Deep Learning Methodologies in Drug Kinase prediction

Severi Vapalahti

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.
Espoo May 15, 2022

Supervisor

Prof. Juho Rousu

Advisor

Wang Tianduanyi

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Severi Vapalahti

Title Deep Learning Methodologies in Drug Kinase prediction

Degree programme Life Science Technologies

Major Bioinformatics and Digital Health **Code of major** SCI3092

Supervisor Prof. Juho Rousu

Advisor Wang Tianduanyi

Date May 15, 2022 **Number of pages** 48 **Language** English

Abstract

The misbehaviour of the enzymatic protein kinases can lead to development of tumors. Small molecule kinase inhibitors are known to be effective therapeutics in cancer treatment but it is difficult to find selective drugs. To avoid laborious biochemical experiments, the drug-target space can be scanned with computer models. Multiple predictive algorithms have been developped based on different machine learning paradigms. This thesis introduces a predictive recurrent convolutional neural network regression model inspired by two previous deep learning algorithms. The model had good performance when tested with a validation set isolated from the training data. However, some of its performance reduced when its generalizability to different dataset was tried out. Nevertheless, taking into account the many potential ways the model can be improved and comparatively light design, similar architectures can have prospect for scanning potential hits in early stage of discovering therapeutic kinase inhibitors.

Keywords drug kinase interactions, drug target interactions, deep learning, machine learning, cancer therapeutics, drug development, affinity

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Severi Vapalahti

Työn nimi Deep Learning Methodologies in Drug Kinase prediction

Koulutusohjelma Life Science Technologies

Pääaine Bioinformatics and Digital Health **Pääaineen koodi** SCI3092

Työn valvoja Prof. Juho Rousu

Työn ohjaaja Wang Tianduanyi

Päivämäärä May 15, 2022 **Sivumäärä** 48 **Kieli** Englanti

Tiivistelmä

Tiivistelmässä on lyhyt selvitys kirjoituksen tärkeimmästä sisällöstä: mitä ja miten on tutkittu, sekä mitä tuloksia on saatu.

Avainsanat kinaasi,syväoppiminen,koneoppiminen,lääkekehitys

Preface

I want to thank my supervisor Professor Juho Rousu for providing intresting topic and my advisor Wang Tianduanyi for helping me out throughout this process.

Otaniemi, May 15, 2022

Severi Vapalahti

Contents

Abstract 3

Abstract (in Finnish) 4

Preface 5

Contents 6

Symbols and abbreviations 7

1 Introduction 8

2 Theoretical Background I: Druggable Kinases 10
2.1 Therapeutic potential of Protein Kinases . . . 10
2.2 Protein Kinases and Cancer Treatment . . . 11

3 Theoretical Background II: Kinase Inhibitors and Deep Learning 14
3.1 Deep Learning Overview . . . 14
3.2 Deep Learning for Drug Target Interactions . . . 20

4 Methods 24
4.1 Data . . . 24
4.2 Novel Method . . . 25
4.3 Hyperparameter Tuning . . . 29

5 Results 34

6 Discussion 40
6.1 Potential Improvements . . . 40
6.2 Benefits of the model . . . 42
6.3 Further Study . . . 42
6.4 Conclusions . . . 43

7 Summary 44

Symbols and abbreviations

Symbols

$\mathbf{x}/\mathbf{X}$	input vector/matrix in a neural network
$\mathbf{h}^{(i)}$	hidden layer i in a neural network before activation
$\mathbf{z}^{(i)}$	hidden layer i in a neural network after activation
$\mathbf{W}^{(i)}$	weight matrix for layer i in a neural network
$\hat{\mathbf{y}}$	output vector / prediction
$\mathbf{y}$	true value

Operators

∇_i	gradient with respect to weight matrix $\mathbf{W}^{(i)}$
$dim(.)$	dimensionality of a vector or matrix
$\sigma(.)$	(undefined) activation function / non-linearity in neural network
$L(.)$	(undefined) loss function
$MSE(.)$	mean squared loss
$\omega(.)$	transformation from the input alphabet to integer value
$\Omega(.)$	one-hot-coding of a character (vector) or string (matrix)
$\mathcal{GP}(.))$	gaussian process
$\mu(.)$	mean/expected value
$k(.)$	kernel between two vectors

Abbreviations

ANN	artificial neural network
API	application programming interface
ATP	adenosine triphosphate
CNN	convolutional neural network
DL	deep learning
DNA	deoxyribonucleic acid
DTI	drug target interaction
DTC	drug target commons
GCN	graph convolutional neural network
GPGS	gaussian process based grid search
GPU	graphical processing unit
GRU	gated recurrent unit
KI	kinase inhibitor
MAPK	mitogen-activated protein kinase (pathway)
ML	machine learning
ReLU	rectified linear unit
RMSE	root mean squared error
RNN	recurrent neural network
SMILES	simplified molecular input line entry system

1 Introduction

At first glance, biology and computer science might seem like the opposite ends of the scientific spectrum. Biology studies organic, living and breathing entities, whereas computer science uses inorganic materials to make numerical computations. Surprisingly, below the surface, they share quite a few characteristics. They are both fields that study mechanisms to solve difficult problems. Biology provides approaches that nature has developed through the process of evolution. Computer science, in turn, offers problem-solving mechanisms invented by us humans through scientific method. Furthermore, given that the central dogma of biology is ultimately an explanation for the direction of information flow, both disciplines essentially grapple with questions relating to information processing.

A computer program is fundamentally a sequence of bytes of information. If such a sequence is sent to another computer with compatible hardware and software, it runs identically. All the necessary information is in the file that contains the source code. These programs are stored in ones and zeros using transistors that are able to hold on to small voltages. Whereas computers store and handle information in binary units, nature has learned to store information in the base-4 system, a macromolecule known as the deoxyribonucleic acid (**DNA**). Reading DNA or even a full genome has become very fast and efficient with the modern next-generation sequencing techniques. Regardless, there is a long way to being able to interpret this genetic data applicably. In computers, the more sophisticated systems are the ones that operate on top of the stored information, reading and writing on it. There exist conceptually similar systems in the biological domain as well. Any living organisms rely on highly organized and detailed procedures that are carried out and regulated by specific proteins.

Proteins are composed of long chains of amino acids and the complete information for synthesizing them is stored in the DNA. Similar to transferring information between computers, the same sequence of genetic information always corresponds to identical protein, even if it's removed from the original organism to another similar biochemical environment. Proteins can be thought of as biological programs executing specific tasks. They are microscopic machines that serve in vital yet intricate processes such as cell metabolism, mitosis, transcription among many others. On top of the individual proteins, there exists cellular signal pathways with enzymes called protein kinases regulating these processes.

From the drug development perspective, kinases are very valuable targets, since they can be used to impact the activation of many important biochemical events. For this reason, Edmond Fischer and Edwin Krebs were awarded a nobel prize for their work with protein phosphorylation, an event that is carried out by kinases. When searching for potential drug compounds, it is important to study how the compound interacts with proteins' binding sites. Only if the molecule binds with one of the binding sites, can it have a therapeutic or adverse effect on the targeted protein. On that account, studying these microscopic interactions reveal highly valuable information for drug development and other pharmaceutical fields. The measure for the magnitude of the binding interaction is referred to as affinity. The traditional

way for obtaining the affinity between a drug-protein pair is conducted in laboratory conditions by examining the level of interaction in-vitro. Obviously, measuring the affinity experimentally is very time consuming, laborious and expensive, considering the number of possible drug-protein combinations. To navigate through this chemical space, computational methods can be applied to learn key characteristics and make predictions about possible interactions.

Today, researchers are able to analyse data with efficiency and accuracy that would have been unimaginable for scientists before the modern age of digital computing. The biomedical field, among many other disciplines, has benefited tremendously from the recent developments in data and computer science. Machine learning (**ML**) is a field in computational statistics that uses algorithmic techniques to infer knowledge from data in an efficient and intelligent way. There exists a variety of different Machine learning subfields. Certainly the most popularized branch by the media of today is deep learning (**DL**), which studies mathematical models called deep neural networks. The idea of a neural network (**NN**) was loosely inspired by the neural connections in the brain, further demonstrating the intersections between biology and computer science.

Many machine learning algorithms including deep learning methods for predicting drug-kinase interactions have emerged recently. As such a benchmarking study known as IDG-DREAM Drug Kinase Binding Prediction Challenge[1] was organized to evaluated and compare the predictive power of various machine learning models. The methods that took part in the competition were form multiple ML families such as deep learning, kernel methods and gradient boosting. In this thesis, a novel deep learning based drug-kinase predictor is introduced and its performance is analysed with the same metrics and testing dataset used in the DREAM-challenge.

2 Theoretical Background I: Druggable Kinases

2.1 Therapeutic potential of Protein Kinases

Proteins are fundamental components in biology and key target for research in drug discovery. Not only are they building blocks for muscles and organs, but they are also responsible of coordinating vital tasks in all living organism. A single protein can have a variety of functions such as transporting small compounds, supporting the cell structure, or attaching molecules to each other. Moreover, in collaboration with each other, proteins are able to carry out very complex processes. Proteins are built out of long chains of amino acids. The amino acid sequence is often easily obtainable but rarely sufficient information to describe the protein or predict its functionality. Additional information such as the 3D-structure is often required to comprehensively understand how a protein acts in its environment.

The biochemical environments are very dynamic, under constant change. The proteins often have very specific functions that might not be required constantly. Some processes might be active only during sleep, digestion or in absence or abundance of a certain chemical compound. This raises the question, how do these macromolecules know when their function is needed? Presumably the most essential regulatory mechanism is the so called protein phosphorylation. It is a reversible covalent modification, where a phosphate group is attached to a protein in a specific position. This forces the protein to undergo a conformational change thereby making it capable or incapable of contributing to a certain reaction.

Protein Phosphorylation

The phosphate ion $[PO_4]^{3-}$ consists of one phosphorus and four oxygen atoms. When this composition of this anion is attached to organic compounds it serves as a functional group. Each of the three negatively charged oxygens are capable of forming an ester bond. This ability to connect organic structures to each other is one of the reasons why phosphates play important role in biochemistry. They connect the nucleotides to each other in both DNA and **RNA** (ribonucleic acid). They also serves as the key energy transportation mechanism in the form of adenosine phosphates. In addition, the phosphates are shown to be the key component in regulating biochemical processes.[2]

The prevalence of the phosphate in biology is very central. The phosphates are ideal for the nature due to their high ionizability and the stable P(V)-oxidation state. This makes them unable to penetrate the cell membrane and therefore preventing anything containing phosphates from escaping from the cell. Also, the ionization makes them potential targets for reactions under right circumstances. By exploiting the specific characteristics of phosphate group, nature has learned to manipulate them in very clever ways.[3].

In a process called phosphorylation, a phosphate group is added to a molecule. This phenomenon is utilised to regulate enzymatic reactions. A catalyst is a substance which lowers the required activation energy for a chemical reaction. The rate of such reaction occurring increases and significantly speeds up the process. The cataylst is

not consumed in the reaction, and therefore is not required in the same magnitudes as the reactants. Enzymes are proteins that act as catalysts and their activity is regulated through phosphorylation. Adding a phosphate to an enzyme makes the enzyme active or deactive to catalyze a certain reaction. The added phosphate groups act as a molecular switches that are essential for navigating through complex biochemical processes. This protein phosphorylation, in turn, is carried out by a specific group of enzymes called the protein kinases.[4]

Function and structure of the Protein Kinase

Protein kinases are enzymes that regulate many biological processes. Kinases are one the three phosphotransferases: Protein kinases add phosphate group from adenosine triphosphate (**ATP**), phosphorylases add phosphate from inorganic compounds and phospholyses remove the phosphate group. Protein Kinases can be categorized into two subgroups, dedicated protein Kinases and Multifunctional protein Kinases. Dedicated protein Kinases phosphorylate either single protein or group of closely related proteins. Multifunctional protein Kinases can catalyze the phosphorylation for larger group of proteins.[CITATION NEEDED]

The first found and isolated kinase was the phosphorylase kinase (**PhK**). PhK phosphorylates an phosphorylase enzyme called glycogen phosphorylase, which in turn catalyses the breakdown of glycogen into glucose. This process known as the glycogenolysis is crucial to stabilise blood sugar level. After the discovery of PhK number of know kinases has grown exponentially. It is known that significant part of all eukaryote genomes are for coding protein kinases and this region is called the Kinome. The human kinome consists of total of 518 protein kinases and they constitute almost 2% of all human genes. [5]

The structure of the protein kinases consists of two domains, catalytic and regulatory Both domains consist of several subdomains. The catalytic domain is the functional part which is responsible for catalysing the targeted reaction. The regulatory subunit, often referred as the ATP-binding site, is location where the phosphorylation occurs. As already stated, the attached phosphate modifies the conformation and changes its activity. Interestingly, the aminoacid sequence of the regulatory domain is well conserved between different kinases, meaning that phosphorylase requires very specific binding. The kinases often phosphorylate other kinases in pathways instead of the end product catalysing enzyme. The aforementioned glycogenolysis for example, requires a phosphorylation cascade of two kinases to activate the glycogen phosphorylase.

