UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

THOMAS VAITSES FONTANARI

Investigating Pooling in Graph Neural Networks for Cancer Genomics Classification and the Generalizability of Pan-cancer Models to Cancer-Specific Predictions

Thesis presented in partial fulfillment of the requirements for the degree of Master of Computer Science

Advisor: Prof. Dra. Mariana Recamonde-Mendoza

Porto Alegre
December 2023

CIP — CATALOGING-IN-PUBLICATION

Vaitses Fontanari, Thomas

Investigating Pooling in Graph Neural Networks for Cancer Genomics Classification and the Generalizability of Pan-cancer Models to Cancer-Specific Predictions / Thomas Vaitses Fontanari. – Porto Alegre: PPGC da UFRGS, 2023.

109 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2023. Advisor: Mariana Recamonde-Mendoza.

1. Cancer genomics. 2. GNN. 3. Pooling. 4. Interpretabiliy. 5. Few-shot learning. I. Recamonde-Mendoza, Mariana. II. Título.

AGRADECIMENTOS

Fazer um mestrado é difícil e exige apoio e participação de muita gente, por isso certamente não conseguirei citar todos que poderiam estar listados aqui. Entre as entidades e pessoas que consigo me lembrar agora, sou grato ao Instituto de Informática (INF), ao Programa de Pós-Graduação em Computação (PPGC) da UFRGS e toda a estrutura que ele oferece, apesar de todas as dificuldades. Entre essa estrutura, sou grato ao GPPD e aos estudantes do grupo, que me ajudaram bastante com o uso do servidor, e aos professores e corpo técnico, que são fundamentais em todas as etapas desse processo.

Agradeço em particular à minha orientadora, Prof. Mariana Recamonde-Mendoza, pelo apoio e disponibilidade durante esse processo. Também devo aos outros estudantes do grupo com quem trabalhei durante o mestrado, que tornam esse processo muito mais rico e agradável do que se fosse feito completamente sozinho.

ABSTRACT

New sequencing technologies have lead to a massive generation of gene expression data, enabling the analysis and modeling of the genomic aspects of critical diseases, such as cancers. In this context, machine learning (ML) models are of fundamental importance, as they can help physicians in clinical settings and also in the identification of biological markers than can lead to the discovery of new therapies. However, it is difficult to model gene expression data due to the general lack of samples of rarer diseases. This has prompted the proposal of various ML models that can work with genomic data and, in particular, to model cancer samples. Among these, different neural network architectures have been developed, and a few recent works have proposed the use of graph neural networks (GNNs) to embed prior biological knowledge into the models. In general, however, these works have not presented any exploration of the pooling operation, which is a significant aspect of graph-level classification with GNNs. Therefore, a major part of this dissertation is devoted to analyzing how pooling and clustering an input biological network impacts the performance of the GNNs in the context of cancer genomics classification tasks. Among our results, we found that multiple coarsening levels of the graph has a general negative impact in the performance, but that this can be partially circumvented when weighted pooling and graph convolutions are used. We also show that these models lead to significant genes when they are interpreted using gradient-based methods, many of which have been previously studied in the context of cancers and cancer therapies. Furthermore, we also interpreted the models at the coarser levels of the graphs generated through the pooling operations, and found that the related clusters of genes were often over-represented in biological processes associated with cancer. As a byproduct of our experiments, we observed that the pan-cancer models achieved a high performance in comparison with cancer-specific ones. Because of that, we also explored in this work how the introduction of samples from different cohorts could improve the results on cohort-specific tasks, focusing only on traditional neural networks, as baseline in this domain. Our results indicated that the use of out-of-cohort samples reduces the variance of the cohort-specific models, improving their performance, and is most beneficial when the dataset is small and class-imbalanced. Finally, we also show that it is possible to have good performance on cohort-specific tasks on cohorts that were not seeing during training.

Keywords: Cancer genomics. GNN. Pooling. Interpretabiliy. Few-shot learning.

Pooling em Redes Neurais em Grafos para Classificação em Genônimca do Câncer e Generalização de Modelos Pan-cancer para Tipos Específicos

RESUMO

Novas tecnologias de sequenciamento levaram à geração massiva de dados de expressão gênica, possibilitando a análise e modelagem dos aspectos genômicos de doenças críticas, como o câncer. Nesse contexto, modelos de aprendizado de máquina (AM) são de fundamental importância, pois podem auxiliar médicos em ambientes clínicos e também na identificação de marcadores biológicos que podem levar à descoberta de novas terapias. No entanto, a alta dimensionalidade e não-linearidade desses dados, aliada à baixa disponibilidade de exemplos, especialmente para tipos mais raros de cânceres, dificulta a sua análise. Esses fatores levaram a propostas de vários modelos de AM que poderiam trabalhar com dados de genômicos de câncer. Dentre esses, diferentes arquiteturas de redes neurais foram desenvolvidas, e alguns trabalhos recentes propuseram o uso de redes neurais de grafo (GNN) para incorporar redes biológicas prévias aos modelos. De forma geral, no entanto, esses trabalhos não exploraram de maneira mais aprofundada a etapa de *pooling*, fundamental na classificação no nível do grafo quando são usadas as GNNs. Assim, uma parte importante dessa dissertação é dedicada a analisar como o *pooling*, baseado no agrupamento hierárquico dos nodos da rede biológica de entrada, impacta no desempenho das GNNs nas tarefas de classificação com dados genômicos de câncer. Entre nossos resultados, descobrimos que múltiplos níveis de agrupamento do grafo têm um impacto geral negativo no desempenho, mas que isso pode ser parcialmente contornado quando o *pooling* com pesos e as convoluções de grafo são usadas. Mostramos também que esses modelos levam a genes significativos quando são interpretados usando métodos baseados em gradientes, muitos dos quais foram estudados anteriormente no contexto de cânceres e terapias contra o câncer. Além disso, interpretamos os modelos nos níveis de menor resolução dos grafos, gerados por meio das operações de agrupamento, e descobrimos que os supernodos, relacionados aos agrupamentos de genes no grafo de entrada, estão frequentemente super-representados em processos biológicos associados a câncer. Como subproduto de nossos experimentos, observamos que os modelos pan-câncer alcançaram alto desempenho em comparação com os específicos para o câncer. Por causa disso, também exploramos neste trabalho como a inclusão de amostras de diferentes tipos de cânceres poderia melhorar os resultados em tarefas de classificação para grupos específicos, focando apenas nas redes neurais tradicionais. Nossos resultados indicaram que a inclusão de amostras de outros tipos de cäncer reduz a variância dos modelos, melhorando seu desempenho, e é mais benéfica quando o conjunto de dados é pequeno e desequilibrado. Finalmente, também mostramos que é possível obter um bom desempenho em tarefas com dados de tipos de câncer que não foram observados no treinamento.

Palavras-chave: genômica do câncer, *GNNs*, *pooling*, interpretabilidade, *few-shot learning*.

LIST OF ABBREVIATIONS AND ACRONYMS

BLCA	Bladder Urothelial Carcinoma
BRCA	Breast Invasive Carcinoma
cDNA	Complementary Deoxyribonucleic Acid
CNN	Convolutional Neural Network
CNV	Copy Number Variation
COAD	Colon Adenocarcinoma
DNA	Deoxyribonucleic Acid
ENSG	Ensembl Gene
ENSP	Ensembl Peptide
ESCA	Esophageal Carcinoma
FC	Fully-connected
FPKM	Fragments Per Kilobase of transcript per Million mapped reads
FPKM-UQ	FPKM Upper Quartile
FSL	Few-shot Learning
GCN	Graph Convolutional Network
GDC	Genomic Data Commons
GEO	Gene Expression Omnibus
GNN	Graph Neural Network
GO	Gene Ontology
HNSC	Head and Neck Squamous Cell Carcinoma
KEGG	Kyoto Encyclopedia of Genes and Genomes
KICH	Kidney Chromophobe
KIRC	Kidney Renal Clear Cell Carcinoma
KIRP	Kidney Renal Papillary Cell Carcinoma

LIHC	Liver Hepatocellular Carcinoma
LUAD	Lung Adenocarcinoma
LUSC	Lung Squamous Cell Carcinoma
MAE	Multi-attention Ensemble
MAP	Maximum a posteriori
miRNA	Micro Ribonucleic Acid
ML	Machine learning
MLP	Multi-layer Perceptron
mRNA	Messenger Ribonucleic acid
NCI	National Cancer Institute
NN	Neural Network
ORA	Over-representation Analysis
PPI	Protein-protein Interaction
PRAD	Prostate Adenocarcinoma
READ	Rectum Adenocarcinoma
RNA	Ribonucleic Acid
scRNA-seq	single-cell RNA-seq
SGD	Stochastic Gradient Descent
SOTA	State-of-the-art
STAD	Stomach Adenocarcinoma
TCGA	The Cancer Genome Atlas
THCA	Thyroid Carcinoma
UCEC	Uterine Corpus Endometrial Carcinoma

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Recent decades have seen the emergence of genome sequencing technologies such as microarry and RNA-seq that lead to a massive generation of gene expression data. The analysis of this data enables one to understand multiple aspects of various diseases at a molecular level, such as its progression and its genetic markers. In particular, much research has been focused towards studying the genomic aspects of cancer, as it is one of the leading causes of death worldwide. Furthermore, as is explained by Mostavi et al. (2021), cancer is a complex disease, with hundreds of types and subtypes, and its development is dependent on its location, stage and also on the patient genotype itself, all of which can affect its genetic signature. Those characteristics make the genomic study of cancer a fertile and relevant subject.

Besides enabling an exploration of the genetic and molecular characteristics of cancer, gene expression data also allows the development of models that can support clinical settings in multiple ways. For example, machine learning (ML) methods have been deployed to develop models that are able to differentiate cancerous tissues from normal ones and to give prognosis of the disease in individual patients (KOUROU et al., 2015). Insight into the biological aspects of the disease can also be obtained from these models, as long as they can be interpreted to identify important gene biomarkers. As a further motivating factor, the recent concepts of personalized medicine and oncology aim at eventually enabling the selection and development of treatments that are specific to the very particular type of cancer that a patient has (CHIU et al., 2020), which is dependent on the design of robust and interpretable gene expression-based models.

It is often argued, however, that modeling gene expression data is difficult, mostly due to the lack of samples in comparison with the dimensionality of the data and the generally non-linear patterns that underlie it (MOSTAVI et al., 2021; HANCZAR; BOURGEAIS; ZEHRAOUI, 2022; SáNCHEZ; GARCíA, 2018). For example, while a large collection of cancer genomic datasets such as The Cancer Genome Atlas (TCGA) makes available expression levels for more than 20 thousand protein-coding genes, some cancer types are represented in just a few dozens of samples. This results in models with limited quality for specific types, which also hinders the quality of the biomarkers that can be extracted from them. Following the popularity of deep learning models in a diverse range of areas, many works have also started proposing neural networks (NN) in the context of genomics, as an alternative to other ML strategies. Besides their popularity, NNs were also considered attractive due to their ability to automatically learn the relevant features and to represent non-linear relations (HANCZAR; BOURGEAIS; ZEHRAOUI, 2022). As argued by Hanczar, Bourgeais and Zehraoui (2022), NNs indeed seem to be the state-of-the-art in cancer genomic classification tasks, but only as long as a sufficient number of samples are available.

Recently, a few works have started proposing the use of graph neural networks (GNNs), instead of traditional NNs, to embed knowledge of gene interactions in the models in the hope that more accurate and robust classifiers could be produced. GNNs produce embeddings for each gene by mixing (in often complex and non-linear ways) its original expression value with that of the gene neighboring genes. The fundamental idea for justifying GNNs here is based on concepts coming from network medicine (BARABáSI; GULBAHCE; LOSCALZO, 2011), where it is known that connected molecular elements in a biological network interact, and if an element of the network is related to a disease, it is more likely that its neighbors will also be. Following that, diseases will be associated with clusters of genes in the biological network. Therefore, by mixing the expressions of close-by genes we can produce more robust estimates of the expressions, and perhaps also identify local patterns in their interactions.

When one is working with gene expression classification, however, the task of interest is often at the graph level. That is, one is interested in obtaining a classification for the entire graph. For example, given the set of gene expression values obtained from a sample, we want to classify whether the sample is tumorous or not. In the graph, each gene is associated with a node, and we wish to obtain a classification that takes all genes into account, and hence the entire graph. Therefore, the node embeddings need to be summarized into a single representation that can then be used to provide a classification for the entire network, in a process that is often done through *pooling* (GRATTAROLA et al., 2022). The resulting embedding can then be used as input to a fully-connected network to produce a model that can be trained end-to-end using gradient descent.

The simplest way to perform pooling consists in applying a global mean or max over all the node embeddings, at the risk of loosing a great amount of information in process. At the other extreme, one could concatenate all the embeddings into a single embedding with an even greater dimensionality than the given input (if each embedding has 64 dimensions, and we are dealing with 20 thousand genes, the final graph-level embedding would have more than a million dimensions). Although many works propose and evaluate different pooling techniques, this step in graph-level classification often does not receive significant attention in the context of genomics data, where it is often done in a mid-way between concatenation and global pooling.

The major part of this dissertation, therefore, is dedicated to exploring the application of pooling over the biological network in the construction of the GNN models. Specifically, we consider here the tasks of pan-cancer cohort classification and of tumor prediction, which consist, respectively, in differentiating the tissue-of-origin of a sample and in predicting whether a sample comes from a normal or tumorous tissue. Previously, other works had found that a single graph convolution, with a single pooling layer, was able to improve performance of GNN models due to a better noise handling obtained by taking the expression values of neighborhood nodes into account (YIN et al., 2022; LI; WANG; NABAVI, 2021). Because of that, we wondered whether further coarsening the input graph could introduce more gains. This lead us to construct hierarchical clusters of the widely used STRING (SZKLARCZYK et al., 2019) biological network and use them as basis for exploring different forms of pooling and a varying number of pooling levels.

In medical situations, the value of a ML model is greatly increased if its outputs can be explained, so that one is able to understand, for example, what were the most important features that lead the model to a particular classification. In the context of cancer genomics, having models that can be interpreted is also fundamental to allow further understanding of the disease and to enable the extraction of biomarkers that can be used for the development of specific treatments. Therefore, we followed other works (MOSTAVI et al., 2020; LYU; HAQUE, 2018) and also employed gradient-based methods to analyze our model's decisions. Additionally, we considered an interpretation at the level of the embeddings produced at the nodes in the coarser levels of the graph. Since these *supernodes* are associated with clusters of genes at the original graph, interpreting their outputs allow a more abstract interpretation of the model. For example, we found that the clusters obtained by applying hierarchical clustering algorithms over the STRING are generally significantly correlated with biological processes. Hence, by considering the model's attributed importancies to the supernodes, one can obtain interpretations at the level of the biological processes.

As a byproduct of our experiments with the GNNs, we observed that the pan-cancer models developed for distinguishing tumorous from normal samples were quite accurate, even though they were trained to classify samples from all cohorts simultaneously. Furthermore, when we studied the most significant genes according to a gradient-based analysis of the model, we found that many of them were associated with various tumor from different tissues of origin, instead of just one. As mentioned previously, the small sample sizes of the genomic datasets is often cited as a major difficult in the development of models. Hence, we followed through with two more questions. First, we wanted to evaluate whether the pan-cancer tumor prediction models would perform better than the cohort-specific models when considering samples coming only from the specific cohort. In other words, can the introduction of samples from other, different cohorts, improve the performance on tumor prediction tasks of a specific cohort? Secondly, we evaluated whether the pan-cancer models were able to distinguish tumorous from normal samples belonging to cohorts that were not seen during training. This question was inspired by recent works on few-shot learning models for genomic tasks (MOSTAVI et al., 2021) and on the fact that the interpretation of the trained models revealed genes significant for a great number of cancers. Both of these tasks fall broadly in the concept of few-shot learning (FSL) (WANG et al., 2020) and are studied here as ways in which we can improve the performance of genomic models on smaller datasets.

The main contributions of this dissertation are summarized below.

1. We show how the performance of GNN models on cancer genomic classification tasks is affected by the various ways in which the input graph can be coarsened (Chapter 5).
2. We show that these models can potentially be used for obtaining meaningful tumor marker genes, and to provide interpretations at the level of the biological processes (Section 5.4).
3. Finally, we demonstrate that the introduction of samples form different cohorts can improve the performance of models on cohort-specific tasks, specially for smaller and imbalanced sets, and that pan-cancer models are able to generalize to unseen cohorts as well (Section 6.2 and 6.1).

Furthermore, Chapter 2 provides a computational and biological background for this dissertation, and, in Chapter 3, we review the main NN-based models that were developed in the last years for cohort and tumor prediction tasks based on genomic data, including convolutional neural networks (CNNs) and GNNs. Finally, Chapter 7 concludes the work and mentions various possibilities of future work that we were not able to explore here.

2 BACKGROUND

A major part of our work deals with the development and interpretation of graph neural network models over biological networks and genomic data. We therefore require an understanding of these concepts before our specific methods and experiments are shown. The next sections review these topics, starting from the Computational Background in Section 2.1 and finalizing with the Biological Background in Section 2.2.

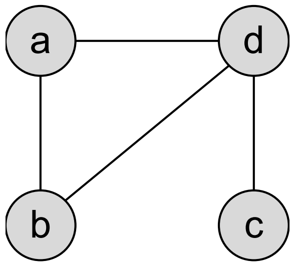
2.1 Computational Background

This work builds primarily on the concepts of neural networks and graph neural networks, which are reviewed in the next sections.

2.1.1 Graphs

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ can be formally defined by a set of nodes, or vertices, $\mathcal{V}$ and a set of edges $\mathcal{E}$ connecting them (HAMILTON, 2020). Each edge in $\mathcal{E}$ going from a node $u \in \mathcal{V}$ to $v \in \mathcal{V}$ is represented as a tuple (u, v). In this work, we will generally consider only undirected graphs, that is, graphs such that the existence of an edge (u, v) imply that there exists an edge (v, u) as well, and vice-versa. Furthermore, multi-graphs (graphs containing more than one edge connecting the same two nodes) not considered here. In Figure 2.1, we show the natural, visual representation of such a graph, containing 4 nodes and 4 undirected edges.

Figure 2.1 – A simple, undirected graph.



Source: The Author

A graph can be represented as an adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$. In an adjacency matrix, each node is associated with a row and a column in the matrix, such that $A[u, v] = 1$ if $(u, v) \in \mathcal{E}$ and $A[u, v] = 0$ otherwise. Taking the graph in Figure 2.1 as an example, we could associate nodes a, b, c and d with indices 1, 2, 3 and 4, respectively, such that the resulting adjacency matrix would be given by

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}. \tag{2.1}$$

Note that this is not the only possible representation, as any other association between the nodes and indices would work. More generally, we will often work with *weighted* edges, where the entries in the adjacency matrix can be any real number instead of only 0 and 1. In these cases, it is common to refer to the matrix as an weighted adjacency matrix. Note also that undirected graphs always lead to an adjacency matrix that is symmetrical.

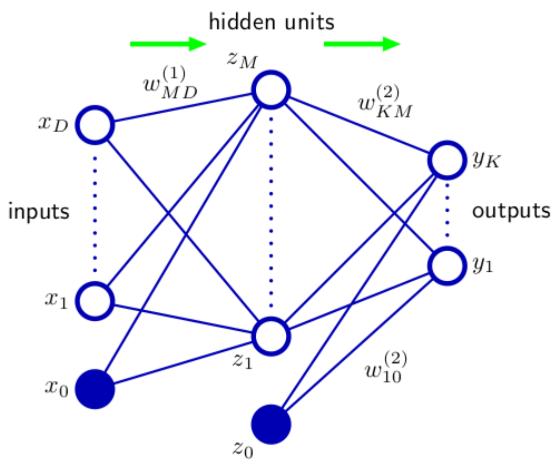
Finally, in the problems studied in this dissertation, the graphs will often have features associated with the nodes. These features can be represented in form of another matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times F}$, where F indicates the dimensionality of the features, and the order of the nodes in the adjacency matrix $\mathbf{A}$ is assumed to be consistent with the ordering in $\mathbf{X}$.

In this work, we will deal with biological graphs. These graphs, often referred to as networks as well, connect biological components together according to some criteria. For example, the nodes in a graph could correspond to genes, and the edges between each node could correspond to the strength of interaction between byproducts of these genes (such as proteins encoded by them). Furthermore, we could measure how expressed each gene is in a certain sample collected from a human tissue. Each gene expression value can then be associated with the corresponding gene, resulting in a feature matrix for the nodes of the graph given by $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times 1}$ (in this case, just a vector).

2.1.2 Neural Networks

Following Bishop (2006), in their simplest form, neural networks are a multi-layer model consisting of non-linear transformations of linear combinations of the inputs at each stage of the model. A 2-layer fully-connected neural network for a classification problem with K classes is illustrated in Figure 2.2.

Figure 2.2 – Illustration of a 2-layer fully-connected neural network.



Source: Bishop (2006)

The input $\mathbf{x} \in \mathbb{R}^D$ is forward propagated through the network. The first layer consists of a non-linear transformation of the inputs according the learnable parameters $\mathbf{w}^{(1)} \in \mathbb{R}^{M \times D}$, where M is the number of hidden units (or neurons). Each hidden neuron can be seen as different learnable transformation of the inputs. The outputs of the hidden units z_j for $j = 1, ..., M$ are given by

$$z_j = h(a_j^{(1)}), \tag{2.2}$$

where $h : \mathbb{R} \to \mathbb{R}$ is a non-linear function and the *activations* a_j are the results of the linear transformation produced by the weights $\mathbf{w}^{(1)}$ and the bias $b^{(1)}$,

$$a_j^{(1)} = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + b_j^{(1)}. \tag{2.3}$$

Note that in Figure 2.2 the bias is represented as the weights of an input feature x_0 whose value is always equals to 1. Similarly, the outputs y_k for $k = 1, ..., K$, are given by a non-linear transformation σ of the activations of layer 2,

$$y_k = \sigma(a_k^{(2)}), \tag{2.4}$$

and

$$a_k^{(2)} = \sum_{j=1}^{M} w_{kj}^{(2)} a_j^{(1)} + b_k^{(2)}. \tag{2.5}$$

In a classification task, it is desirable that the outputs can be interpreted as the probability of each class being the correct one. Therefore, one should have that $0 \leq y_k \leq 1$ and $\sum_{k=1}^{K} y_k = 1$. The softmax function is a non-linear function that guarantees these properties, so that we let

$$y_k = \frac{\exp a_k^{(2)}}{\sum_{j=1}^{K} \exp a_j^{(2)}} \tag{2.6}$$

We have considered a two-layer network in the example above, but the process of aggregating functions can be repeated indefinitely to generate neural networks with tens and even hundreds of layers.

2.1.2.1 Training Neural Networks

Neural networks can be fitted using maximum-likelihood estimation (see Appendix X). We define a dataset given by the tuples $(\mathbf{x}_n, \mathbf{t}_n)$ for $n = 1, ..., N$, where $\mathbf{x}_n \in \mathbb{R}^D$ correspond to the input examples and $\mathbf{t}_n \in \{0, 1\}^K$ gives the class of the associated example in a one-hot representation, *i.e.* $t_{kn} = 1$ if the true class of the example is class k and $t_{kn} = 0$ otherwise. We also let $y_k(\mathbf{x}, \mathbf{w})$ be the network outputs for classes $k = 1, ..., K$ given an input $\mathbf{x}$ and network parameter values $\mathbf{w}$ (note that the outputs can be interpreted as probabilities $y_k(\mathbf{x}, \mathbf{w}) = p(t_k = 1|\mathbf{x}, \mathbf{w})$, and we have included the bias in the set of weights). The error associated with the parameter values $\mathbf{w}$ is then given by the cross-entropy error function

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{kn} \ln y_k(\mathbf{x}_n, \mathbf{w}) \tag{2.7}$$

The problem of training a neural network is therefore equivalently to that of finding a set of parameters $\mathbf{w}$ than minimizes Equation 2.7. A minima of this function will be in a point such that the gradient of E with respect to the parameters vanish, i.e. $\nabla E(\mathbf{w}) = 0$. Note, however, that $E(\mathbf{w})$ will not be convex in general, so that we will typically have to settle for a local minima of the function.

In the context of neural networks, the optimization will typically be performed through gradient descent methods. In the simplest case, this consists of iteratively updating the values of the parameters. Starting with a initial set of parameters $\mathbf{w}^{(0)}$, the update consists of taking small steps in the direction of the negative gradient,

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)}). \tag{2.8}$$

The hyperparameter $\eta > 0$ is the *learning rate* and is typically optimized using a separate validation set. Intuitively, a large value of η can cause the parameters to diverge when they are close to a narrow local minima instead of progressing towards the minimum. On the other hand, if η is too small, the optimization can take too long to converge, specially if the parameters are close to saddle points, where the gradient is typically small.

Note that in Equation 2.7 the loss $E(\mathbf{w})$ is a sum of terms, each corresponding to the error associated with a single example. That is, we can write

$$E(\mathbf{w}) = -\sum_{n=1}^{N} E_n(\mathbf{w}), \tag{2.9}$$

where $E_n(\mathbf{w}) = \sum_{k=1}^{K} t_{kn} \ln y_k(\mathbf{x}_n, \mathbf{w})$. In large datasets, however, evaluating this function can be expensive, since it depends on forwarding all examples in the dataset through the network. In practice, therefore, a *stochastic* version of gradient descent is used. In the most extreme case, the parameters are updated after a single example is passed through the network,

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n(\mathbf{w}^{(\tau)}). \tag{2.10}$$

More commonly, a batch of examples is randomly sampled from the training set and forwarded through the network to obtain more stable estimates of the error function. The number of examples in each batch is referred to as *batch size* and is often treated as a hyperparameter to be tuned as well. A large batch size can produce better estimates of the error function, but will be more computationally intensive to compute and will slow down the training. On the other side, a small batch size will produce poor estimates of the error function, which can also introduce convergence problems. Common values of batch size are 32, 64, or 128. After all examples in the training set have been gone through the network one says that one *epoch* has passed. Training a network typically involves multiple epochs.

In all of the equations above regarding the update of the model parameters, it is necessary to compute the gradient of the error function with respect to the parameters. In general, this is done efficiently thorugh the use of the *backpropagation* algorithm. Additional information about backpropagation can be obtained in Bishop (2006). Finally, we note that various other gradient-based methods have been proposed to deal with different problems of the basic stochastic gradient descent (SGD) methods described here.

2.1.2.2 Regularization

A well known fact is that models with too much parameters can overfit to the training set. Such models are called *overparameterized*, and neural networks generally fall into the category. To avoid overfitting, one generally attempts to control the complexity of the model by adding regularization terms. The influence of the regularization term is controlled by the hyperparameter $\lambda > 0$, which known as the *weight decay* and is usually selected using a validation set. The hyperparameter gets its name because in sequential parameter update methods, such as gradient descent, it encourages the weights to decay towards 0. With regularization, the error becomes

$$E_T(\mathbf{w}) = E(\mathbf{w}) + \lambda E_w(\mathbf{w}), \tag{2.11}$$

where $E_w(\mathbf{w})$ is the regularization function. One of most widely used forms of regularization is L_2 regularization, which consists in using a penalty proportional to the sum of the square of the weights $\|\mathbf{w}\|_2$,

$$E_w(\mathbf{w}) = \mathbf{w}^T\mathbf{w}. \tag{2.12}$$

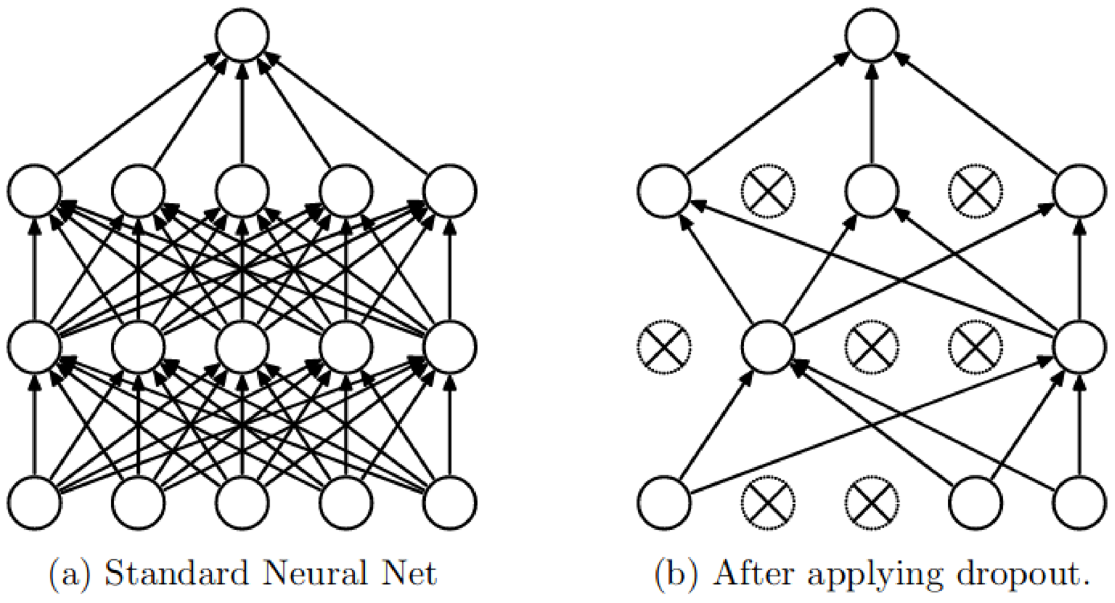
Equivalently, the regularization term is often written with a factor of a half, as in $\frac{1}{2}\lambda\mathbf{w}^T\mathbf{w}$, because it simplifies the equation when the derivative with respect to $\mathbf{w}$ is taken.

L_2 regularization also arises when we consider a maximum a posteriori (MAP) estimation of the model parameters, where the parameters are assumed to have a normal prior distribution with 0 mean.

Another widely studied regularization method is L_1 regularization. In this case, one uses the sum of the absolute values of the model parameters, $\|\mathbf{w}\|_1 = \sum_{\forall w} |w|$. At least in the context of linear models, L_1 regularization is known to induce sparsity. That is, while L_2 regularization can lead to solutions with small values of some parameters w, L_1 regularization leads to solutions where some of the values are actually 0. The typical argument for this can be found in the book by Hastie, Tibshirani and Wainwright (2015).

Finally, another common technique for regularizing neural networks is *dropout* (SRIVASTAVA et al., 2014). Dropout essentially consists of randomly ‘dropping’ units of the NN during training (Figure 2.3). In the simplest case, each neuron has a probability p of being dropped before a training example is passed through the network. The authors argue that dropout prevents overfitting and also provides a way of combining multiple net-

Figure 2.3 – Illustration of applying dropout to a standard neural network. A set of hidden units and their connections are randomly selected to be temporarily removed in the next sample.



