UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

FELIPE COLOMBELLI

Assessing the Applicability of Graph Neural Networks for Cancer Staging Using Sample Similarity Networks

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Science

Advisor: Prof. Dr. Mariana Recamonde Mendoza

Porto Alegre
September 2022

AGRADECIMENTOS

Agradeço primeiramente aos meus pais, Margarete Rathke Colombelli e Alencar José Colombelli, que desde cedo sempre me ensinaram sobre a importância dos estudos por meio de conversas e principalmente exemplos. Agradeço também ao meu irmão, Roberto Colombelli, por ser o meu principal farol nessa jornada em busca pelo conhecimento, o qual de forma completamente orgânica e indireta me ensinou sobre as coisas mais importantes da vida, e, sobretudo, aquelas invioláveis que jamais podemos deixar de lado. Obrigado por todo o amor e proteção, sem vocês eu jamais estaria onde estou.

Este trabalho, porém, também foi resultado da influência de inúmeras outras pessoas que participam ou participaram da minha vida, sem as quais esta seção de agradecimentos estaria incompleta. Agradeço de forma especial:

Aos meus outros importantes membros da família que estiveram sempre me apoiando e nunca deixaram de acreditar em mim: Fabi, Ju e tia Jô. E especialmente à tia Tela, Jaci e Darlei pelo suporte incrível e por me proporcionarem um lugarzinho em Porto Alegre para chamar de minha casa.

Aos meus queridos amigos do interior pelos momentos mais felizes que vivi e por comporem a base da minha saúde mental, em especial: Arthur, Lou, Media, Vivi, Vic, Uel, Theo, Mestre, Shosha e Luci.

Aos meus queridos amigos de Porto Alegre, por todo o apoio e momentos igualmente incríveis, em especial: Amaury, Barboza, Bruno, Clebinho, Ciça, Henrique, Hummes, Luís, Luma, Pacheco, Paola, Rafael e Tanara.

Aos professores que fizeram parte da minha formação e/ou contribuiram significativamente para meu interesse nas ciências exatas e ciências naturais, em especial: Mara Maciel, Sara Wachholz, Marilene Redin Rachele, Samuel Volkweis Leite, André Grahl Pereira, Rodrigo Machado, Luciana Salete Buriol e Ana Bazzan.

Aos professores Anderson Rocha Tavares e Joel Luís Carbonera pelo aceite na participação como banca do trabalho e pelas contribuições que enriqueceram as discussões aqui realizadas.

Aos meus mais que orientadores, Profa. Érika Cota, Profa. Mariana Recamonde Mendoza, Profa. Ursula Matte e Prof. Gabriel de Oliveira Ramos. Especialmente à Professora Mariana por me introduzir formalmente ao universo acadêmico e nutrir em mim os valores e habilidades que acredito serem indispensáveis para qualquer cientista sério que aspire contribuir para um mundo melhor com o seu trabalho e busca pelo conhecimento.

Aos *web-influencers* responsáveis diretos pelo meu interesse insaciável pela ciência e construção da minha mentalidade questionadora e ceticista durante meus anos de escola: Yuri Grecco (do antigo EuAteu), Paulo Miranda Nascimento (Pirulla) e Guilherme Tomishiyo.

Em especial, gostaria de agradecer três amigos que foram fundamentais durante a minha jornada na construção deste trabalho: Fiba, Rafael e Amaury. Agradeço imensamente por amenizarem minhas angústias, presenteando-me com suas companhias nesta última etapa do curso. Obrigado por me levarem para sair e aliviar a cabeça, por ouvirem meus choros e vivas, por me aconselharem, por me incentivarem a desenvolver meu lado pessoal e olhar para além do profissional, pelos memes, risadas, curiosidades, cultura e todo o resto, obrigado, sem vocês estes dias teriam sido muito mais difíceis.

Por fim, gostaria de agradecer individualmente mais dois amigos, Victor Guerreiro Vendruscollo e Cléber Henrique de Araújo Gama, aos quais dedico este trabalho.