2.2 Protein Kinases and Cancer Treatment

An oncogene is a gene that is altered in such a way that it starts to code dysfunctional proteins resulting in cancer cells. Although other causes for cancer exist, this alteration is often caused by a mutations induced by a carcinogenic agent. In the whole human genome, which consists over 30,000 genes, only handful of genes have potential to become oncogenes. These genomic sites are called proto-oncogenes.

Retroviruses inject their genetic information in RNA. As such, they rely on reverse transcription to modify the cell genome. It was noticed that the retroviruses have the ability to rarely change the cellular genome in such a way that some protein kinases have enhanced activity resulting in development of tumor. This discovery revealed the role of kinases as proto-oncogenes. Mitogen-activated protein kinase pathway abreviated as **MAPK** is a kinase pathway. The MAPK pathway is involved in the transcription of genes necessary for cell proliferation and survival. Therefore, the MAPK pathway functioning incorrectly can lead to development of cancer cells.[6]

Since kinases play important role in cell signaling during cell proliferation, small mutations in kinome may result continually dividing cancer cells. By inhibiting the kinase activity, this futile cell multiplication can be mitigated and potentially be used to treat and prevent the development of cancer. As a result, the protein kinase inhibitors (**KI**s) are effective drugs in cancer treatment. For example, using small molecules that inhibit the Raf-kinase, the first kinase of the MAPK-pathway, has resulted with promising outcomes for melanoma patients.[7]. There currently exists 37 clinically accepted KIs[8] and many others are developed and might get approved soon.

Drug Kinase Interactions

The biological activity of a chemical compound is its effect on the living matter. For a compound to have potential therapeutic effects, it needs to have biological activity. Studying the effects of the bioactive compunds requires knowledge on how their molecular composition interact int the environment of the targeted macromolecules. These interactions, labelled as drug target interactions (**DTI**s), offer valuable knowledge for the drug discovery, development and related fields. The DTI research investigates how small molecules bind to targets and modulate their function. When studying drug-target relations, the input domain is the vast map of all possible combinations between a small molecule and the targeted macromolecule. Moreover, it is not enough to study the potential therapeutic effects, but the adverse effects need to be known as well before a drug can be made commercial.

When searching for potential drugs, there exists four potential factors that can be targeted with small molecule therapeutic agents. These are polysaccharides, lipids, nucleic acids and proteins, latter of which are often the most successful targets.[9] For a drug to be effective, it must bind to its molecular target with a reasonable degree of potency. This potency is called the affinity. Binding affinity can be evaluated using several metrics. The most commonly used metrics are K_d, K_i, IC_{50} and EC_{50}.

The K_d value is general dissociation constants. It can be computed as the ratio between concentration of dissociated units of protein and ligands and the ligand-protein complex. High K_d value means that equilibrium is weighted heavily towards the ligands that are not incorporated with the protein in anyway. The inhibition constant K_i is similar to the K_d, but its more specifically a metric for equilibrium between a ligand that operates as an inhibitor and the enzyme.[10] The IC_{50} measures how effectively a substance inhibits the studied reaction catalyzed by the target protein. It stands for the concentration of inhibitor at 50% of the total inhibition.

The EC_{50} stands for effective concentration at 50% of the total effect.[11]

The KIs are often designed to bind in the kinase's regulatory site to prevent phosphorylation occuring by steric hinderance and thereby inhibiting the targeted reaction. Since the ATP-binding site is well conserved throughout the kinome, very selective molecules are required to only bind to specific kinases. Therefore, one problem when developing efficient KIs is that many small molecules have poor selectivity and might target more than 100 kinases at same time.[12] Due to the poor selectivity of KIs, it is crucial to understand all the possible effects in target and off-targets.

Studying drug-kinase interactions in laboratory is expensive and time consuming due huge space of possible drug kinase pairs. To avoid time consuming and costly mapping of the drug-kinase space experimentally, computational modelling can be exploited when searching for potential therapeutics. Compound libraries can be screened for potential hits using computational models. Although predictions are not always correct, the compounds can always be validated in physical laboratory.

3 Theoretical Background II: Kinase Inhibitors and Deep Learning

3.1 Deep Learning Overview

Deep Learning is a family of machine learning methods that are based on a concept known as artificial neural networks (**ANN**). Although neural networks have emerged in the spotlight and gained popularity only in the recent years, the basic framework for ANN has been exiting for decades. The first scientific publication about the topic was published in 1958 by Frank Rosenblatt in his paper about the perceptron model[13]. Despite being over 60 years old discovery, deep learning became practical only after the expansion of the available data and exponential rise in the computing power.

Deep learning algorithms can be used to solve common machine learning problems such as regression and classification tasks, but they are extremely robust and fit for solving even more complex problems. The structure for the networks can vary drastically. In general, the architecture is always based on several computational layers consisting of several neurons. These neurons are small units that carry out relatively simple computations often followed by a non-linear transformation. In the figure 1 is shown the architechture of a very simple ANN with two hidden layers, three inputs and two outputs.

This thesis will not go into detail about the foundations of deep learning. However, the core ideas and formulations that are required to understand the model presented in in the methods chapter, are briefly discussed in this chapter.

Feed Forward Network

A fully connected (FC) layer is considered as the most basic layer in neural networks. Each FC layer i has a set of learnable weights $\mathbf{W}^{(i)}$ which are optimized in the training phase. Similar to linear regression, the output of any layer is computed as a matrix multiplication of it's weights and the input. For example, the first layer takes the n-dimensional data $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T$ as input:

$$\mathbf{h}^{(1)} = \mathbf{W}^{(1)}\mathbf{x} \tag{1}$$

Where the first hidden layer $\mathbf{h}^{(1)} = \begin{bmatrix} h_1^{(1)} & h_2^{(1)} & \dots & h_m^{(1)} \end{bmatrix}^T$ has m neurons. Therefore the length of the input vector and the number of neurons in the layer defines the dimensionality of the weight matrix as $\mathbf{W}^{(1)} \in \mathbb{R}^{(m \times n)}$.

The output dimensionality is always same as the number of neurons. The output is often followed by a non-linearity inducing function $\sigma(.)$, such as the rectified linear unit (ReLU), hyperbolic tangent, sigmoid function or softmax. The activated signal is denoted with $\mathbf{z}$ and can be calculated for any layer i as:

$$\mathbf{z}^{(i)} = \sigma(\mathbf{h}^{(i)}) \tag{2}$$

Without this non-linearity, computing the signal through the multiple linear layers would simply be a sequence of matrix multiplications. This in turn, would result

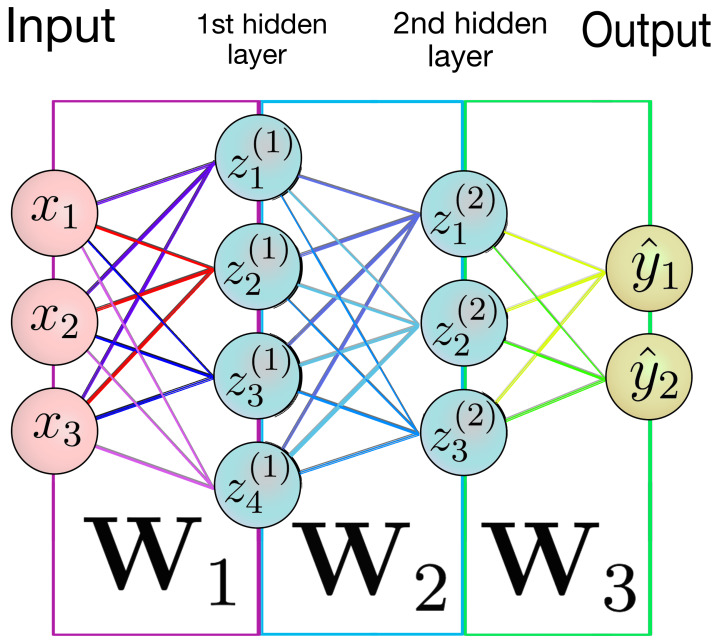


Figure 1: An example artificial neural network with two fully connected layers. The neural network takes three dimensional data x as the input. The architecture consists of two hidden layers with four and three neuron respectively. The output layer is two dimensional

in a single product matrix and the model would collapse into a linear regression model. The non-linearity is therefore a key element in the ANN. The most popular non-linearity today is the ReLU, which returns the original value if the output is positive and zero if the output is negative. This is parallel to action potential between the neurons, which require certain threshold before they transmit the signal forward. Although the biological justification is [very loose], the ReLU activation often results with the best performance and is hence the most commonly used activation unit.

Similar to the first hidden layer, the activated output for any fully connected hidden layer $\mathbf{z}^{(i)}$ can be calculated as a matrix multiplication of it's weights and the signal form the previous layer $\mathbf{z}^{(i-1)}$ as:

$$\mathbf{z}^{(i)} = \sigma(\mathbf{W}^{(i)}\mathbf{z}^{(i-1)}) \tag{3}$$

The dimensionality of the weight matrix is completely determined by the number of neurons as $dim(\mathbf{W}^{(i)}) = (dim(\mathbf{h}^{(i)}), dim(\mathbf{z}^{(i-1)}))$.

Each node of the fully connected layer will take the output of every node in the previous layer as an input, hence the name fully connected. The dimensions for output always depend on number of neurons. This is the case with FC layer and any other layer. Layers often also contain a bias term, which is often included in the notation in input vector as a constant one.

The signal in the ANN is passed from input vector $\mathbf{x}$ through the all the hidden layers in the network until the output layer $\hat{\mathbf{y}}$ is reached. The most task require

more sophisticated units in addition to fully connected layers, but the core structure behind the fully connected layer remains the determining factor behind deep learning. [14]

Backpropagation

As emphasized, the deep learning has gained popularity thanks to the increase in amount of available data and computing power. However, one key observation that enabled the use of neural networks was the backpropagation, the foundation behind the iterative optimization of the weights in the network during the training process. Most deep learning libraries perform the backproagation automatically and so the model designer does not need to go into to much detail about this step. However, understanding the principles behind the backpropagation is very important when considering the choices that need to made when the model is trained.

In a supervised training setting, the output is compared to the known true output and the error is computed given the loss $L(.,.)$ function as:

$$\mathbf{e} = L(\mathbf{y}, \hat{\mathbf{y}}) \tag{4}$$

The core idea is to take the gradient of the loss to see in which direction the error is growing. In a nutshell, backpropagation is simply a gradient decent method. Since the gradient gives the direction of the growing loss, the minimum point can be found by moving to the opposite direction of the gradient. The amount of movement is determined by the learning rate α and its choice is very crucial. Too low α will get stuck in non optimal local minimum or arrive the goal too slowly. Too high value for α will never converge.

The key mathematical idea behind the backpropagation is the simple yet elegant chain rule:

$$\frac{\partial}{\partial \mathbf{x}} f(g(\mathbf{x})) = \frac{\partial f(g)}{\partial g} \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \tag{5}$$

The the gradient of the error with respect to any weight is calculated using this chain rule. For example, if the mean squared error (MSE) is used as the loss function, the gradient with respect to output can be computed using the chain rule as:

$$\nabla_{\hat{\mathbf{y}}} MSE(\mathbf{y}, \hat{\mathbf{y}}) = \nabla_{\hat{\mathbf{y}}} \mathbf{e}^T \mathbf{e} = -2\mathbf{e} \tag{6}$$

By continuing the chain rule, the gradient with respect to the weight matrix in the output layer $\mathbf{W}_{out}$ can be computed as:

$$\begin{aligned} \nabla_{out} MSE(\mathbf{y}, \hat{\mathbf{y}}) &= -2\mathbf{e} \nabla_{out} \hat{\mathbf{y}} \\ &= -2\mathbf{e}\sigma'(\mathbf{h}^{(out)}) \nabla_{out} \mathbf{W}^{(out)} \mathbf{z}^{(out-1)} \\ &= -2\mathbf{e}\sigma'(\mathbf{h}^{(out)}) \mathbf{z}^{(out-1)T} \end{aligned} \tag{7}$$

As mentioned, the gradient points to the direction of the growing error. Therefore, the weights are updated be substituting the gradient of the error multiplied by the learning rate α:

$$\mathbf{W}_{new}^{(out)} = \mathbf{W}^{(out)} + 2\alpha \mathbf{e}\sigma'(\mathbf{h}^{(out)})z^{(out-1)T} \tag{8}$$

The backpropagation is continued to the lower layers of the network and the respective weights are updated until the input layer is reached. This chain rule of the gradient can be applied to all the weights in any neural network. With each iteration the neural networks attempts to moves towards the direction of smaller loss and thereby learns to predict the output.[14][15]

The notation for calculating the gradient of the weight matrices gets very rigorous very quickly. Especially the larger structures become very tedious to express in mathematical formulas. Representing deep learning with pure mathematics is often more difficult than it needs to be, and therefore figures, graphs, tables and other abstractions are extensively used. Although achieving closed form solutions for backpropagation is often very tedious, a clever algorithm can compute the cumulative gradient and update weights without such laborious constructions.

