ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ & ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

ΕΡΓΑΣΤΗΡΙΟ ΒΙΟΙΑΤΡΙΚΗΣ ΤΕΧΝΟΛΟΓΙΑΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Deep learning for signaling network embeddings and inferring them from a compound's chemical structure

Ιωάννης Γεώργιος Ζαζάς

Επιβλέπων: Λεωνίδας Αλεξόπουλος, Καθηγητής ΕΜΠ

Αθήνα, Ιούνιος 2022

Abstract

During the last years, big data and deep learning have become a very effective and state of the art method to deal with many demanding and difficult problems, in multiple scientific fields, from traditional computer science to finance and biology. Up to the present, deep learning was rarely used as a method by bioinformaticians, who prefer gene expression data and try to understand the mechanism of action of drugs to promote research on topics like early drug discovery. To deal with signaling networks, it is more common to use network analysis and dynamical systems modelling. For this thesis, we develop a specific class of graph convolutional neural network, using a very effective architecture that achieves maximum discriminative power among other GNNs' and apply this model to a dataset of biological (protein) signaling networks. We prove that our model can effectively cluster compounds with similar mechanisms of action together and identify compounds with similar signaling networks. Finally, we use the data produced from this model and try to train a model to infer a protein signaling network from a compound's chemical structure.

Περίληψη

Τα τελευταία χρόνια, τα μεγάλα δεδομένα και η βαθιά μάθηση έχουν γίνει μια πολύ αποτελεσματική και τελευταίας τεχνολογίας μέθοδος για την αντιμετώπιση πολλών απαιτητικών και δύσκολων προβλημάτων, σε πολλαπλά επιστημονικά πεδία, από την παραδοσιακή επιστήμη των υπολογιστών μέχρι τη χρηματοδότηση και τη βιολογία. Μέχρι σήμερα, η βαθιά μάθηση χρησιμοποιήθηκε σπάνια ως μέθοδος από βιοπληροφορικούς, οι οποίοι προτιμούν τα δεδομένα γονιδιακής έκφρασης και προσπαθούν να κατανοήσουν τον μηχανισμό δράσης των φαρμάκων για την προώθηση της έρευνας σε θέματα όπως η πρώιμη ανακάλυψη φαρμάκων. Για την αντιμετώπιση των δικτύων σηματοδότησης, είναι πιο συνηθισμένο να χρησιμοποιείται η ανάλυση δικτύου και η μοντελοποίηση δυναμικών συστημάτων. Για αυτή τη διπλωματική εργασία, αναπτύσσουμε μια συγκεκριμένη κατηγορία συνελικτικού νευρωνικού δικτύου γραφημάτων, χρησιμοποιώντας μια πολύ αποτελεσματική αρχιτεκτονική που επιτυγχάνει μέγιστη ισχύ διαχωρισμού μεταξύ άλλων τέτοιων δικτύων και εφαρμόζουμε αυτό το μοντέλο σε ένα σύνολο δεδομένων βιολογικών (πρωτεϊνικών) δικτύων σηματοδότησης. Αποδεικνύουμε ότι το μοντέλο μας μπορεί να ομαδοποιήσει αποτελεσματικά ενώσεις με παρόμοιους μηχανισμούς δράσης και να αναγνωρίσει ενώσεις με παρόμοια δίκτυα σηματοδότησης. Τέλος, χρησιμοποιούμε τα δεδομένα που παράγονται από αυτό το μοντέλο και προσπαθούμε να εκπαιδεύσουμε ένα μοντέλο ώστε να συμπεράνει ένα δίκτυο σηματοδότησης πρωτεΐνης από τη χημική δομή μιας ένωσης.

Acknowledgements

First of all, I would like to kindly thank my supervising Professor, Mr. Leonidas Alexopoulos for providing me the opportunity to work in such an interesting topic in a great environment, as well as for his mentorship and guidance throughout the last two years that I have been a member of the Biomedical Systems Laboratory.

Moreover, I would like to greatly thank both Nikos Meimetis, the former lab member and present PhD student, and Christos Fotis, the PhD student of the lab who both helped me understand the problem of the diploma thesis and develop the model.

Furthermore, I would like to thank all the members of the lab who created a friendly environment and making the lab an exciting workspace.

Finally, I am grateful to my friends and family, for supporting me during not only the diploma thesis, but in the whole journey of my studies at the National Technical University of Athens.

Contents

List of figures

1. Introduction

1.1. Motivation

The motivation of this thesis is to use deep learning methods in order to analyze, quantify and extract meaningful and robust representations from protein signaling networks. Biological networks have been used as data in research purposes as well as data for machine and deep learning models, but the concept of using Graph Neural Networks in processing this data is, to the best of our knowledge, was first introduced by previous lab members, and the purpose of this thesis is to introduce a new method and model to apply to these data. Biological networks, and especially protein signaling networks can possess a huge amount of information about the nodes and connectivity between them, that can be very important for multiple fields of bioinformatics and computational biology. These graphs can be considered as directed acyclic graphs where each node represents a protein, and therefore every branch of the graph contains information on the nodes that construct it and therefore on the whole graph. Therefore, by processing each node and its connections separately and collectively we can extract very important information about the structure and functioning of the graph. In the following pages, we try to use deep learning methods to prove that signaling network data can provide us with significant information if processed properly. Moreover, there is a first approach in trying to use deep learning to infer a protein signaling network from a compound's chemical structure. This method could be extremely important in early drug discovery and CADD since it could help us identify the signaling network (i.e., mechanism of action) of a compound by comparing its chemical structure with other compounds whose signaling network is known. Drug discovery is the process aiming to identify new candidate medications, meaning the development and discovery of new drugs intended to be used for the treatment of specific diseases. Early-stage drug discovery aims to identify the right compound for the right target, for the right disease.

1.2. Systems Biology

Systems biology has been responsible for some of the most important developments in the science of human health and environmental sustainability. It is a holistic approach to deciphering the complexity of biological systems that starts from the understanding that the networks that form the whole of living organisms are more than the sum of their parts. It is collaborative, integrating many scientific disciplines – biology, computer science, engineering, bioinformatics, physics and others – to predict how these systems change over time and under varying conditions, and to develop solutions to the world's most pressing health and environmental issues. [33]

1.3. Proteins

Proteins are large and the most complex molecules known, with very critical roles for the body of a multicellular organism. They are needed in the structure, functioning and organization of tissues, organs, and cells. Proteins are consisted of long chains of numerous smaller units called amino acids which determine the unique 3D structure and specific function of the protein. There are 20 amino acids that can connect in infinite combinations, and each of these amino acids is linked with its neighbor through a covalent peptide bond (another name for proteins is polypeptide). Amino acids are coded by combinations of nucleotides (three DNA building blocks), determined by the sequence of genes.

Proteins can have multiple functions and biological properties which depend on the physical interaction with other molecules. Some of the main functions of proteins are working as antibodies, enzymes, messengers, structural components, and transport.[1]

1.4. Signaling Networks

One of the most important mechanisms in biology is the communication between cells of multicellular organisms. This is achieved through a number of pathways that instantly receive and process signals, with other part of a cell and the external environment. These networks are commonly classified based on the molecules transferred, with two very common being proteins or genes. When cell signaling is involved, meaning the response of a cell to internal and external stimuli (chemical, mechanical nature) and the regulation of its activity is regulated, these networks are called Signaling Networks. [2]

The process can be described by two main steps. At first, an extracellular molecule binds to a specific protein called receptor on a target cell, changing its state to active. Afterwards, the receptor stimulates intracellular biochemical pathways leading to a cellular response, which may involve progression through the cell cycle or changes in gene expression. These internal pathways are controlled and regulated by conserved protein modules which have the ability to mediate interactions between proteins. [3]. Some of the most common of these molecules are Hormones (endocrine system), Neurotransmitters (nervous system), Cytokines (immune system). Signaling networks are extremely important for several functions and mechanisms of multicellular organisms.

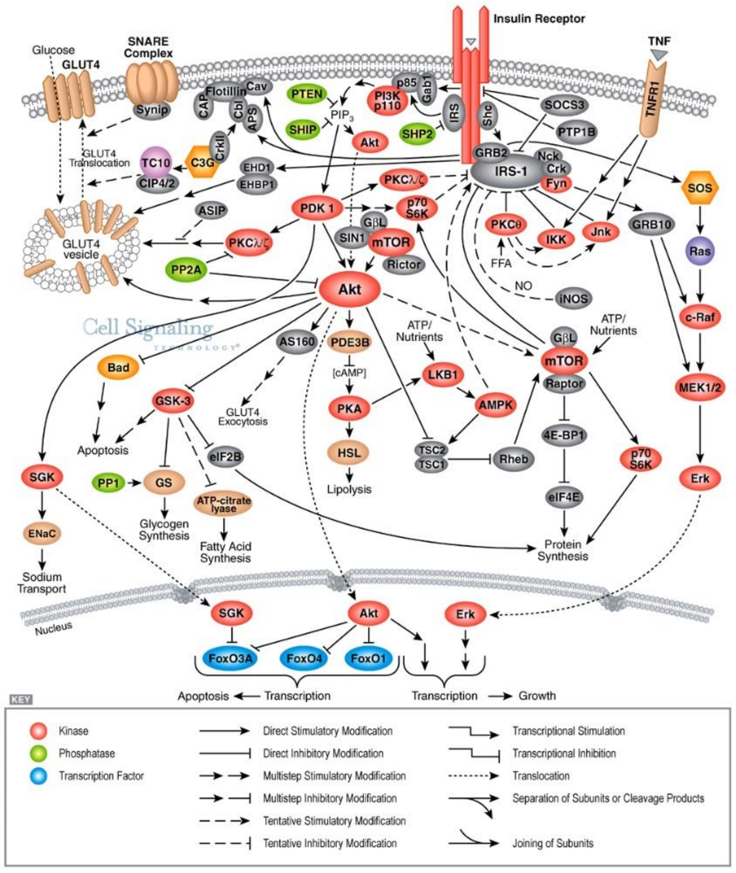


Figure 1: An example of a protein network

1.5. Deep Learning Neural Networks

Neural networks, also known as artificial neural networks, are a method of machine learning that is also the main method used in deep learning algorithms. Their name and functioning are inspired by the human brain and specifically the communication and signal transferring between the neurons of the brain. They are consisted of node layers, starting with an input layer, which is then followed by one or more hidden layers, which end to an output layer. Every node connects to another and has a weight and a threshold, and the node is activated (i.e., sending data to the next layer) when the output of the node is above the specified threshold. Each neuron receives the value of previous connected neurons as input, and maps it into an activation function:

$x_{new} = s(wx_{prev} + b)$

Where s is the activation function and w,b are the trainable parameters.[4]

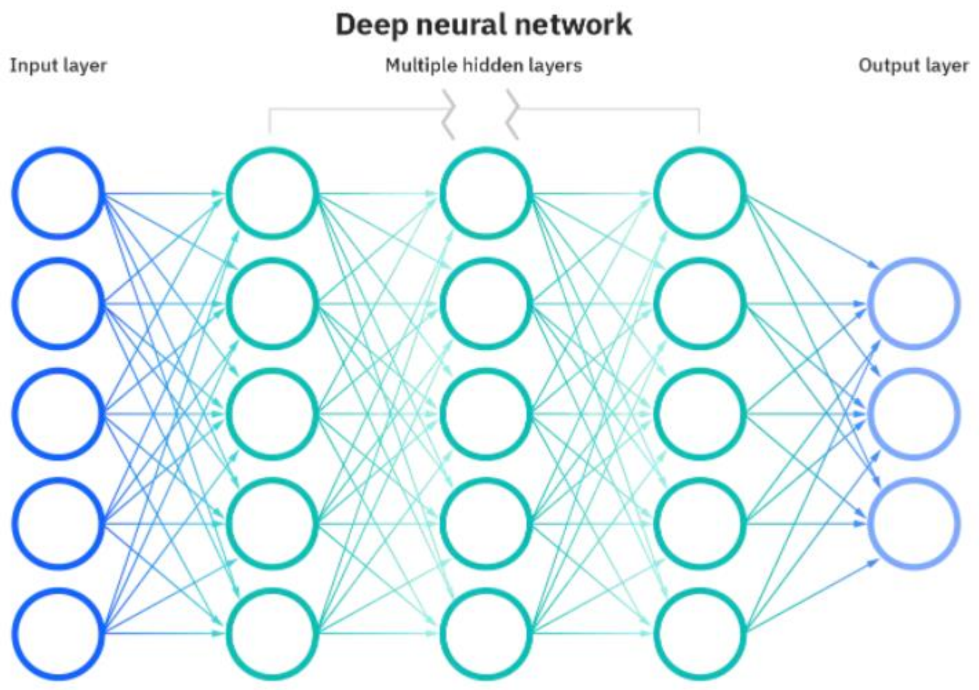


Figure 2: Deep Neural Network Structure

Neural Networks are trained by the optimization of a cost function and rely on training data to learn and improve their accuracy over time.[5]

1.6. Graph Neural Networks

Graph Neural Networks (GNNs) are a class of deep learning methods designed to perform inference on data described by graphs. A GNN is an optimizable transformation on all attributes of the graph (nodes, edges, global context) that preserves graph symmetries (permutation invariances). GNNs adopt a "graph-in, graph-out" architecture meaning that these model types accept a graph as input, with information loaded into its nodes, edges and global-context, and progressively transform these embeddings, without changing the connectivity of the input graph.[34] GNNs are neural networks that can be directly applied to graphs, and provide an easy way to do node-level, edge-level, and graph-level prediction tasks. GNNs can do what Convolutional Neural Networks (CNNs) failed to do. Further information on Graph Neural Networks will be provided in Chapter 3 where the model of the thesis is discussed.

