assays have been done to test compounds on the main proteases. Bioactivities measured in papers by medicinal chemists and biochemists are tracked by The National Center for Biotechnology Information (NCBI) which can be accessed by everyone. We have collected the data from the NCBI site only. We did search for some similar viruses to coronavirus, like SARS, MERS, and HIV. Protein target GI73745819 - SARS Protease, Protein target GI75593047 - HIV pol polyprotein, NS3 - Hep3 protease, and 3CL-Pro - Mers Protease. The goal of this project is to make efforts towards proposing a potentially highly active molecule against a target protein of the 2019 Novel Coronavirus. We will be training our model in such a way that it predicts the binding power of the drug towards COVID-19 protease.

Signature of Student	Signature of Supervisor
Name- Teghdeep Kapoor	Name- Prashant Kaushik
Date- 13th December 2021	Date- 13th December 2021

Signature of Student	Signature of Supervisor
Name- Vardhika Jain	Name- Tanya Pandhi
Date- 13th December 2021	Date- 13th December 2021

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS & ACRONYMS

COVID-19	Coronavirus Disease
SARS-CoV-2	Severe Acute Respiratory Syndrome Coronavirus 2
RNA	Ribonucleic Acid
MERS COV	Middle East Respiratory Syndrome Coronavirus
WHO	World Health Organization
HIV	Human Immunodeficiency Virus
AI	Artificial Intelligence
DNN	Deep Neural Network
SVM	Support Vector Machines
NCBI	National Center for Biotechnology Information
IT	Information Technology
SMILES	Simplified Molecular Input Line Entry System
QSPR	Quantitative Structure-Property Relationship
EPA	Environmental Protection Agency
EPI	Estimation Program Interface
DL	Deep Learning
ANN	Artificial Neural Network
DTBA	Drug-Target Binding Affinity
SEA	Similarity Ensemble Approach
LMCS	Ligand Maximum Common Substructure
GGNN	Gated Graph Neural Network
MPNN	Message Passing Neural Network
EMNN	Edge Memory Neural Network
CPU	Central Processing Unit
GPU	Graphic User Interface
RANC	Reinforced Adversarial Neural Network
TPSA	Topological Polar Surface Area
SRS	Software Requirements Specification
SQL	Structured Query Language
CSV	Comma Separated Values
MSE	Mean Squared Error

CHAPTER -1 INTRODUCTION

1.1 General Introduction

In early 2020, after a December 2019 outbreak in China, the World Health Organization identified SARS-CoV-2 as a new type of coronavirus. The outbreak quickly spread around the world.

COVID-19 is a disease caused by SARS-CoV-2 that can trigger what doctors call a respiratory tract infection. It can affect your upper respiratory tract (sinuses, nose, and throat) or lower respiratory tract (windpipe and lungs). Since the outbreak, many researchers have been interacting, collaborating, and working very hard to stop the spread of the disease and to propose possible treatment plans. The traditional methods for identifying chemical drugs for a virus are quite time-consuming and expensive. Which was the driving force to use Artificial Intelligence to speed up this process. Many scientists are trying to use the power of Machine Learning to predict drugs that can inhibit the growth of the SARS-CoV-2 virus in the human body.

1.2 Problem Statement

Coronaviruses aren't fairly new. They have been proven deleterious to humanity for many decades now. They were first isolated in 1962. Coronavirus is a significant virus that causes illness in both animals and humans. It is a family of RNA viruses that is medium-sized and has a viral RNA genome largest of all known. It is known that some viruses from the coronavirus group infect only certain animals, while some can breach the barrier between species, causing states ranging from a mild cold to severe acute respiratory syndrome (SARS). A new, so far unknown coronavirus, SARS-CoV-2, the cause of COVID-19 disease, belongs to the same subgroup as MERS CoV and SARS-CoV. Coronavirus, which is known to common people as COVID-19, was declared as a pandemic by WHO (The World Health Organization), on March 11, 2020. It has forced the world into a mandatory lockdown.

The spread of this virus in the human race has caused 3.35M deaths in the world as of May 2021 and has brought the economy to a standstill. It has also introduced several challenges worldwide. Scientists and Doctors are working day and night to find a drug that will be able to dilute the symptoms of COVID-19 in the patients. The drug/ medicine is able to dilute the symptoms caused by COVID-19 like sore throat, severe cough, pneumonia, and many other respiratory problems. This process is really tedious and expensive. Classical approaches involve extensive work to select the right chemical descriptors to use as input data. Which was the driving force to use Artificial Intelligence to speed up this process. New advances in machine intelligence have introduced algorithms that can learn important patterns from vast amounts of data, approaching expert-level ability in some tasks. This means that anyone with these models can contribute to the global research effort. That is precisely the goal of this project, to make efforts towards proposing a potentially highly active molecule against a target protein of the 2019 Novel Coronavirus. We will be training our model in such a way that it predicts the binding power of the drug towards COVID-19 protease.

1.3 Significance/Novelty of the Problem

Artificial intelligence (AI) and related technologies are increasingly prevalent in business and society and are beginning to be applied to healthcare. These technologies have the potential to transform many aspects of patient care, as well as administrative processes within provider, payer, and pharmaceutical organizations.

There are already a number of research studies suggesting that AI can perform as well as or better than humans at key healthcare tasks, such as diagnosing disease. Today, algorithms are already outperforming radiologists at spotting malignant tumors, and guiding researchers in how to construct cohorts for costly clinical trials. However, for a variety of reasons, we believe that it will be many years before AI replaces humans for broad medical process domains.

We believe that AI has an important role to play in the healthcare offerings of the future. In the form of machine learning, it is the primary capability behind the development of precision medicine, widely agreed to be a sorely needed advance in care. Although early efforts at providing diagnosis and treatment recommendations have proven challenging, we expect that AI will ultimately master that domain as well. Given the rapid advances in AI for imaging analysis, it seems likely that most radiology and pathology images will be examined at some point by a machine. Speech and text recognition are already employed for tasks like patient communication and capture of clinical notes, and their usage will increase.

It also seems increasingly clear that AI systems will not replace human clinicians on a large scale, but rather will augment their efforts to care for patients. Over time, human clinicians may move toward tasks and job designs that draw on uniquely human skills like empathy, persuasion, and big-picture integration.

1.4 <u>Empirical Study</u>

1.4.1 <u>Machine Learning Basic</u>

Given definitions show the difference between data mining and machine learning, two fields that are built upon each other. Many basics can therefore be considered being data mining techniques, while more advanced topics can be categorized as machine learning techniques. Thus, machine learning and data mining are methods to iteratively detect patterns in data, which is not necessarily structured. This also means that models are not explicitly programmed with a known result [1]. Examples, where machine learning is used in big data, are diverse. This includes:

- Fraud detection to find anomalies in tax or credit card data.
- Prediction of so-called "rest of useful lifetime" in industrial machines.
- Text sentiment analysis and opinion mining in social networks such as Twitter.
- Financial modeling

1.4.2 <u>Supervised and Unsupervised Learning</u>

Supervised learning: "Supervised learning is a type of machine learning algorithm that uses a known dataset (called the training dataset) to make predictions. The training dataset includes input data and response values.", [2].

Unsupervised learning: "Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.", [3].

In data mining and machine learning, an abundance of models and algorithms can be found, but most fundamentally these are divided into supervised and unsupervised learning. One fundamental example has been mentioned in the foregoing section, the clustering of iris-species. Former is a supervised process where data points are labeled ("species A", "species B" or "species C") and labels are calculated for new data points. Comparing calculated labels according to the trained model with the original label gives the model's accuracy, hence supervised.

Unsupervised learning on the other hand does not require any labeling, since the algorithm is searching for a pattern in the data. This might be useful when categorizing customers into different groups without a priori knowledge of which groups they belong to.

1.4.3 <u>Machine Learning Algorithm</u>

Types For machine learning many different algorithms can be found. A wide variety of these is available in Azure Machine Learning Studio. For simplicity, these can be subdivided into four categories, where each category is good for different kinds of problems. Anomaly detection algorithms are good for finding unusual data points. Trained classification algorithms can be used to categorize unseen data. As an example, it could be used to take in data from a phone on movement to categorize what activity is being performed. Clustering algorithms group data into clusters and look for the greatest similarities. This can be used to find unknown connections on huge sets of data. Regression algorithms are used to find patterns and build models to predict numerical values from datasets. These will take multiple inputs and determine how much each input affects the output [1].

Within the regression category, there are eight different basic algorithms, each suited for different kinds of problems, available in Azure Machine Learning Studio. Boosted decision tree regression, which is based on decision trees, where each tree depends on prior trees, uses decision splitting to create stepped functions. It learns by fitting the residuals of preceding trees to improve accuracy. Decision forest regression consists of decision trees in regression and is resilient against noise, due to the fact that many trees form a "forest". This makes it easy to parallelize. Fast forest quantile regression is effective in predicting weak relationships between variables. Unlike linear regression, quantile algorithms try to find patterns in the distribution of the predicted values rather than just predicting values. Linear Regression is the most classic type which solves linear relationships between inputs and outputs. Neural network regression is most common in deep learning and adaptable to regression problems, but might be too complex for simple regression problems and requires thorough training. This method is very stable and is often used when other algorithms can not find a solution. Ordinal regression is found useful for predicting discrete ranking. Poisson regression is useful to predict values if the response variable follows a Poisson distribution

1.4.4 <u>AI in Healthcare</u>

Healthcare is one of the major success stories of our times. Medical science has improved rapidly, raising life expectancy around the world, but as longevity increases, healthcare systems face a growing demand for their services, rising costs, and a workforce that is struggling to meet the needs of its patients.

Demand is driven by a combination of unstoppable forces: population aging, changing patient expectations, a shift in lifestyle choices, and the never-ending cycle of innovation being but a few. Of these, the implications of an aging population stand out. By 2050, one in four people in Europe and North America will be over the age of 65—this means the health systems will have to deal with more patients with complex needs. Managing such patients is expensive and requires systems to shift from an episodic care-based philosophy to one that is much more proactive and focused on long-term care management [4].

Healthcare spending is simply not keeping up. Without major structural and transformational change, healthcare systems will struggle to remain sustainable. Health systems also need a larger workforce, but although the global economy could create 40 million new health-sector jobs by 2030, there is still a projected shortfall of 9.9 million physicians, nurses and midwives globally over the same period, according to the World Health Organization.1 We need not only to attract, train and retain more healthcare professionals, but we also need to ensure their time is used where it adds the most value—caring for patients.

Building on automation, artificial intelligence (AI) has the potential to revolutionize healthcare and help address some of the challenges set out above. There are several definitions of AI, but this report draws from a concise and helpful definition used by the European Parliament, "AI is the capability of a computer program to perform tasks or reasoning processes that we usually associate with intelligence in a human being."2 AI can lead to better care outcomes and improve the productivity and efficiency of care delivery. It can also improve the day-to-day life of healthcare practitioners, letting them spend more time looking after patients and in so doing, raise staff morale and improve retention. It can even get life-saving treatments to market faster. At the same time, questions have been raised about the impact AI could have on patients, practitioners, and health systems, and about its potential risks; there are ethical debates around how AI and the data that underpins it should be used [5].

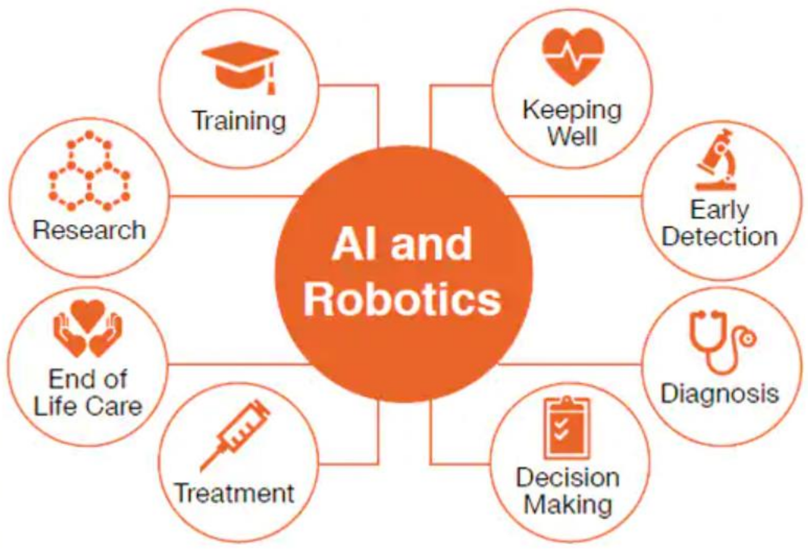


Fig.1 Application of AI in healthcare

1.4.5 AI in Drug Discovery

The process of discovering and developing a drug can take over a decade and costs US$2.8 billion on average. Even then, nine out of ten therapeutic molecules fail Phase II clinical trials and regulatory approval [6, 7]. Algorithms, such as Nearest-Neighbor classifiers, RF, extreme learning machines, SVMs, and deep neural networks (DNNs), are used for VS based on synthesis feasibility and can also predict in vivo activity and toxicity [6, 8]. Several pharmaceutical companies, such as Bayer, Roche, and Pfizer, have teamed up with IT companies to develop a platform for the discovery of therapies in areas such as immuno-oncology and cardiovascular diseases [9]. The aspects of VS to which AI has been applied are discussed below.

A. Prediction of the physicochemical properties

Physicochemical properties, such as solubility, partition coefficient (logP), degree of ionization, and intrinsic permeability of the drug, indirectly affect its pharmacokinetic properties and its target receptor family and, hence, must be considered when designing a new drug [10]. Different AI-based tools can be used to predict physicochemical properties. For example, ML uses large data sets produced during compound optimization done previously to train the program [11]. Algorithms for drug design include molecular descriptors, such as SMILES strings, potential energy measurements, electron density around the molecule, and coordinates of atoms in 3D, to generate feasible molecules via DNN and thereby predict its properties [12].