Convolutional Neural Networks

One major problem with a fully connected layer is that it ignores the spatial structure of the data. This makes it a particularly bad approach for example image recognition tasks, because even a small shift in terms of pixels might change the output completely. To address this issue, convolutional neural networks (**CNN**s) were developed. CNNs introduce two concepts to improve on the fully connected network, local connections and weight sharing.

Local connections denotes that output is computed only with respect to neighbouring input elements. Weight sharing, in turn, means that the same weights are used multiple times for different parts of the input. The figure 2 displays this concept with simple 1-D convolutional layer. The output values of the convolutional layer are is computed as follows:

$$h_i = \sum_{k=0}^{K-1} w_{i+k} x_{i+k} \tag{9}$$

where the K is the kernel size which denotes the size of the neighbourhood that is considered when computing the output. A set of weights in the CNN is often referred to as filter. In many real worlds applications, multiple filters are used in one layer, resulting with different output arrays called channels. Filters do not necessarily need to be applied to every element, if there is no need for such accuracy. The stride refers to how many steps the filter is moved between computations. With a stride of two, for example, the neighbourhood is computed for every second element of the input array.

Since CNNs often have layers with multiple channels, the number of intermediate signals can grow very large. This can be countered using pooling layers after convolutional layers. The pooling layer simply reduces the number of computations by averaging or taking the maximum out of local subsets for instance. Convolutional neural networks also often utilize padding, which simply means adding zero valued elements to avoid shrinking the size of the signal through multiple layers.

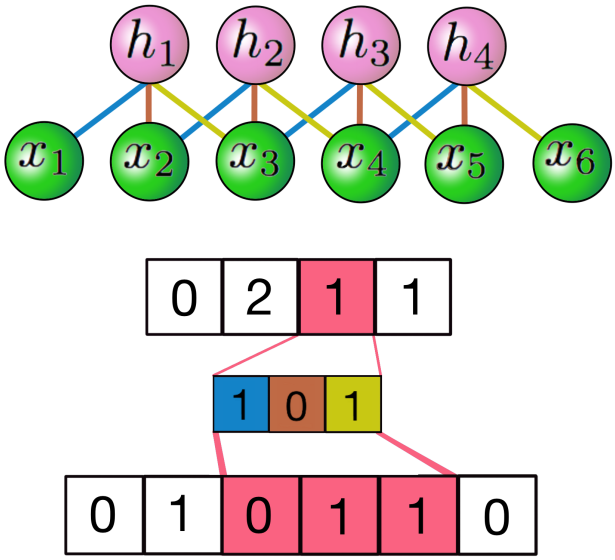


Figure 2: Illustration of a one dimensional convolutional neural network

Local connections and weight sharing make CNNs not only to perform better with spatial data by making the convolutional layers able to recognize patterns regardless of the position in the data. In addition, they also reduce the number of parameters and computational cost in terms of time complexity of the forward and backward computations and memory usage significantly.

When convolutional layers are stacked to construct deeper networks, the latter layers are able to build on abstraction based on the previous layer. The deeper the CNN, the more abstract patterns it is able to recognize. The modern image classification algorithms are often based on this phenomenon. Convolutional layers can be one dimensional, two dimensional or three dimensional, but the basic principle remains the same.

Recurrent Neural Networks

Convolutional neural networks are well suited for image classification and other tasks with fixed sized data, but they run into problems with sequential data with varying length. Such data can be found for example in language processing, which inspired the development of the recurrent neural networks. The structure of the recurrent neural network is illustrated in the figure 3.

The idea behind RNNs is to apply the same computation to each input element regardless of the sequence length. The output of the ith element in the sequence can be computed given the input $\mathbf{x}_i$ and the hidden vector produced by the previous element $\mathbf{h}_{i-1}$ as follows:

$$\mathbf{h}_i = f(\mathbf{x}_i, \mathbf{h}_{i-1}) = \sigma(\mathbf{W}_x \mathbf{x}_i + \mathbf{W}_\mathbf{h} \mathbf{h}_{i-1}) \tag{10}$$

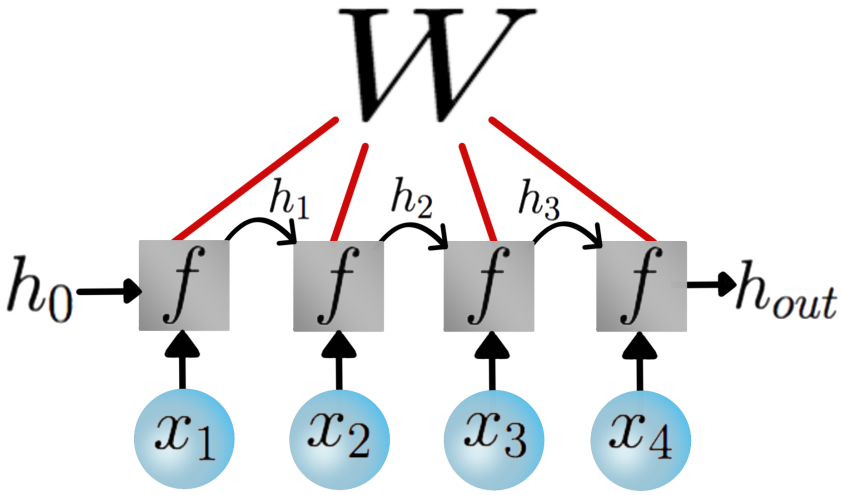


Figure 3: Illustration of a recurrent neural network

where $\mathbf{W}_x$ is the weight matrix corresponding to the input sequence and $\mathbf{W}_h$ for the hidden sequence. The weights are shared between each input. Each element $\mathbf{x}_i$ is commonly a vector obtained by embedding the element into a multidimensional space to achieve better performance. The hidden vector $\mathbf{h}_0$ is initialized often as zero vector and it will be modified during the forward computations. This gives RNNs the ability to gain memory, since the output of the first element will affect on the output of the last element. Unfortunately, regular RNNs are not very good at retaining information for longer strings. Therefore, more sophisticated recurrent neural networks have been developed.

There are several approaches that greatly enhance the performance of RNN based models. Most common approaches are gated recurrent unit (**GRU**) and long short-term memory (**LSTM**). Both of these blocks consist of several inner computational units and can have multiple inner hidden layers. Recurrent units can also be bidirectional meaning that it goes through the sequence in both directions simultaneously.

When the sequence length grows larger, even the GRU and LSTM networks start to lose memory and hence suffer in performance. For this, deep learning mechanism called attention was proposed[16]. These attention based models revolutionized the way RNNs are used today. As an addition information, the attention block gives focus on the most significant parts of the sequence. The learned focus enhances the model's ability to remember the original sequence. Although attention is often used with RNNs, it's not exclusive to only recurrent neural networks and several application exist for CNNs and other models as well.

The recurrent neural networks are usually used in architectures that utilize decoder-encoder structure. In such, the encoder compresses the input sequence into a fixed-size vector which is then transferred for the decoder to interpret. This

vector is the last hidden vector of the recurrent unit. The decoder maps the encoded signal into target space and outputs the target sequence. For example, in machine translation from English to Finnish, the input would be in English and the decoder would make the translation into Finnish based on the output of the Encoder.

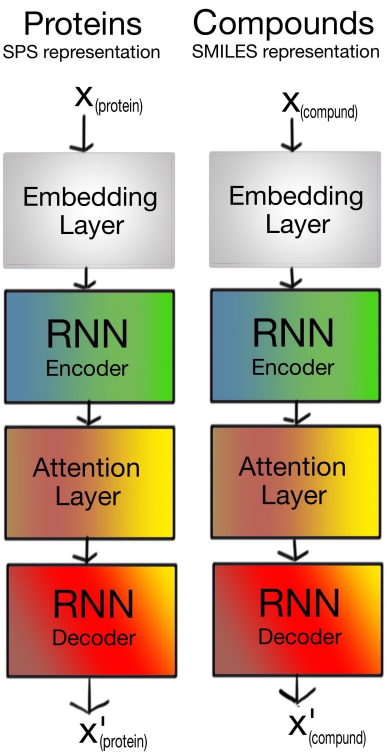
It is often optimal to train the deep learning models using labeled data. However, such data is not always available or it might be scarce. Therefore, sometimes unsupervised methods have to be relied on. Autoencoders are unsupervised learning methods that use encoder-decoder structure. There are several types of autoencoders, but the simplest and the most common one is called bottleneck autoencoder. Bottleneck autoencoders can be classified as data compression methods. However, they have many more applications such as novelty detection and representation learning. The encoder in the bottleneck autoencoder operates by transforming the input into a lower dimensional space. The decoder will use this lower dimensional signal and predict the original input. During the training, the error between the input and the decoder's output is minimized. This way the encoder will learn the important features required for the decoding process often in lower dimensional space. Once the training is complete, the decoder often becomes obsolete.

3.2 Deep Learning for Drug Target Interactions

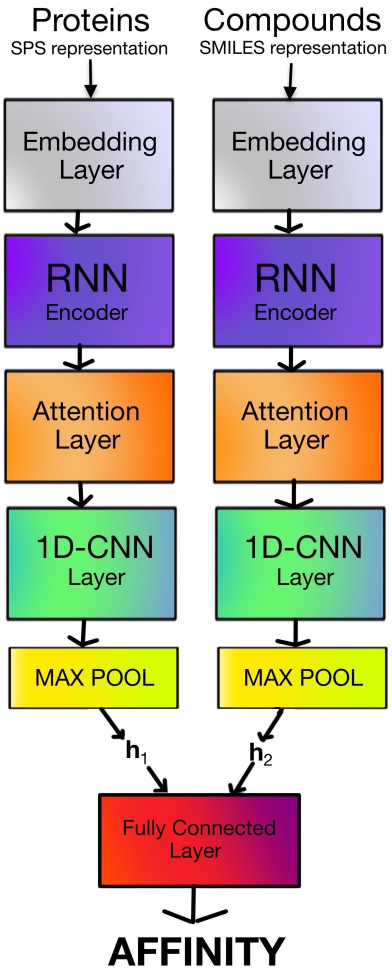
DL-methods are often robust approaches that thrive on large datasets such as the biomedical data. Consequently, the DL-based algorithms are today widely used in the field of bioinformatics. However, deep learning has been criticized for providing black-box approaches that can make accurate predictions without offering any interpretability about the underlying system. This can be an issue when the model is utilized to gain understanding about the studied phenomenon. However, developments such as the autoencoders, have allowed the feature extraction from the raw data, which can be used in representational analysis as well. [REFERENCE NEEDED]

There exists many efficient algorithms for predicting drug target interactions. The problem with most of these methods is that they rely on structural data such as the 3D-structure of the participating agents. Obtaining this data is a doable but demanding procedure, which often dictates the applicability of these methods[17]. It is therefore crucial to develop predictive algorithms that can operate without this hard-to-obtain data. Many such designs do exist, but they are mostly binary classifiers.[REFERENCE NEEDED] In other words, they are able to predict weather a small molecule binds to a target or not, but they do not offer any quantifier for the level of interaction. Unfortunately in many applications, this binary prediction is not enough and more knowledge is required.

The DeepAffinity proposed by Karimi et al.[18] is a DL framework developed to counter the issue of requiring structural data or being limited to only binary outcome prediction. The method is a regression model that employs unified RNN and CNN architecture. The Deep Affinity was modelled based on an algorithm that was originally applied for natural language processing[19]. It takes sequential data as input and It outputs the predicted affinity given the protein sequence and drug



(a) Training Phase *I*
Unsupervised Learning



(b) Training Phase *II*
Supervised Learning

Figure 4: DeepAffinity, the architecture for the two staged training of the model.

compound.

The architecture of the DeepAffinity model is shown in figure 4. It is a bimodal neural network meaning that it has two nested neural networks for the drug and protein data respectively. The word nested here denotes that the architecture contains multiple, and in this case two, inner neural networks. The nested structure consist of unified recurrent and convolutional network with an attention mechanism. The outputs of the two individual NNs of are concatenated to make the final affinity prediction in a fully connected network. The method uses SMILES strings as the drug input and an easy-to-obtain alphabet for the protein sequence. DeepAffinity was able to outperform all the baseline methods within the training and test datasets with large margin. It was even able to generalise to other applications with reasonable performance.

One of the key aspects of the DeepAffinity is that it uses two-stage training. The

first stage is unsupervised and the second is supervised. The method therefore falls under the semi-supervised learning. The first phase of the training is unsupervised, in which the auto-encoders inside the nested NNs are trained using unlabelled data. The encoder-decoder structure utilizes bi-directional gated recurrent unit (GRU) with attention mechanism. The purpose for this is to learn the structure of the inputs, by encoding the sequence and trying to replicate the original input precisely from the decoded information. This representation learning is most likely the reason why DeepAffinity is capable to operate efficiently with low-level sequential data. In the second part of the training, DeepAffinity is trained using labeled bioactivity data to predict the affinity. Before the supervised training the decoders are removed and 1-D convolutional layers with maximum pooling layers are added. The convolutional layers are able to read the signal of the pre trained RNN encoder. In this phase, all the weights in the whole architecture are trained.

