International Journal of
Molecular Sciences

MDPI

Article

Drug Potency Prediction of SARS-CoV-2 Main Protease Inhibitors Based on a Graph Generative Model

Sarah Fadlallah [1], Carme Julià [1], Santiago García-Vallvé [2], Gerard Pujadas [2] and Francesc Serratosa [1,*]

[1] Research Group ASCLEPIUS: Smart Technology for Smart Healthcare, Departament d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili, 43007 Tarragona, Spain; sarah.fadlallah@urv.cat (S.F.); carme.julia@urv.cat (C.J.)
[2] Research Group in Cheminformatics and Nutrition, Departament de Bioquímica i Biotecnologia, Universitat Rovira i Virgili, 43007 Tarragona, Spain; santi.garcia-vallve@urv.cat (S.G.-V.); gerard.pujadas@gmail.com (G.P.)
* Correspondence: francesc.serratosa@urv.cat

Abstract: The prediction of a ligand potency to inhibit SARS-CoV-2 main protease (M-pro) would be a highly helpful addition to a virtual screening process. The most potent compounds might then be the focus of further efforts to experimentally validate their potency and improve them. A computational method to predict drug potency, which is based on three main steps, is defined: (1) defining the drug and protein in only one 3D structure; (2) applying graph autoencoder techniques with the aim of generating a latent vector; and (3) using a classical fitting model to the latent vector to predict the potency of the drug. Experiments in a database of 160 drug-M-pro pairs, from which the $pIC50$ is known, show the ability of our method to predict their drug potency with high accuracy. Moreover, the time spent to compute the $pIC50$ of the whole database is only some seconds, using a current personal computer. Thus, it can be concluded that a computational tool that predicts, with high reliability, the $pIC50$ in a cheap and fast way is achieved. This tool, which can be used to prioritize which virtual screening hits, will be further examined in vitro.

Keywords: virtual screening; graph autoencoders; graph regression; graph convolutional networks; neural networks; molecular descriptors; molecular potency; SARS-CoV-2; drug; prediction

check for updates

Citation: Fadlallah, S.; Julià, C.; Garcia-Vallvé, S.; Pujadas, G.; Serratosa, F. Drug Potency Prediction of SARS-CoV-2 Main Protease Inhibitors Based on a Graph Autoencoder and Regression. *Int. J. Mol. Sci.* **2023**, *24*, 8779. https://doi.org/10.3390/ijms24108779

Academic Editors: Kamalendra Singh and Christian Lorson

Received: 20 April 2023
Revised: 4 May 2023
Accepted: 6 May 2023
Published: 15 May 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Many efforts were made at the start of the COVID-19 pandemic to identify a drug that would stop the replication of the SARS-CoV-2 virus [1]. The main protease (M-pro) and the RNA-dependent RNA polymerase have been investigated as two major targets, and a drug for each target, nirmatrelvir (PF-07321332) [2] and remdesivir [3], has been approved by the European Medicines Agency and the U.S. Food and Drug Administration to treat COVID-19 [4].

Virtual screening (VS) and other computer-aided drug design techniques have been widely used to suggest new compounds that inhibit M-pro [5–7] and other SARS-CoV-2 targets [8–10]. A crucial component of drug research is drug potency expressed in terms of the amount required to generate an effect of a specific strength. The hit compounds suggested by a VS typically do not have enough potency to be used as drugs but may be the starting point for a process of hit optimization [11–13]. A prediction of a compound's potency would be a highly helpful addition to a VS process. The most potent compounds might then be the focus of further efforts to experimentally validate their potency and improve them. Free-energy simulations, such as free-energy perturbation, have been used to accurately predict protein–ligand free energies [14]. Nonetheless, these methods require a great amount of computing power. Specifically, their application to calculate ΔG_{bind} in VS require the use of supercomputer or cloud-computing resources (e.g., [15,16]).

Molecular graphs are an example of a very natural way to describe a set of atoms and their interactions [17–19]. A graph, in general, is a data structure depicting a collection of

entities (e.g., atoms), represented as nodes, and their pairwise relationships, represented as edges. There is a growing interest in having graph-based techniques applied to machine learning [20,21]. This can be attributed to their effectiveness in visualizing and characterizing instances of data with complex structures and rich attributes [22], capturing the inter-relationships between a system and its components.

In this paper, we propose a computational method to predict the quantitative activity of potential non-covalent inhibitors of the SARS-CoV-2 Mpro, which is quantified by the corresponding pIC_{50} (i.e., the negative log of the half maximal inhibitory concentration value when converted to molar). It is called ReGenGraph: Regression on Generated Graphs. The method's input is the crystallographic pose of a compound at the catalytic site of M-pro, but docked poses could also be used. The crystallographic or docked pose is then treated as a molecular graph. We apply graph regression techniques based on graph autoencoders (GAEs) [17,23] to predict the pIC_{50}. The drug potency prediction is achieved in two steps. First, the 3D structure of the M-pro/drug complex is converted into an interaction graph, which represents the M-pro/drug complex as a whole structure. Then, the potency value is deduced through an autoencoder and a graph autoencoder. Since the reconstruction of the interaction graph is needed for learning purposes, we can visualize the reconstructed M-pro/drug complex and verify its quality. Figure 1 summarizes the scheme of the proposed approach.