Obrigado, Vic, por me ensinar o que é matemática "de verdade", transformando completamente minha vida ao mostrar de onde vinha a equação da diagonal do quadrado (e o quão óbvio era esse fato), incentivando-me a aprender geometria – dentre outras áreas da matemática – através da lógica em detrimento da pura memorização. Obrigado, amigo, por fazer o que muitos outros professores não foram capazes; por me iluminar os caminhos mais incríveis da matemática; por mostrar que eu também era capaz e que eu poderia escolher a ciência da computação como formação sem medo algum. Foste de extrema importância para chegar onde cheguei e almejar o que almejo.

Também te agradeço imensamente, Clebinho, pelo crescimento pessoal exponencial que me proporcionaste nos anos intensos que passamos muito próximos e unidos. Obrigado por me fazer refletir sobre coisas de suma importância que nunca antes tinha gasto tanta energia pensando; por me ensinar sobre o valor incalculável da amizade e do amor, sobre a importância do diálogo e dos sentimentos, sobre o funcionamento das pessoas, sobre a necessidade da coragem para agir e se posicionar, sobre o valor dos momentos compartilhados com outras pessoas e do agora, sobre o mundo, o universo e tudo mais, obrigado.

ABSTRACT

Cancer staging is a challenging classification task in which, given the samples' characteristics, the employed strategy needs to categorize them into typically one out of four stages. As more public biological data becomes available, such task starts receiving more attention from the scientific community, and questions like the integration and how to use these varied sources of information emerge. Because of the classification task's complexity, employing accurate machine learning models could significantly help in related clinical practices since the cancer stage information is crucial for adopting a successful patient's treatment. In particular, deep learning strategies can be very useful as they have been successfully applied in a wide range of similarly difficult classification tasks. With that in mind, our study proposes to investigate the applicability of a data modeling approach based on sample similarity networks to deal with this multi-sourced information, shifting the problem's representation to a node classification problem. The Graph Attention Network and Graph Convolutional Network algorithms are applied for classifying the samples and their performance is compared to a more traditional Multilayer Perceptron algorithm. Our main hypothesis, supported by similar studies, is that, by introducing something like the samples' correlation as a measure of similarity, the ones with the same class will tend to be highly correlated and form a connection in the network, thus, helping in the node classification task that typically assumes the neighborhood influences in a node's characteristics. Additionally, with such problem representation, we can also achieve greater flexibility regarding the data modeling, allowing even semi-supervised learning techniques to be used. After analyzing the results, we observed no significant performance gains by using the network-based strategy compared to the Multilayer Perceptron algorithm.

Keywords: Cancer staging. multi-omics. similarity networks. graph neural networks.

Avaliando a aplicabilidade de *Graph Neural Networks* para estadiamento tumoral utilizando redes de similaridade de amostras

RESUMO

O estadiamento tumoral é uma tarefa de classificação desafiante na qual, dadas as características das amostras, a estratégia empregada deve as categorizar em tipicamente um dos quatro estágios tumorais. À medida em que mais dados biológicos públicos se tornaram disponíveis, tal tarefa começou a receber mais atenção da comunidade científica, dando origem a questões como a integração e a forma de utilizar estas variadas fontes de informação. Devido à complexidade da tarefa de classificação, o emprego de modelos de aprendizado de máquina acurados tem a possibilidade de ajudar significativamente nas práticas clínicas relacionadas, uma vez que a informação do estágio tumoral é crucial para a adoção de um tratamento bem sucedido do paciente. Em particular, estratégias de aprendizado profundo podem ser muito úteis, visto que têm sido aplicadas com sucesso numa vasta gama de tarefas de classificação igualmente difíceis. Tendo isso em mente, o nosso estudo se propõe a investigar a aplicabilidade de uma abordagem de modelagem de dados baseada em redes de similaridade de amostras para lidar com esta informação de múltiplas fontes, deslocando a representação do problema para um problema de classificação de nós em um grafo. Os algoritmos *Graph Attention Network* e G*raph Convolutional Network* são aplicados para classificar as amostras e o seu desempenho é comparado com um algoritmo mais tradicional, o *Multilayer Perceptron*. A nossa hipótese principal, apoiada por estudos semelhantes, é que, ao introduzir algo como a correlação das amostras como medida de similaridade, aquelas de mesma classe tenderão a estar altamente correlacionadas e a formar uma conexão na rede, auxiliando, assim, na tarefa de classificação de nós que tipicamente assume que a vizinhança influencia nas características de um nó. Além disso, com tal representação de problema, é possível também alcançar uma maior flexibilidade no que diz respeito à modelagem dos dados, permitindo inclusive a utilização de técnicas de aprendizado semi-supervisionado. Após a análise dos resultados, não observamos quaisquer ganhos significativos de desempenho ao utilizar a estratégia baseada em redes se comparada com o algoritmo tradicional *Multilayer Perceptron*.