1.7. Graph Convolutional Networks

Graph Convolutional Networks were first introduced by Kipf and Welling [6] and are the main deep learning method used when working with graphs. Convolution refers to the same operation as in simple Convolutional NN. The input neurons are multiplied with a set of weights known as filters or kernels, which slide through the image as a window and enable CNNs to learn features from neighboring cells, while the same filter will be used within the same layer (weight sharing). GCN perform in a very similar way and the model learns features by inspecting neighboring nodes. GCN are a generalized version of CNN since the nodes can be unordered and the connections between them can be complex (where CNN work on structured data), and the equation that describes the problem is $G = (V, E)$ where G is the graph and V, E are the vertices and edges. [7]

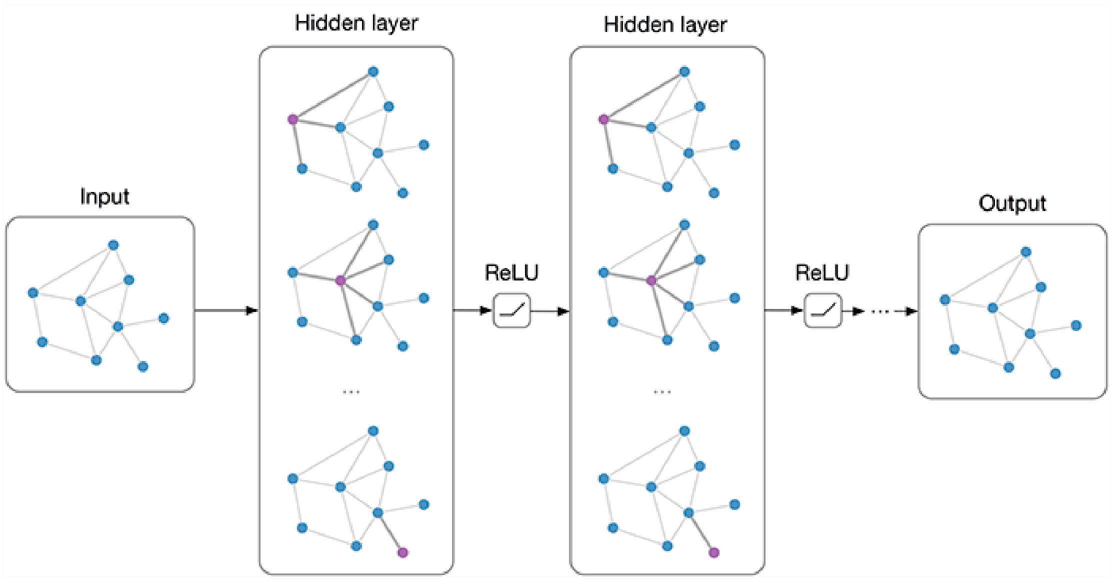


Figure 3: Graph Convolutional Network Structure

The goal is to learn a function of signals/features on a graph $G = (V, E)$ which takes as input:

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

and produces a node-level output Z (an $N \times F$ feature matrix, where F is the number of output features per node). Graph-level outputs can be modeled by introducing some form of pooling operation. [7]

Every neural network layer can then be written as a non-linear function

$H^{(l+1)} = f\left(H^{(l)}, A\right)$,

with $H^{(0)} = X$ and $H^{(l)} = Z$, L being the number of layers.[8]

A very simple and common propagation rule would be $f\left(H^{(l)}, A\right) = \sigma(AH^{(l)}W^{(l)})$ where W is the weight matrix and σ being the activation function (a very common one is ReLu). There are many ways GCNs are implemented, some of which are used in the following thesis.[9]

1.8. Auto Encoders

Autoencoders are a specific type of feedforward neural networks where the input is the same as the output. The input is compressed into a lower-dimensional code and then the output is reconstructed from this representation.

An autoencoder consists of 3 components: the encoder which compresses the input and produces the code from which the decoder reconstructs the input. The encoder and the decoder are fully connected feedforward NN, and the decoder architecture is the mirror image of the encoder. These components use an encoding method, a decoding method, and a loss function to compare the output with the target. [10]

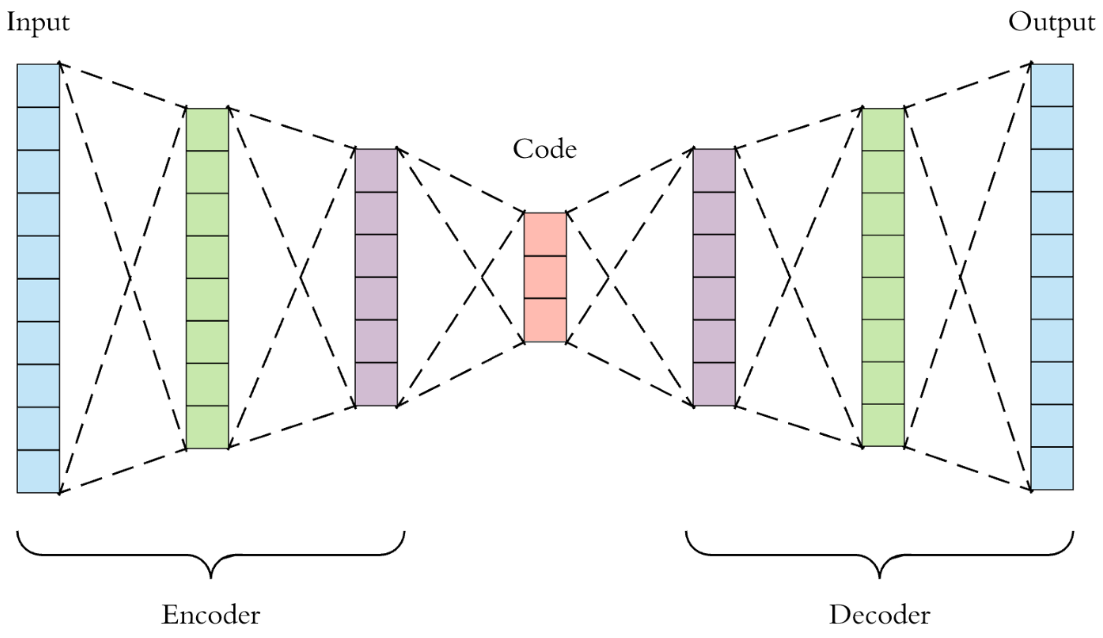


Figure 4: Encoder - Decoder Structure

Autoencoders are mainly a dimensionality reduction (or compression) algorithm and the most important properties of autoencoders are:

- Data-specific: Autoencoders can meaningfully compress data dimensionally similar to the data they have been trained on.
- Lossy: The output of the autoencoder will not be the same as the input, it will be a close but degraded representation.
- Unsupervised: Autoencoders don't need explicit labels to train and therefore are considered an unsupervised learning method. They can be considered *self-supervised* since they generate their own labels from the training data.

Four hyperparameters need to be set before training an autoencoder:

- Code size: number of nodes in the middle layer (smaller size means more compression).
- Number of layers: the autoencoder can have as many layers as we like.
- Number of nodes per layer: the number of nodes per layer decreases with each subsequent layer of the encoder, and increases back in the decoder (decoder is symmetric to the encoder in layer structure)
- Loss function: two very common loss functions are mean squared error (MSE) and binary crossenttropy. [11]

2. Data

2.1. Preprocessing and Quality Control

The effectiveness and predicting capability of even the best neural network architectures and methods is highly depended on the quality of the data selected to accomplish the specific task. Without data of high quality, even the best deep learning algorithms would fail to provide robust predicting results. So, it is very important to analyze the ways data was gathered and processed for our problem. A big part of this data is available on the GitHub page of the NTUA's BioSys Lab, however due to size restrictions, not all data is available online.

2.1.1. CMAP

Connectivity Map (CMAP) is a project by the Broad Institute LINCS Center for Transcriptomics, that has provided the laboratory with the transcriptomic signatures (level 5 transformed z-score) needed to develop the appropriate protein signaling networks that will be used for this thesis. The version of CMAP that was used was the GSE92742 and it is important to mention that only the differential expression of the 978 landmark genes in the L1000 assay was considered. For each gene expression signature, a quality score was calculated, based on transcriptional activity score (TAS), number of biological replicates and whether the signature is an exemplar. The signatures with the highest quality score were selected, based on the process that is described below. [18]

The filtered CMap dataset contains 7722 transcriptomic signatures from 3005 compounds tested across 70 cell lines. During the filtering process, for each compound per cell line, its signature with the highest quality across different dosages and time points was selected. The assigned quality score based on TAS, number of replicates and whether the signature is considered an exemplar is presented in Table S1. Only signatures with Quality score of 1 were used.

Table 1: Signature Quality Score [28]

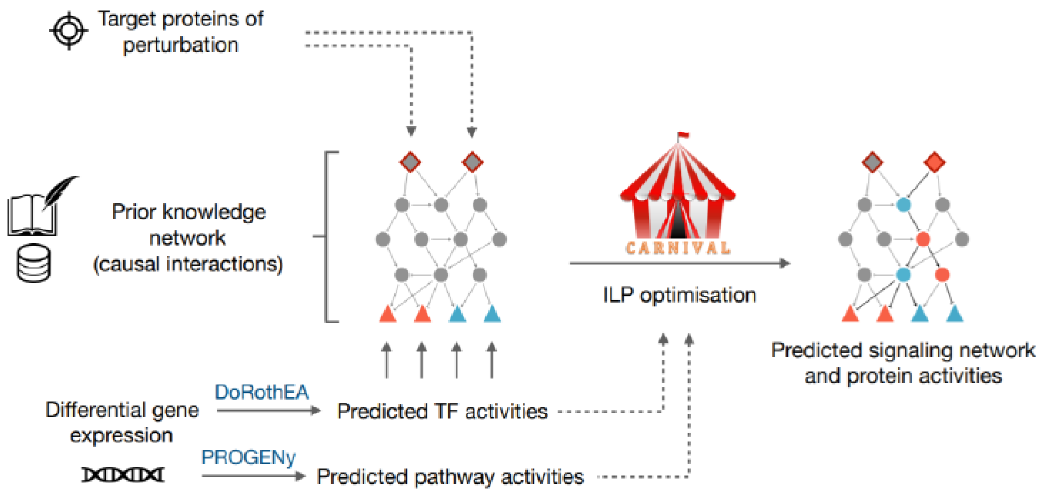
Quality score	TAS	Number of replicates	Exemplar
Q1	> 0.4	> 2	True
Q2	0.2 – 0.4	> 2	True
Q3	0.2 – 0.4	≤ 2	True
Q4	0.1 – 0.2	> 2	True
Q5	0.1 – 0.2	≤ 2	True
Q6	< 0.1	> 2	True
Q7	< 0.1	≤ 2	True
Q8	< 0.1	< 2	False

2.1.2. CARNIVAL

Causal Reasoning pipeline for Network identification using Integer Value programming, or simply CARNIVAL,[19] is a causal network contextualization tool identifying upstream regulatory signaling pathways by using downstream gene expression data. It uses different sources of prior research, as signed, and directed interaction networks between proteins, pathway signatures and transcription factor targets.[20] In order for the protein signaling networks to be produced, the quality 1 data from L1000 CMap was integrated and processed with carnival along with some other resources.

For each signature, DoRothEA R package (gene set resource with signed transcription factor – target interactions) was used in order to infer the transcription factor (TF) activity scores [16]. This method utilizes the VIPER enrichment algorithm and a knowledge base of signed TF-target interactions called Regulons to calculate TF activity scores [17]. After that, the TF activities for each compound perturbation inferred by DoRothEA were transformed into signaling networks using the CARNIVAL pipeline. CARNIVAL solves an ILP optimization problem to infer a family of highest scoring subgraphs, from a prior knowledge network of signed and directed protein-protein interactions, which best explain the TF activities, subject to specific constraints. In our approach OmniPath network was used as the global prior knowledge network [21].

After this process, that was done by previous members of the lab,7788 weighted, signed, and directed signaling networks along with their corresponding unweighted networks per signature (5 – 100 per weighted signaling network) were produced. The weighted networks are produced by adding the unweighted networks, so edge weights refer to the percentage of times a certain edge appeared in the unweighted graphs.



Target proteins of perturbation
Prior knowledge network (causal interactions)
CARNIVAL
ILP optimisation
Predicted signaling network and protein activities
Differential gene expression
DoRothEA
Predicted TF activities
PROGENy
Predicted pathway activities

Figure 5: CARNIVAL Functioning

2.2. Tensorizing the signaling networks

After the process of creating the signaling networks using CARNIVAL, there are 7788 graphs corresponding to the quality 1 signatures. In order for these graphs to be appropriate for integration to the deep learning methods and models that will be described later, their nodes and edges need to be mathematically represented to be fed in the neural networks.

