Εθνικό Μετσόβιο Πολυτεχνείο

Εργαστήριο Βιοϊατρικών Συστημάτων

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Διπλωματική Εργασία

Deep learning protein and molecule representations for drug design

Φοιτητής: Τερζόπουλος Παναγιώτης

Επιβλέπων καθηγητής: Λεωνίδας Αλεξόπουλος

Αναπληρωτής Καθηγητής ΕΜΠ

ΑΘΗΝΑ, ΙΟΥΛΙΟΣ 2019

Εθνικό Μετσόβιο Πολυτεχνείο

Εργαστήριο Βιοϊατρικών Συστημάτων

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Διπλωματική Εργασία

Deep learning protein and molecule representations for drug design

Φοιτητής: Τερζόπουλος Παναγιώτης

Επιβλέπων καθηγητής: Λεωνίδας Αλεξόπουλος

Αναπληρωτής Καθηγητής ΕΜΠ

Εγκρίθηκε την 1η Ιουλίου 2019 από την τριμελή επιτροπή:

Αλεξόπουλος Λεωνίδας Αναπληρωτής Καθηγητής ΕΜΠ	Αντωνιάδης Ιωάννης Καθηγητής ΕΜΠ	Κυριακόπουλος Κωνσταντίνος Καθηγητής ΕΜΠ

ΑΘΗΝΑ, ΙΟΥΛΙΟΣ 2019

Abstract

This thesis studies the ability to automatically predict the best possible ligand binders for protein kinases targets, through the use of deep learning. A large number of literature datasets on protein-ligand binding were curated and merged into one, engineered to be ready to use for machine learning and made publicly available. The ways of comparing molecules were also analyzed and a new, custom similarity metric was introduced, based on the aggregation of the different compressed molecule representations (fingerprints) and different similarity scoring rules.

Furthermore, a graph convolutional neural network (GCNN) was developed (among other models) to predict the disassociation constant between drugs and proteins and managed better accuracy than the baseline model of the literature. An ensemble of many different GCNNs was also created and proved to be able to fairly generalize even in the cold-drug case (previously unseen and structurally different molecules), where almost all of the literature models collapse entirely.

Finally, predicting confidence intervals alongside with the point predictions in deep learning was investigated. Three different methods to estimate the uncertainty of a prediction were proposed and, for the first time, directly compared in both the conventional and the cold-drug case.

Περίληψη

Σε αυτήν την εργασία μελετάται η δυνανότητα της αυτόματης πρόβλεψης των καλύτερων δυνατών μικρομορίων ως δέτες σε πρωτεΐνες, μέσα από τη χρήση αλγορίθμων μηχανικής μάθησης. Ένα μεγάλο μέρος των βάσεων δεδομένων της βιβλιογραφίας όσον αφορά τη σύνδεση φαρμάκων-πρωτεϊνών ενοποιήθηκε σε ένα και φτιάχτηκε με τέτοιο τρόπο ώστε να είναι εύκολο για χρήση σε αλγορίθμους μηχανικής μάθησης. Τα δεδομένα αυτά δημοσιοποίθηκαν για ελεύθερη χρήση. Οι τρόποι σύγκρισης μη όμοιων μορίων επίσης αναλύθηκαν κι ένας νέος, προσαρμοσμένος τρόπος μέτρησης της μοριακής ομοιότητας εισήχθη, βασισμένος στην ενοποίηση διαφορετικών συμπιεσμένων τρόπων αναπαράστασης των μορίων και διαφορετικών κανόνων βαθμολόγησης.

Ακόμα, ένα νευρωνικό δίκτυο γραφικών συνελίξεων αναπτύχθηκε (μεταξύ άλλων μοντέλων) για την πρόβλεψη της σταθεράς πρόσδεσης μεταξύ φαρμάκων και πρωτεϊνών και κατάφερε καλύτερη ακρίβεια από το στάνταρ μοντέλο της βιβλιογραφίας. Ένα σύνολο από πολλά τέτοια δίκτυα επίσης δημιουργήθηκε και αποδείχθηκε η ικανότητα γενίκευσης των προβλέψεών του ακόμα και στην περίπτωση των νέων φαρμάκων (φάρμακα που δεν έχει ξαναδεί το δίκτυο κατά τη διάρκεια της εκπαίδευσης), περίπτωση στην οποία σχέδον όλα τα μοντέλα της βιβλιογραφείας αποτυγχάνουν τελείως.

Τέλος, μελετήθηκε η δυνατότητα πρόβλεψης διαστημάτων βεβαιότητας μαζί με τις σημειακές προβλέψεις στην μηχανική μάθηση. Τρεις διαφορετικές μέθοδοι για την πρόβλεψη της αβεβαιότητας προτάθηκαν και, για πρώτη φορά, συγκρίθηκαν άμεσα, τόσο στην απλή περίπτωση όσο και στην περίπτωση των νέων φαρμάκων.

Acknowledgements

This thesis could not have been completed without the support of many amazing people.

The Mentors

Firstly, I would like to thank my advisor Leo Alexopoulos for giving me the chance to work in a very open minded and close to the real world challenges environment for the Greek standards, and for being always available for meetings and guidance.

I 'd really like to thank Christos Fotis for his constant support and excitement for the work of this thesis. Not only was he always there to help me with any technical issue or bug, but his hard work and dedication to science also motivated me to give my best every single day during my months at the lab.

I would also like to thank Thrasyvoulos Karydis for introducing me - from overseas - to some of the most challenging problems in biology, for giving me the chance to work with some of the best people in the field and for continuously inspiring me through the state of the art science he conducts and the way he manages work-life balance.

The ~~Lab-mates~~ Friends

I am grateful to Antonis, Nikos, Danae and the rest of the people in the lab for offering me a very cool environment to work and for helping the time pass easily with their jokes, support and friendly atmosphere they created.

The Family

I would like to thank Filippos and Giannis for their valuable friendship and guidance through all the hard decisions of my academic and personal life. A special thanks to Giorgos who has always been there through ups and downs and always helped me see the real purpose of my decisions through his amazing understanding and patience.

Finally, I carry the greatest gratitude and appreciation for my parents and sister, for their unconditional love and guidance.

Contents

List of Figures

List of Tables

1 Introduction

I imagine a day, some time in the not too distant future, when we will be able to create our own genetically engineered machines, cure the diseases of the old and find new ways to build and extend the capabilities of humanity, moving from programming software to programming the physical world, through biology.

-Ryan Bethencourt, 2015

Artificial intelligence is advancing various industries, including healthcare and the pharmaceutical industry. Deep neural networks started to become particularly popular around 2012, when researchers from the University of Toronto [1] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). In recent years, this brand of machine learning techniques has revolutionized several artificial fields such as computer vision, natural language processing, and game playing.

Will deep learning be able to similarly transform chemistry, biology, and medicine? Experimental sciences give rise to a wealth of unstructured, noisy, and sometimes poorly understood data. One central attraction of deep neural nets is their ability to build complex and suitable featurizations without the need to handcraft them explicitly. A method also known as feature or representational learning (see section 2.2).

1.1 Motivation - Drug Discovery

Proteins perform the vast majority of functions responsible for our health and well-being. Thus, designing substances that can regulate or alter the function of these is paramount for the implementation of safe and successful therapeutics. To conclude the introduction, we present to the reader the main motivational driver behind this work, namely drug discovery.

Drug Design: Drugs are, in majority, organic small molecules purposed to activate or inhibit a specific function of a multifunctional protein or disrupt protein-protein interactions, resulting in the treatment of a disease [2]. Despite the paramount advances in many of the scientific, technological and managerial factors that should tend to raise the efficiency of commercial drug research and development, the number of new drugs approved per billion US dollars spent on R&D is steadily falling [3]. In the heart of this declining curve, is our inability to understand the complex mechanisms involved in the process of biomolecular recognition between a protein and its epitope. Biomolecular recognition drives fundamental biological functions in any living organism, including gene regulation and sensing environmental stimuli [4]. It involves the interaction of a protein, usually a receptor or enzyme, with molecules present in living organisms, ranging from parts of macromolecules such as pieces of DNA to small molecules such as neurotransmitters and odorants. Efficient biomolecular recognition requires exquisite control of affinity and specificity [5]. In the lack of that control, conventional drug discovery is performed by screening the target proteins through large molecular assays (typically in the order of a million molecules) to identify molecules that inhibit or activate their function. These screenings are usually informed by structure based strategies [6] and in some times are preceded by computational screening [7]. Unfortunately, through the above process, while

it is possible to find molecules that work, there is no information gained about their mechanism of action. Thus, every new target requires the same amount of time and effort spent to screen through the entire molecular library, as if we were to discover a drug for the very first time. On the other hand, computational drug design when coupled with machine learning algorithms results in a net gain of information from each drug discovery, which in turn makes the discovery of the next one faster and easier.

2 Background

2.1 Deep Learning - An Overview

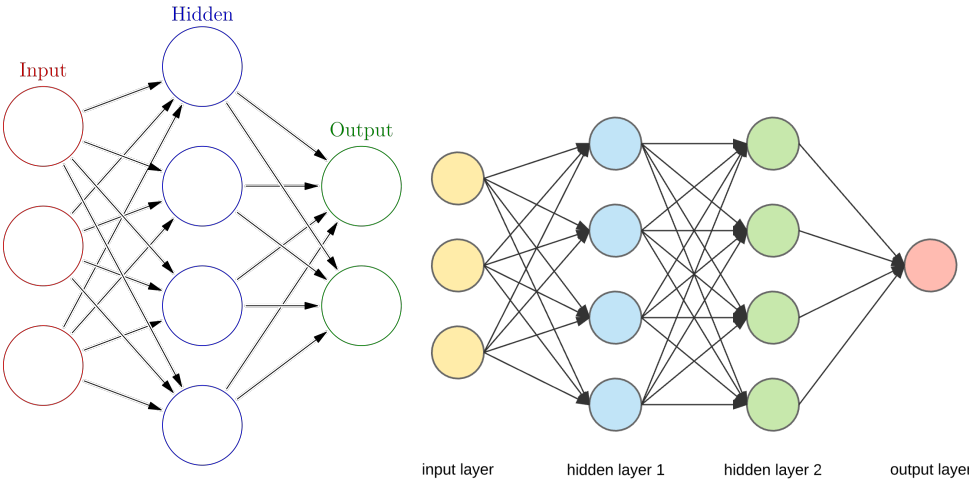
An extensive analysis of the math behind all the machine learning and deep learning techniques used for the development of this work's models is out of the scope of this thesis. However, in this section we provide an introductory overview into the theory of deep neural networks and their unique properties that distinguish them from traditional machine learning algorithms used in cheminformatics. The majority of deep learning algorithms currently developed are based on artificial neural networks, and for the purpose of this review we will focus on deep neural networks exclusively.

Artificial neural networks (ANNs), on which most deep learning algorithms are based on, are a class of machine learning algorithms inspired by biological neural networks, used to estimate or approximate functions by translating a large number of inputs into a target output (see figure 1). ANNs are constructed from a series of layers, and each layer comprises many "neurons". Each neuron accepts an input value from the previous layer, and maps it onto a non-linear function. The output of this function is used as the input for the next layer in the ANN, until it reaches the last layer, where the output corresponds to the objective that is to be predicted. In addition, a tunable parameter, the "weight" (or coefficient) of each neuron's function is adjusted in the construction of this model to minimize the error of the predicted value, a process known as "training" the neural network. Figuratively, the collection of these neurons in ANNs mimics the way neurons work in biological systems, hence its name, artificial neural network.

To train an ANN one must assign a cost or objective function between the output and the ground-truth labels. The models developed in this thesis belong to the category of maximum likelihood estimators with an i.i.d. data assumption (see section 4.1) that can be generalized to estimate a conditional probability $P(y|x;\theta)$ in order to predict y given x by optimizing the parameters θ. This is actually the most common situation because it forms the basis for most supervised learning. If X represents all our inputs and Y all our observed targets and m the number of training examples, then the conditional maximum likelihood estimator is

$$\Theta_{ML} = argmax_{(\theta)} \sum_{i=1}^{m} logP(y^{(i)}|x^{(i)};\theta)$$

In order to determine how to assign error attribution and make corrections to the weights of



(a) A traditional feedforward artificial neural network (ANN) with one hidden layer. Each neuron denoted as circle accepts a series of n input values and maps it to an output using a non-linear function, with a bias term applied to all neurons in the hidden layer.

(b) Deep neural network (DNN) differ from ANN by having multiple ($n > 3$) hidden layers as depicted in the schematic diagram, the bias term is omitted here for simplicity.