Palavras-chave: estadiamento tumoral. multi-ômicas. redes de similaridades. graph neural networks.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

DNA	Deoxyribonucleic acid
RNA	Ribonucleic acid
mRNA	Messenger RNA
miRNA	MicroRNA
FPKM	Fragments per kilobase of transcript per million mapped reads
SC	Spectral Clustering
ML	Machine Learning
ANN	Artificial Neural Network
MLP	Multi-Layer Perceptron
GNN	Graph Neural Network
GAT	Graph Attention Network
GCN	Graph Convolutional Network
FL	Focal loss
GDC	Genomic Data Commons
TCGA	The Cancer Genome Atlas Program
COAD	Colon adenocarcinoma
KIRC	Kidney renal clear cell carcinoma
LUAD	Lung adenocarcinoma
CVD	Cardiovascular disease
CPD	Chronic pulmonary disease
CGEN	Correlation network based on gene expression levels
CMON	Correlation network based on multi-omics data
SNF	Similarity Network Fusion
DAE	Denoising Autoencoder

CONTENTS

1 INTRODUCTION

According to Ahmad and Anderson (2021), cancer is one of the leading causes of death in the US, only behind heart disease. While cancer can be very deadly, its survivability is intrinsically linked to how advanced the disease is at the time of the diagnosis and the start of the intervention. Cheaib et al. (2020) report a 5-year renal cell carcinoma survival rate after nephrectomy of 97.4% for disease stage I (out of four), reaching as low as 26.7% for disease stage IV. Additionally, the cancer stage assessment is clinically important for planning a proper therapeutics or surgical intervention (CHEAIB et al., 2020). Although cancer stage classification is such an important task, it remains rather underexplored. The majority of research efforts focus on cancer diagnosis and prognostic prediction, leaving the clinical practices for tumor stage assessment with obsolete long-standing techniques (YU et al., 2020), which is highly undesirable considering the informative molecular data we have been conquering with recent technology advancements.

Among the most prominent data with the capability to help solve this question is the omics data. Cancer research has exploded with the increasingly publicly and freely available data provided by initiatives such as The Cancer Genome Atlas (TCGA) (PRIOR et al., 2013; KOUROU et al., 2015), thanks to the advancements in computational power and the emergence of high-throughput sequencing technologies (REUTER; SPACEK; SNYDER, 2015). This handful source of data has attracted a lot of attention from machine learning researchers, posing challenging tasks like cancer diagnosis, early detection, classification, staging, and treatment response prediction (BERTSIMAS; WIBERG, 2020). There are also a number of efforts to uncover new potential cancer biomarkers by utilizing techniques such as feature selection (ZHANG; JONASSEN; GOKSØYR, 2021) and ensemble feature selection (TREVIZAN; RECAMONDE-MENDOZA, 2021; COLOMBELLI; KOWALSKI; RECAMONDE-MENDOZA, 2022).

As will be discussed in Section 2.1, there are different kinds of omics, depicting the biological state of the organism through different perspectives. The gene expression data, under the realms of transcriptomics data, is among the most utilized ones, as it translates to a closer quantification of the proteins being expressed in the body (SUPPLITT et al., 2021). However, other omics such as miRNomics and methylomics are also important indicators of disease establishment. Thus, some efforts have been made to integrate the multiple omics for joint analysis, giving birth to the multi-omics investigation topic (MANZONI et al., 2018).