<u>Nodes</u>

Each node of these graphs represents a specific protein of a cell signaling network. Hence every one of these nodes must have a multi-dimensional distributed representation that describes the proteins actions and modes using mathematical representations. One way that this can be done is by encoding the protein function and structure using each protein’s amino acid sequence. One very effective and state of the art way is presented in the following paper [14] where proteins are represented as continuous vectors. SeqVeq (Sequence-to-Vector) is inspired by Natural Language Processing (NLP) tasks and uses a bi-directional model to capture the biophysical properties of sequences from big unlabeled data (UniProt50 database). This method is very effective in predicting results in various tasks by using protein sequence data and has proven to be more effective than other similar methods.

<u>Edges</u>

In each of the graphs the edges represent the connection between two nodes i.e., proteins. In the case of our problem the (directed) connection between two nodes represents two different functions.

The first one is protein interaction. Every cell signaling network is formulated as a directed acyclic graph and each of the proteins either upregulates or downregulates the next protein that is connected, which is presented as 1 and -1 correspondingly. In order for this to be appropriately fed to neural networks, 1 and -1 are formulated into vectors of ones and zeros, and the connection is categorically attributed.

The second one is edge weight. This function is used in weighted graphs, and it quantifies the appearance of the edge in the unweighted graphs. The number ranges from 0 to 1 with the maximum value meaning that an edge was in every unweighted graph.

3. Model

3.1. Theoretical Background of GNN

Let $G = (V, E)$ denote a graph with node feature vectors X_v for $v \in V$. There are two different tasks of interest that can be performed by GNN models: (1) Node classification: each node $v \in V$ has an associated label y_v and the goal is to learn a representation vector h_v of v such that v's label can be predicted as $y_v = f(h_v)$; (2) Graph classification: given a set of graphs $\{G_1,...,G_N\} \subseteq G$ and their labels $\{y_1,...,y_N\} \subseteq Y$, the goal is to learn a representation vector h_G that helps predict the label of an entire graph, $y_G = g(h_G)$.

Graph Neural Networks. GNNs use the node features X_v and graph structure to learn a representation vector of the entire graph, h_G or a node, h_v. A neighborhood aggregation strategy is followed by modern GNNs, where the representation of a node is updated by aggregating representations of its neighbors. After k iterations of aggregation, a node's representation captures the structural information within its k-hop network neighborhood.[12] Formally, the k-th layer of a GNN is

$$\alpha_v^{(k)} = AGGREGATE^{(k)}(\{h_u^{(k-1)}: u \in N(v)\}), h_v^{(k)} = COMBINE^{(k)}(h_v^{(k-1)}, \alpha_v^{(k)})$$

where $h_v^{(k)}$ is the feature vector of node v at the k-th iteration/layer. The initial condition is $h_v^{(0)} = X_v$, and $N(v)$ is a set of nodes adjacent to v. The choice of $AGGREGATE^{(k)}(\cdot)$ and $COMBINE^{(k)}(\cdot)$ in GNNs is very important, and numerous architectures have been proposed.

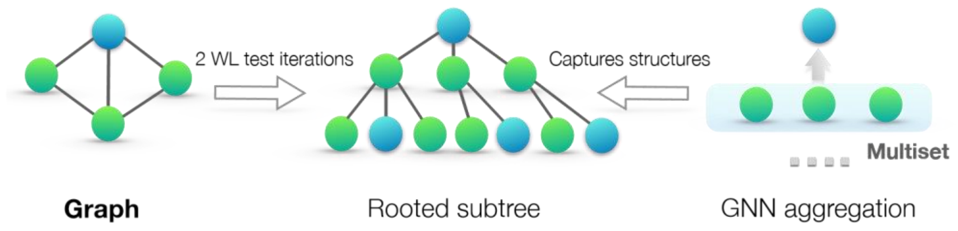


Figure 6: An overview of the theoretical framework.

The figure above is explained:

- Middle panel: rooted subtree structures (at the blue node) used by the WL test to distinguish different graphs.
- Right panel: if a GNN's aggregation function captures the *full multiset* of node neighbors, the GNN can capture the rooted subtrees in a recursive manner and be as powerful as the WL test.

For node classification, the node representation $h_v^{(k)}$ of the final iteration is used for prediction. For graph classification, the READOUT function aggregates node features from the final iteration to obtain the entire graph's representation h_G

$$h_G = READOUT(\left\{h_v^{(k)} \middle| v \in G\right\})$$

3.1.1. Weisfeiler-Lehman test

The question the graph isomorphism problem tries to solve is whether two graphs are topologically identical. This problem is very demanding and there are no polynomial-time algorithms known for it. The Weisfeiler-Lehman (WL) test of graph isomorphism [15] is a computationally efficient and effective test that distinguishes a wide class of graphs. Its 1-dimensional form, known as "naïve vertex refinement", is analogous to neighbor aggregation in GNNs. The WL test iteratively (1) aggregates the labels of nodes and their neighborhoods, and (2) hashes the aggregated labels into *unique* new labels. When the labels of the nodes between two graphs differ, then the algorithm classifies them as non-isomorphic.

With the background of the WL test, a WL subtree kernel that measures the similarity between graphs was proposed. The kernel uses the counts of node labels at different iterations of the WL test as the feature vector of a graph. Intuitively, a node's label at the k-th iteration of WL test represents a subtree structure of height k rooted at the node. Thus, the graph features considered by the WL subtree kernel are essentially counts of different rooted subtrees in the graph.

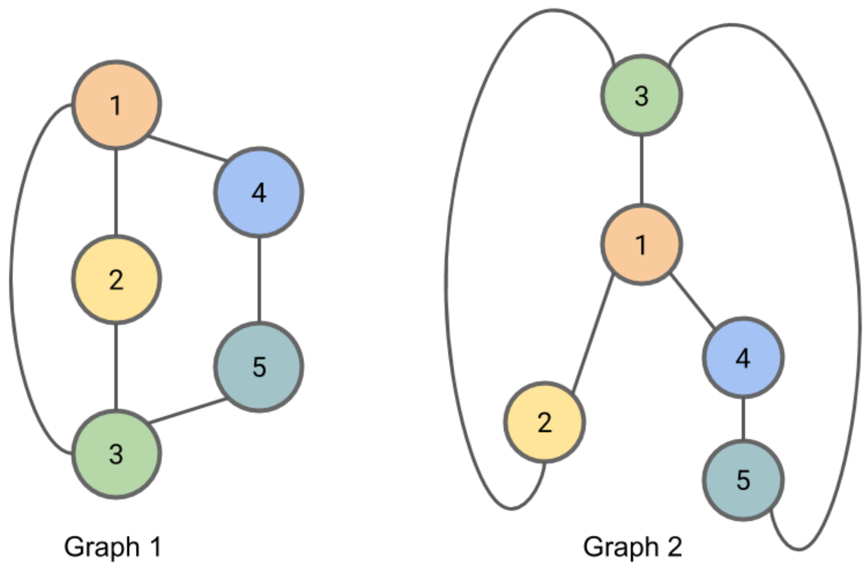


Figure 7: Graph 1 and Graph 2 are isomorphic. The correspondance between nodes is illustrated by the node colors and numbers.

3.1.2. Theoretical Framework: Overview

A GNN recursively updates each node's feature vector to capture the network structure and features of other nodes around it, i.e., its rooted subtree structures. Node input features are considered from a countable universe. Node feature vectors at deeper layers of any fixed model are also considered from a countable universe in finite graphs. For simplicity, each feature vector is assigned a unique label in {a,b,c...}. Then, feature vectors of a set of neighboring nodes form a multiset: the same element can appear multiple times since different nodes can have identical feature vectors.[12]

Definition 1 (Multiset). A multiset is a generalized concept of a set that allows multiple instances for its elements. More formally, a multiset is a 2-tuple X = (S,m) where S is the underlying set of X that is formed from its distinct elements, and m : S → $N \geq 1$ gives the multiplicity of the elements.

The representational power of a GNN is studied by analyzing when a GNN maps two nodes to the same location in the embedding space. A maximally powerful GNN maps two nodes to the same location only if they have identical subtree structures with identical features on the corresponding nodes. Subtree structures are defined recursively via node neighborhoods, and therefore the analysis leads to the question whether a GNN maps two neighborhoods to the same embedding or representation. A maximally powerful GNN would never map two different neighborhoods, i.e., multisets of feature vectors, to the same representation. This means its aggregation scheme must be injective. Thus, we abstract a GNN's aggregation scheme as a class of functions over multisets that their neural networks can represent and analyze whether they are able to represent injective multiset functions.

This reasoning is used to develop a maximally powerful GNN.

3.1.3. Building Neural Networks

The first step is characterizing the maximum representational capacity of a general class of GNN-based models. The ideal scenario is a maximally powerful GNN being able to distinguish different graph structures by mapping them to different representations in the embedding space. However, mapping graphs to different embeddings depends on solving the graph isomorphism problem, meaning isomorphic graphs to be mapped to the same representation and non-isomorphic ones to different representations. For the layer that will be constructed for this thesis, the representational capacity of GNNs is characterized via a slightly weaker criterion: a powerful heuristic called Weisfeiler-Lehman (WL) graph isomorphism test. [12]

Lemma 2. Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $A: G \rightarrow R^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.

Therefore, any aggregation based GNN can be as powerful as the WL test in distinguishing different graphs. Based on theorem 3, we can assume that if the neighbor aggregation and graph-level readout functions are injective, then the resulting GNN is as powerful as the WL test.

Theorem 3. Let $A: G \rightarrow R^d$ be a GNN. With a sufficient number of GNN layers, A maps any graphs G_1 and G_2 that the Weisfeiler-Lehman test of isomorphism decides as non-isomorphic, to different embeddings if the following conditions hold:

a) A aggregates and updates node features iteratively with

$$h_v^{(k)} = \varphi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in N(v)\right\}\right)\right)$$

where the functions f, which operates on multisets, and φ are injective.

b) A's graph-level readout, which operates on the multiset of node features $\{h_v^{(k)}\}$, is injective.

Injectiveness characterizes whether a function preserves the distinctness of inputs in countable sets, and the focus is on this case.

Lemma 4. Assume the input feature space X is countable. Let $g^{(k)}$ be the function parameterized by a GNN's k-th layer for k = 1,...,L, where $g^{(1)}$ is defined on multisets X ⊂ X of bounded size. The range of $g^{(k)}$, i.e., the space of node hidden features$h_v^{(k)}$, is also countable for all k = 1,...,L.

Another important advantage of GNNs beyond distinguishing different graphs, is capturing similarity of graph structures. In the WL test, node feature vectors are one-hot encodings and therefore they cannot capture similarity of subtrees. In contrast, a GNN satisfying the criteria in Theorem 3 generalizes the WL test by learning to embed the subtrees to low-dimensional space. This gives GNNs the ability not only to discriminate different structures, but also map similar graph structures to similar embeddings.

3.2. Graph Isomorphism Network – GIN

After analyzing the conditions behind a powerful GNN, the next step of the thesis is to create an architecture that probably satisfies theorem 3. Graph Isomorphism Network (GIN) generalizes the WL test and consequently has the most discriminative power of other GNNs.

To model injective multiset functions for the neighbor aggregation, this thesis follows a theory of "deep multisets", i.e., parameterizing universal multiset functions with neural networks. Based on the following lemma, we assume that sum aggregators can represent injective, universal functions over multisets.

Lemma 5. Assume X is countable. There exists a function f: $X \rightarrow R^n$ so that $h(X) = \sum_{x \in X} f(x)$ is unique for each multiset $X \subset X$ of bounded size. Moreover, any multiset function g decomposed as $g(X) = \varphi(\sum_{x \in X} f(x))$ for some function φ.

A significant difference between deep multisets and sets is that some popular injective set functions, such as the mean aggregator, are not injective multiset functions. Using the mechanism for modeling universal multiset functions from Lemma 5 as a building block, aggregation schemes that can represent universal functions over a node and the multiset of its neighbors satisfying Theorem 3 can be conceived. The following corollary provides a concrete formulation among many such aggregation schemes.

Corollary 6. Assume X is countable. There exists a function $f : X \rightarrow R^n$ so that for infinitely many choices of , including all irrational numbers, $h(c, X) = (1 + \varepsilon) \cdot f(c) + \sum_{x \in X} f(x)$ is unique for each pair (c,X), where $c \in X$ and $X \subset X$ is a multiset of bounded size. Moreover, any function g over such pairs can be decomposed a $g(c, X) = \varphi((1 + \varepsilon) \cdot f(c) + \sum_{x \in X} f(x))$ for some function ϕ.

Based on the universal approximation theorem [29][30] multi-layer perceptrons (MLPs) can be used to model and learn f and ϕ in Corollary 6. In practice, we model $f^{(k+1)} \circ \phi^{(k)}$ with one MLP, because MLPs can represent the composition of functions. In the first iteration, if input features are one-hot encodings as their summation alone is injective MLPs before summation are not needed. We can make a learnable parameter or a fixed scalar. Then, GIN updates node representations as

$$h_v^{(k)} = MLP^{(k)}((1 + \varepsilon^{(k)}) \cdot h_v^{(k-1)} + \sum_{u \in N(v)} h_u^{(k-1)})$$

GIN is a maximally powerful GNN, while being simple.