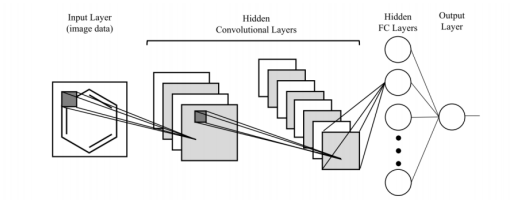
Figure 1: Artificial Neural Networks

the ANN, a common technique is by working backwards originating from the predicted output, and back through the neural network. This backwards propagation of errors is known formally as "backpropagation". During the process of backpropagation, an algorithm known as gradient descent is used to find the minimum in the error surface caused by each respective neuron when generating a corresponding output.

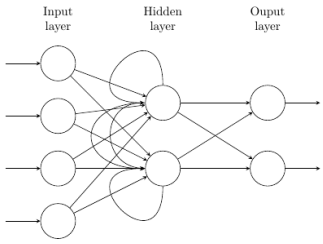
A key issue with backpropagation is that the error signals become progressively more diffused as the signal goes back through each hidden layer. In addition, the nonlinear transformation functions, such as sigmoids, have finite dynamic range, so error signals tend to decay as they pass through multiple layers, which is more commonly known as the "vanishing gradient problem" [8]. Nowadays, the most efficient way to deal with the vanishing gradient problem is to use rectified linear activation (ReLU) functions. The ReLU's first derivative is precisely unity or 0, generally ensuring that error signals can back-propagate without "vanishing" or "exploding".

As these methods enable the training of deeper and more complex neural network architectures, overfitting also becomes more of an issue, which led to the development of the dropout algorithm. In dropout, for each epoch of the training process, a fixed proportion of neurons are randomly selected to be temporarily excluded from the model. The net effect of dropout is that it simulates many different architectures during training, which prevents co-dependency among neurons and reduces overfitting [9].

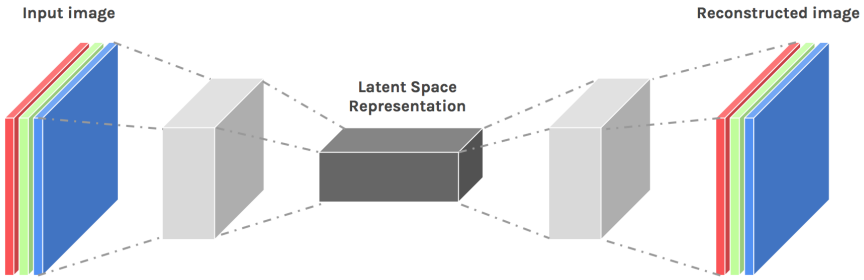
In addition to the traditional feedforward DNN (figure 1) that has been discussed so far, more sophisticated architectures include convolutional neural networks (figure 2a) [1], [10], recurrent neural networks (figure 2b) [11] and autoencoders (figure 2c) [12] that have been highly successful in computer vision and natural language processing applications. In this thesis we use all of these three architectures and compare between them as for which solves best the protein-drug binding affinity prediction problem (see section 5)



(a) Schematic diagram of a convolutional neural network (CNN). CNNs are designed with the explicit assumption that the input is in the form of image data. Each convolutional layer extracts and preserves the spatial information and learns a representation which is then typically passed onto a traditional fully-connected feedforward neural network before the output layer.



(b) Schematic diagram of a recurrent neural network (RNN). RNNs at its simplest implementation are a modification of the standard feedforward neural network where each neuron in the hidden layer receives an additional input from the output from the previous iteration of the model, denoted as circular arrows.



(c) Schematic diagram of an autoencoder, which are neural networks used in unsupervised learning. In autoencoders, the objective is to learn the identity function of the input layer, and in the process, a compressed representation of the original data in the hidden layers is learned.

Figure 2: CNNs, RNNs & Autoencoders

A technical discussion of the various DNN architectures while informative to understanding the deep learning literature, is beyond the scope of this thesis, therefore, we refer the readers to the following publications summarizing this research topic [13], [14], [15], [16]. A very concise schematic representation of all the different neural network architectures that include both traditional machine learning and deep learning techniques can be found in figure 3.

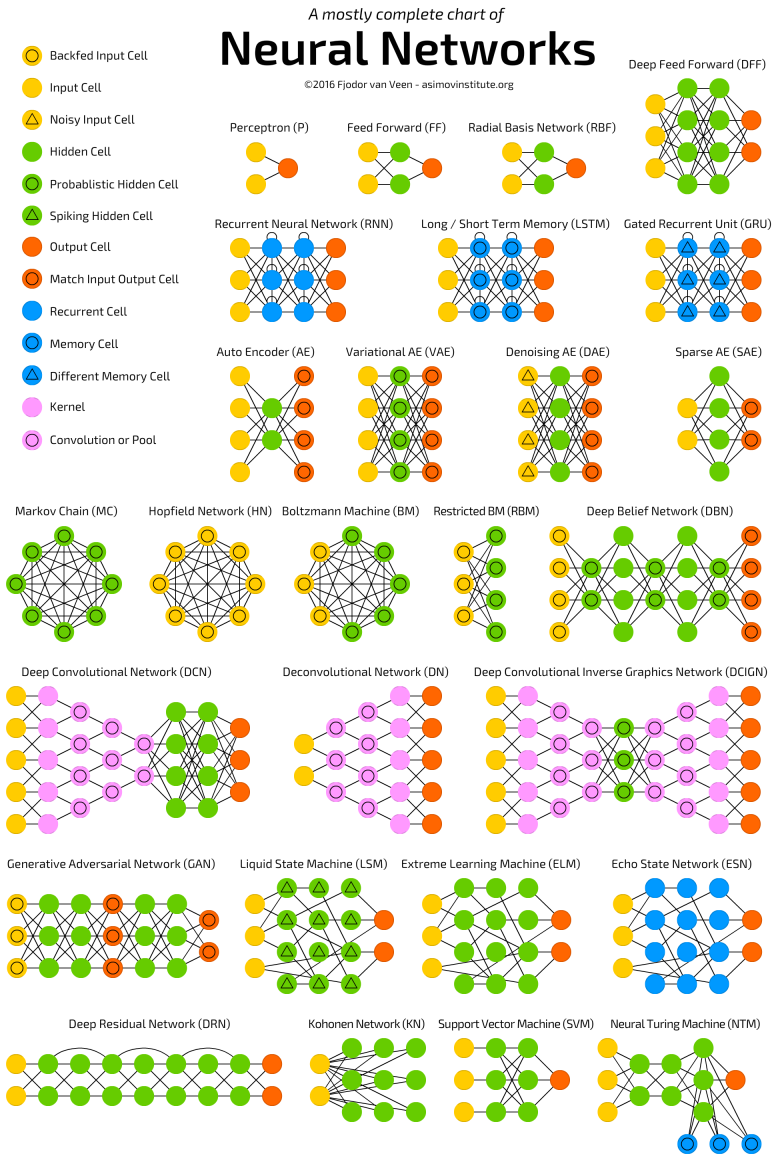


Figure 3: The most famous neural network architectures

2.2 Representation Learning for Computational Chemistry

Many decades of chemistry research has led to the development of several thousand molecular descriptors that describe a range of properties of conceivably any compound (see section 2.3.1).

Molecular descriptors thus serve as features constructed using chemical knowledge and intuition (i.e. domain expertise) that can be used in traditional machine learning models, which have achieved reasonable success in computational chemistry applications [17], [18] and [19].

In this section, however, we would like to highlight furthermore the ability of deep learning to automatically discover the representations needed for feature detection or classification from raw data. This replaces manual feature engineering and allows a machine to both learn the features and use them to perform a specific task. The so-called "feature learning" is especially useful when trying to learn representations in a space so vast as the chemical space of the 10^{60} possible molecular compounds that could work as potential drugs [20].

In traditional (shallow) machine learning the features are not extracted automatically. The features are provided by a domain expert. In a process known as feature extraction and/or engineering, various transformations and approximations are applied, which can be motivated from first principles, or may be well-known approximations, or even educated guesses. Secondly, shallow learning is template matching. It does not learn a representation of the problem, it merely learns how to precisely balance a set of input features to produce an output. Third, its expressive power grows with the number of terms (i.e. parameters to be fitted), but it may require exponentially many terms if the nonlinear transformations are chosen poorly. For example, a simple power series expansion will require an extremely large amount of terms (and parameters) to fit functions with large amounts of oscillations.

The mapping of features into an output using the function provided is the task of a processing unit, and deep learning algorithms are constructed from a collection of these processing units, arranged in a layered and hierarchical fashion. Therefore, unlike simpler machine learning algorithms, it maps features through a series of non-linear functions that are stitched together in a combinatorial fashion to optimally maximize the accuracy of the model. As a result of this complex architecture, a deep learning algorithm learns multiple levels of representations that correspond to different levels of abstraction, forming a hierarchy of concepts. By constructing these so-called "hierarchical representations," deep learning has an internal state that may be transferred to new problems, partially overcoming the template matching problem. Ultimately, these unique characteristics of deep learning enables it to utilize raw data directly as opposed to engineered features, and often the resulting models constructed produce a comparable level of predictive accuracy. Deep learning achieves this ability because within the multiple layers of the non-linear functions, the algorithm transforms the raw data and maps it to intermediate "output" that serve as input (features) for the latter layers in the algorithm, in the process gradually transforming raw data into learned features. In short, deep learning algorithms are potentially capable of automatically (i.e. without expert intervention) engineering the necessary features that are relevant to optimally predict the output of interest.

Automatic feature extraction, a process that requires no domain knowledge, is therefore one of the most significant benefits of a deep learning algorithm. This is unlike traditional machine learning algorithms, where a model must be carefully constructed with the "correct" features based off chemical knowledge and intuition for it to perform and generalize well.

2.3 The problem of molecular similarities in Machine Learning

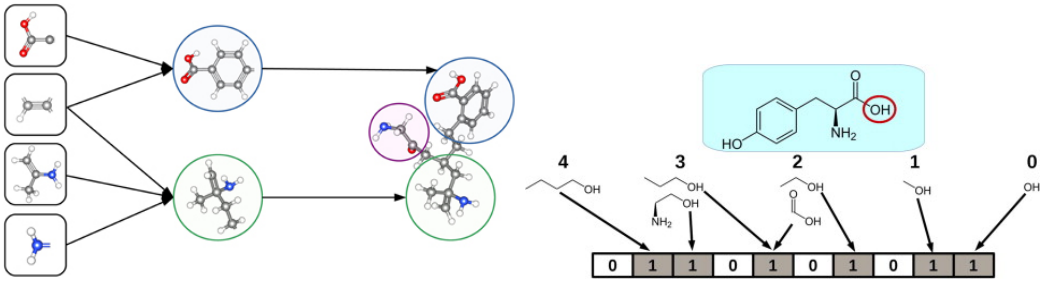
Molecular Informatics utilizes many ideas and concepts to find relationships between molecules. The concept of similarity, where molecules may be grouped according to their biological effects or physicochemical properties has found extensive use in drug discovery. Molecular similarity, as a paradigm, contains many implicit and explicit assumptions in particular with respect to the prediction of the binding and efficacy of molecules at biological receptors [21]. How molecular information is represented, called the representation problem, is also very important to the type of molecular similarity analysis (MSA) that can be carried out in any given situation [22]. Furthermore, in pharmaceutical research and drug development, machine learning methods play an important role in virtual screening and ADME/Tox prediction and for the application of such methods a formal measure of similarity between molecules is essential. Such a measure, in turn, depends on the underlying molecular representation.

The reason why molecular similarities metrics are so important is because the whole drug discovery procedure is based on the assumption that "similar" molecules will also have similar properties. For example, when medicinal chemists know a compound causes the death of cancer cells but also has toxic consequences for the rest of the human organism, they probably try to find a molecule "similar" (defined by a similarity metric) to that, which keeps the "death of cancer cells" property but the lacks the toxicity. Although that is the only way to proceed experimentally when the space to be explored is as vast as 10^{60} compounds, we cannot ignore the fact that experimental biologists explore only small neighborhoods of already know to be successful compounds while the real solution may be located somewhere "far" in the chemical space, where the "distance" is again defined by a similarity measure. That is why, in this thesis, we investigate the problem of finding a more general similarity metric as well as the ability of deep learning algorithms to generalize to compounds that are highly dissimilar as defined by various hand-engineered similarity metrics experimental biologists use today.

2.3.1 Molecular Fingerprints

Molecular fingerprints are a way of encoding the structure of a molecule. The most common type of fingerprint is a series of binary digits (usually around 1000 to 4000 bits) that represent the presence or absence of particular substructures in the molecule (see figure 4). As directly finding common substructures between a group of molecules is an NP-complete problem [23], comparing fingerprints allows you to determine the similarity between two molecules and to quickly find matches to a query substructure in a large database. To be more specific, to achieve high efficacy of similarity-based screening of databases containing millions of compounds, molecular structures are usually represented by molecular screens (structural keys) or by fixed-size or variable-size molecular fingerprints. Molecular screens and fingerprints can contain both 2D- and 3D-information. However, the 2D-fingerprints, which are a kind of binary fragment descriptors, dominate in this area.