Zang *et al.* created a quantitative structure-property relationship (QSPR) workflow to determine the six physicochemical properties of environmental chemicals obtained from the Environmental Protection Agency (EPA) called the Estimation Program Interface (EPI) Suite [11]. Neural networks based on the ADMET predictor and ALGOPS program have been used to predict the lipophilicity and solubility of various compounds [13]. DL methods, such as undirected graph recursive neural networks and graph-based convolutional neural networks (CNN), have been used to predict the solubility of molecules [14].

In several instances, ANN-based models, graph kernels, and kernel ridge-based models were developed to predict the acid dissociation constant of compounds [11, 15]. Similarly, cell lines, such as Madin-Darby canine kidney cells and human colon adenocarcinoma (Caco-2) cells have been utilized to generate cellular permeability data of a diverse class of molecules, which are subsequently fed to AI-assisted predictors [10].

Kumar *et al.* developed six predictive models [SVMs, ANNs, k-nearest neighbor algorithms, LDAs, probabilistic neural network algorithms, and partial least square (PLS)] utilizing 745 compounds for training; these were used later on 497 compounds to predict their intestinal absorptivity based on parameters including molecular surface area, molecular mass, total hydrogen count, molecular refractivity, molecular volume, logP, total polar surface area, the sum of Estates indices, solubility index (log S), and rotatable bonds [16]. On similar lines, RF and DNN-based *in silico* models were developed to determine human intestinal absorption of a variety of chemical compounds [17]. Thus, AI has a significant role in the development of a drug, to predict not only its desired physicochemical properties but also the desired bioactivity.

B. Prediction of bioactivity

The efficacy of drug molecules depends on their affinity for the target protein or receptor. Drug molecules that do not show any interaction or affinity towards the targeted protein will not be able to deliver the therapeutic response. In some instances, it might also be possible that developed drug molecules interact with unintended proteins or receptors, leading to toxicity. Hence, drug-target binding affinity (DTBA) is vital to predict drug–target interactions. AI-based methods can measure the binding affinity of a drug by considering either the features or similarities of the drug and its target. Feature-based interactions recognize the chemical moieties of the drug and that of the target to determine the feature vectors. By contrast, in similarity-based interaction, the similarity between drug and target is considered, and it is assumed that similar drugs will interact with the same targets [18].

Web applications, such as ChemMapper and the similarity ensemble approach (SEA), are available for predicting drug–target interactions [19]. Many strategies involving ML and DL have been used to determine DTBA, such as KronRLS, SimBoost, DeepDTA, and PADME. ML-based approaches, such as Kronecker-regularized least squares (KronRLS), evaluate the similarity between drugs and protein molecules to determine DTBA. Similarly, SimBoost utilized regression trees to predict DTBA, and considers both feature-based and similarity-based interactions. Drug features from SMILES, ligand maximum common substructure (LMCS), extended connectivity fingerprint, or a combination thereof can also be considered [18].

DL approaches have shown improved performance compared with ML because they apply network-based methods that do not depend on the availability of the 3D protein structure [19]. DeepDTA, PADME, WideDTA, and DeepAffinity are some DL methods used to measure DTBA. DeepDTA accepts drug data in the form of SMILES, whereby the amino acid sequence is entered for protein input data and for the 1D representation of the drug structure [20]. WideDTA is CVNN DL method that incorporates ligand SMILES (LS), amino acid sequences, LMCS, and protein domains and motifs as input data for assessing the binding affinity [21].

DeepAffinity and Protein And Drug Molecule interaction prEdiction (PADME) are similar to the approaches described earlier [22]. DeepAffinity is an interpretable DL model that uses both RNN and CNN and both unlabeled and labeled data. It takes into account the compound in the SMILES format and protein sequences in the structural and physicochemical properties [23]. PADME is a DL-based platform that utilizes feed-forward neural networks for predicting drug target interactions (DTIs). It considers the combination of the features of the drug and target protein as input data and forecasts the interaction strength between the two. For the drug and the target, the SMILES representation and the protein sequence composition (PSC) are used for illustration, respectively [22]. Unsupervised ML techniques, such as MANTRA and PREDICT, can be used to forecast the therapeutic efficacy of drugs and target proteins of known and unknown pharmaceuticals, which can also be extrapolated to the application of drug repurposing and interpreting the molecular mechanism of the therapeutics. MANTRA groups compounds based on similar gene expression profiles using a CMap data set and clusters those compounds predicted to have a common mechanism of action and common biological pathway [19]. The bioactivity of a drug also includes ADME data. AI-based tools, such as XenoSite, FAME, and SMARTCyp, are involved in determining the sites of metabolism of the drug. In addition, software such as CypRules, MetaSite, MetaPred, SMARTCyp, and WhichCyp were used to identify specific isoforms of CYP450 that mediate a particular drug metabolism. The clearance pathway of 141 approved drugs was done by SVM-based predictors with high accuracy [24].

1.4.7 Neural Networks

1.4.7.1 Biological Neurons

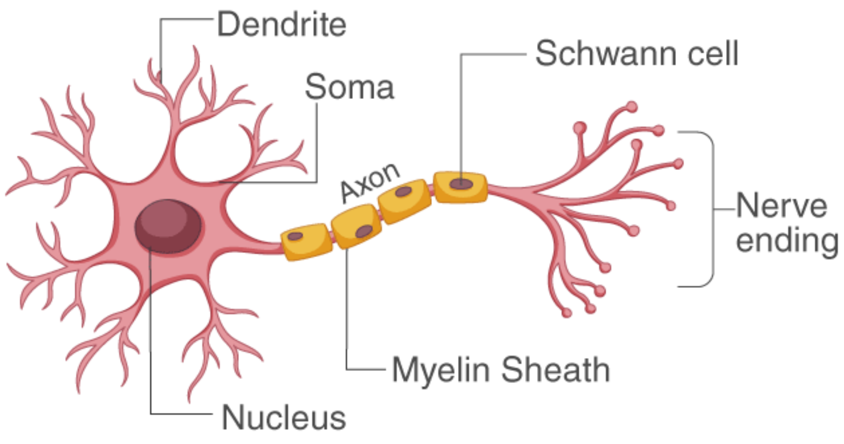


Fig.2 Biological neuron

A neuron (or nerve cell) is a special biological cell that processes information (see Figure 2). It is composed of a cell body, or soma, and two types of out-reaching tree-like branches: the axon and the dendrites. The cell body has a nucleus that contains information about hereditary traits and plasma that holds the molecular equipment for producing material needed by the neuron. A neuron receives signals (impulses) from other neurons through its dendrites (receivers) and transmits signals generated by its cell body along the axon (transmitter), which eventually branches into strands and substrings. At the terminals

These strands are the synapses. A synapse is an elementary structure and functional unit between two neurons (an axon strand of one neuron and a dendrite of another). When the impulse reaches the synapse's terminal, certain chemicals called neurotransmitters are released. The neurotransmitters diffuse across the synaptic gap. to enhance or inhibit, depending on the type of the synapse, the receptor neuron's own tendency to emit electrical impulses. The synapse's effectiveness can be adjusted by the signals passing through it so that

The synapses can learn from the activity's dependence on history acts as memory **[25]**. in which they participate. This a memory, which is possibly responsible for human

The cerebral cortex in humans is a large flat sheet of neurons about 2 to 3 millimeters thick with a surface area of about 2,200 cm2, about twice the area of a standard computer keyboard. The cerebral cortex contains about 10" neurons, which is approximately the number of stars in the Milky Way." Neurons are massively connected, much more complex and dense than telephone networks. Each neuron is connected to 103 to 104 other neurons.

In total, the human brain contains approximately 1014 to 1015 interconnections Neurons communicate through a very short train of pulses, typically milliseconds in duration. The message is modulated on the pulse transmission frequency. This frequency can vary from a few to several hundred hertz, which is a million times slower than the fastest switching speed in electronic circuits. However, complex perceptual decisions such as face recognition are typically made by humans within a few hundred milliseconds. These decisions are made by a network of neurons whose operational speed is only a few milliseconds. This implies that the computations cannot take more than about 100 serial stages. In other words, the brain runs parallel programs that are about 100 steps long for such perceptual tasks. This is known as the hundred-step rule [26]. The same timing considerations show that the amount of information sent from one neuron to another must be very small (a few bits). This implies that critical information is not transmitted directly, but captured and distributed in the interconnections-hence the name, connectionist model, is used to describe A's [27].

1.4.7.2 <u>Message Passing Neural Network</u>

The Message Passing Neural Network [28] is a deep learning architecture designed for implementation in chemical, pharmaceutical, and material science contexts. They were introduced as a framework to generalize several proposed techniques [14, 24, 25, 28, 29, 37, 38], and have demonstrated state-of-the-art results on multiple related benchmarks. For the specific MPNN implementations used for experiments in this paper, the most important predecessor is the Gated Graph Sequence Neural Network (GGNN) [27].

In simplistic terms, MPNNs operate by the following mechanism: An initial set of states is constructed, one for each node in the graph. Then, each node is allowed to exchange information, to “message”, with its neighbors. After one such step, each node state will contain awareness of its immediate neighborhood. Repeating the step makes each node aware of its second-order neighborhood, and so forth. After a chosen number of “messaging rounds”, all these context-aware node states are collected and converted to a summary representing the whole graph. All the transformations in the steps above are carried out with neural networks, yielding a model that can be trained with known techniques to optimize the summary representation for the task at hand.

More formally, MPNNs contain three major operations: message passing, node update, and readout. Using a message passing neural network entails iteratively updating a hidden state $h_v \in \mathrm{R}^{\mathrm{D}}$, of each node v. This is done according to the following formulas:

$$m_v^{(t)} = \sum_{w \in N(v)} M_t \left(h_v^{(t)}, h_w^{(t)}, e_{vw} \right) \tag{1}$$

$$h_v^{(t+1)} = U_t \left(h_v^{(t)}, m_v^{(t)} \right) \tag{2}$$

Where M_t is the message function, U_t is the node update function, $N(v)$ is the set of neighbors of node v in the graph G, $h_v^{(t)}$ is the hidden state of the node v at time t, and $m_v^{(t)}$ is a corresponding message vector. For each atom v, messages will be passed from its neighbors and aggregated as the message vector $m_v^{(t)}$ from its surrounding environment. Then the atom is hidden state h_v is updated by the message vector.

The formula for the readout function is shown in formula 3:

$$\hat{y} = R \left(\left\{ h_v^{(K)} | v \in G \right\} \right) \tag{3}$$

where y is a resulting fixed-length feature vector generated for the graph, and R is a readout function that is invariant to node ordering, an important feature that allows the MPNN framework to be invariant to graph isomorphism. The graph feature vector y then is passed to a fully connected layer to give a prediction. All functions M_t, U_t and R are neural networks and their weights are learned during training.

1.4.7.3 <u>EMNN(Edge Memory Neural Network)</u>

The message passing concept in the MPNN framework computes the message to a center atom by aggregating information from its neighborhood atoms in a symmetric fashion. Another MPNN-inspired model in our study has a hidden state in each directed edge (every bond has two directed edges in the directed graph) instead of in the nodes. In the directed graph, each bond (node–node connection) has two directed edges, thus two hidden states. The hidden state of a directed edge is updated based on hidden states of edges whose heads coincide with its tail (Fig. 3). We call this model an Edge Memory Neural Network (EMNN). In the resulting message passing step, the update of a hidden state has a corresponding direction [28].

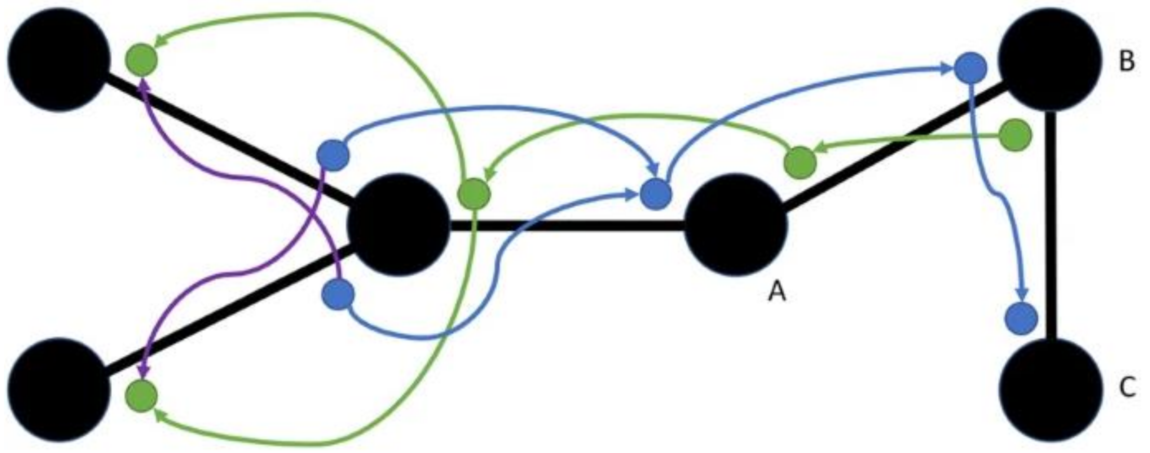


Fig.3 EMNN message passing

This model shares underlying principles with the D-MPNN architecture proposed by Yang et al. [28] which also uses directed edges to improve MPNN performance. Their proposed model also injects additional chemical descriptor information alongside the FFNN after the message passing stage. Another notable difference between these architectures is our implementation of the aforementioned attention mechanism in the aggregation function. We include the D-MPNN model in our result and discussion to compare implementations and contrast the performance benefits of additional descriptor information, as has been explored in other literature [28]. We refer to their manuscript for further details on their implementation and architecture.