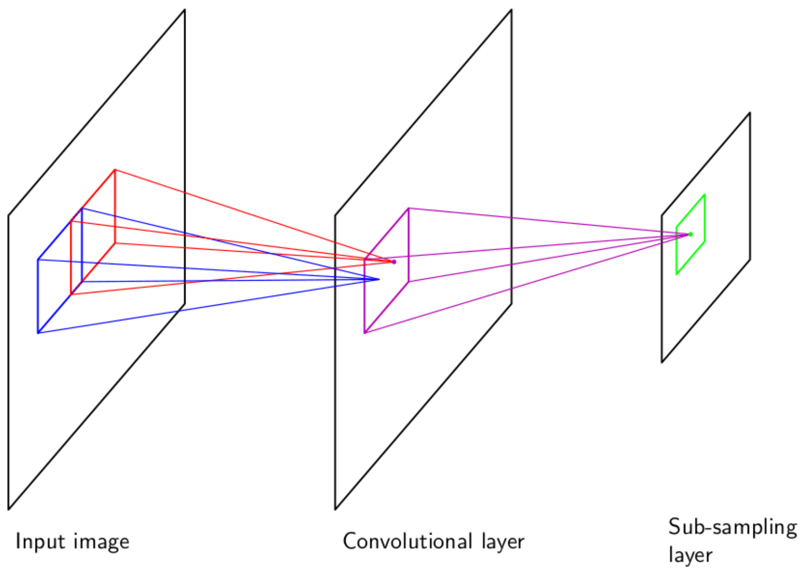
Source: Srivastava et al. (2014)

works efficiently. The reasoning is that dropping units out consists in sampling a different, sparser, neural network from the original one. If the original NN has n parameters, then there are 2^n different combinations of weights, each of which can be seen as a different NN. During test, a single neural network is used, without dropout, but with the outputs of the units that were previously dropped scaled by p. The hyperparameter p can be tuned using a separate validation set, but is often left at 0.5, which as the authors argue produce good results in general.

2.1.2.3 Convolutional Neural Networks

Convolutional neural networks (CNN) are a variant of the NNs adapted to image-like data. The CNNs are able to exploit the fact that in images nearby pixels are more likely to correlate with each other. The basis of the CNN are the convolutional and the sub-sampling layers, illustrated in Figure 2.4. In a convolutional layer, the outputs are composed of multiple feature maps. Each output unit in a feature map correspond to only a small region of the input and all the units in a feature map share the same weights. The units can be thought of as classical feature detectors in images, and the feature map correspond to applying these same detectors to various regions of the image. Because of that, the layer actually behaves like a convolution where the shared weights of the feature map

Figure 2.4 – Illustration of a convolutional neural network.



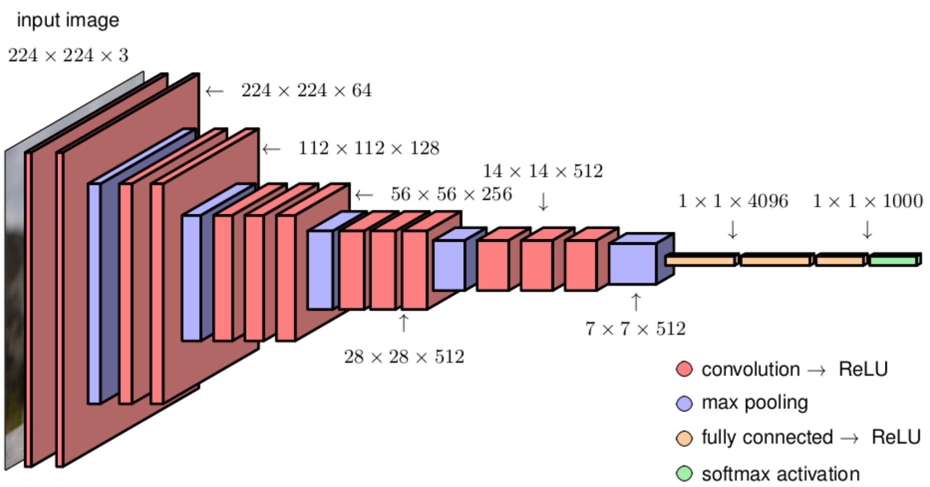
Source: Bishop (2006)

behave like the typical kernel of a convolution. Differently from a classical convolution in image processing, however, the kernel weights in a CNN are learned during training. Finally, the feature maps are typically sub-sampled by aggregating nearby regions. For example, each 2-by-2 square is pooled to generate a single value. Typical pooling functions are *max pooling*, which keeps only the maximum value in the region, and *average pooling*, which computes the average of all the units in the region. The convolutional and sub-sampling layers can be stacked to produce CNNs with multiple convolutional layers, which are able to represent higher-order filters. In general, as well, these layers will be followed by a fully-connected network to produce the final output of the model. Figure 2.5 illustrates the architecture of a CNN containing multiple convolutional layers followed by multiple fully-connected networks and a softmax activation function at the output.

2.1.3 Graph Neural Networks

Graph neural networks (GNN) consist of an adaptation of deep neural networks aiming at tackling particularities of graph data (HAMILTON, 2020), similarly to how convolutional networks are defined for image data. Currently, the preferred way of studying GNNs seems to be through the lens of the message passing framework (GILMER

Figure 2.5 – Illustration of the VGG-16 CNN architecture with 1000 output nodes.



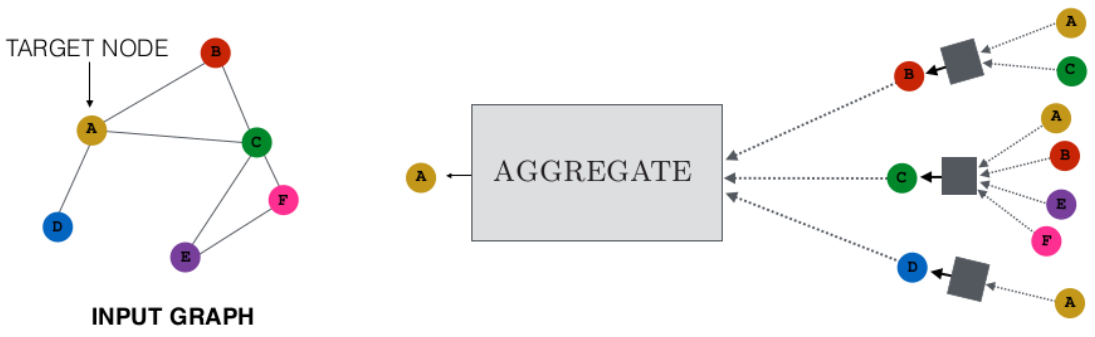
Source: Bishop (2024)

et al., 2017). Following Hamilton (2020), a message passing iteration is defined using an aggregate and an update function. At each step, the embedding of $\mathbf{h}_u$ each node u is updated according to its previous value and to the previous values of its neighbors $\mathcal{N}(u)$,

$$\mathbf{h}_u^{(k+1)} = \mathsf{UPDATE}^{(k)}(\mathbf{h}_u^{(k)}, \mathsf{AGGREGATE}^{(k)}(\{\mathbf{h}_v^{(k)}, \forall v \in \mathcal{N}(u)\})), \tag{2.13}$$

where $\mathbf{h}_u^{(k)}$ denotes the embedding of node u at step k and $\mathbf{h}_u^{(0)}$ is defined as the input features of node u. Note also that the UPDATE and AGGREGATE functions are in general dependent of the step k and that the input to AGGREGATE is a set of values, so that it is invariant to the order of the inputs. The aggregation of the neighbor's embeddings to generate the embeddings of a node is illustrated in Figure 2.6. A wide range

Figure 2.6 – Illustration of the message passing approach to a 2-layer (2 steps) GNN.



Source: Hamilton (2020)

of GNN architectures are encompassed by this approach, such as the widely used Graph Convolutional Network (GCN) (KIPF; WELLING, 2016) and the GraphSAGE (HAMILTON; YING; LESKOVEC, 2017). We note also that the idea of generating continuous representations of nodes in a graph has also been explored previously for learning task-independent representations, such as in the node2vec method (GROVER; LESKOVEC, 2016).

2.1.3.1 Spectral Graph Convolutions

In this section, we give a mathematically heavy description of the ChebConv GNN (DEFFERRARD; BRESSON; VANDERGHEYNST, 2016), since it was the mostly used graph convolutional layer in this work and that was the way in which it was developed by the original authors. Nevertheless, an in-depth understanding of the model is not entirely necessary for understanding this dissertation, and one should be able to skip this section without significant loss in the comprehension of our results and further explanations.

Besides of the message-passing approach, the GNNs were also developed as generalizations of the convolution operation to graphs (BRUNA et al., 2014; DEFFERRARD; BRESSON; VANDERGHEYNST, 2016) – hence the name *graph convolutions*. The ChebConv (Section 2.1.3.2) introduced by Defferrard, Bresson and Vandergheynst (2016), in particular, builds over the definition of the convolution in the spectral domain of the graph. The fundamental idea here is based on the fact that the convolution between two signals is equivalent to a point-wise multiplication of the spectral representation of the signals. Additionally, the spectral representation of a signal is given by the projection of the signal over the eigenfunctions of the Laplacian operator. These ideas allow one to define the convolution over a graph as the point-wise multiplication of the signals projected over the eigenvectors of the Laplacian of the graph.

It turns out that the Laplacian matrix of a weighted undirected graph with N nodes is given by (ORTEGA et al., 2018)

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{2.14}$$

where $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the symmetrical weighted adjacency matrix and $\mathbf{D}$ is a diagonal matrix with entries $(\mathbf{D})_{ii} = \sum_{j=1}^{N} (\mathbf{W})_{ij}$ and $(\mathbf{D})_{ij} = 0$ if $i \neq j$, similarly to a degree matrix. We note that the Laplacian matrix if often defined as $\mathbf{L} = \mathbf{W} - \mathbf{D}$ as well. For an intuitive justification for the graph Laplacian expression, see Appendix B. Often one also normalizes the Laplacian using the degrees of each node. Specifically, the symmetric normalized Laplacian is given by $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$.

Using the definitions of the graph Laplacian, we can have a notion of the *spectrum* of the graph, which will be given by the eigenvectors and eigenvalues of the graph Laplacian. The eigendecomposition of the Laplacian matrix is

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T, \tag{2.15}$$

where $\mathbf{\Lambda}$ is a diagonal matrix whose elements are the eigenvalues of the Laplacian matrix and $\mathbf{U}$ is a $N \times N$ matrix whose columns are given by the eigenvectors of the Laplacian matrix. Finally, the graph convolution of a signal can be obtained through its definition in the spectral domain, where it corresponds simply to a point-wise multiplication. Specifically, the graph convolution between vectors $\mathbf{f} \in \mathbb{R}^N$ and $\mathbf{h} \in \mathbb{R}^N$, both defined over the graph nodes, is

$$\mathbf{f} *_G \mathbf{h} = \mathbf{U}(\mathbf{U}^T\mathbf{f} \circ \mathbf{U}^T\mathbf{h}) = (\mathbf{U}diag(\theta_h)\mathbf{U}^T)\mathbf{f}, \tag{2.16}$$

where $diag(\theta)$ is the diagonal matrix whose diagonal elements are given by the values $\theta_h = \mathbf{U}^T\mathbf{h}$. Note, however, that this non-parametric filter θ_h is of the same dimensionality as the number of vertices in graph. Hence, in general, it does not correspond to a localized filter as we generally expect when working with convolutional networks. This issue can be handled by parameterizing θ_h as a (K-1)-th order polynomial of the eigenvalues matrix. That is, we let

$$diag(\theta_h) = p_K(\mathbf{\Lambda}) = \sum_{i=0}^{K-1} \theta_k \mathbf{\Lambda}^k, \tag{2.17}$$

where θ_k for $k = 0, ..., K-1$ are scalars. Substituting Equation 2.17 into Equation 2.16,

$$\mathbf{f} *_G \mathbf{h} = (\mathbf{U}p_K(\mathbf{\Lambda})\mathbf{U}^T)\mathbf{f} = p_K(\mathbf{L})\mathbf{f}, \tag{2.18}$$

which follows from the property that $\mathbf{A}^n = \mathbf{U_A}\mathbf{\Lambda_A}^n\mathbf{U_A}^T$, for every diagonalizable symmetric matrix $\mathbf{A}$ whose eigendecomposition is given by $\mathbf{A} = \mathbf{U_A}\mathbf{\Lambda_A}\mathbf{U_A}^T$. Hence, the convolution operation corresponds to a multiplication with a polynomial of order (K-1) over the Laplacian matrix. The fact that the Laplacian is non-zero only for non-zero entries of the adjacency matrix (and in the diagonal) implies that $\mathbf{L}^k$ will be localized in the neighborhood of the node including only neighbors at a maximum distance of k hops.

2.1.3.2 ChebConv

Defferrard, Bresson and Vandergheynst (2016) argue that the computational cost of the filtering operation can be further reduced if one parametrizes $p_K(\mathbf{\Lambda})$ using a polynomial that can be constructed recursively from $\mathbf{L}$. They propose the Chebyshev expansion as a candidate, so that Equation 2.17 becomes

$$diag(\theta_h) = \sum_{i=0}^{K-1} \theta_k \mathbf{T}_k(\tilde{\mathbf{\Lambda}}), \tag{2.19}$$

where $\mathbf{T}_k(\tilde{\mathbf{\Lambda}}) = 2\tilde{\mathbf{\Lambda}}\mathbf{T}_{k-1}(\tilde{\mathbf{\Lambda}}) - \mathbf{T}_{k-2}(\tilde{\Lambda})$, with $\mathbf{T}_0(\tilde{\mathbf{\Lambda}}) = \mathbf{I}$ and $\mathbf{T}_1(\tilde{\mathbf{\Lambda}}) = \tilde{\mathbf{\Lambda}}$. The matrix $\tilde{\mathbf{\Lambda}}$ corresponds to a scaled diagonal eigenvalue matrix given by $\tilde{\mathbf{\Lambda}} = 2\mathbf{\Lambda}/\lambda_{max} - I$, so that the elements are in the range $[-1, 1]$. In depth information on these approximations can be found in Hammond, Vandergheynst and Gribonval (2011) but is beyond the scope of this dissertation. Finally, the filtering operation can be performed using the scaled Laplacian $\tilde{\mathbf{L}} = 2\mathbf{L}/\lambda_{max} - \mathbf{I}$ and is written as

$$\mathbf{f} *_G \mathbf{h} = \sum_{k=0}^{K-1} \theta_k \mathbf{T}_k(\tilde{\mathbf{L}})\mathbf{f}, \tag{2.20}$$

where θ_k are learnable parameters.

2.1.3.3 Pytorch Geometric Implementation

The ChebConv is implemented inside Pytorch Geometric[1] as a Pytorch-compatible module. The implementation allows for the use of multiple filters in a single convolutional layer (that is, a single convolutional layer performs more than one convolution, which create various output channels), and can also be performed over multiple input channels. Let $\mathbf{X} \in \mathbb{R}^{N \times F_{in}}$ be the feature matrix associated with the nodes of a graph, where each row corresponds to a node and the columns represent the F_in input features (i.e., input channels) associated with each node. The parameters of the convolutional layer are written as $\mathbf{\Theta}^{(k)} \in \mathbb{R}^{F_{in} \times F_{out}}$, for $k \in \{0, ..., K-1\}$ with F_{out} being the number of output features (or channels) and K the number of Chebyshev coefficients (which also denotes the filter

[1]https://github.com/pyg-team/pytorch_geometric

size). The output of a ChebConv layer is then given by

$$\mathbf{X}' = \sum_{k=0}^{K-1} \mathbf{Z}^{(k)}\mathbf{\Theta}^{(k)}, \tag{2.21}$$

with $\mathbf{Z}^{(0)} = \mathbf{X}$, $\mathbf{Z}^{(1)} = \tilde{\mathbf{L}}\mathbf{X}$ and $\mathbf{Z}^{(k)} = 2\tilde{\mathbf{L}}\mathbf{Z}^{(k-1)} - \mathbf{Z}^{(k-2)}$. The matrix $\tilde{\mathbf{L}}$ is the scaled normalized Laplacian $\tilde{\mathbf{L}} = 2\mathbf{L}/\lambda_{max} - \mathbf{I}$ and we have generally used the symmetric normalized Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, where $\mathbf{W} \in \mathbb{R}^{N\times N}$ is the (weighted) adjacency matrix of the graph.

2.1.4 Pooling

The message passing framework generalized in Equation 2.13 produces node-level embeddings. However, we need an embedding of the entire graph in order to produce classifications at the graph level. Pooling node representations to construct a single graph embedding, or to generate a coarsened representation of the graph (that is, one containing fewer nodes), is often referred to as graph pooling (GRATTAROLA et al., 2022). Simple approaches to graph pooling include concatenating the embeddings of each node, as in Yin et al. (2022), and other global pooling techniques, such as computing the mean or sum of all the node embeddings. Although the global pooling techniques seem to discard a large amount of information, they are known to perform well in various settings (MESQUITA; SOUZA; KASKI, 2020). The mean and sum pooling schemes are also examples of set pooling, since these functions are invariant to the order of the nodes.

As explained in Hamilton (2020), however, these strategies do not explore the structure of the graph. Some techniques, therefore, try to cluster and coarse the nodes in order to produce hierarchical pooling schemes (e.g., DiffPool (YING et al., 2019)). The general idea is that the graph is iteratively coarsened, producing at each step a smaller graph with transformed node embeddings and different edges. The final graph embedding is then computed using one of the simpler schemes above, such as averaging or concatenating the nodes of the final graph. A complete review of the various methods for pooling graphs in graph classification is referred to Grattarola et al. (2022).

In general, a dataset will be composed of examples that do not exhibit a fixed graph. For example, in a classification task designed to predict chemical properties of a molecule, the graph of the molecules will be different for each example. Some tasks, however, present a fixed graph across all examples, similarly to how images of fixed dimensionality (e.g., 256 by 256 pixels) show a fixed 2-dimensional grid. In fact, some of the earliest graph convolutional approaches were designed with this kind of task in mind (DEFFERRARD; BRESSON; VANDERGHEYNST, 2016). Defferrard, Bresson and Vandergheynst (2016), for example, used the coarsening steps of the Graclus algorithm for graph partitioning (DHILLON; GUAN; KULIS, 2007). The idea there is to coarsen the input graph level by level, until only a small number of nodes remain. As summarized by Grattarola et al. (2022), the coarsening step of Graclus consists in halving the nodes of the graphs iteratively by combining randomly selected nodes with its closest neighbor. The definition of closest is made in such a way that the method tends to maximize objectives based on the spectrum of the graph adjacency matrix.

Another heuristic for coarsening a graph iteratively is that of heavy edge matching, which has been used in some works that used GNNs for genomic classification tasks (LI; WANG; NABAVI, 2021; WANG; BAI; NABAVI, 2021a). The methods consists simply in visiting unmatched the nodes in random order and matching each with the (unmatched) neighbor possessing the maximum edge weighted. At each matching, the graph-cut of the resulting coarser graph is smaller (KARYPIS; KUMAR, 1998). Finally, as stated by Bruna et al. (2014), one could use any hierarchical graph clustering method.

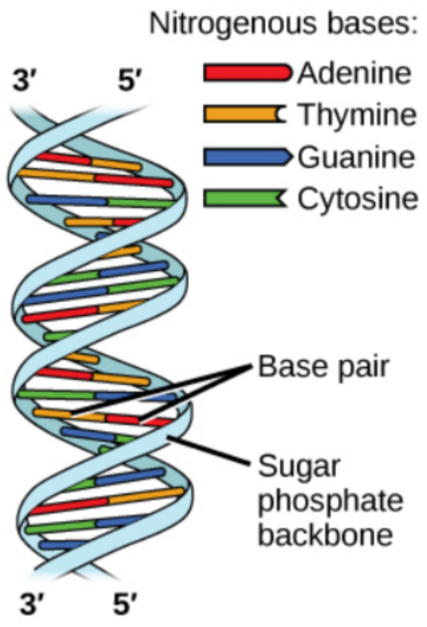
2.2 Biological Background

In this dissertation, we discuss various results and applications based on genomic data and, in particular, on gene expression data. To better appreciate it, we review below essential concepts of genomics and the technologies used to produce expression data.

2.2.1 The genome

Our genome, the collection of all the DNA sequences in our cells, determines, together with environmental and epigenetic factors, our phenotype, that is, the collection of all our visible traits (FOWLER; ROUSH; WISE, 2013). This includes not only macroscopic traits such as our height and hair color, but also microscopic ones such as deficiencies in the production of important cellular components. DNA stands for deoxyribonucleic acid and consists of a long molecule composed of two strands of 4 repeating nucleotides, Adenine (A), Thymine (T), Guanine (G) and Cytosine (C). The two strands are paired together by nucleotide base pairs, as is shown in Figure 2.7.

Figure 2.7 – Illustration of a DNA molecule. The nitrogenous bases A, C, G and T determine the nucleotide. The two DNA strands are joined together by base pairs consisting of interactions between C and G or A and T. A sugar phosphate backbone gives support to the helicoidal structure.



Source: adapted from Fowler, Roush and Wise (2013)

A major function of the genome is to encode the information necessary to build all the proteins in the cell, which in turn are responsible for performing all of its functions. The synthesis of proteins happens in two steps: transcription and translation. Transcription is the process in which sequences of the DNA are 'read' by celullar components in order to synthesize a new RNA molecule. Those sequences in the DNA that can be transcribed into functional RNAs are what we usually refer to as genes, and the transcription of the gene is what makes it *expressed*. The particular type of RNA that is synthesized in the process of generating a new protein is called a messenger RNA (mRNA) and, in the translation step, the mRNA is used as a template by organelles in the cell to synthesize the protein itself.

The genome, however, is not composed only of protein-coding genes (that is, genes that are transcribed into mRNA molecules for synthesizing proteins). In fact, this amounts to only 2-3% of the total human genome. Another part of the human genome is responsible to encode non-coding RNA molecules. These include, among others, microRNA (miRNA) and small interfering RNAs, both of which are involved in regulating the expression of genes.

Not all genes are expressed at all the times in the cell. In order to function properly, the cell controls which genes will be transcribed at each time to synthesize the needed proteins or other RNA molecules. In fact, studying the patterns of expression of genes at different times can help uncover the mechanisms of various diseases, such as cancers. Moreover, since almost all types of cell possess the entire genome in their nucleus, the differences between cells from different tissues are consequences of the genes that are expressed in them.

2.2.2 Gene Expression Data

A gene is expressed when it is transcribed into a RNA molecule. The set of all the RNA molecules that a cell contains at a certain moment is called its transcriptome, which gives rise to the term *transcriptomics*. Figure 2.8 shows a table with an example of gene expression data. The rows refer to genes, which can be coding or non-coding, and the columns are tissue samples. The *ENSG* identifiers stand for *ensembl gene* and each is associated with a specific gene, whereas the column labels with the *TCGA* prefix correspond to tissue samples collected by the The Cancer Genome Atlas (TCGA) project, a cancer genomics program jointly created by NCI and the National Human Genome Research that collected and analyzed data for a variety of cancer types. The project collected tumor and matching control samples amounting to more than 20000 samples spanning 33 types of cancer. Each element in the table is a measure of how much each gene is expressed in each sample. The measure is obtained by counting the number of RNA molecules found that are associated with the gene in the tissue and on a set of preprocessing steps performed before any machine learning method is used (Section 2.2.2.1).

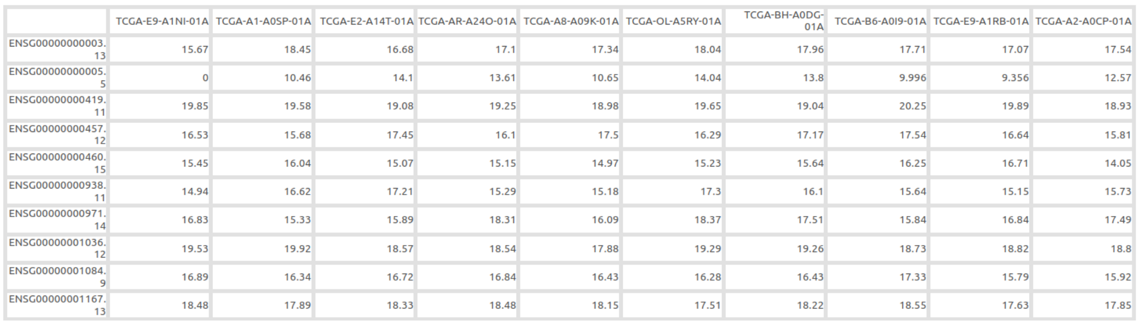
There are two major technologies that are used to count the amount of RNA molecules in a tissue: microarray and RNA-seq. Microarray is based on running multiple *hybridization* experiments simultaneously (LESK, 2012), a technique that is able to detect whether pre-determined sequences of nucleotides are present. RNA-seq, on the other hand, is based on applying high-throughput sequencing methods to the samples and does not require that a set of sequences is defined prior to the sequencing process. The RNA sequences present in the samples are fragmented into various pieces, known as reads. The reads then are aligned to a known reference genome, with more reads aligning to more expressed genes.

2.2.2.1 RNA-seq data

In this dissertation, we have worked mostly with RNA-seq gene expression data processed using the Genomic Data Commons (GDC) pipeline and downloaded through the Xena platform. GDC provides access to data generated by the National Cancer Institute (NCI) from some of the largest and most comprehensive cancer genomic datasets, such as The Cancer Genome Atlas (TCGA). The datasets provided by GDC have been processed using a common set of bioinformatics pipelines, allowing direct comparison of the data.

The general approach for obtaining gene expression data using RNA-seq is summarized in Figure 2.9 (LOWE et al., 2017). The mature mRNA molecule is generated within the organism (*in vivo* step in Figure 2.9). This corresponds to the transcription process explained previously. In general, however, the gene is not contiguous in the genome of eukaryotes, and instead is composed of interleaving *exons* (from the word expressed) and *introns* (from *interleaving*). The primary RNA transcribed contains both the introns and exons, but the introns are removed in the intron splicing step to form a mRNA molecule (FOWLER; ROUSH; WISE, 2013). After being extracted from the organism, the RNA molecule is processed *in vitro*. First, it is splitted into multiple fragments and converted into cDNA in the reverse transcription step. The conversion into cDNA is important because the DNA molecules are more stable and allow the use of more established sequencing technologies. Finally, the high-throughput sequencing methods are used to obtain the sequences of nucleotides in each of the fragments. Each fragment is

Figure 2.8 – Example of a gene expression dataset. Only the 10 first rows and columns are shown. The rows correspond to genes and the columns identify the tissue samples.

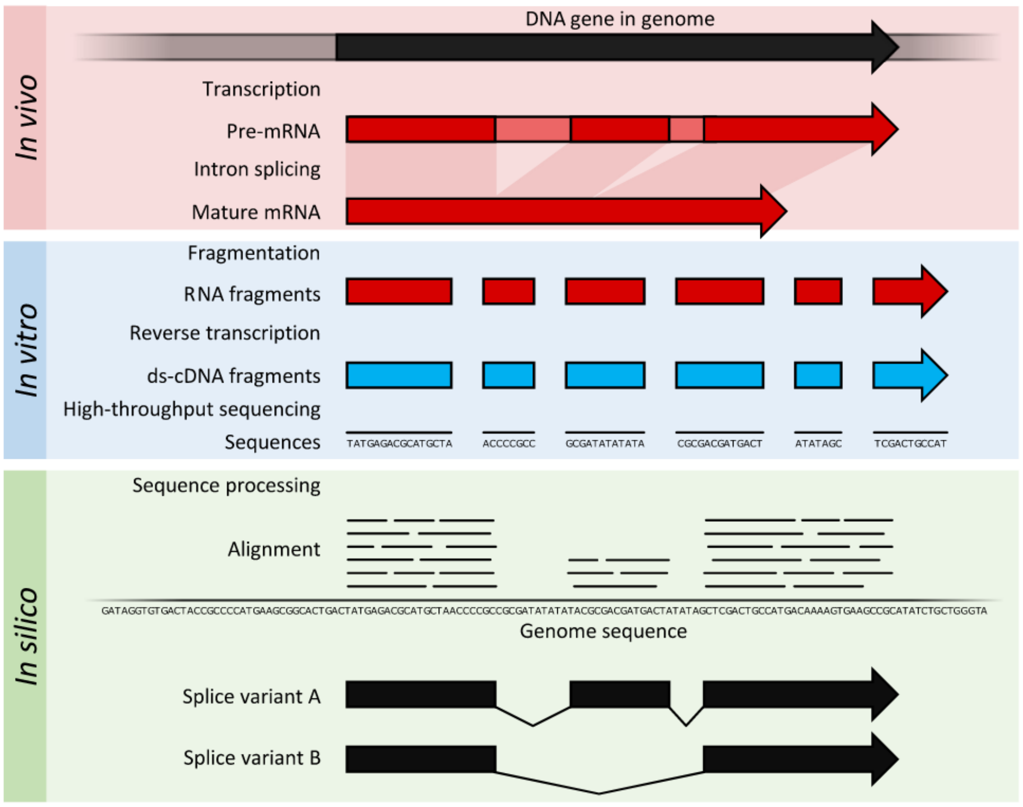


	TCGA-E9-A1NI-01A	TCGA-A1-A0SP-01A	TCGA-E2-A14T-01A	TCGA-AR-A24O-01A	TCGA-A8-A09K-01A	TCGA-OL-A5RY-01A	TCGA-BH-A0DG-01A	TCGA-B6-A0I9-01A	TCGA-E9-A1RB-01A	TCGA-A2-A0CP-01A
ENSG00000000003.13	15.67	18.45	16.68	17.1	17.34	18.04	17.96	17.71	17.07	17.54
ENSG00000000005.5	0	10.46	14.1	13.61	10.65	14.04	13.8	9.996	9.356	12.57
ENSG00000000419.11	19.85	19.58	19.08	19.25	18.98	19.65	19.04	20.25	19.89	18.93
ENSG00000000457.12	16.53	15.68	17.45	16.1	17.5	16.29	17.17	17.54	16.64	15.81
ENSG00000000460.15	15.45	16.04	15.07	15.15	14.97	15.23	15.64	16.25	16.71	14.05
ENSG00000000938.11	14.94	16.62	17.21	15.29	15.18	17.3	16.1	15.64	15.15	15.73
ENSG00000000971.14	16.83	15.33	15.89	18.31	16.09	18.37	17.51	15.84	16.84	17.49
ENSG00000001036.12	19.53	19.92	18.57	18.54	17.88	19.29	19.26	18.73	18.82	18.8
ENSG00000001084.9	16.89	16.34	16.72	16.84	16.43	16.28	16.43	17.33	15.79	15.92
ENSG00000001167.13	18.48	17.89	18.33	18.48	18.15	17.51	18.22	18.55	17.63	17.85

Source: Xena Browser

typically around 100 base pairs (bp) in length, but can vary from 30bp to 10000 depending on the technology (LOWE et al., 2017).

Figure 2.9 – Overview of the RNA-Seq process. See main text for explanations.



Source: Lowe et al. (2017)

The output of the high-throughput sequencing technologies is typically represented in file formats such as the FASTQ (COCK et al., 2010). The FASTQ files contain a list of *reads* and the quality associated with each nucleotide in each read. The reads are represented in general using sequences of the letters A, C, G and T, matching the nucleotide representation. These raw reads then need to be aligned to a reference genome sequence in a computationally intensive process (*in silico* in Figure 2.9). Prior to the alignment, it is not known from which region of the genome (and therefore gene) the read originated in the fragmentation step. The alignment consists in associating each read to its best possible position in genome, which allows one to know which region of the genome originated the read. The regions of the genome can be further annotated with the genes associated with the region and by counting the number of reads that have aligned to parts of the gene in the genome one can estimate how expressed the gene was in that sample.