Some of the most common fingerprints include the Morgan or **Circular Fingerprints** which take into consideration the atom types, the connectivity (Element, #heavy neighbors, #Hs, charge,



(a) Molecular substructures (b) A binary molecular 2D fingerprint

Figure 4: Molecular substructures and fingerprints

isotope, inRing) and the chemical features (Donor, Acceptor, Aromatic, Halogen, Basic, Acidic) and are calculated for the neighborhood of each atom, the **Atom-Pair and Topological-Torsion Fingerprints** which account for the atom-type (Element, #heavy neighbors, #pi electrons), the atom Pair (Atom-type – topological distance – Atom-type) and the topological Torsion (Atom-type – Atom-type – Atom-type – Atom-type) and the **2D Pharmacophore Fongerprints** which identify feature points in a molecule, calculate the inter-feature topological distances and assign bit id to feature – distance combination.

2.3.2 Similarity Measures

Attempts to quantify the similarity of compounds have a long history in chemical informatics and drug discovery. Many computational methods employ similarity measures to identify new compounds for pharmaceutical research [24]. There are many ways one might choose to measure such similarity; for example a chemical approach might rank them according to the number of physical properties and reactions they share, whereas a mathematical approach might rank them according to their similarity in three dimensions. Note that the similarity measures are independent of the molecular feature descriptors, although some measures may work best for some specific kind of fingerprints.

As far as measuring the similarity between different fingerprints is concerned, we can describe the molecules by the presence or absence of features and then the binary association coefficients or similarity measures are based on the four terms a, b, c, d shown in the table 1.

		Molecule B		
		0	1	Totals
Molecule A	0	d	b	d+b
	1	A	c	a+c=A
	Totals	a+d	b+c=B	n

Table 1: Binary association coefficients for fingerprint comparison

Where: **a** is the count of bits on in object A but not in object B. **b** is the count of bits on in object B but not in object A. **c** is the count of the bits on in both object A and object B. **d** is the count of the bits off in both object A and object B.

There are a lot of different similarity metrics defined with the help of these coefficients. The most common are shown in table 2, whereas for a full review of them the reader is referred to Holliday et al [25].

Measure	**Range**	**Formula**
Cosine	0-1	$\frac{c}{\sqrt{(a+c)(b+c)}}$
Dice	0-1	$\frac{2c}{(a+c)+(b+c)}$
Euclid	0-1	$\sqrt{\frac{c+d}{a+b+c+d}}$
Tanimoto	0-1	$\frac{c}{a+b+c}$

Table 2: Similarity metrics for fingerprints

Over the years there has been much discussion as to which type of similarity metric to use. In chemistry it has generally been thought that, as most descriptor features are absent in most molecules, i.e. the bit string descriptors in the fingerprints contains mainly zeros, coefficients such as the Tanimoto are more appropriate. Thus, the most common similarity metric used is the Tanimoto which one can see that is independent of d. It may be regarded as the proportion of the "on-bits" which are shared. The Dice similarity metric is also used in combination with specific fingerprints.

2.3.3 Our Similarity Measure

As surprising as it may seem, two molecules can be calculated to be very similar when measured by the Tanimoto, for example, distance for a specific kind of fingerprint and very dissimilar when measure encoded according to another kind of fingerprint. As a matter of fact, the choice of fingerprints will have a major impact on the Tanimoto similarity value and the correlation between the similarities of the same drugs using different fingerprints isn't always high [26], [27]. Generally there is no golden rule on which fingerprint to use but it depends of what one is trying to achieve. Usually, if we are interested in assessing molecular diversity we would use topological fingerprints like extended connectivity fingerprints (e.g. ECFP4) as they are, in the writer's opinion, well suited to capture structural differences (atom type, bond type, connectivity etc). If we are interested in identifying compounds that are similar from a "molecular recognition" point of view (capable of establishing similar interactions with an hypothetical biological target) then we would use Pharmacophore based fingerprints (e.g. FCFP4 or PHFP3) as they are (possibly) better suited to capture the molecular recognition similarity. Note for instance that the two molecules chloro-benzene and bromo-benzene (figure 5) have a Tanimoto similarity of 1 using FCFP4 (the two molecules are perceived as identical on the assumption that the two halogen atoms, Cl and Br, are equivalent from a ligand-protein recognition point of view) whereas the Tanimoto similarity is just 0.35 if ECFP4 are

used. The low value is down to the lower molecular complexity (that is indeed related to size) of the two molecules in the example. That is just because of the Tanimoto similarity metric definition.

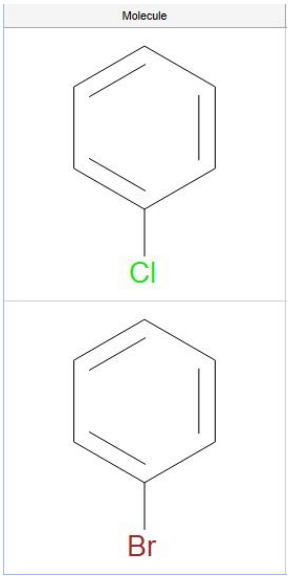


Figure 5: Are chloro-benzene and bromo-benzene similar molecules?

In the basis of this thesis work, an effort has been made to discover a new similarity metric as an aggregation of the different similarities measured using different fingerprints. Nine different fingerprints have been used including Circular, Topological and Pharmacophore fingerprints to encode the final 16506 molecules of our dataset (see section 4.2) into nine different representations, between which the similarity was measured using either the Tanimoto or the Dice distance metric, according to the fingerprint type. Next, the similarities were made binary (similar or dissimilar) according to a threshold tuned differently for every fingerprint, so that the number of similar drugs for every drug in the dataset is in the same order of magnitude for every fingerprint. Having done that, we needed not nine but only one number to show if a drug is similar to another.

In other words, we needed to aggregate the "votes" of the different fingerprints to get one similarity binary result. To do that, we had to consider that some fingerprint similarities are dependent (correlated) and others are not. Thus, a simple majority voting system was not appropriate. Our goal was to consider each fingerprint equal as for its contribution to the similarity decision between two drugs. However, if we just take a majority vote between our fingerprints, then, if some of them are dependent (e.g. ecfp4 and ecfp6), it's like we consider them more important than the others. What we did is, having the nine similarity matrices that contain the similarities of every drug with all the rest for every fingerprint, we compared them to extract Spearman correlations between the fingerprints. According to the spearman correlations we decided which sets of fingerprints (referred to as criteria from now on) can be considered independent between them and which dependent. Finally, to aggregate them we follow a weighted majority rule that is:

If S a set of independent criteria then we assign weight $W = 1/|S| + \epsilon$ to every

criterion in the set, where ϵ a small number (it does not matter how small). If a criterion is not dependent to any else then its weight is assigned $W = 1$.

That way if all the criteria accord with each other in S then their " “" has in total a weight of $1 + |S|\epsilon$ (which is just something above 1). The idea is that they will all count like one “vote” but in case two sets S, S' disagree, the most weighted is the “vote” of the one that includes the more criteria. This whole procedure to find a generalized way to count how many molecules are similar to a specific one in a dataset was carried out in order to make validation sets for our model that are equally hard to the test-set of a competition we participated in. For more details see section 4.3 and section 9.

3 Related Work

In computer-aided drug design, traditional machine learning algorithms have a long history in the field of cheminformatics, notably in their contribution to quantitative structure activity relationship (QSAR) applications. In QSAR, the output to be predicted is usually the biological activity of a compound. Usually regression models are used, and the input data are molecular descriptors, which are precomputed physicochemical properties of the molecule, designed from chemistry domain knowledge. Early work in QSAR applications used linear regression models, but these were quickly supplanted by Bayesian neural networks [28] ,[29], followed by RFs [17] and SVMs [30]. Practitioners in the field have historically favored models that allow for variable selection so that an informed chemist can determine if selected features made sense. In addition, models that allowed assessment of uncertainty of output predictions were also preferred. The field of QSAR is vast, and we refer readers to the following reviews for key historical technical developments [31], [32]. For the purpose of this review, we will limit the scope of discussion to the performance of DNN-based QSAR models and appropriate comparisons to traditional machine learning models.

The first foray of deep learning into QSAR was the Merck challenge in 2012. The winning group used DNN models, led by Dahl who was part of Hinton's research team. Notably, the team had no formally trained computational chemist in the group; they were from the computer science department. Later, in 2014, Dahl et, al. submitted an arxiv paper exploring the effectiveness of multi-task neural networks for QSAR applications, based on the algorithms used in the Merck challenge. In this work, the authors used a multi-task DNN model. In an accuracy performance benchmark against other traditional machine learning algorithms, such as gradient-boosted decision trees and logistic regression, the DNN-based model outperformed all others in 14 of 19 assay predictions by a statistically significant margin and was comparable in terms of performance in the remaining 5 assay prediction.

A subsequent study in 2015 published by Merck, comprehensively analyzed the training of DNNs and compared their performance to the current state of the art used in the field, RF-based models, on an expanded Merck challenge dataset [33]. The key issue associated with training deep neural networks, particularly in the number of tunable parameters was also investigated. The authors discovered that most single task problems could be run on architectures with two hidden layers, using only 500-1000 neurons per layer and 75 training epochs.

In 2014, Hochreiter and co-workers published a peer-reviewed paper at the Neural Information Processing Systems (NIPS) conference on the application of multi-task DNNs for QSAR application on a significantly larger dataset. The authors benchmarked the accuracy performance of the DNN model across 1230 targets, and compared them against traditional machine learning models, including SVMs, logistic regression and others. it was demonstrated that DNNs outperformed all models they tested on, which also included 2 commercial solutions and 3 currently implemented solutions by pharmaceutical companies.

Similar studies were conducted by Google and Pande group in 2015 [34] that focused on the mulit-task learning and evaluated the transferability to tasks not contained in the training set, while others like Swamidass and coworkers in 2015 used DNNs to predict toxicity and other properties [35]. Swamidass and co-workers results were particularly distinctive, because they developed a DNN model to predict the specific location on a molecule that undergoes epoxidation, i.e. its site of epoxidation (SOE). A larger scale study on chemical toxicity was also published by the Hochreiter group in 2016 [36]. DeepTox model developed by Hochreiter and co-workers had the highest performance of all methods submitted to the Tox21 challenge [37]. Further analysis of their model indicated that using a multi-task DNN model led to consistent outperformance against single-task models in 10 out of 12 assay predictions.

In line with the progress in QSAR and toxicity prediction, deep learning algorithms have also started to make an impact in other aspects of computer-aided drug design. In 2013, Baldi and co-workers reported using a DNN model to predict molecule solubility [38]. Other research developments in this direction was also submitted to arxiv by Pande and co-workers, where they developed a multi-task DNN model for predicting not just solubility, but the entire spectrum of ADMET properties. Deep learning may also have a future in virtual screening as a viable alternative or complement to existing docking methods. In 2016, an arxiv paper was submitted by AtomNet, a startup that developed a DNN model to classify the activity of small molecules docked in protein binding pockets [39]. For additional recent developments of deep learning in applications that are more closely aligned to computational biology, we refer our readers to the following review that focuses on that research topic [40].

4 Data

4.1 Data über alles

The world of data is expanding at an unimaginable pace. It's estimated that by 2020, every human would create 1.7MB of information each second. The true power of data can be unlocked when it is refined and transformed into a high quality state where we can realize its true potential. Bad data is sighted as one of the root causes for failure of data-dependent initiatives as machine learning is.

An ML algorithm is a self-teaching entity that usually learns from available data about which we typically make a set of assumptions known collectively as **the i.i.d. assumptions**. These

assumptions are that the examples in each dataset are independent from each other, and that the training set and test set are identically distributed, drawn from the same probability distribution as each other.

Most of the popular deep learning models nowadays (including the models of this thesis) are estimators based on the maximum likelihood principle. One way to interpret maximum likelihood estimation is to view it as minimizing the dissimilarity between the empirical distribution $\hat{p}_{data}$, defined by the training set and the model distribution, with the degree of dissimilarity between the two measured by the KL divergence. Minimizing this KL divergence corresponds exactly to minimizing the cross-entropy between the distributions. We can thus see maximum likelihood as an attempt to make the model distribution match the empirical distribution $\hat{p}_{data}$. Ideally, we would like to match the true data-generating distribution p_{data}, but we have no direct access to this distribution.

It is, thus, obvious the great importance of the unbiased data, their diversity and reliability. The more diverse the data is, the more our model can generalize to new input data and learn how to represent a larger part of the true data-generating distribution, which, in our case, is the chemical space (10^{60} different molecules). As a matter of fact, it is a property of Maximum Likelihood that the true distribution p_{data} must lie within the model family $p_{model}(x;\theta)$. Otherwise, no estimator can recover p_{data} no matter how big the number of training examples is.