The aforementioned IDG-DREAM Drug Kinase Binding Prediction Challenge was a benchmarking study. It was held in order to compare predictive algorithms for drug-kinase affinity data. The data provided in the challenge was sequential and the structural data wasn't included. In the challenge, there were two deep learning models that performed in the top five out of of all participants. These models were named as DMIS and Gregory Koytiger, (referred as Gregory Method in this thesis). The Gregory method CNN based model, whereas the DMIS was a multi-task graph convolutional neural network (**GCN**).

Out of all the particiapting methods in the challenge, the Gregory method was the best performing model in terms of root mean squared error (**RMSE**) and Spearman correlation in the second round. The pipeline for the Gregory method is displayed in figure 5. Similar to DeepAffinity, Gregory method uses multimodal deep neural networks, where the drug compound and the protein have structurally identical neural networks with unique weights. Unlike the DeepAffinity, the Gregory method does not use any unsupervised training but requires additional information as molecular fingerprints and kinase family. The method used the ATP-binding site along with the kinase family to represent the kinase information and the smiles string with the molecular fingerprints to represent the chemical compound.

The Gregory method uses convolutional layer as the first layer in the network, and as such the input sequences must have fixed size. However, the write-up provided by the model designer does not go into detail weather the equal length is achieved by padding, feature selection or using other approach. The Gregory method requires input strings and the kinase family in one-hot-encoded format, which is a very common way to represent categorical data in machine learning. A one-hot vector has one at the position corresponding to the class and zero everywhere else. With the same logic way strings of characters can be transformed in to one-hot-encoded matrices where each row (or column) represent one character.

The drug and kinase sequences are fed to the Gregory model as one-hot matrices. The architecture of the deep learning model uses two nested neural networks containing 2D-convolutional layer with 64 filters followed by a maximum pooling layer and, dropout connection and a fully connected layer. Outputs of the two nested networks are concatenated with the molecular fingerprints and the one-hot-encoded kinase

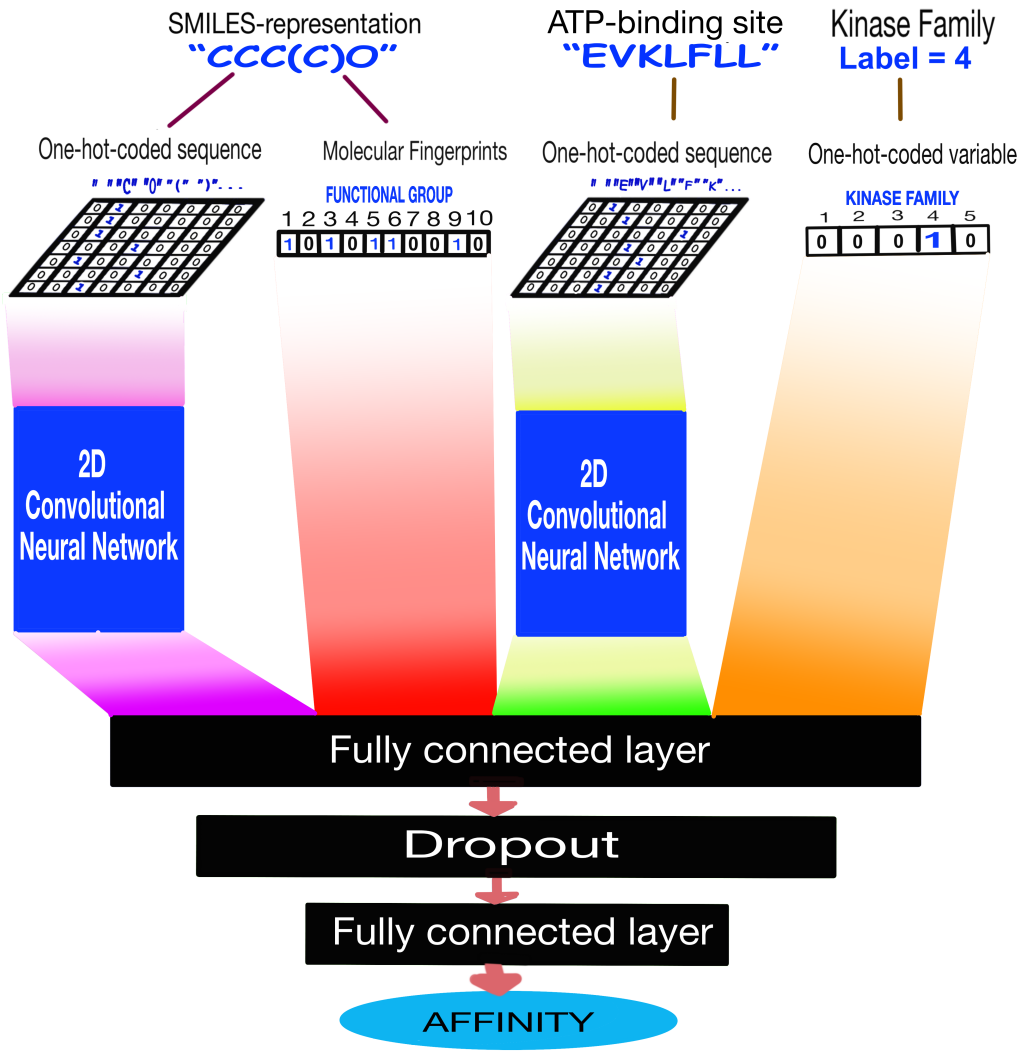


Figure 5: Gregory method, architecture of the neural network

family. The concatenated vector is then fed to a fully connected layer followed by a dropout connection a final fully connected layer. Then final output layer is used to map the prediction to a numerical affinity scalar value.

There exists many other DL algrothims in the field, GCNs to mention just a one. Graph convolutional neural networks are deep learning algorithms that combine the graph representation with convolutional layers. GCNs are utilized effectively in drug-target affinity prediction due to graph structure being efficient way to represent chemical information.[20]. However, The two models, DeepAffinity and Gregory method, reviewed in this chapter will serve as the basis for the novel method represented in this thesis. The concepts introduced by the two, such as multimodal architecture, recurrent convolutional neural networks and using molecular fingerprints and kinase family as additional information, will be adopted to method presented and analysed in the thesis.

4 Methods

4.1 Data

As previously stated, searching for possible druggable kinase inhibitors by exhaustively measuring the interactions between all possible drug-kinase pairs is not feasible. The developments in data and computer science have shifted the drug discovery from wet labs to computer models. Fortunately, there exists many comprehensive and publicly available bioactivity databases that are well suited for training DTI predictors.[21]

The training data used in this thesis was obtained from the drug target commons (**DTC**), which is a crowd-sourced platform for managing bioactivity data.[22] DTC was established to be an standardised interface for tasks related to processing bioactivity data. The data itself can be viewed and downloaded from the webpage or alternatively accessed directly using an **API** (application programming interface). The full dataset holds bioactivity measurements with varying protein types and measuring standards. The data therefore needs to be filtered to meet the needs of training a quantitative drug-kinase affinity predictor.

The DTC dataset contains multiple statistics for each sample. The measured level of interaction between a drug-kinase pair is provided with the used metric among other details about the measurement. The identifiers for the drug molecule and the protein kinase are provided as well. These identifiers are later used as standardised keys for collecting the sequential data from third party databases. The final training set, which was filtered from the full DTC dataset, consists of 152,544 drug-kinase pairs containing 415 unique kinases from 106 different kinase families and 75,107 different small molecules.

The binding affinity is acquired as a scalar value. In addition to the quantitative value for the level of interaction, DTC provides additional information about the measurement as standard type, standard relation and standard units. These features can be used to filter the data. The standard type refers to which equilibrium constant (*e.g.* dissosiation constant K_d or inhibition constant K_i) was used to measure the activity. The standard relation in turn refers to the accuracy on how small concentrations the experiment was carried out. Lastly, the standard units gives out in which SI units the activity is measured.

The identification code obtained from the DTC dataset is used to access the chemical information about the molecular compound by scanning the ChEBML database.[23] ChEMBL is a large high-quality database that contains information about chemical compounds with drug-like properties. The data is gathered from several sources such as such as scientific literature and public databases. The information can be obtained directly from the web page, but there exists a python API for accessing the data directly, which can be extremely useful for larger queries.

The ChEMBL database holds detailed information about the bioactive chemical compounds of interest. In this thesis we are only interested on the SMILES [24] (simplified molecular-input line-entry system) string. SMILES strings are very commonly used data structure in chemical information processing. The system is relatively old but still commonly used. A SMILES-strings represents the molecular

formula of the chemical compound in ASCII-strings incorporating graph theory to capture the chemical structure of the compound.

The information about the kinases is gather from theUniProt database[25]. The UniProt knowledge base is a free, comprehensive and supervised database of sequence annotations and functional information for over 190 million proteins. From the database, the aminoacid sequences for the relevant kinases are exported as a fasta file. In addition, a mapping for different kinase families is obtained. Fasta format is a commonly used text based method for storing and representing nucleotides or aminoacid sequences in using characters. The full amino acid sequence of the kinase is used when training the model.

4.2 Novel Method

This thesis introduces a novel multimodal deep learning model with nested unified recurrent and convolutional neural networks to predict drug-kinase affinity. The architecture of the model is inspired by the DeepAffinity and the Gregory method. Many of the parameters are adopted from these two models, but a parameter search is conducted as well for four model parameters. The Gregory model performed well in the DREAM competition and therefore was a good starting point for possible modifications. The DeepAffinity, on the other hand, has shown that unified RCNs are certainly interesting research target, when studying predictive drug-target models that are well suited for raw sequential data. The recurrent layer in the unified RCN also tackles the Gregory method's issue with the convolutional layer requiring fixed size input. RNNs are designed for sequential data such as the data in hand.

Data Preprocessing

The raw data goes trough a preprocessing module before it can be fed into to a DL-model. The model takes four input parameters which need to be prepared. These are the sequential data of both the drug compund and the kinase, molecular fingerprints of drug compound and the kinase family. These inputs need to be read database and converted into proper format. The pipeline for the preprocessing routine of the training data is represented in the figure 6.

The data is read from the csv file provided by the DTC. The identification codes for the drug compound and the kinase along with the affinity value, standard type, standard relation and standard units are collected and stored. The standard type, relation and units are used to filter the data. Only measurements with K_d, K_i, IC_{50} and EC_{50} as the standard type, nanomolars (nM) as the standard unit and equal standard relation are accepted. Furthermore, entries with invalid values or identifiers are filtered out as well. In case a drug-kinase pair is encountered multiple times, the mean between the duplicate samples is select. Also, the two testsets, which are used later in model evaluation stage, are utilized to make sure that training and testing datasets do not contain any identical observations. Naturally, testing data is not used for any other purpose during preprocessing or training in order to avoid biases.

Next, the identifiers are used to scan the respective data bases for the sequential

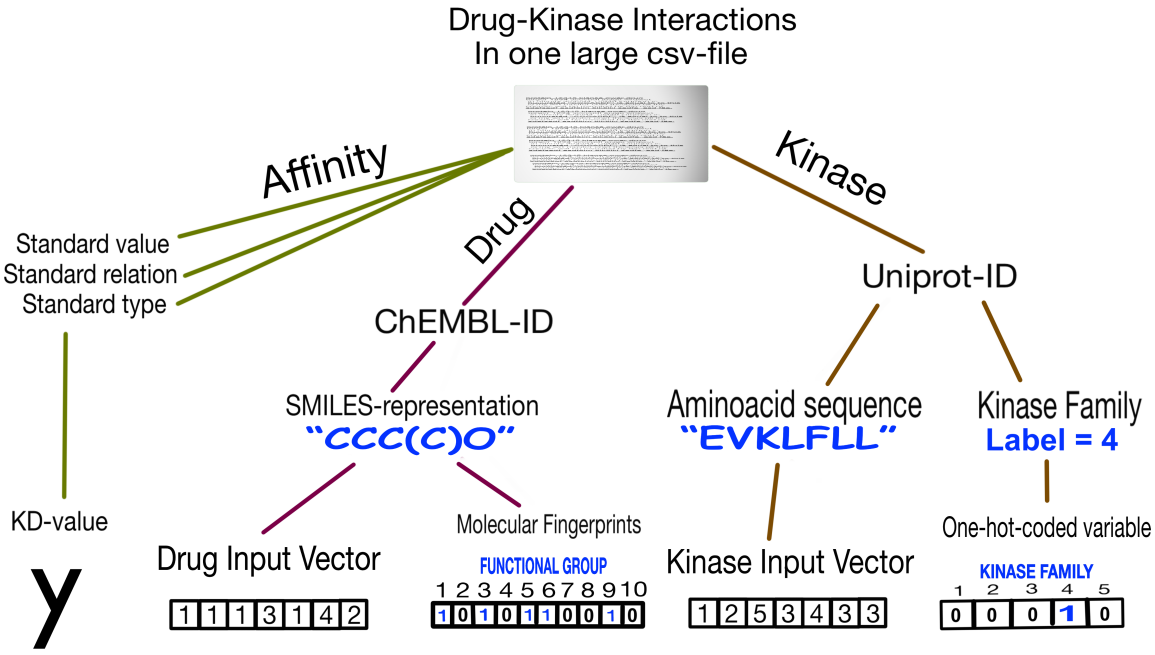


Figure 6: Pipeline for preprocessing the data from the source.

data. The ChEMBL-ID is used to scan the ChEMBL-database for the SMILES-strings, and the UniProt-ID is used to obtain the full amino acid sequence and the kinase family. In case of identifiers failing to retrieve meaningful data for any reason, the dataset is again filtered out for entries with such redundant identifiers. Both drug and the kinase sequence are stored in strings using the standard unicode format. The strings are then encoded into integer vectors, using unique integer for each character. The encoding is done by scanning all the characters encountered in the training data and saving a new integer for each character. The vectors can be used as input in this format for the deep learning model. The mapping is also saved for new data.