2.2.2.2 The GDC Pipeline

In this work, we have used gene expression data processed by the GDC mRNA pipeline. The pipeline starts by aligning the reads with a reference genome. Specifically, it takes as input raw read counts and uses STAR (DOBIN et al., 2013) to align the reads to the reference genome GRCh38[2]. This step outputs gene ids associated with read counts at the gene ids. The next step is to normalize the read counts associated with each gene, which can be done using a few different techniques, and is a necessary step in order to compare expression values between samples. In our work, we have used the data normalized through the Fragments Per Kilobase of transcript per Million mapped reads upper quartile (FPKM-UQ) [3]. The normalized value of the expression of gene g becomes

$$\mathsf{FPKM_g} = \frac{\mathsf{RM_g}}{\mathsf{RM_{75}}L_g}10^9, \tag{2.22}$$

where $\mathsf{RM_g}$ is the number of reads mapped to the gene (that is, the actual read counts that mapped to the gene), $\mathsf{RM_{75}}$ is the number of reads that were mapped to the gene at quantile 75% and L_g is the length of the gene in base pairs. The normalization using $\mathsf{RM_{75}}$ is a form of quantile normalization (BULLARD et al., 2010) and is due to the fact that the read counts often contain a lot of 0s and low values, which causes the median to be less useful. Furthermore, normalizing by L_g (the length of the gene) is needed in order to compare expression values of genes within the sample as well, because longer genes will have more reads mapping to their regions than smaller ones. We have simplified the process in the description above, but a more thorough description can be found in the pipeline's description in GDC website [4].

[2]https://gdc.cancer.gov/about-data/gdc-data-processing/gdc-reference-files
[3]https://docs.gdc.cancer.gov/Encyclopedia/pages/FPKM-UQ/
[4]https://docs.gdc.cancer.gov/Data/Bioinformatics_Pipelines/Expression_mRNA_Pipeline/

3 RELATED WORK

This Chapter reviews the mains works related to ours. Our study is primarily related to the use of NN-based models to supervised classification tasks in cancer genomics. Because of that, we review some of the recent works that have incorporated fully-connected NNs in cancer genomics in Section 3.1. After that, we look into convolutional neural networks that were proposed for these kind of tasks in Section 3.2. Only then we consider the recent works that have dealt with graph convolutions, in Section 3.3. In this case, we have chosen to also consider a few works that used GNNs in single-cell RNA-seq tasks, even if not related to cancer specifically. We opted for that because single-cell RNA-seq is in many ways similar to RNA-seq data and GNNs are a quite recent development. Therefore, including these works would allow a better comprehension of how the GNNs are being studied in the context of genomics data. To contextualize and justify our experiments on the generalizability of the pan-cancer tumor prediction models, we also include a review of the few works we have found that broadly fall in the category of FSL techniques in the genomics context. This is presented in Section 3.4. Finally, Section 3.5 contrasts the reviewed works with the research conducted in this dissertation.

3.1 Neural Network Methods for Genomic Classification Tasks

Various recent works propose different neural network-based approaches for genomic classification tasks, such as tumor/normal prediction, cancer cohort classification in pan-cancer scenarios, subtype classification, metastasis and other phenotypes. Yu et al. (2019) studied extensively different neural network architectures, including multi-layer perceptron (MLP) and convolutional neural networks (CNN), for distinguishing between tumor and non-tumor samples and to predict disease stages. They compared the models with other machine learning algorithms (e.g. random forests and support vector machines) and concluded that the shallow MLP models (with one or two hidden networks) are enough to present the best performance in general and robust models. In a similar fashion, Hanczar, Bourgeais and Zehraoui (2022) compared the performance of NN models with other ML models and whether different forms of transfer learning could improve the performance of the NNs. By analyzing the size of datasets given as input to the models, they conclude that the NN models outperform other methods when there are enough samples in the pan-cancer tasks of cohort and tumor/normal samples classification. In other scenarios, such as tumor/normal sample prediction using a single cancer type (in contrast to the pan-cancer situation), the results are similar to the other methods.

Besides those ample studies experimenting with NN architectures, a few works have considered more elaborate and specific schemes. Joshi and Dhar (2022) proposed to use Bayesian Neural Networks with epistemic uncertainty for classifying cancer types and subtypes using RNA-seq data. They argued that introducing uncertainty correction improved the performance and robustness of the models. Another interesting model was proposed by Zhang et al. (2022) that studied the currently popular Transformer architecture for cancer-related phenotype prediction tasks and also explored datasets with single-cell RNA-seq. They also showed how the self-attention mechanisms could be used to interpret the models for biomarker identification. Divate et al. (2022) also considered the importance of enabling interpretability of these models and used SHapley Additive exPlanations (SHAP) (LUNDBERG; LEE, 2017) to analyze a 5-layer fully connected network for pan-cancer classification tasks and extract cancer biomarkers. Bourgeais, Zehraoui and Hanczar (2022) considered how to develop a self-explainable approach and developed an interesting approach embedding the Gene Ontoloy (GO) (CONSORTIUM, 2004) in the hidden layers of a neural network. In their model, each neuron of the network corresponds to a GO concept, and the gene expression of the genes is propagated from the genes to more general GO concepts. A selection layer is added that takes as inputs the concatenation of the activation of non-input neurons and identify the most activated neurons. Since they correspond to GO terms, the model prediction can be explained by considering the selected neurons and their score (activation).

3.2 Convolutional Neural Networks on Genomic Classification Tasks

A set of works explored primarily the use of CNNs for genomic classification tasks. Lyu and Haque (2018) argued that adjacent genes ordered by chromossome position were more likely to interact. Hence, they ordered the gene expression data of each patient following the chromossome position of the genes and reshaped each sample from 10381 features (the genes) to 102 x 102 matrices. Following that, they trained a three-layer CNN for cohort classification and used GradCam (SELVARAJU et al., 2017) (a gradient-based interpretation method for neural networks), to analyze and extract gene biomarkers. A similar framework was used by Guia, Devaraj and Leung (2019).

A more thorough exploration of the CNN models was performed by Mostavi et al.

(2020). In their work, Mostavi et al. (2020) argued that previous works had the problem that their models were not trained to ignore the tissue-of-origin, so that it is possible that they learn not cancer-specific genes, but tissue-specific genes instead. Furthermore, they also compared the 2-dimensional inputs suggested in the works above with applying a 1-dimensional convolution directly on the gene vector of gene expression data. Besides evaluating these models on the pan-cancer cohort classification task, they also explored it for classification of breast cancer subtypes. In order to interpret the model's decisions, they used saliency maps (SIMONYAN; VEDALDI; ZISSERMAN, 2013), another gradient-based method for interpreting neural networks. At a similar time, Shah et al. (2020) also considered CNNs whose inputs were the 1-dimensional vectors of gene expression data. However, they focused on analyzing how the number of features used affected the performance of the model. To do that, they propose a framework that applies a Laplacian-score-based feature selection method for choosing genes that serve as inputs to the CNNs and vary the number of retained genes.

Another framework for cancer type classification based on CNNs was developed by Mohammed et al. (2021). They begin by selecting the differentially expressed genes. This step, however, still retain most of the genes. Therefore, they also used a Lasso regression to obtain a final number of only 173 genes. Next, they construct an ensemble using 5 CNNs with the selected genes and a final NN is used for combining the outputs of the CNNs. Similar to Hanczar, Bourgeais and Zehraoui (2022), a recent work by Liu et al. (2022) explored if fine-tuning (a form of transfer learning) could improve the results on cancer/normal sample prediction on datasets with small sample sizes, but using the CNNs proposed in Mostavi et al. (2020) instead of NNs. In general, they found that fine-tuning improved the results. Finally, another interesting method by Chuang et al. (2021) attempted to use CNNs but instead of inputting directly the reshaped gene expression matrix, they generated 2-dimensional representations by combining protein-protein interaction networks and RNA-seq data with a method based on spectral clustering.

3.3 Graph Neural Networks in Genomic Classification Tasks

The works mentioned previously, however, did not experiment with graph convolutions in their models. The CNNs, in particular, make assumptions about the interaction of adjacent genes. However, one can instead use molecular interaction networks available in the biological literature that represent the associations between biological components using a graph. Then, one can use the recent generalizations of the convolution operation to non-euclidean domains, such as graphs. This is the approach taken in a series of recent works.

One of the first studied GNNs was based on the ChebConv and was developed for classifying breast cancer subtypes (RHEE; SEO; KIM, 2017). The model used the STRING PPI network as the backbone and RNA-seq data from TCGA. Besides using a GNN, they incorporated also a Relation Network in the model (SANTORO et al., 2017). Another GNN was proposed for pan-cancer classification of cancer cohort by Ramirez et al. (2020). In their work, they explored the use of two backbone networks: one constructed from the STRING PPI network and another one constructed from a correlation matrix built using the gene expression data. In order to identify biomarkers from their model, they perturbed the dataset by setting a gene to a value of 0 or 1 and passing the data through the model again. Then, they take the genes that caused the largest changes in accuracy as the most discriminative ones. Later on, they also integrated clinical data with the GCN models for cancer survival prediction (RAMIREZ et al., 2021). In this work, they did not use a perturbation-based method for interpreting the model, and instead explored a two-step based technique. First, they look for the important neurons of the hidden layer of a NN that takes the outputs from the GCN. Then, they look for the genes that were the most correlated with the values of the most important hidden neurons found.

A multi-omics GNN for classifying cancer subtypes using gene expression and copy number variation data was also proposed by Li, Wang and Nabavi (2021). They also used a GNN based on the ChebConv. The resulting node values produced by the convolution are pooled and then flatted to be passed to a FC network that produced the final graph level embedding. Furthermore, the GNN is trained in parallel to a NN, since, as they argue, the graph convolution does not capture global information. The outputs of the NN and the graph level embedding are concatenated and passed to a FC network that outputs the probability of each cancer type. Moreover, the graph level embedding is passed to a decoder module that is trained to reconstruct the gene expressions.

Another related area that has seen the development of GNN models is that of classification of cell types. Differently from RNA-seq, single-cell RNA-seq (scRNA-seq) data contains gene expression on the level of cells, instead of the tissue. Nevertheless, they share similar features and can be combined with similar biological networks. Yin et al. (2022) developed a GNN model for classifying cell types using single-cell RNA-seq data. They argued that one could take advantage of a backbone biological network to better handle technical noise in the data. Their model consisted of a single GraphSAGE convolutional layer, followed by a neural network that used as input the concatenated node embeddings produced by the graph convolution layer and outputs the probabilities of the input example belonging to each type of cell. Furthermore, they added a learnable edge importance score vector to the GraphSAGE architecture used. After training, these edge scores can be used for interpreting the model by sorting the gene interaction pairs and selecting the top unique genes in the list. In Wang, Bai and Nabavi (2021b), the authors used a very similar model as in Li, Wang and Nabavi (2021) but to classify cell types using scRNA-seq.

Some works have also taken advantage of the GNNs in non-usual ways to produce explainable and accurate models. Lee et al. (2020) developed a framework using GCNs, KEGG pathways, multi-attention ensembles (MAE) and network propagation for cancer subtype prediction. Specifically, they fit one GNN for each of the 287 pathways used. In each model, the graph convolution outputs are passed to a MLP to produce probabilities for each cancer subtype. A MAE is then used to aggregate the results of each pathway model. The model decisions can be explained at the level of the KEGG pathways by analyzing the attention vectors of the MAE. Hayakawa et al. (2022) also developed a model containing one graph per KEGG pathway utilized, targeting specifically the classification of subtypes of diffuse large B-cell lymphoma. Their models uses two GCN (KIPF; WELLING, 2016) layers and a pooling layer applied to each pathway graph separately. The outputs of the pooling layers of each graph are then passed to a fully-connected layer to predict the subtype of the sample. Another novelty in their work is the application of SHAP to analyze not only the inputs of the model, but also the outputs of the pooling layers of each pathway. Finally, in Chereda et al. (2021) a GNN model based on the ChebConv is proposed for metastasis prediction in breast cancer. They also analyzed the model using a layer-wise relevance propagation method for explaining the model decisions and finding the most relevant genes. These relevant genes were then used to plot patient-specific networks.

3.4 Few-shot learning methods

Wang et al. (2020) define few-shot learning as a type of classification problem where a limited number of labeled samples are available for training. A few works have explored few-shot learning methods on cancer genomic classification tasks. Transfer learning has been studied in various ways for cancer type prediction (HANCZAR; BOURGEAIS; ZEHRAOUI, 2022; KHORSHED; MOUSTAFA; RAFEA, 2020), in the context of cancer survival prediction (LóPEZ-GARCíA et al., 2020) and also for handling single-cell RNA-seq data (PARK; HAUSCHILD; HEIDER, 2021; WANG et al., 2019). Metric learning has also been explored as a way of learning a distance function that can be used to compare the gene expression of tissue or cell samples. In the context of single-cell data (KOH; HOON, 2021; MA et al., 2022), learning a deep metric has been studied to develop models that can differential cell types across different experiments. In the context of cancer genomics tasks, Mostavi et al. (2021) argued that the accuracy in classifying cancer types is highly dependent on the number of available samples. To overcome this, they designed a model using siamese networks that learns a distance function between tissue samples, the model can then be used for one-shot learning on unseen cancer types, thus enabling cancer classification even when a single sample is available. Finally, multi-tasking (that is, designing models that share parts of its parameters across different tasks and output predictions for each of them) has been explored by Liao et al. (2019). In their work, they implemented a DNN that output classifications for different cancer genomic datasets. In the model, a hidden unit was shared among the tasks.

3.5 Discussion

In this section, we summarize the works discussed previously and focus on the general differences with this dissertation. Section 3.5.1 contrasts our work in terms of the model explored and the interpretation technique, while Section 3.5.2 deals with the few-shot learning approaches.

3.5.1 GNN Models for Cancer Genomic Classification Tasks

We summarize the related work discussed previous on neural network methods (including CNNs and GNNs) in Table 3.1.

Table 3.1 – Related work on cancer genomic classifications tasks using NN-based models.

Reference	**Model**	**Features**	**Task**	**Interpretability**	**Networks**
Yu et al. (2019)	NN vs Others	RNA-seq	Cancer Stage Classification and Cancer Prediction	No	-
Hanczar, Bourgeais and Zehraoui (2022)	NN vs Others	RNA-seq and Microarray	Pan-cancer Cohort Classification and Cancer Prediction	No	-
Joshi and Dhar (2022)	Feature Selection + Bayesian NN with Uncertainty Correction	RNA-seq	Pan-cancer Cohort and Subtype Classification	No	-
Zhang et al. (2022)	Transformers	RNA-seq and scRNA-seq	Pan-cancer Cohort and Single-cell Classification	Yes	-
Divate et al. (2022)	NN	RNA-seq	Pan-cancer Cohort Classification	Yes	-
Bourgeais, Zehraoui and Hanczar (2022)	NN with the Gene Ontology	RNA-seq and Microarray	Pan-cancer Cohort Classification and Cancer Prediction	Yes	GO
Lyu and Haque (2018)	2-D CNN	RNA-seq	Pan-cancer Cohort Classification	Yes	-

Reference	Model	Features	Task	Interpretability	Networks
Guia, Devaraj and Leung (2019)	2-D CNN	RNA-seq	Pan-cancer Cohort Classification	Yes	-
Mostavi et al. (2020)	2-D and 1-D CNNs	RNA-seq	Pan-cancer Cohort Classification	Yes	-
Shah et al. (2020)	Feature Selection + 1-D CNNs	Microarray	Cancer Prediction and Subtype Classification	No	-
Mohammed et al. (2021)	Feature Selection + Ensemble of 1-D CNNs	RNA-seq	Pan-cancer Cohort Classification	No	-
Liu et al. (2022)	2-D and 1-D CNNs	RNA-seq	Cancer Prediction	No	-
Chuang et al. (2021)	2-D CNNs	RNA-seq	Pan-cancer Cohort Classification	No	Various PPIs
Rhee, Seo and Kim (2017)	ChebConv + Relation Net GNN	RNA-seq	Cancer Subtype Classification	No	STRING PPI
Ramirez et al. (2020)	GCN-based GNN	RNA-seq	Pan-cancer Cohort Classification	Yes	STRING PPI and Co-expression
Ramirez et al. (2021)	GCN-based GNN + NN	RNA-seq + Clinical	Cancer Survival Prediction	Yes	Genemania GGI and PPI, and Co-expression

Reference	Model	Features	Task	Interpretability	Networks
Li, Wang and Nabavi (2021)	ChebConv-based GNN + AutoEncoder + NN	RNA-seq + CNV	Cancer Subtype Classification	No	STRING PPI, GGI (BioGrid), and Co-expression
Yin et al. (2022)	GraphSAGE-based GNN	scRNA-seq	Single-cell Classification	Yes	STRING PPI and Others
Wang, Bai and Nabavi (2021b)	ChebConv-based GNN + AutoEncoder + NN	scRNA-seq	Single-cell Classification	No	STRING PPI
Lee et al. (2020)	GCN-based GNN per KEGG pathway	RNA-seq	Cancer Subtype Classification	Yes	KEGG pathways
Hayakawa et al. (2022)	GCN-based GNN per KEGG pathway	Microarray	Cancer Subtype Classification	Yes	KEGG pathways
Chereda et al. (2021)	ChebConv-based GNN	Microarray	Metastasis Prediction	Yes	HPRD PPI
Ours	**ChebConv-based GNN + Multiple pooling layers**	**RNA-seq**	**Pan-cancer Cohort Classification and Cancer Prediction**	**Yes**	**STRING**

The table shows the main models, features, classification tasks and biological networks considered in each work, as well as whether they took the problem of interpreting the model into account. We refer to task of classifying the tissue-of-origin of each sample generically as the Pan-cancer Cohort Classification task. The number of cohorts actually considered in each work varied from 5 (MOHAMMED et al., 2021) to 33 (MOSTAVI et al., 2020). The task Subtype Classification in the table involved the classification of tissue in subtypes of specific cancers. In some works as Li, Wang and Nabavi (2021) this was also considered in a pan-cancer scenario, where subtypes from different cohorts were used simultaneously. The Cancer Prediction task refers to the classification between tumor and normal samples, and we will often refer to it as Tumor Prediction as well. In general, this classification task is specific to a single cohort, where one wishes to determine whether a tissue sample is tumorous or not. Two works (YU et al., 2019; HANCZAR; BOURGEAIS; ZEHRAOUI, 2022) provided a comparison between NNs for cancer genomic classification versus other commonly used ML algorithms. In general, their results points towards the idea that neural networks have results on the level of the state-of-the-art.

Although some of the works (BOURGEAIS; ZEHRAOUI; HANCZAR, 2022) have considered models embedding prior knowledge in the NNs, the pure NN models disscussed above do not make use of any prior biological knowledge, in particular in the form of biological networks. This is in contrast to the GNN models that we study in this work, where we consider models that are built over the STRING network database. An approach that considers some form of interaction between genes and that has drawn a lot of attention in the recent years is the use of CNNs to model omics data. To the best of our knowledge, a CNN for modeling gene expression data for cancer classification tasks was first proposed by Lyu and Haque (2018). In their work, they argued that genes that are physically close together in the chromossome are more likely to interact. This assumption would support the use of CNNs, since they are designed to take advantage of localized patterns in the data. Later works based on CNNs, however, were not always clear whether they took the order of the genes into account. Instead of assuming the genes that are close in chromosome are more likely to interact, the GNNs-based approach taken in our work and others use biological networks that are designed to model the interactions between the genes and/or molecular components. Furthermore, network-based approaches to understanding diseases is a well established field of research (BARABáSI; GULBAHCE; LOSCALZO, 2011). The previous works based on GNNs, however, were mostly based on the application of a single graph convolutional layer and pooling step. Since modules in biological networks such as the STRING are related to biological processes, we hypothesized that more steps of convolutional networks and pooling could provide further improvement in the results. Furthermore, these models would require much less parameters than a general neural network, since the convolutional layers share weights when computing their outputs.

Our work also differs from the previous in the ways for interpreting the model. Most of the related work considered interpreting at the level of the genes, in order to identify gene biomarkers. We have also done that by considering the saliencies of the input genes. Furthermore, our work considered the importancies of the embeddings produced at the final coarsening layers of the model (more details are given in Section 4). Since these embeddings have a correspondence with the initial sets of clustered genes, we interpreted their importance as the importancy of the over represented biological processes in the cluster, allowing a higher level interpretation of the model's decision. Other works have attempted to also provide high level interpretations by considering directly KEGG pathways in the network architecture. Although quite interesting, these works are considerably different from ours since they are based on the idea of self-explainable models.

A few methods have considered frameworks that included feature selection approaches as well. In our case, we were interested in analyzing the effects of pooling, such that adding feature selection would introduce one more variable of interest and complicate the analysis. It is reasonable to expect variations if feature selection is used both performance-wise and in the interpretation results. In future work, we plan to consider smaller networks generated from selected genes.

To finalize, we briefly review here what are the general results in terms of the state-of-the-art (SOTA) performance. First, two recent works have compared the accuracies of NNs with other traditionally used methods and concluded that the performance of NNs are on the level or higher than that of the SOTA (HANCZAR; BOURGEAIS; ZEHRAOUI, 2022; YU et al., 2019). Hanczar, Bourgeais and Zehraoui (2022), in particular, states that this is true, however, only when there is a sufficient number of samples in the datasets. For the Cohort Classification task (referred in their work as the TCGA Type task), they reported a mean accuracy of 98.89%. In our own tests, we obtained accuracies for the NNs closer to 96%. We believe that a significant part of the difference can be attributed to the selected cohorts. In their work, they have considered only cohorts with more than 350 samples available, which led to 11 cohorts in total. More importantly, this implied that the READ and COAD cohorts were not included. As we will see in Section 5, most of the error is concentrated in distinguishing between these two types. Recent works proposing different methods for handling gene expression data have also reported higher accuracies. In Joshi and Dhar (2022), for example, the authors report an accuracy for their method of 97.83% on the test set. However, we did not encounter any variance information for this result. In our experiments, we found that the performance of the model can vary significantly depending on the dataset split. We executed a 5-times holdout evaluation scheme and the best and worst results for the NN were 96.37% and 94.69%, respectively.

In general, each work performs experiments in a different way, such that a fair comparison between them is difficult to perform without re-executing all the methods. More importantly, the reported results often concentrate around accuracy, but since classifying between cohorts is generally a quite imbalanced problem, accuracy might not be the best metric to consider. Because of that and the works mentioned above suggesting that the NNs are the state-of-the-art for gene expression data, we were led to the conclusion that there is no clear evidence of any method performing better than the others, and there is also no significant evidence that they are able to perform better than the NNs. Therefore, we believe that NNs can be used as a safe baseline for comparison with the results that we obtain in our work, without having to go through the difficult task of executing models from various different works. We note, nevertheless, that raw performance is not the only relevant aspect of a model, in particular on gene expression data, where we are often interested in interpreting the model as well.

3.5.2 Few-shot Learning Approaches

We summarized some of the recent works that have explored the use of few-shot learning methods in the context of cancer genomic classification tasks in Section 3.4. In our work, we focus on understanding whether the introduction of samples from different cohorts could improve performance in a cohort-specific task. In particular, we were interested in understanding whether there could be improvements in the performance of the more imbalanced and smaller cohorts. In alignment with the work developed by Mostavi et al. (2021), we also looked into whether the pan-cancer model was able to predict cancer from cohorts that it has not seen during training. To the best of our knowledge, a study comparing pan-cancer with cohort-specific models for cancer prediction and their use in previously unseen cohorts has not been considered before.

4 MATERIALS AND METHODS

In this chapter, we describe the data and models used in this work. In particular, we detail the datasets and pre-processing steps considered in general in this work, and the pooling schemes that we have explored and its relations to other ideas. Finally, we also describe the training and evaluation procedures used here.

4.1 Data Preprocessing

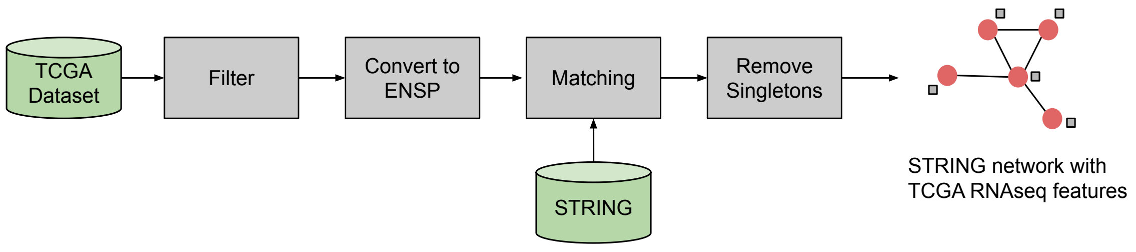
We obtained RNA-seq data for various types of cancer from The Cancer Genome Atlas (TCGA) [1]. Specifically, we downloaded the upper-quartile FPKM (UQ-FPKM) data from Xena (GOLDMAN et al., 2020) for each of the 33 cancer cohorts available and retained only cohorts that had at least 10 Primary Tumor samples and 10 Normal Tissue samples. For each cohort, we removed genes and samples containing more than 20% of missing values, similarly to Duan et al. (2021), Albaradei et al. (2021), Chaudhary et al. (2018) and Wang et al. (2014). We used the log-transformed FPKM values, so that the features follow an approximate Gaussian distribution.

In our experiments with biological networks, we have generally used version 11.6 of the STRING network, restricted to the Homo sapiens (SZKLARCZYK et al., 2019). The nodes of the stringdb network represent proteins and the edges indicate the strength of the connection with a score between 0 and 1. This score is computed based on various evidence channels, including mining of scientific texts and databases, and the analysis of high-throughput data. The features of the TCGA datasets are genes named using *ensembl* gene identifier (ENSG), whereas the nodes of the STRING network use *ensembl* peptide (ENSP). Therefore, we mapped each ENSG value to their best matching ENSP id using the string API [2]. This reduced the number of features in the dataset significantly to 14148 genes. Furthermore, this process also created 15 singleton nodes in the resulting graph, besides one big connected component containing all the remaining nodes. Singleton nodes would cause problems for the clustering algorithms, and we have therefore chosen to drop these nodes, since they represented a very small percentage of the total network. This resulted in a single connected component containing 14133 genes, each of which is associated with the expression value of a gene, here represented by the

[1]https://www.cancer.gov/ccg/research/genome-sequencing/tcga

[2]https://string-db.org/cgi/help.pl?subpage=api

Figure 4.1 – Preprocessing of the TCGA and STRING datasets.



Source: The Author

best matching ENSP identifier. A summary of this process is shown in Figure 4.1. The resulting number of examples in each TCGA pan-cancer dataset and of each type are summarized in Table 4.1.

Table 4.1 – Number of examples from each cohort and type of sample in the TCGA pan-cancer datasets after the preprocessing.

Cohort	Primary Tumor	Solid Tissue Normal	Total
Bladder Urothelial Carcinoma (BLCA)	411	19	430
Breast invasive carcinoma (BRCA)	1097	113	1210
Colon adenocarcinoma (COAD)	469	41	510
Esophageal carcinoma (ESCA)	161	11	172
Head and Neck squamous cell carcinoma (HNSC)	500	44	544
Kidney Chromophobe (KICH)	65	24	89
Kidney renal clear cell carcinoma (KIRC)	533	72	605
Kidney renal papillary cell carcinoma (KIRP)	288	32	320
Liver hepatocellular carcinoma (LIHC)	371	50	421
Lung adenocarcinoma (LUAD)	524	59	583
Lung squamous cell carcinoma (LUSC)	501	49	550
Prostate adenocarcinoma (PRAD)	498	52	550
Rectum adenocarcinoma (READ)	166	10	176
Stomach adenocarcinoma (STAD)	375	32	407
Thyroid carcinoma (THCA)	502	58	560
Uterine Corpus Endometrial Carcinoma (UCEC)	547	35	582
Total	7008	701	7709

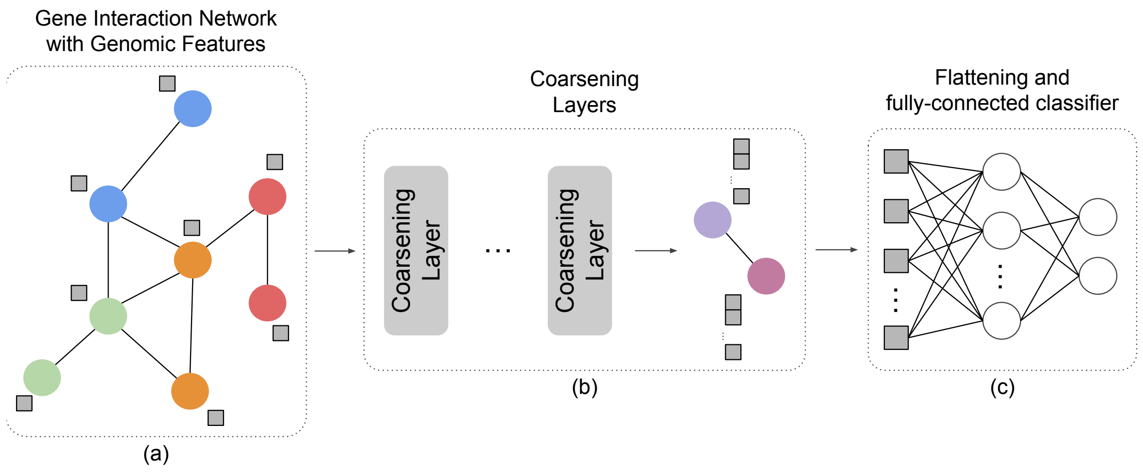
Source: The Author

4.2 GNNs on Biological Networks with Pooling

We consider in this work graph neural networks models based on multiple layers of graph convolution and pooling operations, built over gene-interaction networks. The general structure of these models is shown in Figure 4.2. The model takes as inputs a gene interaction network and genomic data associated with the genes (Figure 4.2-(a)). We represent the genomic data associated with a node by the gray square by its side. In this work, we have used RNA-seq data, so that there is a single feature associated with each node. However, it is quite simple to transform this kind of model in a multi-modal one, since it only requires that the additional features are concatenated to the previous ones for each node. For instance, we could include copy number variation (CNV) data besides RNA-seq, so that two squares would be represented in the figure. In fact, other works have included CNV data along with RNA-seq and showed that it improved the results (LI; WANG; NABAVI, 2021), but we leave that as future work.

In Figure 4.2-(b), the network and the associated data goes through multiple coarsening layers, which include convolution operations, pooling and a non-linearity. At each step, a new graph whose vertices are supernodes associated with the supernodes in the previous graph are generated. Finally, the embeddings of the supernodes at the final layer are flattened and passed through a fully-connected network (Figure 4.2-(c)).

Figure 4.2 – General architecture of the GNN models used in this work.



Source: The Author

The number of coarsening layers, the presence and type of convolutional network and the pooling operation are design parameters that we explore in the experiments. We use the term *coarsening layer* to refer to the module in Figure 4.3, in order to make it clear that the convolution operation is not mandatory in this context, and that the pooling operation can also vary. The graph convolution operation generates new feature maps by applying local convolutions. This is represented in Figure 4.3 by the increase in the number of squares from 1 to 4 after the convolution. In general, we have used the ChebConv (Section 2.1.3.1) with $K = 2$, that is, where the convolutions include only the node itself and its immediate neighbors.