3.3. Attention Layer

After the inputs have been integrated and processed by the GIN layer, the three output layers (there are three GIN layers followed by a batch normalization and an activation function ReLu) can be added. Instead of this method, for the purpose of this thesis we choose to have one more layer, called Multi Scale Node Attention layer. This layer provides attention (through a changeable weight matrix) to each node, i.e. protein based on the connections and structure of the graph [22]. The equation of the layer is the following:

$$ATT_{\Theta}\left(U_g\right) = \sum_{n=1}^{N} \sigma(u_n^T ReLU(\Theta(\frac{1}{N}\sum_{m=1}^{N} u_m)))\, u_n$$

- σ sigmoid
- N the number of nodes
- $\Theta^{(K)}$ weight parameters for k-th node embedding layer

The intuition behind this equation is that, during the generation of graph-level embeddings, the attention weight assigned to each node should be adaptive to the graph proximity metric. This can be achieved by determining the weight by both the node embedding u_n and a learnable graph representation. The learnable graph representation is adaptive to a particular graph proximity via the learnable weight matrix (k). Specifically, this equation used all the node outputs, adds them and produces a vector, and after multiplying this vector with a weight matrix, an activation function ReLU, appropriate matrix multiplications and a sigmoid it produces a final vector. After all these vectors are summed, the three layers are concatenated (instead of added) using the following equation

$$h_G = MLP_W(||_{k=1}^{K} ATT_{\Theta^{(k)}}(U_g))$$

- || concatenation
- K the number of neighbors aggregation layers
- ATT multi-head attention mechanism that transforms the node embeddings into a graph level embedding
- MLP

The intuition behind this equation is that, instead of only using the node embeddings generated by the last neighbor aggregation layer, we use the node embeddings generated by each of the K neighbor aggregation layers.

After this, the outputs are integrated into a simple dense layer so that the appropriate dimensions are assigned.

3.4. Mutual Information

3.4.1. Theoretical Background on Information Theory

For the purpose of the thesis, training of the encoder of the signaling networks to produce the appropriate embeddings will be made using mutual information. Before analyzing the construction of the algorithm, some theoretical background and definitions will be provided. [31]

Entropy

Let X be a random variable on a (discrete) space X, and x an element from X. For every positive integer d, we denote by X a d-dimensional random vector$(X_1, \dots, X_d) \in X^d$, and by the letter x an element from X^d.

The (Shannon) entropy [4] of a random variable X on a discrete space X is a measure of its uncertainty during an experiment. It is defined as

$H[X] = -\sum_{x \in X} \Pr[X = x] \cdot \log(\Pr[X = x])$

The joint entropy of a pair of random variables (X, Y) expresses the uncertainty one has about the combination of these variables:

$H[X, Y] = -\sum_{x \in X, y \in Y} \Pr[X = x, Y = y] \cdot \log(\Pr[X = x, Y = y])$

Finally, the conditional entropy of a random variable X given another variable Y expresses the uncertainty on X which remains once Y is known:

$H[X|Y] = -\sum_{x \in X, y \in Y} \Pr[X = x, Y = y] \cdot \log(\Pr[X = x|Y = y])$

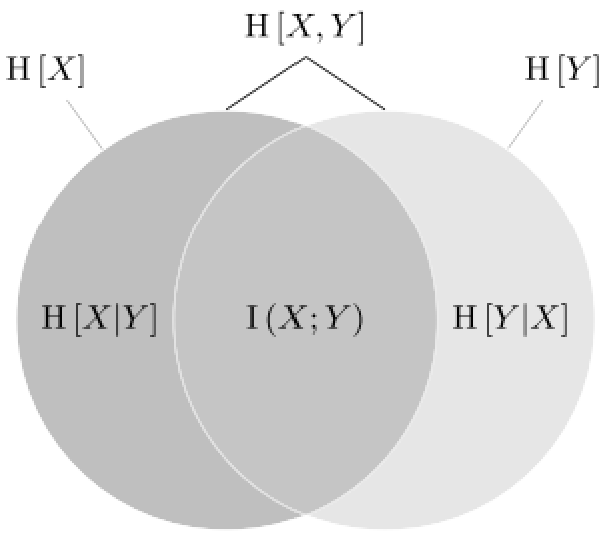


Figure 8: Mutual Information and Entropy

Mutual Information

The mutual information is a general measure of the dependence between two random variables. It expresses the quantity of information one has obtained on X by observing Y . On a discrete domain, the mutual information of two random variables X and Y is defined as:

$$I(X;Y) = \sum_{x \in X, y \in Y} \Pr[X = x, Y = y] \cdot \log \left(\frac{\Pr[X=x, Y=y]}{\Pr[X=x] \cdot \Pr[Y=y]}\right)$$

It can be seen as the Kullback–Leibler divergence [4] between the joint distribution $\Pr[X = x, Y = y]$ and the product distribution $\Pr[X = x] \cdot \Pr[Y = y]$.

In terms of Shannon entropy, MI can be defined as:

$$\begin{aligned} I(X;Y) &= H[X] - H[X|Y] \\ &= H[X] + H[Y] - H[X,Y] \\ &= H[X,Y] - H[X|Y] - H[Y|X] \end{aligned}$$

3.4.2. Training with Mutual Information

Based on the available research (Deep Graph Infomax) the encoder should be trained by maximizing local mutual information. Through that, the encoder will obtain node representations that capture the global information content of the entire signaling network. The signaling network is represented by a summary vector $\vec{s}$ and each node's patch representation is $\vec{h}$. [31]

For the local MI to be optimized, we construct a discriminator $D: \mathbb{R}^{d_k} \times \mathbb{R}^{d_k} \rightarrow \mathbb{R}$ that is used for each node i and $D(\vec{h}_i, \vec{s})$ represents each node's probability scores assigned to the summary-patch pair.

The discriminator in a GAN is simply a classifier [23]. It tries to distinguish real data from the data created by the generator. It could use any network architecture appropriate to the type of data it's classifying. The architecture of the discriminator is described below. We assume that the input of the discriminator is x. The discriminator contains three consecutive simple dense layers followed by an activation function ReLU. After the input is integrated into the model and into the three layers, the output is x_1. Moreover, the input x is also integrated into a simple dense layer called dense shortcut and the output of this is x_2. The final output of the discriminator is $x_1 + x_2$.

This architecture is applied twice. Local Discriminator uses the node encodings as input and Global Discriminator uses the sum of the node encodings as input. Finally, those two are integrated into a dot layer, which is a Layer that computes a dot product between samples in two tensors and the final output is used in the mutual information training as the one output(prediction).

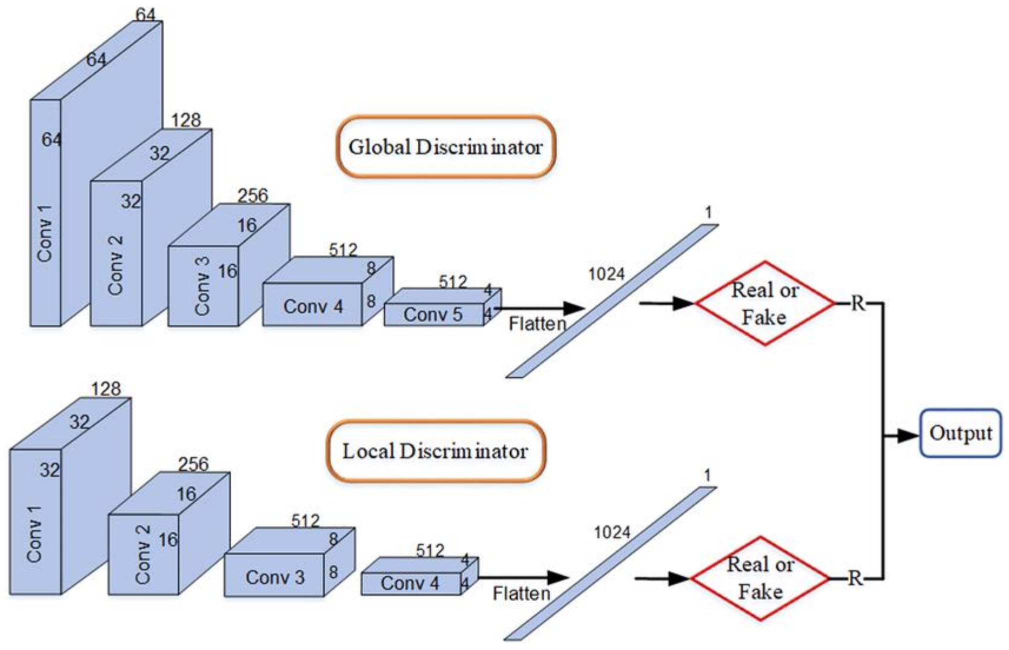


Figure 9: Discriminators

For the discriminator to work properly, both positive and negative samples must exist.

For the graph G with a summary $\vec{s}$ negative samples are produced by pairing the summary with patch representations from another graph $\tilde{G}$ namely $\vec{\tilde{h}}_j$. For the purpose of this thesis, we consider two different classes of positive and negative samples. Two graphs are considered positive or negative base on the following conditions.

Positives: Same signature id (same experiment) or duplicates (same experiment processed in different time)

Negatives: Not same signature id or not duplicates

For the loss function to be defined, we will use the Jensen Shannon Mutual Information estimator, described in the paper (Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization.) We consider $I_{\varphi,\psi}$ the estimator which is modeled by a discriminator D_ψ parametrized by a neural network with parameters ψ. The Jensen-Shannon criteria is:

$$I_{\varphi,\psi}(h_\varphi^i, s_\varphi) := E_P[-sp\left(-D_{\psi,\varphi}(h_\varphi^i, s_\varphi)\right)] - E_{\tilde{P}}[sp(D_{\psi,\varphi}(\widetilde{h_\varphi^i}, s_\varphi))]$$

P represents the empirical distribution of the input data set, $\tilde{P}$ represents the negative distribution from which we sample from and sp is the softplus function $sp(z) = (1 + e^{z})$

Adjusting the Jensen-Shannon estimator to our problem, we calculate two different losses, one loss $loss_{sig\_id}$ for the positive and negative samples regarding same signature id and one loss for the positive and negative samples regarding duplicates $loss_{duplicates}$.

Moreover, for the final loss of the model to be constructed, there is another term referring to a regularization loss, which denotes matching the pushforward distribution of our summary vectors to a prior distribution, with the most effective being the uniform distribution. This happens through a prior discriminator which has a simple architecture consisting of three dense layers followed by an activation function ReLU. The encoded sum is integrated to this discriminator and terms of a uniform distribution are also processed by the discriminator and the outputs are added together. The final loss, $loss_{prior}$ is equal to the negative value of the mean of the encoded sum.

Therefore, the loss function through which the model is trained is equal to

$$loss = loss_{sig\_id} + loss_{duplicates} + loss_{prior}$$

3.5. Structure of the model

After creating the graphs using Carnival, they are processed as analyzed previously so that four different vectors or matrices are produced. The goal is to tensorize the signaling networks into nodes, edges, edge attributes, activity matrix. These data are the inputs of the model. At first, the activity matrix is integrated into a Projection Model, which contains three simple dense layers, so that its dimensions are increased. This is very important, because at first the dimensions were one, and the importance of the activity matrix would be lost when compared to the higher dimensions of the nodes. After passing through the projection model, the activity matrix is concatenated with the nodes. The concatenated nodes matrix along with the edges and edge attribute matrices are integrated into the first encoder which is consisted of three consecutive GIN layers followed by a batch normalization, an activation function ReLU , and an attention layer. The output of the first three layers is x_i, and the output of the attention layer is a_i. The outputs of this function are two: a layer concatenating the outputs of the three attention layers (node embeddings) and the sum of the three layers x_i (all nodes' embeddings). After that, the node embeddings are integrated into another encoder. This second encoder consists of three simple dense layers followed by a batch normalization, an activation function ReLU, a drop dense function and a normalization. The output of this function is called encoded and is then integrated into the global discriminator. All node embeddings are integrated into the local discriminator and the outputs of the two discriminators are passed through a layer that produces their dot product; the output is called result. The encoded is processed by the prior discriminator and the logarithm of the output is added to a term containing samples from a uniform distribution, constructing a term called prior sum. Finally, the nodes, edges, edge attributes, activity matrix are used as the input of the mutual information training model, and the result and prior sum are used as the outputs that will help the model train through minimizing the loss function. Finally, the mutual information training model is fit into the data produced by the train generator, which produces the four matrices analyzed below along with the masks for the same signature id and the duplicate that are needed.

Table 2: Training Hyperparameters

Hyperparameters	**Value**
Batch Size	96
Epochs	8
Optimizer	Adam
ReduceLROnPlateau	-
Learning Rate	0.001
Batch Normalization momentum	0.6
Weight Initializer	Glorot Normal

4. Embeddings Quality Evaluation

After training the model and producing the 128 dimension embeddings , it is of vital significance to evaluate their quality, so that there is certainty that the model works appropriately and the embeddings can be used in other deep learning models that will be analyzed in this thesis. In order to evaluate the embeddings, three different tasks were used, that analyze the quality of embeddings regarding their differentation regarding same signature graphs, duplicates and mechanism of action. The three tasks were written in programming language R and their results will be explained .

For the following plots , G2V refers to Graph2Vector, a graph embedding approach based on the idea of the doc2vec approach that uses the skip-gram network, GT-MI refers to Graph Transformers Mutual Information, a model constructed bya previous lab member, and MI-GIN-TF2 meaning Mutual Information- GIN Layer-Tensorflow 2 refers to the model constructed in this thesis.