One hypothetical advantage compared to MPNN is explained in the following. Consider a small graph of three nodes A, B, and C connected as A–B–C, as illustrated on the right-hand side of Fig. 3. If information passage from A to C is relevant to the task, two message passes are necessary with conventional MPNN. In the first pass, information is passed from A to B, as desired. However, information is also passed from C to B, so that part of B's memory is being occupied with information that C already has. This back-and-forth passing of information happening in an MPNN hypothetically dilutes the useful information contained in the hidden state of node B. When hidden states instead reside in the directed edges as per EMNN, this cannot happen. The closest thing corresponding to a hidden state in B is the hidden states in the edges AB, CB. The update of BC uses information from AB, but not from CB.

As shown in Fig. 3, the flow of messages in each edge is directional where the message flows from a node (tail node) to another node (head node). Formally, the set of edge hidden states taken into account when updating edge
(v,w) of the directed graph G=(V,E) is

$$S_{vw}^{(t)} = \{h_{kv} | k \in N(v), k \neq w\}.$$

In the EMNN, before message passing takes place, the two-node features are embedded into an edge feature by feeding a concatenation of the original edge and node feature vectors through a FFNN

$$e'_{vw} = f_{NN}^{emb}\left(\left(e_{vw}, h_v^{(0)}, h_w^{(0)}\right)\right)$$

At the initial state (t=0), $e_{vw}, h_v^{(0)}$ are the raw bond feature vector and atom feature vector respectively, and (,) refers to the concatenation operation.

The edge hidden state $h_{vw}^{(t)}$ of (v, w) at time t is updated according to Eqs. 8–10:

$$\begin{cases} m_{vw}^{(t)} = A_t\left(e_{vw}', S_{vw}^{(t)}\right) \\ h_{vw}^{(t+1)} = U_t\left(h_{vw}^{(t)}, m_{vw}^{(t)}\right) \end{cases} . \qquad (8)$$

Note that each directed edge has both a static edge feature e_{vw}' and the time-mutated edge state $h_{vw}^{(t)}$ contributing. $h_{vw}^{(0)}$ is instantiated as a vector of zeros. One choice of aggregation function At is

$$A_t^e\left(e'_{vw}, S_{vw}^{(t)}\right) = \sum_{x \in S'^{(t)}_{vw}} f_{NN}(x) \odot \frac{\exp(g_{NN}(x))}{\sum_{x' \in S'^{(t)}_{vw}} \exp(g_{NN}(x'))} \text{ where } S'^{(t)}_{vw} = S_{vw}^{(t)} \cup \{e'_{vw}\} \qquad (9)$$

$$h_{vw}^{(t+1)} = \text{GRU}\left(h_{vw}^{(t)}, m_{vw}^{(t)}\right) \qquad (10)$$

$m_{vw}^{(t)}$ is the message for edge (v,w) at iteration t. A_t^e is an attention-based aggregation function similar to the one used in the AMPNN. $S'^{(t)}_{vw}$ means all the edges involving nodes including the edge (v,w) itself. Equation 10 is the update of edge (v,w) using a GRU unit.
After K message passing iterations, a node hidden state for each node is taken as the sum of the edge hidden state of edges that the node is an end to,

$$h_v^{(K)} = \sum_{w \in N(v)} h_{vw}^{(K)}$$

This is done to be able to utilize the same readout functions as seen effective for the MPNNs.

1.5 Brief Description of Solution Approach

- Construction of a dataset with known bio-activities against similar proteins.
- Predictive deep learning model using an "Edge Memory Neural Network". A list of PubChem compounds with unknown activities is then used to predict activities.
- Promising compounds were then docked to the protein target using conventional molecular docking software (Autodock Vina) to return energy scores, suggesting if they may be important drug candidates.
- Highest scoring candidates are reported

1.6 Comparison of existing approaches to the problem framed

1.6.1 Traditional Vs. Al-Based Drug Discovery

Traditional	AI-based
1. Target-driven	1. Data-driven
2. Work well for easily druggable targets that have a well-defined structure and whose interactions inside the cell are understood in detail	2. Complex algorithms and machine learning can extract meaningful information from a large dataset
3. Extremely limited due to the complex nature of cellular interactions & limited knowledge of intricate cellular pathways	3. Identify compounds that could bind to 'undruggable targets,' i.e., proteins whose structures are not defined

Table 1: Comparison between Traditional and AI-based Drug Discovery Approach

1.6.2 <u>Benefits Of Applying AI To Drug Discovery</u>

The application of AI to drug discovery can revolutionize the current time scale and scope of drug discovery [30].

- AI does not rely on predetermined targets for drug discovery. Therefore, subjective bias and existing knowledge are not a factor in this drug development process.
- AI utilizes the latest advances in biology and computing to develop state-of-the-art algorithms for drug discovery. With the rapid increase in processing power and reduction in processing cost, AI has the potential to level the playing field in drug development.
- AI has a higher predictive power to define meaningful interactions in a drug screen. Therefore, the potential for false positives can be reduced by carefully designing the assay parameters in question.
- Most importantly, AI has the potential to move drug screening from the bench to a virtual lab, where results of a screen can be obtained with greater speed, and promising targets can be shortlisted without the need for extensive experimental input and manpower hours.

1.6.3 <u>Challenges Of Applying AI To Drug Discovery</u>

As is the case with any advance that brings a paradigm shift in our understanding of existing technology, AI still cannot replace a human scientist entirely in the process of drug discovery.

- AI predictions are as good as the algorithms used to investigate a dataset. The algorithm should clearly lay out the criteria used to parse out meaningful information when the results are in the 'grey zone' of interpretation.
- AI can suffer from algorithm bias, where the creators' own bias manifests itself in the way information is processed to generate predictions. Therefore, the process is not entirely objective.
- While the cost of supercomputing and high-throughput screening has decreased appreciably over the past decade, establishing these pipelines still requires significant investment.
- Ultimately, a computer's predictions have to be verified by scientists to make sure they are valid.

CHAPTER -2 LITERATURE SURVEY

2.1 Literature Overview

Michael Withnall et al. in their research paper explained the drug discovery using Attention and Edge Memory schemes to the existing Message Passing Neural Network framework for graph convolution, and benchmark the approaches against eight different physical-chemical and bioactivity datasets from the literature. They removed the need to introduce a priori knowledge of the task and chemical descriptor calculation by using only fundamental graph-derived properties. Their results consistently performed on par with other state-of-the-art machine learning approaches and set a new standard on sparse multi-task virtual screening targets.

Altae-Tran et al. in their research explained the applicability of the techniques using deep learning has been limited by the requirement for large amounts of training data. In this work, they demonstrated how one-shot learning can be used to significantly lower the amounts of data required to make meaningful predictions in drug discovery applications. They introduce a new architecture, the iterative refinement of long short-term memory, that, when combined with graph convolutional neural networks, significantly improves the learning of meaningful distance metrics over small molecules.

Rohrer et al. used refined nearest neighbor analysis which was recently introduced for the analysis of virtual screening benchmark data sets. It constitutes a technique from the field of spatial statistics and provides a mathematical framework for the nonparametric analysis of mapped point patterns. Here, refined nearest neighbor analysis is used to design benchmark data sets for virtual screening based on PubChem bioactivity data. A workflow is devised that purges data sets of compounds active against pharmaceutically relevant targets from unselective hits. Topological optimization using experimental design strategies monitored by refined nearest neighbor analysis functions is applied to generate corresponding data sets of actives and decoys that are unbiased with regard to analog bias and artificial enrichment. These data sets provide a tool for Maximum Unbiased Validation (MUV) of virtual screening methods.

Adam Paszke et al described an automatic differentiation module of PyTorch — a library designed to enable rapid research on machine learning models. It builds upon a few projects, most notably Lua Torch, Chainer, and HIPS Autograd, and provides a high-performance environment with easy access to automatic differentiation of models executed on different devices (CPU and GPU). To make prototyping easier, PyTorch does not follow the symbolic approach used in many other deep learning frameworks, but focuses on differentiation of purely imperative programs, with a focus on extensibility and low overhead.

Lindelöf et al. explained the lengthy and expensive process of developing new medicines. Which is a driving force in the development of machine learning on molecules. Classical approaches involve extensive work to select the right chemical descriptors to use as input data. They tried to explain how neural network architectures learn directly on raw molecular graphs, thereby eliminating the feature engineering step. The starting point of experimentation is a reimplementation of the previously proposed message-passing neural networks framework for learning on graphs, analogous to convolutional neural networks in how it updates node hidden states through the aggregation of neighborhoods. Three modifications of models in this framework were proposed and evaluated: employment of a recently introduced activation function, a neighborhood aggregation step involving weighted averaging, and a message-passing model incorporating hidden states in the graph's directed edges instead of its nodes. The resulting models were hyperparameter optimized using a parallelized variant of Bayesian optimization.

Wan F. and Zeng J. described that drug-protein interactions have a vital role in the success of therapy. The prediction of the interaction of a drug with a receptor or protein is essential to understand its efficacy and effectiveness, allows the repurposing of drugs, and prevents polypharmacology. Various AI methods have been useful in the accurate prediction of ligand-protein interactions, ensuring better therapeutic efficacy 55, 59. Wang et al. reported a model using the SVM approach, trained on 15 000 protein-ligand interactions, which were developed based on primary protein sequences and structural characteristics of small molecules to discover nine new compounds and their interaction with four crucial targets.

Yu et al. exploited two RF models to predict possible drug-protein interactions by the integration of pharmacological and chemical data and validating them against known platforms, such as SVM, with high sensitivity and specificity. Also, these modes were capable of predicting drug-target associations that could be further extended to target–disease, and target–target associations, thereby speeding up the drug discovery process. Xiao et al. adopted the Synthetic Minority Over-Sampling Technique and the Neighborhood Cleaning Rule to obtain optimized data for the subsequent development of iDrugTarget. This is a combination of four sub predictors (iDrug-GPCR, iDrug-Chl, iDrug-Enz, and iDrug-NR) for identifying interactions between a drug and G-protein-coupled receptors (GPCRs), ion channels, enzymes, and nuclear receptors (NR) respectively. When this predictor was compared with existing predictors through target-jackknife tests, the former surpassed the latter in terms of both prediction accuracy and consistency.

Li X. explained that drug-protein interactions can also predict the chances of polypharmacology, which is the tendency of a drug molecule to interact with multiple receptors producing off-target adverse effects. Reddy A.S. and Zhang S. stated that AI can design a new molecule based on the rationale of polypharmacology and aid in the generation of safer drug molecules. Achenbach J. explained AI platforms such as SOM, along with the vast databases available, can be used to link several compounds to numerous targets and off-targets. Bayesian classifiers and SEA algorithms can be used to establish links between the pharmacological profiles of drugs and their possible targets.

Li *et al.* demonstrated the use of KinomeX, an AI-based online medium using DNNs for the detection of polypharmacology of kinases based on their chemical structures. This platform uses DNN trained with ~14 000 bioactivity data points developed based on >300 kinases. Thus, it has practical application in studying the overall selectivity of a drug towards the kinase family and particular subfamilies of kinases, thus helping to design novel chemical modifiers. This study used NVP-BHG712 as a model compound to predict its primary targets and also its off-targets with reasonable accuracy. One prominent instance is Cyclica's cloud-based proteome-screening AI platform, Ligand Express, which is used to find receptors that can interact with a particular small molecule (the molecular description of which is in SMILE string) and produce on and off-target interactions. This helps in understanding the possible adverse effects of the drug.

Hessler G. and Baringhaus K.-H explained that *AI in* de novo *drug design* Over the past few years, the *de novo* drug design approach has been widely used to design drug molecules. The traditional method of *de novo* drug design is being replaced by evolving DL methods, the former having shortcomings of complicated synthesis routes and difficult prediction of the bioactivity of the novel molecule. Corey E. and Wipke W.T. stated Computer-aided synthesis planning can also suggest millions of structures that can be synthesized and also predicts several different synthesis routes for them.

Grzybowski *et al.* developed the Chematica program, now renamed Synthia, which has the ability to encode a set of rules into the machine and propose possible synthesizing routes for eight medicinally essential targets. This program has proven to be efficient both in terms of improving the yield and reducing expenses. It is also capable of providing alternate synthesizing strategies for patented products and is said to be helpful in the synthesis of compounds that have not yet been synthesized. Similarly, DNN focuses on rules of organic chemistry and retrosynthesis, which, with the aid of Monte-Carlo tree searches and symbolic AI, help in reaction prediction and the process of drug discovery and design, which is much faster than traditional methods.

Coley *et al.* developed a framework in which a rigid forward reaction template was applied to a group of reactants to synthesize chemically feasible products with a significant rate of reaction. ML was used to determine the dominant product based on a score given by the NNs. Putin *et al.* explored a DNN architecture called the reinforced adversarial neural computer (RANC) based on RL for *de novo* design of small organic molecules. This platform was trained with molecules represented as SMILES strings. It then generated molecules with predefined chemical descriptors in terms of MW, logP, and topological polar surface area (TPSA). RANC was compared with another platform, ORGANIC, where the former outperformed in generating unique structures without sufficient loss of their structure length.