Among the integration techniques, similarity-based networks have been receiving some attention lately, with methods like the Similarity Network Fusion (WANG et al., 2014). These methods build a graph from the tabular data, using nodes to represent samples and the edges between these nodes to represent a significant similarity between them. An example could be a patients network in which two patients are connected if their gene expression levels have a Pearson correlation coefficient of at least 0.7. For measuring these similarities, different methods could be used, like the linear correlation between the samples or their euclidean distance. Some form of sparsification is also usually required for preventing the resulting sample-sample similarity matrix to produce a dense graph, in which all the nodes are connected to each other.

As reviewed in Chapter 3, very little effort has been made to investigate the application of such data modeling techniques to cancer-related questions, like diagnosis. However, the application of machine learning algorithms for cancer diagnosis and subtyping modeled as a node classification problem in sample similarity networks demonstrated promising results in previous studies (PAI et al., 2019; ZHANG et al., 2022; BAUL et al., 2022).

In particular, Graph Neural Networks (GNNs) are successful recently proposed deep learning algorithms specially designed to deal with learning tasks on graphs, taking advantage of the relational non-euclidean structure of this type of data. This characteristic makes these algorithms of special interest to the presented node classification problem. Nonetheless, for the two only other works exploring patient similarity networks based on omics data we could find (ZHANG et al., 2022; BAUL et al., 2022), neither of them detailed, for example, the sparsification strategy employed (if any) to the networks construction process. The sparsification greatly impacts the final modeled sample-sample network and, thus, should be investigated more closely. Additionally, one of the mentioned studies did not utilize multi-omics data for the task they were investigating. To the extent of our knowledge, this is the first work investigating GNN algorithms for cancer staging using a patient similarity network computed from multi-omics data.

The questions we are interested in answering through our analyzes are:

1. How well can GNNs applied to multi-omics patient similarity networks perform for the cancer staging task?
2. Do these GNNs perform better than simpler and computationally cheaper counterparts, like a multilayer perceptron applied to the integrated tabular data?
3. What are the impacts on classification performance yielded by different sample

similarity network generation strategies and thresholds?

4. Are the observed results consistent among different cancer datasets?
5. What are the best hyperparameters for each investigated algorithm?

Regarding question 3 specifically, we argue that threshold-based approaches for sparsifying adjacency matrices that resulted from pair-wise similarity computations are more adequate than using top-*k* methods. Top-*k* methods connect each node in the network with its *k* most similar other nodes. This type of sparsification can not result in a typical real-life complex network, where some nodes, also known as hubs, are highly connected to other nodes, while some other nodes have significantly small degrees (BARABÁSI, 2013). Additionally, because we deal with different similarity calculations, we chose to investigate the application of quantile-based thresholds instead of a specific number. This choice enables a more appropriate comparison between different similarity computation strategies.

In the remaining text we provide the reader with the necessary background for understanding and reproducing the investigations (Chapter 2); review the related literature (Chapter 3), pointing out findings and gaps when appropriate; present our adopted investigation methodology (Chapter 4); discuss and reason about our results (Chapter 5); and finally conclude the study (Chapter 6) wrapping up with the significant findings, pointing out some limitations as well as research opportunities. We note that some findings are surprisingly not in accordance with previous literature; for this, we present some possible explanations for why this was observed in our experiments.

2 BACKGROUND

While many computational methods could be used for the task of cancer staging, our work focuses on a promising direction of research centered on the sample similarity network problem modeling. Such structures could be effectively exploited by a recent and very successful class of algorithms: the Graph Neural Networks (GNNs). In particular, we model our problem as node classification and apply both the Graph Attention Network (GAT) and the Graph Convolutional Network (GCN) algorithms.

In this chapter, we describe in detail the selected methods and techniques that will guide the investigation. We start, however, by situating the reader in the biological context of the selected data, why it is important and informative for the classification task at hand, and what their numerical values actually mean.

2.1 Biological background and the omics data

Cancer disease is mainly characterized by uncontrolled cell growth and proliferation (WEINBERG, 2013). This abnormal cell function is driven by a lot of complex biological processes, being always associated with genetic and epigenetic changes accumulated within the cell (SUPPLITT et al., 2021). The genetic component of such a disrupted system is mainly characterized by the gene expression levels of the organism. Figure 2.1 illustrates the different omics interested in gene expression levels. As Supplitt et al. (2021) points out, genomics is the farthest from the actual phenotype, since only a small percentage of the genome is expressed to the protein level. The final gene expression is a product influenced by numerous regulation processes and molecules, such as miRNAs and DNA methylation.