4.1. Task 1 : Same Signature ID vs Different Signature

The purpose of the first task is to differentiate the signaling networks that have the same signaling network id (meaning the same experiment) from those that have different signatures. As we can see , the model of this thesis differentiates the two categories, since the mean of the same sigantures is near zero and the mean of different signatures is above 0.5 . Both samples have some outliers, however the model performs very effectively, since there is no significant overlap between the two. Moreover, it can be said that our model performs better than the other two, since the mean of the same signatures is very near 0, which is the desirable situation, since these samples share a lot of mutual information and therefore their embeddings are accepted to be very similar and close and consequently the distances of the embeddings very near 0.

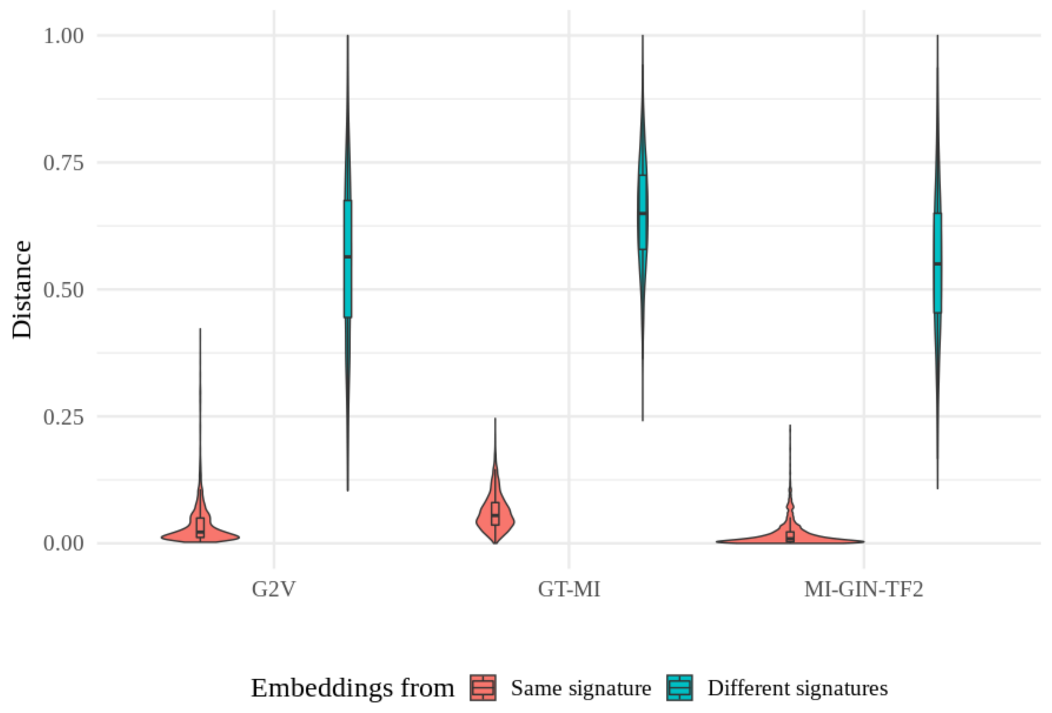


Figure 10: Embeddings Quality Evaluation Task 1

4.2. Task 2: Duplicates vs Random Signatures

The purpose of the second task is to differentiate signaling networks that are duplicates (same experiment, i.e. same drug, dosage, duration but conducted at different moments) from random signatures. As we can see, our model is very effective in this task since it manages to differentiate the two categories. Specifically, the mean distance of the duplicate embeddings is near 0.2, whereas the mean of the randoms is over 0.5. Moreover, as we can see our model can be consider more effective than the other two, since there is less overlap between the two violin plots and the two mean distances are further than the other two methods. Therefore, our model proves successful in this task and our embeddings have effective quality in differentiating duplicates from randoms.

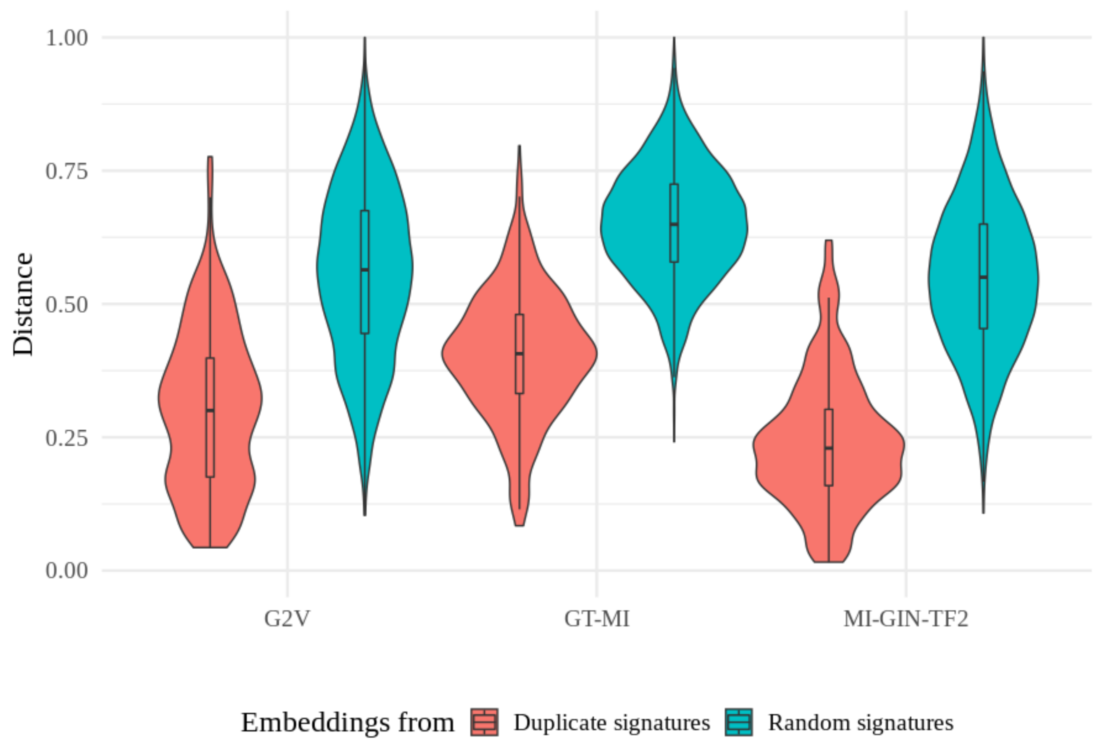


Figure 11: Embeddings Quality Evaluation Task 2

4.3. Task 3: t-SNE Visualization - Mechanism of Action

In medicine, a term used to describe how a drug or other substance produces an effect in the body. For example, a drug's mechanism of action could be how it affects a specific target in a cell, such as an enzyme, or a cell function, such as cell growth. Knowing the mechanism of action of a drug may help provide information about the safety of the drug and how it affects the body. It may also help identify the right dose of a drug and which patients are most likely to respond to treatment. Also called MOA. [13]

The following plot represents the t-SNE Visualization of the embeddings. T-distributed stochastic neighbor embedding (t-SNE) is a statistical method for visualizing high-dimensional data by giving each datapoint a location in a two or three-dimensional map. Therefore, the dimensionality of the embeddings is reduced from 128 to 2. Each point in the graph represents a signaling network. As we can see, the following graph is consisted of a big cloud instead of clusters.

However, there are some clusters, for example HDAC inhibitors, MTOR inhibitors and protein synthesis inhibitors. This task is very difficult since the embeddings contain a lot of information about the signaling networks and reducing their dimension from 128 to 2 means that a huge proportion of this information is lost. However, the fact that there are some clusters of experiments with similar mechanism of action, as well as the fact that the other methods (GT-MI) had very similar t-SNE visualizations, proves that our model is successful in this task.

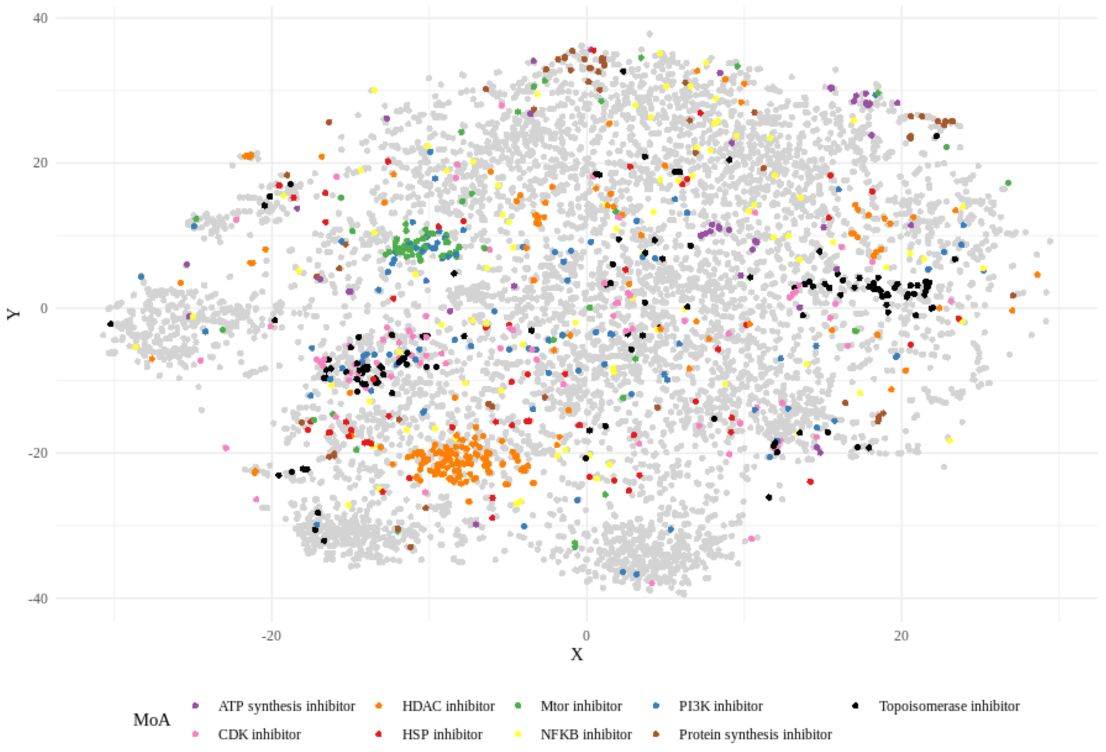


Figure 12: Embeddings Quality Evaluation Task 3

5. Infer a protein signaling network from a compound’s chemical structure

5.1. Introduction

After producing the embeddings for the signatures, these will be processed in order to be integrated into another deep learning model called DeepSIBA [32]. The purpose of this part of the thesis is to infer a protein signaling network from a compound’s chemical structure.

Identification of protein interactions (PPIs) is at the center of molecular biology considering the unquestionable role of proteins in cells. There has recently been a rapid progress in computational methods for determining protein targets of small molecule drugs, which will be termed as compound protein interaction (CPI). Data for CPI has been accumulated and curated significantly both in quantity and quality. Computational methods have become powerful ever to analyze such complex the data.

To achieve generalization of our current knowledge on CPI prediction using AI methods, the computational methods are grouped into five categories: tree-based ML, network- and kernel-based ML, and three deep learning (DL) based architectures. Specifically, with this approach, the goal is to be able to predict whether two compounds will activate similar protein signaling networks based on their chemical structure. This is very significant since it could prove very beneficial for early drug discovery. Predicting whether two compounds will activate similar protein to protein interaction, and therefore similar mechanism of action based only on their chemical structure is one state of the art method that would be very helpful for drug discovery researchers, The long-term goal of this is being able to find out the protein signaling network of compounds constructed on paper without any further experiments. For this thesis, a deep learning-based architecture will be used, specifically the deepSIBA model, a graph convolution model. [24][25]

Predicting whether a chemical structure leads to a desired or adverse biological effect can have a significant impact for in silico drug discovery. For this thesis, we used a deep learning model where compound structures are represented as graphs and then linked to their biological footprint. To make this complex problem computationally tractable, compound differences were mapped to biological effect alterations using Siamese Graph Convolutional Neural Networks. In previous work and research, the model was able to encode molecular graph pairs and identify structurally dissimilar compounds that affect similar biological processes with high precision. Additionally, by utilizing deep ensembles to estimate uncertainty, the model provided reliable and accurate predictions for chemical structures that are very different from the ones used during training. Therefore, this model will be used for this thesis.

Following, some basic background of the deepSIBA model along with how it was adjusted for the purpose of this thesis will be mentioned.

5.2. Background on deepSIBA

Transcriptomic signatures from compound perturbations along with their respective chemical structures were retrieved from the CMap dataset. For each compound perturbation, the embeddings that were calculated by the model of this thesis were used. Specifically, every different signature had up to 100 different signaling networks, and 128 size embeddings were produced for each of them. Using a code in R, a mean value was calculated for these different signaling networks, so that each signature referred to only 128 embeddings. Afterwards, pairwise distances for these signatures were calculated, using the Euclidean distance and cosine similarity functions. Cosine similarity had better results and therefore was used for the training of the model. During the learning phase, the proposed model is trained to predict the pairwise distance between compounds' using only their chemical structure as input. The input chemical structures are represented as undirected graphs, with nodes being the atoms and edges the bonds between them and encoded using a Siamese GCNN architecture. During inference, the model is tasked to predict the biological effect distance between reference and unknown compounds.