4.2 Data Augmentation

A large part of this thesis concerns an effort that has been made to create a large dataset out of literature sources that will contain binding affinity data, curated as for its quality and integrity, between compounds and protein kinases measured by the binding affinity and equilibrium dissociation constant Kd. Generally speaking, the curation process involves several manual and automated steps and aims to: (1) maximize data accessibility and comparability; (2) improve data integrity and flag outliers, ambiguities and potential errors; and (3) add further curated annotations and mappings thus increasing the usefulness and accuracy of the data for all users and modellers in particular.

This section concerns the gathering and organizing of the various protein kinases-micromolecules binding datasets that exist for free in the web and in the literature as for the date of this thesis' writing. The various data of experimental biological labs around the world that comes out publicly available is unfortunately not yet formatted universally. The differences are usually spotted either on the compound or protein representations (i.e. different type of SMILES formatting, custom IDs and numbering, lack of unique UniprotKB IDs etc) or on the way each lab measures the affinity metric. In this work, an effort has been made to clean and format universally the binding affinity data between protein kinases and ligands, that comes from different sources, experiments, labs and is expressed in various metrics such as Kd, Ki, IC50 etc.

The goal of this work is to create a unique overall dataset that will contain the most probable Kd values of all the interactions between kinases and drugs found in the literature, specifically

formatted to be easy to use for machine learning applications and competitions such as the IDG-DREAM challenge [41] without any further work.

4.2.1 Useful data in the literature

DTC dataset

Drug Target Commons (DTC) is a publicly available web platform (database with user interface) for community-driven bioactivity data integration and standardization for comprehensive mapping, reuse and analysis of compound–target interaction profiles.

DTC contains 1509 different types of bioactivity data between drugs and kinases gathered from the literature and other open access databases. Out of those types 4 are considered to be dissociation constants (Kd, KD, KDAPP, PKD). The drug-kinase pairs in DTC that have at least one of those four types are 85475 (not unique). After filtering for NaN values as well as for Kds and units that were not concentrations (not in PKD) 2895 pairs were removed. Out of the 82580 pairs remaining, 2308 were removed due to having no protein ID.These 2308 pairs consist of 617 unique proteins that we have to fix. In addition 2821 pairs were missing the inchikey identifier making the process of acquiring their smiles a bit harder. 1247 pairs were mapped to smiles based on their ChemblID and chemical name while the rest 1574 were mapped to their smiles using the UNC dataset (see below) that we extracted from a pdf. From this approach all but 2 drugs (13 pairs) (were restricted drugs) out of the 550 that had no inchi keys were mapped to their smiles.

Next, the pairs that contained inchikeys were mapped to their smiles using Chembl database. On this front, 186 drugs involved in 767 interactions were not found in Chembl. 128/186 were mapped to their smiles using BindingDB database. The remaining 58 drugs involved in 586 interactions were removed.

In DTC there exist 3144 pairs of data points (compounds) with multiple target proteins separated by comma. Those data points were kept by splitting each data point into multiple interactions creating 11529 interactions. Although this approach might not be the cleanest we don't think it will cause any issues, because in the final step we filter to keep only the unique pairs based on the standard deviation of the Kd values for multiple pairs.

The aminoacid sequences for the proteins in the pairs were all collected from UniprotKB using the uniprot IDs of the pairs as identifiers.

Next, after observing the range of Kd values, the max value for Kd was set to 10000 NM and the min to 0.001 NM resulting in a Pkd metric in the range of 5 to 12.

Finally, because DTC reports the sign for each interaction as well ($<, >, =$) some inconsistencies were removed. That is if the sign was $<$ and the value between 100 (pkd=7) and 10000 (pkd=5) NM the pair was removed due to uncertainty. Pairs with $K_d < 100$ and sign $<$ were kept being considered as "bound". In addition, if the sign was $>$ and the value was $< 1000 (pkd > 6)$ the pair

was removed. Pairs with sign $>$ and $value > 1000 (pkd < 6)$ were left in as they were considered inactive.

Note: These "sign" filters were also applied to every dataset from the ones below where is needed.

PKIS dataset

The Published Kinase Inhibitor Set (PKIS) is a collection of 376 compounds that have been made available by GSK for screening by external groups; all compounds have been published in the scientific literature. The PKIS contains 3719 interactions, out of which only 1084 concern Kd values. We recover the drug SMILES by mapping the PKIS InChI-keys on the CHEMBL database. The sequences are taken from UniprotKB and the Kd values are converted to pKd. The max pKd value is also set to pKd = 12 and larger values are removed. Note: These filters and mappings are applied to every dataset.

KKB database

The Kinase Knowledgebase (KKB) is Eidogen-Sertanty's database of kinase structure-activity and chemical synthesis data. The KKB contains over 1.8M kinase structure-activity data points reported in peer-reviewed journals and patents but one has to pay for it to be accessed. Rajan Sharma et al. [42] have extracted and made available publicly 258K structure activity data points and 76K associated unique chemical structures across eight kinase targets. Out of these 258K datapoints, one can extract 6227 Kd values associated with 233 compounds.

DAVIS dataset [43]

This dataset contains the interactions (Kd) of 72 kinase inhibitors with 442 kinases covering $>$ 80% of the human catalytic protein kinome. Small note: the drug names included in DAVIS dataset must be matched to InChI keys by mapping the CTS (Chemical Translation Service).

BindingDB database

BindingDB is a public, web-accessible database of measured binding affinities, focusing chiefly on the interactions of protein considered to be drug-targets with small, drug-like molecules. BindingDB contains 1,587,753 binding data, for 7,235 protein targets and 710,301 small molecules. Out of this data the interactions containing only kd metric have been kept (70300 interactions) out of which 69429 remained after mapping the InChI keys and applying the "sign" and other filters.

HMS LINCS database

The Library of Integrated Network-based Cellular Signatures (LINCS) program is developing a library of molecular signatures that describes how different types of cells respond to a variety of agents that disrupt normal cellular functions. Under this database the Harvard medical school in collaboration with the NIH have made publicly available multiple kinases-compounds binding experiments using the KINOMEscan technology. These datasets may contain directly Kd values or

concentration of compound values that need to be multiplied by the control percentage to become Kd (units also differ between the different datasets in the website so that is also to be kept in mind). The datasets also identify the proteins and the compounds by custom LINCS IDs that need to be converted to UniprotKB IDs and canonical SMILES using other datasets provided in the website. Out of the 182 different csv files that can be downloaded from the LINCS database website and contain experiments with kinases using KINOMEscan, one can extract 401 proteins, 162 drugs and 42610 unique interactions between the protein-drug pairs out of the 63700 total values, throwing away NaN values, restricted SMILES and after applying all the filters.

UNC dataset

Another useful kinases-ligands dataset can be found in [44] from the university of North Carolina. Extracting information out of this dataset is of particular difficulty due to the .pdf format of the data. However, interactions of great quality (relatively rare and structurally different molecules) can be found in here. After a lot of custom export functions 361 drugs, 339 proteins and 1793 unique interactions were added.

4.2.2 "Useless" (non Kd) data in the literature

GOSTAR database

Huge database which belongs to Excelra Knowledge Solutions (formerly GVK Informatics) company (part of the 1 billion dollars GVK group). This database is probably not freely available, although was not able to request a licence through the webpage, since it seems there is a bug when the website is trying to read the user's IP address.

EMD Millipore database

Pairwise assays between purified recombinant human kinases and a collection of small molecule inhibitors were carried out by the EMD Millipore Corporation (now known as Millipore Sigma) using a filter binding radioactive ATP transferase assay. However, the metrics used for this database are the Km and S score, which unfortunately cannot be directly related to Kd. More information about this work can be found in [45].

Kinase Inhibitor Resource (KIR) database

In [46] Anastasiadis et al. used a custom affinity metric that was cheaper for them to evaluate. The metric is defined as the average of two replicates which is shown as percent remaining kinase activity in the presence of the compound relative to solvent control. These values, unfortunately, cannot be directly interpreted as Kd values and thus they are "useless" for this work.

4.2.3 Merging the datasets

After combining the data from all the different datasets all the interactions that exist several times should be treated so that they are included just once. When merged initially the datasets contained 260571 pairs and 16871 unique drugs in total. From the duplicates we kept the ones that their Kd value differ less than 10%. As a Kd value we kept their mean. From the ones that exist more than twice, we kept the ones that their std/mean is less than 10%. As a Kd value we kept their mean. Finally, 103741 unique interactions are included in our dataset.

In order to have a universal SMILES format (see section 4.4) (SMILES canonicalization) all the SMILES were converted to molecular graphs and back to SMILES using the RDkit anaconda tool.

Regarding the tools used in this data curation work, it is obvious that the size of the datasets is big enough to prevent any manual editing. All of the work was carried out using scripts. Most of the datasets manipulation was done using R in RStudio platform. Python was also used supportively to extract some information in specific cases as is the UNC, the LINCS and the PKIS datasets.

4.3 Creation of difficult validation sets

As already mentioned, the generalization in previously unseen and structurally different molecular scaffolds is the great challenge of deep learning and it is truly difficult. As a matter of fact, a great study by Wallach et. al. [47] reveals that most deep learning classification benchmarks actually reward memorization rather than generalization. In order for one to be able to judge objectively if his/her deep learning model can generalize well, a difficult validation set is needed. By the term difficult we mean a set of molecules that are structurally different from the ones that the model has used to train on, where the "difference" or "similarity" is measured by a specific similarity metric.

In this work, we used a generalized custom similarity metric (see section 2.3.3) to create validation sets that contain molecules that are very dissimilar to the ones used in the training sets of our models. We validated both our models and the literature baselines to datasets of different difficulty or similarity levels and showed how the validation mean square error increases rapidly when validating on dissimilar molecules. In order to create the validation sets, other factors had also to be taken into account. These include the labels (Kd) distribution of the validation sets to be close to the training set, the molecules included in the validation sets not to have too many protein-drug pairs in the initial dataset so that the training set does not become too small and the distribution of labels (Kd values) in both the training and validation sets to stay, as much as possible, non-biased, which means close to uniform.

For more details on how the experiments of the increasing validations sets difficulty were conducted and the comparison between them see section 7.

4.4 Input Data

In machine learning input samples have traditionally been modeled as vectors. Consequently, molecules are represented to machine learning algorithms in a vectorized form using molecular descriptors. While this approach is straightforward, it has its shortcomings. Amongst others, the interpretation of the learned model can be difficult, e.g. when using fingerprints or hashing. Structured representations of the input constitute an alternative to vector based representations, a trend in machine learning over the last years. For molecules, there is a rich choice of such representations. Popular examples include the molecular graph, molecular shape and the electrostatic field [48].

In deep learning we prefer, as analyzed in section 2.2, to provide the algorithm the data in a raw form and let it learn the most appropriate representation alone. On this front, we use either simple one-hot encoded SMILES to represent the molecules and one-hot encoded amino-acid sequences to represent the proteins or one-hot encoded graph representation of the molecules when training graph convolutional neural networks (see section 5.2).

The **S**implified **M**olecular-**I**nput **L**ine-**E**ntry **S**ystem (**SMILES**) is a specification in the form of a line notation for describing the structure of chemical species using short ASCII strings. SMILES strings can be imported by most molecule editors for conversion back into two-dimensional drawings or three-dimensional models of the molecules (figure 6). SMILES are an one-to-one correspondence of molecules to strings and include information about the molecule atoms, bonds, rings, aromaticity, branching and stereochemistry.

As for the one-hot encoding, in machine learning one-hot is a group of bits among which the legal combinations of values are only those with a single high (1) bit and all the others low (0). In other words, when encoding with one-hot a SMILES string the result is an array of ones and zeros of shape=(length of the SMILES dictionary, length of the SMILES string) where all the values in each column are zeros except for the one which corresponds to the character of the SMILE string in that position (see figure 7). The same way, we encode the amino-acid sequences to represent the proteins, where the dictionary is the 20 different amino-acids.

5 Deep Learning for Protein-Drug Binding Affinity Prediction

Our efforts focused on building a deep end-to-end model that takes as input either a SMILES representation or a graph representation of a compound and the amino-acid sequence of a protein and outputs the Kd value of the compound-protein pair. On this front, we investigated several deep architectures in order to represent the SMILES-sequence pairs in a latent space that best captures the nature of the binding affinity prediction.