2.2 Integrated Summary of the literature studied

Paper Name and Authors	**Topic of Interest**	**Approaches Used and Research Analysed**
Building attention and edge convolution neural networks for bioactivity and physical-chemical property prediction. Withnall, M., Lindelöf, E., Engkvist, O. and Chen, H.	Attention and Edge Memory Schemes and Message Passing Neural Network	Drug discovery using Attention and Edge Memory schemes to the existing Message Passing Neural Network framework for graph convolution, and benchmark the approaches against eight different physical-chemical and bioactivity datasets from the literature
Low Data Drug Discovery with OneShot Learning. H. Altae-Tran, B. Ramsundar, A. S. Pappu, and V. Pande	Graph Conventional Network with Long Short Term Memory	They demonstrated how one-shot learning can be used to significantly lower the amounts of data required to make meaningful predictions in drug discovery applications.
Deep Learning for Drug Discovery, Property Prediction with Neural Networks on Raw Molecular Graphs Lindelöf, E.	Edge Memory Neural Network	They tried to explain how neural network architectures learn directly on raw molecular graphs, thereby eliminating the feature engineering step. The starting point of experimentation is a reimplementation of the previously proposed message-passing neural networks framework for learning on graphs
Computational screening for active compounds targeting protein sequences: methodology and experimental validation Wang F.	Support Vector Machines	They reported a model using the SVM approach, trained on 15 000 protein-ligand interactions, which were developed based on primary protein sequences and structural characteristics of small molecules to discover nine new compounds and their interaction with four crucial targets

iDrug-Target: predicting the interactions between drug compounds and target proteins in cellular networking via benchmark dataset optimization approach Xiao X.	Synthetic Minority Over-Sampling Technique and the Neighborhood Cleaning Rule	It is a combination of four sub predictors (iDrug-GPCR, iDrug-Chl, iDrug-Enz, and iDrug-NR) for identifying interactions between a drug and G-protein-coupled receptors (GPCRs), ion channels, enzymes, and nuclear receptors (NR) respectively
Cyclica Launches Ligand Express™, a Disruptive Cloud-Based Platform to Revolutionize Drug Discovery Cyclica.	Cyclica's cloud-based proteome-screening AI platform	It is an AI platform, Ligand Express, which is used to find receptors that can interact with a particular small molecule (the molecular description of which is in SMILE string) and produce on and off-target interactions.
Reinforced adversarial neural computer for de novo molecular design Putin E.	DNN architecture called the reinforced adversarial neural computer (RANC)	It is based on RL for *de novo* design of small organic molecules. This platform was trained with molecules represented as SMILES strings. It then generated molecules with predefined chemical descriptors in terms of MW, logP, and topological polar surface area (TPSA).

Table 2 : Integrated Summary of the literature studied

CHAPTER -3 REQUIREMENT ANALYSIS AND SOLUTION APPROACH

3.1 Project Description

This is a development cum research project in which we are developing a deep neural network to efficiently identify covalent inhibitors against the target protease of the COVID-19 virus. We developed an Edge-memory neural network to predict the bioactivity of existing compounds and selected the top 10 compounds with the highest bioactivities. These compounds were then fed to Auto-dock vina, a conventional molecular docking software, and their docking scores against the target protease were calculated.

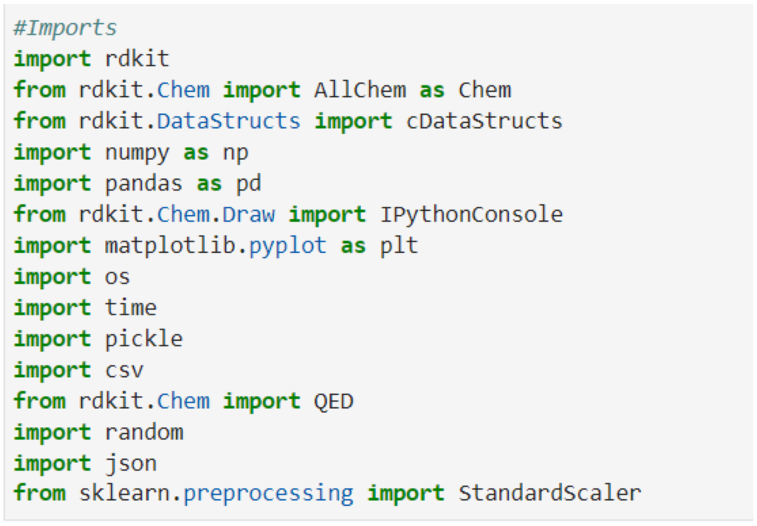
3.2 Requirement Analysis

System requirements are expressed in a software requirement document. The Software Requirements Specification (SRS) is the official statement of what is required by the system developers. This requirement document includes the requirements definition and the requirements specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

The software specification document satisfies the following: -

- It specifies the external system behavior.
- It specifies constraints on the implementation.
- It is easy to change.
- It serves as a reference tool for system maintainers.
- It records forethought about the life cycle of the system.
- It characterizes the acceptable responses to undesired events.

3.2.1 Software Requirements



```python
#Imports
import rdkit
from rdkit.Chem import AllChem as Chem
from rdkit.DataStructs import cDataStructs
import numpy as np
import pandas as pd
from rdkit.Chem.Draw import IPythonConsole
import matplotlib.pyplot as plt
import os
import time
import pickle
import csv
from rdkit.Chem import QED
import random
import json
from sklearn.preprocessing import StandardScaler
```

Fig.4 Software Requirements

1) **Pandas:** The library helps in performing data analysis and modeling, enabling the entire data processing workflow in Python. It is a great tool for data handling and manipulation as well as for reading and writing files of various formats such as CSV, TXT, XLS, and SQL databases. It helps in the flexible alignment and reshaping of data as well as filling the gaps in missing data. It also supports slicing, indexing and subsetting of large datasets [31].
2) **Numpy**: It is the fundamental package and library to perform scientific computation in Python. It provides a high-performance multidimensional array object and tools for working with these arrays. It also provides functions and tools for integration with C++ and is useful when computing and processing large amounts of data as well as calculations of useful linear algebra, Fourier transform, and random number capabilities. It has also been used as a multidimensional container for useful data and providing means to store word embeddings in a file and access them later on for insertion in the embedding layer in the encoder-decoder sequence [32].
3) **Sklearn:** Scikit-learn is an open-source Python library that implements a range of machine learning, preprocessing, cross-validation, and visualization algorithms using a unified interface. It has simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machine, random forests, gradient boosting, k-means, etc. It has been built on top of existing libraries like NumPy and is used to assist with the clustering of sentences using K Means and Birch algorithms [33].

4) **RDKit:** RDKit has a Business-friendly BSD license collection of cheminformatics and machine-learning software written in C++ and Python. It is used to draw the molecules from SMILES. It has Python 3.x wrappers generated using Boost Python and Java and C# wrappers generated with SWIG. It also supports both 2D and 3D molecular operations [34].
5) **Matplotlib:** Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It is a Python 2D plotting library that produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. One of the greatest benefits of visualization is 34 that it allows us visual access to huge amounts of data in easy-to-understand plots and graphs. Matplotlib consists of several plots like line, bar, scatter, histogram, etc [35].
6) **PyTorch:** PyTorch is a library for Python programs that facilitates building deep learning projects. It emphasizes flexibility and allows deep learning models to be expressed in idiomatic Python. This approachability and ease of use found early adopters in the research community, and in the years since its first release, it has grown into one of the most prominent deep learning tools across a broad range of applications. PyTorch provides a core data structure, the tensor, which is a multidimensional array that shares many similarities with NumPy arrays. Around that foundation, PyTorch comes with features to perform accelerated mathematical operations on dedicated hardware, which makes it convenient to design neural network architectures and train them on individual machines or parallel computing resources [36].
7) **PyMol:** PyMOL, a cross-platform molecular graphics tool, has been widely used for three-dimensional (3D) visualization of proteins, nucleic acids, small molecules, electron densities, surfaces, and trajectories. It is also capable of editing molecules, ray tracing, and making movies. This Python-based software, alongside many Python plugin tools, has been developed to enhance its utilities and facilitate the drug design in PyMOL [37].
8) **AutoDock Vina:** AutoDock Vina is an open-source program for doing molecular docking. AutoDock is a molecular modeling simulation software. It is especially effective for protein-ligand docking and virtual screening. Vina uses a sophisticated gradient optimization method in its local optimization procedure to give us the docking score [38].

3.2.2 Hardware Requirements

For this project, we required a strong Graphic card from the Geforce series. We used the NVIDIA Geforce MX250. The Nvidia GeForce MX250 is a dedicated entry-level mobile graphics card for laptops. It is based on the same Pascal GP108 chip as the predecessor, the GeForce MX150 / desktop GeForce GT 1030 but features increased clock speeds. As with the MX150, the MX250 is available in two versions, the normal 25-Watt version (1D13 device ID) and a low power version with 10 Watt TDP (1D25 device ID) and reduced performance (The MX150 with 10W had a 32% lower clock speed). The GDDR5 memory interface now also supports speeds up to 3.5 GHz = 7 GHz effective (up from 3 GHz).

3.3 Solution Approach

The First Step was finding the crystal structure of the main protease of the virus which can be obtained from the findings of Liu et al., found at https://www.rcsb.org/structure/6LU7. The structure is complexed with a ligand called N3, which serves as an excellent starting point for new drug candidate investigations.

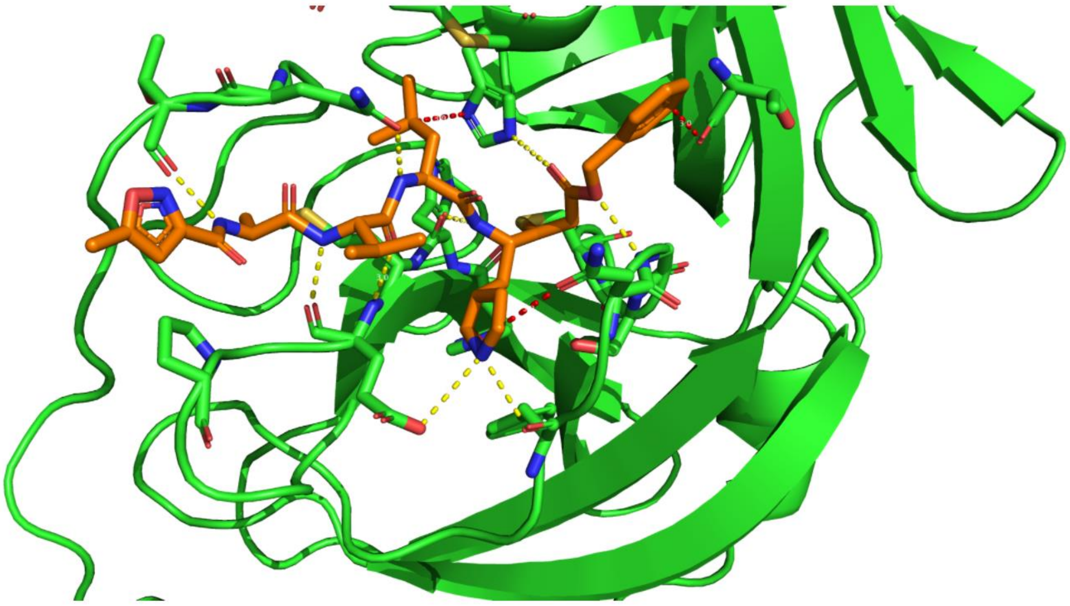


Fig.5 Visualization of N3 Ligand

The above figure shows the interactions happening in the binding site between the N3 ligand and the protein, with yellow interactions being Hydrogen Bonds, and Red interactions being sites where hydrogen bonds could be possible but are not occurring in the structure. It is also worth mentioning that the ligand shown here is what is called a "covalent inhibitor" meaning it is chemically bound to the protein. This can be seen on the right side of the orange ligand molecule, where it is connected to the yellow sulfur atom of the protein

After constructing the dataset with the bio-activity against proteins similar to COVID-19 proteases like SARS, MERS, Hepatitis, and HIV. The dataset was preprocessed for using it in the predictive model by scaling the values of activity to have mean 0 and unit variance. After then building a predictive model using the edge memory neural network technique to predict the activity of these ligands against the protease of COVID-19 so that we can identify efficient covalent inhibitors.

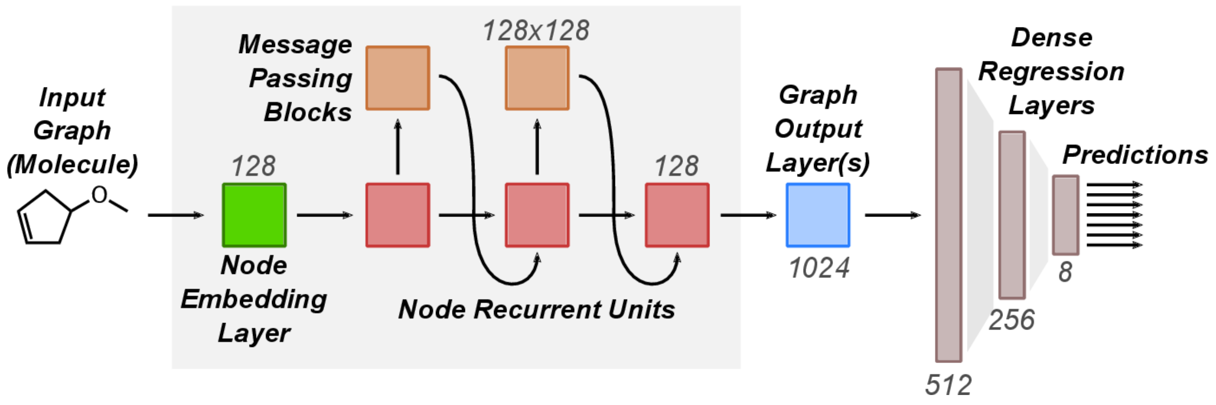


Fig.6 Edge Memory Neural Network process

The Edge Memory Neural Network is a message-passing neural network that has a hidden state in each directed edge instead of in the nodes. After the neural network is trained the N3 ligand from the crystal structure is extracted and its smiles string was searched on PubChem to find similar structures, which were then saved and their activities can be predicted using the newly trained model.

From these predicted ligands we will be selecting the top 10 ligands with the highest bioactivity which are then docked to the target protein of COVID-19 using conventional molecular docking software (Autodock Vina). AutoDock is a molecular modeling simulation software. It is especially effective for protein-ligand docking and virtual screening. Vina uses a sophisticated gradient optimization method in its local optimization procedure to give us the docking score. Docking Score is the scoring function used to predict the binding affinity of both ligand and target once it is docked. Higher the docking score better the inhibiting quality exhibited by a ligand. These docking scores would then be compared with the already known docking score of the N3 ligand that is 7.9 kcal/mol. Best compounds would then be reported as potential drugs against Covid-19.