Although proteomics represents a much closer picture of the cell phenotype (QIN; NIU; ZHAO, 2019), its quantification technologies, such as protein microarrays, are very limited due to the challenges associated with their utilized techniques and the higher costs (BHAWE; AGHI, 2015). Thus, transcriptomics has become a key element for analyzing gene expression levels in search of answers to various biological questions related to cancer disease. The transcripts are RNAs that were transcribed using the DNA sequence. In particular, messenger RNAs (mRNAs) are the transcripts that will actually code proteins, which will perform various functions within the organism, determining its phenotype. The number of transcripts produced by a cell can be quantified and characterized by some

Figure 2.1 – Gene expression measurement approaches.

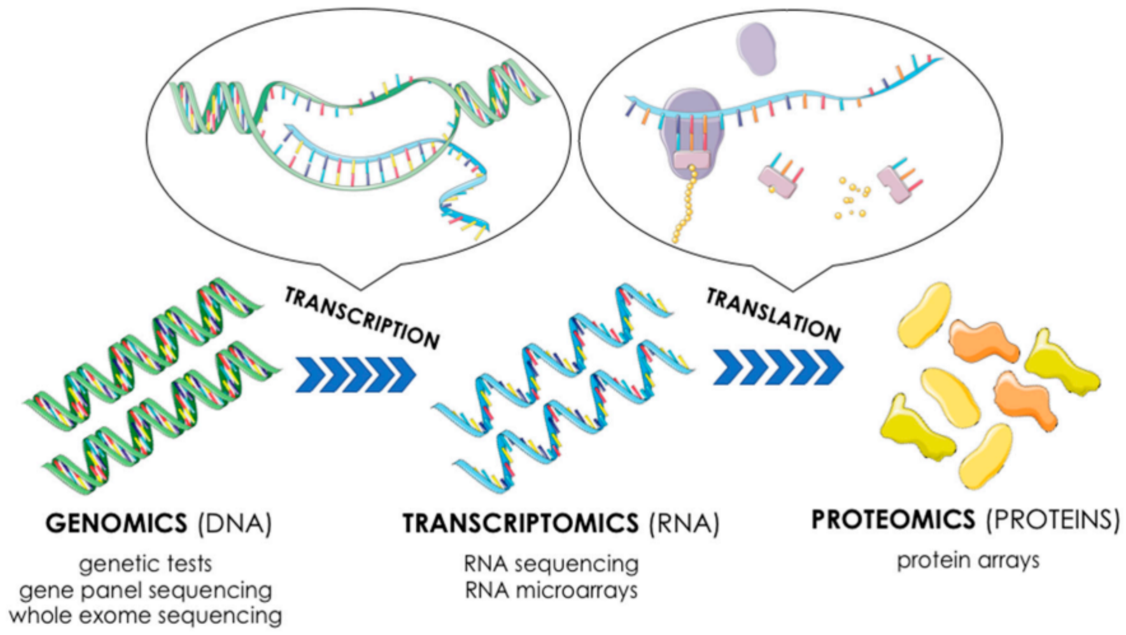


Image from Supplitt et al. (2021).

methods, such as mRNA Sequencing (mRNA-Seq).

RNA-Seq pipelines usually isolate the mature mRNA and break it into small fragments (typically around 100 bp). After that, these fragments are converted into double-stranded DNA, which are more stable molecules than RNA itself and, thus, more suitable to be sequenced. This DNA is then sequenced using high-throughput sequencing technologies (REUTER; SPACEK; SNYDER, 2015), which can identify the nucleotides of the fragments. Finally, the sequenced strands can be aligned to a reference genome, which allows the researcher to determine which regions were being transcribed and annotate the gene expression levels of the organism under investigation. We refer the reader to Lowe et al. (2017) for a more detailed explanation of RNA-Seq methods. Figure 2.2 illustrates the explained process.