At the same time we implemented a baseline model, namely the DeepDTA model [49], trained on the same augmented dataset that we made (see section 4.2) in order to have a fair comparison

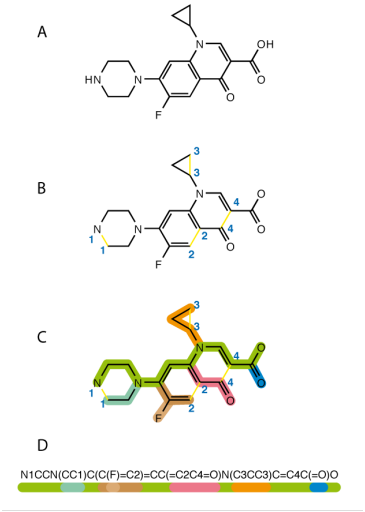


Figure 6: Generation of SMILES: Break cycles, then write as branches off a main backbone. (Ciprofloxacin)

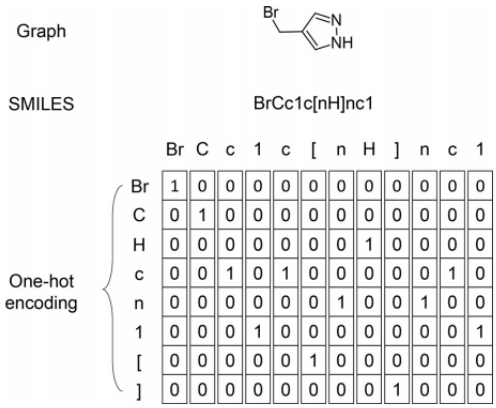


Figure 7: Exemplary generation of the one-hot representation derived from the SMILES of 4-(bromomethyl)-1H-pyrazole.

between our models and the literature.

5.1 DeepDTA Model

The work of the DeepDTA model concerns the modeling of protein sequences and compound 1D representations with convolutional neural networks (CNNs) [49]. The paper that was published in

June 2018 managed to overpass the previous baselines that were the KronRLS regression algorithm [50] and the SimBoost method [51] and thus became a new baseline that we used for our models comparison.

The architecture of the model is rather simple, as they use 1D convolutions to encode separately the SMILES and the protein sequences and then concatenate the encodings to further produce a binding affinity output through some fully connected layers (see figure 8. More detailed description of the architecture is out of the scope of this thesis, as it is also described in their paper [49].

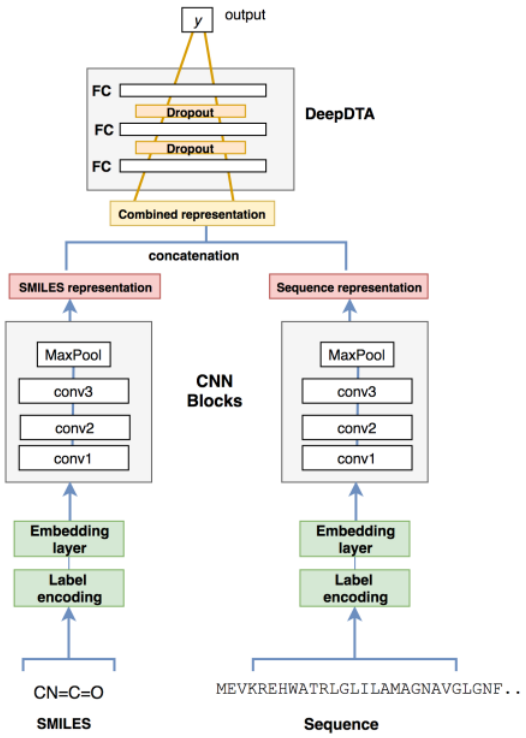


Figure 8: DeepDTA model with two CNN blocks to learn from compound SMILES and protein sequences.

5.2 Graph Convolutions

Low-dimensional vector embeddings of nodes in large graphs have proved extremely useful as feature inputs for a wide variety of prediction and graph analysis tasks [52], [53], [54]. In computational chemistry graphs are also very useful to representing molecular compounds. In these graphs, vertices usually represent individual atoms and edges represent bonds.

On the front of trying to learn the function that computes molecular fingerprint vectors (see

representational learning in section 2.2) instead of learning directly on handcrafted fingerprints as the ones described in section 2.3.1, we developed a deep graph convolutional neural network (GCNN) similar to the one developed by D.Duvenaud et al. [55]. The GCNN takes as input, instead of just the one-hot encoded SMILES string (see section 4.4), a matrix containing one-hot encoded information of other atom features as well (apart from the atom type) such as the solubility and hydrophobicity of the atoms. More formally, for these models the goal is to learn a function of signals/features on a graph $G = (V, E)$ which takes as input:

- A feature description x_i for every node i; summarized in a $N \times D$ feature matrix X (N: number of nodes, D: number of input features)
- A representative description of the graph structure in matrix form; typically in the form of an adjacency matrix A (or some function thereof)

and produces a node-level output Z (an $N \times F$ feature matrix, where F is the number of output features per node).

More details into how one can generalize convolutional neural networks to work on arbitrarily structured graphs is out of the scope of this thesis but the interested reader is referred to the work of Defferrard et al. [56] and Kipf & Welling [57]. A GCNN is depicted schematically in figure 9 (the figure was borrowed from Expero, Inc.:`https://www.experoinc.com/post/node-classification-by-graph-convolutional-network`).

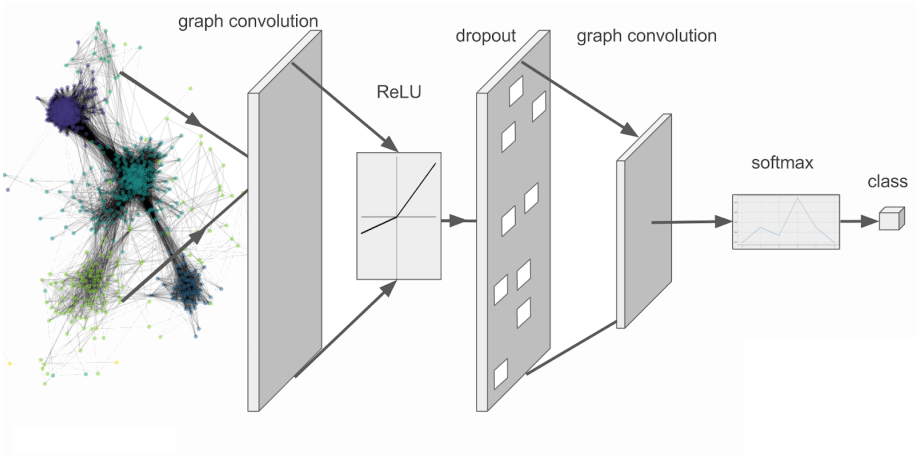


Figure 9: A two layer graph convolutional network using one intermediate dropout layer.

Our architecture consists of a 3 layer deep graph convolutional network to extract application specific neural fingerprints from the compound structures. These fingerprints are then concatenated with the output of a 3 layer deep CNN that encodes the amino-acid sequences of the proteins (see figure 10. The combined feature vector is fed through 2 fully connected layers for the final Kd prediction. Batch normalization layers and ReLU activations are used throughout the network

except for the final prediction layer. In order to reduce overfitting, dropout and L2 regularization is also used between the fully connected layers.

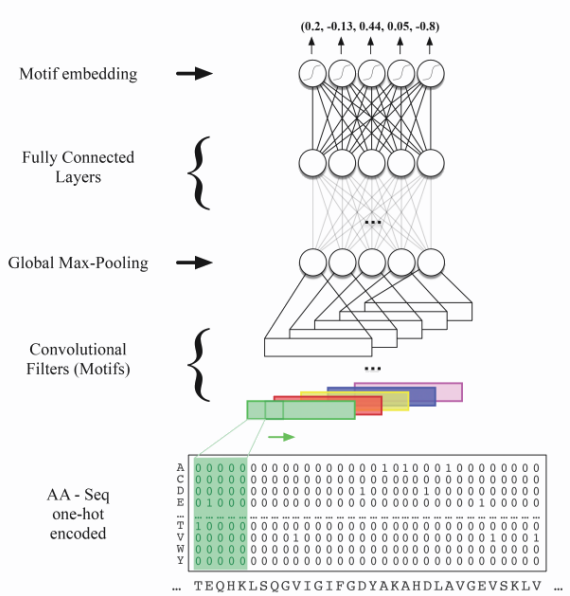


Figure 10: Encoding 1hot amino-acid sequences through a CNN

5.3 LSTM Autoencoder

An **autoencoder** is a type of artificial neural network used to learn efficient data codings in an unsupervised manner [58], [59]. It learns to compress data from the input layer into a short code, and then uncompress that code into something that closely matches the original data (see figure 2c). This forces the autoencoder to engage in dimensionality reduction, for example by learning how to ignore noise.

Long short-term memory (**LSTM**) is an artificial recurrent neural network (RNN) architecture. Unlike standard feedforward neural networks, LSTM has feedback connections that make it a "general purpose computer" (that is, it can compute anything that a Turing machine can) [60]. LSTMs were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs.

The exact details of the autoencoder and especially the LSTM architecture are definitely out of the scope of this thesis, as they are rather complicated. In our work however we created an LSTM autoencoder to learn a (as general as we could) compound representation and combined it with a CNN for the proteins representation, in order to predict the binding affinity between the compounds and the proteins.

To be more specific, we first trained a deep LSTM autoencoder on the SMILES sequences of all the compounds in the training, validation and test sets. The autoencoder used as input the one-hot representation of the SMILES and was tasked to predict the next letter in the sequence. The output of the trained encoder can serve as a compressed latent representation of the SMILES space. The idea behind training an autoencoder first, is that the encoder learns to represent all the available SMILES (training and test) and the final model should perform better than a fully end-to-end architecture that has never seen the structures of the test set. Thus, the final model used as input the output of the encoder along with the one-hot encoded amino-acid sequences. The sequences were encoded again using a 3 layer deep CNN and concatenated with the output of the encoder to build the final feature vector. This feature vector was then passed through 2 fully connected layers for the final Kd prediction. Batch normalization layers and ReLU activations were used throughout the network except for the final prediction layer. In order to reduce overfitting, dropout and L2 regularization was used between the fully connected layers.

6 Predicting with confidence

6.1 Background

The problem of uncertainty prediction in machine learning is very important although not very well-known. One of the disadvantages of machine learning as a discipline is the lack of reasonable confidence intervals on a given prediction. Neural networks specifically suffer from two basic limitations despite their popularity. The first problem is the unsatisfactorily low prediction performance when there exists uncertainty in the data. The reliability of point forecasts significantly drops as a result of the prevalence of uncertainty in operation of the system. The second problem of NNs is that they only provide point predictions without any indication of their accuracy. Point predictions are less reliable and accurate if the training data is sparse, if targets are multivalued, or if targets are affected by probabilistic events. To improve the decision making and operational planning, the modeler should be aware of uncertainties associated with the point forecasts. It is important to know how well the predictions generated by NN models match the real targets and how large the risk of un-matching is.

Test-time dropout is one of the most convenient and used ways to get an estimation of the uncertainty of deep learning predictions. Test-time dropout which is also known as **Bayesian deep learning** has been proved to be a variational approximation to a Gaussian process and therefore uncertainty estimates using the variation of the different predictions can be extracted. For more details see section 6.2.

Another way one could estimate uncertainty is to check how close an example is to the training data points in the input space or in the latent (learned) space before the final prediction layer. This method works under the assumption that the more similar a query is to an example that the algorithm has previously seen (training example) the more confident should the model be for its prediction.

Furthermore, **ensemble learning** can be also used to estimate the confidence of the prediction. Ensemble learning is about using many different models and getting a consensus of their predictions and assumes that an ensemble of NN models will produce a less biased estimate of the true regression of the targets [61]. It can refer to bootstrap aggregating (also called bagging) which involves the generation of many different training sets by sampling the main one uniformly and with replacement, to training the "same" model but with different architecture configurations on the same data or to training many different models again on the same data. To predict confidence intervals out of ensemble learning, one can take the standard deviation of the different models predictions as a metric of uncertainty.

Finally, another important method proposed in the literature for trying to estimate the prediction uncertainties in models that contain non-linearities is inductive **conformal prediction**, which uses past experience to determine precise levels of confidence in new predictions. Given an error probability ϵ, together with a method that makes a prediction $\hat{y}$ of a label y, it produces a set of labels, typically containing $\hat{y}$, that also contains y with probability $1-\epsilon$. Conformal prediction can be applied to any method for producing $\hat{y}$: a nearest-neighbor method, a support-vector machine, ridge regression etc. [62].