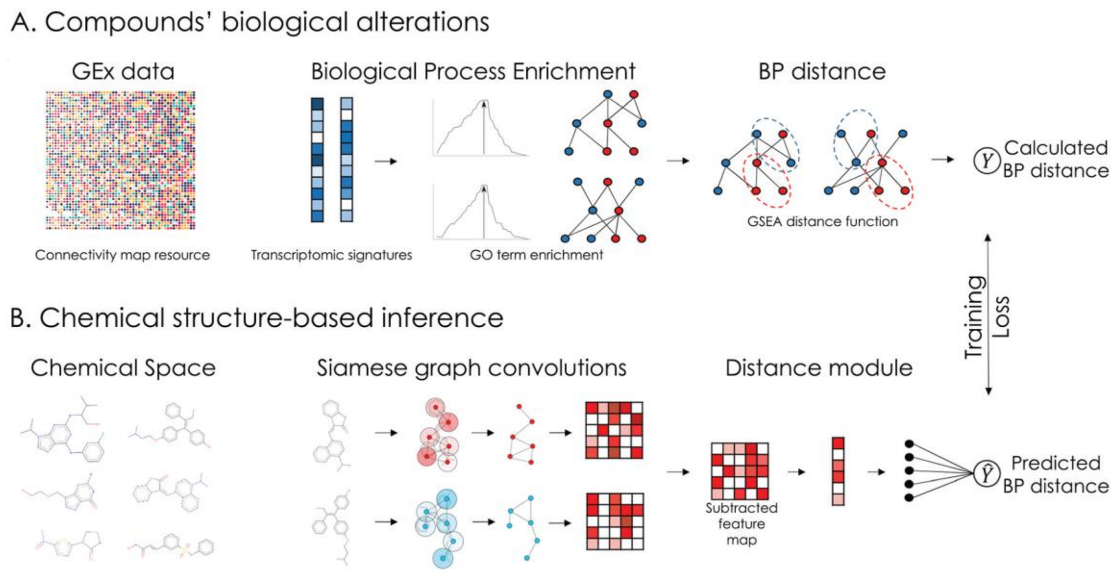

A. Compounds' biological alterations
GEx data
Connectivity map resource
Biological Process Enrichment
Transcriptomic signatures
GO term enrichment
BP distance
GSEA distance function
Calculated BP distance
Training Loss
B. Chemical structure-based inference
Chemical Space
Siamese graph convolutions
Distance module
Subtracted feature map
Predicted BP distance

Figure 13: Overview of deepSIBA model

5.3. Siamese GCNN

A schematic representation of the model's architecture is presented in the figure below. The learning model takes as input the chemical structures of compound pairs and predicts their biological distance

The input of the model, chemical structures, are represented as undirected graphs, where atoms represent the nodes and bonds between them represent the edges. Every chemical structure is encoded with 3 matrices: the atom array, which contains atom-level features, the bond array, which contains bond-level features and the edge array, which describes the connectivity of the compound.

The learning model consists of two Siamese encoders (shared weights) that embed the input graphs into a high dimensional latent space and a trainable distance module that outputs the final distance prediction. The overall goal of the Siamese encoder is to learn task-specific compound representations. Siamese encoders have 3 graph convolutional layers that learn neighborhood-level representations, and a convolutional layer that extracts compound level features.

The feature maps of the last Siamese layers are then subtracted, and their absolute difference is passed to the distance module. The distance module consists of 2 convolutional layers, which extract important features from the difference of the feature maps and 3 fully connected layers that aim to combine those features, while progressively reducing the dimensions. Finally, a Gaussian regression layer outputs a mean and variance of the biological effect distance between the compound pair. By treating the distance as a sample from a Gaussian distribution with the predicted mean and variance, the model is trained end-to-end by minimizing the negative log-likelihood criterion given by [26]

$$-log p_\theta(y_n|X_n) = -\frac{1}{2} log \sigma_\theta^2(x) - \frac{1}{2\sigma_\theta^2(x)}\left(y - \mu_\theta(x)\right)^2 + const$$

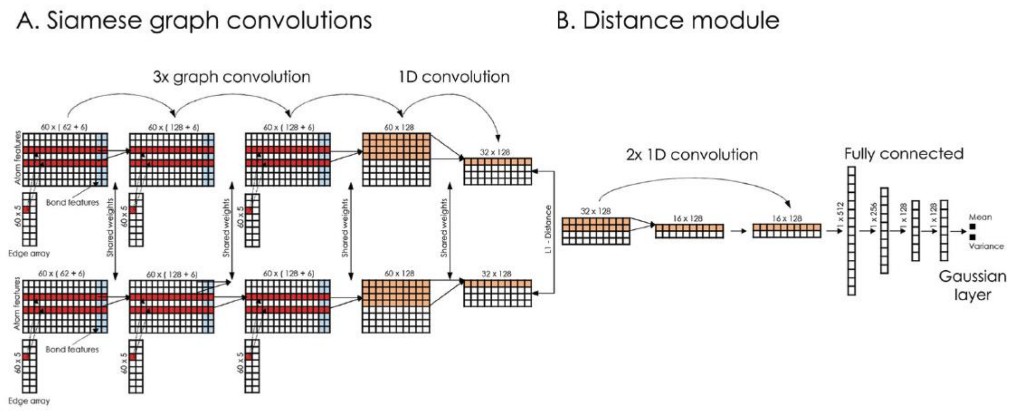


Figure 14: Siamese Graph Convolutions

In general, deepSIBA is used as a black box since for the purpose of this thesis we only integrate the data produced by the model without altering or analyzing the parameters and structure of the deepSIBA architecture and model. Therefore, it is expected that the results will not be great, since deepSIBA was designed for another problem and not for predicting signaling network embeddings. However, since this model has proven very powerful in previous research and is based on compounds' chemical structure and pairwise distances to train, it is appropriate for the purpose of this thesis, which is a first trial in training a deep learning model to predict the similarity of the protein signaling network of two different compounds. We will not analyze further the functionality of deepSIBA, which can be found in [32] and we will proceed with the data processing and the performance evaluation of the training.

5.4 Data

Of all the different cell lines for which the CARNIVAL has produced signaling networks, four of them have the most compounds with quality score 1 whose embeddings will be used in the training of the deepSIBA model. It is very important to keep only these cell lines, because based on research of previous thesis of the lab, these are the most credible and therefore will produce the most robust results. These four cell lines are all human cancer cell lines.

Table 3: Number of compounds for different cell lines

Cell Line	Compounds
A375	711
MCF7	813
PC3	729
VCAP	730

For each of these different cell lines, the pairwise differences for all different compounds were calculated. To train and validate the deep siba model, these pairwise distances were divided to train and test set. For splitting the total, we used a combination of the 80-20 rule (80% of the sample is train set and 20% of the sample is test set) and an algorithm of previous research. This algorithm divides the compounds based on their similarity, so as similar compounds will be divided into the train and test sets, and there will not be bias in the model. After splitting the totals, and removing some samples that did not have embeddings of good quality (multiple zeros etc), the train and test set for each cell line contained the following number of samples

Table 4: Number of pairwise distances for train and test set

Cell Line	Total	Train	Test
A375	216520	172167	44353
MCF7	285542	238871	46671
PC3	221716	178595	43121
VCAP	234326	203590	30736

The pairwise distances for the different cell lines were calculated using the cosine similarity function:[27]

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Before training the model, it is very significant to validate that the distributions of the distances for the train and tests sets are similar, since this is a very important factor for an effective and robust training. As seen below, all cell lines have similar distributions between the two sets.