CHAPTER -4 MODELING AND IMPLEMENTATION DETAILS

4.1 Control Flow Diagram

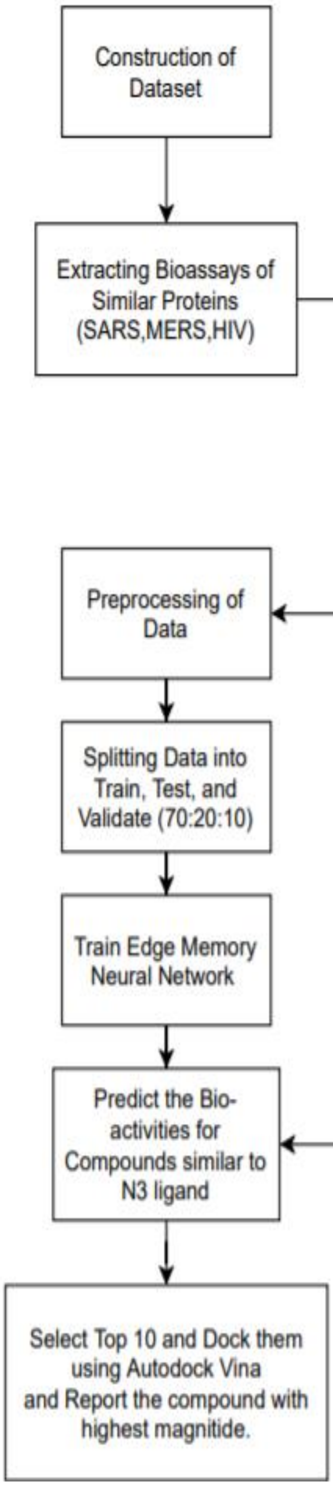
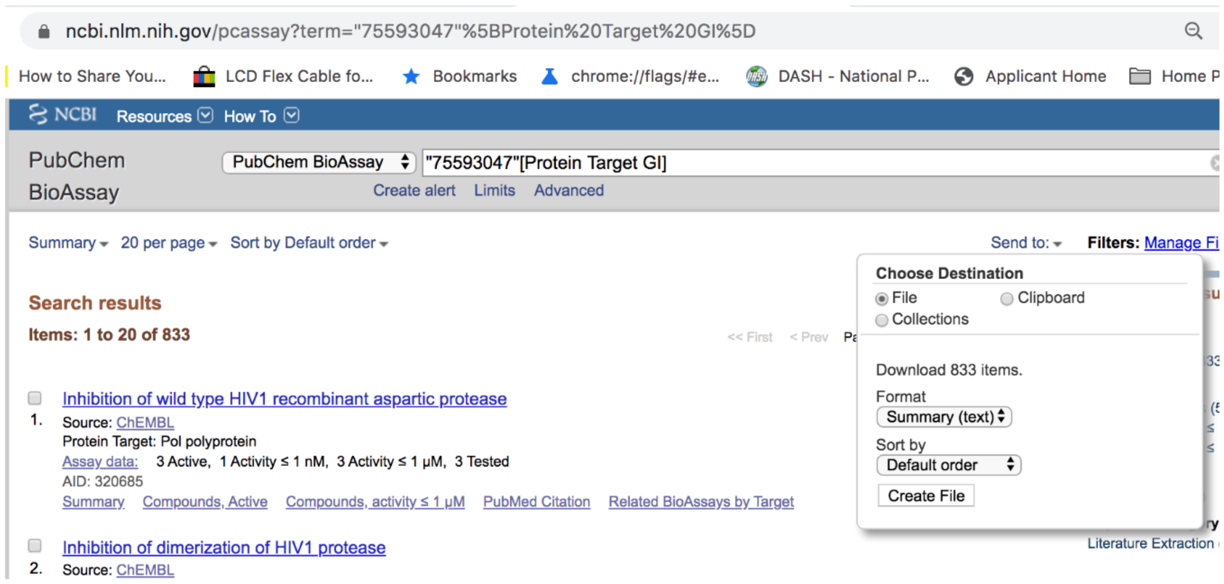

Construction of Dataset
Extracting Bioassays of Similar Proteins (SARS,MERS,HIV)
Selecting Bio-activities with a threshold of 1 micro-molar
Selecting the Bio-activities with only IC50 activity score
Preprocessing of Data
Splitting Data into Train, Test, and Validate (70:20:10)
Train Edge Memory Neural Network
Download the Similar compounds to N3 Ligand
Predict the Bio-activities for Compounds similar to N3 ligand
Select Top 10 and Dock them using Autodock Vina and Report the compound with highest magnitide.

Fig.7 Flow Chart

4.2 Implementation details and issues

1. General Dataset Preparation

The dataset generation begins by simply searching the NCBI website, a bioassay search found at https://www.ncbi.nlm.nih.gov/pcassay/advanced, to try and find relevant assays. Then running python commands to extract only the assay ID's of the results.


ncbi.nlm.nih.gov/pcassay?term="75593047"%5BProtein%20Target%20GI%5D

How to Share You... LCD Flex Cable fo... Bookmarks chrome://flags/#e... DASH - National P... Applicant Home Home P

NCBI Resources How To

PubChem BioAssay | PubChem BioAssay | "75593047"[Protein Target GI]

Create alert Limits Advanced

Summary 20 per page Sort by Default order | Send to: | Filters: Manage Fi

Search results

Items: 1 to 20 of 833

<< First < Prev Pa

1. Inhibition of wild type HIV1 recombinant aspartic protease
Source: ChEMBL
Protein Target: Pol polyprotein
Assay data: 3 Active, 1 Activity ≤ 1 nM, 3 Activity ≤ 1 µM, 3 Tested
AID: 320685
Summary Compounds, Active Compounds, activity ≤ 1 µM PubMed Citation Related BioAssays by Target

2. Inhibition of dimerization of HIV1 protease
Source: ChEMBL

Choose Destination
File Clipboard
Collections

Download 833 items.

Format
Summary (text)

Sort by
Default order

Create File

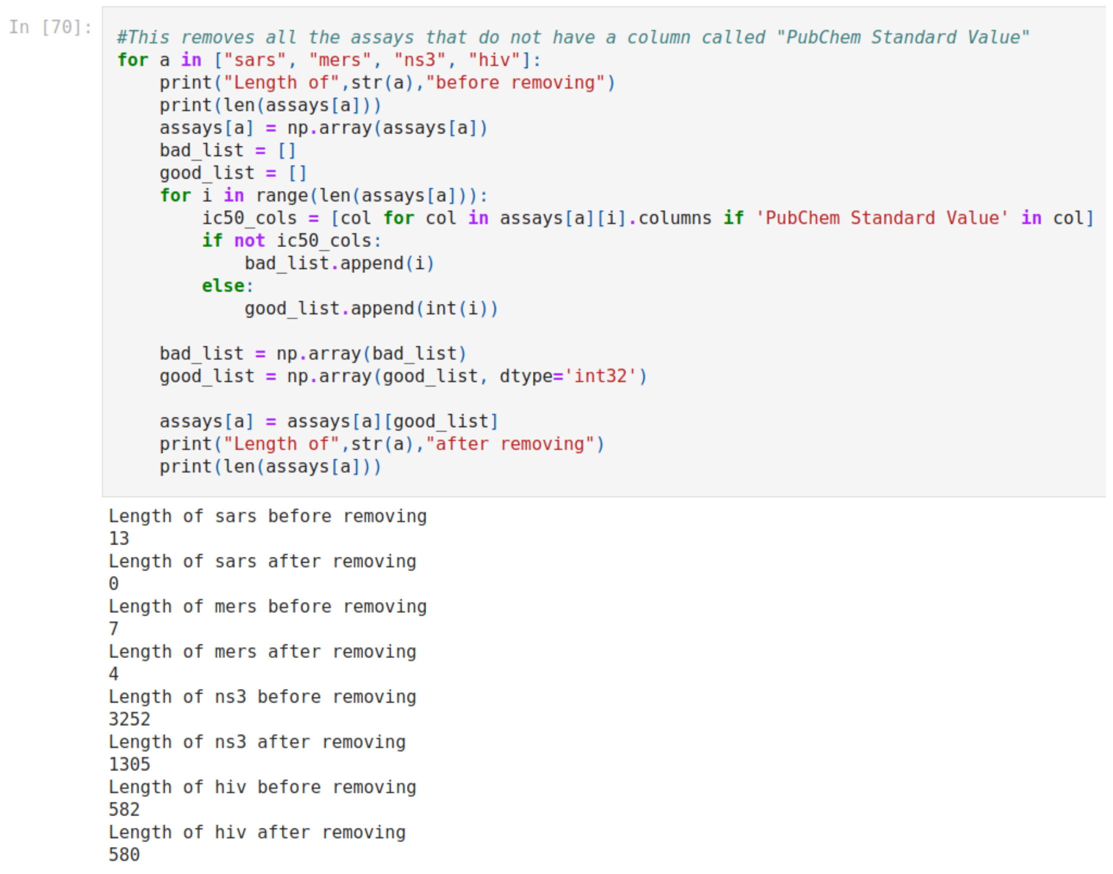
Literature Extraction

Fig.8 NCBI Website

SARS and MERS are both coronavirus variants that are very similar and since their respective outbreaks, many biological assays have been done to test compounds on their main proteases. Bioactivities measured in papers by medicinal chemists and biochemists are tracked by The National Center for Biotechnology Information (NCBI) and are freely available. A database of protease inhibitors will be built using this data. The searches we used to generate a good AID list are:

1. Protein target GI73745819 - SARS Protease
2. Protein target GI75593047 - HIV pol polyprotein
3. NS3 - Hep3 protease
4. 3CL-Pro - Mers Protease

Also for this project, we focused on the "PubChem Standard Value" which is normally a standardized value using some metric.



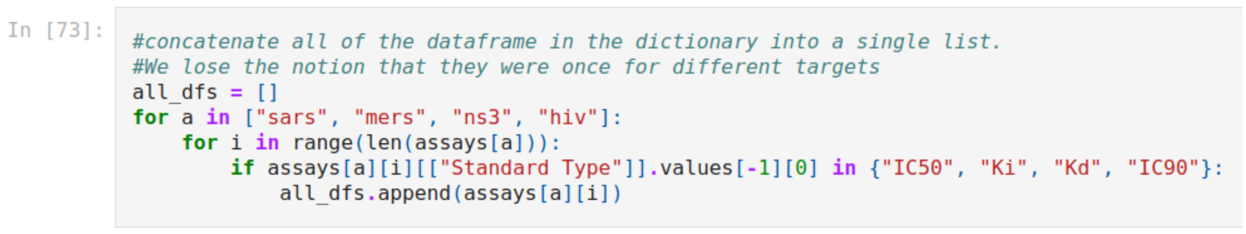
```
In [70]:
#This removes all the assays that do not have a column called "PubChem Standard Value"
for a in ["sars", "mers", "ns3", "hiv"]:
    print("Length of",str(a),"before removing")
    print(len(assays[a]))
    assays[a] = np.array(assays[a])
    bad_list = []
    good_list = []
    for i in range(len(assays[a])):
        ic50_cols = [col for col in assays[a][i].columns if 'PubChem Standard Value' in col]
        if not ic50_cols:
            bad_list.append(i)
        else:
            good_list.append(int(i))

    bad_list = np.array(bad_list)
    good_list = np.array(good_list, dtype='int32')

    assays[a] = assays[a][good_list]
    print("Length of",str(a),"after removing")
    print(len(assays[a]))
```

```
Length of sars before removing
13
Length of sars after removing
0
Length of mers before removing
7
Length of mers after removing
4
Length of ns3 before removing
3252
Length of ns3 after removing
1305
Length of hiv before removing
582
Length of hiv after removing
580
```

Fig.9 Dataset Acquisition

There are different kinds of Bioactivities that an assay can report. Depending on what was relevant to the scientists involved in the study, various values can be used. Possibly most importantly for generating this dataset though is to not confuse the different kinds of activities. We will focus on IC50, which is the concentration of the compound at which 50% inhibition is observed. The value is normally reported as a "Micromolar concentration". The lower the value, the better the compound is at inhibiting the protein. It is important to not be tempted to use the "Activity" reported in some assays, which is normally a % and corresponds to how much that compound inhibits the protein at a given concentration. We're sticking with IC50 because this value is very information-rich and actually many "Activity" experiments go into producing 1 IC50 value. Also, they are more easily comparable, as we don't need to standardize concentration across the assays.

As mentioned above, we will focus on only IC50 values. We know from enzyme kinetics that when a ligand binds to a protein in an uncompetitive scenario (i.e. an assay) the Ki value determined is equal to the IC50, so we can include it too. Also, the Kd value is a more general way of referring to the Ki value, so it can be included. Finally, we add IC90 values, which are defined as similar to IC50 values, but for 90% inhibition. This was done to increase the size of the dataset, knowing that it will, unfortunately, introduce bias into our model. The justification for doing this is that the units are the same and that there's a good chance that a compound with a low IC90 will have a low IC50

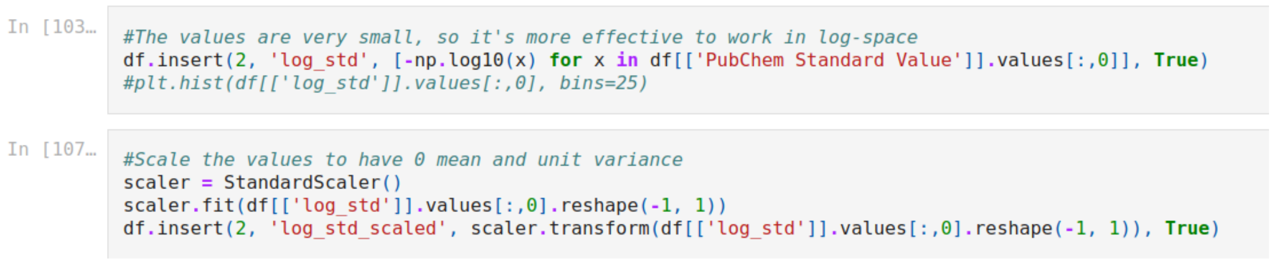


```
In [73]:  #concatenate all of the dataframe in the dictionary into a single list.
          #We lose the notion that they were once for different targets
          all_dfs = []
          for a in ["sars", "mers", "ns3", "hiv"]:
              for i in range(len(assays[a])):
                  if assays[a][i][["Standard Type"]].values[-1][0] in {"IC50", "Ki", "Kd", "IC90"}:
                      all_dfs.append(assays[a][i])
```

Fig.10 Extracting compounds that have IC50, Ki, Kd, IC90 activities.