Following that, the pooling operation aggregates the representations of nodes that belong to the same clusters (the clusters are precomputed, see Section 4.2.1). In this work, we have considered two forms of pooling: a simple sum-pooling and a *weighted* pooling approach. Another option would be concatenation, but this would produce very high-dimensional embeddings, requiring large amounts of memory to perform the next convolutions. The sum pooling is a traditional operation used in the context of CNNs, while the weighted pooling is inspired by the *CancelOut* layer introduced by Borisov, Haug and Kasneci (2019). We can think of the hierarchical clusters in the network as assignment matrices $\mathbf{S}^{(l)} \in \{0, 1\}^{N_l \times N_{l+1}}$, where N_l and N_{l+1} are the number of supernodes before and after pooling, $l = 0, ..., L$ with L being the number of coarsening layers and N_0 the number of nodes in the original network. The assignment matrix is given by $S_{ij}^{(l)} = 1$ if node i is assigned to cluster j and 0 otherwise. Then we can write the sum-pooling operation as

$$\mathbf{H}_{P_S}^{(l)} = (\mathbf{S}^{(l)})^\intercal \mathbf{H}_c^{(l)}, \tag{4.1}$$

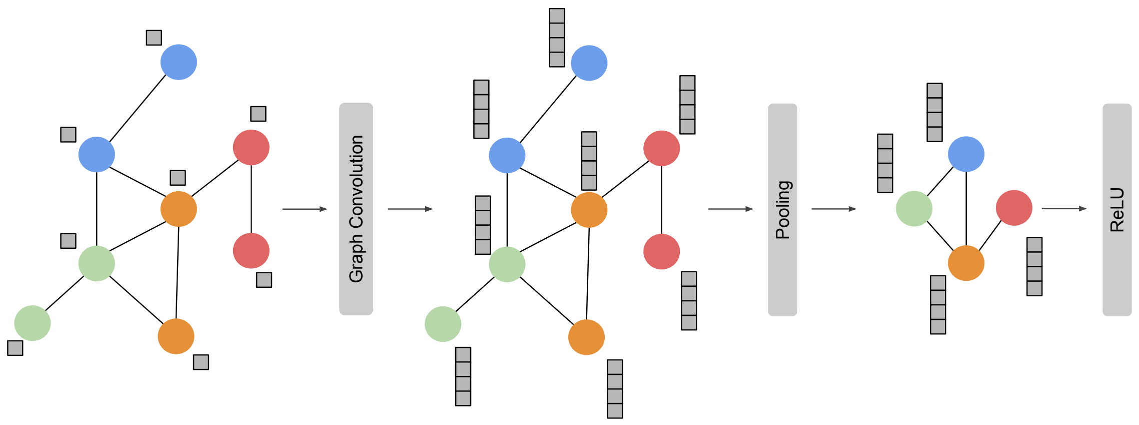
where $\mathbf{H}_c^{(l)} \in \mathbb{R}^{N_l \times F_l}$ is the feature matrix of the nodes produced after the graph convolution and $\mathbf{H}_{P_S}^{(l)} \in \mathbb{R}^{N_{l+1} \times F_l}$. The weighted pooling layer is similar, with the difference that the node features are multiplied point-wisely by a learnable weight vector $\mathbf{w}^{(l)} \in \mathbb{R}^{N_l}$,

$$\mathbf{H}_{P_w}^{(l)} = (\mathbf{S}^{(l)})^\intercal (\mathbf{w}^{(l)} \odot \mathbf{H}_c^{(l)}). \tag{4.2}$$

Note that $\mathbf{w}^{(l)}$ is broadcasted to the shape $\mathbb{R}^{N_l \times F_l}$.

There are two reasons why we considered a weighted pooling approach. First, convolutions are invariant to translations of the graph signal and apply the same filters on all the nodes. In images, this is generally desirable. However, in our context, we do not want to completely ignore the identity of each gene, since they are important by themselves. Therefore, we added weights to the pooling operation so that the model can learn parameters that are dependent on the genes themselves.

Figure 4.3 – General structure of a coarsening layer. The presence of the convolution, the number of convolution filters, and the type of pooling operation are design parameters.



Source: The Author

Secondly, we observed that the sum-pooling operation alone was not able to provide good discrimination between the importances of each feature, specially when no convolution was used. This can be understood by considering the derivatives with respect to the inputs in the case of the sum pooling. In a simplified setting, without non-linearities and convolutions, the result of the coarsening layers for a supernode i corresponding to a cluster of nodes C_i is

$$h_i = \sum_{k \in C_i} x_k. \tag{4.3}$$

The fully-connected network then transforms the supernode values $\mathbf{h}$ according to a function $f(\mathbf{h}) : \mathbb{R}^M \to [0,1]$. The derivative of f with respect to a certain input feature x_j is given by

$$\frac{\partial f(\mathbf{h})}{\partial x_j} = \nabla_h f(\mathbf{h})^\mathsf{T} \frac{\partial \mathbf{h}}{\partial x_j}. \tag{4.4}$$

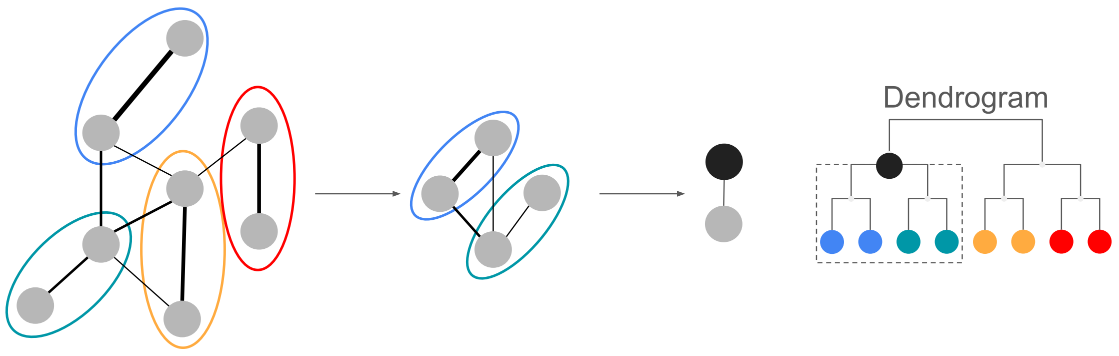
From Equation 4.3, we have that $\frac{\partial \mathbf{h}}{\partial x_j} = 1$ if feature x_j belongs to cluster C_i and 0 otherwise. Hence, for all $x_j \in C_i$ we have that $\frac{\partial f(\mathbf{h})}{\partial x_j} = \frac{\partial f(\mathbf{h})}{\partial h_i}$. This means that the derivatives with respect to the features in a certain cluster are all the same, and hence the model would not distinguish between genes that belong to the same clusters. When convolutions and ReLUs are used as well the situation is more complex and the derivatives can be different depending on the input features, but nevertheless the argument above is suggestive of the fact that using a sum-pooling has less power to determine the important features than a weighted approach. It is also interesting to note that, when only the weighted pooling operation is used, without convolutions, we have a model that is similar to a deep neural network, but each layer is restricted in its connections to only its neighbors.

Finally, we have generally used a neural network containing a single hidden layer with 256 neurons for the final classifier. We chose this architecture because we have observed when trying different architectures that these shallow networks performed just as well as deeper ones; furthermore, Yu et al. (2019) studied a range of neural networks and concluded that shallow NNs outperformed deeper ones for omics data. In the coarsening models, the number of input neurons was equal to the number of features in the flattened embeddings after the coarsening layers. When only the neural network was used (as the baseline), the number of inputs was equal to the number of genes. After each layer besides the last one, we have used dropout, batch normalization and the ReLU as the non-linear activation function. For the last layer of the model, however, we have either used a sigmoid or the softmax function, depending on whether it was a binary or multi-category classification task.

4.2.1 Hierarchical Clustering of the Biological Network

The GNN models we described above require that the hierarchical clustering of the biological network used are computed prior to training. In Figures 4.2 and 4.3, for example, the prior clustering computed is represented by the color of the nodes. There exists a variety of methods that can be used for clustering graphs hierarchically. Following similar works (WANG; BAI; NABAVI, 2021a), we have chosen to use the heavy-edge matching scheme (see Section 2.1.4). We have also considered using Louvain's algorithm, but we found that the dendrogram generated was not sufficiently detailed. Specifically, we applied the heavy edge matching algorithm to the STRING network with 14133 nodes generated as described in Section 4.1 and obtained seven coarsened versions of the graph, each with approximately half the number of the nodes of the other, with a final graph containing 111 supernodes. Each supernode in a coarsened graph is associated with a cluster of genes sharing a neighborhood in the STRING network. This is illustrated in Figure 4.4, where the supernode in black and its associated input nodes are highlighted as an example. Since an important part of our work deals with the interpretability at these coarser levels of the graph, we performed an over-representation analysis of the clusters obtained at the coarsest versions of the network.

Figure 4.4 – Two steps of the heavy-edge matching algorithm (see Section 2.1.4). The process implicitly creates a dendrogram and each supernode is associated with a cluster of input nodes.



Source: The Author

4.3 Model Training and Evaluation

As is common practice, we have used the cross-entropy loss in all the models explored here, with a few variations due to particularities of the problems. Gene expression data is generally imbalanced and can easily lead to overfitting if not handled properly. Therefore, we have used a weighted version of cross-entropy that increases the penalty associated with mistakes of the minority classes in proportion to the inverse of their sizes. That is, the weight for class $c \in 1, ..., C$ is given by $w_c = \frac{N}{CN_c}$, where N stands for the total number of examples in the dataset and N_c is the number of examples in class c. This approach is used in *scikit-learn* and is based on the heuristic in King and Zeng (2001). For a matrix of outputs $\mathbf{h} \in \mathbb{R}^{N \times C}$, the cross-entropy loss becomes

$$L_{ce}(\mathbf{h}, \mathbf{y}) = \sum_{n=1}^{N} \frac{l_n}{\sum_{n=1}^{N} w_{y_n}}, \tag{4.5}$$

with

$$l_n = -w_{y_n} \log \left(\frac{\exp h_{n,y_n}}{\sum_{c=1}^{C} \exp h_{n,c}} \right) \tag{4.6}$$

In our experiments, as will be explained in Chapter 5, we have dealt with both multi-task and single-task models. Specifically, the multi-task models aimed at modeling simultaneously the cohort classification task (a multiclass problem) and the tumor prediction task (a binary problem). In these cases, we added the losses of both tasks together. When we were training models with weighted pooling, we also included L_1 regularization on the weights of the pooling operation to induce sparsity.

We have used AdamW (LOSHCHILOV; HUTTER, 2017) for optimizing all models and cosine annealing with warm restarts (LOSHCHILOV; HUTTER, 2016) for dynamically updating the learning rate during the training process. AdamW is similar to Adam, with the difference that the weight decay parameter is decoupled from the learning rate and is equivalent to L_2 regularization. Adam computes individual step sizes for the parameters based on the learning rate and on estimates of the moments of the gradients (KINGMA; BA, 2017). Cosine annealing is a scheduling method for the learning rate that reduces the overall learning rate at each batch from an initial to a final value proportionally to half period of a cosine function. With warm restarts, the learning rate is set again to its initial value after a certain number of epochs have passed and another cycle of cosine annealing begins. This sudden increase in the learning rate simulates a restart of the optimized parameters to a nearby region. If the current parameters do not correspond to a sufficiently wide local minimum of the loss function, the increase in the learning rate can cause the parameters to jump out of it, which is desirable as it is believed that wider local minima are more likely to generalize well in the test set. The learning rate update equation is given by

$$\eta_t = \eta^i_{min} + \frac{1}{2}(\eta^i_{max} - \eta^i_{min})(1 + \cos(\frac{T_{cur}}{T_i}\pi)), \tag{4.7}$$

where T_{cur} is the number of epochs that have passed since the last restart and T_i is the number of epochs between two restarts. We note that a similar approach of using cosine annealing with warm restarts was used for training GNNs applied to scRNA-seq data (YIN et al., 2022). In all cases, we have used 5 cycles of warms restarts, which corresponds to 31 epochs.

In our experiments, unless stated differently, we have always used 5-times repeated holdout for evaluating performance. Holdout consists in separating the dataset into disjunct train and test sets, each time with a different set of samples going to each of the splits. The train set is used for fitting the model, including the selection of hyperparameters, and the test set is only used to evaluate the final performance of the models. This process is repeated 5 times, and the reported performances are generally the mean value from all runs and the standard deviation. In our case, we separated 80% of the data for training and 20% for test in each run.

Moreover, we have further splitted the training set into the actual training set used for fitting the model, and a validation set used for tuning the hyperparameters. In terms of proportions from the original dataset, we have used 60% of the data for fitting the models, 20% for the validation set, used for choosing the hyperparameters, and 20% for testing the final selected model. The final selected model was fitted using the whole training set (consisting of 80% of the samples) and the select set of hyperparameters. In all models considered in our experiments, we have tuned three hyperparameters using random search with 8 samples: the initial learning rate, the weight decay associated with AdamW, and the L_1 regularization weight, when appropriate. We have used the *Ray Tune* (LIAW et al., 2018) library to execute the tuning steps. The set of hyperparameters that obtained the minimum loss value in the validation set was selected.

Most often, we report here the F1-score macro-averaged over the classes, since it is a more appropriate metric to be used in imbalanced scenarios. The F1-score of a class is simply the harmonic mean between the precision and the recall for that class. The precision and recall, in turn, can be computed using the confusion matrix entries. Specifically, the precision is given by the formula $tp/(tp + fp)$, and recall is given by $tp/(tp + fn)$, where tp, fp and fn stand for the number of true positives, false positives and false negatives, respectively. The F1-scores are computed per class, and it is necessary to reduce it to a single value that encompasses F1-scores of all classes. We have done it using the macro average, which consists in simply averaging the F1-scores obtained for each class. The macro average was chosen because we wanted to assign equal weights to every class, independently of the number of samples that they had.

4.4 Model Interpretation using Saliency Maps

A classification model produces scores for each of the classes being classified which indicate how likely it is that the input belongs to that class. In general, we can write $\mathbf{S} = f_\theta(\mathbf{x})$, where θ indicates the set of parameters, $\mathbf{x}$ is the input vector and $\mathbf{S} = (S_1, S_2, ..., S_C)$ is the output vector containing scores for each of the C classes. The fundamental idea of the saliency maps proposed by Simonyan, Vedaldi and Zisserman (2013) is based on the first order approximation of the score S_c produced by the model given an input $\mathbf{x}_0$ for a class c,

$$S_c(x_0) \approx \mathbf{w}^T \mathbf{x}_0 + b \tag{4.8}$$

The value of **w** is then given by the gradient

$$\mathbf{w} = \left.\frac{\partial S_c}{\partial \mathbf{x}}\right|_{x_0} \tag{4.9}$$

One can therefore interpret the absolute values of the gradient as an indication of which input features need to change the least to produce the greatest variation in this class score. It is reasonable then to consider those as the main features in a model's decision. The absolute value of the gradient of a class score with respect to an input sample is referred to as the class saliencies. Mostavi et al. (2020) used saliency maps to analyze a CNN proposed for classifying types of cancer. In their work, they summarized the main genes for each cohort by taking the mean value of each gene saliency over all the samples that belong to the cohort. This implicitly assumes that the models are basing themselves on similar sets of genes for making decisions related to samples of the same cohort. Although this is reasonable in the context of gene expression classification where one expects to find a consistent set of genes, in our work we generally analyzed the distribution of saliencies across samples. We also note that it can be useful to consider the gradient itself instead of only its magnitude, as it might indicate that the model is affected in different ways by the same input feature, depending on the given sample. This assertion follows from Eq. 4.8, where a positive variation in the input feature can produce either a positive or a negative change in the score, depending on the sign of the elements of **w**.

In our work, we have also considered the saliencies of the embeddings at the coarsest levels of the graph. This was partially inspired by the work in Hayakawa et al. (2022) (see also Section 3), where they propose a model that allows interpretation at the level of the KEGG pathways. In it, the authors use one graph neural network per pathway in the KEGG database, where each node in the KEGG networks corresponds to a gene. The GNNs produce embeddings that are then aggregated together in a fully connected layer and used for classification of diffuse large B-cell lymphoma subtypes. They then show that SHAP values can be used for computing the importance of the embeddings generated by each GNN. This is reasonable, since the embeddings can be seen as the attributes of a new dataset that was the result of transforming the gene input data through the graph convolution layers, and SHAP values (as saliency maps) are traditionally used for analyzing the importances of the attributes of a dataset. Moreover, since there is as correspondence between the GNNs and the KEGG pathways, the importance of the embeddings can be interpreted as the importance of each pathway.

In our work, the GNN module of the model can be seen as a learned dimensionality reduction method that transforms the dataset into a lower dimensional space composed of the embeddings of each node of the most coarsened level of the graph. When we analyze these embeddings using the saliency methods as described above, we are therefore analyzing the inputs of this transformed dataset. Furthermore, since the graph convolutions and pooling are local operations, each supernode has a direct correspondence with the initial nodes (genes) that were clustered together during the pooling operations. Hence, by analyzing the influence of the supernode on the outputs (that is, by analyzing its saliencies) we are equivalently studying the effect that the cluster of genes has, as a group, in the outputs of the model.

In order to analyze the cluster of genes associated with a supernode, we perform an over-representation analysis (ORA) using the WebGestalt tool[3]. The basic idea of an ORA is to compute how likely it was that the given set of genes would overlap with another known gene set (for example, a pathway or a biological processes) if the genes were randomly selected from a reference set. In our experiments, we have generally used clusters of 128 genes and a reference set containing all protein-coding genes (since we have restricted ourselves to the STRING network). If, for instance, our set has 90 genes overlapping with a biological processes that encompasses 100 genes, then it is very unlikely that this was only due to random chance, and we can say that this biological processes is *enriched*. On the other hand, if we find that only 1 gene overlapped, then our set of genes is no more related to that biological process than a random set of genes is.

[3]https://www.webgestalt.org/

5 EXPERIMENTS ON THE GNNS WITH MULTIPLE POOLING LEVELS

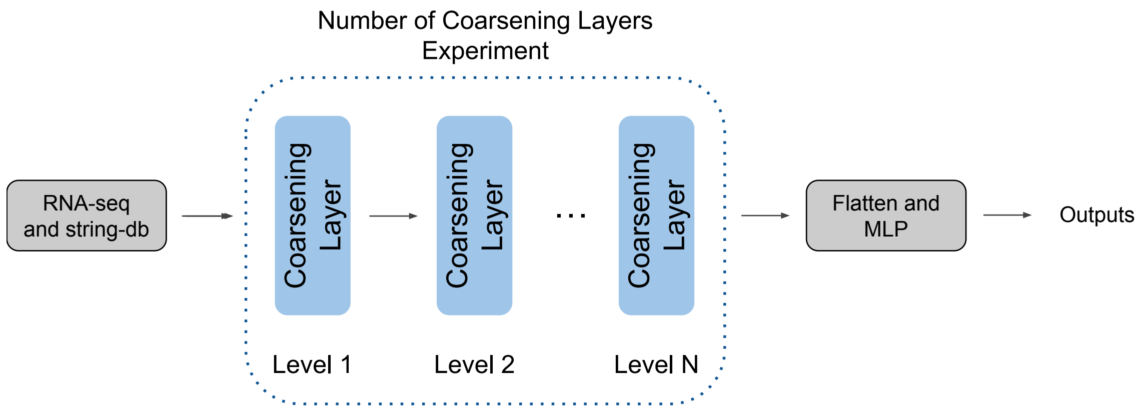
In this chapter, we explore two questions related to the use of GNN models with multiple levels of pooling. The first question is how coarsening affects the performance of the model in a variety of tasks. In a few related works, the models based on graph convolutions achieved performances higher than other state-of-the-art approaches and, in particular, against the use of neural networks. Our second question aims at evaluating the interpretability of these graph-based models built over the STRING network. A deeper description of these experiments is provided in Section 5.1. Following that, Sections 5.2 and 5.3 describe the results pertaining to the first question using various datasets, and Section 5.4 deals with the interpretability of one of the GNN models. Finally, Section 5.5 discusses the results and some of its limitations.

5.1 Overview of the Experiments

The experiments in this section are aimed at answering two questions related to GNN models based on gene expression data and biological networks. Firstly, we wish to understand how multiple levels of pooling affect the performance of the GNNs. This is inspired by some of the prior works that reported good performance of the GNNs when a single level of pooling was used. Yin et al. (2022) argued that a graph convolution applied over the original biological network improved performance. Furthermore, they reasoned that this improvement was because the convolution operation helped the model to handle noisy values of gene expression.

Therefore, we wondered whether further coarsening and pooling of the STRING network could improve the accuracy of the models. This leads to one set of experiments summarized in Figure 5.1, where N coarsening layers are shown to indicate that we experimented using up to N levels. More specifically, we evaluated the performance of the models using 1 to 6 coarsening layers. Moreover, we varied the structure of the coarsening layers in four different ways: only sum pooling, only weighted pooling, sum pooling with graph convolutions and weighted pooling with graph convolutions. These variations were designed to evaluate whether any changes in performance were either due to the convolutions or simply related to coarsening the graph. Lastly, at each coarsening layer, the convolutions double the number of output channels with respect to the number of input channels up to a maximum of 32 channels (this means that for 5 and 6 coarsening

Figure 5.1 – Experiments performed on the effect of the number of coarsening layers in the performance.

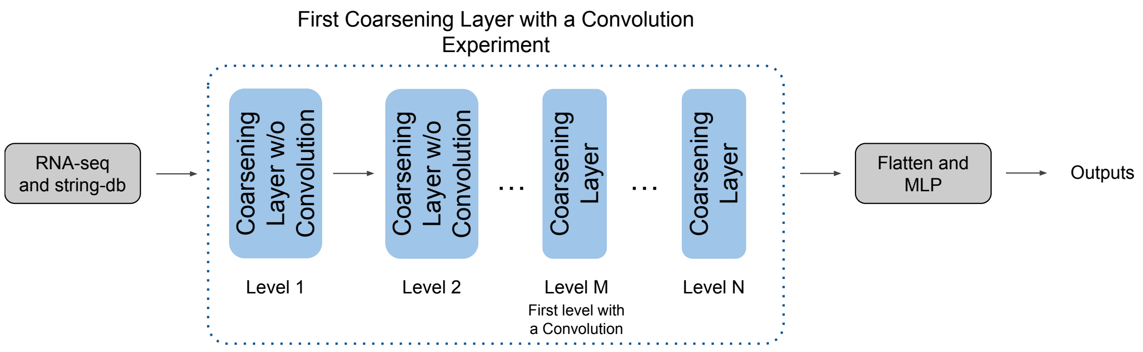


Source: The Author

layers, there are 32 features per output node). This might seem small, but is close to the value used in other works in the area. For instance, Li, Wang and Nabavi (2021) uses 5 output channels in their graph convolution layer.

Also aiming at exploring the performance of the GNN models, we considered how would graph convolutions and weighted pooling affect performance given that we wanted to coarsen the graph to obtain a specific number of supernodes, but only applying convolutions at the coarser levels of the graph. Biological processes have in general from tens to a few hundred genes associated with them and in order to enable an interpretation at this level, we need to coarsen the network until the clusters containing a few hundred genes are aggregated together under the umbrella of a single supernode embedding. For example, if we coarsen the STRING graph used in this work by merging two nodes at each step, then we can reduce 14133 nodes to 111 supernodes in seven coarsening layers. Assuming that we indeed want to generate 111 supernodes, each associated with circa 128 genes, by coarsening the graph seven times as described previously, we wish to answer whether there is any benefit in using graph convolutions or if simply aggregating the node embeddings by summing or taking their means is enough. This set of experiments is summarized in Figure 5.2. Furthermore, the graph convolution operation is very costly to apply at big graphs such as the STRING. Obtaining performance gains by using convolutions only at the coarser levels of the graph instead of using them at the original graph would enable a significant reduction in computational cost. Hence, we aimed at answering whether delaying the introduction of the convolutions to the last layers of the model could still retain performance. To evaluate that, we considered the sum-pooling and weighted-pooling schemes, as before.

Figure 5.2 – Experiments performed on the influence of the first coarsening layer where a graph convolution is performed.


First Coarsening Layer with a Convolution Experiment
RNA-seq and string-db
Coarsening Layer w/o Convolution
Level 1
Coarsening Layer w/o Convolution
Level 2
...
Coarsening Layer
Level M
First level with a Convolution
...
Coarsening Layer
Level N
Flatten and MLP
Outputs

Source: The Author

Finally, we have decided to use the performance of a neural network to compare our results against. The reasoning for this is that the NNs can be reasonably considered the state-of-the-art in terms of performance when dealing with gene expression data, as is discussed in Section 3.5.

For our second question, of the interpretability of these models, we analyzed one of the GNNs on a specific task explored in the performance evaluations. In particular, we hypothesized that the embeddings at the coarsened levels of the graph would allow meaningful interpretations of the model at a more abstract level, such as the important pathways, and that we could find meaningful biomarkers by analyzing the saliencies of the input genes. These experiments are shown in Section 5.4.

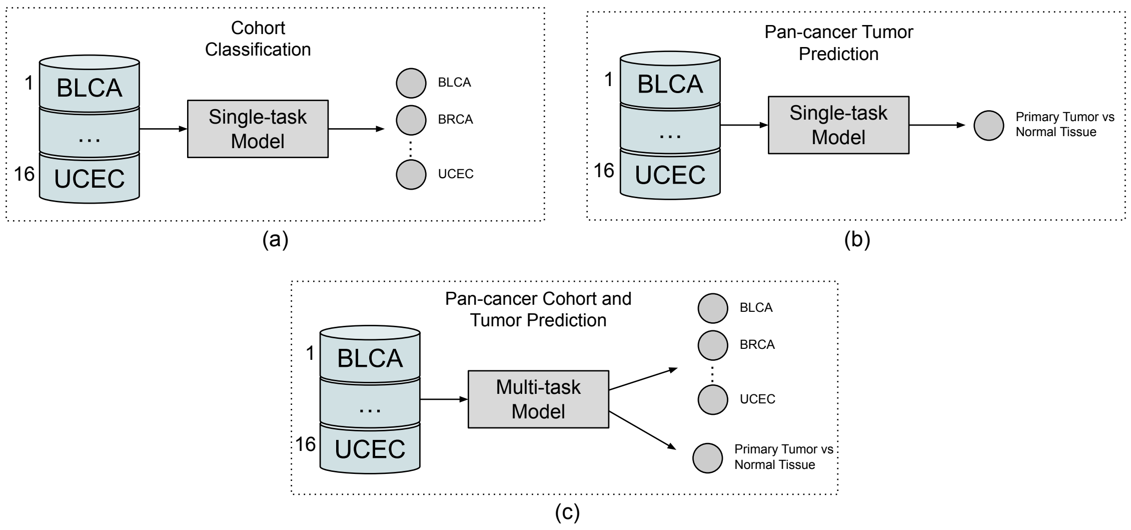
Finally, we have executed most of these experiments with GPUs available through the PCAD[1] infrastructure at INF/UFRGS.

5.2 Pan-Cancer Cohort Classification and Tumor Prediction

In this section, we consider the performance of the GNN models in three different pan-cancer classification scenarios: cohort classification (or tissue-of-origin classification), tumor prediction, and both scenarios simultaneously in a multi-task model. These tasks are summarized in Figure 5.3. In all cases, we have considered the two sets of experiments described in Section 5.1 in Figures 5.1 and 5.2, namely the study of the variation in performance as the number of coarsening levels is increased, and the variation in performance of different designs where the graph is coarsened without convolutions up to a level. The datasets were preprocessed as described in Section 4.1. Prior to the execution

[1]http://gppd-hpc.inf.ufrgs.br

Figure 5.3 – Pan-cancer classification tasks considered in this work: the multiclass task of identifying the tissue-of-origin of a sample (a), the binary task of predicting whether a sample corresponds to a tumor or not (b), and a multi-task version of the problem, where the model must learn both tasks simultaneously (c).

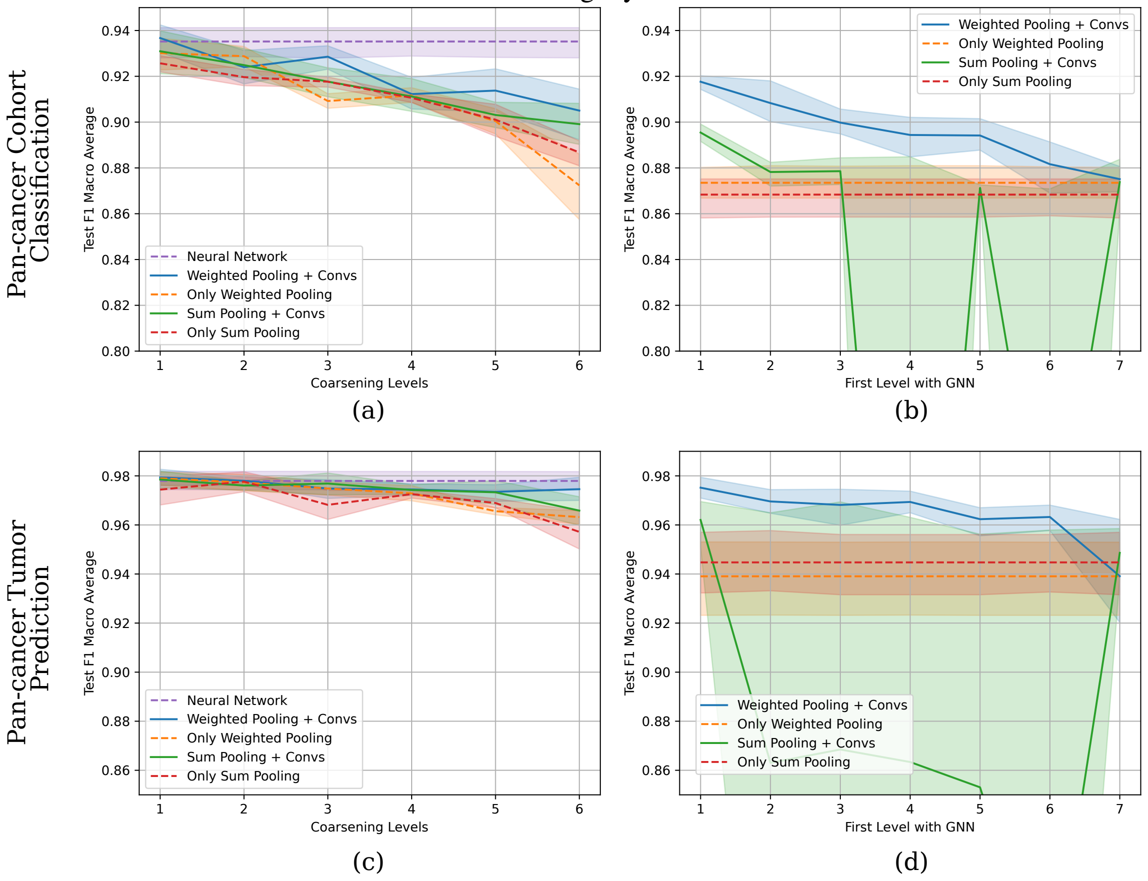


Source: The Author

of each experiment, however, we have standardized the gene expression data to have a 0 mean and a standard deviation of 1 across the samples. That is, in the pan-cancer experiments, we have standardized using the full set of cohorts, while for the cohort-specific experiments (Section 5.3) each cohort was standardized separately.

The results on the test sets obtained for the single-task models are shown in Figure 5.4. In both tasks, the fully-connected networks (represented by the dashed purple line) achieve the best performance, or are very close to it. Their mean F1 macro average score in the pan-cancer cohort classification task is 0.935, which is slightly below that of the weighted pooling model with one graph convolution, at 0.938. When the graph has more than one coarsening layer, the fully-connected network performs better. This is consistent with other previous work in the area that has found NNs in general to achieve the state-of-the-art performance (Section 3.5.1). For the pan-cancer cohort classification task, there is a clear downtrend in performance when the number of coarsening layers increase (Figure 5.4 - (a)). This downtrend, however, is diminished when both weighted pooling and convolutions are present in the coarsening layers. The weighted pooling model with graph convolutions, for instance, goes from a mean value of 0.937 at one level to 0.905 at six coarsening levels, whereas the performance drops down to 0.872 when only weighted pooling is used, without convolutions. A similar trend is seen in Figure 5.4 - (b), where there is a visible benefit in having the convolutions at the more detailed levels of the graph, particularly with weighted pooling. With sum pooling, however, the training often found

Figure 5.4 – Results on the pan-cancer tasks using single-task GNN models with multiple coarsening layers.