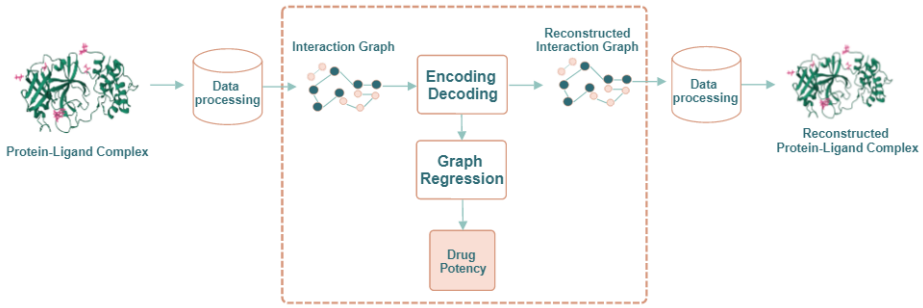

Figure 1. General scheme of our approach: ReGenGraph. The input is a protein–ligand 3D complex, while the output is their reconstructed structure and also the predicted drug potency.

In the next subsections, we briefly explain the main ideas behind autoencoders, which are the basics of our method.

1.1. Autoencoders

Autoencoders are a particular class of neural networks that are employed in machine learning to capture the most basic representations of an entity. To achieve this, they are trained to reconstruct the input data after having generated an intermediate data called latent space [24]. Autoencoders can be used for dimensionality reduction, data denoising, or anomaly detection. The obtained intermediate representations can also be used as learning tokens for classification and prediction tasks or for the generation of synthetic data.

An autoencoder consists of two components: an encoder that converts the input space into a latent space, resulting in a latent vector, $\boldsymbol{Z}$, and a decoder that converts the lower-dimensional representation back to the original input space. We define $\boldsymbol{W_0}$ and $\boldsymbol{W_1}$ as the trainable weights in the encoder and decoder, respectively. The latent space, $\boldsymbol{Z} \in \mathbb{R}^{N \times a}$, is defined by the number of entities, $\boldsymbol{N}$, (e.g., atoms in a molecule) and the features extracted in the latent space, $\boldsymbol{a}$. Encoders and decoders include non-linear activation functions. This non-linearity typically increases the expressive ability of the network and enables it to learn a range of tasks at various levels of complexity.

1.2. Graph Autoencoders

There has been a growth in using neural networks on data represented as graphs across various domains, despite the complexity of graphs that results from their inter-

twined characteristics. For the scope of this work, we focus on applications concerning drug potency prediction [25]. The currently used techniques can be divided into four categories: recurrent graph neural networks, convolutional graph neural networks, graph autoencoders, and spatial–temporal graph neural networks [22].

A graph with attributes, represented by a node attribute matrix, $\boldsymbol{X}$, and an adjacency matrix, $\boldsymbol{A}$, can be represented as **G(X,A)**, where $\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{f}}$ is a matrix of size $n \times f$, with $\boldsymbol{n}$ being the number of nodes and $\boldsymbol{f}$ being the number of attributes. The adjacency matrix, $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{n}}$, is of size $n \times n$, where $\boldsymbol{A}_{i,j} = 1$ if there is an edge between the ith and the jth node, and 0 otherwise. The graph's edges are unattributed and undirected, meaning that if there is an edge from node i to node j, there is also an edge from node j to node i, which is represented by the equality, $\boldsymbol{A}_{i,j} = \boldsymbol{A}_{j,i}$.

GAEs are based on the concept of a graph convolutional network (GCN), which, in turn, is built on the notion of generalizing convolution-like processes on normal grids, e.g., images to graph-structured data through neural network layers [17].

The key idea behind GCNs is to define the neighborhood of a node in the graph using the information from the neighboring nodes to update the node's representation. This can be accomplished by defining a convolution operation on the graph, which is typically implemented as a weighted sum of the representations of the neighboring nodes. A learnable weight matrix is often used to determine the weights of this sum, which the network learns as it updates the node's representation [26]. Node attributes can also be used to infer global properties about the graph's structure and the links between its nodes.

GAEs are composed of two main components: an encoder and a decoder. The encoder embeds input graphs through a GCN, as defined in [17], returning a latent matrix, $\boldsymbol{Z} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{b}}$, with the graph unique properties. The number of features in the latent space is $\boldsymbol{b}$. Equation (1) shows the encoder's function:

$$\boldsymbol{Z} = GCN(\boldsymbol{X}, \boldsymbol{A}) = \tilde{\boldsymbol{A}} ReLU\left(\tilde{\boldsymbol{A}} \boldsymbol{X} \boldsymbol{W}_{\boldsymbol{0}}'\right) \boldsymbol{W}_{\boldsymbol{1}}' \tag{1}$$

where $\tilde{\boldsymbol{A}}$ is a symmetrically normalized adjacency matrix computed from $\boldsymbol{A}$, while $\boldsymbol{W}_{\boldsymbol{0}}'$ and $\boldsymbol{W}_{\boldsymbol{1}}'$ are the weight matrices for each layer, which are learned through a learning algorithm. Note that *ReLU* is the classical non-negative linear equation.

The decoder is defined as Equation (2):

$$\boldsymbol{A}^* = \sigma\left(\boldsymbol{Z}\boldsymbol{Z}^{\boldsymbol{T}}\right) \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function and $\boldsymbol{T}$ means the transposed matrix. The output, $\boldsymbol{A}^*$, is a matrix of real numbers between 0 and 1 that represents the probability of an existing edge in the reconstructed adjacency matrix. Note that, in order to deduce the final reconstructed matrix, a round function is applied to $\boldsymbol{A}^*$ to discern between non-edge and edge, i.e., zero and one values.