Apart from the ones mentioned above, other methodologies have been proposed in the literature as well. However most of them fail in quality (quantification of the performance of each method for producing high quality prediction intervals), repeatability (information about how each method will do in worst cases, which methods are prone to fail, and which methods do well even in their bad runs), computational load or variability (the response of the predictions intervals to the level of uncertainty associated with data). Briefly we mention:

- Generating a distribution of predictions and calculating the variation between them using simple cross-validation, producing that way an average confidence interval. This method, however, shows more about the robustness of the model and reflects less the ground truth correctness likelihood of the prediction labels.
- Delta method: the delta method has its roots in theories of nonlinear regression and interprets an NN as a nonlinear regression model that allows us to apply asymptotic theories for prediction intervals construction.
- Bayesian method: Bayesian NNs, which learn a distribution over weights, are currently the state-of-the-art for estimating predictive uncertainty; however these require significant modifications to the training procedure and are computationally expensive compared to standard (non-Bayesian) NNs.
- Mean-variance estimation (MVE) method: works like the Bayesian method but also estimates the target variance using a dedicated NN.
- PAC approaches: probably approximately correct (PAC) learning. Although well established theoretically, not enough applied work exist in the literature for one to implement this easily in deep frameworks.

Naturally, when trying to develop a model that apart from the point predictions also outputs

uncertainty estimations, one must have a validation metric to judge these uncertainty predictions after the training. This is not such a trivial problem to think about but for the purpose of this thesis it has been decided that the validation of the uncertainty estimates can be calculated using the absolute error of the point predictions (i.e. the mean of all the predictions for an input), expecting the data points with the larger error to also have the maximum uncertainty. That way, the validation accuracy is defined as the correlation between the predicted uncertainties and the error of the correspondent inputs.

6.2 Estimating uncertainties using Dropout

The Gaussian process is a powerful tool in statistics that allows us to model distributions over functions. It has been applied in both the supervised and unsupervised domains, for both regression and classification tasks [63], [64]. The Gaussian process offers nice properties such as uncertainty estimates over the function values, robustness to over-fitting, and principled ways for hyper-parameter tuning.

It's long been known that deep learning models can be related to Gaussian processes. Take a neural network (a recursive application of weighted linear functions followed by non-linear functions), put a probability distribution over each weight (a normal distribution for example), and with infinitely many weights you recover a Gaussian process [65], [66]. For a finite number of weights you can still get uncertainty by placing distributions over the weights. This was originally proposed by MacKay in 1992 [67], and extended by Neal [65] in 1995.

It can be proven that dropout neural networks are identical to variational inference in Gaussian processes (variational inference is an approach of approximating model posterior distribution which would otherwise be difficult to work with directly). Actually, averaging forward passes through the network (which is another way to say test-time dropout), is equivalent to Monte Carlo integration over a Gaussian process posterior approximation [68].

As a part of this thesis, we incorporate test-time dropout in our deep graph convolutional neural network and validate our uncertainty estimates (see section 7.3).

6.3 Estimating uncertainties using latent space norm

In this section we are trying to answer the question of whether the position of a data point (in our case protein-drug pair) in the latent space learned by the algorithm is a good measure for the uncertainty of the correspondent prediction. On this front, in the 128-dimensional latent space of our deep graph convolutional neural network we calculated all the euclidean distances between every validation and training point for a random-split validation set.

For estimating the uncertainty of a validation point prediction we followed three approaches. First, for a given validation point, we counted its "neighbors" in the 128-dimensional sphere of

a finite radius and considered the inverse of this number as a metric of uncertainty. Second, we calculated uncertainty as the normalized median of all the distances between this point and the training set. Third, we did not use any distances but calculated the uncertainty as the level of the point prediction error of the neighboring training points in the latent space. We assumed that validation points that are included in areas of high error in the latent space should also be accompanied with a high uncertainty prediction.

For the diagrams and other visualizations of the uncertainty experiments using the latent space of our GCNN model see section 7.3.

6.4 Estimating uncertainties using a deep ensemble

There are quite a few sophisticated ways to combine models in an ensemble, as analyzed in [61]. In this section we implemented the uncertainty estimation using an ensemble of deep models trained on the same data set (108 slightly different graph convolutional neural networks as for their architecture configuration) that predict two values instead of one. To be more specific, we tried to recreate a state of the art model that was published by Google's DeepMind in 2017 [69] and constitutes the best approach for estimating uncertainties in deep learning according to the author's opinion.

The goal is to have a neural network whose predictions are also calibrated in the sense of a frequentist notion of uncertainty which measures the discrepancy between subjective forecasts and (empirical) long-run frequencies. In the implemented model the traditional cost function (e.g. Mean Square Error) is altered so that it also captures predictive uncertainty. Like in [70] we use a network that outputs two values in the final layer, corresponding to the predicted mean $\mu(x)$ and variance $\sigma^2(x) > 0$. By treating the observed value as a sample from a (heteroscedastic) Gaussian distribution with the predicted mean and variance, we minimize the negative log-likelihood criterion:

$$-log(p_\theta(y_n|x_n)) = \frac{log(\sigma_\theta^2(x))}{2} + \frac{(y - \mu_\theta(x))^2}{2\sigma_\theta^2(x)} + constant$$

Thus, each of our 108 models learns to represent the output as a Gaussian distribution by learning both the mean and variance. The combination of these models output distributions provides the robustness advantage of the traditional ensembling methods as well. Ensemble results are treated as a uniformly-weighted mixture model, and, as mentioned before, the ensemble prediction is approximated for simplicity to be a Gaussian whose mean and variance are respectively the mean and variance of the mixture. Thus the mixture parameters are given by:

$$\mu_*(x) = \frac{1}{M}\sum_m \mu_{\theta m}(x)$$

$$\sigma_*^2(x) = \frac{1}{M}\sum(\sigma_{\theta m}^2(x) + \mu_{\theta m}^2(x)) - \mu_*^2(x)$$

where M is the number of the models, θ the parameters and m each model in the sum.

We note that the parameters that we altered between the 108 models are the learning rate (values: [0.005, 0.001, 0.0005]), the number of convolutional filters (values: [16, 32, 64]), the size of the convolutional filters (values: [16, 32, 64]), the size of protein encoding (values: [8, 16]) and the size of the latent feature vector (values: [96, 128]). The results of this approach are also included in section 7.3.

7 Experiments

In this section we describe all the experiments that we did, or, in other words, the training of the three models (deepDTA, Graph Convolutional Neural Network & LSTM Autoencoder), their validation and the machines that we used for it. Starting with the latter we used the programming language R and the tools of RStudio almost entirely for all the data management work (data augmentation, training-validation sets splitting etc) whereas the models were built in Python 3.6 using keras and tensorflow as back end. Training and testing was mostly performed on a local NVIDIA GPU GTX-1080Ti, but some extra cloud computing services from the Paperspace platform were also used when the need was higher.

7.1 Easy and Difficult Predictions

As mentioned before, in this thesis, apart from developing a deep learning model that is as good as the baseline or beats it, we also wish to show the difficulty of all machine learning algorithms to generalize to previously unseen and very dissimilar data, especially in the case of protein-drug binding affinity prediction. Thus, we test our three models in two ways or settings, the "easy" and the "difficult" one.

Firstly, the "easy" setting is regards testing to validation sets generated randomly from the whole data that we possess (classic random split validation set generation and cross-validation). If we see our data as a large matrix with the proteins in the rows and the compounds in the columns where each cell corresponds to a Kd value (the binding strength of the specific protein - compound pair), then we can think of this setting as hiding some cell values from the algorithm when training and then calling the model to predict these cell values in testing time.

The more "difficult" setting is about testing the model on validation sets containing compounds not only previously unseen from the algorithm, but also quite dissimilar from the ones that has seen (i.e. the ones in the training set). This setting corresponds to hiding entire columns of the matrix from the algorithm (that also concern structurally different molecules) and then call the model to fill cells in these columns or, in other words, predict interactions of previously unseen compounds.

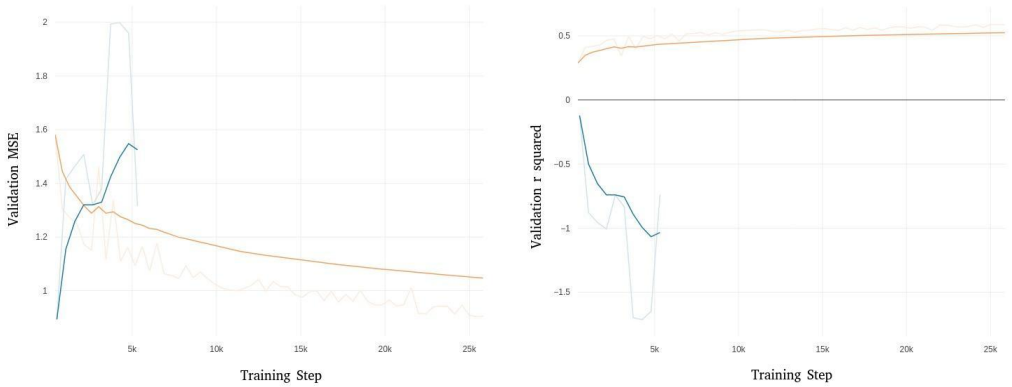
In either setting we created five validation sets and performed cross-validation to increase our confidence on the results. For more details about how we identified the most different compounds for the "difficult" setting see section 2.3.3.

7.2 Models Comparison

In this section we present our results, through our validation accuracy metrics, in the form of diagrams, for the different deep learning models that we used in this thesis. All the diagrams are made in an "easy to see" way and so the more faded lines in the background are the exact values and the bold ones are a polynomial inference on the former.

In figure 11, figure 12 and figure 13 we plot the the accuracy of deepDTA, GCNN and "autoencoder" model respectively for both the "difficult" and the "easy" validation sets (see section 7.1). In figure 14 we plot the validation accuracy for both the baseline model deepDTA and our model GCNN, when trained and validated on the exact same datasets.

The detailed observations and conclusions we can draw from all these figures are analyzed in section 8.



(a) The validation mean square error versus training time diagram for the baseline deepDTA model. The blue line represents validating in previously unseen compounds while the brown one validating only in previously unseen pairs.

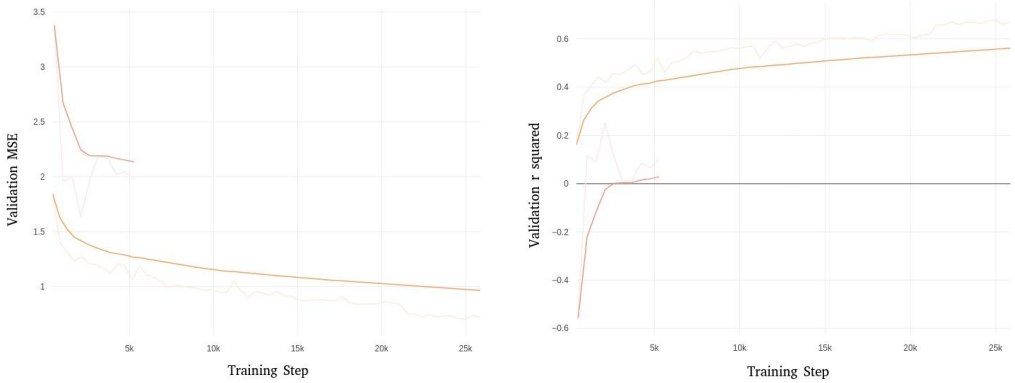
(b) The validation r^2 metric versus training time diagram for the baseline deepDTA model. The blue line represents validating in previously unseen compounds while the brown one validating only in previously unseen pairs.

Figure 11: Baseline model's accuracy on "easy" and "hard" validation sets

7.3 Uncertainty estimation

In this section we present our results, in the form of diagrams, as far the uncertainty prediction frameworks that we developed in this thesis are concerned.

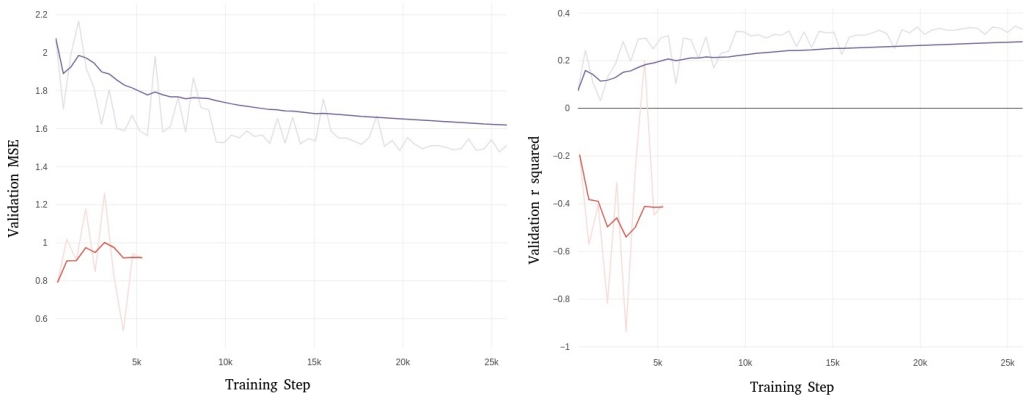
In figures 15 and 16 we present the diagrams of the test-time dropout approach for the "easy" and "difficult" setting respectively. In figure 15b we can see a scatter of the predicted points (drug-protein pairs) where the x axis is the absolute error of each point prediction (i.e. the mean of all the



(a) The validation mean square error versus training time diagram for our graph convolutional neural network model. The brown line represents validating in previously unseen compounds while the red one validating only in previously unseen pairs.