2. Dataset Preprocessing

The dataset was preprocessed for using it in the predictive model by scaling the values of activity to have mean 0 and unit variance. The value of Activity is very small, so it was more effective to work with the log base 10 value of the activity.



```
In [103…  #The values are very small, so it's more effective to work in log-space
          df.insert(2, 'log_std', [-np.log10(x) for x in df[['PubChem Standard Value']].values[:,0]], True)
          #plt.hist(df[['log_std']].values[:,0], bins=25)

In [107…  #Scale the values to have 0 mean and unit variance
          scaler = StandardScaler()
          scaler.fit(df[['log_std']].values[:,0].reshape(-1, 1))
          df.insert(2, 'log_std_scaled', scaler.transform(df[['log_std']].values[:,0].reshape(-1, 1)), True)
```

Fig.11 Dataset Preprocessing

3. Training of Edge Memory Neural Network

Architecture of Edge Memory Neural Network:

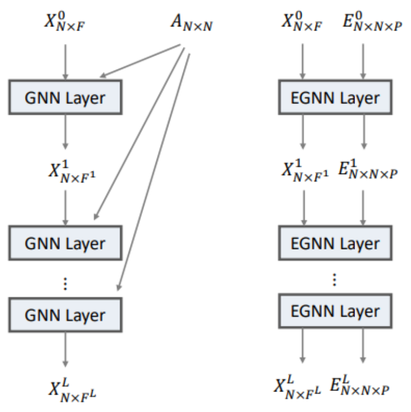
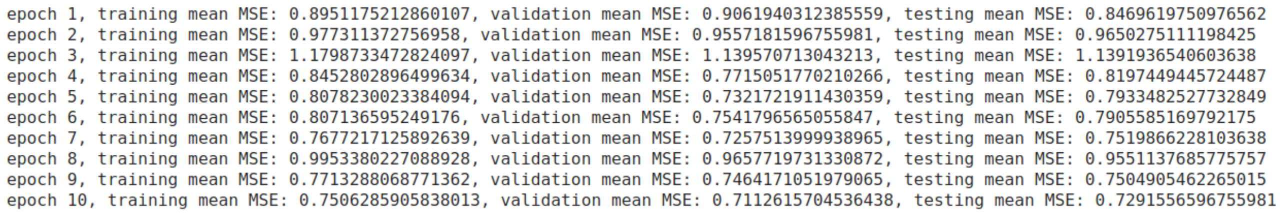


Fig.12 EMNN Architecture

Our proposed network has a multi-layer feed-forward architecture. We use superscript l to denote the output of the lth layer. The inputs to the network are denoted by X0 and E0. After passing through the first EGNN layer, X0 is filtered to produce an N × F1 new node feature matrix X1. In the meantime, edge features are adapted to E1 that preserves the dimensionality of E0. The adapted E1 is fed to the next layer as edge features. This procedure is repeated for every subsequent layer. Within each hidden layer, non-linear activations can be applied to the filtered node features Xl. The node features XL can be considered as an embedding of the graph nodes in an F L-dimensional space. For a node classification problem, a soft-max operator will be applied to each node embedding vector XL. Along the last dimension. For a whole-graph prediction (classification or regression) problem, a pooling layer is applied to the first dimension of XL so that the feature matrix is reduced to a single vector embedding for the whole graph. Then a fully connected layer is applied to the vector, whose output could be used as predictions for regression, or logits for classification. The weights of the network will be trained with supervision from ground truth labels

Training of Model:

First, we train the model using the dataset obtained after preprocessing. Then we can use the trained model to look at commercially available libraries of molecules where it would be impossible to do docking studies on each compound. In this way, the machine learning model serves to "thin the herd" of potential compounds so that we can identify candidates for docking studies.

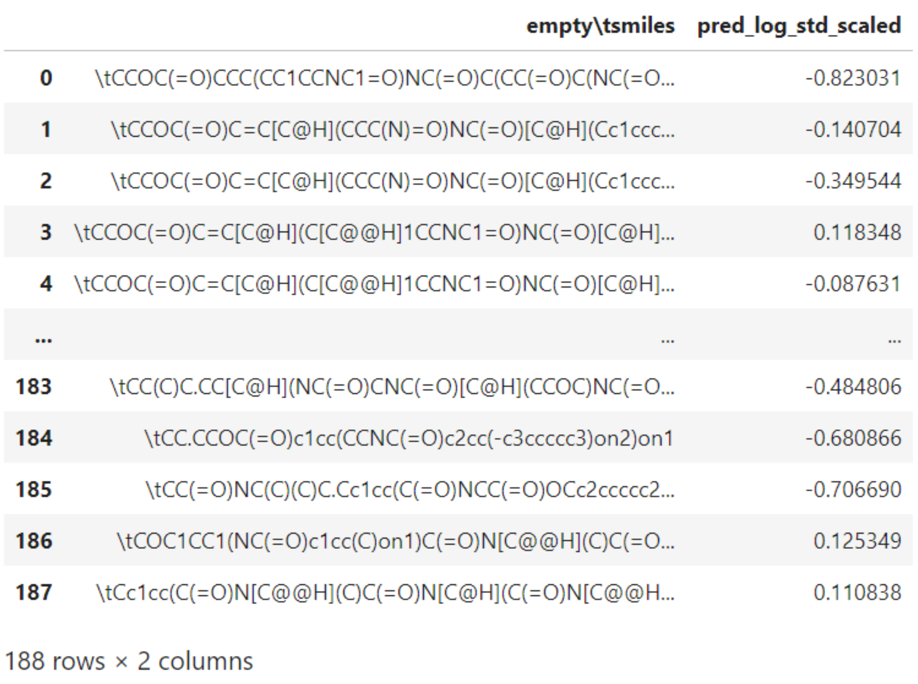


```
epoch 1, training mean MSE: 0.8951175212860107, validation mean MSE: 0.9061940312385559, testing mean MSE: 0.8469619750976562
epoch 2, training mean MSE: 0.977311372756958, validation mean MSE: 0.9557181596755981, testing mean MSE: 0.9650275111198425
epoch 3, training mean MSE: 1.1798733472824097, validation mean MSE: 1.139570713043213, testing mean MSE: 1.1391936540603638
epoch 4, training mean MSE: 0.8452802896499634, validation mean MSE: 0.7715051770210266, testing mean MSE: 0.8197449445724487
epoch 5, training mean MSE: 0.8078230023384094, validation mean MSE: 0.7321721911430359, testing mean MSE: 0.7933482527732849
epoch 6, training mean MSE: 0.807136595249176, validation mean MSE: 0.7541796565055847, testing mean MSE: 0.7905585169792175
epoch 7, training mean MSE: 0.7677217125892639, validation mean MSE: 0.7257513999938965, testing mean MSE: 0.7519866228103638
epoch 8, training mean MSE: 0.9953380227088928, validation mean MSE: 0.9657719731330872, testing mean MSE: 0.9551137685775757
epoch 9, training mean MSE: 0.7713288068771362, validation mean MSE: 0.7464171051979065, testing mean MSE: 0.7504905462265015
epoch 10, training mean MSE: 0.7506285905838013, validation mean MSE: 0.7112615704536438, testing mean MSE: 0.7291556596755981
```

Fig.13 Model Training

4. Prediction

The N3 ligand from the crystal structure was extracted and its smiles string was searched on PubChem to find similar structures, which were saved, and their activities were predicted using the model trained above.



	empty\tsmiles	pred_log_std_scaled
0	\tCCOC(=O)CCC(CC1CCNC1=O)NC(=O)C(CC(=O)C(NC(=O...	-0.823031
1	\tCCOC(=O)C=C[C@H](CCC(N)=O)NC(=O)[C@H](Cc1ccc...	-0.140704
2	\tCCOC(=O)C=C[C@H](CCC(N)=O)NC(=O)[C@H](Cc1ccc...	-0.349544
3	\tCCOC(=O)C=C[C@H](C[C@@H]1CCNC1=O)NC(=O)[C@H]...	0.118348
4	\tCCOC(=O)C=C[C@H](C[C@@H]1CCNC1=O)NC(=O)[C@H]...	-0.087631
...	...	...
183	\tCC(C)C.CC[C@H](NC(=O)CNC(=O)[C@H](CCOC)NC(=O...	-0.484806
184	\tCC.CCOC(=O)c1cc(CCNC(=O)c2cc(-c3ccccc3)on2)on1	-0.680866
185	\tCC(=O)NC(C)(C)C.Cc1cc(C(=O)NCC(=O)OCc2ccccc2...	-0.706690
186	\tCOC1CC1(NC(=O)c1cc(C)on1)C(=O)N[C@@H](C)C(=O...	0.125349
187	\tCc1cc(C(=O)N[C@@H](C)C(=O)N[C@H](C(=O)N[C@@H...	0.110838

188 rows × 2 columns

Fig.14 List of Predicted Compounds

Since we have no way of verifying the accuracy of the predicted values for these compounds, we're going to instead just take the 10 highest predicted compounds and dock them using Autodock. By doing this we actually don't even need to re-scale the scaled standard activity value, we can just take the compounds knowing that they are the compounds with the highest bioactivity.

5. Molecular Docking Studies Using Autodock Vina

First step in the validation of proposed structures is to re-dock the N3 ligand into the protease structure, to get a baseline for the energy score associated with their binding. It is important to note that the N3 ligand shown in the X-Ray structure is a covalent inhibitor, which means it actually reacts with the active site of the protein. This results in a much stronger bond between ligand and target than non-covalent inhibition. This means that the re-docked structure may not be the same as the x-ray, since it lacks the covalent bond to the protein. The following is a procedure for the docking of the N3 ligand into the protein receptor. The same procedure is used for all of the altar dockings of the candidate molecules. The procedure requires 3 programs:

- Pymol - https://pymol.org/2/ or https://github.com/schrodinger/pymol-open-source
- Autodock Vina - http://vina.scripps.edu/download.html
- Autodock Tools, found in MGL tools - http://mgltools.scripps.edu/downloads

Docking procedure with Autodock Vina:

1. Open the structure of the protein and ligand complex (.cif crystallographic information file)
2. Select the ligand chain (in the bottom right, click "residues" so that it switches to "chains" to be able to select a chain)
3. Delete the ligand, and save the file as a .pdb
4. Re-load the original file and this time select the protein and delete, saving only the ligand, also as a .pdb file
5. Open autodock tools, load the protein target molecule with File>Read Molecule (.pdb file)
6. Add hydrogens (Edit>Hydrogens>Add>Polar_only>Okay)
7. View Mol surface (mention binding site)
8. Select the 10 residues involved in the binding site: THR26,LEU27,HIS41,MET49,ASN142,CYS145, HIS163 GLU166, HIS172, GLN189
9. Go to Grid>Gridbox and show the gridbox, then manipulate it by changing the centre and size so that it's completely enclosing the selected sidechains. Remember the coordinates. They should be: centre: x=-11.963,y=15.683,z=69.193, spacing: 1A, points: x=20, y=24, z=22
10. Flexible Residues>choose molecule
11. Flexible Residues>Choose Torsions in selected residues> then accept the defaults. Should see various bonds on the 10 selected residues be different colours. THIS IS A VERY IMPORTANT STEP - NOT CONSIDERING FLEXIBILITY IN THE PROTEIN WILL AFFECT ACCURACY OF THE SCORING
12. Flexible Residues>Output>SaveRigid. Save the rigid part of the protein as a .pdbqt file
13. Flexible Residues>Output>Saveflexible. Save the flexible part of the protein as a .pdbqt file
14. Now delete or hide the receptor and load the ligand with Ligand>input>open>ligand.pdb>ok
15. Add hydrogens to the ligand edit>hydrogens>add>polar_only
16. Export this as a pdb file (with hydrogens now)
17. Re-load the updated pdb file
18. Define the rotatable bonds Ligand>TorsionTree>ChooseTorsions>okay

19. Ligand>Output> Save as PDBQT
20. Close autodock tools
21. Create a configuration file for vina that matches the structure of conf.txt found in the Docking/ directory of this repo. Exhaustiveness is proportional to time and is how thoroughly the conformational space is searched.
22. Run vina using ./vina --config conf.txt

4.3 <u>Risk Analysis and Mitigation</u>

The main limitations that often come under scrutiny when implementing the following algorithms are:

1. The algorithm processes data files of huge sizes, so running the algorithm often takes a lot of time and it wouldn't be far-fetched to call it slow.
2. The collection of data files in itself is an issue that crops up as there is a wide variety of datasets available, each yielding its own set of results so testing for the best use case dataset is important.
3. Due to an excessive amount of data the model was initially overfitting. Overfitting means that the model fits the training data "too well" relative to a set of performance criteria and exhibits poor prediction performance when tested out of the sample. We overcame the problem by implementing Dropout layers which randomly removed certain features by setting them to zero.
4. Data privacy attacks where an attacker is potentially able to infer the data set used to train the model, thereby potentially compromising the privacy of the data. An adversary could potentially infer sensitive information from the training data set by analyzing the parameters or querying the model. Two major attack types in data privacy include membership inference and model inversion attacks.
5. AI systems could potentially amplify risks relating to unfairly biased outcomes or discrimination. For example, the subjects of data ethics, fairness and the possibility of unfairly biased outcomes from the use of AI are still evolving. It is evident, however, that, depending on the use case, there is a risk that AI systems could potentially lead to unfairly biased outcomes for individuals and/or organizations. Furthermore, AI-driven unfairly biased outcomes could have privacy compliance implications, constitute regulatory, litigation and result in customer dissatisfaction and attrition.