As the aim of the GAE is to reconstruct the adjacency matrix such that it is similar to the original one, the learning algorithm minimizes the mean square distance between these matrices defined by Equation (3):

$$\boldsymbol{\mathcal{L}} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{pos} A_{i,j} log A_{i,j}^* + w_{neg} (1 - A_{i,j}) log(1 - A_{i,j}^*) \tag{3}$$

where w_{pos} and w_{neg} are introduced to deal with the value imbalance between pairs of nodes with an edge and pairs of nodes without an edge.

2. Results and Discussion

A total of 160 M-pro crystallized structures bound to a non-covalent inhibitor for which its pIC_{50} is known were used. Details about this database can be found in Section 3.1.

2.1. Molecule Reconstruction

The aim of this section is to give an example of the reconstruction of the ligand. As commented in the previous section, it seemed logical to think that a latent vector, $\boldsymbol{Z}$, is representative enough of whether the system is able to return a good approximation of the ligand it comes from.

The adjacency matrix, $\boldsymbol{A}^*$, produced by the GAE decoder (Equation (2)), is utilized to generate the bonds of the ligand. Note that the elements in $\boldsymbol{A}^*$ are real numbers between 0 and 1. As a consequence, a link between atoms i, j was to be imposed if $\boldsymbol{A}^*_{i,j} > 0.5$ and no link otherwise. Additionally, the atomic number is reconstructed by the decoder of the autoencoder.

Figure 2 illustrates the ligand *Mpro-x0830* from the selected database and the compound generated by our method. It is evident that three chemical bonds, the edges in the graph, were not reconstructed properly. For future work, the imposed bond could be set to have a maximum length. As mentioned earlier, graph reconstruction is not the primary objective but rather the ability of the latent space to capture different graph structures by reconstructing them. In this sense, it could be the case that both the original and the reconstructed compounds produce almost the same latent vector despite not being identical compounds. Therefore, the fitting module might deduce similar properties, given the compounds are not identical.

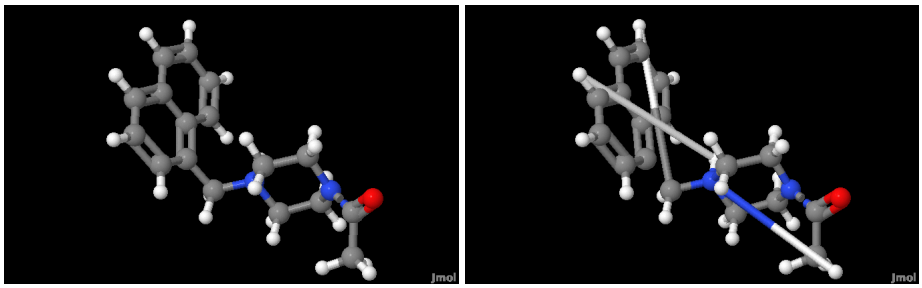

Figure 2. (**Left**) A ball-and-stick representation of ligand *Mpro-x0830*. (**Right**) The compound generated by our autoencoder and GAE. The ligand *Mpro-x0830* was randomly selected from the database.

2.2. Drug Potency Prediction

As mentioned in the Introduction section, a dual method was defined, in which the chemical compound composed of a drug and a protein is reconstructed through an autoencoder and a GAE. Since this is a novel method, the aim is to heuristically validate the need of using an autoencoder and a GAE instead of applying a classical scheme that is composed of only one of them, namely, an autoencoder or GAE.

Figure 3 shows three scatter plots of computed and experimental pIC_{50} values corresponding to the compounds in the database. In the first case, only an autoencoder and a fitting function were used. That is, the module for encoding–decoding in Figure 1 was composed of a single autoencoder. Note that, in this scenario, the bonds of the compounds are not reconstructed. In the second case, only a GAE and a fitting function were used, meaning that the encoding–decoding mechanism in Figure 1 was composed of a single GAE. In this scheme, the compound can be reconstructed. Finally, our method was applied by combining latent representations derived from both the autoencoder and the GAE to be used by the fitting function.

The first technique returns the highest mean square error (MSE), with a value of 0.82, followed by the GAE technique where the MSE = 0.77, and, finally, the proposed method, with an MSE = 0.67. These outcomes validate the architecture in a practical example, which is based on splitting the node attributes—the features of the atom—into two parts: one that is independent of the graph edges—the existence and type of bond—while the other remained dependent on them. Hence, by carefully deciding which attributes should be taken into account and which ones should be discarded, it was demonstrated that it is worthwhile to define a dual model that applies this split of attributes.