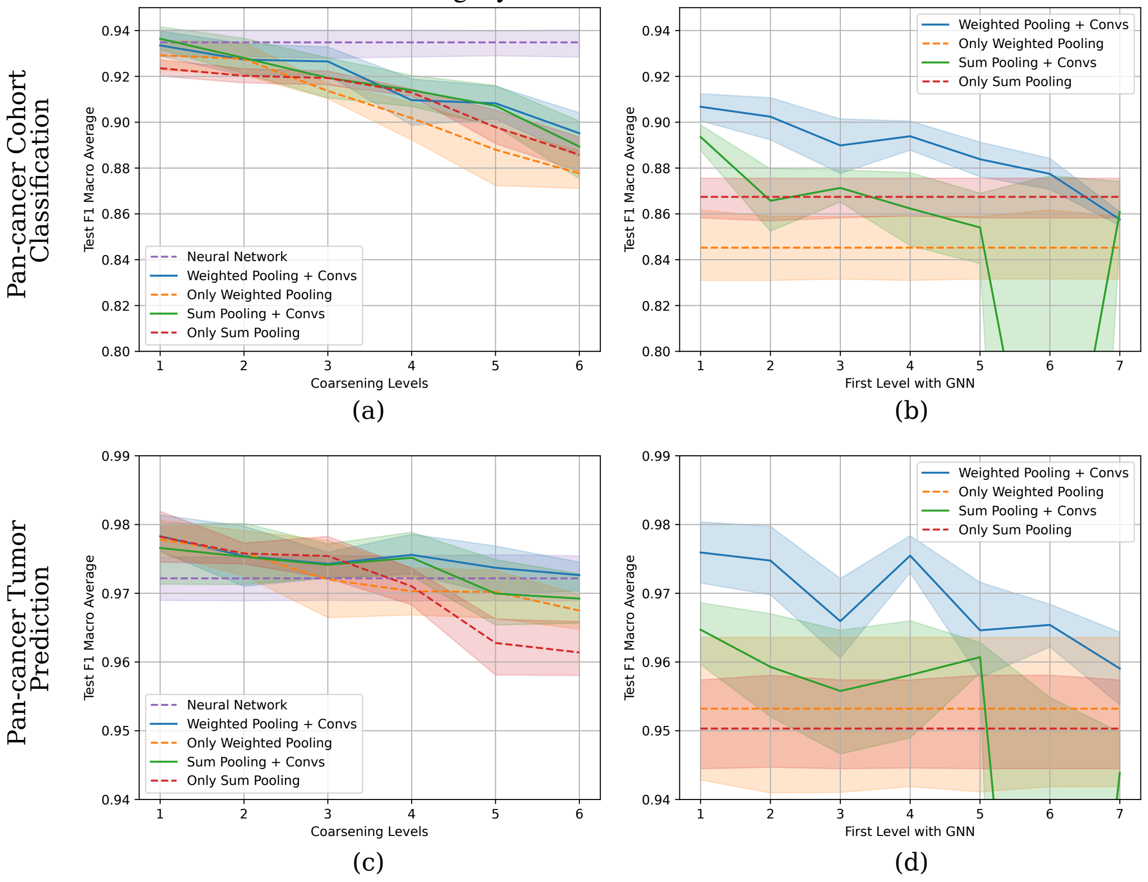
Source: The Author

poor solutions that produced lower performance (green full line in Figure 5.4 - (b)). To facilitate the visualization of the trends, we have fixed the limits of the y axis from values going from 0.8 to 0.95, which caused the two bad points to be cut out from the plot. In the worst case, the sum pooling operation achieved a mean value of 0.533, pulled down by a poorly fitted run. We note, however, that this execution was not overfitted, having performed poorly in the training set as well, with an accuracy on it of only 0.074.

When we consider the pan-cancer task of distinguishing tumorous from non-tumorous samples (Pan-cancer Tumor Prediction task in Figures 5.4 - (c) and 5.4 - (d)), there is a much less significant downtrend in performance when increasing the number of coarsening levels. With a single convolution layer and weighted pooling, the mean F1 score was 0.979, very close to the NN model (at 0.978). The model's performance then drops to 0.975 when six coarsening layers are used. The drop is more significant for the coarsening models that do not have convolutional layers: the sum-pooling-only model performance goes to 0.957 with six coarsening layers, from a value of 0.974 with one layer. As observed in the Pan-cancer Cohort Classification task, there is also a benefit in having the convolutions at the finer levels of the graph, although less marked here for the weighted pooling model with convolutions (Figure 5.4 - (d)). Similarly as well, we see the same pattern of poorer fits with the sum pooling model with convolutions, where a few bad training executions produced low-quality models.

We also trained a multi-task model with shared coarsening layers for the Pan-cancer Cohort Classification and Tumor Prediction tasks. The performance results for these models are shown in Figure 5.5. The observed trends in this case are similar to

Figure 5.5 – Results on the pan-cancer tasks using a multi-task GNN model with multiple coarsening layers trained for both tasks.



Source: The Author

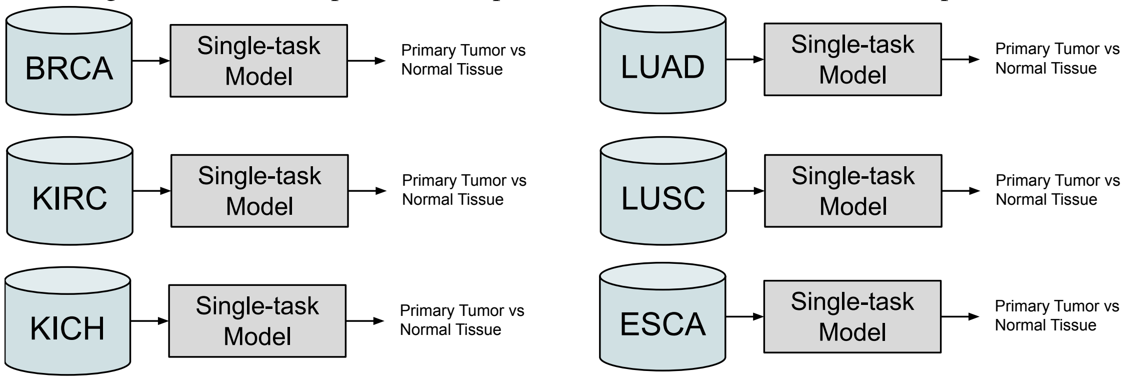
the ones found for the single-task models (a direct comparison between the single-task and multi-task performances is presented in Section 5.5.2). One can observe the same drop in performance as more coarsening layers are introduced, particularly for the Pan-cancer cohort classification task. The performance of the weighted pooling model with convolutions, for instance, drops from 0.934 when one coarsening layer is used to 0.895 when there are six layers. Interestingly, the best result in this task was obtained with the coarsening model with a single convolution and sum pooling (the green line in Figure 5.5 - (a)), with a mean F1 score of 0.936. We note however, that this result is quite similar to those of the NN and the analogous weighted pooling model, which had mean scores of 0.935 and 0.934, respectively, so that there is likely no real difference between their results. Similarly to the the single-task model, Figure 5.5 - (b) shows that there is a significant gain in using convolutions with weighted pooling at the finer levels of the graph instead of only at the coarser ones.

Finally, we also observe the phenomenon of poor model fits when sum pooling is used with the convolutions starting at the more coarse levels of the graph. When considering the multitask models for the Pan-cancer Tumor Prediction task (that is, tumor vs non-tumor prediction), the coarsened models in general had better performance than that of the NN (Figure 5.5 - (c)). While the mean F1 score of the multitask NN for tumor prediction was 0.972, both the pure sum pooling model and the weighted pooling with a single convolution model had mean F1 scores of 0.978. The performance drop for the models with only pooling, however, was more significant and from the fourth layer on they were already smaller than that of the neural network, with the sum pooling approach achieving a mean score of 0.961. Lastly, we also observe the same patterns as before in Figure 5.5 - (d), where the best performing models with seven coarsening layers are those that include both weighted pooling and convolutions, and the models with convolutions and sum pooling resulting in poor fits that cause significant drop in their average performances. In Section 5.5.2, we discuss whether there is any improvement in using either a multi-task or single-task model.

5.3 Cohort-specific Tumor Prediction

In this section, we describe the performance of the coarsening models on cohort-specific tumor prediction tasks, where the model needs to learn to distinguish tumorous from non-tumorous samples from a specific tissue. Once again, we have considered both experiments described in Figures 5.1 and 5.2. We have selected a total of six specific cohorts for analysis. BRCA, KIRC, LUAD and LUSC were chosen as they were among the top datasets with more examples of Normal Tissue and Primary Tumor samples. Furthermore, we selected two cohorts that possess fewer samples in order to evaluate whether this would entail any notable difference. Specifically, we chose ESCA and KICH as the small-sample-size datasets. The set of cohort-specific classification tasks is summarized in Figure 5.6. Later on, in Chapter 6, we perform a direct comparison between cohort-specific models and the performance of pan-cancer tumor prediction models on

Figure 5.6 – Cohort-specific tumor prediction tasks considered in these experiments.


BRCA
Single-task Model
Primary Tumor vs Normal Tissue
KIRC
Single-task Model
Primary Tumor vs Normal Tissue
KICH
Single-task Model
Primary Tumor vs Normal Tissue
LUAD
Single-task Model
Primary Tumor vs Normal Tissue
LUSC
Single-task Model
Primary Tumor vs Normal Tissue
ESCA
Single-task Model
Primary Tumor vs Normal Tissue

Source: The Author

the cohort-specific tasks.

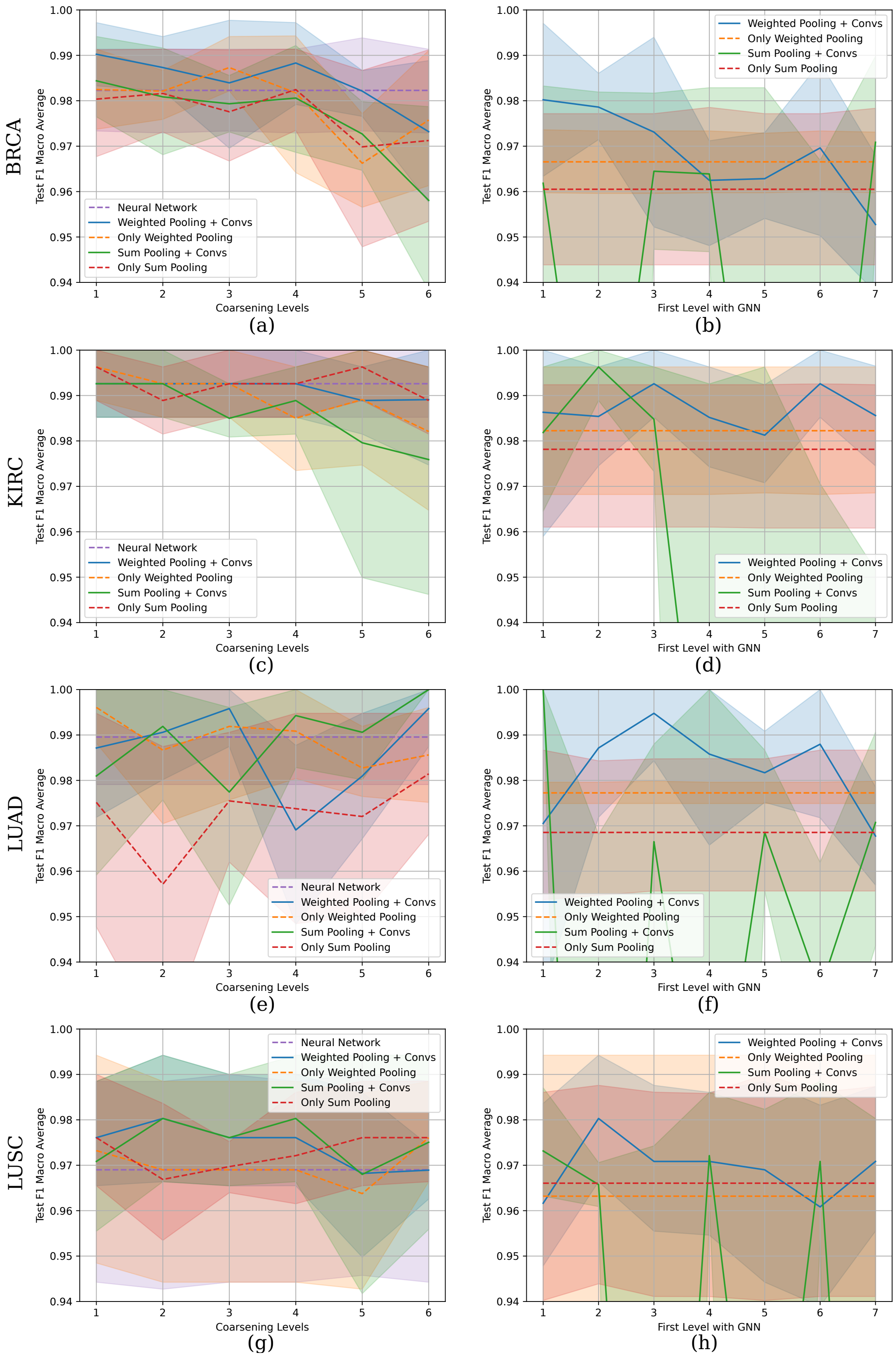
The results for the four mid-size datasets are shown in Figure 5.7. We can observe a similar trend of performance reduction on the BRCA dataset that was seen in the pan-cancer models. In general, however, the mean performance of the coarsening model with weighted pooling and convolutions was slightly above that of the NN, reaching a mean F1 score of 0.990 with 1 coarsening layer against 0.982 for the NN. In the other datasets, there was not any clear trend and the results varied around the same values. The exception is again the coarsening model with sum pooling and convolutions, which shows here as well a tendency to fit poorly particularly when the convolutions start at the less detailed levels of the graph, producing mean results that are pulled down by the poor fits.

When we considered the two datasets with fewer examples, we are not able to observe any particular trend, other than all the models achieving a performance that is quite similar to each other (Figure 5.8). This could be due to the greater variation in the results, since it would obscure any tendency not sufficiently strong. The results of the NN model trained on the ESCA dataset, in particular, ranged from a maximum of 0.921 to a minimum of 0.635, and similar ranges were found for the other models as well. Because the datasets are very small, it is reasonable that there was so much variation. The results on the KICH dataset also showed great variability, although smaller than that of ESCA.

5.4 Interpretation of the Tumor Prediction and Cohort Classification Model

The tumor prediction and cohort classification task described in Section 5 can be used to construct models that can be interpreted in order to extract meaningful biomarkers and pathways. With this in mind, we trained a model using the architecture containing

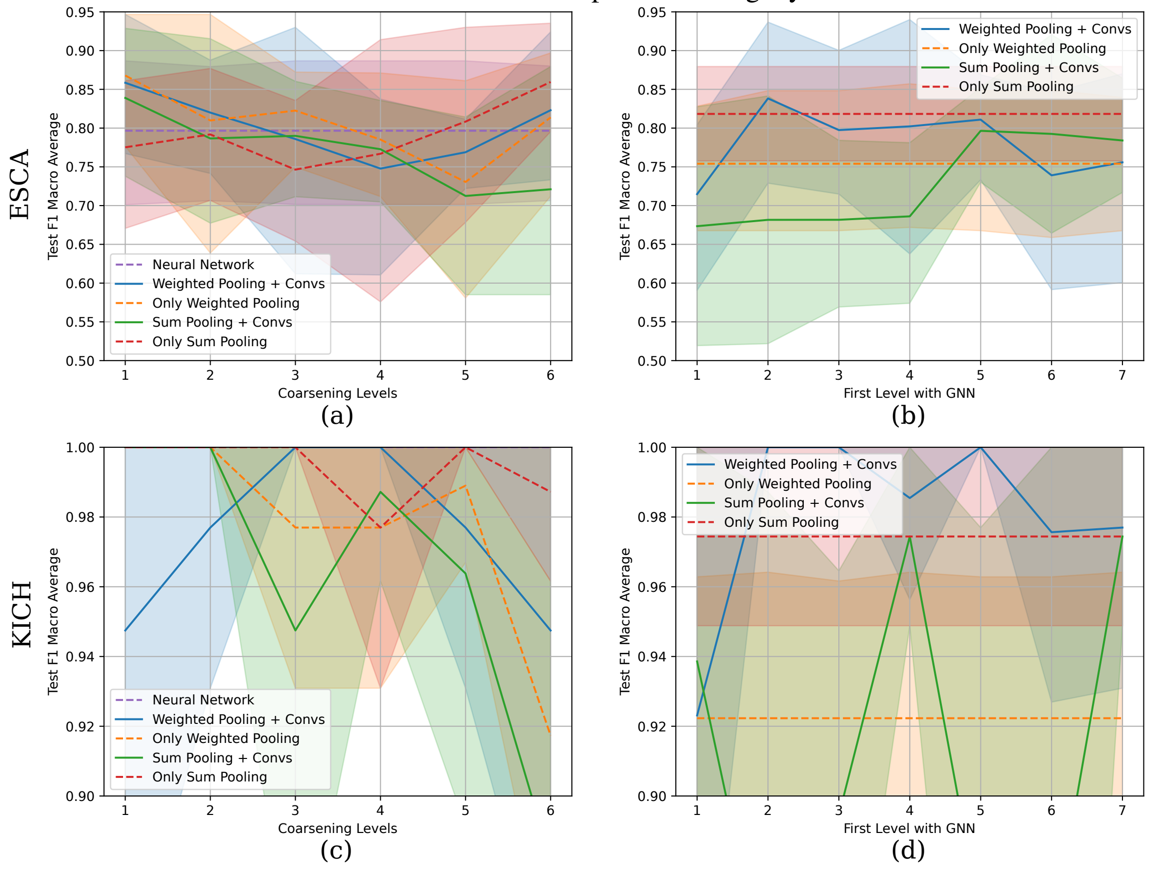
Figure 5.7 – Results on the cohort-specific Tumor Prediction tasks using GNN models with multiple coarsening layers.



Source: The Author

seven coarsening levels, with weighted pooling and GNNs starting at the level 5 and fitted it from scratch on a training set. Other architectures could also be explored, but we

Figure 5.8 – Results on the cohort-specific Tumor Prediction tasks of the small datasets, using GNN models with multiple coarsening layers.

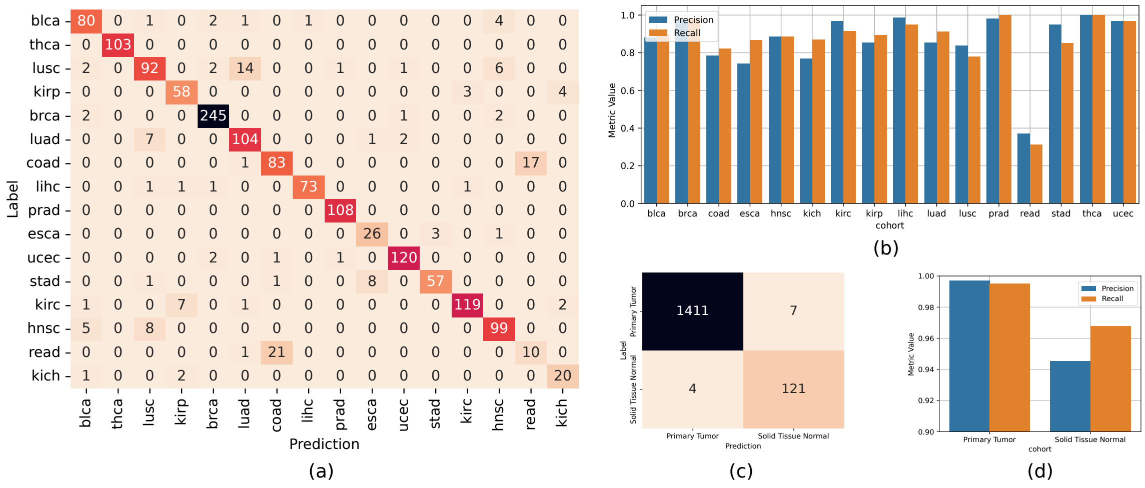


Source: The Author

selected this one because the supernodes at the final coarsening levels were associated with 128 genes, which is a good number of genes to perform over representation analysis (ORA), while retaining sufficient performance. Furthermore, this architecture is interesting because it represents a significant reduction in the dimensionality of the original network without the costly graph convolutions at the more detailed levels of the graphs. Specifically, the nodes at the fifth coarsening level (the first where a convolution is performed in this model) correspond to clusters of 16 genes, aggregated using only weighted pooling and non-linearities. Since there are only three convolutional layers, the model is more easily interpretable as well, since it induces 8-dimensional embeddings at each node of the coarsest graph. Using the model possessing seven coarsening levels, all with convolutions, would result in 32 dimensions per node in our case, which could complicate the analysis. Finally, it is interesting to note that in comparison with the neural network model, this architecture has less than 10% of the number of learnable parameters.

Before interpreting the model to extract biomarkers, it is important to verify if it is able to classify the dataset correctly. We show in Figure 5.9 confusion matrices for cohort and type classification along with relevant metrics. One can see that the model is able

Figure 5.9 – Test results on cohort classification and the tumor prediction tasks using a multi-task model. The confusion matrix for the Pan-cancer Cohort classification is shown in (a) together with the precision and recall obtained for each category in (b), whereas the confusion matrices and precision-recall plots are shown in (c) and (d) for the Pan-cancer Tumor Prediction task.



Source: The Author

to achieve almost perfect separation among tumorous and non-tumorous samples, with an accuracy over all samples, including all cohorts, of 99.3%, which is on the level of other state-of-the-art works. The model is also able to distinguish appropriately between most of the cohorts, but performs poorly for some of them (Figures 5.9-(a) and 5.9-(b)). In particular, most of the error is in distinguishing samples for the READ and COAD cohorts. This is also the case in other works, such as Mostavi et al. (2020). One could argue that this is due to the lack of READ samples, however, KICH and ESCA cohorts also possess a similar amount of examples and show better recall and precision. A more likely explanation is that these cancers share similar processes due to their anatomical proximity.

As a first step in interpreting the model, we consider here the ‘importancies’ assigned by the model to each of the input genes in the Tumor Prediction task. To do that, we computed the saliencies of the input genes using the guided backpropagation technique, as explained in Section 4.4. The distribution of saliencies showed a power law, with a few genes showing a significant higher saliency than the remaining ones. Analyzing the top ranked genes, we find various known cancer-related genes. The top 10 genes are summarized in Table 5.1.

By searching in the literature, we found all of these genes to have been previously studied in the context of cancer in various forms. More interestingly, five of them were

Table 5.1 – Main genes obtained through the saliency analysis of the multi-label model for Primary Tumor vs Normal Tissue classification.

Rank	STRING Protein Id	Gene Symbol
1	ENSP00000436785	*SLC35F2*
2	ENSP00000325808	*LRRN4CL*
3	ENSP00000416508	*ATP13A3*
4	ENSP00000309432	*RETREG3*
5	ENSP00000360540	*CEP55*
6	ENSP00000428263	*DGLUCY*
7	ENSP00000437550	*LATS1*
8	ENSP00000272348	*SNRPG*
9	ENSP00000344456	*CTNNB1*
10	ENSP00000478783	*LTN1*

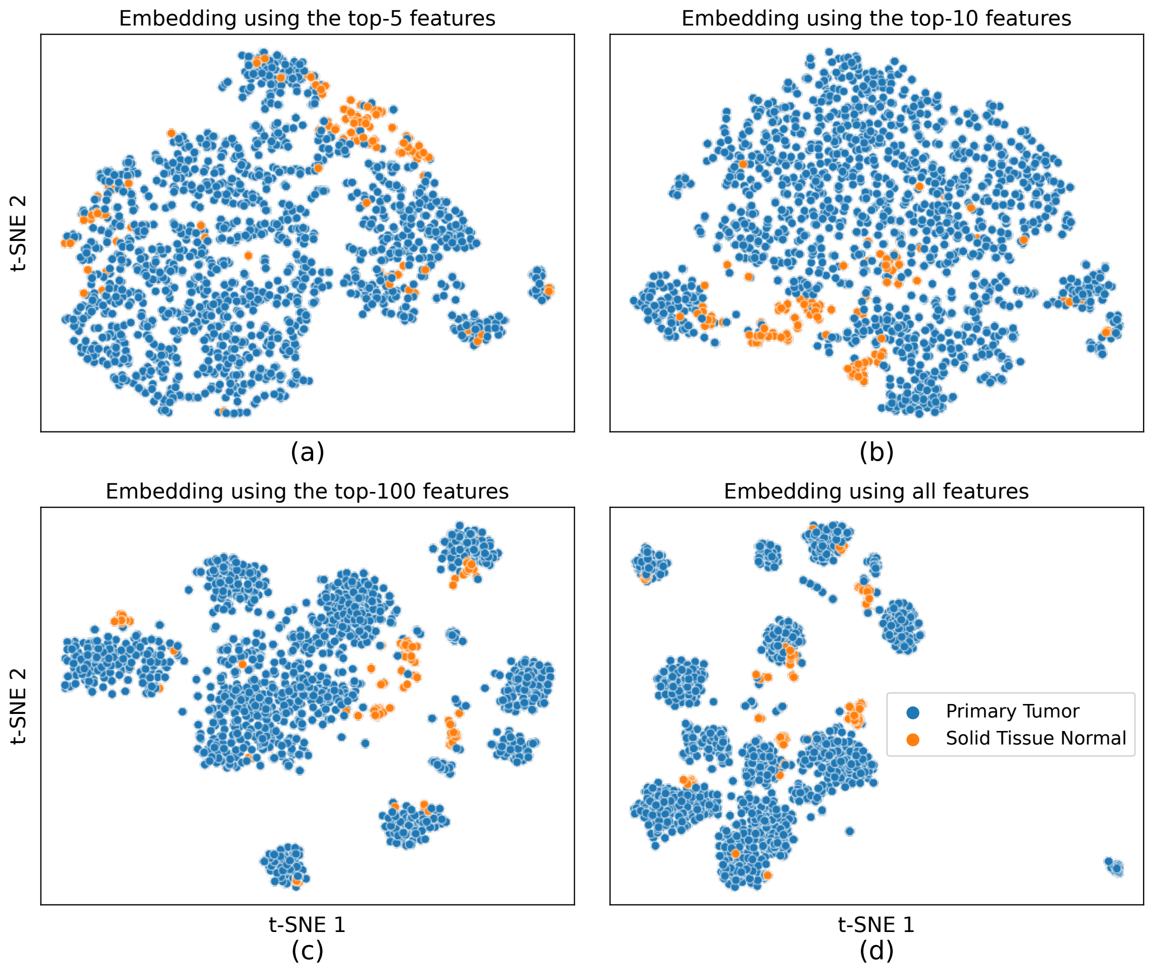
Source: The Author

associated with more than one type of cancer, indicating that the pan-cancer tumor prediction model is at some level aiming at general cancer genes, instead of building a list of cohort-specific biomarkers. One of these five genes is *SLC35F2*, which was found to be highly expressed in various human cancers (WINTER et al., 2014). *LATS1*, which encodes the Large Tumor Suppressor Kinase 1, was also associated with multiple tumors and is related with cancer cell growth (PAN et al., 2019). *SNRPG* also plays a role in tumor development and has been associated with various human cancers. Furthermore, it has been suggested to have potential for oncogenic drug discovery (MABONGA; KAPPO, 2019). Lastly, mutations in the *CTNNB1* genes are related to multiple cancer types (GAO et al., 2018) as well and *CEP55* over-expression was previously correlated with poor prognosis of various tumor types (JEFFERY et al., 2016).

We found the other genes to be associated with cohort-specific cancers. One of these, *LRRN4CL*, was recently associated with pulmonary metastasis in mice and correlates with decreased survival of melanoma patients (WEYDEN et al., 2021). *ATP13A3* has been considered as a biomarker for pancreatic cancer therapies (MADAN et al., 2016) and is associated with a decreased overall survival in pancreatic cancer (SEKHAR; ANDL; PHANSTIEL, 2022). *RETREG3* (*FAM134C*) is a member of the *FAM134* family (REGGIO et al., 2021). One of the members of this family, *FAM134B*, has been studied in the context of colorectal cancer (KASEM et al., 2014). Finally, *DGLUCY* is related with progression of gastric cancer (ZHU et al., 2019) and *LTN1* was linked with ovarian cancer prognosis.

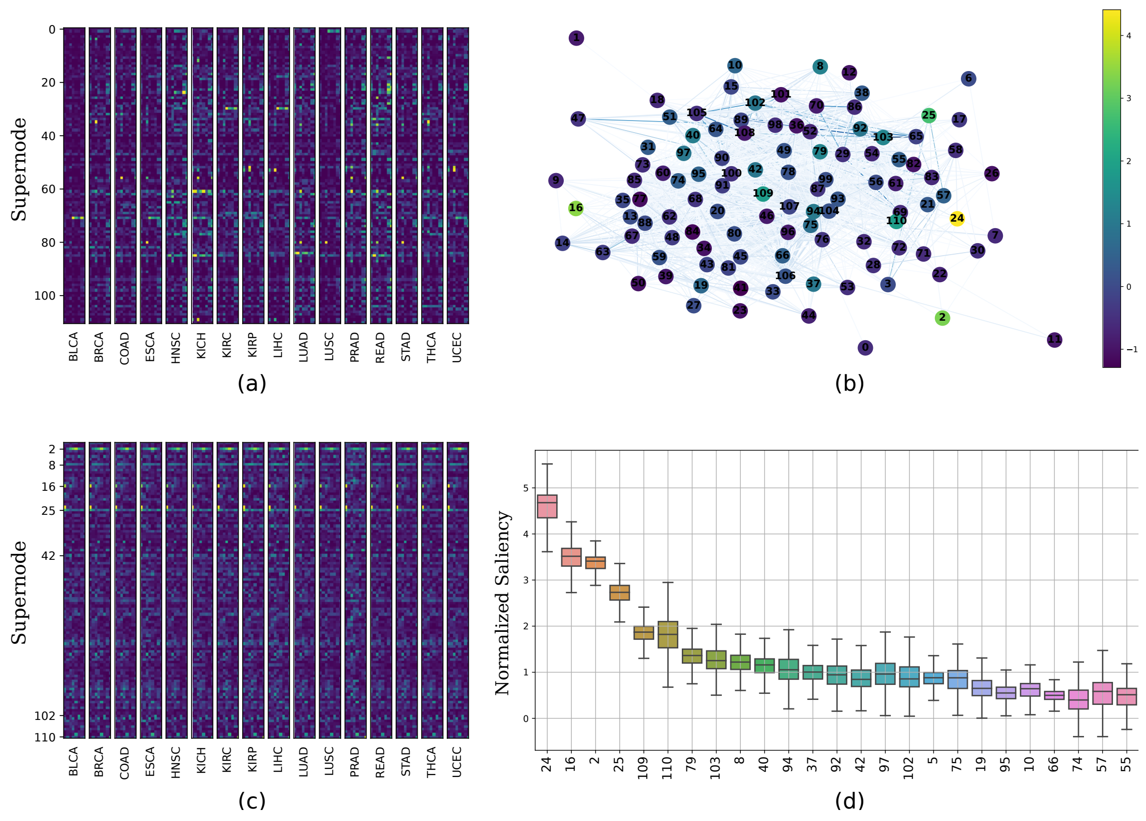
To further confirm that the ranking of genes obtained was significant, we decided to evaluate whether the top genes were able to preserve the separation between primary tumor and normal samples in the datasets. For that, we produced embeddings of the data using only the top ranked genes. Figure 5.10 presents the results for 5, 10 and 100 hundred top genes, as well as the embedding produced using all the genes. It can be seen that with 10 and 100 genes groups become visible in the embeddings. With 5 genes, although some clustering can be perceived, the normal tissue samples are more spread out through the primary tumor samples. Increasing the number of features also make the various primary tumor sample clusters more visible. These clusters are associated with the 16 cohorts present in the dataset, but we did not color the plot using them to make the differentiation related to tumor vs non-tumor samples more apparent.