(b) The validation r^2 metric versus training time diagram for our graph convolutional neural network model. The red line represents validating in previously unseen compounds while the brown one validating only in previously unseen pairs.

Figure 12: GCNN model's accuracy on "easy" and "hard" validation sets

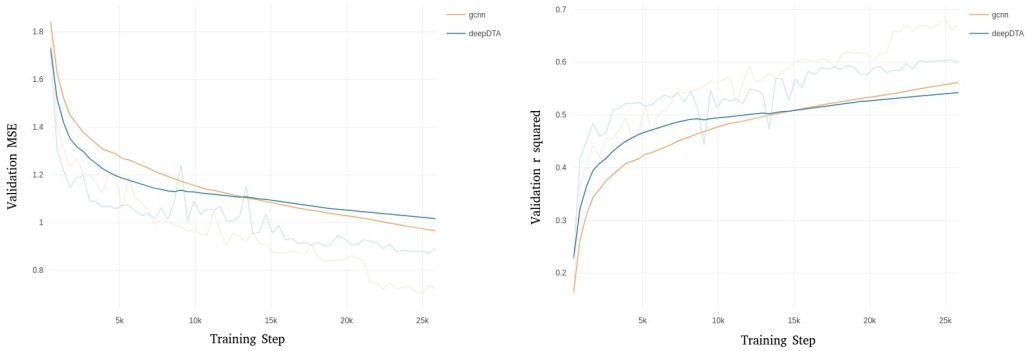


(a) The validation mean square error versus training time diagram for our Kd-with-encoder model. The red line represents validating in previously unseen compounds while the purple one validating only in previously unseen pairs.

(b) The validation r^2 metric versus training time diagram for our Kd-with-encoder model. The red line represents validating in previously unseen compounds while the purple one validating only in previously unseen pairs.

Figure 13: Kd-with-encoder model's accuracy on "easy" and "hard" validation sets

predictions for the specific drug-protein pair) and y axis is the uncertainty estimation of our test-time dropout model (i.e. the standard deviation of all the prediction for the specific drug-protein

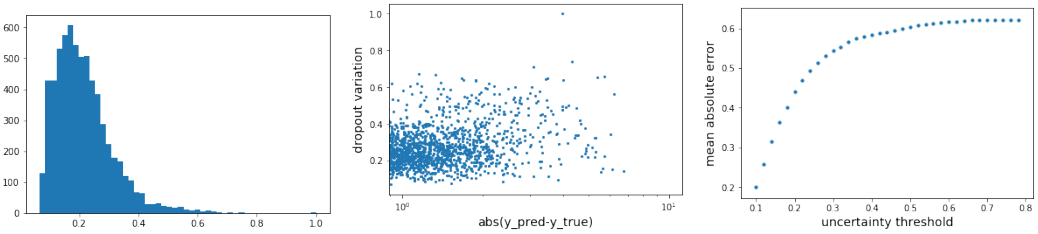


(a) The validation mean square error versus training time diagram for both the baseline and our GCNN model.

(b) The validation r^2 metric versus training time diagram for both the baseline and our GCNN model.

Figure 14: Accuracy comparison between our GCNN model and the baseline (deepDTA)

pair). In figure 15a we observe the distribution of the uncertainty predictions of our model in a histogram with 50 bars (bins). More interestingly we construct from diagram 15b the diagram 15c that results by filtering our test examples to a particular uncertainty threshold and plot the mean error for this threshold. That way looking at cases only where the uncertainty is e.g. 60%, we expect higher error than cases where uncertainty is 50%, hence the curve should be monotonically increasing. If the application demands an error less than x%, we can trust the model only in cases where the uncertainty is smaller than the corresponding threshold. Hence, by using figure 15c we can compare accuracy of the models for a desired confidence threshold of the application.



(a) Histogram of the uncertainty predictions

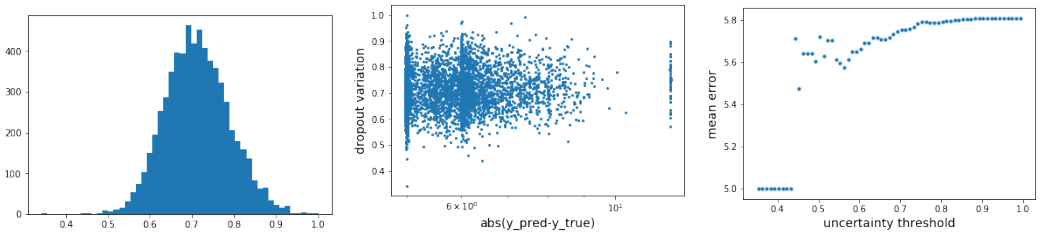
(b) Uncertainty VS error

(c) Thresholded uncertainties VS absolute error

Figure 15: Uncertainties from the "test-time Dropout", "easy setting" approach

On the same front, we present the scatter diagrams, histograms and thresholded uncertainties figures for the latent space norm and the ensemble approaches. The metric represented by the x axis (absolute error) is always the same, only the measure of uncertainty changes.

More specifically, in figures 17, 18, 19 we show the diagrams provided by the three ways to define uncertainty using the position in the latent space as discussed in section 6.3 whereas figure



(a) Histogram of the uncertainty predictions

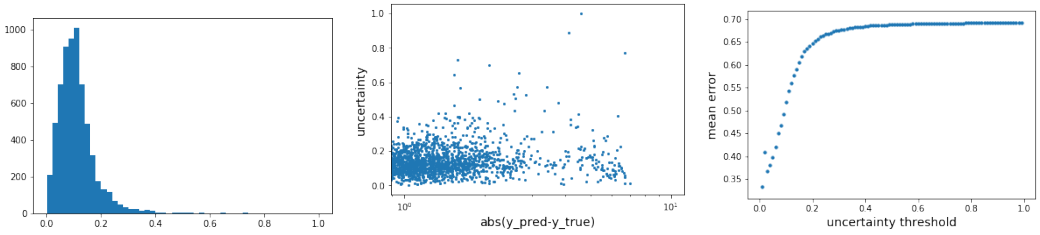
(b) Uncertainty VS error

(c) Thresholded uncertainties VS absolute error

Figure 16: Uncertainties from the "test-time Dropout", "difficult setting" approach

20 refers to the ensemble approach.

Approaches that include a finite n-dimensional radius definition (or norm limit) were tried with many different limits and the ones that yielded the best results are the ones presented in these diagrams. For more information about the results with other "radius" see table REFER HERE IN SECTION REFER HERE. All these diagrams for both the latent distance and the ensemble approach refer to the cold drug ("difficult") setting. The ones for the "easy" setting are very similar and are omitted for simplicity (in contrast with what happens with the test-time dropout approach).



(a) Histogram of the uncertainty predictions

(b) Uncertainty VS error

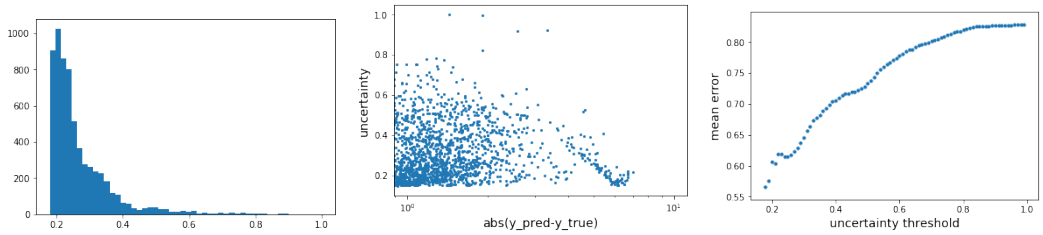
(c) Thresholded uncertainties VS absolute error

Figure 17: Uncertainties from the "error of neighbors in the latent space" approach

8 Results, Contributions & Discussion

8.1 A better model than the baseline

In figure 14 one can observe a comparison between the validation accuracy (Mean Square Error and r-squared correlation) of the baseline model before this thesis work (deepDTA) and the graph convolutional neural network developed in this thesis. Since the two models are trained and validated on the exact same dataset one can compare their accuracies directly. Thus, seeing the

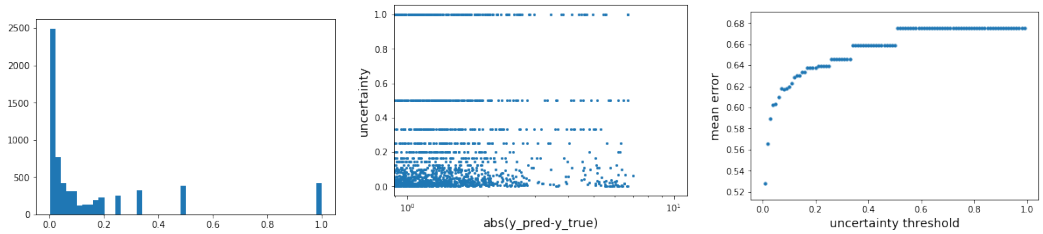


(a) Histogram of the uncertainty predictions

(b) Uncertainty VS error

(c) Thresholded uncertainties VS absolute error

Figure 18: Uncertainties from the "all median distances in the latent space" approach

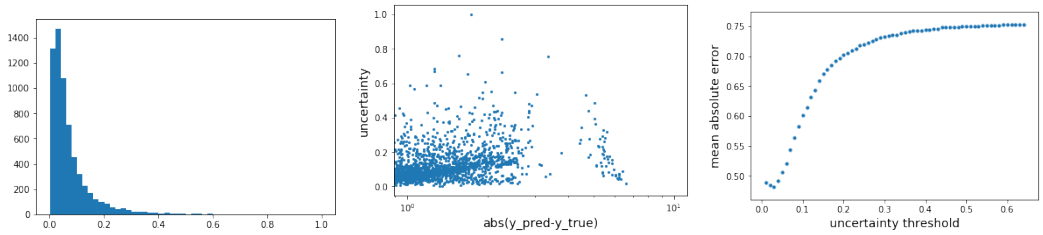


(a) Histogram of the uncertainty predictions

(b) Uncertainty VS error

(c) Thresholded uncertainties VS absolute error

Figure 19: Uncertainties from the "number of neighbors in the latent space" approach



(a) Histogram of the uncertainty predictions

(b) Uncertainty VS error

(c) Thresholded uncertainties VS absolute error

Figure 20: Uncertainties from the "deep ensembles" approach

obvious lower mean square error and highest r-squared correlation obtained by our model versus the deepDTA, we can conclude that **we beat the baseline and created a new one**, stronger on predicting the binding accuracy between protein kinases and compounds.

The better results of our GCNN model are probably due to a more detailed representation of the compounds, namely the graph neural fingerprints. In other words, the SMILES representation used by deepDTA is likely to miss too much information of the molecular physics needed to describe the

protein binding, whereas the neural fingerprints that describe the connectivity between the atoms of the molecule in a much more detailed way (kind of atom, bonds, edges) provides the algorithm with the complexity needed to learn a better binding probability distribution.

8.2 We know what we know

An important contribution of this thesis is also the exploration of the uncertainty predictions in machine learning, the different ways one can predict a level of confidence alongside with point predictions and, most importantly, validate these uncertainty estimates. We explored three ways to estimate uncertainty: the test-time dropout, the position of the test-point in the learned latent space and the deep ensembles. We validated our uncertainty predictions using the pearson correlation metric between the uncertainty and the absolute error of the validation points. In table 3 we show the correspondent results for the three ways to calculate uncertainty, while in table 4 we focus on the latent distance approach and show some results on how the different norms (finite n-dimensional radius) influence the pearson correlations (which is equivalent to the validation accuracy of the uncertainty frameworks). Table 3 includes the results for the best "radius".

It is very important to mention that both the tables 3 and 4 are made for the "easy" setting (i.e. random split validation set). In that case the dropout method ranks as second best (as we can see from table 3). However, when calculating the same pearson correlations for the "difficult" (cold-drug) setting, the dropout method collapses entirely. To be more specific, although the uncertainties that it gives us are very small (high-confidence predictions), as we can see from figure 16b these uncertainties have no correlation at all with the absolute error (the pearson correlation in that case is less than 0.007). That shows us that in cases where the model doesn't learn much the deviation of the predictions with test-time dropout is pretty small (and hence small uncertainties) although the absolute error is huge. In other words, the weights are robust, the algorithm is certain that it hasn't learn anything! In contrast, the latent space and the the ensemble methods give similar results for both the random and the cold-drug setting, which makes them more appropriate to use when we the predictions of the model are expected to be difficult to make.