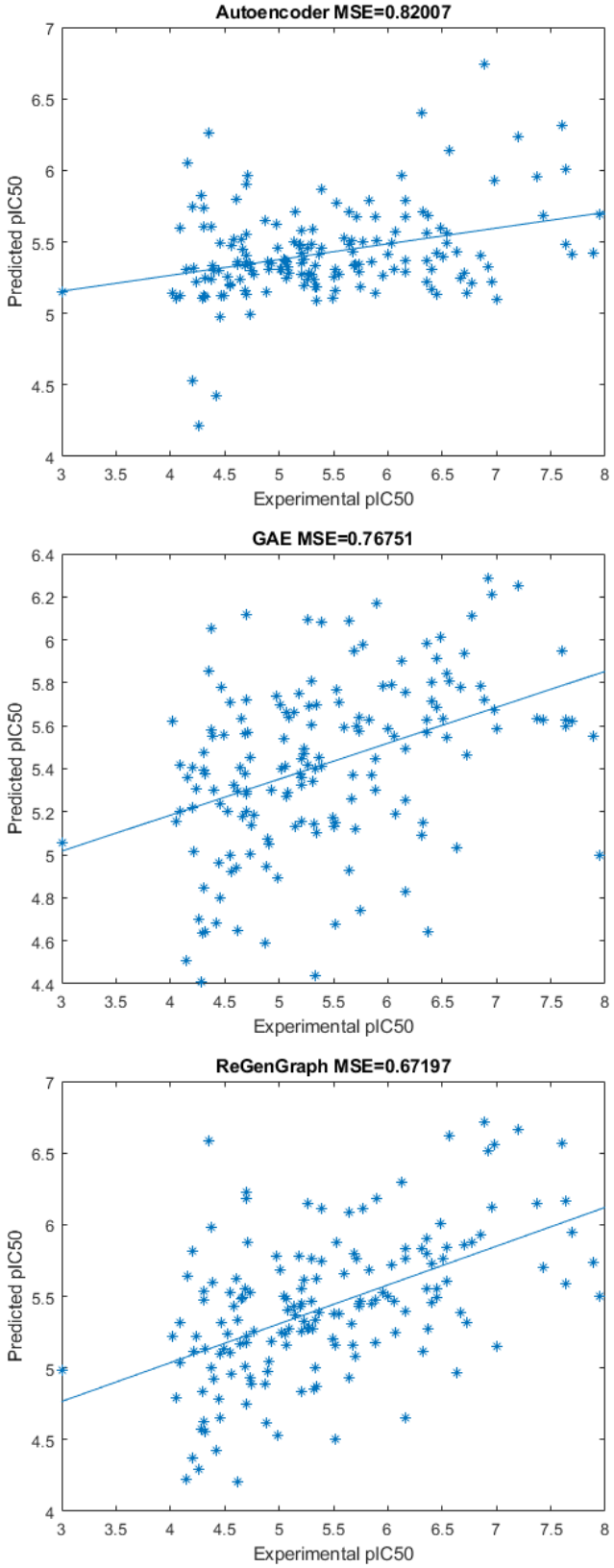

Figure 3. Three scatter plots showing the predicted and experimental pIC_{50} of the compounds in the database. From top to bottom: using only an autoencoder, using only a GAE, and ReGenGraph (proposed model). The mean square errors appear on the top of the scatters.

As practical analysis confirmed the potential of our proposal, another set of experiments was carried out. To compensate for the small size of the dataset, this experiment was conducted with the "leave-one-out" method. That is to say, all the graphs were used for training while reserving one graph for testing. This process was repeated until all the data were used for both training and testing the model. The pIC_{50} was predicted given the resulting vectors from the autoencoder, the GAE presented in [17], followed by the concatenated vectors from ReGenGraph (the proposed method).

The first column of Table 1 shows the mean of MSE and the standard deviation, given the semantic vectors resulting from the autoencoder applied to regression. The second column shows the same measures of the regression module applied to vectors obtained through a GAE, where the classical method was followed by using both semantic and structural knowledge without splitting. The third column shows the regression module applied to the ReGenGraph. The results indicate that there was a reduction in error when applying our approach in comparison to a classical one. Moreover, the standard deviation drastically decreased, which means ReGenGraph is less dependent on the data.

Table 1. MSE and standard deviation obtained by an autoencoder, GAE, and ReGenGraph (our proposal).

	Autoencoder	**GAE**	**ReGenGraph**
Mean	0.83576	0.7456	0.6717
Std. Dev.	0.3188	0.9382	0.1796

2.3. Runtime Analysis

The runtime for each training cycle varied from a few minutes to up to 40 min. **Technical specifications:** the experiments were conducted on a 2.4 GHz dual-core Intel Core i7 processor using Matlab R2022a.

3. Materials and Methods

The database is detailed in Section 3.1, our specific architecture is detailed in Section 3.2, and, finally, the learning algorithm is explained in Section 3.3.

3.1. SARS-CoV-2 M-pro Database

As mentioned previously, the dataset used consisted of 160 M-pro crystallized structures bound to an inhibitor for which its pIC_{50} is known. A total of 53 of them came from the well-know Protein Data Bank (PDB) database, and the other 107 structures came from FRAGALYSIS [27] database. Table A1 in the appendix shows a list of the M-pro crystallized structures used for training the model.

Given a pair of ligand–protein, only one attributed graph was generated. This graph represents the whole ligand and only the atoms and bonds of the protein that are close to some atom of the ligand, specifically at a distance lower than seven Å. Graph nodes are atoms of both the ligand and the protein. Graph edges represent bonds of both the ligand and the protein. Attributes on the nodes represent the three-dimensional positions of the atoms and their atomic number. Edges are unattributed, and there is an edge if there is any type of bond between atoms. The maximum number of atoms in the compound composed of "ligand + binding site atoms" is 146, and for this reason, all the generated graphs have 146 nodes.

As an example, Figure 4 shows the ligand at the x0689 FRAGALYSIS entry with (right) or without (left) the binding site environment. Note that only the parts of the Mpro with a distance smaller than 7 Å to the ligand are displayed, which is the part used for our purposes.