The final product of this whole pipeline for data acquisition is the raw counts. This data can be carried out in a bulk, single cell, or spatial RNA-Seq analysis. The bulk RNA-Seq averages the counts of a pool of cells, while the single cell RNA-Seq uses the counts of each cell separately, and the spatial RNA-Seq dissects the RNA activities spatially (LI; WANG, 2021).

These counts are also normalized with techniques such as the fragments per kilobase of transcript per million mapped reads (FPKM). The FPKM of a sample can be calculated using Equation 2.1 (ZHAO et al., 2021), in which q_i represents the count of reads

Figure 2.2 – Typical RNA-Seq pipeline. In the *in vivo* part of the pipeline, the mature RNA is isolated; the *in vitro* procedures include the RNA fragmentation, conversion into double-stranded DNA and sequencing; finally, in the *in silico* step, the sequences are aligned to a reference genome. Detailed information can be found in Lowe et al. (2017).

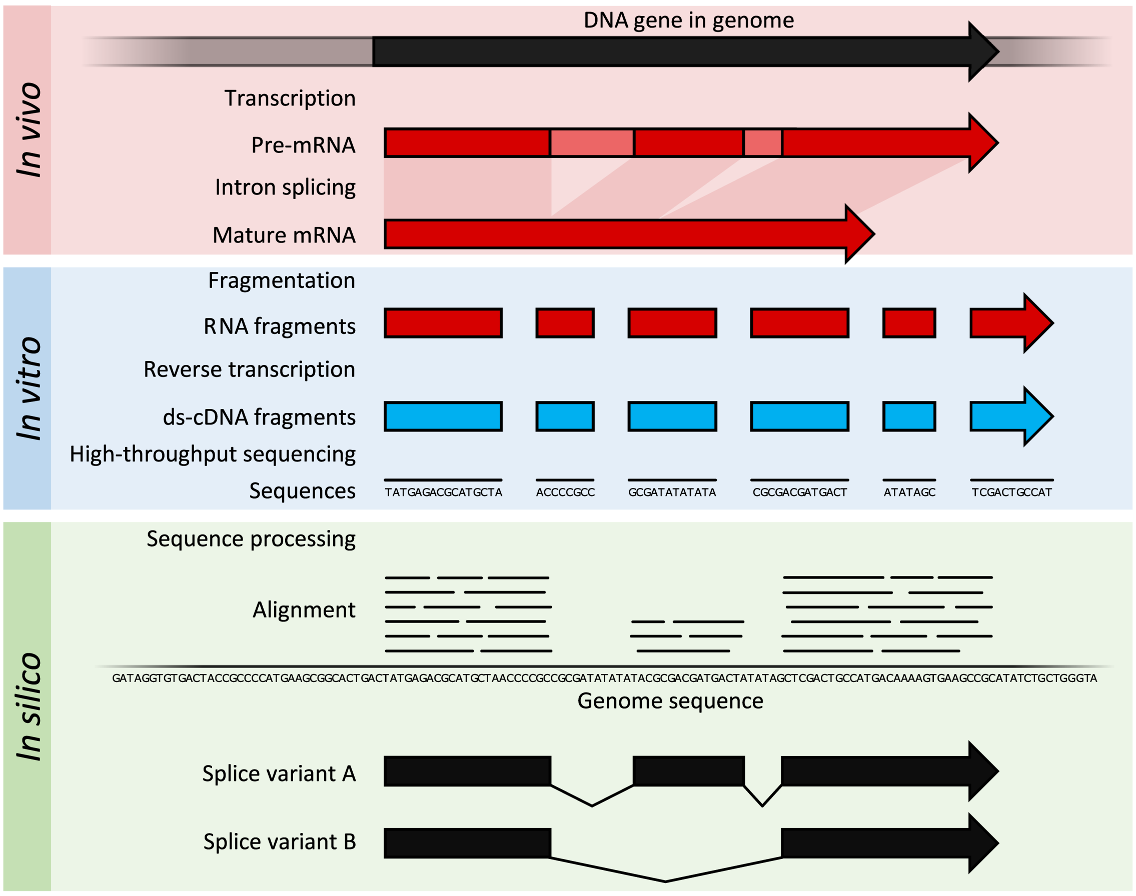


Image from Lowe et al. (2017).