6. Testing and validation of AI/ML systems may pose challenges relative to traditional systems as certain AI/ML systems are inherently dynamic, apt to change over time, and by extension, may result in changes to their outputs. Testing for all scenarios, permutations and combinations of available data may not be possible, thus leading to potential gaps in coverage. The severity of these gaps may vary with each system and its applications.

CHAPTER -5 TESTING

5.1 Testing Plan

While reporting evaluation metrics is certainly a good practice for quality assurance during model development, I don't think it's sufficient. Without a granular report of specific behaviors, we won't be able to immediately understand the nuance of how behavior may change if we switch over to the new model. Additionally, we won't be able to track (and prevent) behavioral regressions for specific failure modes that had been previously addressed [39].

This can be especially dangerous for machine learning systems since oftentimes failures happen silently. For example, you might improve the overall evaluation metric but introduce a regression on a critical subset of data. Or you could unknowingly add a gender bias to the model through the inclusion of a new dataset during training. We need more nuanced reports of model behavior to identify such cases, which is exactly where model testing can help.

For machine learning systems, we should be running model evaluation and model tests in parallel.

- Model evaluation covers metrics and plots which summarize performance on a validation or test dataset.
- Model testing involves explicit checks for behaviors that we expect our model to follow.

Both of these perspectives are instrumental in building high-quality models. Developing model tests for machine learning systems can offer a systematic approach towards error analysis.
There are two general classes of model tests that we'll want to write.

- Pre-train tests allow us to identify some bugs early on and short-circuit a training job.
- Post-train tests use the trained model artifact to inspect behaviors for a variety of important scenarios that we define.

5.1.1 Pre-train tests

There's some tests that we can run without needing trained parameters. These tests include:

- Checked the shape of our model output and ensured it aligns with the labels in your dataset

- Checked the output ranges and ensured it aligns with our expectations (eg. the output of a classification model should be a distribution with class probabilities that sum to 1)
- Made sure a single gradient step on a batch of data yielded a decrease in our loss
- Checked for label leakage between our training and validation datasets

The main goal here is to identify some errors early so we can avoid a wasted training job.

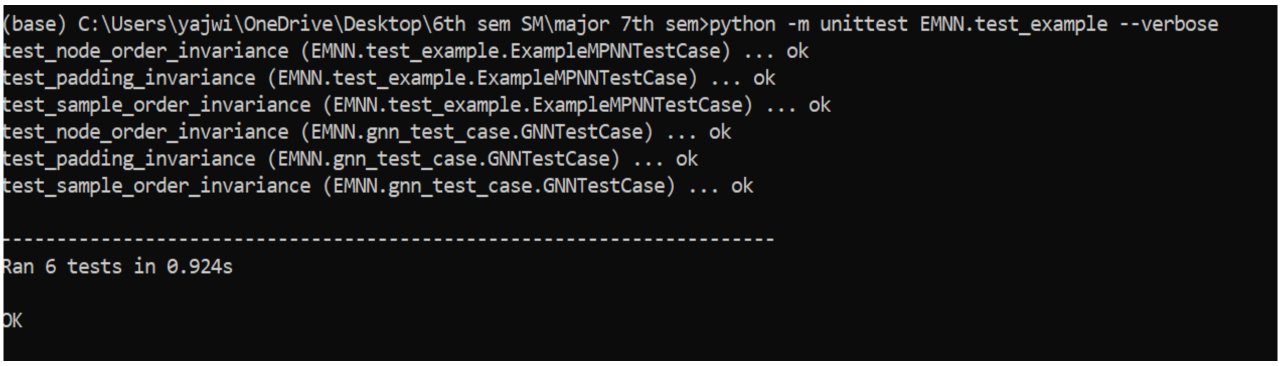
5.1.2 <u>Post-train tests</u>

However, in order for us to be able to understand model behaviors we'll need to test against trained model artifacts. These tests aim to interrogate the logic learned during training and provide us with a behavioral report of model performance.

<u>Invariance Testing</u>

Invariance tests allow us to describe a set of perturbations we should be able to make to the input without affecting the model's output. We can use these perturbations to produce pairs of input examples (original and perturbed) and check for consistency in the model predictions. This is closely related to the concept of data augmentation, where we apply perturbations to inputs during training and preserve the original label.

We wrote a script that checks whether the EMNN implementations can be initialized and trained for a few iterations without crashing, and that they fulfill the principles of invariance to **node order, padding size and sample order**.



```
(base) C:\Users\yajwi\OneDrive\Desktop\6th sem SM\major 7th sem>python -m unittest EMNN.test_example --verbose
test_node_order_invariance (EMNN.test_example.ExampleMPNNTestCase) ... ok
test_padding_invariance (EMNN.test_example.ExampleMPNNTestCase) ... ok
test_sample_order_invariance (EMNN.test_example.ExampleMPNNTestCase) ... ok
test_node_order_invariance (EMNN.gnn_test_case.GNNTestCase) ... ok
test_padding_invariance (EMNN.gnn_test_case.GNNTestCase) ... ok
test_sample_order_invariance (EMNN.gnn_test_case.GNNTestCase) ... ok

----------------------------------------------------------------------
Ran 6 tests in 0.924s

OK
```

Fig.15 Testing Model

5.2 Limitation of the solution

The limitations found in the solution are as follows:

1. When designing a GNN architecture, one easily ends up with a large number of hyperparameters. This necessitates the use of well-engineered methods to optimize them, such as a properly set up Bayesian optimization. The automation of domain specific feature engineering comes at the cost of increasing the size of this extra layer of parameters in need of tuning.
2. GNNs require a long time to train, as does any large neural network. This is despite the use of powerful GPU acceleration. During some quick sanity-check experiments during the project work a known to be fast random forest classifier has been seen to train to convergence in less than one minute on hardware that needs an hour to train a GNN. The long training time becomes especially problematic given the necessity to train many models to find good hyperparameters.
3. Using EMNN architectures is computationally expensive, and the new modifications make them more so. While in principle the methods work on any graphs, the cost makes them infeasible for graphs that are large and/or dense. It can be argued that for the specific domain of drug discovery, large computational cost is not an issue in light of the unavoidable large cost of developing any medicine. For other domains and applications, this might not be the case.

CHAPTER -6 FINDINGS, CONCLUSION, AND FUTURE WORK

6.1 Findings of the Study

6.1.1 MSE of EMNN Model

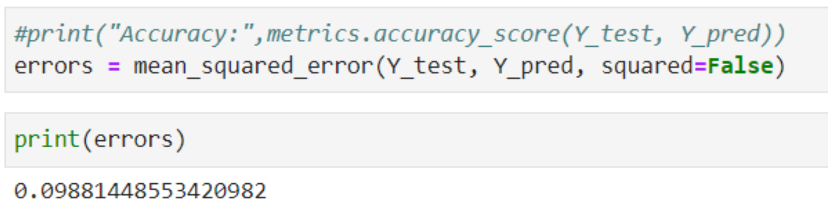
MSE is used to check how close estimates or forecasts are to actual values. This is used as a model evaluation measure for regression models and the lower value indicates a better fit. To calculate the MSE, you take the difference between your model's predictions and the ground truth, square it, and average it out across the whole dataset.

The MSE will never be negative, since we are always squaring the errors. The MSE is formally defined by the following equation:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Where N is the number of samples we are testing against.

The MSE is great for ensuring that our trained model has no outlier predictions with huge errors, since the MSE puts larger weight on these errors due to the squaring part of the function.



```
#print("Accuracy:",metrics.accuracy_score(Y_test, Y_pred))
errors = mean_squared_error(Y_test, Y_pred, squared=False)
```

```
print(errors)
```

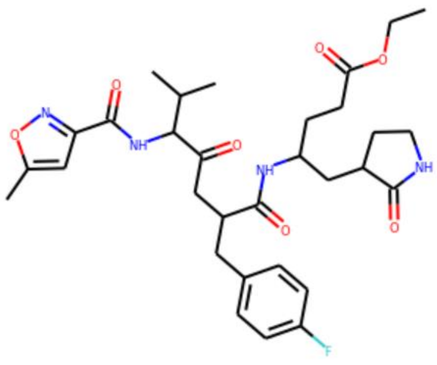
```
0.09881448553420982
```

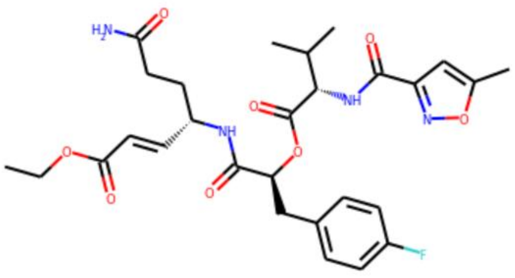
Fig.16 MSE value

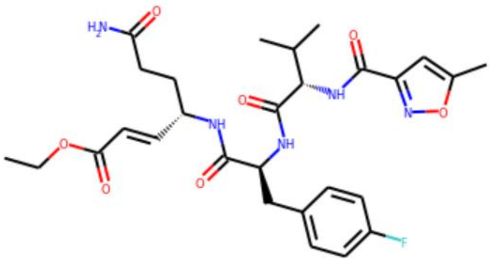
Our MSE came out to be **0.098** which is closer to 0. Lower the MSE, the closer it is forecast to actual. Thus, our model generalizes well.

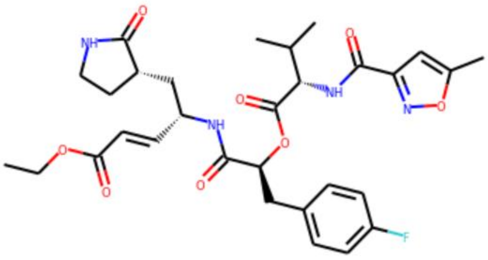
6.1.2 Docking Score

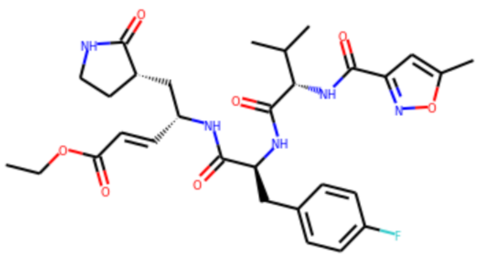
We used Auto Dock vina to calculate the docking score of each promising compound. Docking Score is the scoring function used to predict the binding affinity of both ligand and target once it is docked. Higher the docking score better the inhibiting quality exhibited by a ligand. The same procedure as mentioned in section 4.1.2, was followed for each of the candidates. The structures were saved as PDB files, to be opened in AutoDockTools.

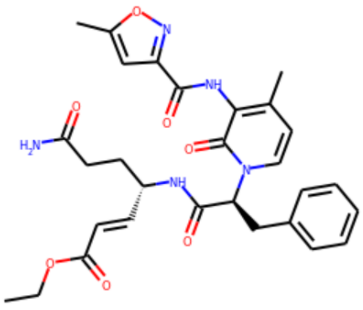

-8.2


-7.5


-8.0


-7.6


-8.5


-7.5

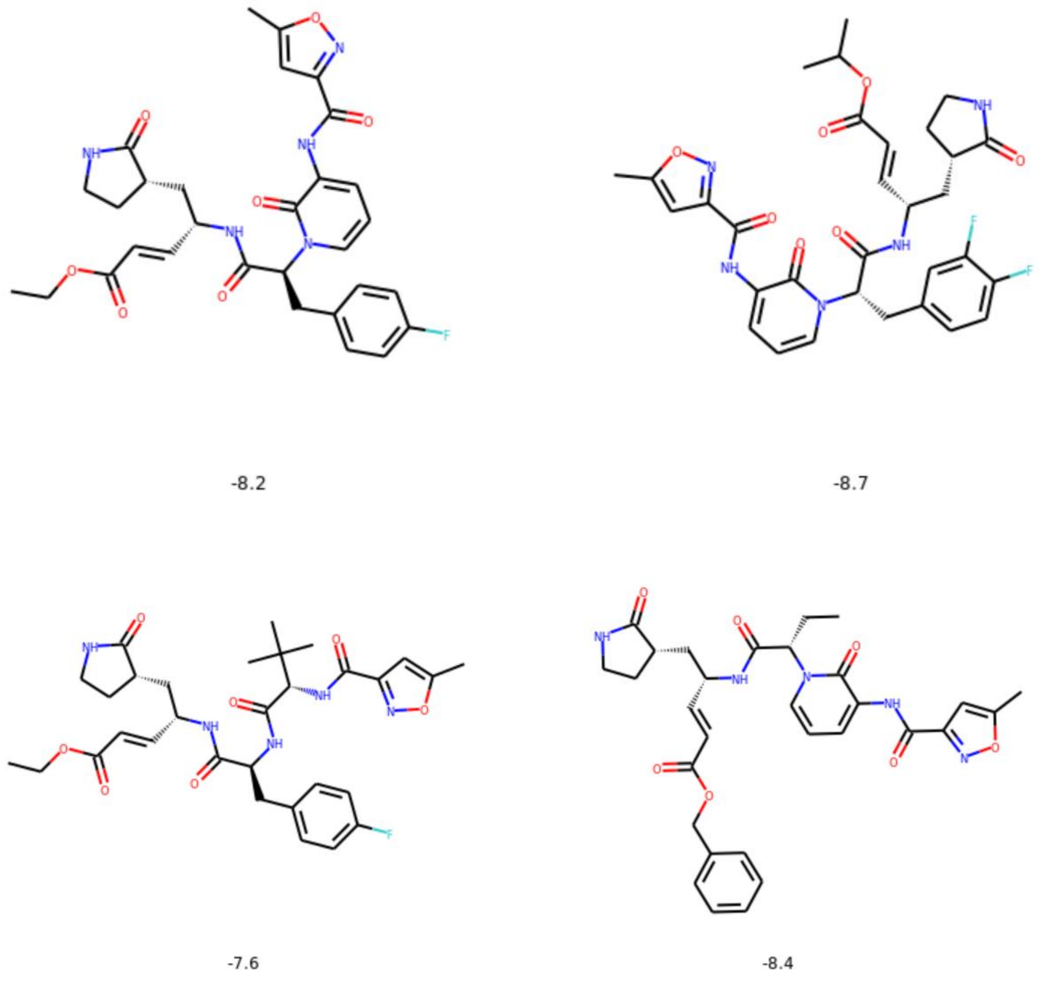


Fig-17 Top 10 promising compounds with their respective docking score

These docking scores were compared with the already known docking score of the N3 ligand that is 7.9 kcal/mol.