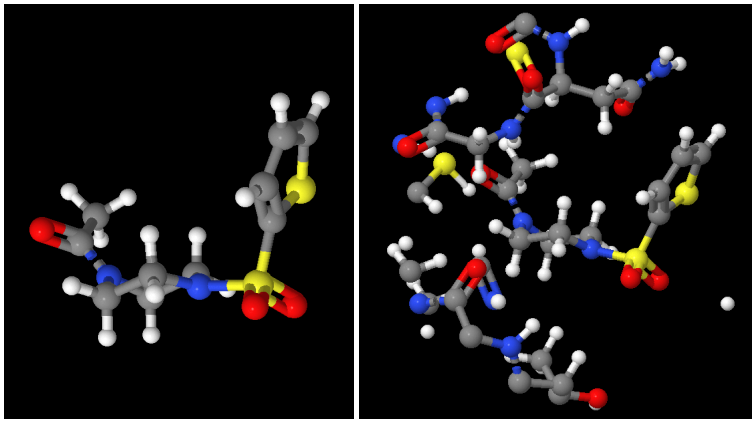

Figure 4. (**Left**): Ligand in complex x0689. (**Right**): Ligand and only the part of the Mpro close to the ligand (distance lower than 7 Å).

3.2. Architecture Configuration

The basis of the GAE approaches is the constraint that knowledge associated with nodes is related to knowledge attached to edges and vice versa [17]. That is, it is assumed that there is a relationship between the local structural pattern and the node attributes. In our case, the node attributes consist of the three-dimensional position of the atom and its atomic number. In the case of the first attribute, one can observe a clear relationship between having a bond, an edge in the graph, between two atoms, two nodes in the graph, and the proximity between these atoms. Contrarily, in the case of the second attribute, there is no relationship between the type of atom and being connected to a similar one. The contrary option would be, for instance, that oxygen tends to be connected to oxygen but not to other atoms.

The designed model was based on a GAE that handles graphs with nodes that have these two types of attributes: those that are impacted by structural patterns and those that are not related to edges. Specifically, our approach is based on two modules that work accordingly. The first one is an autoencoder [28] that captures semantic information, i.e., atomic number, without structural relations but rather by only utilizing certain node attributes. The other module is a GAE [17] that captures structural knowledge, i.e., atomic three-dimensional position, which is achieved by exploiting the remaining node attributes and edges. Both modules project their data into a latent domain, which is then used for any fitting mechanism, as shown in Figure 5. The GAE architecture defined in [29] and summarized in Section 1.2 was used. It is important to note that both the autoencoder and the GAE are used for extracting features in the encoder stage that can be used in a prediction or classification model. Nevertheless, whole models and encoder and decoder stages are also useful for reconstructing the graph.

The decision on which node attributes to use in the autoencoder and which to use in the GAE is made through a validation process. This can involve randomly selecting attributes for each architecture and determining the combination that results in the lowest loss for both. However, in specific problems, the user can make this decision based on their knowledge of the problem.

The latent space of the proposed architecture is created by combining the latent space of the autoencoder, represented as $\boldsymbol{Z_{sem}}$, and the latent space of the GAE, represented as $\boldsymbol{Z_{str}}$. Graphs are structures that must be invariant to the order of the nodes, meaning they have the property of being node-position invariant. A common way to achieve this property is by computing the sum, mean, minimum, or maximum of each feature for all nodes. The choice was settled on calculating the mean, as it makes the architecture independent of the number of nodes. Applying this mean is commonly known as the global average pooling. Then, the given $\boldsymbol{Z_{str}}$ vector $\boldsymbol{r_{str}}$ is generated by computing their mean. Note that the length of the vector, $\boldsymbol{r_{str}}$, is independent of the number of nodes, n.

This is an important feature because it means that we can fit the system with graphs that have different numbers of nodes.

Finally, the fitting module utilizes the concatenated vector composed of $\boldsymbol{Z_{sem}}$ and $\boldsymbol{r_{str}}$. This vector is used to determine the global property of the graph, which is in the current approach the drug potency.

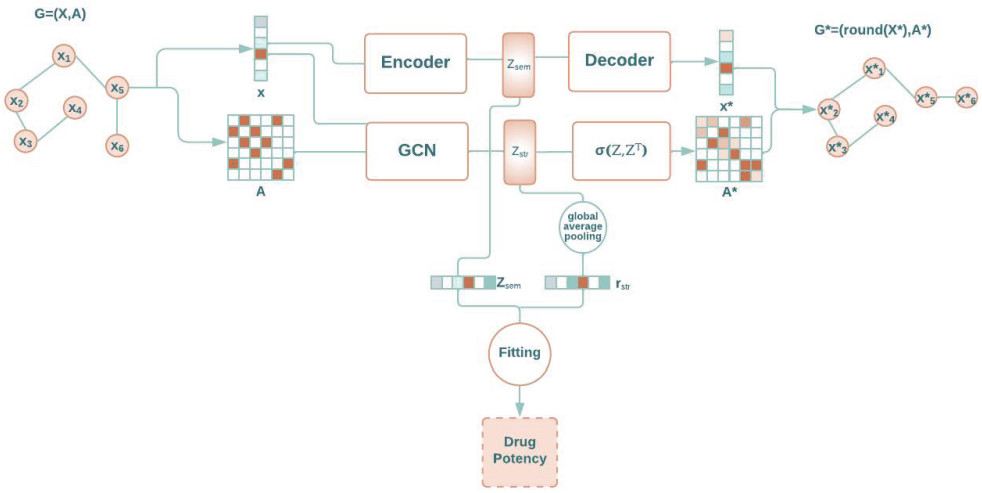

Figure 5. Schematic view of our architecture for graph regression based on an autoencoder, a graph autoencoder, and a fitting module.

The autoencoder was modeled with a fully connected neural network, which only has one hidden layer with 20 neurons, and the length of $\boldsymbol{Z_{sem}}$ is 20. The input and output layers have 146 neurons. $\boldsymbol{W_0} \in \mathbb{R}^{\mathbf{146\times 20}}$ and $\boldsymbol{W_1} \in \mathbb{R}^{\mathbf{20\times 146}}$. Moreover, the hidden layer used a sigmoid activation function, while the output layer used a linear function. The back-propagation algorithm was used for learning.