As additional information about the drug, the SMILES strings are also used to compute the molecular fingerprint vector. Molecular fingerprint vector is representation commonly used in chemical computing. Each value of the high-dimensional fingerprint vector is a chemical descriptor. The fingerprint representation is an effort to capture the 3D-structure and the chemical properties of a molecule efficiently in vector structure.[26]

The information about the kinase family is represented using one-hot-encoding. Since the kinase family is a classifier, any particular kinase belonging to that family can be expressed with a unique position in the vector. Similar to characters present in the sequential data, family-integer mapping is established by scanning all the kinase families present in the training data. These integers are then used as positions in the one-hot vector.

Lastly, the affinity value is transformed into pK-values, which is the negative logarithm of the affinity value in molars. Since the the original value was provided in $[nM] = [M] \times 10^{-9}$, the transformation into pK values can be computed as:

$$[pK] = 9 - log_{10}([nM]) \tag{11}$$

The different measurements can have molar values with wide range, which can be challenging for the algorithm. The logarithmic scale can be used to tackle this problem. Moreover, the dataset used later in the model testing uses the logarithmic scale, and therefore, it should make sense to use the same scaling.

The preprocessing routine for the testing data, used in the model evaluation, is very similar to the preprocessing module for training data. However, the ChEMBL queries can be bypassed since the SMILES strings are already provided in the test data. The exceptions caused by unseen dataset are handled accordingly. The test dataset is not filtered out of samples with unseen instances, such as SMILES-characters, amino acids or protein families not present in the training data. Instead zero values are used to represent objects not present in the input alphabet learned from the training data.

Architecture of the Model

The architecture of the deep learning model is shown in the figure 7. The top level architecture is similar to the architecture of the Gregory method. Multimodal structure with two nested neural networks is utilized and the molecular fingerprint and the kinase family are used as the additional information. The model takes the preprocessed integer vectors as the input along with additional information and outputs a scalar value for the predicted affinity given the drug-kinase pair.

The vectors containing the SMILES and aminoacid information are fed into the separate nested RCNs. The architecture inside the nested NN, resembles that of the second stage structure of the DeepAffinity. The first layer is an embedding layer, which maps the sequences into multidimensional embedding space. Similar to any other layer in neural networks, the embedding layer has trainable weights that are tuned during the training process. Before the embedding can be applied, the input sequences are padded with a global padding value in order to obtain sequences with equal length. After the embedding, so called packing is applied, which provides the information of when the sequence ends and the padding starts. The embedded sequences are then fed to the recurrent layer, containing a one directional gated recurrent unit with one hidden layer is utilized. The output from the recurrent layer is the last hidden output inside of the GRU.

The recurrent block will transform the sequences with varying length into fixed sized vector. This useful for the following convolutional layer, since the convolutional neural networks can not deal properly with varying sized inputs. The 1D-CNN contains 64 channels with a the kernel size of 4 and the stride of 2. The output signal is followed by a ReLU activation, one dimensional max pooling and a dropout connection with a probability of 0.4. The signals are then concatenated into a vector which is fed into fully connected layer with 256 neurons and no activation. The output from the fully connected layer is the final output of the nested neural network.

The both outputs of the nested RCNs are concatenated with the molecular fingerprints and the one-hot-encoded kinase family into a one long array. This

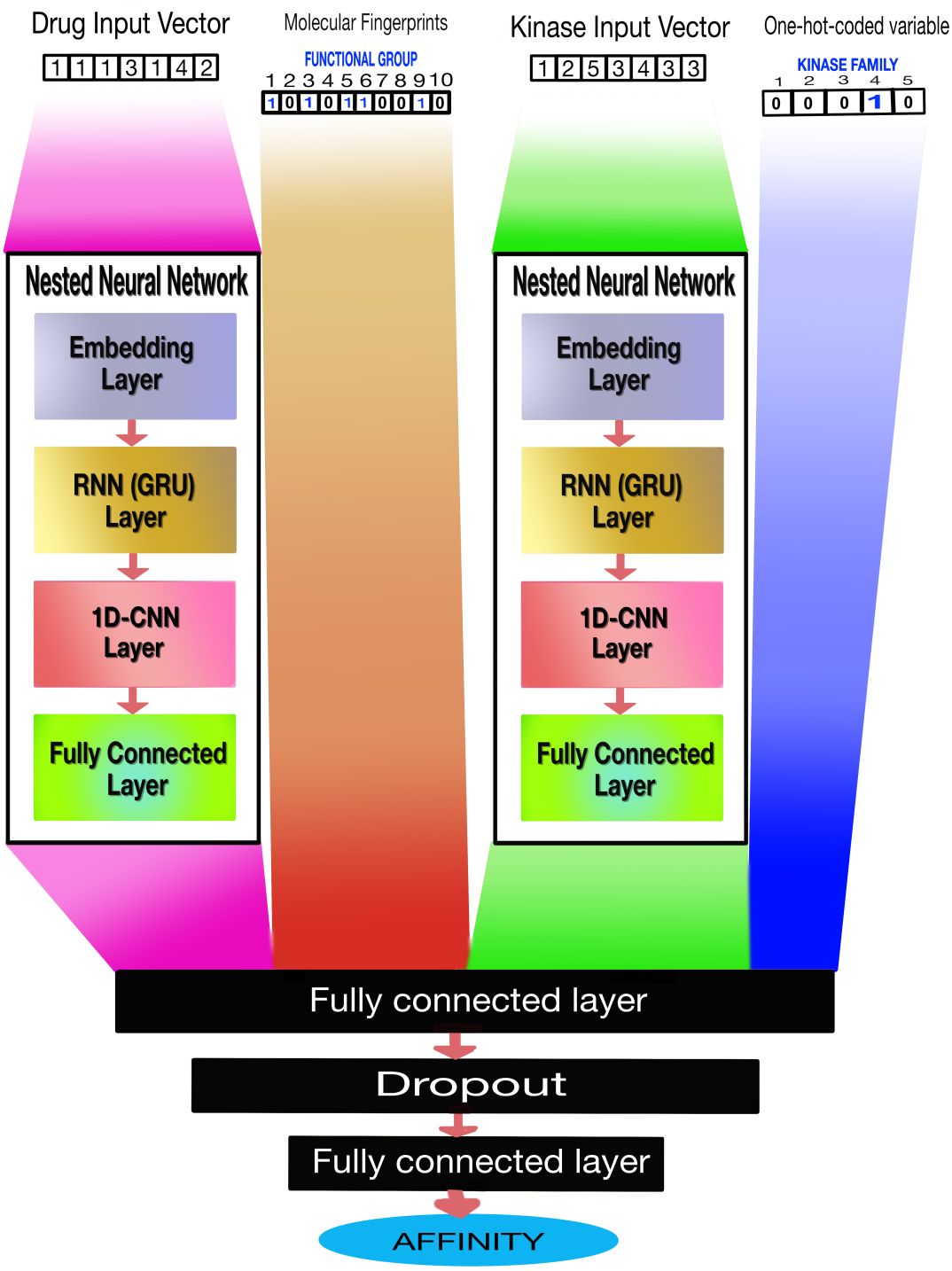


Figure 7: The nested neural networks takes the preprocessed sequence of the drug compound and the kinase. The outputs from the inner neural network are concatenated with the molecular fingerprints and the one-hot coded kinase family. After concatenation, fully connected layer followed by a dropout and another full connected layer. The final prediction is a scalar value for the affinity.

array is then passed through a fully connected layer followed by a dropout with a probability of 0.5. The final fully connected layer maps the the signal into a scalar value which is the predicted affinity value between the compound and the kinase in the pK_d domain.

Model Training

The DTC databese provides true experimental result for the affinity between the given drug molecule and the kinase. The true pK values for the affinity are used for supervised training of this regression model. The loss between the predicted value and the true value is computed using MSE. The justification for MSE is that the DREAM used RMSE in their error metrics. The RMSE points always to the same direction as as the MSE. Furthermore the mean squared error is commonly used loss function in regression problems.

For the optimizer, a stochastic gradient descent (SGD) was chosen. SGD is a variation of the regular gradient descent method. Instead of computing the full gradient of the error, SGD takes a randomly chosen subset of data and provides an estimate of the gradient. SGD does not only speed up the training process by reducing the number of computations but it also can help arriving at smaller error.[CITATION NEEDED] The optimizer also contains a momentum factor which allows previous updates to effect the next update.[CITATION NEEDED] The loss is propagated backwards in the architecture and the weights are updated using the SGD with momentum factor. The optimal values for the learning rate and the momentum factor were estimated during the parameter search.

First the model was trained with for 48 epochs with using the batch size of 50. Then batch size was change to 20 and training was continued for 132 epochs. Finally, the batch size was change one more time to 10 and learning rate was increased and the model was trained for additional 12 epochs. The model training for the whole method took less than two days to complete and the trained model was saved using serializer for later evaluation.

4.3 Hyperparameter Tuning

Unfortunate reality in machine learning is that even the mildly complex models can not learn the optimal solution purely from the data without making any assumptions or prior choices. For example, when performing the gradient decent we have to choose the learning rate in order to arrive at the optimal solution. If the learning rate is poorly chosen, then gradient decent method never converges at the minimum. The method can not learn the right parameter itself and might not work properly without tuning. Therefore, it is always important to justify every decision, such as how the learning rate or the batch size is chosen. Often the hardest part in machine learning is to finetune these parameters, called the hyperparameters, to the optimal values, so that the actual model operates in a space where it can thrive.

It is crucial to select hyperparameters trough an intelligent process. Sometimes, the designer of the algorithm might choose the hyperparameters based on their

domain knowledge. Either trough experience or reasoning. More often than not, such knowledge is not available and the hyperparameters need to be search experimentally. As mentioned earlier, in this thesis many of the model parameters were adopted from DeepAffinity and the Greogory model. Nevertheless, total of six parameters were optimised, and the approach provided can be applied for more parameters as well.

Gaussian Processes based Grid search

Gaussian processes (**GP**s) are ML techniques, that make use of the robust attributes that multivariate gaussian distribution provides. They are stochastic processes which can be applied to a wide range of machine learning problems. The core mathematical operation behind the GPs is the kernel function $k(\mathbf{x}, \mathbf{x}')$, which maps a similarity value between two points $\mathbf{x}$ and $\mathbf{x}'$. The exist wide range of different kernel function. The origin of the kernel function is the covariance, but probably the most commonly used kernel is the gaussian kernel denoted as:

$$k(x, x') = \alpha exp(\frac{||\mathbf{x} - \mathbf{x}'||^2}{l^2}) \tag{12}$$

Here l refers to the scale, which determines how much the neighbouring data points affect to the fit. If the scale is very small only the values very close have any effect on the resulting mean. The kernel can be computed between two arrays or functions g and h resulting in a kernel matrix. This is denoted with K_{gh} where:

$$[K_{gh}]_{(ij)} = k(g(x_i), h(x_j)) \tag{13}$$

Given the data $\mathbf{X}$ with the known values $\mathbf{y}$ The underlying Gaussian estimation for the new datapoints $\mathbf{X}^*$ can be computed using the kernel function and normality assumption. The mean $\mu(\mathbf{X}^*)$ and standard deviation $\Sigma(\mathbf{X}^*)$ for new datapoints are estimated as:

$$\mu(\mathbf{X}^*) = K_{\mathbf{X}^*\mathbf{X}}(K_{\mathbf{XX}} + s^2 I)^{-1}(\mathbf{y} - p(\mathbf{y})) + p(\mathbf{y}) \tag{14}$$

$$\Sigma(\mathbf{X}^*) = K_{\mathbf{X}^*\mathbf{X}^*} = K_{\mathbf{X}^*\mathbf{X}}(K_{\mathbf{XX}} + s^2 I)^{-1} K^T_{\mathbf{X}^*\mathbf{X}} \tag{15}$$

where s^2 is the noise within $\mathbf{y}$ and $p(\mathbf{y})$ is the prior of y. The $diag(\Sigma(\mathbf{X}^*))$ gives the variance $\sigma^2(\mathbf{x}^*)$ for $\forall \mathbf{x}^* \in \mathbf{X}^*$.