Choosing the best scoring compound from these promising compounds, we get the following compound from the prediction method, with a score of -8.7kcal/mol.

```
best_predicted_mols[7]
```

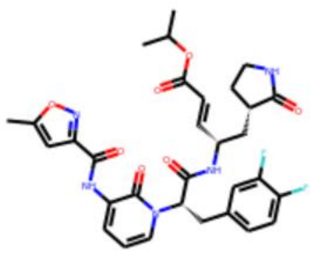


Fig.18

Ethyl(2E,4S)-4-({(2S)-3-(4-fluorophenyl)-2-[3-{[(5-methyl-1,2-oxazol-3-yl)carbonyl]amino}-2-oxo-1(2H)-pyridyl]propanoyl}amino)-5-[(3S)-2-oxo-3-pyrrolidinyl]-2-pentenoate

The compound shown above is reported as potential drugs against Covid-19.

6.1.3 Visualizing the high scoring compounds in the active site

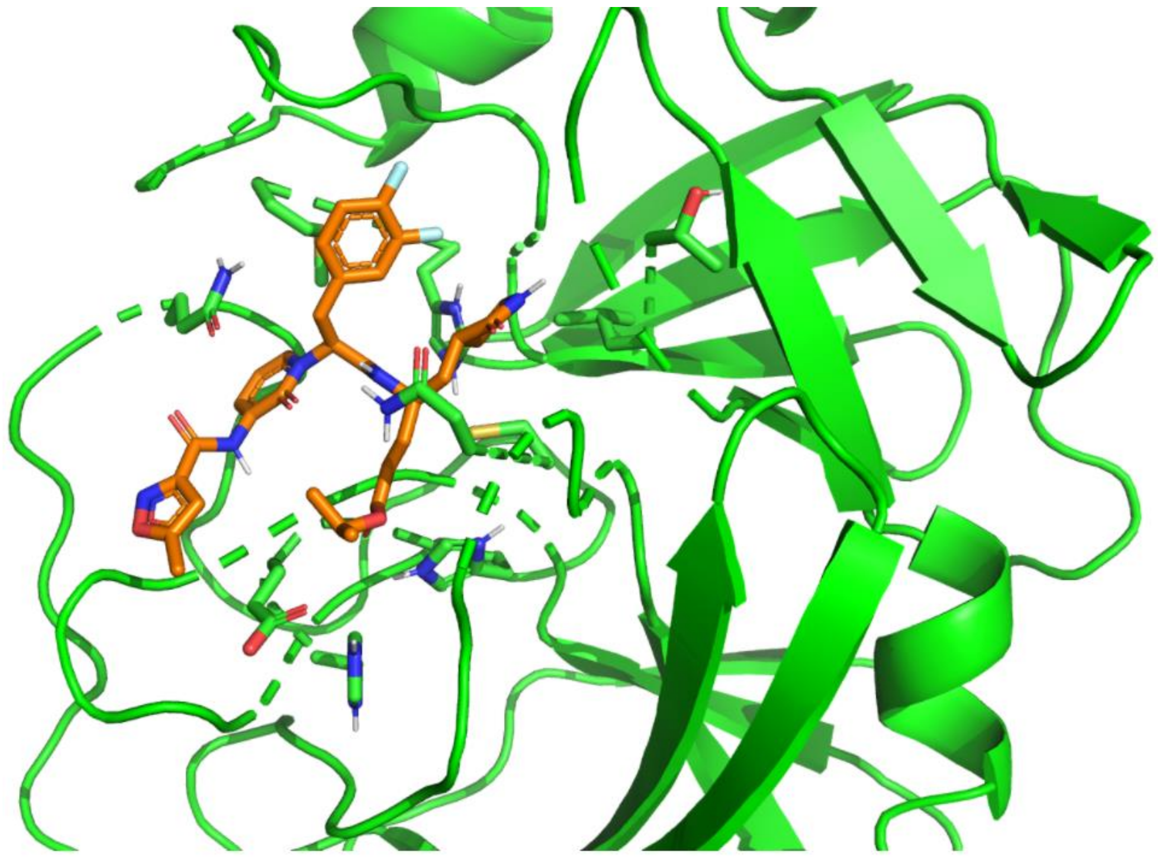


Fig.19 Visualizing best predicted compound against COVID-19

The above figure shows the interactions happening in the binding site between the highest scoring predicted compound and the protein, with yellow interactions being Hydrogen Bonds, and Red interactions being sites where hydrogen bonds could be possible, but are not occurring in the structure. This ligand chemically binds to the protein on the right side of the orange ligand molecule, where it is connected to the yellow sulfur atom of the protein.

6.2 Conclusion

The goal of this research was to develop a predictive deep learning model that could identify efficient covalent inhibitors against COVID-19 protease. The model was trained on a self-generated set of protease inhibitors, and the pubchem literature searched for 183 compounds that are similar to the n3 ligand. Predictions were made on these compounds and those with the 10 best predictive scores were docked to the ligand. The highest scoring compound is shown above and has a score of -8.7kcal/mol. We made use of the trained model to look at commercially available libraries of molecules where it would be impossible to do docking studies on each compound. In this way, the machine learning model served to "thin the herd" of potential compounds which played an instrumental role in speeding up the process of identifying candidates for docking studies. The use of Graph Neural Networks with Message Passing nodes also known as Edge Memory Neural Networks over conventional Grid Neural Networks also known as Convolutional Neural Networks, proved to be a turning point in this research work. It increased the accuracy and generalization of the model by manifolds. This played a crucial role in finding one of the best compounds(or ligand) against COVID-19 protease with a docking score of over -8.7kcal/mol much higher than the docking score of N3 ligand of about -7.8kcal/mol. This study proposes $C_{30}H_{32}FN_5O_7$
as a potential drug against COVID-19.

6.3 Future Work

In future we plan on building a generative model too.

1. Instead of taking only similar compounds to N3 ligands, we will also consider other medicinal compounds which might be useful in this research.
2. In this research work, we tried to predict the potential ligands for COVID-19 protease. In further study, we would like to make a generative model, in which we would generate potential ligands that could inhibit target protease.
3. We would like to expand this study to other virus proteases as well other than COVID-19. Thereby build a general model which can be used to predict covalent inhibitors for any virus protease in future.
4. We would like to introduce more validation evaluation criterias for the model to get better evaluation for our model.

References

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6616181/

[2] Fakoor R, Ladhak F, Nazi A, Huber M. *Using deep learning to enhance cancer diagnosis and classification*. A conference -presentation The 30th International Conference on Machine Learning, 2013. [Google Scholar]

[3] Vial A, Stirling D, Field M, et al. The role of deep learning and -radiomic feature extraction in cancer-specific predictive modeling: a review. *Transl Cancer Res* 2018;7:803–16. [Google Scholar]

[4]https://www.mckinsey.com/industries/healthcare-systems-and-services/our-insights/transforming-healthcare-with-ai

[5]https://www.market-prospects.com/articles/modernsolid

[6] Álvarez-Machancoses Ó, Fernández-Martínez J.L. Using artificial intelligence methods to speed up drug discovery. Expert Opin. Drug Discovery. 2019;14:769–777. [PubMed] [Google Scholar]

[7] Fleming N. How artificial intelligence is changing drug discovery. Nature. 2018;557 S55–S55. [PubMed] [Google Scholar]

[8] Dana D. Deep learning in drug discovery and medicine; scratching the surface. Molecules. 2018;23:2384. [PMC free article] [PubMed] [Google Scholar]

[9] Mak K.-K., Pichika M.R. Artificial intelligence in drug development: present status and future prospects. Drug Discovery Today. 2019;24:773–780. [PubMed] [Google Scholar]

[10] Zang Q. In silico prediction of physicochemical properties of environmental chemicals using molecular fingerprints and machine learning. J. Chem. Inf. Model. 2017;57:36–49. [PMC free article] [PubMed] [Google Scholar]

[11] Yang X. Concepts of artificial intelligence for computer-assisted drug discovery. Chem. Rev. 2019;119:10520–10594. [PubMed] [Google Scholar]

[12] Hessler G., Baringhaus K.-H. Artificial intelligence in drug design. Molecules. 2018;23:2520. [Google Scholar]

[13] Lusci A. Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. J. Chem. Inf. Model. 2013;53:1563–1575. [PMC free article] [PubMed] [Google Scholar]

[14] Kumar R. Prediction of human intestinal absorption of compounds using artificial intelligence techniques. Curr. Drug Discovery Technol. 2017;14:244–254. [PubMed] [Google Scholar]

[15] Rupp M. Estimation of acid dissociation constants using graph kernels. Mol. Inf. 2010;29:731–740. [PubMed] [Google Scholar]

[16] Chai S. A grand product design model for crystallization solvent design. Comput. Chem. Eng. 2020;135:106764. [Google Scholar]

[17] Thafar M. Comparison study of computational prediction tools for drug–target binding affinities. Frontiers Chem. 2019;7:1–19. [PMC free article] [PubMed] [Google Scholar]

[18] Öztürk H. DeepDTA: deep drug–target binding affinity prediction. Bioinformatics. 2018;34:i821–i829. [PMC free article] [PubMed] [Google Scholar]

[19] Lounkine E. Large-scale prediction and testing of drug activity on side-effect targets. Nature. 2012;486:361–367. [PMC free article] [PubMed] [Google Scholar]

[20] Mahmud S.H. iDTi-CSsmoteB: identification of drug–target interaction based on drug chemical structure and protein sequence using XGBoost with over-sampling technique SMOTE. IEEE Access. 2019;7:48699–48714. [Google Scholar]

[21] Gao K.Y. Interpretable drug target prediction using deep neural representation. In: Lang Jérôme., editor. Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence; IJCAI; 2018. pp. 3371–3377. [Google Scholar]

[22] Feng Q. Padme: a deep learning-based framework for drug–target interaction prediction. arXiv. 2018 arXiv:1807.09741. [Google Scholar]

[23] Karimi M. DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. Bioinformatics. 2019;35(18):3329–3338. [PMC free article] [PubMed] [Google Scholar]

[24] Pu L. eToxPred: a machine learning-based approach to estimate the toxicity of drug candidates. BMC Pharmacol. Toxicol. 2019;20:2. [PMC free article] [PubMed] [Google Scholar]

[25] https://www.cse.msu.edu/~cse802/notes/ArtificialNeuralNetworks.pdf/

[26] http://cse.msu.edu/~cse802/notes/ArtificialNeuralNetworks.pdf

[27] https://www.passeidireto.com/arquivo/49953286/artificial-neural-networks-tutorial

[28] https://jcheminf.biomedcentral.com/articles/10.1186/s13321-019-0407-y

[29]http://tesi.cab.unipd.it/40822/1/NEURAL_NETWORKS_FOR_MEDICAL_DECISION.

[30] https://doi.org/10.1038/d41586-018-05267-x

[31] https://pandas.pydata.org/

[32] https://numpy.org/

[33] https://scikit-learn.org/stable/

[34] https://www.rdkit.org/

[35] https://matplotlib.org/

[36]https://pytorch.org/

[37] https://pymol.org/2/

[38] https://vina.scripps.edu/

[39] https://www.jeremyjordan.me/testing-ml/

[40]Wan F., Zeng J. Deep learning with feature embedding for compound–protein interaction prediction. *bioRxiv.* 2016;2016 [Google Scholar]

[41]Wang F. Computational screening for active compounds targeting protein sequences: methodology and experimental validation. *J. Chem. Inf. Model.* 2011;51:2821–2828. [PubMed] [Google Scholar]

[42]Xiao X. iDrug-Target: predicting the interactions between drug compounds and target proteins in cellular networking via benchmark dataset optimization approach. *J. Biomol. Struct. Dyn.* 2015;33:2221–2233. [PubMed] [Google Scholar]

[43]Li X. Prediction of synergistic anticancer drug combinations based on drug target network and drug induced gene expression profiles. *Artif. Intell. Med.* 2017;83:35–43. [PubMed] [Google Scholar]

[44]Reddy A.S., Zhang S. Polypharmacology: drug discovery for the future. *Expert Rev. Clin. Pharmacol.* 2013;6:41–47. [PMC free article] [PubMed] [Google Scholar]

[45]Achenbach J. Computational tools for polypharmacology and repurposing. *Fut. Med. Chem.* 2011;3:961–968. [PubMed] [Google Scholar]

[46] Li Z. KinomeX: a web application for predicting kinome-wide polypharmacology effects of small molecules. *Bioinformatics.* 2019;35:5354–5356. [PubMed] [Google Scholar]

[47]Cyclica . Cyclica; 2017. Cyclica Launches Ligand Express™, a Disruptive Cloud–Based Platform to Revolutionize Drug Discovery. [Google Scholar]

[48]Corey E., Wipke W.T. Computer-assisted design of complex organic syntheses. *Science.* 1969;166:178–192. [PubMed] [Google Scholar]

[49] Grzybowski B.A. Chematica: a story of computer code that started to think like a chemist. *Chem.* 2018;4:390–398. [Google Scholar]

[49[Klucznik T. Efficient syntheses of diverse, medically relevant targets planned by computer and executed in the laboratory. *Chem.* 2018;4:522–532. [Google Scholar]

[50] Segler M.H. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature.* 2018;555:604–610. [PubMed] [Google Scholar]

[51] Putin E. Reinforced adversarial neural computer for de novo molecular design. *J. Chem. Inform. Modeling.* 2018;58:1194–1204. [PubMed] [Google Scholar]