The input $\boldsymbol{X}$ of the GAE is composed of a matrix of 146 (number of nodes) times 4 (3D position + atomic number). Additionally, the input $\boldsymbol{A}$ of the GAE is composed of a square matrix of 146 times 146. $\boldsymbol{W_0'} \in \mathbb{R}^{\mathbf{146\times 20}}$ and $\boldsymbol{W_1'} \in \mathbb{R}^{\mathbf{20\times 20}}$. $\boldsymbol{Z_{sem}}$ is a matrix of 146 times 20, and thus, $\boldsymbol{r_{str}}$ is a vector that has a length of 20.

Finally, the fitting function is modeled by a classical regression. Thus, it receives a vector of 40 elements, composed of 20 elements from $\boldsymbol{Z_{sem}}$ and 20 elements from $\boldsymbol{r_{str}}$. It outputs only one real number that represents the pIC_{50}.

3.3. The Learning Process

The learning process was achieved in two steps. Initially, by both weights, $\boldsymbol{W_0}$, $\boldsymbol{W_1}$, of the autoencoder, and $\boldsymbol{W_0'}$, $\boldsymbol{W_1'}$ in the GAE are learned given all graphs G^g, where $g = 1, ..., k$. Following that, the regression weights are learned, given the returned latent vectors $\boldsymbol{Z^g_{sem}}$ and $\boldsymbol{Z^g_{str}}$ of all graphs G^g in the training set, where $g = 1, ..., k$. For the scope of this paper and its application, we focus on GAEs. More on the learning process of the autoencoder and weights, $\boldsymbol{W_0}$ and $\boldsymbol{W_1}$, can be found in the original work [24].

GAEs (Section 1.2) were modeled to reconstruct only one, usually huge, graph. Thus, the aim of the learning process, which minimizes Equation (3), is to reconstruct this unique graph. In that case, $\boldsymbol{Z}$ would have to be defined such that it resembles the inherent properties of this graph. We are in a different scenario. We wish that all latent spaces, $\boldsymbol{Z^g}$, generated by all k graphs G^g are able to reconstruct their corresponding graphs G^{*g}, given only one GAE, i.e., the same weights for all the graphs. In this way, the minimization

criterion was redefined as the sum of Equation (3) to represent the loss function of all k graphs in the dataset as expressed in Equation (4):

$$\mathcal{L} = \frac{1}{k} \sum_{g=1}^{k} \mathcal{L}^g \tag{4}$$

where

$$\mathcal{L}^g = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{pos} A_{i,j}^g log A_{i,j}^{*g} + w_{neg}(1 - A_{i,j}^g) log(1 - A_{i,j}^{*g}) \tag{5}$$

describes the loss function per each graph G^g.

4. Conclusions

Finding a fast and inexpensive method for predicting the potency of antiviral drugs against SARS-CoV-2 has been a cornerstone of research in drug discovery in the last two years. Given the experimental data on 160 M-pro/drug non-covalent complexes, this aim can be achieved by modern computational methods based on machine learning. A drug potency predictor of non-covalent ligand inhibitors was presented, which was based on two steps. The first part is the conversion of the ligand–protein complex into the interaction graph. The second is a new architecture composed of an autoencoder, a graph autoencoder, and a regression module. Additionally, a third step can be introduced to reconstruct the ligand, allowing one to visualize and evaluate the reconstructed compound.

A key aspect of our approach is the separation of the semantic and the structural knowledge of the compounds. The first is processed through the autoencoder, while the second is processed through the graph autoencoder. This main feature is independent of the application, which means that the proposed method could have different applications in other fields. The only important aspect to be considered is discerning between attributes that are dependent on the structure and attributes that are not.

Practical experiments show the ability of ReGenGraph to predict drug potency. In addition to that, they also show that the mean square error of the drug potency prediction using a graph autoencoder is larger than using our method.

In future work, we plan to test our proposal by using different architectures for the autoencoder and also to apply other fitting functions in the regression model, such as neural networks. Despite the simplicity of the chosen functions, the results are promising.

Author Contributions: Data curation, S.G.-V. and G.P.; machine learning system, S.F., C.J. and F.S. All authors have written, reviewed, and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Universitat Rovira i Virgili through the Martí Franquès program in addition to AGAUR research groups (2021SGR-00111—ASCLEPIUS: Smart Technology for Smart Healthcare; 2021SGR-00031—Quimioinformàtica i Nutrició).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The scripts and data used for this experiment can be found on GitHub, https://github.com/ASCLEPIUS-URV/Mpro-complex-GAE, accessed on 5 May 2023.

Acknowledgments: We acknowledge Guillem Macip for his collaboration on the creation of the database.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations were used in this manuscript:

GAE	Graph Autoencoder
GCN	Graph Convolutional Network
MSE	Mean Squared Error
VS	Virtual Screening

Appendix A

Table A1. List of M-pro ligands used in this study.