This probabilistic approach offered by GPs can be used make predictions about underlying functions given the data. In this thesis we utilize GPs to conduct parameter tuning by estimating the optimal values. Grid search is a brute force method, which goes in small steps trough a chosen interval and chooses the optimal value. This approach always arrives close to global minimum within the range of the step size. However, the problem arises when multiple parameters need to be tuned, which often is the case. For example, if three different parameters with only $n = 10$ different values for each are tried, then model needs to be trained $n^3 = 1000$ times. The computational cost becomes cubic, which is particularly bad when the training of the model even once might take hours or even days.

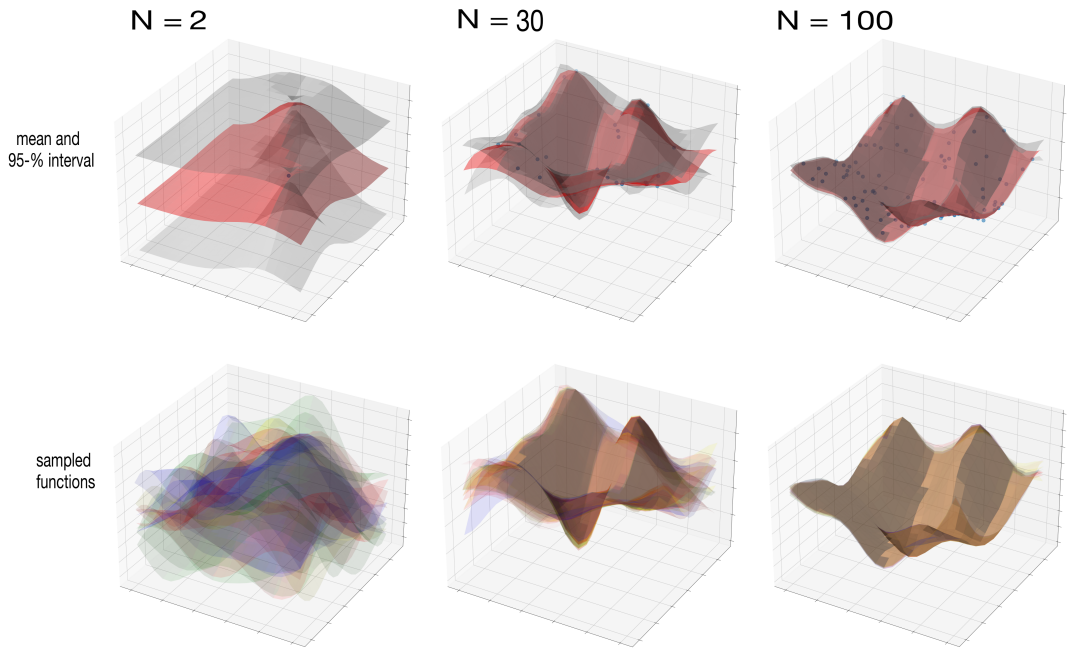


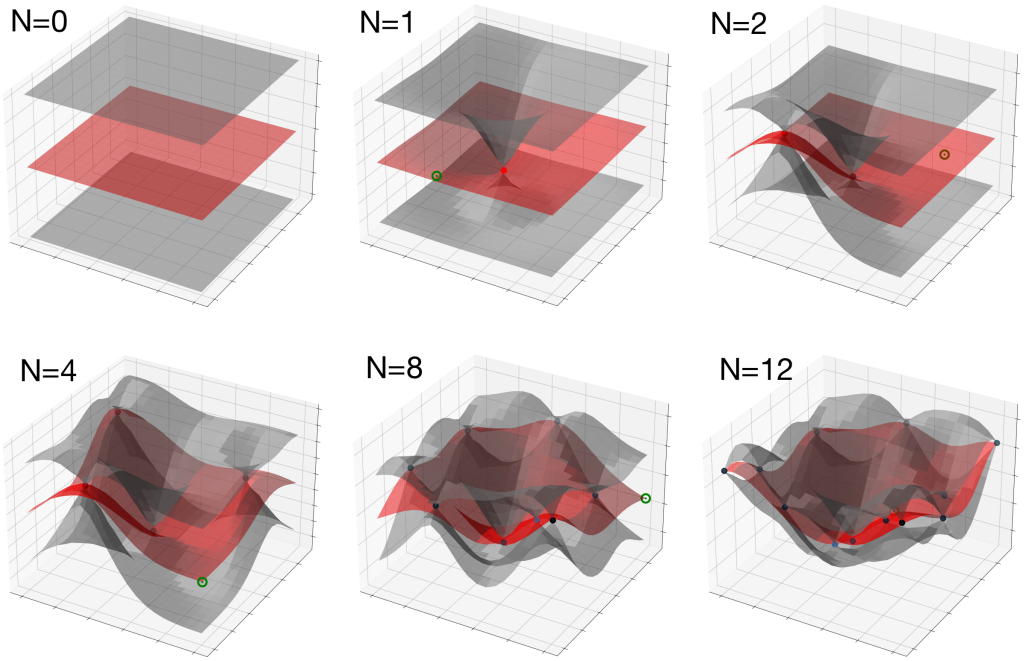
Figure 8: Estimating the underlying function with gaussian processes using. The row above demonstrates how mean and standard deviation are estimated and the row belove shows ten randomly sampled functions from the posterior distribution.

Often, it is not efficient to try out all the possible values. This is also the case when searching for optimal machine learning models by training different variations to see which parameters give the best performance. In this thesis, the underlying error is estimated using gaussian process based grid search (**GPGS**), which uses similar grid search approach as the regular grid search. Instead of searching all the values exhaustively, the probabilistic framework of the Gaussian Processes are used to estimate the optimal value. The loss throughout the grid is predicted given the already tested parameter values. The figure 8 illustrates how to gaussian process learns to estimate the underlying function better and better when more date is added.

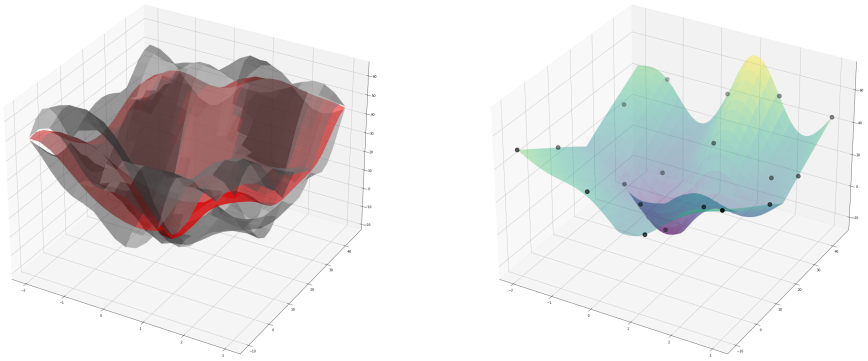
With this approach, it is possible estimate underlying function, but the issue is that it still needs lots of samples. When sampling randomly, the expected value of new information for each sample gets lower and lower the more samples are drawn. The better strategy is two learn from previous samples and try finding the most optimal next data point, in terms of new information towards the global minimum. The bayesian reasoning can be used to estimate the next guess. The most likely lowest point based on previous samples and the prior assumption can be predicted with so called acquasition function. The acquasition function $A(.)$ used in this thesis for new datapoint $\mathbf{x}$ is defined as follows:

$$A(\mathbf{x}) = \mu(\mathbf{x}) - \lambda\sigma(\mathbf{x}) \tag{16}$$

where λ is hyperparameter known as acquasition rate, which determines how much the exploring is favoured. The mean $\mu(.)$ and standard deviation $\sigma(.)$ are calculted



(a) GPGS estimating the underlying function using by the acquasition function to determine next parameters.



(b) Estimated function by the GPGS after 20 searches and the true function

as in equation 14 and 15. The next optimal parameter to be trained is found as:

$$\mathbf{x}_{next} = \arg\min_{\mathbf{x}} A(\mathbf{x}) \tag{17}$$

Finding the minimizing parameters is simply done by computing the kernel for all the untested parameters in the grid and selecting the ones with lowest acquasition value.

The figure 9a illustrates the parameter search for GPGS when next parameter is chosen by minimizing the acquasition function. It is notable that the underlying function starts to resemble the true function with less conducted searches and the

frames of the functions can be seen already after 12 iterations. The true underlying function along with the tested points and the predicted function after 20 searches is shown in the figure 9b.

The GPGS used for parameter search in this thesis, requires normalization of the data. Otherwise the kernel function would be bias towards values with larger range, since that would cause the similarity would be smaller as well. The hyperparameters α,l^2, λ, s^2 are not learned from the data and they need to be tuned manually. The prior distribution need to be chosen as well. The parameters and prior function were tuned during the search using data visualization to help determine suitable values. The final selection after parameter search is the posterior minimum of the kernel mean.

5 Results

Results for Parameter Search

The parameter optimization was conducted for four different model parameters and two training parameters using the previously introduced GPGS method. The optimized model parameters were the embedding dimension for the kinase and the drug sequence and the hidden size in the the gated recurrent unit for both nested neural networks. The optimized training parameters were the learning rate and the momentum factor of the SGD optimizer. Rest of the parameters were adopted from either Gregory or DeepAffinity models. The underlying function for error predicted by the GPGS method along with the individual results for each training run is displayed in the figure 10. The standard deviation for each parameter search is also provided on the right side of the figure. The error is computed as the mean square error using the validation data set, which is an isolated dataset from training data that the model has not seen during the training. The posterior minima are shown with green star marker. During each of the three parameter search phases, the hyperparameters in the GPGS were tuned manually using the visualization of the error. The obtained parameters were used for the final training of the neural network.

The parameter search was conducted for two parameters at the same time in three different searches. First, the optimal embedding dimension for both drug and kinase was traced. The grid was set to be from 50 to 400. After training 33 different parameter combinations, the GPGS predicted the posterior minimum to have embedding size of 262 for the drug sequence and 400 for the kinase sequence. Next, using the obtained embedding sizes, the grid search for the optimal hidden sizes of the recurrent units was carried out. The grid was initialized between 50 and 300 neurons. After 35 runs, the predicted posterior minimum was found to be 50 for the drug compound and 182 for the kinase. In each run, the model was trained four full epochs of the data and lowest validation error was selected. The figure shows that embedding size and the hidden size had very marginal effect to the error.

Finally, the search for learning rate and momentum factor was conducted using the previously obtained embedding and hidden sizes. The learning rates displayed in the figure are negative logarithms of the true learning rate. Since required training time depends highly on the training parameters, the number of epoch was raised up to ten. The parameters that resulted with error above three or led to diverging, were set to be equal to three in order to avoid too large differences that GPGS module could not interpret correctly. The training parameters had substantial effect on error. The posterior minimum after searching for 58 observation was found in 3.07 for the learning rate and 0.57 for the momentum factor.