Figure 5.10 – Embeddings of the dataset produced using the top 5 (a), 10 (b) and 100 (c) genes obtained for the sample type classification task. The embedding produced using all the input genes is shown in (d). The blobs are associated with the different cohorts present in the dataset, but are not differentiated here to avoid cluttering the plot.)



Source: The Author

Figure 5.11 – Example embeddings (a) and their saliencies (c) produced when distinguishing normal from tumorous samples. Each row corresponds to a supernode in a coarsened version of the STRING. In (b), we show the complete coarsened graph with its 111 vertices colored by their saliencies. The distribution of each supernode saliency over the dataset examples is shown in (d).



Source: The Author

Having confirmed that the GNN model was able to find significant and biologically meaningful input genes, we turned our attentions towards an interpretation of the model supernodes in the Tumor Prediction task. Recall that the output of the coarsening layers of the model considered here consists in 8-dimensional embeddings, one for each of the 111 supernodes in the coarsest graph. Furthermore, each supernode is primarily associated with the cluster of protein-encoding genes in the original STRING network that were aggregated by the coarsening layers, until arriving at the particular supernode. We computed the saliencies for each of the embeddings in the same way as done with the input genes, which resulted in a 111×8 matrix of supernode saliencies. Figures 5.11-(a) and 5.11-(c) show a set of examples of the resulting supernode embeddings and their corresponding saliencies for the Tumor Prediction task. We randomly selected one example from each cohort.

What is interesting to note is that, independently of the cohort, the distinction between tumorous and normal samples are always related to a small set of supernodes, even though the embeddings differ depending on the cohort. Figure 5.11-(d) further confirms this, where we show the distribution of the normalized values of the saliencies. Specifically, for each example individually, we normalized the values of the 128 saliencies (one for each supernode) to zero mean and unit standard deviation, so that their values are comparable across the examples. If the model were using different processes to predict tumor samples coming from different cohorts, then we would expect that the saliencies would show a multimodal distribution or at least be more spread over the importance range. However, we see that the top-ranked supernodes are consistently in the top.

The question remains to whether these supernodes are related to meaningful sets of genes. To evaluate this, we performed an over-representation analysis using WebGestalt [2] on the clusters of input genes associated with the top-4 supernodes. The results, presented in Table 5.2, show that the supernodes are enriched with respect to a few biological processes, some of which have been studied with relation to various cancer types. The mitochondrial respiratory chain complex assembly gene set, associated with cluster 25, for example, has been studied in the context of cancer aggressiveness (SIMONNET et al., 2002) and tumorigenesis (LEMARIE; GRIMM, 2011). With respect to supernode 16, the cluster associated with it was significantly over-represented in the glycoprotein metabolic process, which is defined in GeneOntology (GO) as all the pathways and chemical reactions involving glycoproteins[3]. Searching the literature, we found the work by Kailemia, Park and Lebrilla (2017), which reviewed various glycoproteins biomarkers that are used for monitoring and screening patients with a wide sort of cancer types. Supernode 2 was also associated with interesting processes, such as microtubule anchoring. The GO entry for microtubule anchoring states that it encompasses any process where a microtubule is maintained in a specific cell locations [4]. We found that there is indeed evidence suggesting the existence of dysfunctions in microtubule-related processes in cancer cells (DRáBER; DRáBEROVá, 2021). Interestingly, microtubules have also been targeted for anticancer therapies for decades (DUMONTET; JORDAN, 2010). Finally, the gene set relating to the isoprenoid metabolic process was over-represented in the genes associated with supernode 24. According to the GO entry for the process, it comprises the chemical processes and pathways involving isoprenoid components [5]. Following Wiemer, Hohl and Wiemer (2009), isoprenoid biosynthesis is related to cancer cell growth and metastasis,

[2]https://www.webgestalt.org/
[3]https://www.ebi.ac.uk/QuickGO/term/GO:0009100
[4]https://www.ebi.ac.uk/QuickGO/term/GO:0034453
[5]https://www.ebi.ac.uk/QuickGO/term/GO:0006720

Table 5.2 – Gene sets obtained from the over-representation analysis performed on the gene clusters associated with the supernodes selected through their saliencies on the test data.

Supernode	**Gene Set**	**Description**	**Enrichment Ratio**	**FDR**
24	GO:0008202	steroid metabolic process	27.130	<2.2e-16
	GO:0006720	isoprenoid metabolic process	20.738	<2.2e-16
	GO:0062012	regulation of small molecule metabolic process	8.9674	8.9462e-12
	GO:0042737	drug catabolic process	11.341	2.1444e-6
16	GO:0006022	aminoglycan metabolic process	34.386	<2.2e-16
	GO:0009100	glycoprotein metabolic process	20.225	<2.2e-16
	GO:1903509	liposaccharide metabolic process	14.361	2.7721e-8
2	GO:0034453	microtubule anchoring	62.127	1.2218e-7
	GO:0044839	cell cycle G2/M phase transition	7.5836	0.018845
25	GO:0010257	NADH dehydrogenase complex assembly	73.730	<2.2e-16
	GO:0033108	mitochondrial respiratory chain complex assembly	49.153	<2.2e-16
	GO:0009141	nucleoside triphosphate metabolic process	19.269	<2.2e-16
	GO:1902600	proton transmembrane transport	18.420	<2.2e-16
	GO:0099132	ATP hydrolysis coupled cation transmembrane transport	24.821	8.6641e-9

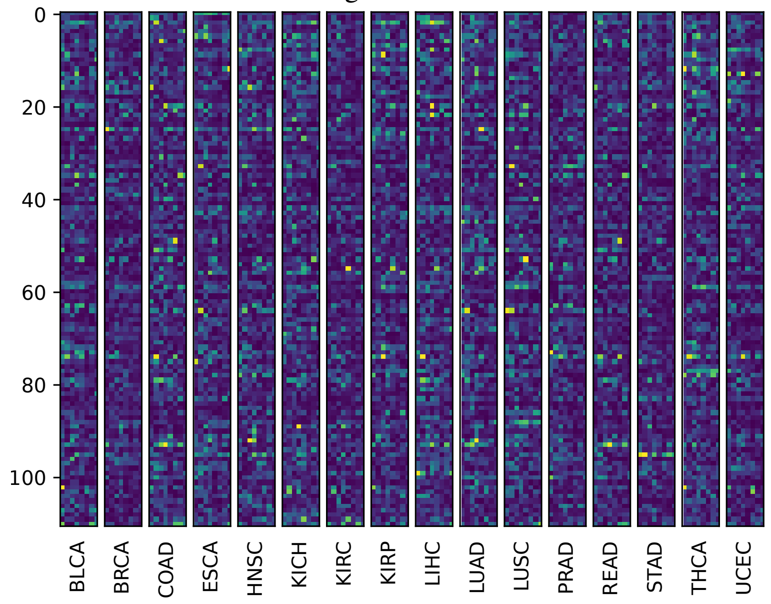
Source: The Author

making it a target for anticancer therapies.

Previously, we were describing the results of the analysis performed for the Tumor Prediction task. However, we have also considered the supernode saliencies with respect to the Cohort Classification task. These saliencies are shown in Figure 5.12 for the same set of examples considered in Figure 5.11. Differently from the Tumor Prediction task, the embedding saliencies vary significantly depending on which cohort is given as input (Figure 5.12). This is an indication that the model bases itself on different biological processes in order to determine into which cohort a sample falls.

To verify this, we used t-SNE to construct a 2-dimensional plot of the samples based on their saliency values, with the goal of visualizing whether samples from the same cohort would share similar saliencies (that is, whether they would be close together in the t-SNE plot). For each sample, we recorded the 111×8 matrices of embeddings produced at the supernodes and the corresponding saliency matrices. We flattened the

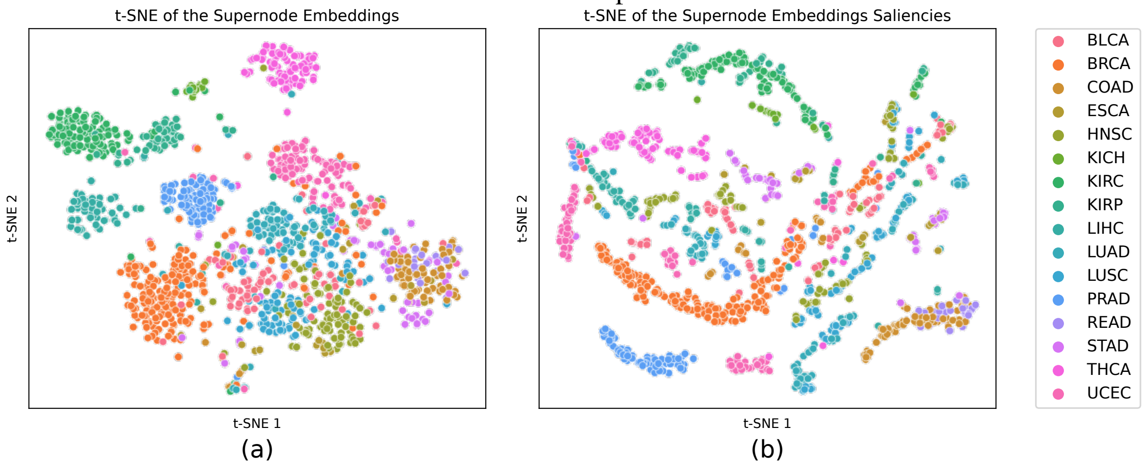
Figure 5.12 – Examples of the saliencies of the embeddings produced by classifying samples between the different cohorts available. These saliencies correspond to the same examples in Figure 5.11



Source: The Author

representations so that we obtained two matrices each with 888 columns. t-SNE was then used to reduce the 888 dimensions to a plane so that we could visualize whether the supernode embeddings and their saliencies would cluster by cohort. The resulting t-SNE plot for the supernode embeddings and their saliencies are shown in Figures 5.13 - (a) and (b), respectively. We use the plots computed from the test samples to show that there is no overfitting to the training data. In Appendix C, we also show and discuss the t-SNE plots produced when considering the gradients themselves with respect to the supernode embeddings besides the saliencies.

Figure 5.13 – Supernode embeddings and their saliencies produced when predicting the cohort of the test samples.

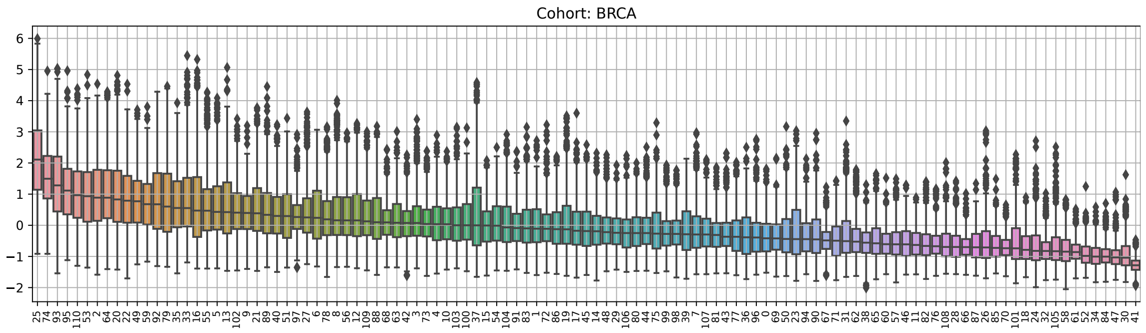


Source: The Author

In general, it is possible to perceive that the clusters in the plot correlate with the cohorts. Interestingly, we note that the READ samples are clustered together, but in the same space as part of the COAD samples. If we look at the model's performance in Figure 5.9 - (b) we note that the READ samples are indeed the most misclassified, and they are often confused with COAD samples, as can be seen in the confusion matrix in Figure 5.9 - (a).

Exploring the biological meaning of the clusters in the problem of cohort classification is more intricate than in the case of primary tumor identification. The major issue we found was that many more supernodes were considered relevant than in the case of Tumor Prediction. To illustrate this, Figure 5.14 shows the standardized saliencies obtained when considering the BRCA output and training samples in the Cohort Classification task. Even though one can see that supernode 25 is on average more salient than the others, the difference between the top supernodes and the others is significantly less than in the Tumor Prediction task in Figure 5.11. This indicates that the model's decision is more complex here than when distinguishing tumorous from non-tumorous samples, which complicates the analysis. One of the reasons for that could be related to biologi-

Figure 5.14 – Distribution of supernode saliencies obtained for the BRCA output in the Cohort Classification task.



Source: The Author

cal aspects that are not properly modeled by the clusters of supernodes available for the model, so that the NN that gets the embeddings as inputs needs to mix them in various ways instead of discarding a greater number of supernodes. Another reason for the more homogeneous distribution of saliencies could be related to how the model's probability for one cohort is affected by the other cohorts. The NN can give a higher probability for a cohort because some of its inputs point positively to it, but also because other inputs point negatively to other cohorts. Therefore, reaching a conclusion about what exactly is leading to the model's decisions would require a deeper analysis of the biological processes obtained.

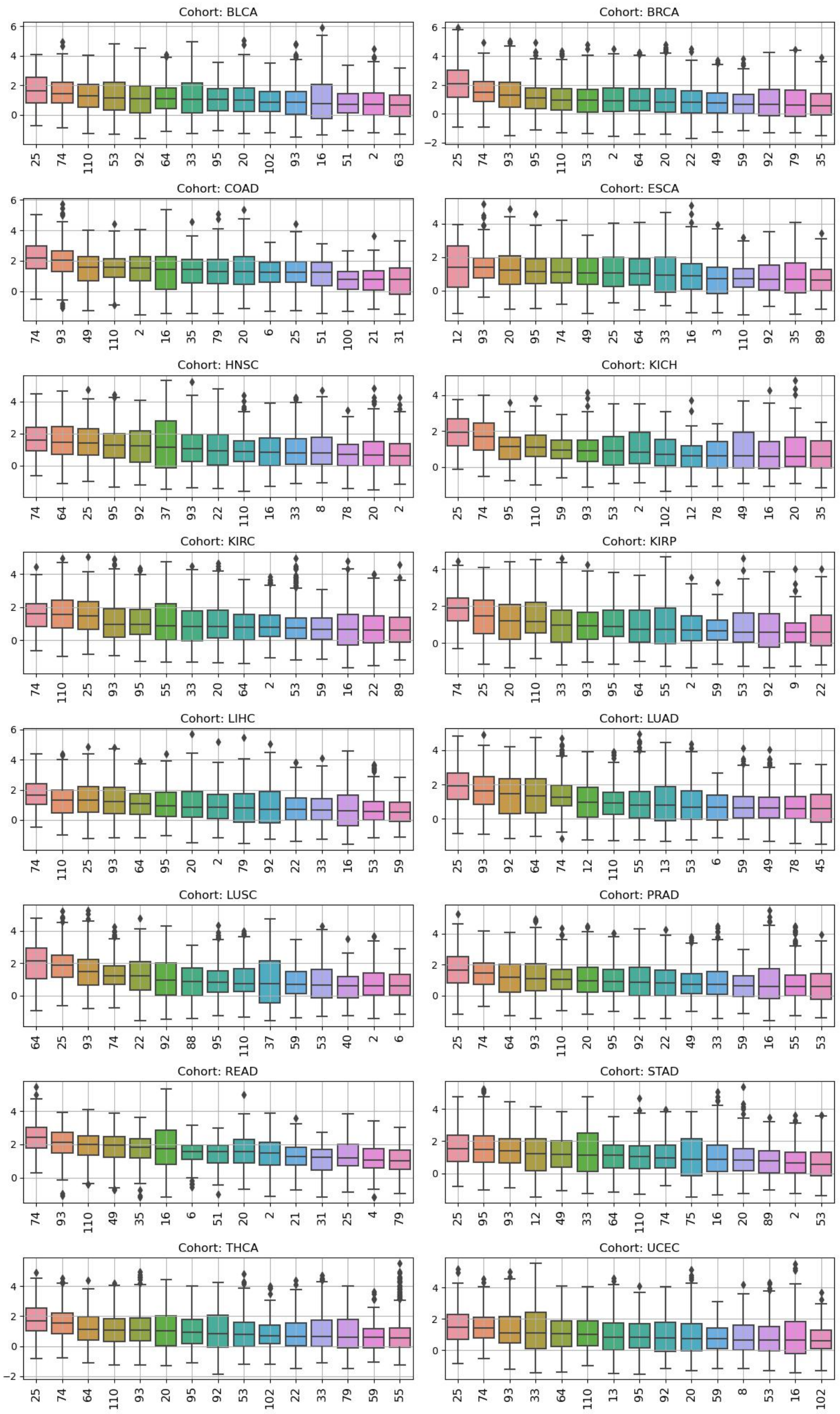
The same observations are true for the other cohorts as well. Figure 5.15 shows the top-15 supernodes found for other cohorts, where one can see how the various supernodes have similar saliencies, and none of them is significantly different than the others. Nevertheless, we were able to find some interesting facts by searching the literature for some of the over-represented gene sets associated with the top supernodes. As one example, supernode 25, for instance, was among the top ones for almost all of the cohorts. Besides being associated with the mitochondrial respiratory chain complex assembly, as described previously, and which is related to various cancer types, the supernode is also associated with the NADH dehydrogenase complex assembly. This biological process has been studied before as it relates with risk factors in breast cancer (CZARNECKA et al., 2010).

Another interesting pathway obtained by analyzing the embeddings is that of the Notch pathway. We have found it as an over-represented set of supernode 13, which was among the top-15 most important supernodes for the cohorts LUAD and UCEC. Interestingly, we found that the gradient of these cohort's outputs with respect to the supernode embeddings point in different directions, suggesting that the model is influenced differently by this supernode depending on the type of cancer (a discussion about the relation between the gradient and saliency is given in Appendix C). By searching the literature, we found that this pathway was previously associated with various types of cancer (ANUSEWICZ; ORZECHOWSKA; BEDNAREK, 2021). Its relation with cancer, however, is complicated by the fact that it can act both as a tumor suppressor and an oncogenic process, depending on the context (ANUSEWICZ; ORZECHOWSKA; BEDNAREK, 2021).

5.5 Discussion and Limitations

We conclude this chapter by discussing the results of the experiments described previously. In Section 5.5.1 we summarize and elaborate a few conclusions about the performance of the GNN models. Section 5.5.2 expands on the specific comparison between the multi-task and single-task models for the pan-cancer tasks. Finally, in Section 5.5.3 we discuss the results and limitations of the interpretability of the GNN models.

Figure 5.15 – Boxplots showing the saliencies of the top-15 supernodes found for each cohort. For each sample, the saliencies of the supernodes were standardized to have 0 mean and standard deviation of 1 across the supernodes, so that the samples are comparable with each other.



Source: The Author

5.5.1 Graph Coarsening Performance

In the previous sections, we studied the impact that coarsening the biological network have on the performance of various gene expression classification tasks. A few trends emerged from these results. First, it seems clear that increasing the number of coarsening levels does not improve the performance on the studied tasks. This is particularly true for the multi-category task of pan-cancer cohort classification. We note, however, that using a single coarsening level produced results that were as good as the neural network. In most works that reported improvements due to the introduction of graph convolutions, indeed, just a single convolution was used (see, for example, Yin et al. (2022), Ramirez et al. (2020), Ramirez et al. (2021)).

Among the coarsening models, the ones that followed a weighted-pooling approach were almost always better or as good as the best ones. One reason for that could be related to an increase in flexibility of the model to learn weights for each gene and supernodes individually, allowing it to learn to ignore irrelevant genes, increase the weights of more important ones, and to rescale gene expression and supernode values where it is appropriate. This is not possible when only sum-pooling is used, since the nodes are weighted equally, by definition. The benefit of using weighted pooling is most clear in the experiments where convolutions are not present in the first layers (right columns in the performance plots), where the sum-pooling methods often lead to poorer solutions. Additionally, the introduction of the graph convolutions retains performance better than the analogous coarsening models without convolutions. This suggests a positive aspect of the graph convolutions in this context, even though it was not possible to improve the results over the state-of-the-art.

Although we had hypothesized that coarsening the graph with the graph convolutions would provided further performance improvements, at least when only a few coarsening layers were used, it is nevertheless interesting how some models obtain a still high performance even though they greatly simplify the data. For example, the single-task model with only sum-pooling for cohort classification (the dashed red line in Figure 5.4) reaches a mean F1 score of 0.868. That is significantly below the performance of the best models, at around 0.937, but it is still noteworthy when we consider that this model is reducing the dimensionality of the data to only 128 features, and each feature is simply an unsupervised sum-based aggregated of nodes that belong to the same cluster.

In fact, when a simple sum operation is used for pooling, the model does not learn any aspect of the pooling operation, and is the same independent of the classification task. This implies that the expression data associated with the graph is simply aggregated together with other the expression of genes in the neighborhood. Hence, one can think of it as a simple method for network-aware dimensionality reduction. More elaborate approaches for reducing the dimensionality of the dataset constructed over a graph exist. One of such is that of projecting the dataset over the eigenfunctions of the graph, as in Rapaport et al. (2007). The simple sum pooling operation is also related to some feature selection methods. In particular, Moradi and Rostami (2015) represent a dataset using a graph and compute clusters in this graph. Then, node centrality measures are used together with the clusters in order to select feature sets with low redundancy. In their work, they construct a weighted graph based on the correlation between features, but one could perform similar steps over a pre-defined graph.

The tumor prediction tasks using cohort-specific data represent simpler problems (in the sense that they correspond to binary classification) but they also have fewer samples available. In general, the results obtained from these experiments support the evidence that the use of coarser models do not lead to significant improved performance. Here, however, it is harder to argue that they lead to a performance drop. Instead, it is more likely that the coarsening models retain performance, as the task is simpler. We have also observed that the cohort-specific tasks showed higher variance than that of the pan-cancer model, but this was independent of whether a neural network or a coarsening model was being considered. This is one of the reasons why in Chapter 6 we deepen our analysis on the differences between the pan-cancer and cohort-specific models, but focusing only on the NNs.

Although we were able to see some consistent trends as described above, it is important to consider some limitations of these work. First of all, even though we have tuned some hyperparameters of the model, it is possible that overall better results could be obtained with further tuning and exploration of other architectural details of the models – for example, the size and number of filters at each convolutional layer, the width and depth of the FC model, other weight initialization techniques and optimizers, etc. However, these variations would likely improve the results of all models explored, such that it is reasonable to expect that the general trends described would remain.

A more interesting aspect to consider is the use of other biological networks to serve as backbones. We have used the STRING because it was adopted in various works, as explored in Chapter 3, but there are reasons to suspect that improvements could be obtained by varying the backbone network. Yin et al. (2022), for example, when working with single-cell RNA-seq data, found that the best results were actually obtained when only the top-1% strongest edges were kept in the STRING network. When we considered that, we observed that such reduction in the network created a significant number of singleton nodes. In the context of a single pooling layer, this results in just a few actual convolutions being applied, since the singleton nodes do not have neighbors. Furthermore, applying hierarchical clustering algorithms over graphs with singleton nodes will in general lead to the existence of graph clusters containing only the singleton node, which would make it impossible to perform an over-representation analysis.

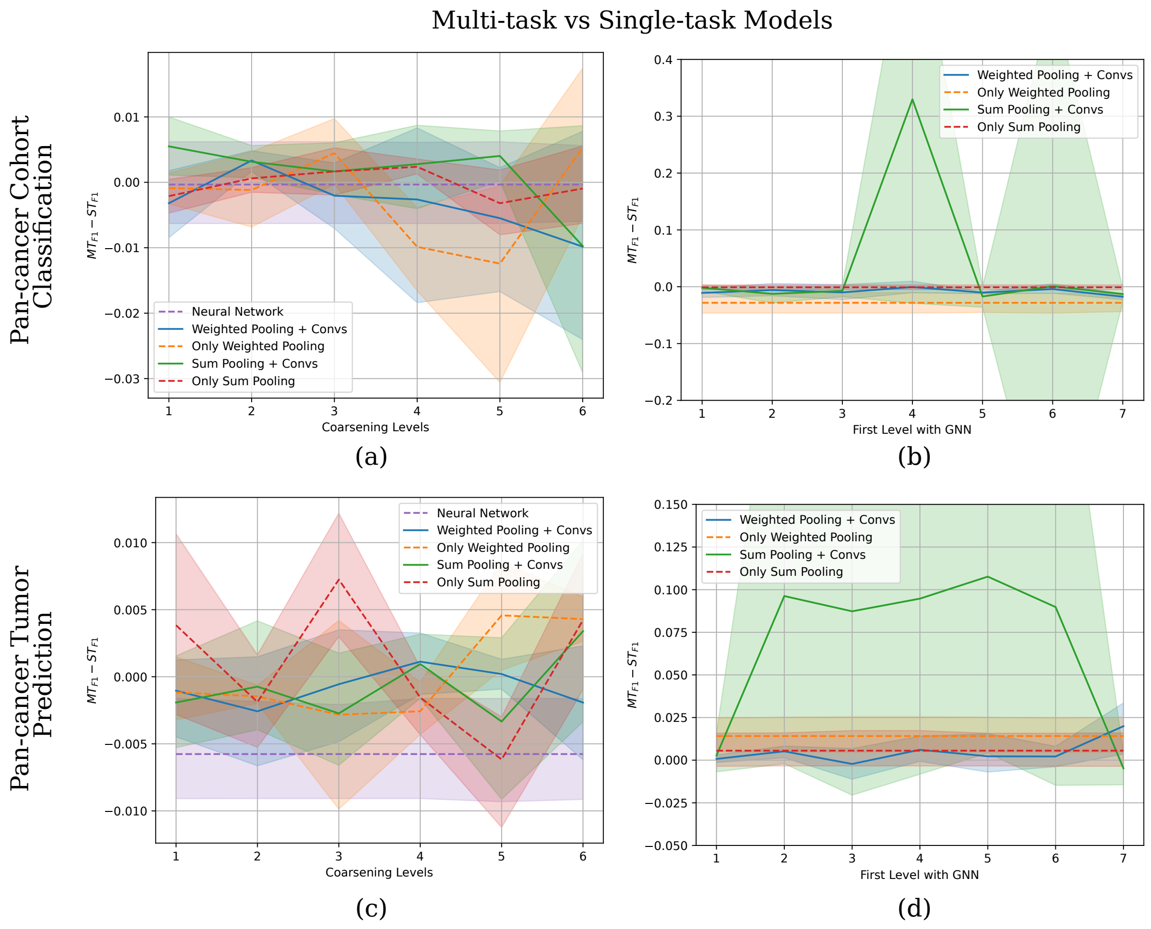
The pooling approach considered here is also quite restrictive, as we use a fixed hierarchical cluster structure computed prior to the model training. The benefit of this approach is that it is computationally tractable, which is important in our context, as we deal with networks that have tens of thousands of nodes. However, it is possible that trainable graph pooling approaches that allowed the pre-computed hierarchical structure to be adapted during training could lead to different and perhaps improved results. Finally, this work has focused on the ChebConv, as have some of the other works in the area. We chose this network because we had difficult in fitting other architectures such as the GCN (KIPF; WELLING, 2016) and the GraphSAGE (HAMILTON; YING; LESKOVEC, 2017), and the available Graph Attention Network (GAT) (VELIčKOVIć et al., 2018) implementations could not be easily integrated in our pipeline. Recently, also, transformer architectures (JADERBERG et al., 2015) have achieved achieved prominence and also explored in the context of graph ML, as in the Graph Transformer Network (YUN et al., 2019), but we did not consider them here. It is not unreasonable, therefore, to expect that different choices of architectures would influence the results.

5.5.2 Multitask vs Single Task Learning

In the previous sections, we have considered models that learn both the cancer prediction and cohort classification tasks simultaneously. This kind of model is interesting when developing cohort classification models, as it allows the model to behave differently for samples that belong to the same cohort, but where one is tumorous and the other is not. In Mostavi et al. (2020), for instance, they argue that ignoring the type of the sample (tumorous vs non-tumorous) is not ideal, particularly if one is trying to find cohort-specific gene biomarkers, as one would be unable to assert whether the biomarker is only a tissue biomarker or a biomarker of a tumor in that particular tissue.

However, the performance of the multitask models is not necessarily going to be the same as the performance of the individual single-task models, and it is not established whether the multi-tasking approach would lead to a performance improvement or reduction. We hypothesized that there could be performance gains in using a multi-task model, as the introduction of classification tasks could induce more constraints to the model, which is in general highly overparameterized. To evaluate that, we used our results from the single-task and multi-task pan-cancer classification models to calculate the difference in the F1 scores for the neural networks and the coarsened models as well. Figure 5.16 shows the difference between the multi-task and the single-task models for the pan-cancer classification tasks. The biggest differences occur in the sum pooling models

Figure 5.16 – Difference in the F1 score between multi-task and single-task models in the pan-cancer classification tasks.



Source: The Author

with convolutions in 5.16 - (b) and (d). These results, however, are mostly due to the high variability of these models, as discussed in the previous sections, and it is therefore hard to conclude anything from them. Besides that, it is in general not possible to infer any significant change between the single-task and multi-task models, indicating that there isn't any benefit nor disadvantage in using them. We can see this, for example, in the neural network models (dashed purple lines in Figures 5.16 - (a) and 5.16 - (b)). In (a), the neural network model achieves the same performance in both its multi-task and single-task forms, whereas in (b) the multi-task model has a mean F1 score -0.006 below that of the single-task model. Similarly, the coarsened models oscillate around 0, with no particular tendency. We note that the lack of performance improvement in this particular context should not be seen as a general statement on the lack of value of multi-tasking. In fact, multi-task was found to increase performance in gene expression tasks such as in Liao et al. (2019), it is however necessary to perform a deeper exploration of model architectures particularly aimed at taking advantage of the information that is introduced by the various tasks, which is not developed here.

5.5.3 Interpretation of the Coarsened Models

In Section 5.4, we have used concepts from saliency analysis to interpret a multi-task model for cohort classification and tumor prediction. The analysis results in various widely known cancer-related genes, and we were also able to find interesting biological processes that have been studied in the context of cancer and even considered for the development of treatments. Although this is encouraging, we need to consider some limitations that we hope to tackle in future works.

First of all, when we interpret the supernodes, we generally find more than one significantly over-represented gene sets. However, it is not clear which of them, or if all, is the most important, and it is also not clear whether it is just a smaller set of genes in the cluster that is relevant, instead of the biological process itself. One possible solution to build models that could overcome this is to introduce learnable clustering functions instead of the fixed, pre-computed, hierarchical clusters as we have done. In doing that, we would hope that the learned graph clusterings would include reduce to a single biological meaningful pathway for the task.