Source	Ligand Code
FRAGALYSIS	Mpro-x0689, Mpro-x0691, Mpro-x0755, Mpro-x0770, Mpro-x0830, Mpro-x10236, Mpro-x10322, Mpro-x10338, Mpro-x10371, Mpro-x10387, Mpro-x10417, Mpro-x10422, Mpro-x10423, Mpro-x10466, Mpro-x10535, Mpro-x10565, Mpro-x10638, Mpro-x10679, Mpro-x10789, Mpro-x10820, Mpro-x10870, Mpro-x10871, Mpro-x10876, Mpro-x10942, Mpro-x10959, Mpro-x11011, Mpro-x11271, Mpro-x11276, Mpro-x11294, Mpro-x11313, Mpro-x11317, Mpro-x11318, Mpro-x11366, Mpro-x11368, Mpro-x11454, Mpro-x11458, Mpro-x11488, Mpro-x11498, Mpro-x11499, Mpro-x11501, Mpro-x11507, Mpro-x11508, Mpro-x11530, Mpro-x11541, Mpro-x11542, Mpro-x11543, Mpro-x11548, Mpro-x11562, Mpro-x11564, Mpro-x11609, Mpro-x11612, Mpro-x11616, Mpro-x11641, Mpro-x11642, Mpro-x11723, Mpro-x11742, Mpro-x11743, Mpro-x11757, Mpro-x11764, Mpro-x11789, Mpro-x11790, Mpro-x11797, Mpro-x11798, Mpro-x11801, Mpro-x11810, Mpro-x11812, Mpro-x11813, Mpro-x11831, Mpro-x12000, Mpro-x12073, Mpro-x12143, Mpro-x12171, Mpro-x12177, Mpro-x12202, Mpro-x12207, Mpro-x12300, Mpro-x12321, Mpro-x12419, Mpro-x12423, Mpro-x12582, Mpro-x12587, Mpro-x12659, Mpro-x12661, Mpro-x12674, Mpro-x12679, Mpro-x12686, Mpro-x12692, Mpro-x12695, Mpro-x12696, Mpro-x12698, Mpro-x12699, Mpro-x12710, Mpro-x12715, Mpro-x12716, Mpro-x12731, Mpro-x12740, Mpro-x1336, Mpro-x1386, Mpro-x1418, Mpro-x2563, Mpro-x2572, Mpro-x2646, Mpro-x2649, Mpro-x2908, Mpro-x2910, Mpro-x2912, Mpro-x3303
PDB	6M2N, 6W63, 7AU4, 7B2J, 7B2U, 7B5Z, 7B77, 7E18, 7E19, 7KX5, 7L0D, 7L10, 7L11, 7L12, 7L14, 7LCT, 7LMD, 7LME, 7LMF, 7M8M, 7M8N, 7M8O, 7M8P, 7M8X, 7M8Y, 7M8Z, 7M90, 7M91, 7N44, 7N8C, 7NT3, 7O46, 7P2G, 7QBB, 7RLS, 7RM2, 7RMB, 7RME, 7RMT, 7RMZ, 7RN4, 7RNH, 7RNK, 7S3K, 7S3S, 7S4B, 7TVX, 7VIC, 7VLP, 7VLQ, 7VTH, 7VU6, 7VVP, 7VVT, 7X6K, 8ACD

References

1. Achdout, H.; Aimon, A.; Bar-David, E.; Morris, G.M. COVID Moonshot: Open Science Discovery of SARS-CoV-2 Main Protease Inhibitors by Combining Crowdsourcing, High-Throughput Experiments, Computational Simulations, and Machine Learning. *bioRxiv* **2020**. [CrossRef]
2. Owen, D.R.; Allerton, C.M.; Anderson, A.S.; Aschenbrenner, L.; Avery, M.; Berritt, S.; Boras, B.; Cardin, R.D.; Carlo, A.; Coffman, K.J.; et al. An oral SARS-CoV-2 Mpro inhibitor clinical candidate for the treatment of COVID-19. *Science* **2021**, *374*, 1586–1593. [CrossRef] [PubMed]
3. Eastman, R.T.; Roth, J.S.; Brimacombe, K.R.; Simeonov, A.; Shen, M.; Patnaik, S.; Hall, M.D. Remdesivir: A Review of Its Discovery and Development Leading to Emergency Use Authorization for Treatment of COVID-19. *ACS Cent. Sci.* **2020**, *6*, 672–683. [CrossRef] [PubMed]
4. U.S. Food and Drug Administration to Treat COVID-19. Available online: https://fda.gov/drugs/emergency-preparedness-drugs/coronavirus-covid-19-drugs (accessed on 20 March 2023).
5. Gimeno, A.; Mestres-Truyol, J.; Ojeda-Montes, M.J.; Macip, G.; Saldivar-Espinoza, B.; Cereto-Massagué, A.; Pujadas, G.; Garcia-Vallvé, S. Prediction of novel inhibitors of the main protease (M-pro) of SARS-CoV-2 through consensus docking and drug reposition. *Int. J. Mol. Sci.* **2020**, *21*, 3793. [CrossRef]
6. Macip, G.; Garcia-Segura, P.; Mestres-Truyol, J.; Saldivar-Espinoza, B.; Ojeda-Montes, M.J.; Gimeno, A.; Cereto-Massagué, A.; Garcia-Vallvé, S.; Pujadas, G. Haste makes waste: A critical review of docking-based virtual screening in drug repurposing for SARS-CoV-2 main protease (M-pro) inhibition. *Med. Res. Rev.* **2022**, *42*, 744–769. [CrossRef]
7. Macip, G.; Garcia-Segura, P.; Mestres-Truyol, J.; Saldivar-Espinoza, B.; Pujadas, G.; Garcia-Vallvé, S. A review of the current landscape of SARS-CoV-2 main protease inhibitors: Have we hit the Bullseye yet? *Int. J. Mol. Sci.* **2021**, *23*, 259. [CrossRef]