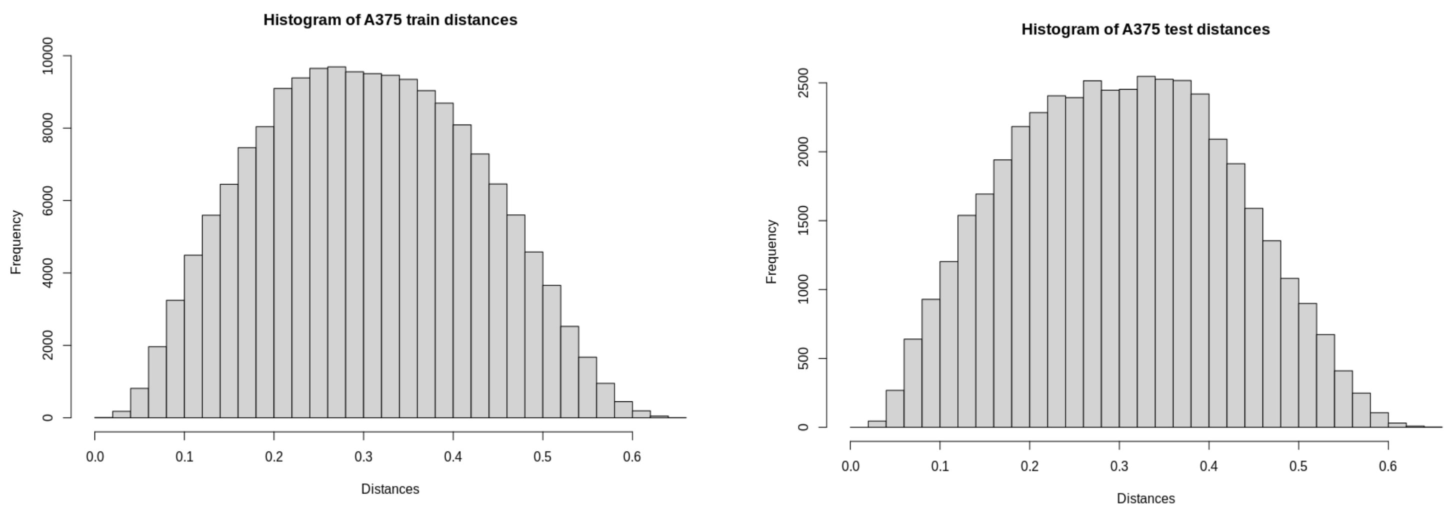


Figure 15: A375 Distances Histogram

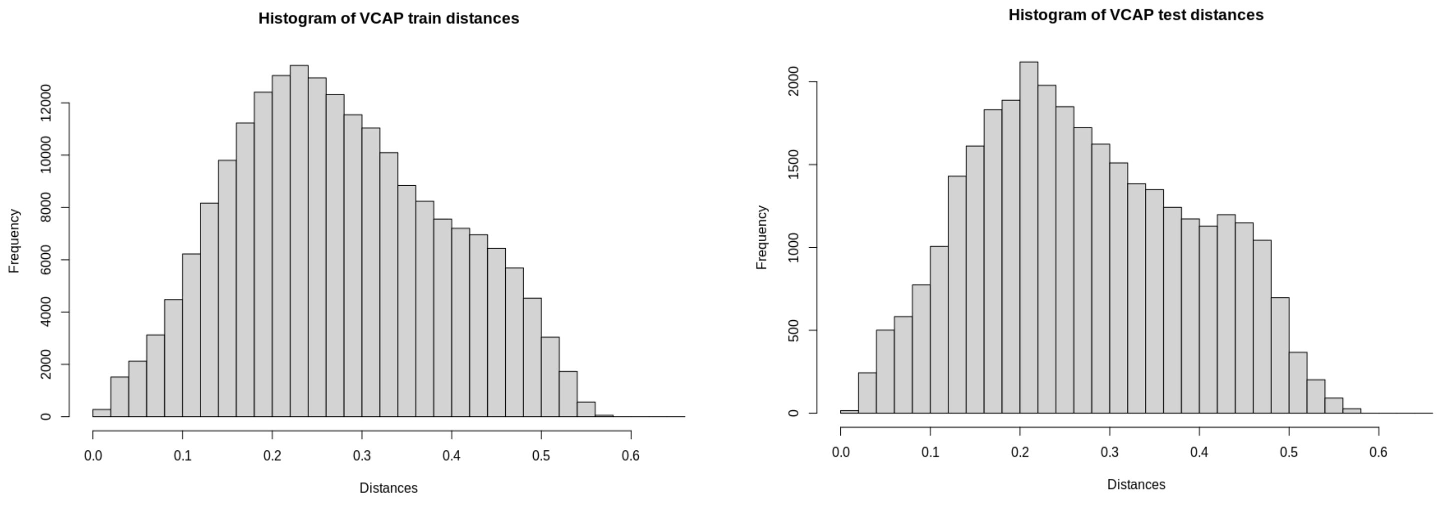


Figure 16: VCAP Distances Histogram

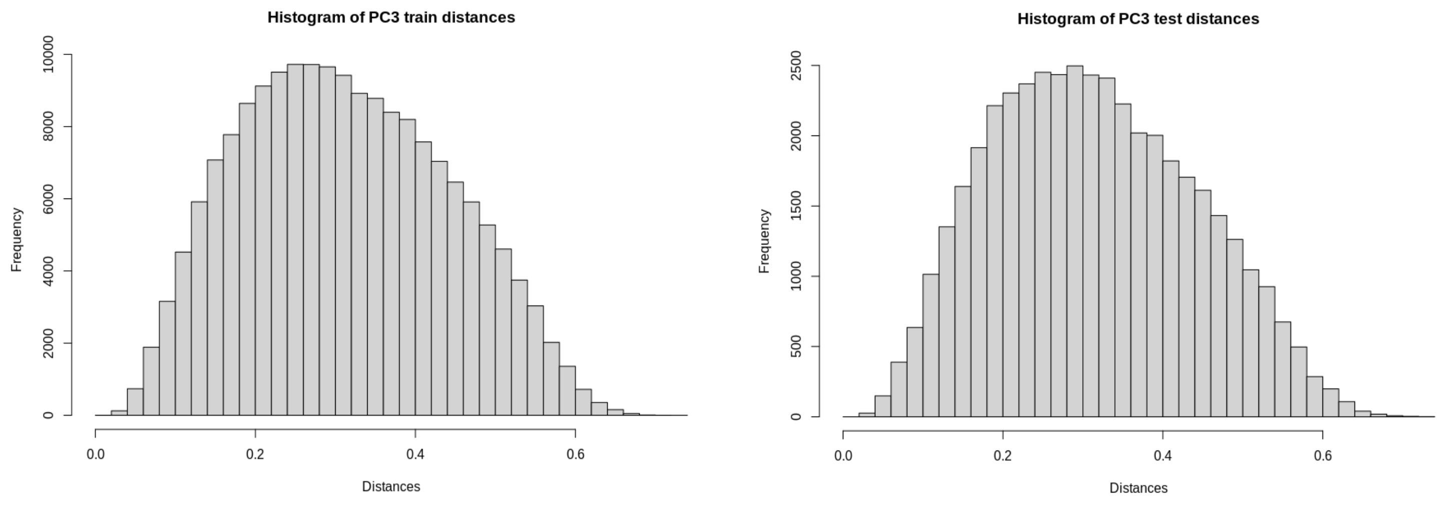
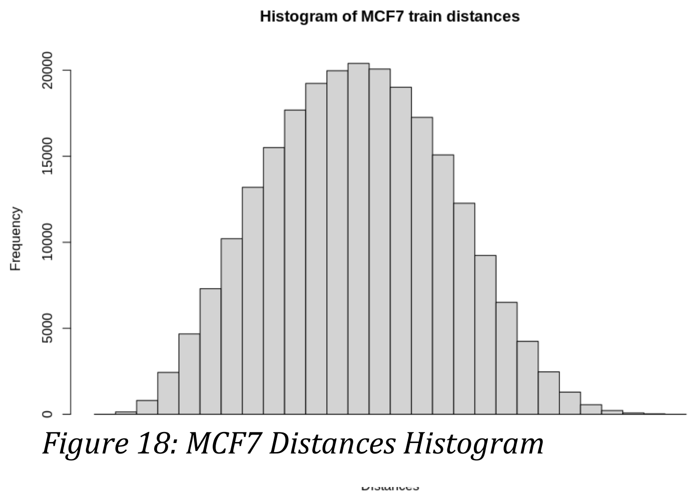


Figure 17: PC3 Distances Histogram



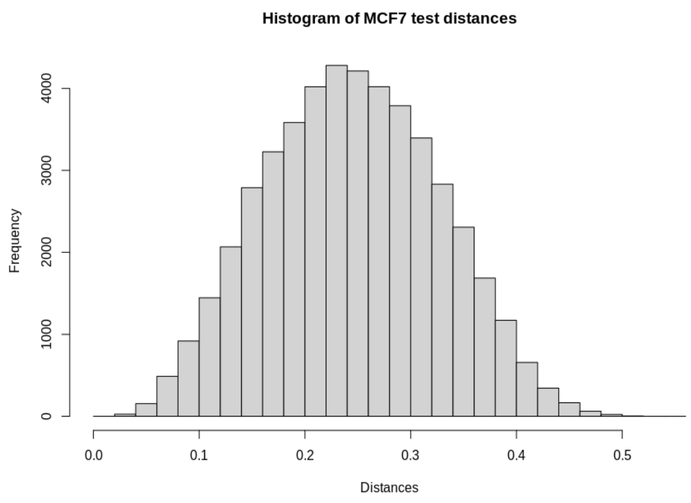


Figure 18: MCF7 Distances Histogram

6. Performance Evaluation

Across all test scenarios, model performance was evaluated in terms of Mean Squared Error (MSE), Pearson's r and precision. MSE and Pearson's r were calculated between the predicted and computed distance values. To calculate precision, the continuous distance values were transformed to binary form by comparing them with an appropriate distance threshold. Even though the learning task is a regression problem, given its nature and potential applications, high precision ($\frac{true\ positives}{positives}$) is important in order to avoid false positive hits for validation experiments.

Table 5: Training Metrics

Metric	Definition	Formula
MSE	The **mean squared error** (MSE) tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line and squaring them. The squaring is necessary to remove any negative signs. It also gives more weight to larger differences. It's called the mean squared error as you're finding the average of a set of errors. The lower the MSE, the better the forecast.	$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y_i})^2$
Pearson's r	Pearson's correlation coefficient is the covariance of the two variables divided by the product of their standard deviations. The form of the definition involves a "product moment", that is, the mean (the first moment of the origin) of the product of the mean-adjusted random variables; hence the modifier *product-moment* in the name.	$r = \frac{n\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2]\,[n\sum y^2 - (\sum y)^2]}}$

Precision	**Precision** (also called positive predictive value) is the fraction of relevant instances among the retrieved instances	$precision = \frac{true\ positive}{true\ positive + false\ positive}$
Accuracy	Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right	$accuracy = \frac{\#\ correct\ predictions}{\#\ total\ predictions}$

Below, the matrix shows the number of positives, meaning number of similars for each cell line when the structural distance threshold increases.

Table 6: Number of positives for different thresholds

P-Threshold	A375	PC3	VCAP
0.15	-	42	60
0.2	259	697	1037
0.25	1190	3701	4134
0.3	4927	12489	16008

As we can see, as the threshold rises, the model finds more positives. For all these different cell lines, we choose a threshold between 0.2 and 0.25 so that the number of positives, meaning the number of similar compounds, is neither huge nor negligible. After that we run the model once again, so that the evaluation metrics for each of these cell lines based on the chosen threshold are produced. In the following matrix, the metrics for these tests can be seen, so that a first view of the performance of the model can be derived.

Table 7: Metrics values in training

Cell Line	MSE	MSE Similar	Pearson r	Precision	Accuracy	Similars
A375	0.018	0.028	0.17	0.59	0.64	821
VCAP	0.014	0.013	0.35	0.6	0.7	1126
PC3	0.016	0.02	0.26	0.56	0.66	1231

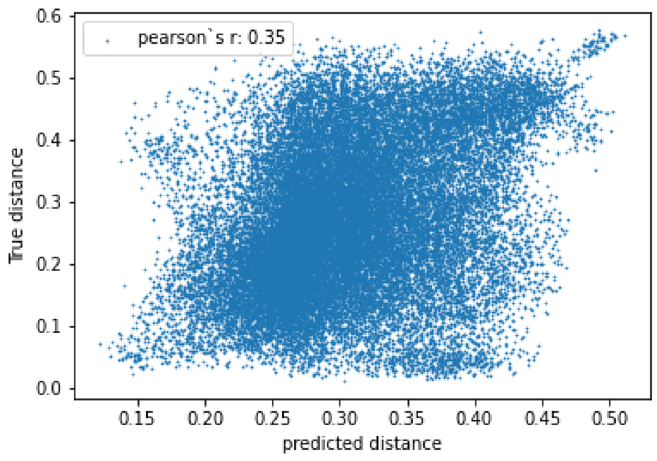


Figure 19: VCAP predicted-true values

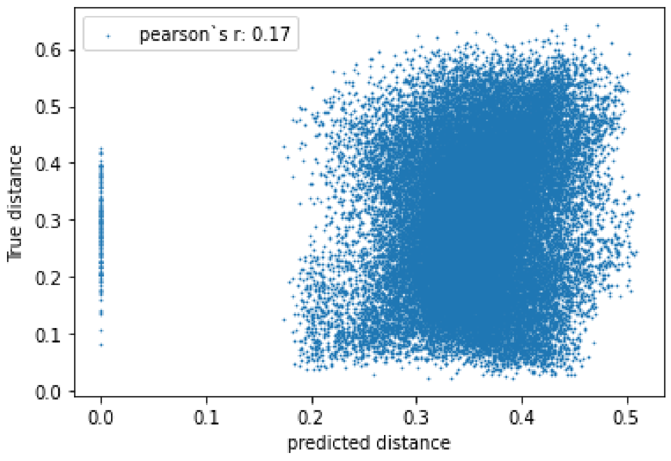


Figure 20: A375 predicted-true values

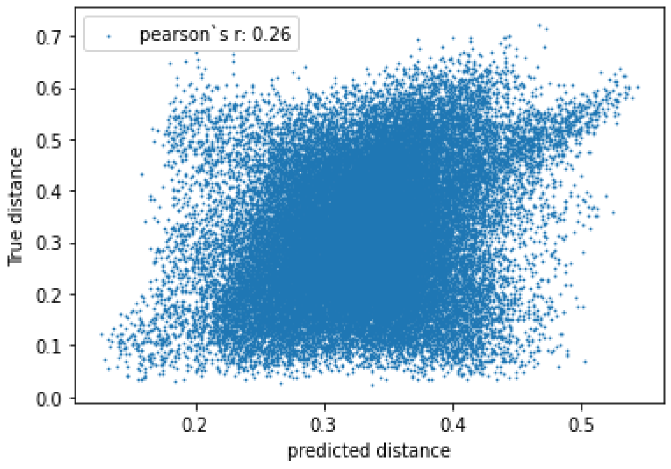


Figure 21: PC3 predicted-true values

As we can see, the training of this data is not very effective. Pearson's r value is low in all 3 different cell lines, meaning that there is a positive correlation between the predicted values and the true ones, however this correlation cannot lead to the conclusion that the training was very effective. For the VCAP cell line, pearson's r is 0.35 which shows a significant positive correlation which however is not above 0.5 that would be the threshold showing that the predicted data actually approaches the true values.

Precision is around 60% for all three different cell lines. This number shows that 60% of the predicted positive values (similar) are similar and 40% of them are not. This value shows that the model proved an ability to correctly identify whether two compounds have a close distance (meaning close embeddings and therefore similar signaling networks). However, for the training to be characterized robust this number should be higher. The same applies for accuracy.

Finally, the MSE value for all 3 different cell lines varies from 0.01 to 0.02 and therefore is very low meaning that the average set of errors is very low and therefore is a positive index for the training.

Following that first analysis, we will try to maximize the two most important metrics MSE and precision by altering the CV threshold.

Quantifying predictive uncertainty can lead to more accurate results in virtual screening applications. For this reason, we investigated the relationship between the uncertainty estimate and the performance of the model. Our model estimates predictive uncertainty as the coefficient of variation (CV) of the mixture of each model's Gaussian in the ensemble. MSE and precision were calculated for samples in the test set, with CV lower than an increasing threshold