Also related to the interpretation of the supernodes when convolutions are present, we note that, even though the graph convolutions are designed as local operations, modeling only interactions between neighboring nodes, they lead to interactions between nodes that will not necessarily be pooled together in the same final supernode. Therefore, another interesting question that emerges is whether the attributed importance of a supernode is due to the genes belonging to the associated cluster itself, or if it is more related to how this cluster of genes is interacting with its neighbors. One way in we which we consider deepening our analysis in this sense is in introducing learnable edge weights, as was already done by Yin et al. (2022). In doing that, we hope that the edge weights associated with a supernode will diminish towards 0, if the supernode is relevant only through the genes it encompasses and the biological process it represents. Similarly, if it is the interactions with particular neighbors that are important, then the edge weights would be non-zero. This behaviour of the weights could be encouraged through regularization.

Finally, the full validity of the extracted interpretations need to be evaluated by specialized professionals. Although we found meaningful genes and biological processes, it is not clear how exactly they relate to each cancer itself, and understanding this will, in general, require specialized knowledged.

6 GENERALIZABILITY OF PAN-CANCER MODELS TO UNSEEN AND SPECIFIC COHORTS

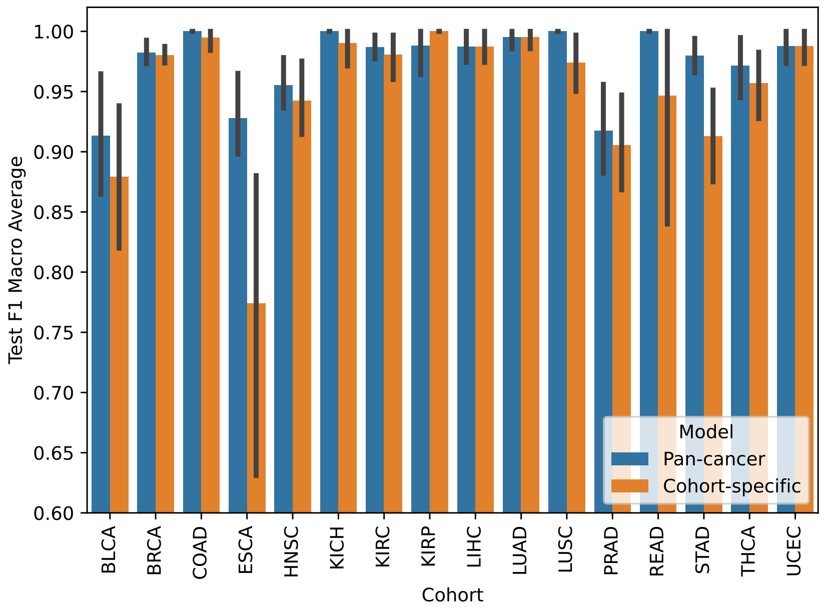
Previous work by Mostavi et al. (2021) evaluated how metric learning could be used for developing cohort-classification models that are able to differentiate samples from cohorts that were not seen during training. Similar ideas have been developed recently in the context of single-cell as well (MA et al., 2022; KOH; HOON, 2021). The reasoning behind these approaches is related to the fact that the dimensionality of gene expression data is in general much higher than are samples available for the specific cell or tissue types that one wishes to study. Additionally, if new cell or cancer types are discovered or described, it is unlikely that a great number of examples will be promptly available (MOSTAVI et al., 2021). Thus, in the same spirit of these works, we wondered whether cohort-specific classification tasks could be improved by introducing data from other sets of cohorts, and whether models built for classification using one set of cohorts would generalize to other, unseen cohorts. Moreover, we were encouraged by our results in Chapter 5, where the pan-cancer tumor prediction models showed good performance and the interpretation of the models gave us genes and biological processes often related to more than a single cancer type.

In Section 6.1, therefore, we compare the performance of the pan-cancer tumor prediction NNs on specific cohorts against NNs developed using only the cohort-specific data. The idea is to evaluate if samples from different tissues can improve performance on cohort-specific tasks. In particular, we were also interested in understanding whether a performance improvement would be more significant for smaller and imbalanced datasets. Furthermore, in Section 6.2, we evaluate whether the pan-cancer NNs developed for tumor prediction are able to distinguish tumorous from non-tumor samples coming from cohorts that it has not seen during training.

6.1 Comparison between Pan-cancer and Cohort-specific Classifiers

In this section, we compare the performance of the pan-cancer and cohort-specific NNs models on the tumor prediction task. Specifically, we grouped the predictions of the pan-cancer model on the test sets by cohort and computed F1 scores on the tumor prediction task given the cohort. To make a fair comparison, we have fitted and tested the cohort-specific models on the same samples of the cohort that were present in the pan-cancer models. That is, the pan-cancer and cohort-specific models are trained and tested on the same set of samples of the cohort of interest, but the pan-cancer is also fitted using data from other cohorts. Note that this implies that the samples used here for training and evaluating the cohort-specific classifiers are not in general the same as the ones used in Section 5.3.

Figure 6.1 – Comparison between the pan-cancer and cohort-specific NN models constructed for the tumor prediction task.

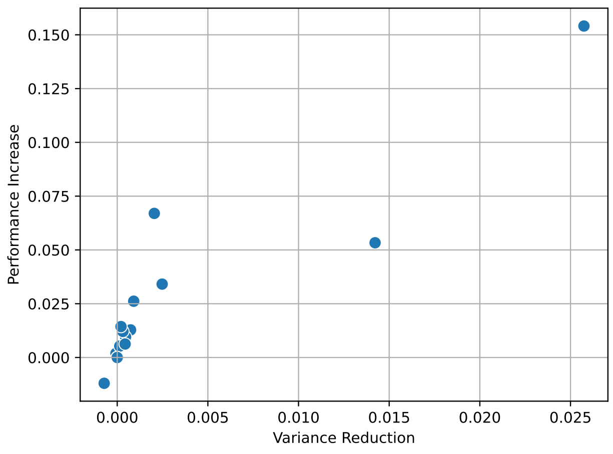


Source: The Author

The mean F1 scores obtained using 5-times holdout are shown in Figure 6.1. The pan-cancer model (in blue) performed better or equal to in all but one of the cohorts. The main difference was found for the ESCA dataset, where the mean F1 score obtained by the pan-cancer models was almost 20% higher. The difference was significantly impacted also by the higher variance presented by the cohort-specific model, shown by the black bar in the figure. We also observed this trend when evaluating the effect of coarsening in the ESCA dataset, in Figure 5.8. Next to ESCA, the STAD and READ cohorts were the cohorts that most improved with the pan-cancer model, with values 7.3% and 5.6% higher, respectively. In both cases, we see again this trend where the variance obtained for the cohort-specific models was higher. In three cases (LIHC, LUAD, and UCEC) the performance was exactly the same for the pan-cancer and cohort-specific models, and for KIRP the cohort-specific performance was actually 1.2% higher. Interestingly, we can observe that the variance of the cohort-specific KIRP model was smaller than its variance with the pan-cancer model.

The observations above suggested that the main cause of improvement related to the pan-cancer model is the reduction in variance of the trained models, instead of a reduction in bias. To evaluate that, we computed the correlations between the difference in mean and the difference in variance of the scores. Figure 6.2 shows the difference in the mean F1 scores plotted against the reduction in variance obtained using the pan-cancer model. The data showed a Pearson correlation coefficient of 0.91, indicating that the performance improvement obtained by the pan-cancer model is indeed strongly related to the reduction in variance.

Figure 6.2 – Increase in F1 scores versus the reduction in variance when using a pan-cancer neural network instead of a cohort-specific one in the Tumor Prediction task.

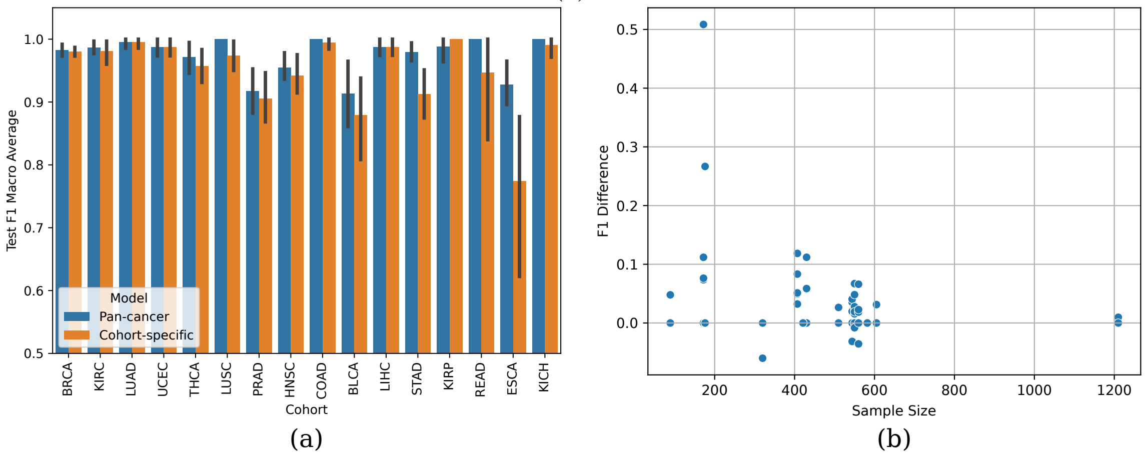


Source: The Author

We also hypothesized that the higher performance of the pan-cancer models could be more significant on the smaller datasets. Figure 6.3 - (a) shows the F1 scores of the pan-cancer and cohort-specific models ordered by decreasing sample size, and a scatter plot of the F1 difference versus the size of dataset is shown for each of the 5 holdouts in Figure 6.3 - (b). Visually, the variables seem to be correlated, although not quite strongly. We calculated the Pearson correlation coefficient between the performance improvement and the sample size and obtained a value of -0.243, with a p-value of 0.029 for the hypothesis that the correlation is nonzero. This indicates a small but statistically significant correlation between the sample sizes and the performance gain of using the pan-cancer model, where smaller datasets benefit more from the introduction of other cohorts.

It is often argued that one of the difficulties in dealing with gene expression data is its imbalance, which is exacerbated when few samples are available. Therefore, we have also evaluated whether the gain in performance of the pan-cancer model was higher

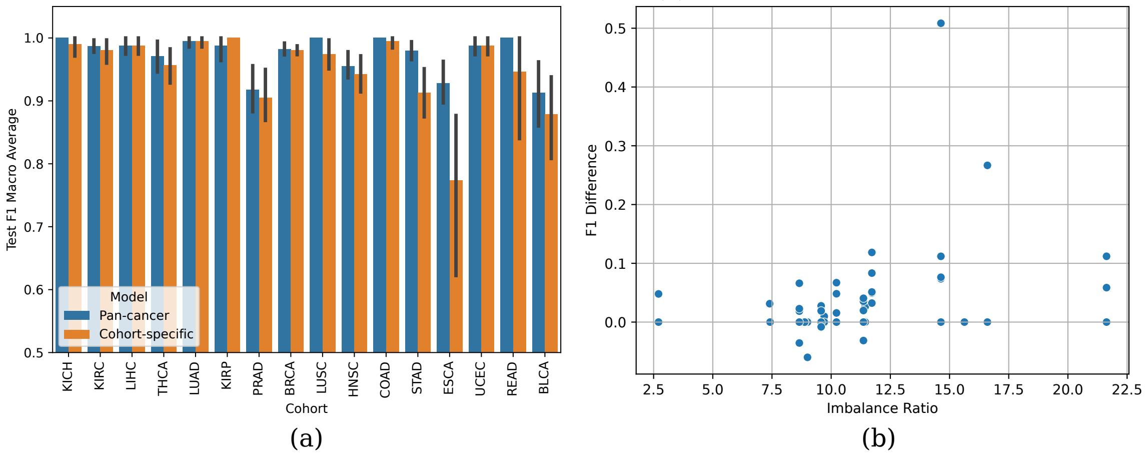
Figure 6.3 – F1 scores obtained on the pan-cancer and cohort specific models ordered by decreasing sample size (a) and a scatter plot of the F1 score improvement versus the sample size (b).



Source: The Author

for more imbalanced datasets. The justification here is that by introducing tumorous and normal samples from other cohorts, one would reduce the effect of the imbalance that is related to the sample size of the dataset. Figure 6.4 - (a) shows the performances of the pan-cancer and cohort-specific neural networks ordered by imbalance ratio, and a scatter plot of the performances in each run is shown versus the imbalance is shown in Figure 6.4 - (b). We have computed the Imbalance Ratio (IR) as the number of samples in the Primary Tumor class divided by the number of samples in the Normal Tissue class. Visually, the results seem to indicate a small tendency of increased benefit in using the

Figure 6.4 – F1 scores obtained on the pan-cancer and cohort specific models ordered by increasing Imbalance Ratio (a) and a scatter plot of the F1 score improvement versus the imbalance ration (b).



Source: The Author

pan-cancer neural network as the imbalance increases. The variables showed a Pearson correlation coefficient of 0.236, with a p-value of 0.035. This indicates that there is also a significant, but small, tendency that the imbalance of the datasets imply a increased benefit in using the pan-cancer model. We recall that during training we have used a weighted cross-entropy loss function in order to reduce the effects of the imbalance, so that the observed tendencies are despite that.

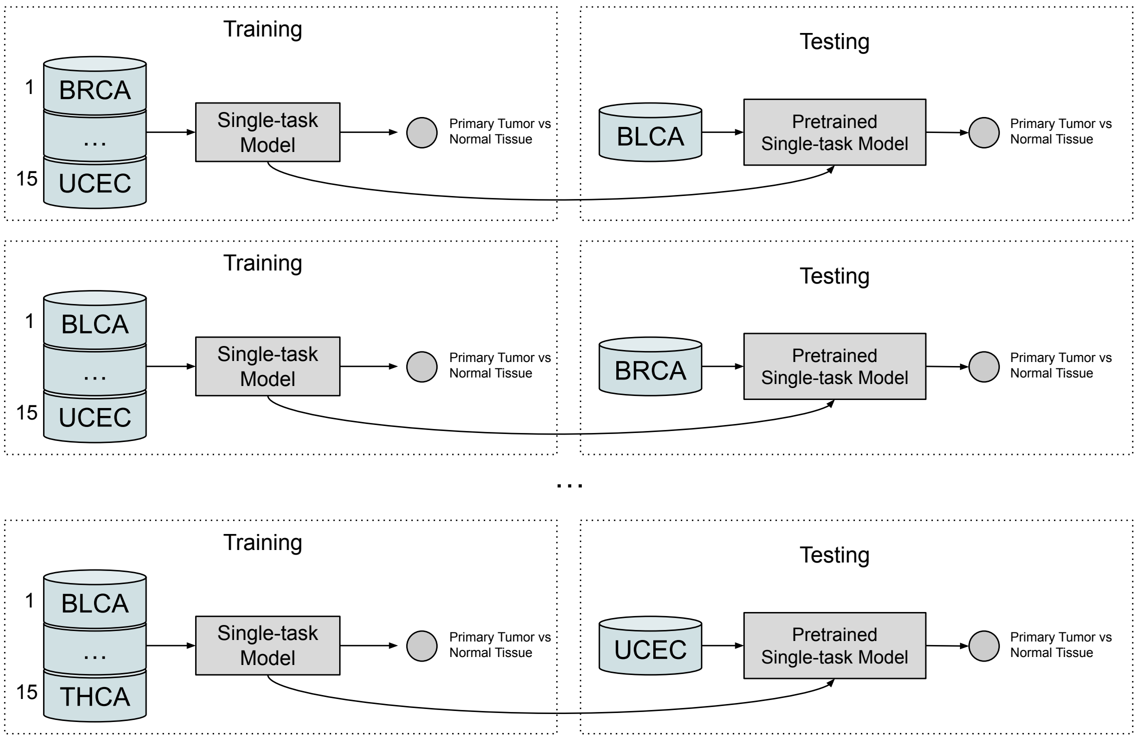
6.2 Pan-Cancer Tumor Prediction on Unseen Cohorts

This section is concerned with evaluating whether it is possible to generalize pan-cancer tumor prediction models to differentiate normal from tumorous samples of cohorts that were not used for training the pan-cancer model. To do that, we trained a different model for each cohort in our pan-cancer set. In each case, we trained the model using samples from all cohorts except those that corresponded to the specific cohort being evaluated. Then, we tested the resulting pan-cancer model on a set containing only samples from the separated cohort. Furthermore, we used the same cohort samples that were used for testing in the cohort-specific scenario described in Section 6.1, so that a fair comparison between them could be performed. In total, each model was fitted and tested 5 times in a repeated holdout fashion, as in the previous sections. These experiments are summarized in Figure 6.5.

The mean F1 scores for the pan-cancer and cohort-specific models for each cohort are presented in Figure 6.6. As in Section 6.1, we tested whether the variation in performance was correlated to sample size, imbalance or a change in variance. However, we found no statistically meaningful relation, with Pearson correlation p-values of 0.627, 0.068 and 0.285 for sample size, imbalance and change in variance, respectively.

A more interestingly hypothesis, however, seems to be that the ability of the pan-cancer model to generalize to an unseen cohort is primarily dependent of whether the unobserved cohort is anatomically close to a cohort that was observed during training. In Figure 6.6 we can seen this most clearly on the COAD and READ, and on the KICH, KIRC and KIRP datasets. COAD and READ are both types of colorectal cancer (CRC), and are known to share similar molecular mechanisms, besides their anatomical proximity (ZUO; DAI; REN, 2019). Indeed, the performance of both COAD and READ were visibly higher when using the pan-cancer model with unobserved samples than with the cohort-specific one. In these cases, the performance was very similar to the performance obtained with the pan-cancer model that included both READ and COAD samples. The same could be said for the KIRC, KICH and KIRP, which are all types of kidney cancer.

Figure 6.5 – Experiments performed to evaluate the generalizability of tumor-prediction pan-cancer models to samples from unobserved cohorts.


Training
1
BRCA
…
15
UCEC
Single-task Model
Primary Tumor vs Normal Tissue
Testing
BLCA
Pretrained Single-task Model
Primary Tumor vs Normal Tissue
Training
1
BLCA
…
15
UCEC
Single-task Model
Primary Tumor vs Normal Tissue
Testing
BRCA
Pretrained Single-task Model
Primary Tumor vs Normal Tissue
…
Training
1
BLCA
…
15
THCA
Single-task Model
Primary Tumor vs Normal Tissue
Testing
UCEC
Pretrained Single-task Model
Primary Tumor vs Normal Tissue

Source: The Author

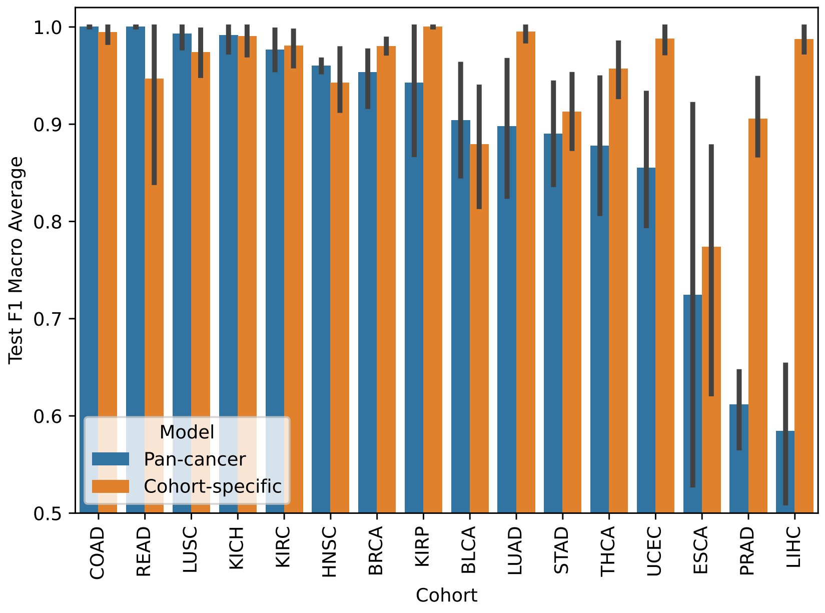
KICH and KIRC, in particular, perform as good as the cohort-specific models, whereas there is a more relevant drop in the case of KIRP. Nevertheless, the performance is still considerable.

In some cases, we can also see good performance and even improvement for runs where the training set does not have samples from any cohort that is clearly related to the unobserved cohort. For example, we can see that the models trained without the HNSC and BLCA cohorts performed better than their cohort-specific counterparts, even though there no anatomical proximity between them and other cohorts. On the other hand, we note that for some cohorts there is a significant drop in performance, such as for LIHC and PRAD.

6.3 Discussion and Limitations

Our results provide evidence that the use of samples from different tissues can improve performance in cohort-specific classification tasks, and even enable classification of new, unseen cohorts. As discussed earlier, these kind of approaches can help in building models of gene expression datasets that have few available samples. Nevertheless, our

Figure 6.6 – Mean F1 macro-average scores for the pan-cancer model tested on an unobserved cohort and the cohort-specific results. The cohorts were ordered by the pan-cancer score.



Source: The Author

work has some limitations that are important to mention.

We have restricted ourselves to the cohort-specific task of Tumor Predictions, that is, differentiating normal from tumorous samples. There are, however, other gene-expression based tasks that are equally important in a clinical setting and can also help in the identification of biomarkers. Examples of interesting cohort-specific problems that should be considered as well are: prognosis prediction for cancer (WONG; ROSTOMILY; WONG, 2019; RAMIREZ et al., 2021), where one attempts to predict survival outcomes, cancer recurrence (SHI; ZHANG, 2011), and metastasis identification (CHENG et al., 2023).

Another limitation of our results is that we have worked only with the TCGA RNA-seq data. However, it is important to consider whether the same effects would be obtained if we were to generalize to and from more heterogeneous sources. For example, would TCGA data help improve the results in general datasets collected from sources such as the GEO[1]? Also interesting would be to evaluate whether we could use TCGA RNA-seq data to improve methods developed for classifying scRNA-seq, specially because scRNA-seq is quite a recent development and it has not been so much explored as RNA-seq.

[1]https://www.ncbi.nlm.nih.gov/geo/

7 CONCLUSION

In this work, we have explored two problems in the context of gene expression models for cancer classification tasks. The majority of the work was dedicated to an investigation of pooling in GNNs applied to RNA-seq cancer datasets, but we have also devoted a set of experiments to understanding the generalization capabilities of pan-cancer models to cohort-specific tasks.

In our GNN experiments, we have considered the effects that multiple coarsening levels had on performance and whether different forms of applying pooling and the use of convolutions would alter the results. In general, we have observed that coarsening the input graph for more than one layer either reduced or just maintained the performance, in comparison to a NN. This downtrend, however, could be softened by the use of the ChebConvs and weighted pooling schemes. We have also explored the use of saliency analysis to interpret our studied GNNs. Besides using these methods to extract biomarkers at the gene level, as was already done in previous works using NNs (MOSTAVI et al., 2020), we have also considered the saliencies of the supernodes, generated through the pooling steps performed over a pre-computed hierarchical cluster structure. In both cases, we obtained interesting biomarkers and biological processes, many of which were previously studied in relation with cancer.

Nevertheless, as was discussed in Section 5.5, there are still quite a few issues that can be worthwhile to consider as future work regarding this first set of experiments. First, it would be interesting to look at a more diverse set of backbone networks. It is possible that other backbones could improve the results, and perhaps halt the observed downtrends with the coarsening levels. Furthermore, we plan to evaluate different forms of clustering the nodes in the graph for the pooling operations. In particular, we are working on the use of learnable pooling schemes, with the hope that these methods can lead to improved performance and more precise interpretations. With the same goal, it could be also beneficial to include learnable edge parameters in the model, leading to a better understanding about which aspects of the supernodes are actually relevant.

Our second section of experiments dealt with the generalization capability of pan-cancer models. In particular, we aimed at answering whether introducing samples from different cohorts improve cohort-specific performance, and if the pan-cancer models were able to generalize to cohorts that were not used in training. We obtained positive results in both cases. In particular, we observed that the benefits for cohort-specific tasks were more substantial when the original cohort contained fewer samples and was more imbalanced, even when correcting imbalance using weighted cross-entropy. The pan-cancer models were also able to perform classification on unseen cohorts, specially when related tissues were used to train the model. Despite these positive results, there are some important questions that should be explored in future work. Specifically, we hope to include a wider set of cohort-specific classification tasks besides that of tumor prediction, and we wish to also analyze if our results would also be true when trying to generalizing to and from more heterogeneous sources.

REFERENCES

ALBARADEI, S. et al. MetaCancer: A deep learning-based pan-cancer metastasis prediction model developed using multi-omics data. **Computational and Structural Biotechnology Journal**, v. 19, p. 4404–4411, 2021. ISSN 20010370. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S2001037021003378>.

ANUSEWICZ, D.; ORZECHOWSKA, M.; BEDNAREK, A. K. Notch signaling pathway in cancer—review with bioinformatic analysis. **Cancers**, MDPI, v. 13, n. 4, p. 768, 2021.

BARABáSI, A.-L.; GULBAHCE, N.; LOSCALZO, J. Network medicine: a network-based approach to human disease. **Nature Reviews Genetics**, v. 12, n. 1, p. 56–68, jan. 2011. ISSN 1471-0056, 1471-0064. Available from Internet: <http://www.nature.com/articles/nrg2918>.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: Springer, 2006.

BISHOP, H. B. C. M. **Deep Learning: Foundations and Concepts**. [S.l.]: Springer, 2024.

BORISOV, V.; HAUG, J.; KASNECI, G. Cancelout: A layer for feature selection in deep neural networks. In: **Artificial Neural Networks and Machine Learning–ICANN 2019: Deep Learning: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part II 28**. [S.l.]: Springer, 2019. p. 72–83.

BOURGEAIS, V.; ZEHRAOUI, F.; HANCZAR, B. GraphGONet: a self-explaining neural network encapsulating the Gene Ontology graph for phenotype prediction on gene expression. **Bioinformatics**, v. 38, n. 9, p. 2504–2511, abr. 2022. ISSN 1367-4803, 1460-2059. Available from Internet: <https://academic.oup.com/bioinformatics/article/38/9/2504/6546279>.

BRUNA, J. et al. **Spectral Networks and Locally Connected Networks on Graphs**. arXiv, 2014. ArXiv:1312.6203 [cs]. Available from Internet: <http://arxiv.org/abs/1312.6203>.

BULLARD, J. H. et al. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. **BMC Bioinformatics**, v. 11, n. 1, p. 94, dec. 2010. ISSN 1471-2105. Available from Internet: <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-11-94>.

CHAUDHARY, K. et al. Deep learning–based multi-omics integration robustly predicts survival in liver cancer. v. 24, n. 6, p. 1248–1259, 2018. ISSN 1078-0432, 1557-3265. Available from Internet: <https://aacrjournals.org/clincancerres/article/24/6/1248/475/Deep-Learning-Based-Multi-Omics-Integration>.

CHENG, N. et al. Prediction of lung cancer metastasis by gene expression. **Computers in Biology and Medicine**, v. 153, p. 106490, feb. 2023. ISSN 00104825. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0010482522011982>.

CHEREDA, H. et al. Explaining decisions of graph convolutional neural networks: patient-specific molecular subnetworks responsible for metastasis prediction in breast cancer. **Genome Medicine**, v. 13, n. 1, p. 42, dec. 2021. ISSN 1756-994X. Available from Internet: <https://genomemedicine.biomedcentral.com/articles/10.1186/s13073-021-00845-7>.

CHIU, Y.-C. et al. Deep learning of pharmacogenomics resources: moving towards precision oncology. **Briefings in Bioinformatics**, v. 21, n. 6, p. 2066–2083, dec. 2020. ISSN 1467-5463, 1477-4054. Available from Internet: <https://academic.oup.com/bib/article/21/6/2066/5669856>.

CHUANG, Y.-H. et al. Convolutional neural network for human cancer types prediction by integrating protein interaction networks and omics data. **Scientific Reports**, v. 11, n. 1, p. 20691, dec. 2021. ISSN 2045-2322. Available from Internet: <https://www.nature.com/articles/s41598-021-98814-y>.

COCK, P. J. A. et al. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. **Nucleic Acids Research**, v. 38, n. 6, p. 1767–1771, abr. 2010. ISSN 0305-1048, 1362-4962. Available from Internet: <https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkp1137>.

CONSORTIUM, G. O. The gene ontology (GO) database and informatics resource. **Nucleic Acids Research**, Oxford University Press, v. 32, n. suppl_1, p. D258–D261, 2004.

CZARNECKA, A. M. et al. Mitochondrial nadh-dehydrogenase polymorphisms as sporadic breast cancer risk factor. **Oncology Reports**, Spandidos Publications, v. 23, n. 2, p. 531–535, 2010.

DEFFERRARD, M.; BRESSON, X.; VANDERGHEYNST, P. Convolutional neural networks on graphs with fast localized spectral filtering. In: LEE, D. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2016. v. 29. Available from Internet: <https://proceedings.neurips.cc/paper/2016/file/04df4d434d481c5bb723be1b6df1ee65-Paper.pdf>.

DHILLON, I. S.; GUAN, Y.; KULIS, B. Weighted graph cuts without eigenvectors a multilevel approach. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 29, n. 11, p. 1944–1957, 2007.

DIVATE, M. et al. Deep learning-based pan-cancer classification model reveals tissue-of-origin specific gene expression signatures. **Cancers**, MDPI, v. 14, n. 5, p. 1185, 2022.

DOBIN, A. et al. STAR: ultrafast universal RNA-seq aligner. **Bioinformatics**, v. 29, n. 1, p. 15–21, jan. 2013. ISSN 1367-4811, 1367-4803. Available from Internet: <https://academic.oup.com/bioinformatics/article/29/1/15/272537>.

DRáBER, P.; DRáBEROVá, E. Dysregulation of Microtubule Nucleating Proteins in Cancer Cells. **Cancers**, v. 13, n. 22, p. 5638, nov. 2021. ISSN 2072-6694. Available from Internet: <https://www.mdpi.com/2072-6694/13/22/5638>.

DUAN, R. et al. Evaluation and comparison of multi-omics data integration methods for cancer subtyping. **PLoS Computational Biology**, v. 17, n. 8, p. 1–33, 2021. ISSN 15537358. Available from Internet: <http://dx.doi.org/10.1371/journal.pcbi.1009224>.

DUMONTET, C.; JORDAN, M. A. Microtubule-binding agents: a dynamic field of cancer therapeutics. **Nature Reviews Drug Discovery**, v. 9, n. 10, p. 790–803, oct. 2010. ISSN 1474-1776, 1474-1784. Available from Internet: <https://www.nature.com/articles/nrd3253>.

FOWLER, S.; ROUSH, R.; WISE, J. **Concepts of Biology**. [S.l.]: OpenStax, 2013.

GAO, C. et al. Exon 3 mutations of *CTNNB1* drive tumorigenesis: a review. **Oncotarget**, v. 9, n. 4, p. 5492–5508, jan. 2018. ISSN 1949-2553. Available from Internet: <https://www.oncotarget.com/lookup/doi/10.18632/oncotarget.23695>.

GILMER, J. et al. Neural message passing for quantum chemistry. In: **Proceedings of the 34th International Conference on Machine Learning - Volume 70**. [S.l.]: JMLR.org, 2017. (ICML'17), p. 1263–1272. Event-place: Sydney, NSW, Australia.

GOLDMAN, M. J. et al. Visualizing and interpreting cancer genomics data via the Xena platform. **Nature Biotechnology**, v. 38, n. 6, p. 675–678, jun. 2020. ISSN 1546-1696. Available from Internet: <https://doi.org/10.1038/s41587-020-0546-8>.