8. Siddiqui, S.; Upadhyay, S.; Ahmad, R.; Gupta, A.; Srivastava, A.; Trivedi, A.; Husain, I.; Ahmad, B.; Ahamed, M.; Khan, M.A. Virtual screening of phytoconstituents from miracle herb nigella sativa targeting nucleocapsid protein and papain-like protease of SARS-CoV-2 for COVID-19 treatment. *J. Biomol. Struct. Dyn.* **2022**, *40*, 3928–3948. [CrossRef]
9. Chan, W.K.; Olson, K.M.; Wotring, J.W.; Sexton, J.Z.; Carlson, H.A.; Traynor, J.R. In silico analysis of SARS-CoV-2 proteins as targets for clinically available drugs. *Sci. Rep.* **2022**, *12*, 5320. [CrossRef]
10. Gogoi, M.; Borkotoky, M.; Borchetia, S.; Chowdhury, P.; Mahanta, S.; Barooah, A.K. Black tea bioactives as inhibitors of multiple targets of SARS-CoV-2 (3CLpro, PLpro and RdRp): A virtual screening and molecular dynamic simulation study. *J. Biomol. Struct. Dyn.* **2022**, *40*, 7143–7166. [CrossRef]
11. Deshmukh, M.G.; Ippolito, J.A.; Zhang, C.H.; Stone, E.A.; Reilly, R.A.; Miller, S.J.; Jorgensen, W.L.; Anderson, K.S. Structure-guided design of a perampanel-derived pharmacophore targeting the SARS-CoV-2 main protease. *Structure* **2021**, *29*, 823–833.e5. [CrossRef]
12. Glaser, J.; Sedova, A.; Galanie, S.; Kneller, D.W.; Davidson, R.B.; Maradzike, E.; Del Galdo, S.; Labbé, A.; Hsu, D.J.; Agarwal, R.; et al. Hit expansion of a noncovalent SARS-CoV-2 main protease inhibitor. *ACS Pharmacol. Transl. Sci.* **2022**, *5*, 255–265. [CrossRef] [PubMed]
13. Gao, S.; Sylvester, K.; Song, L.; Claff, T.; Jing, L.; Woodson, M.; Weiße, R.H.; Cheng, Y.; Schäkel, L.; Petry, M.; et al. Discovery and crystallographic studies of trisubstituted piperazine derivatives as non-covalent SARS-CoV-2 main protease inhibitors with high target specificity and low toxicity. *J. Med. Chem.* **2022**, *65*, 13343–13364. [CrossRef] [PubMed]
14. Wang, L.; Wu, Y.; Deng, Y.; Kim, B.; Pierce, L.; Krilov, G.; Lupyan, D.; Robinson, S.; Dahlgren, M.K.; Greenwood, J.; et al. Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. *J. Am. Chem. Soc.* **2015**, *137*, 2695–2703. [CrossRef]
15. Li, Z.; Wu, C.; Li, Y.; Liu, R.; Lu, K.; Wang, R.; Liu, J.; Gong, C.; Yang, C.; Wang, X.; et al. Free energy perturbation–based large-scale virtual screening for effective drug discovery against COVID-19. *Int. J. High Perform. Comput. Appl.* **2023**, *37*, 45–57. [CrossRef]
16. Li, Z.; Li, X.; Huang, Y.Y.; Wu, Y.; Liu, R.; Zhou, L.; Lin, Y.; Wu, D.; Zhang, L.; Liu, H.; et al. Identify potent SARS-CoV-2 main protease inhibitors via accelerated free energy perturbation-based virtual screening of existing drugs-19. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 27381–27387. [CrossRef]
17. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
18. Garcia-Hernandez, C.; Fernández, A.; Serratosa, F. Ligand-based virtual screening using graph edit distance as molecular similarity measure. *J. Chem. Inf. Model.* **2019**, *59*, 1410–1421. [CrossRef]
19. Rica, E.; Álvarez, S.; Serratosa, F. Ligand-based virtual screening based on the graph edit distance. *Int. J. Mol. Sci.* **2021**, *22*, 12751. [CrossRef]
20. Garcia-Hernandez, C.; Fernandez, A.; Serratosa, F. Learning the Edit Costs of Graph Edit Distance Applied to Ligand-Based Virtual Screening. *Curr. Top. Med. Chem.* **2020**, *20*, 1582–1592. [CrossRef]
21. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph Neural Networks: A Review of Methods and Applications. *arXiv* **2018**, arXiv:1812.08434.
22. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 4–24. [CrossRef]
23. Le, T.; Le, N.; Le, B. Knowledge graph embedding by relational rotation and complex convolution for link prediction. *Expert Syst. Appl.* **2023**, *214*, 119122. [CrossRef]
24. Kramer, M.A. Nonlinear principal component analysis using autoassociative neural networks. *Aiche J.* **1991**, *37*, 233–243. [CrossRef]
25. Fadlallah, S.; Julià, C.; Serratosa, F. Graph Regression Based on Graph Autoencoders. In *Proceedings of the Structural, Syntactic, and Statistical Pattern Recognition*; Krzyzak, A., Suen, C.Y., Torsello, A., Nobile, N., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 142–151.
26. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. *arXiv* **2017**, arXiv:1710.10903.
27. Mpro: Fragalysis. Available online: https://fragalysis.diamond.ac.uk/viewer/react/preview/target/Mpro (accessed on 20 March 2023).
28. Majumdar, A. Graph structured autoencoder. *Neural Netw.* **2018**, *106*, 271–280. [CrossRef] [PubMed]
29. Kipf, T.N. Deep Learning with Graph-Structured Representations. Ph.D. Thesis, University of Amsterdam, Amsterdam, The Netherlands, 2020.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.