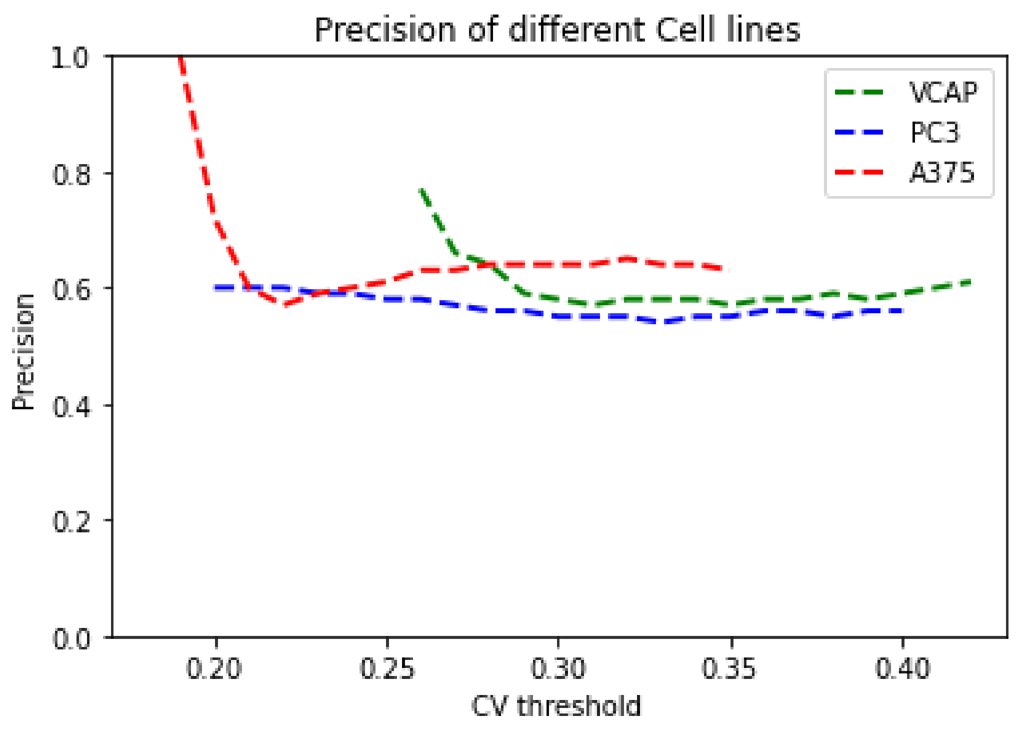


Figure 22: Precision - CV threshold for different cell lines

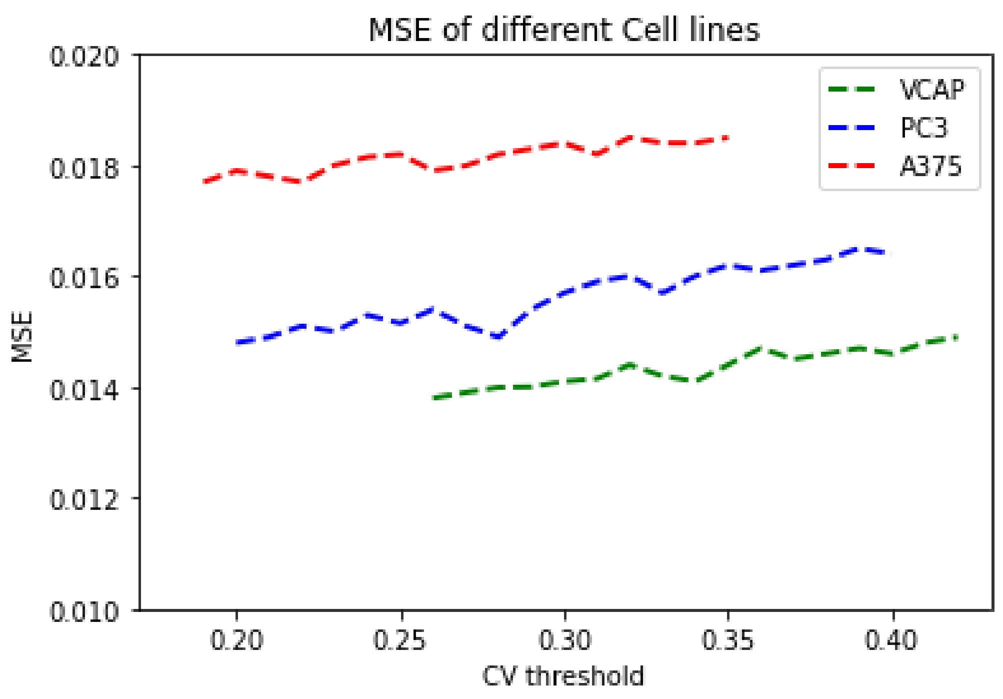


Figure 23: MSE - CV threshold for different cell lines

As we can see for both metrics the values remain the same for the most part. Precision starts with higher values in PC3 and A375 cell lines because for smaller CV threshold, the model finds less positives (similars) and therefore it is easier for the precision to be higher. Due to the low number of false positives for all the models, precision is largely unaffected by the CV threshold. MSE remains very low for all three cell lines and is slowly increasing for each cell line. As the CV threshold increases and more samples with higher CV are included in the evaluation, the MSE of the models increases as well. This implies that point predictions with lower uncertainty are closer to the target value. The conclusion is that for our problem the important metrics do not change much when the CV threshold changes.

Overall, the conclusion is that all metrics have values that indicate that the data is somehow trained using the deepsiba model, however the values are not good enough to prove that the training is very effective and robust. This mostly happens because deepsiba was constructed for a different problem and therefore the architecture, hyperparameters and structure are not appropriate enough for the data that we integrated. The fact that despite these limitations the data is trained shows that inferring a protein signaling network from a compound's chemical structure is a possible prospect.

7. Conclusion

The purpose of this thesis was to construct a deep learning model that would be able to extract embeddings from the signaling networks that would describe them effectively and separate them based on the signature id of the compound that caused them. The main idea was to use graph neural networks, which is a method not usually used in such problems. For this purpose, a very robust and state of the art architecture, GIN Layer, was used, along with other common deep learning techniques and architectures, such as simple dense layers, attention layers, projection models etc. The whole concept was to create an effective and ideally better alternative to a solution given to this problem by a previous lab member. It was proved that our model, despite possible flaws of the data available, was capable of providing embeddings of very good quality, even better to compared to previous methods and solutions. By succeeding in the 3 significant tasks our model proved to create very good embeddings that could effectively separate the experiments with same signature ID from the ones with different ones as well as the duplicate embeddings from the different ones. Therefore, the primary purpose of the thesis was fulfilled. Following this, the secondary purpose of the thesis was to infer a protein signaling network from a compound's chemical structure. Using the embeddings created by our model, and the deepSIBA model the goal was to train the model to predict whether two compounds have similar signaling networks based only on their chemical structures. For this purpose, pairwise distances were calculated between the embeddings and these distances along with their corresponding compounds were integrated to deepSIBA. The conclusion was that this task was not easy, since the deepSIBA model was constructed for a different purpose. However, the model managed to provide some valuable information and a first sufficient approach to this problem, since the model managed to train a proportion of the data effectively.

8. Limitations and Further Research

To begin with, using GNNs to process signaling networks is an approach that is not largely developed or tested, and therefore there is not enough research or bibliography to refer to. The primary focus of the model was to "reduce" the size of biological signaling networks to a single representation so that our methods can be used to find a drug's mechanism of action. Our model passed the test and provided embeddings of good quality. However, a significant limitation and potential problem was limited amount and questionable biological completeness and validity of our data. This did not allow a high degree of predictive confidence in further evaluation tests. Besides, the input data (signaling networks) were heavily based on the public Protein to Protein Interaction network and the hyperparameters of CARNIVAL. Even if our model was complete and constructed of the best graph neural network architectures, it's effectivity and robustness is doubtful. Specifically, if the input data are incomplete, the results will be even more incomplete. Furthermore, even though CARNIVAL is a great and very useful tool, was not designed for large amounts of signaling networks and therefore it's performance might have been affected. Finally, it is important to mention that the mediocre results of the deepSIBA training happened because deepSIBA was not designed for this problem. For better results, we should have optimized the hyperparameters and potentially the architecture.

For further research, there are multiple potential alternatives to further analyze this problem. To begin with, using the encoder of deepSIBA we could encode the chemical structures of drugs into vectors and train them with the mutual information method. Then, we could construct a final neural network that would take as inputs both the vector of our model and of the deepSIBA and train based on both. The long-term goal of this is using a signaling network as reference one could use the model to and the model would propose some potential drugs that could create this signaling network, and most importantly vice versa by using a chemical structure as reference.

Another potential step would be to create similar models as the one proposed in this thesis, to create vectors not only for the signaling networks of a drug but also for the genes, transcriptional factors and GO terms. Then by concatenating them and integrating them to an encoder we could have a unique vector for each drug that based on the 4 previous characteristics it would be robust and contain very important information on the mechanism of action of each drug.

Finally, we could train the deepSIBA model for GO terms (already exists) , genes , and transcriptional factors. Therefore, the model would be able to propose similar drugs to the one that was used as the input in the level of GO terms, genes and transcriptional factors. After finding all the similar drugs, it would be very intriguing and challenging to use the signaling networks of these drugs and try to create an optimization model that would propose a specific signaling network.

9. Bibliography

1. Medline Plus. "What Are Proteins and What Do They Do?" *Medlineplus.gov*, 26 Mar. 2021, medlineplus.gov/genetics/understanding/howgeneswork/protein/.
2. Jordan JD, Landau EM, Iyengar R. Signaling networks: the origins of cellular multitasking. Cell. 2000 Oct 13;103(2):193-200. doi: 10.1016/s0092-8674(00)00112-4. PMID: 11057893; PMCID: PMC3619409.
3. Pawson T. Protein modules and signalling networks. Nature. 1995 Feb 16;373(6515):573-80. doi: 10.1038/373573a0. PMID: 7531822.
4. IBM Cloud Education. "What Are Neural Networks?" *Www.ibm.com*, IBM, 17 Aug. 2020, www.ibm.com/cloud/learn/neural-networks.
5. Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444. doi:10.1038/nature14539
6. Thomas N. Kipf, Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2016. arxiv:1609.02907 [cs.LG]
7. Mayachita, Inneke. "Understanding Graph Convolutional Networks for Node Classification." *Medium*, 18 Aug. 2020, towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b.
8. Kipf, Thomas. "How Powerful Are Graph Convolutional Networks?" *Tkipf.github.io*, tkipf.github.io/graph-convolutional-networks/.
9. Zhang, S., Tong, H., Xu, J., & Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. In Computational Social Networks (Vol. 6, Issue 1). Springer Science and Business Media LLC. https://doi.org/10.1186/s40649-019-0069-y
10. Dertat, Arden. "Applied Deep Learning - Part 3: Autoencoders." *Medium*, Towards Data Science, 3 Oct. 2017, towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798.

11. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. In Neurocomputing (Vol. 234, pp. 11–26). Elsevier BV. https://doi.org/10.1016/j.neucom.2016.12.038

12. Keyulu Xu, Weihua Hu, Jure Leskovec, Stefanie Jegelka. *How Powerful are Graph Neural Networks*. 2018. arXiv:1810.00826[cs.LG]

13. "NCI Dictionary of Cancer Terms - National Cancer Institute." *Www.cancer.gov*, 2 Feb. 2011, www.cancer.gov/publications/dictionaries/cancer-terms/def/mechanism-of-action.

14. Heinzinger, M., Elnaggar, A., Wang, Y. *et al.* Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics* **20,** 723 (2019). https://doi.org/10.1186/s12859-019-3220-8

15. Weisfeiler and Lehman, A.A. (1968) A Reduction of a Graph to a Canonical Form and an Algebra Arising during This Reduction. Nauchno-Technicheskaya Informatsia, 9.

16. Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. 2016.arXiv: 1606.00709 [stat.ML].

17. Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. 2018. arXiv: 1807.03748 [cs.LG].

18. Aravind Subramanian et al. _A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles. In: Cell 171.6 (Nov. 2017), 14371452.e17. doi: 10.1016/j.cell.2017.10.049.

19. Anika Liu et al. From expression footprints to causal pathways: contextualizing large signaling networks with CARNIVAL. In: npj Systems Biology and Applications (2019). doi: 10.1038/s41540-019-0118-z.

20. F. Jordan, T.-P. Nguyen, and W.-c. Liu. Studying protein-protein interaction networks: a systems view on diseases. In: Briefings in Functional Genomics 11.6 (Aug. 2012), pp. 497504. doi: 10.1093/bfgp/els035.

21. Erick Moen et al. Deep learning for cellular image analysis. In: Nature Methods 16.12 (May 2019), pp. 12331246. doi: 10.1038/s41592-019-0403-1.

22. "A Beginner's Guide to Using Attention Layer in Neural Networks." *Analytics India Magazine*,4 Dec. 2021, analyticsindiamag.com/a-beginners-guide-to-using-attention-layer-in-neural-networks/.

23. Jason Brownlee. "A Gentle Introduction to Generative Adversarial Networks (GANs)." *Machine Learning Mastery*, 16 June 2019, machinelearningmastery.com/what-are-generative-adversarial-networks-gans/.

24. Lim S, Lu Y, Cho CY, Sung I, Kim J, Kim Y, Park S, Kim S. A review on compound-protein interaction prediction methods: Data, format, representation, and model. Comput Struct Biotechnol J. 2021 Mar 10;19:1541-1556. doi: 10.1016/j.csbj.2021.03.004. PMID: 33841755; PMCID: PMC8008185.

25. Keskin O, Tuncbag N, Gursoy A. Predicting Protein-Protein Interactions from the Molecular to the Proteome Level. Chem Rev. 2016 Apr 27;116(8):4884-909. doi: 10.1021/acs.chemrev.5b00683. Epub 2016 Apr 13. PMID: 27074302.

26. B. Lakshminarayanan, A. Pritzel and C. Blundell, Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017, 6402–6413.

27. Prabhakaran, Selva. "Cosine Similarity – Understanding the Math and How It Works (with Python Codes)." *Machine Learning Plus*, 22 Oct. 2018, www.machinelearningplus.com/nlp/cosine-similarity/.

28. C. Fotis et al. DeepSIBA: Chemical Structure-based Inference of Biological Alterations. 2020. arXiv: 2004.01028 [q-bio.QM].

29. Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. In Neural Networks (Vol. 2, Issue 5, pp. 359–366). Elsevier BV. https://doi.org/10.1016/0893-6080(89)90020-8

30. Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. In Neural Networks (Vol. 4, Issue 2, pp. 251–257). Elsevier BV. https://doi.org/10.1016/0893-6080(91)90009-t

31. Batina, L., Gierlichs, B., Prouff, E., Rivain, M., Standaert, F.-X., & Veyrat-Charvillon, N. (2010). Mutual Information Analysis: a Comprehensive Study. In Journal of Cryptology (Vol. 24, Issue 2, pp. 269–291). Springer Science and Business Media LLC. https://doi.org/10.1007/s00145-010-9084-8

32. Fotis, C., Meimetis, N., Sardis, A., & Alexopoulos, L. G. (2021). DeepSIBA: chemical structure-based inference of biological alterations using deep learning. In Molecular Omics (Vol. 17, Issue 1, pp. 108–120). Royal Society of Chemistry (RSC). https://doi.org/10.1039/d0mo00129e

33. “What Is Systems Biology · Institute for Systems Biology.” Institute for Systems Biology, 2015, isbscience.org/about/what-is-systems-biology/.

34. Sanchez-Lengeling, Benjamin, et al. “A Gentle Introduction to Graph Neural Networks.” Distill, vol. 6, no. 8, 17 Aug. 2021, 10.23915/distill.00033.