	Latent Distance Approach			Dropout	Ensemble
	number of neighbors	median of distances	error of neighbors		
Pearson Correlation	0.13	0.14	0.29	0.4	0.44

Table 3: Comparison of different uncertainty estimation methods using pearson correlation

8.3 Data contribution

Another important contribution of this thesis is, in the author's opinion, a large unbiased, diverse and reliable dataset on the binding affinity of protein kinases with compounds. Although it may seem trivial, the unification of many different resources into one universally formatted

	"Radius"	Pearson Correlation
Number of Neighbors	3	0.07
	2	0.08
	1	0.12
	0.67	0.13
	0.5	0.1
	0.3	0.05
Error of Neighbors	3	0.26
	2	0.28
	1	0.29
	0.67	0.28
	0.5	0.27
	0.3	0.26

Table 4: Comparison of the latent space uncertainty methods for different "radius"

dataset is quite time consuming and presents unexpected difficulties. At the same time, having a common dataset that includes all the current publicly available knowledge of humanity about protein kinases Kd binding, is a very useful tool for all the scientific community that uses deep learning to address the specific problem. The dataset made out of this thesis work has been made publicly available and can be found on the github of the BioSystems Lab of NTUA (`https://github.com/BioSysLab/Drug-Kinase-Binding`).

9 The IDG-DreamChallenge Competition

A large part of the work that led to this thesis was carried out while participating, as a lab, in the IDG-DREAM Drug-Kinase Binding Prediction Challenge [41]. Through the challenge we did most of the data augmentation work (see section 4.2), as well as we developed our graph convolutional neural network (see section 5.2) and LSTM autoencoder (see section 5.3). The best predictions for the test set of the competition came out of the predict-with-encoder architecture we implemented. Having in mind how difficult it is to really generalize to new compound scaffolds never previously seen during training [47], an encoder that has been trained to represent the combined train and test-set distribution was expected to boost performance when its encoded feature vector was fed for further training.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[2] Ulf Madsen Kristian Stromgaard, Povl Krogsgaard-Larsen. *Textbook of Drug Design and Discovery*. CRC Press, 5 edition, 8 2016.

[3] Scannell JW1, Blanckley A, Boldon H, and Warrington B. Diagnosing the decline in pharmaceutical r&d efficiency. *Nat Rev Drug Discov.*, 3:191, Mar 2012.

[4] Mohammed Zourob. *Recognition Receptors in Biosensors*. Springer, New York, NY, 2010.

[5] J Andrew McCammon. Theory of biomolecular recognition. *Current Opinion in Structural Biology*, 8(2):245–249, 1998.

[6] Irwin D. Kuntz. Structure-based strategies for drug design and discovery. *Science*, 257(5073):1078–1082, 1992.

[7] Rachelle J. Bienstock. Computational drug design targeting protein-protein interactions. *Current Pharmaceutical Design*, 18(9):1240–1254, 2012.

[8] John F. Kolen. *A Field Guide to Dynamical Recurrent Networks*. Wiley-IEEE Press, 2001.

[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[11] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, May 2013.

[12] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, December 2010.

[13] Y Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35:1798–1828, 08 2013.

[14] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85 – 117, 2015.

[15] Itamar Arel, Derek Rose, and Thomas Karnowski. Deep machine learning - a new frontier in artificial intelligence research [research frontier]. *Computational Intelligence Magazine, IEEE*, 5:13 – 18, 12 2010.

[16] Erik Gawehn, Jan A. Hiss, and Gisbert Schneider. Deep learning in drug discovery. *Molecular Informatics*, 35(1):3–14, December 2015.

[17] Vladimir Svetnik, Andy Liaw, Christopher Tong, J. Christopher Culberson, Robert P. Sheridan, and Bradley P. Feuston. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of Chemical Information and Computer Sciences*, 43(6):1947–1958, November 2003.

[18] Jianlin Cheng and Pierre Baldi. Improved residue contact prediction using support vector machines and a large feature set. *BMC Bioinformatics*, 8(1):113, Apr 2007.

[19] Paul Raccuglia, Katherine C. Elbert, Philip D. F. Adler, Casey Falk, Malia B. Wenny, Aurelio Mollo, Matthias Zeller, Sorelle A. Friedler, Joshua Schrier, and Alexander J. Norquist. Machine-learning-assisted materials discovery using failed experiments. *Nature*, 533(7601):73–76, May 2016.

[20] Jean-Louis Reymond and Mahendra Awale. Exploring chemical space for drug discovery using the chemical universe database. *ACS Chemical Neuroscience*, 3(9):649–657, May 2012.

[21] Andreas Bender and Robert C. Glen. Molecular similarity: a key technique in molecular informatics. *Org. Biomol. Chem.*, 2:3204–3218, 2004.

[22] Gerald M. Maggiora and Veerabahu Shanmugasundaram. *Molecular Similarity Measures*, pages 39–100. Humana Press, Totowa, NJ, 2011.

[23] Andre Droschinsky, Nils M. Kriege, and Petra Mutzel. Finding largest common substructures of molecules in quadratic time. *CoRR*, abs/1610.08739, 2016.

[24] Gerald Maggiora, Martin Vogt, Dagmar Stumpfe, and Jürgen Bajorath. Molecular similarity in medicinal chemistry. *Journal of Medicinal Chemistry*, 57(8):3186–3204, November 2013.

[25] Hu C-Y. Willett P. Holliday, J.D. *Combinatorial Chemistry & High Throughput Screening*. Bentham Science Publishers, 2002.

[26] Pierre Baldi and Ramzi Nasr. When is chemical similarity significant? the statistical distribution of chemical similarity scores and its extreme values. *Journal of Chemical Information and Modeling*, 50(7):1205–1222, June 2010.

[27] Noel M. O'Boyle and Roger A. Sayle. Comparing structural fingerprints using a literature-based similarity benchmark. *Journal of Cheminformatics*, 8(1), July 2016.

[28] Frank R. Burden and David A. Winkler. Robust QSAR models using bayesian regularized neural networks. *Journal of Medicinal Chemistry*, 42(16):3183–3187, July 1999.

[29] Frank R. Burden and David A. Winkler. A quantitative structure-activity relationships model for the acute toxicity of substituted benzenes toTetrahymena pyriformisUsing bayesian-regularized neural networks. *Chemical Research in Toxicology*, 13(6):436–440, June 2000.

[30] Hongying Du, Jie Wang, Zhide Hu, Xiaojun Yao, and Xiaoyun Zhang. Prediction of fungicidal activities of rice blast disease based on least-squares support vector machines and project pursuit regression. *Journal of Agricultural and Food Chemistry*, 56(22):10785–10792, November 2008.

[31] Paola Gramatica. Principles of QSAR models validation: internal and external. *QSAR & Combinatorial Science*, 26(5):694–701, May 2007.

[32] Jitender Verma, Vijay Khedkar, and Evans Coutinho. 3d-QSAR in drug design - a review. *Current Topics in Medicinal Chemistry*, 10(1):95–115, January 2010.

[33] Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, February 2015.

[34] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively Multitask Networks for Drug Discovery. *arXiv e-prints*, page arXiv:1502.02072, Feb 2015.

[35] Tyler B. Hughes, Grover P. Miller, and S. Joshua Swamidass. Modeling epoxidation of drug-like molecules with a deep machine learning network. *ACS Central Science*, 1(4):168–180, June 2015.

[36] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, and Sepp Hochreiter. DeepTox: Toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3, February 2016.

[37] ncats. Nih, 2014.

[38] Alessandro Lusci, Gianluca Pollastri, and Pierre Baldi. Deep architectures and deep learning in chemoinformatics: The prediction of aqueous solubility for drug-like molecules. *Journal of Chemical Information and Modeling*, 53(7):1563–1575, July 2013.

[39] Izhar Wallach, Michael Dzamba, and Abraham Heifets. AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. *arXiv e-prints*, page arXiv:1510.02855, Oct 2015.

[40] Polina Mamoshina, Armando Vieira, Evgeny Putin, and Alex Zhavoronkov. Applications of deep learning in biomedicine. *Molecular Pharmaceutics*, 13(5):1445–1454, March 2016.

[41] (Author Name Not Available). Idg-dream drug-kinase binding prediction challenge, 2018.

[42] Rajan Sharma, Stephan C. Schürer, and Steven M. Muskal. High quality, small molecule-activity datasets for kinase research. *F1000Research*, 5:1366, October 2016.

[43] Mindy I Davis, Jeremy P Hunt, Sanna Herrgard, Pietro Ciceri, Lisa M Wodicka, Gabriel Pallares, Michael Hocker, Daniel K Treiber, and Patrick P Zarrinkar. Comprehensive analysis of kinase inhibitor selectivity. *Nature Biotechnology*, 29(11):1046–1051, October 2011.

[44] David H. Drewry, Carrow I. Wells, David M. Andrews, Richard Angell, Hassan Al-Ali, Alison D. Axtman, Stephen J. Capuzzi, Jonathan M. Elkins, Peter Ettmayer, Mathias Frederiksen, Opher Gileadi, Nathanael Gray, Alice Hooper, Stefan Knapp, Stefan Laufer, Ulrich Luecking, Michael Michaelides, Susanne Müller, Eugene Muratov, R. Aldrin Denny, Kumar S. Saikatendu, Daniel K. Treiber, William J. Zuercher, and Timothy M. Willson. Progress towards a public chemogenomic set for protein kinases and a call for contributions. *PLOS ONE*, 12(8):e0181585, August 2017.

[45] Yinghong Gao, Stephen P. Davies, Martin Augustin, Anna Woodward, Umesh A. Patel, Robert Kovelman, and Kevin J. Harvey. A broad activity screen in support of a chemogenomic map for kinase signalling research and drug discovery. *Biochemical Journal*, 451(2):313–328, April 2013.

[46] Theonie Anastassiadis, Sean W Deacon, Karthik Devarajan, Haiching Ma, and Jeffrey R Peterson. Comprehensive assay of kinase catalytic activity reveals features of kinase inhibitor selectivity. *Nature Biotechnology*, 29(11):1039–1045, October 2011.

[47] Izhar Wallach and Abraham Heifets. Most ligand-based classification benchmarks reward memorization rather than generalization. *Journal of Chemical Information and Modeling*, 58(5):916–932, April 2018.

[48] M. Rupp, E. Proschak, and G. Schneider. Molecular similarity for machine learning in drug development. *Chemistry Central Journal*, 2(1):P10, Mar 2008.

[49] Hakime Öztürk, Elif Ozkirimli, and Arzucan Özgür. Deepdta: Deep drug-target binding affinity prediction. 2018.

[50] T. Pahikkala, A. Airola, S. Pietila, S. Shakyawar, A. Szwajda, J. Tang, and T. Aittokallio. Toward more realistic drug-target interaction predictions. *Briefings in Bioinformatics*, 16(2):325–337, April 2014.

[51] Tong He, Marten Heidemeyer, Fuqiang Ban, Artem Cherkasov, and Martin Ester. SimBoost: a read-across approach for predicting drug–target binding affinities using gradient boosting machines. *Journal of Cheminformatics*, 9(1), April 2017.

[52] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep. In *Proceedings of the 24th International on Conference on Information and Knowledge Management*. ACM Press, 2015.

[53] Aditya Grover and Jure Leskovec. node2vec. In *Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining*. ACM Press, 2016.

[54] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. 2015.

[55] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints, 2015.

[56] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. 2016.

[57] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[58] Cheng-Yuan Liou, Jau-Chi Huang, and Wen-Chie Yang. Modeling word perception using the elman network. *Neurocomputing*, 71(16-18):3150–3157, October 2008.

[59] Cheng-Yuan Liou, Wei-Chen Cheng, Jiun-Wei Liou, and Daw-Ran Liou. Autoencoder for words. *Neurocomputing*, 139:84–96, September 2014.

[60] Wikipedia. Long short-term memory. http://en.wikipedia.org/w/index.php?title=Long\%20short-term\%20memory\&oldid=898571612, 2019. [Online; accessed 03-June-2019].

[61] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms (Chapman & Hall/Crc Machine Learnig & Pattern Recognition)*. Chapman and Hall/CRC, 2012.

[62] Glenn Shafer Vladimir Vovk, Alexander Gammerman. *Algorithmic Learning in a Random World*. Springer-Verlag, 2005.

[63] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2005.

[64] Michalis Titsias and Neil D. Lawrence. Bayesian gaussian process latent variable model. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 844–851, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[65] Radford M. Neal. *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*. Springer, 1996.

[66] Christopher K. I. Williams. Computation with infinite neural networks. *Neural Computation*, 10(5):1203–1216, July 1998.

[67] David J C Mackay. A practical bayesian framework for backprop networks. 1991.

[68] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, 2015.

[69] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles, 2016.

[70] D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pages 55–60 vol.1, June 1994.