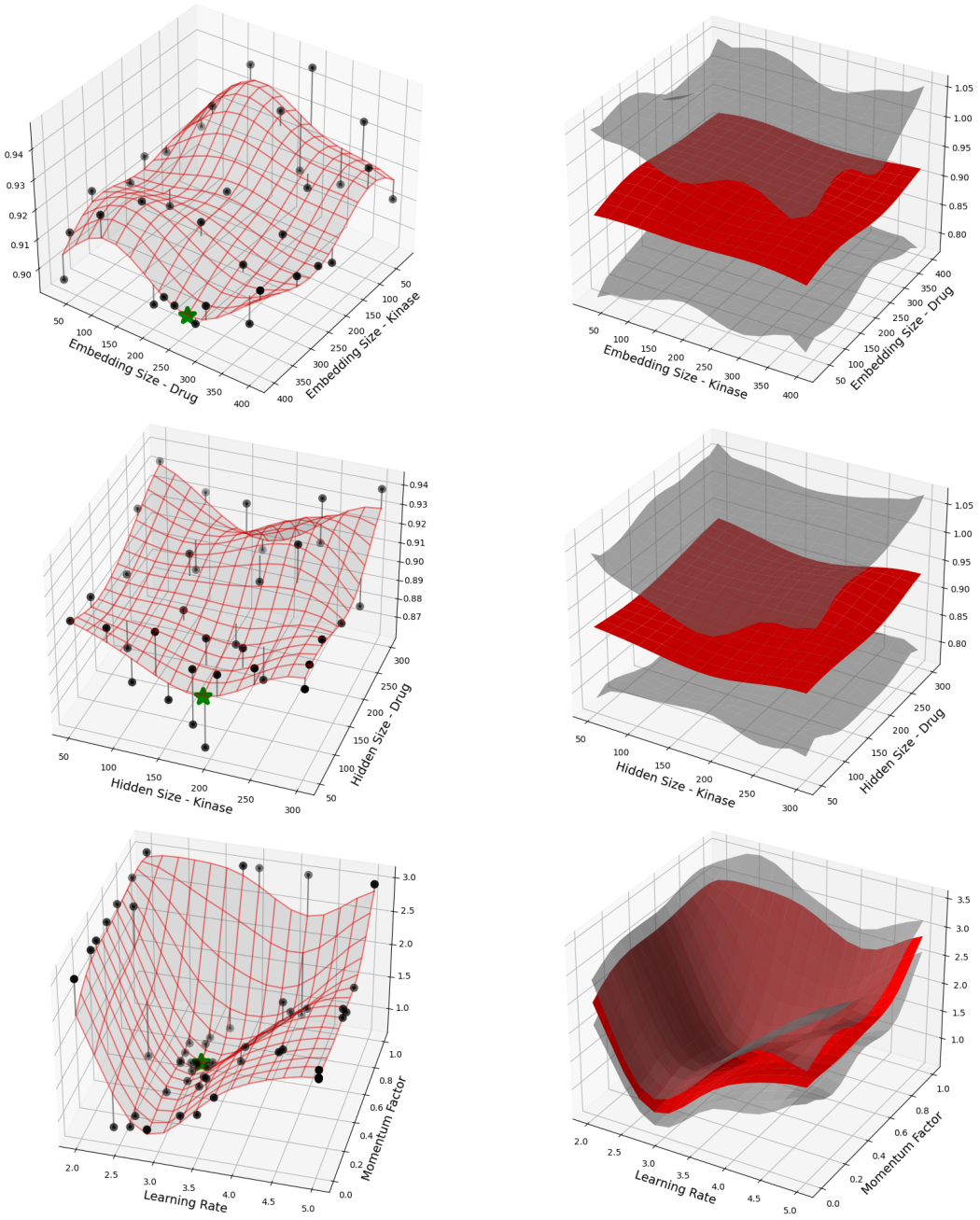


Figure 10: Distribution of different kinase families present in the datasets

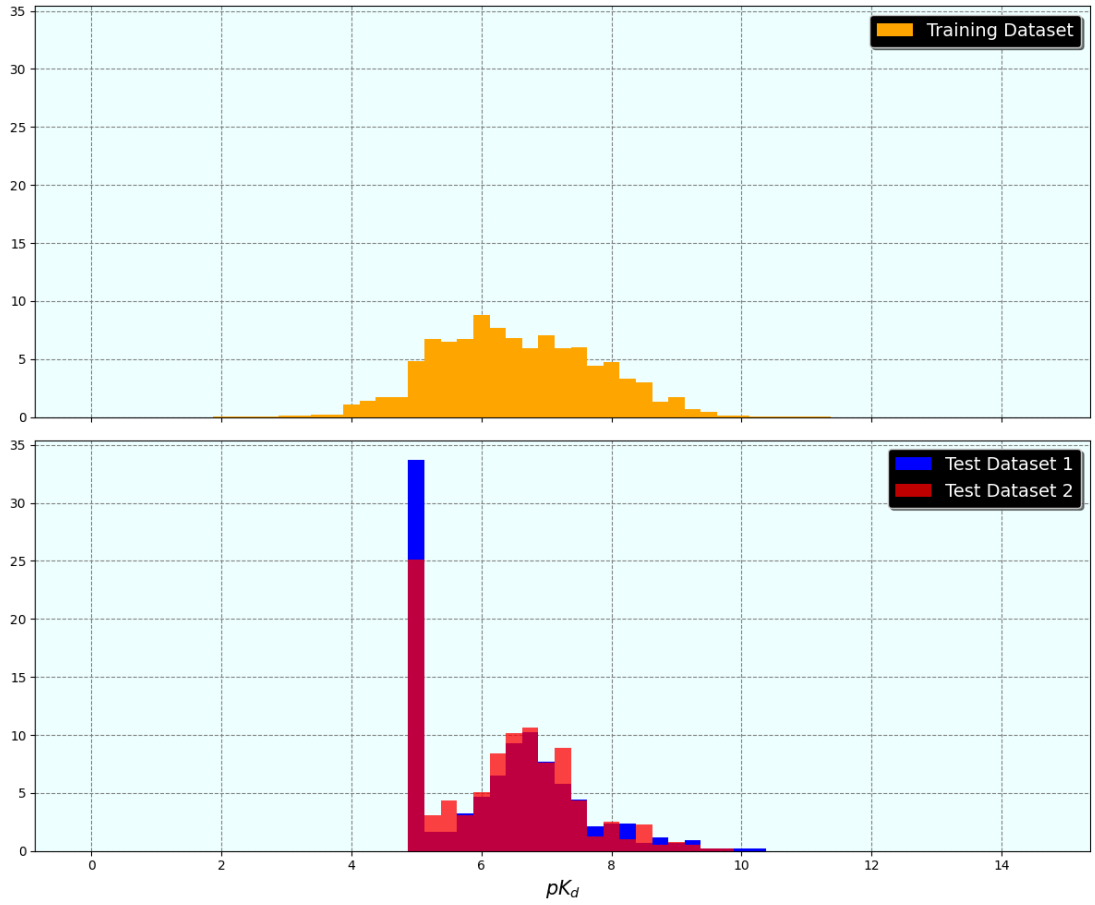


Figure 11: Distribution of affinity values in the datasets

Input Domain Analysis

After the training, the model was tested with two separate testing datasets provided in the DREAM challenge paper. The distribution of the affinity values in both test datasets and the training dataset shown in figure 11. The distribution of the training data differs from the test data notably. In the lower end of the pK_d spectrum, both test datasets have five as the lowest possible pK_d value for the affinity. The lowest value is clearly overrepresented in the data with over one third of the first and one fourth of the second testing dataset having pK_d value equal to five. This is due to interpreting all the values below five having no bioactivity. This difference to the training dataset is countered by setting minimum value of five in the training dataset as well.

The distribution differs in other areas as well. The training data is more evenly distributed throughout the spectrum where as the test datasets are more concentrated between six and seven. The training dataset has much more density in the higher pK_d values and some rare observation above twelve which are not present in the testdata which only has observations up to $pK_d = 10.5$. The higher bioactivity values

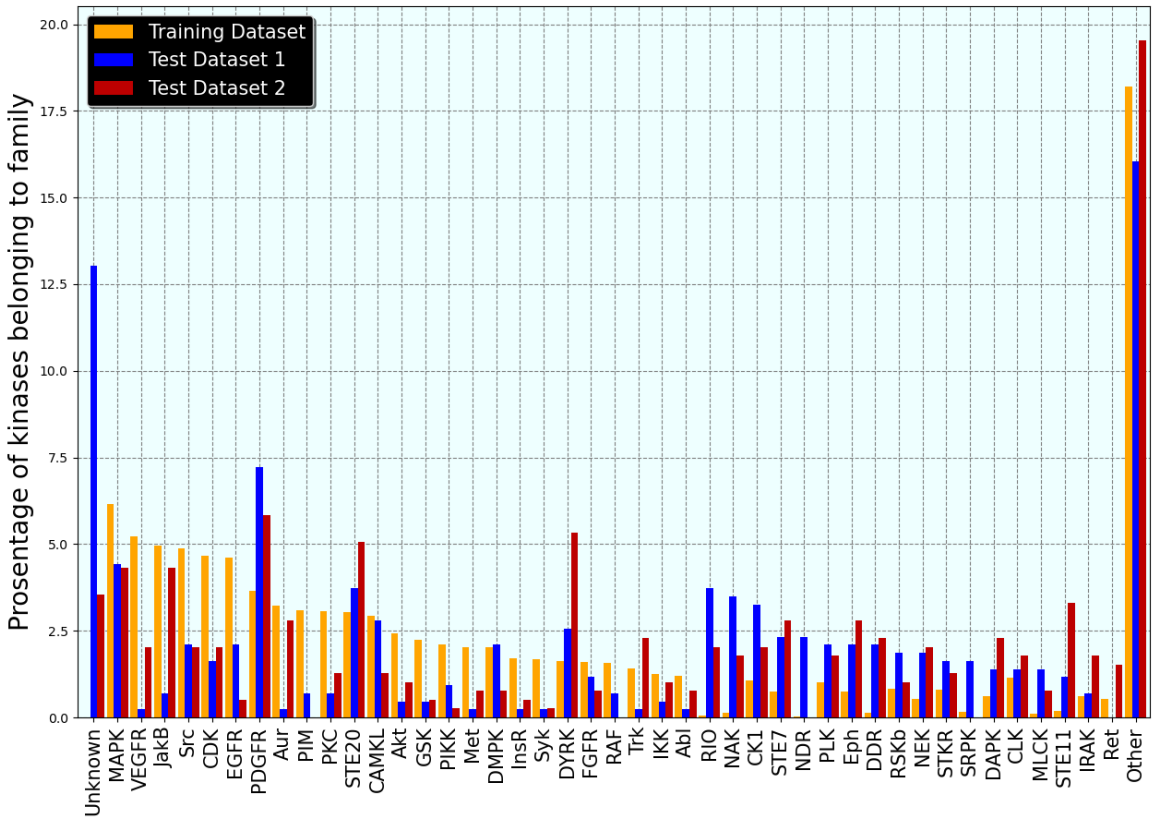


Figure 12: Distribution of different kinase families present in the datasets

in the training data are not truncated unlike the lower values.

The dataset also differs in terms of kinase representation. The figure 12 shows the most common kinase families in the datasets. In the training dataset all the kinase families are known, where as testsets also contain kinases with unknown families this is due to both test datasets having an emphasis on understudied protein kinases called dark kinases [27]. Especially the test dataset 1 has a significant portion of the samples from unknown kinase families with over one eight of kinases not belonging to known family.

Model Performance

The model performance is tested with data provided in the DREAM-challenge. The data contained (at the time) unpublished bioactivity data about understudied human kinases, called dark kinases. The differences to training dataset is that SMILES strings are provided in the test data so access to ChEMBL database is not required. Also, the affinity is provided as pK_d-value which is the negative logarithm of the dissociation constant.

The test data is divided in two individual datasets, which were originally introduced in different rounds of the competition. The first dataset consists of 430 drug-kinase interactions having total of 70 individual small molecules and 199 kinases.

	Validation Dataset	Test Dataset 1	Test Dataset 2
RMSE	0.910	1.231	1.156
Spearman Correlation	0.637	0.222	0.253
Pearson Correlation	0.643	0.218	0.231

Table 1: Model performance within different datasets

The second dataset contains 394 observations between 25 drug compounds and 207 kinases respectively. RMSE and spearman correlations are computed for the both datasets. A validation set, which contains samples from the training set but unseen by the algorithm is computed as well. The validation set is used in the hyperparameter search in order to avoid using test datasets in model selection.

The correlation within the validation dataset and the two testsets is shown in the figure 12. The trendline of the plot is done with regular linear regression. As can be seen, the correlation within the validation set is significant but less notable in the testing datasets. The actual numerical result for RMSE, spearman correlation and pearson correlation are provided in the table 1. The result show that method had very strong predictive capability within the validation dataset but some of the performance was lost when generalising to the testing datasets. The model comparison plot in the figure 13 shows model performance comparison of the participating methods in the DREAM challenge in terms of RMSE and spearman correlation. The plot only shows the methods with RMSE above two and our model outperformed many participating methods in the challenge. However, it can not compete with the best performing models. The selected baseline model was an experimentally validated kernel based regression approach [28], which was trained with data collected from the DTC database. In comparison to the baseline model, our model had similar performance in terms of the RMSE but the baseline model had clearly higher spearman correlation.

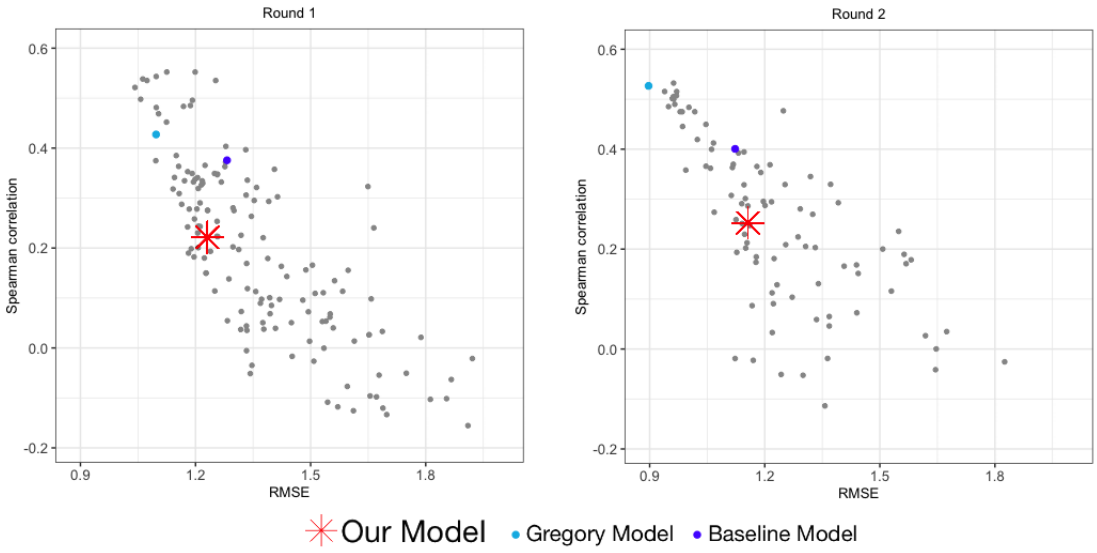


Figure 13: Comparison between the models in the DREAM challenge paper

6 Discussion

6.1 Potential Improvements

The model introduced in this thesis showed some potential for further development. It was on par to average contestant in the DREAM challenge, but it wasn't able to compete with the top performing models. Moreover, the baseline model outperformed the novel method in terms of Spearman correlation. However, there are lot of things that could be tried to enhance the model performance to make it more competitive. To asses the full capability of the model, more variations of the model and data need to be tested in order to arrive at the optimal model architecture.