GRATTAROLA, D. et al. Understanding Pooling in Graph Neural Networks. **IEEE Transactions on Neural Networks and Learning Systems**, p. 1–11, 2022. ISSN 2162-237X, 2162-2388. Available from Internet: <https://ieeexplore.ieee.org/document/9836996/>.

GROVER, A.; LESKOVEC, J. node2vec: Scalable Feature Learning for Networks. In: **Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. San Francisco California USA: ACM, 2016. p. 855–864. ISBN 978-1-4503-4232-2. Available from Internet: <https://dl.acm.org/doi/10.1145/2939672.2939754>.

GUIA, J. M. de; DEVARAJ, M.; LEUNG, C. K. DeepGx: deep learning using gene expression for cancer classification. In: **Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining**. [S.l.: s.n.], 2019. p. 913–920.

HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive Representation Learning on Large Graphs. In: GUYON, I. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2017. v. 30. Available from Internet: <https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf>.

HAMILTON, W. L. Graph representation learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan and Claypool, v. 14, n. 3, p. 1–159, 2020.

HAMMOND, D. K.; VANDERGHEYNST, P.; GRIBONVAL, R. Wavelets on graphs via spectral graph theory. **Applied and Computational Harmonic Analysis**, v. 30, n. 2, p. 129–150, 2011. ISSN 1063-5203. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S1063520310000552>.

HANCZAR, B.; BOURGEAIS, V.; ZEHRAOUI, F. Assessment of deep learning and transfer learning for cancer prediction based on gene expression data. **BMC bioinformatics**, Springer, v. 23, n. 1, p. 262, 2022.

HASTIE, T.; TIBSHIRANI, R.; WAINWRIGHT, M. **Statistical learning with sparsity: the Lasso and generalizations**. [S.l.]: CRC press, 2015.

HAYAKAWA, J. et al. Pathway importance by graph convolutional network and Shapley additive explanations in gene expression phenotype of diffuse large B-cell lymphoma. **PLOS ONE**, v. 17, n. 6, p. e0269570, jun. 2022. ISSN 1932-6203. Available from Internet: <https://dx.plos.org/10.1371/journal.pone.0269570>.

JADERBERG, M. et al. Spatial transformer networks. In: CORTES, C. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2015. v. 28. Available from Internet: <https://proceedings.neurips.cc/paper_files/paper/2015/file/33ceb07bf4eeb3da587e268d663aba1a-Paper.pdf>.

JEFFERY, J. et al. Beyond cytokinesis: the emerging roles of cep55 in tumorigenesis. **Oncogene**, Nature Publishing Group, v. 35, n. 6, p. 683–690, 2016.

JOSHI, P.; DHAR, R. Epicc: A bayesian neural network model with uncertainty correction for a more accurate classification of cancer. **Scientific Reports**, Nature Publishing Group UK London, v. 12, n. 1, p. 14628, 2022.

KAILEMIA, M. J.; PARK, D.; LEBRILLA, C. B. Glycans and glycoproteins as specific biomarkers for cancer. **Analytical and Bioanalytical Chemistry**, v. 409, n. 2, p. 395–410, jan. 2017. ISSN 1618-2642, 1618-2650. Available from Internet: <http://link.springer.com/10.1007/s00216-016-9880-6>.

KARYPIS, G.; KUMAR, V. A fast and high quality multilevel scheme for partitioning irregular graphs. **SIAM Journal on Scientific Computing**, SIAM, v. 20, n. 1, p. 359–392, 1998.

KASEM, K. et al. The roles of jk-1 (fam134b) expressions in colorectal cancer. **Experimental cell research**, Elsevier, v. 326, n. 1, p. 166–173, 2014.

KHORSHED, T.; MOUSTAFA, M. N.; RAFEA, A. Deep Learning for Multi-Tissue Cancer Classification of Gene Expressions (GeneXNet). **IEEE Access**, v. 8, p. 90615–90629, 2020. ISSN 2169-3536. Available from Internet: <https://ieeexplore.ieee.org/document/9087866/>.

KING, G.; ZENG, L. Logistic regression in rare events data. **Political Analysis**, Cambridge University Press, v. 9, n. 2, p. 137–163, 2001.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. arXiv, 2017. ArXiv:1412.6980 [cs]. Available from Internet: <http://arxiv.org/abs/1412.6980>.

KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. In: **J. International Conference on Learning Representations (ICLR 2017)**. [S.l.: s.n.], 2016.

KOH, W.; HOON, S. MapCell: Learning a Comparative Cell Type Distance Metric With Siamese Neural Nets With Applications Toward Cell-Type Identification Across Experimental Datasets. **Frontiers in Cell and Developmental Biology**, v. 9, p. 767897, nov. 2021. ISSN 2296-634X. Available from Internet: <https://www.frontiersin.org/articles/10.3389/fcell.2021.767897/full>.

KOUROU, K. et al. Machine learning applications in cancer prognosis and prediction. **Computational and Structural Biotechnology Journal**, v. 13, p. 8–17, 2015. ISSN 20010370. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S2001037014000464>.

LEE, S. et al. Cancer subtype classification and modeling by pathway attention and propagation. **Bioinformatics**, v. 36, n. 12, p. 3818–3824, jun. 2020. ISSN 1367-4803, 1460-2059. Available from Internet: <https://academic.oup.com/bioinformatics/article/36/12/3818/5811233>.

LEMARIE, A.; GRIMM, S. Mitochondrial respiratory chain complexes: apoptosis sensors mutated in cancer? **Oncogene**, Nature Publishing Group, v. 30, n. 38, p. 3985–4003, 2011.

LESK, A. M. **Introduction to genomics**. 2nd ed. ed. Oxford ; New York: Oxford University Press, 2012. OCLC: ocn755071575. ISBN 978-0-19-956435-4.

LI, B.; WANG, T.; NABAVI, S. Cancer molecular subtype classification by graph convolutional networks on multi-omics data. In: **Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics**. Gainesville Florida: ACM, 2021. p. 1–9. ISBN 978-1-4503-8450-6. Available from Internet: <https://dl.acm.org/doi/10.1145/3459930.3469542>.

LIAO, Q. et al. Multi-task deep convolutional neural network for cancer diagnosis. **Neurocomputing**, v. 348, p. 66–73, jul. 2019. ISSN 09252312. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0925231218312827>.

LIAW, R. et al. Tune: A research platform for distributed model selection and training. **arXiv preprint arXiv:1807.05118**, 2018.

LIU, Z. et al. Research on fine-tuning CNN for cancer diagnosis with gene expression data. In: **2022 14th International Conference on Machine Learning and Computing (ICMLC)**. Guangzhou China: ACM, 2022. p. 140–145. ISBN 978-1-4503-9570-0. Available from Internet: <https://dl.acm.org/doi/10.1145/3529836.3529844>.

LOSHCHILOV, I.; HUTTER, F. Sgdr: Stochastic gradient descent with warm restarts. **arXiv preprint arXiv:1608.03983**, 2016.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. **arXiv preprint arXiv:1711.05101**, 2017.

LOWE, R. et al. Transcriptomics technologies. **PLOS Computational Biology**, v. 13, n. 5, p. e1005457, may 2017. ISSN 1553-7358. Available from Internet: <https://dx.plos.org/10.1371/journal.pcbi.1005457>.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. **Advances in neural information processing systems**, v. 30, 2017.

LYU, B.; HAQUE, A. Deep learning based tumor type classification using gene expression data. In: **Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics**. [S.l.: s.n.], 2018. p. 89–96.

LóPEZ-GARCíA, G. et al. Transfer learning with convolutional neural networks for cancer survival prediction using gene-expression data. **PLOS ONE**, p. 24, 2020.

MA, Z. et al. Metric learning for comparing genomic data with triplet network. **Briefings in Bioinformatics**, v. 23, n. 5, p. bbac345, sep. 2022. ISSN 1467-5463, 1477-4054. Available from Internet: <https://academic.oup.com/bib/article/doi/10.1093/bib/bbac345/6679451>.

MABONGA, L.; KAPPO, A. P. The oncogenic potential of small nuclear ribonucleoprotein polypeptide g: a comprehensive and perspective view. **American Journal of Translational Research**, e-Century Publishing Corporation, v. 11, n. 11, p. 6702, 2019.

MADAN, M. et al. Atp13a3 and caveolin-1 as potential biomarkers for difluoromethylornithine-based therapies in pancreatic cancers. **American Journal of Cancer Research**, e-Century Publishing Corporation, v. 6, n. 6, p. 1231, 2016.

MESQUITA, D.; SOUZA, A. H.; KASKI, S. **Rethinking pooling in graph neural networks**. arXiv, 2020. ArXiv:2010.11418 [cs]. Available from Internet: <http://arxiv.org/abs/2010.11418>.

MOHAMMED, M. et al. A stacking ensemble deep learning approach to cancer type classification based on TCGA data. **Scientific Reports**, v. 11, n. 1, p. 1–22, 2021. ISSN 20452322. Publisher: Nature Publishing Group UK ISBN: 0123456789. Available from Internet: <https://doi.org/10.1038/s41598-021-95128-x>.

MORADI, P.; ROSTAMI, M. A graph theoretic approach for unsupervised feature selection. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 44, p. 33–45, 2015.

MOSTAVI, M. et al. CancerSiamese: one-shot learning for predicting primary and metastatic tumor types unseen during model training. **BMC Bioinformatics**, v. 22, n. 1, p. 244, dec. 2021. ISSN 1471-2105. Available from Internet: <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-021-04157-w>.

MOSTAVI, M. et al. Convolutional neural network models for cancer type prediction based on gene expression. **BMC Medical Genomics**, Springer, v. 13, p. 1–13, 2020.

ORTEGA, A. et al. Graph Signal Processing: Overview, Challenges, and Applications. **Proceedings of the IEEE**, v. 106, n. 5, p. 808–828, may 2018. ISSN 0018-9219, 1558-2256. Available from Internet: <https://ieeexplore.ieee.org/document/8347162/>.

PAN, W.-W. et al. Cell type-dependent function of lats1/2 in cancer cell growth. **Oncogene**, Nature Publishing Group UK London, v. 38, n. 14, p. 2595–2610, 2019.

PARK, Y.; HAUSCHILD, A.-C.; HEIDER, D. Transfer learning compensates limited data, batch effects and technological heterogeneity in single-cell sequencing. **NAR Genomics and Bioinformatics**, v. 3, n. 4, p. lqab104, oct. 2021. ISSN 2631-9268. Available from Internet: <https://academic.oup.com/nargab/article/doi/10.1093/nargab/lqab104/6426025>.

RAMIREZ, R. et al. Classification of cancer types using graph convolutional neural networks. **Frontiers in Physics**, v. 8, p. 203, 2020. Publisher: Frontiers Media SA.

RAMIREZ, R. et al. Prediction and interpretation of cancer survival using graph convolution neural networks. **Methods**, v. 192, p. 120–130, aug. 2021. ISSN 10462023. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S1046202321000165>.

RAPAPORT, F. et al. Classification of microarray data using gene networks. **BMC Bioinformatics**, v. 8, n. 1, p. 35, dec. 2007. ISSN 1471-2105. Available from Internet: <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-8-35>.

REGGIO, A. et al. Role of FAM134 paralogues in endoplasmic reticulum remodeling, ER-phagy, and collagen quality control. **EMBO reports**, v. 22, n. 9, p. e52289, 2021.

RHEE, S.; SEO, S.; KIM, S. Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification. **arXiv preprint arXiv:1711.05859**, 2017.

SANTORO, A. et al. A simple neural network module for relational reasoning. **Advances in Neural Information Processing Systems**, v. 30, 2017.

SEKHAR, V.; ANDL, T.; PHANSTIEL, O. ATP13A3 facilitates polyamine transport in human pancreatic cancer cells. **Scientific Reports**, v. 12, n. 1, p. 4045, mar. 2022. ISSN 2045-2322. Available from Internet: <https://www.nature.com/articles/s41598-022-07712-4>.

SELVARAJU, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. p. 618–626, 2017.

SHAH, S. H. et al. Optimized gene selection and classification of cancer from microarray gene expression data using deep learning. **Neural Computing and Applications**, oct. 2020. ISSN 0941-0643, 1433-3058. Available from Internet: <https://link.springer.com/10.1007/s00521-020-05367-8>.

SHI, M.; ZHANG, B. Semi-supervised learning improves gene expression-based prediction of cancer recurrence. **Bioinformatics**, v. 27, n. 21, p. 3017–3023, nov. 2011. ISSN 1367-4811, 1367-4803. Available from Internet: <https://academic.oup.com/bioinformatics/article/27/21/3017/216999>.

SIMONNET, H. et al. Low mitochondrial respiratory chain content correlates with tumor aggressiveness in renal cell carcinoma. **Carcinogenesis**, Oxford University Press, v. 23, n. 5, p. 759–768, 2002.

SIMONYAN, K.; VEDALDI, A.; ZISSERMAN, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. 2013.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. **The Journal of Machine Learning Research**, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014.

SZKLARCZYK, D. et al. String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. **Nucleic acids research**, Oxford University Press, v. 47, n. D1, p. D607–D613, 2019.

SáNCHEZ, J. S.; GARCíA, V. Addressing the links between dimensionality and data characteristics in gene-expression microarrays. **ACM International Conference Proceeding Series**, 2018. ISBN: 9781450353045.

VELIčKOVIć, P. et al. **Graph Attention Networks**. arXiv, 2018. ArXiv:1710.10903 [cs, stat]. Available from Internet: <http://arxiv.org/abs/1710.10903>.

WANG, B. et al. Similarity network fusion for aggregating data types on a genomic scale. v. 11, n. 3, p. 333–337, 2014. ISSN 1548-7091, 1548-7105. Available from Internet: <http://www.nature.com/articles/nmeth.2810>.

WANG, J. et al. Data denoising with transfer learning in single-cell transcriptomics. **Nature Methods**, v. 16, n. 9, p. 875–878, sep. 2019. ISSN 1548-7091, 1548-7105. Available from Internet: <http://www.nature.com/articles/s41592-019-0537-1>.

WANG, T.; BAI, J.; NABAVI, S. Single-cell classification using graph convolutional networks. **BMC Bioinformatics**, BioMed Central, v. 22, n. 1, p. 1–23, 2021.

WANG, T.; BAI, J.; NABAVI, S. Single-cell classification using graph convolutional networks. **BMC Bioinformatics**, v. 22, n. 1, p. 364, dec. 2021. ISSN 1471-2105. Available from Internet: <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-021-04278-2>.

WANG, Y. et al. Generalizing from a Few Examples: A Survey on Few-Shot Learning. **ACM Computing Surveys**, v. 53, n. 3, jun. 2020. ISSN 0360-0300. Place: New York, NY, USA Publisher: Association for Computing Machinery. Available from Internet: <https://doi.org/10.1145/3386252>.

WEYDEN, L. van der et al. Crispr activation screen in mice identifies novel membrane proteins enhancing pulmonary metastatic colonisation. **Communications Biology**, Nature Publishing Group UK London, v. 4, n. 1, p. 395, 2021.

WIEMER, A. J.; HOHL, R. J.; WIEMER, D. F. The Intermediate Enzymes of Isoprenoid Metabolism as Anticancer Targets. **Anti-Cancer Agents in Medicinal Chemistry**, v. 9, n. 5, p. 526–542, jun. 2009. ISSN 18715206. Available from Internet: <https://www.eurekaselect.com/69326/article>.

WINTER, G. E. et al. The solute carrier slc35f2 enables ym155-mediated DNA damage toxicity. **Nature Chemical Biology**, Nature Publishing Group US New York, v. 10, n. 9, p. 768–773, 2014.

WONG, K. K.; ROSTOMILY, R.; WONG, S. T. C. Prognostic Gene Discovery in Glioblastoma Patients using Deep Learning. **Cancers**, v. 11, n. 1, p. 53, jan. 2019. ISSN 2072-6694. Available from Internet: <https://www.mdpi.com/2072-6694/11/1/53>.

YIN, Q. et al. scGraph: a graph neural network-based approach to automatically identify cell types. **Bioinformatics**, v. 38, n. 11, p. 2996–3003, abr. 2022. ISSN 1367-4803. _eprint: https://academic.oup.com/bioinformatics/article-pdf/38/11/2996/43857941/btac199.pdf. Available from Internet: <https://doi.org/10.1093/bioinformatics/btac199>.

YING, R. et al. **Hierarchical Graph Representation Learning with Differentiable Pooling**. arXiv, 2019. ArXiv:1806.08804 [cs, stat]. Available from Internet: <http://arxiv.org/abs/1806.08804>.

YU, H. et al. Architectures and accuracy of artificial neural network for disease classification from omics data. **BMC Genomics**, v. 20, n. 1, dec. 2019. ISSN 1471-2164. Available from Internet: <https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-019-5546-z>.

YUN, S. et al. Graph transformer networks. Curran Associates, Inc., v. 32, 2019. Available from Internet: <https://proceedings.neurips.cc/paper_files/paper/2019/file/9d63484abb477c97640154d40595a3bb-Paper.pdf>.

ZHANG, T.-H. et al. Transformer for Gene Expression Modeling (T-GEM): An Interpretable Deep Learning Model for Gene Expression-Based Phenotype Predictions. **Cancers**, v. 14, n. 19, p. 4763, sep. 2022. ISSN 2072-6694. Available from Internet: <https://www.mdpi.com/2072-6694/14/19/4763>.

ZHU, Y.-M. et al. C14orf159 suppresses gastric cancer cells invasion and proliferation by inactivating ERK signaling. **Cancer Management and Research**, Volume 11, p. 1717–1723, feb. 2019. ISSN 1179-1322.

ZUO, S.; DAI, G.; REN, X. Identification of a 6-gene signature predicting prognosis for colorectal cancer. **Cancer Cell International**, BioMed Central, v. 19, n. 1, p. 1–15, 2019.

APPENDIX A — RESUMO EXPANDIDO

Novas tecnologias de sequenciamento levaram à geração massiva de dados de expressão gênica, possibilitando a análise e modelagem dos aspectos genômicos de doenças críticas, como o câncer. Nesse contexto, modelos de aprendizado de máquina (AM) são de fundamental importância, pois podem auxiliar médicos em ambientes clínicos e também na identificação de marcadores biológicos que podem levar à descoberta de novas terapias. No entanto, a alta dimensionalidade e não-linearidade desses dados, aliada à baixa disponibilidade de exemplos, especialmente para tipos mais raros de cânceres, dificulta a sua análise. Esses fatores levaram a propostas de vários modelos de AM que poderiam trabalhar com dados de genômicos de câncer. Dentre esses, diferentes arquiteturas de redes neurais foram desenvolvidas, e alguns trabalhos recentes propuseram o uso de redes neurais de grafo (GNN) para incorporar redes biológicas prévias aos modelos.

De forma geral, no entanto, esses trabalhos não exploraram de maneira mais aprofundada a etapa de *pooling*, fundamental na classificação no nível do grafo quando são usadas as GNNs. Assim, uma parte importante dessa dissertação é dedicada a analisar como o *pooling*, baseado no agrupamento hierárquico dos nodos da rede biológica de entrada, impacta no desempenho das GNNs nas tarefas de classificação com dados genômicos de câncer. Especificamente, esse trabalho estuda como mais níveis de *pooling* (e a consequente perda de resolução do grafo) afeta a performance dos modelos. Além disso, realizamos a interpretação de um dos modelos gerados, usando métodos baseados em gradiente (*saliency maps*), para averiguar se genes já conhecidos estavam sendo usados na decisão do modelo, e se os supernodos (nodos das versões agrupadas do grafo de entrada, que estão associados a conjuntos de nodos do grafo inicial) de maior saliência se relacionavam com processos biológicos relevantes. Fora o efeito do número de níveis, consideramos também o caso em que, nos níveis iniciais, o grafo de entrada é reduzido sem o uso de convoluções, a fim de analisar o efeito das convoluções em comparação com a aplicação exclusiva do *pooling*, e tendo em vista também que as convoluções são bastante custosas computacionalmente quando o número de nodos é grande. Consideramos também duas formas distintas de *pooling*, uma forma ponderada por pesos aprendidos durante o treino, e outra forma correspondente apenas à soma dos valores dos nodos agrupados.

Em nossos experimentos, usamos como rede biológica de entrada a STRING e computamos um agrupamento hierárquivo sobre ela usando o método de *heavy-edge* *matching*, ambos usados em outros trabalhos na área. Nos atemos apenas a dados de expressão gênica obtidos do TCGA, através do portal Xena Browser. Para analisar os efeitos do *pooling*, consideramos dois problemas pan-câncer: no primeiro, procuramos classificar o tipo de câncer de diversas amostras, tumorais ou não; no segundo, predizemos se uma exemplo corresponde a um tumor ou a uma amostra normal, independentemente do tipo de câncer do conjunto original. Além dessas tarefas pân-cancer, consideramos o problema de distinguir amostras tumorais de normais em 6 datasets contendo apenas um grupo de amostras. Especificamente, consideramos BRCA, KIRC, KICH, LUAD, LUSC e ESCA. Em todos os problemas, os valores de entrada correspondem à expressão de cada gene na amostra, e cada valor é associado a melhor proteína correspondente na rede STRING.

Entre nossos resultados, descobrimos que múltiplos níveis de agrupamento do grafo têm um impacto geral negativo no desempenho, mas que isso pode ser parcialmente contornado quando o *pooling* com pesos e as convoluções de grafo são usadas. Mostramos também que esses modelos levam a genes significativos quando são interpretados usando métodos baseados em gradientes, muitos dos quais foram estudados anteriormente no contexto de cânceres e terapias contra o câncer. Como exemplo, identificamos o gene LATS1, que codifica a quinase *Large Tumor Suppressor Kinase 1*, já previamente associada na literatura biológica a múltiplos tipos de tumores e ao crescimento de células cancerígenas. Além disso, interpretamos os modelos nos níveis de menor resolução dos grafos, gerados por meio das operações de agrupamento, e descobrimos que os supernodos, relacionados aos agrupamentos de genes no grafo de entrada, estão frequentemente super-representados em processos biológicos associados a câncer.

Como subproduto de nossos experimentos, observamos que os modelos pan-câncer alcançaram um alto desempenho em comparação com os modelos específicos para câncer na tarefa de distinguir amostras tumorais de amostras normais, além de termos obtidos genes relacionados simultaneamente a um grande número de cânceres quando interpretamos os modelos. Por causa disso, também exploramos neste trabalho como a inclusão de amostras de diferentes grupos de cânceres poderia melhorar os resultados em tarefas de classificação para grupos específicos, focando apenas nas redes neurais tradicionais. Especificamente, estudamos duas questões. Na primeira, consideramos a tarefa de distinguir tumor de não-tumor para um tipo específico de câncer (por exemplo, BRCA). Verificamos então se a inclusão de amostras de outros grupos, como BLCA, KICH, etc, melhoraria os resultados. Realizamos esse experimento para cada um dos 16 tipos de câncer considerados. Além disso, examinamos o problema de predizer tumores de tipos de câncer que não teriam sido observados durante o treino. Para tanto, treinamos um modelo pan-cancer utilizando 15 grupos de câncer, e separamos um para teste. Realizamos o experimento para cada um dos 16 tipos de câncer Comparamos então os resultados com aqueles obtidos treinando modelos usando apenas amostras do grupo específico de câncer modelado.

No que diz respeito à primeira questão, constatamos que a inclusão de amostras de outros grupos reduz a variância da performance nos dados de teste em comparação com os modelos de referência, atingindo assim performances quase sempre maiores ou iguais. Em particular, notamos que os ganhos são mais significativos quando o conjunto de dados do tipo de cânce considerado tem poucas amostras e é desbalanceado. Os resultados também foram encorajadores para a segunda questão, a respeito da capacidade dos modelos pan-câncer de distinguirem tumores de amostras normais para grupos de câncer não vistos. Neste caso, observamos que as melhores performances eram obtidas quando havia proximidade anatômica entre os grupos de câncer, como, por exemplo, no caso de COAD e READ.

APPENDIX B — LAPLACIAN OF A GRAPH

In this section we wish to provide an intuitive argument for the equation of the Laplacian of a graph. To do that, we imagine a a discrete signal and view it as a circulant graph, where connection between two vertices indicates adjacency in the domain of the discrete signal. The Laplacian of this discrete signal can be obtained by considering a discrete approximation of the operator,

$$\frac{d^2 f(x)}{dx^2} \approx \frac{1}{h}(f'(x+h/2) - f'(x-h/2)) \tag{B.1}$$

$$= \frac{1}{h}\left(\frac{f(x+h)-f(x)}{h} - \frac{f(x)-f(x-h)}{h}\right) \tag{B.2}$$

$$= \frac{1}{h^2}(f(x+h)+f(x-h)-2f(x)). \tag{B.3}$$

The discrete version can then be taken by letting $h = 1$ and considering a discrete version of f, $f[n]$ with $n \in \mathcal{Z}$. To generalize this result for a general graph, we interpret the term $f[n-1]$ and $f[n+1]$ as the values of f at the neighbors of vertex n and the term $2 * f[n]$ as the value of f at vertex n multiplied by its degree (we note that this notion is valid also if we consider a rectangular grid and analyze it as a graph). With that, we can write the result of the graph Laplacian operator applied to the signal f, for each node i, as

$$\mathbf{L}f[i] = \sum_{j \in \mathcal{N}_i} f[j] - f[i]|\mathcal{N}_i| = (\mathbf{A} - \mathbf{D})f. \tag{B.4}$$

In the general case where we have weighted edges, the Laplacian matrix of a graph is

$$\mathbf{L} = \mathbf{W} - \mathbf{D} \tag{B.5}$$

.

APPENDIX C — T-SNE PLOTS OF THE GRADIENTS

As discussed in Section 4.4, one can analyze the importance of a feature of an input (or its latent representation) $\mathbf{x} \in \mathbb{R}^M$ by considering the point-wise absolute value of the gradient of the output with respect to the input,

$$\mathbf{s} = \left| \left. \frac{\partial S_c}{\partial \mathbf{x}} \right|_{x_0} \right| . \tag{C.1}$$

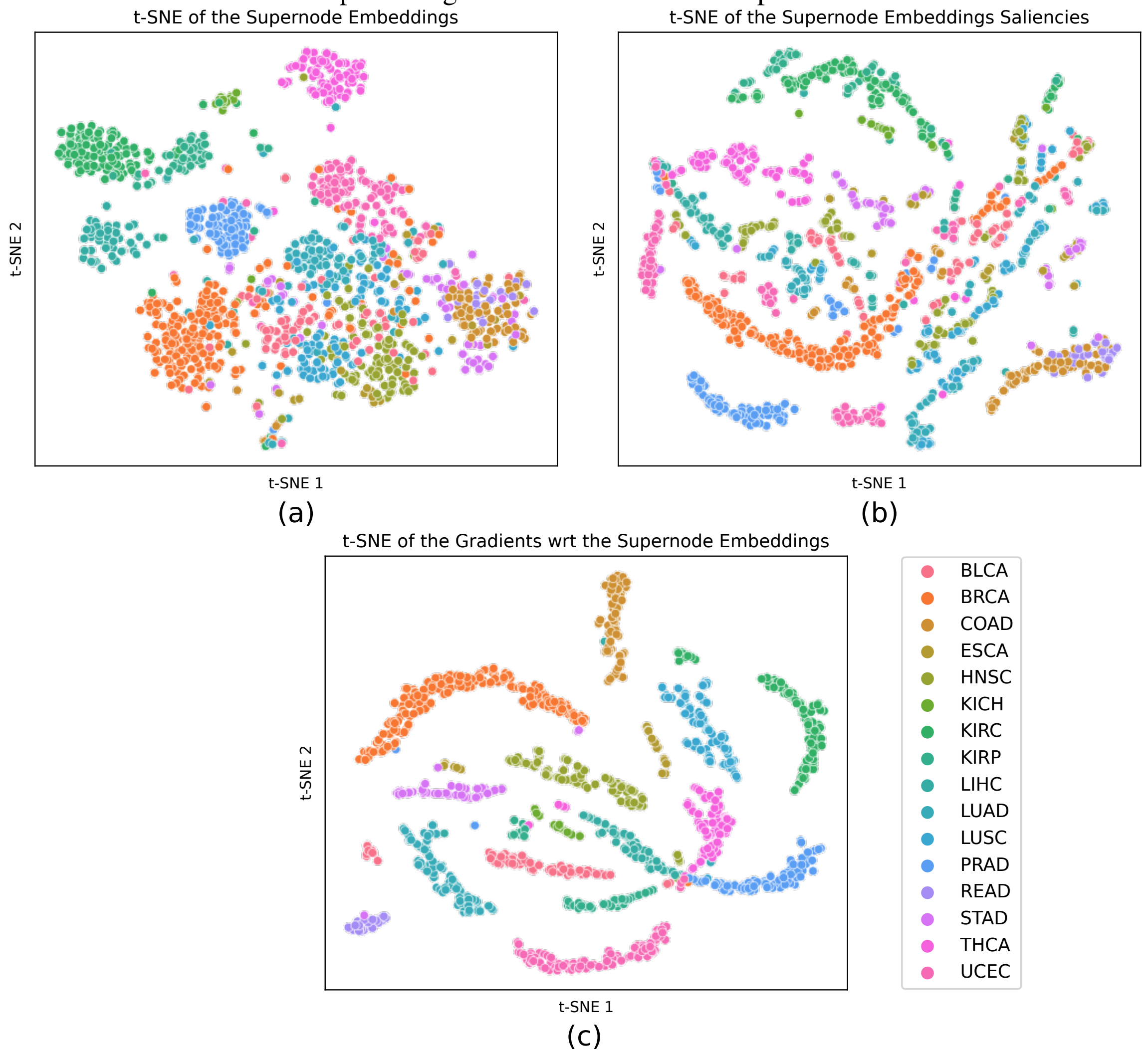
One could also wonder, however, if it would not be interesting to consider the gradient itself instead of its absolute value. The reasoning for that follows from the first-order approximation to the class output. We consider sets of parameters $\mathbf{w}_c \in \mathbb{R}^M$ that are dependent on the class c, such that

$$S_c(\mathbf{x}) \approx \mathbf{w}_c^T \mathbf{x} + b. \tag{C.2}$$

Then it is possible that for a class c_1, a positive variation in the i-th feature of $\mathbf{x}$, x_i, will result in a positive variation of $S_c(\mathbf{x})$, as long as w_{1i} (the weight associated with feature i for class c_1) is also positive. On the other hand, a class c_2 could have a negative w_{2i}, such that a *negative* variation in x_i would induce an increase in the class score S_{c_2}. In other words, considering the gradient instead of its absolute value allows one to identify if the score for a certain class reacts positively to an increase or to a decrease of an input feature.

We discuss this here because we have observed this behaviour when analyzing the gradients of the embeddings of the supernodes. We show the supernode embeddings, their saliencies and their gradients as well in Figure C.1. It is interesting to observe that the gradients cluster the cohorts better than the saliencies are able to. One hypothesis that could explain this is based on the idea that the model reacts differently to an increase or decrease of an input feature depending on the class. The reason for that could either be biological, or could come from the fact that the model puts more weight to a class when it observes that the sample is *not* from another class. In particular, we note that the READ and COAD classes are overlapping when visualized using the saliencies, but not in the gradients plot. This is an indication that the model uses the same features to classify both of them, but in different ways.

Figure C.1 – Supernode embeddings (a), their saliencies (b) and the gradients (c) produced when predicting the cohort of the test samples.



Source: The Author