Improving the training data

When training a machine learning model, the training data is crucial factor in determining the performance. The Gregory method performed very well and one factor for its success might be that it used the ATP-binding site along with the kinase family instead of the full amino acid composition. The regulatory binding site is the area where the most KIs bind and inhibit the activity, therefore it can be sufficient information for predicting the bioactivity.

In the data preprocessing phase, all the affinity standards K_d, K_i, IC_{50}, EC_{50} were used as equal metrics. All though the values are closely related, some data harmonization could improve the model performance. A single standard could have also been used, like was done in the test data, however this would have risk the dataset been too small. The data also consisted only [418] kinases out [518] know human kinases. By extending the dataset, the model could have learned to generalize for all the kinases better. Also, we didn't check whether some drugs or kinases were overrepresented in the data and therefore causing bias.

The preprocessing algorithm for test dataset is scripted to treat unseen data objects such as unseen protein families, amino acids or SMILES-characters using zeros in one hot-coded vectors and matrices in such exceptions. Whether or not this is a good practise is open for debate.

Improving the Deep Learning model

The architecture of the deep learning model is most robust part in terms of directions of improvement. Other approaches, such as GNNs and GCN could be experimented with, but within the recurrent convolutional assumption, there are many ways to develop the model further.

One key component and target for modification is the recurrent block in the nested neural network. The DeepAffinity used attention mechanism with the recurrent unit which would most likely enhance the results. Also, other mechanism that could be adopted from DeepAffinity is the two staged training with autoencoders. [DeepAffinity-Paper: "to exploit both unlabeled and labeled data, for jointly encoding molecular representations and predicting affinities."] The recurrent unit would better

learn the representation of the data, which would be more informative for the convolutional layer to read.

Furthermore, the embedding done in the data preprocessing could be done more intelligently which would lead to more sufficient input data for the recurrent block. Also, LSTM etc. could be tested.

The convolutional layer could also be modified. [MORE CONVOLUTIONAL LAYERS] [There are not that many parameters thought that could be tuned. Mostly, kernel size and number of channels.]

Fingerprints - the method was chosen quite arbitrarily and more though could be put in to it.

The model selections requires lost of experimenting with different options.

The graph representation could also be adopted instead of strings. Graph neural networks are popular approach to process chemical data. Graphs could be experimented either with both kinase and drug data or with the small molecules only.

Improving the optimization

The GPGS is an efficient way to counter the brute force scanning of the parameter space. However, even with the GPGS, testing training several models with a large dataset takes very long. Testing enough hyperparameters for the GPGS to converge a solution takes more or less 45 minutes to train one single model for four epochs of the full data with a computer equipped with a graphical processing unit (**GPU**).

Even though hyperparameter search with only three paremeters takes significant amoun tof time, the search could be tested with even more parameters. It is unclear how much adding new parameters would increase the time required for converging to a result. One bottle neck is computing the kernel in the GPGS. It requires a matrix multiplication for very large matrix which size will grow exponentially when more dimension are added. One problem is that there exists no good library yet that computes the kernel efficiently and instead python code is used which is very slow. Simply, writing the code in C and potentially utilising GPU efficiently could speed up computation tremendously, allowing the kernel computation for larger matrices.

The GPGS also estimates the underlying function by estimating the value in each point defined in the grid search. There are likely ways to carry out this more intelligently without requiring so many individual estimations. One such approach could be using gradient descent to find the lowest point in the acquasition function, since that is ultimately the only point of interest when deciding the next set of parameters in the search.

The gaussian process based method that was used was very vanilla, but it managed to serve its purpose. However, the gaussian process comes with multiple hyperparameters that are not learned from the data. Choosing these parameters properly is very difficult, but still manageable by exploiting data visiualisation as done in this thesis. This is very time consuming and can be done only with dimensions that can can be visualized (1-3), in order to adjust. More sophisticated gaussian process based methods usually do the hyperparameter tuning automatically and which makes them capable to optimize more than three parameters at once.

6.2 Benefits of the model

Gregory method uses 2D-convolutional layers to read the input data, which in turn, requires input strings to have a constant length. Preparing such data certainly requires some feature selection and can be a daunting task. The greatest benefit that RCN offers in comparison to CNN is the capability to translate varying length sequence data to constant length vector that is better suited for CNN to process. Therefore, novel method does not require any feature selection but instead is capable to process raw input sequences. This makes it quick approach for early stage drug-kinase interaction testing.

During early model developments, the performance of the novel model was tested against replica of the Gregory method, without any feature selection. The replicated Gregory model was trained with different data same DTC dataset as the novel method. The novel method outperformed the replicated Gregory method in all the validations with large margin. Since no feature selection was apllied, the input matrices were set have length equal to maximum length of the sequence encountered in dataset. These might be the reasons why replicated Gregory model couldn't perform as well as in the DREAM challenge. This shows that RCN is likely a better choice than plain CNN for data in hand if feature selection is not performed.

The novel model is very light in terms of training since the unsupervised learning phase is not performed. Where the DeepAffinity took 8 days of training, our method can be trained in one to two days. This is huge benefit when applying the model for new dataset, since the novel method can be trained for four dataset in the same time DeepAffinity is trained for one.

6.3 Further Study

RNNs are good to handle sequential data with varying length where as CNNs are good are recognising patterns in spatial data. The combination of RNN CNN works well with bioactivity data since RNN can transform the varying length data vectors into a one dimensional vector that CNNs can work with and recognize patterns from.

The Gregory method performed surprisingly well using only one convolutional layer in the networks. The weakness of this approach is that data size of the data matrices need to be constant. Also, the input matrix contains lots of zeros and single one for each row. This is not particularly kind of data that CNNs are regularly know to thrive in.

Despite the aforementioned [issues], the performance of the Gregory method indicates that some aspects of the architecture must be very suitable for the drug-kinase affinity prediction. To counter these issues recurrent neural networks can work as the transformer between the data input and the convolutional layer. They are more robust and can perform with varying length inputs.

DeepAffinity has shown that unified convoltional neural networks are [GOOD]. The utilization of unified RCNs can definetly hold potential for for further study. This simply requires lots of testing with different and possibly deeper models.

6.4 Conclusions

It is apparent that developing an effective method to predict the drug-kinase interaction requires lots of the experimenting in the model selection. There are countless ways that a model can be further modified. Although that may seem disconcerting, its always good thing to have options for further improvements.

Taking into account of the mentioned potential targets for improvement and the countless other ways that model could be enhanced, the performance of the model is somewhat promising. It was able to perform well within the validation set, but some issues emerged when trying to generalize to different dataset.

Although, the model presented in this thesis requires lots of work, further study to recurrent convolutional neural networks in the drug-kinase prediction certainly seems beficial. The certain benefit with using the recurrent neural networks is that they are able to perform well with the full amino acid sequence of the protein, which often is easily available. The gregory method required fixed length sequences for both amino acids and SMILES sequences in the convolution layer. The RCN counters this problem by compressing the varying length strings into fixed length in GRU before the convolutional layer. This makes RCN approach more robust to only using the CNN with feature selections.

Furthermore, the results with DeepAffinity and the performance of the Gregory method in the DREAM competition show that bimodal neural networks for string processing is a good approach.

7 Summary

Protein Kinases play cruicial role in cell signalning. Kinase overactivity or other form of deregulation can lead development of tumour. Small molecule kinase inhibitors are known to be effective therapeutics in cancer treatment. Due to conserved regulatory site, it is difficult to find selective drugs. Many studied bioactive inhibitory drug compounds target multiple kinases.

Due to this nonselective nature of kinase inhibitors, it is important to know the effects in off-targets as well. To avoid laborious biochemical experiments, the drug-target space can be scanned with computer models. Multiple predictive algorithms have been developped based on different machine learning paradigms.

This thesis introduced a predictive deep learning model which performance was evaluated using the same metrics and test data as in the IDG-DREAM drug-kinase prediction challenge. The model performed well with the evaluation dataset isolated from the training data. However, it had some issues when trying to generalize to the test datasets.

Nevertheless, taking account the many potential ways that model performance can be enhanced by simple modifications, there could be lots of potential within such recurrent convolutional neural network models within drug-kinase interaction prediction.

References

[1] A. Cichońska, B. Ravikumar, R.J. Allaway, et al. Crowdsourced mapping of unexplored target space of kinase inhibitors. *Nature Communications*, 12, 2021.

[2] F. H. Westheimer. Why nature chose phosphates. *Science*, 235(4793):1173–1178, 1987.

[3] Kyle W Knouse, Dillon T Flood, Julien C Vantourout, Michael A Schmidt, Ivar M Mcdonald, Martin D Eastgate, and Phil S Baran. Nature chose phosphates and chemists should too: how emerging p (v) methods can augment existing strategies. *ACS Central Science*, 7(9):1473–1485, 2021.

[4] Fatima Ardito, Michele Giuliani, Donatella Perrone, Giuseppe Troiano, and Lorenzo Lo Muzio. The crucial role of protein phosphorylation in cell signaling and its use as targeted therapy. *International Journal of Molecular Medicine*, 40:271–280, 2017.

[5] Gerard Manning, David B Whyte, Ricardo Martinez, Tony Hunter, and Sucha Sudarsanam. The protein kinase complement of the human genome. *Science*, 298(5600):1912–1934, 2002.

[6] Tony Hunter. The proteins of oncogenes. *Scientific American*, 251(2):70–79, 1984.

[7] G Maurer, Bartek Tarkowski, and Manuela Baccarini. Raf kinases in cancer–roles and therapeutic opportunities. *Oncogene*, 30(32):3477–3488, 2011.

[8] M.M. Attwood, D. Fabbro, A.V. Sokolov, et al. Trends in kinase drug discovery: targets, indications and inhibitor design. *Nature Reviews Drug Discovery*, 20:839–861, 2021.

[9] Andrew L. Hopkins and Colin R. Groom. The druggable genome. *Nature Reviews Drug Discovery*, 1:727–730, 2002.

[10] Giancarlo Franchini, Andrea Marchetti, Lorenzo Tassi, and Giuseppe Tosi. Ionization and dissociation of weak electrolytes. an initial approach to ki and kd evaluation. *Analytical Chemistry*, 62(10):1004–1010, 1990.

[11] E. MarÉchal. Measuring bioactivity: Ki, ic50 and ec50. Springer, Berlin, Heidelberg, 2011.

[12] Susan Klaeger, Stephanie Heinzlmeir, Mathias Wilhelm, Harald Polzer, Binje Vick, Paul-Albert Koenig, Maria Reinecke, Benjamin Ruprecht, Svenja Petzoldt, Chen Meng, et al. The target landscape of clinical kinase drugs. *Science*, 358(6367):eaan4368, 2017.

[13] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[17] Oleg Trott and Arthur J Olson. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31:455–461, 2010.

[18] Mostafa Karimi, Di Wu, Zhangyang Wang, and Yang Shen. Deepaffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. *Bioinformatics*, 35(18):3329–3338, 2019.

[19] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1700–1709, 2013.

[20] Wen Torng and Russ B Altman. Graph convolutional neural networks for predicting drug-target interactions. *Journal of chemical information and modeling*, 59(10):4131–4149, 2019.

[21] Lina Humbeck and Oliver Koch. What can we learn from bioactivity data? chemoinformatics tools and applications in chemical biology research. *ACS Chemical Biology*, 12(1):23–35, 2017. PMID: 27779378.

[22] Jing Tang et al. Drug target commons: A community effort to build a consensus knowledge base for drug-target interactions. *Cell Chemical Biology*, 20:224–229, 2018.

[23] Anna Gaulton, Anne Hersey, Michał Nowotka, A Patricia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J Bellis, Elena Cibrián-Uhalte, et al. The chembl database in 2017. *Nucleic acids research*, 45(D1):D945–D954, 2017.

[24] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

[25] The UniProt Consortium. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, 11 2020.

[26] Yu-Chen Lo, Stefano E Rensi, Wen Torng, and Russ B Altman. Machine learning in chemoinformatics and drug discovery. *Drug discovery today*, 23(8):1538–1546, 2018.

[27] Nguyen et al. Pharos: Collating protein information to shed light on the druggable genome. *Nucleic Acids Research*, 45(D1):D995–D1002, 11 2016.

[28] Anna Cichonska, Balaguru Ravikumar, Elina Parri, Sanna Timonen, Tapio Pahikkala, Antti Airola, Krister Wennerberg, Juho Rousu, and Tero Aittokallio. Computational-experimental approach to drug-target interaction mapping: a case study on kinase inhibitors. *PLoS computational biology*, 13(8):e1005678, 2017.