aligned to gene/transcript i, l_i represents the length of this gene/transcript, and $\sum_{j=1} q_j$ the total number of reads.

$$FPKM_i = \frac{10^9 q_i}{l_i \sum_{j=1}(q_j)} \tag{2.1}$$

Besides the mRNA, RNA-Seq technologies can also be used to quantify the expression level of other important transcripts, such as miRNAs. These small non-coding RNAs are endogenous molecules that have been reported to regulate the gene expression levels of several oncogenes and tumor suppressor genes. For example, Reddy (2015) reports that data collected from several different cancer tissues consistently show aberrant miRNA expression. The author mentions the involvement of these abnormal miRNA expression levels in many types of cancer, including breast, colon, gastric, lung, prostate, and thyroid.

DNA methylation was originally associated with gene silencing and generally refers to the addition of methyl groups to the cytosine carbon rings (LI; TOLLEFSBOL, 2021). This chemical interaction is said to be an epigenetic modification, which means that it occurs naturally depending on different environmental exposures. DNA methylation influences local gene expression levels (SCHÜBELER, 2015) in addition to playing important roles in other biological processes, like cellular identity establishment and maintenance (LI et al., 2019). Interestingly, as pointed out by Li and Tollefsbol (2021), DNA methylation has been widely appreciated as a reversible change, and it is highly associated with carcinogenesis (ALVAREZ et al., 2011), motivating a lot of research efforts from the scientific community to investigate its potential clinical applications for cancer disease, as reviewed by (LOCKE et al., 2019).

For measuring the DNA methylation, some technologies like Infinium HumanMethylation450 BeadChip[1] have been proposed, which covers over 450,000 methylation sites per sample at single nucleotide resolution. At each site, the platform measures the intensity of the methylated signal (M), and the intensity of the unmethylated signal (U). A β-value is then calculated using the formula $M/(M + U + 100)$. This value is continuous ranging between 0 (indicating 0% methylation) and 1 (indicating 100% methylation) for each probe corresponding at each DNA site (LI; TOLLEFSBOL, 2021). After additional data processing steps (see Wang, Wu and Wang (2018) for detailed information), the dataset ends up with around 450,000 features for the mentioned technology. A common post-processing technique utilized in this type of data is to select only the probes

[1]https://www.illumina.com/documents/products/datasheets/datasheet_humanmethylation450.pdf

associated with promoter regions, mapping them to particular genes, and, since multiple probes could be associated with the same gene, take a statistics measurement out of those probes, for example, the median value (DUAN et al., 2021). Then, the resulting dataset's features become genes, resulting in, usually, around 20,000 to 24,000 features.

As discussed, all the presented types of omics data provide valuable biological insights that could be useful for investigating complex genetic diseases like cancer. With initiatives like The Cancer Genome Atlas (TCGA), a huge amount of publicly available omics data has unlocked several research efforts dedicated to analyzing it for answering various scientific questions. One of the most promising research trends has been investigating how to integrate all the different omics data, which provide information about the samples from different biological perspectives, and analyze this data with machine learning (ML) methods (NICORA et al., 2020), as we will discuss in greater detail in Chapter 3.

2.2 Similarity Network Fusion

Section 2.1 explained three types of omics data, pointing out the importance of integrating them for analyzing patients with a complex diseases such as cancer. The Similarity Network Fusion (SNF) method was proposed by Wang et al. (2014) with the goal of integrating this multi-sourced data through sample similarity networks. The algorithm computes the similarity of each pair of samples, *e.g* patients, separately, establishing a dense adjacency matrix for each type of omic. After that, it fuses these matrices through an iterative process into one final aggregated network, which can be useful to perform analyses such as disease subtyping, diagnosis, prognosis, *etc*. Figure 2.3 extracted from the original article, illustrates this process.

For formally defining the method, we use the same notation and symbols as described in the original work (WANG et al., 2014). Thus, the reader must be aware that this is a special section in the manuscript with its own notation.The matrices in Figure 2.3b are calculated by a euclidean distance with scaled exponential similarity kernel, as denoted by Equation 2.2. Assume $\{x_1, x_2, ..., x_n\}$ as the n patients (samples); $\mathbf{W}$ as the $n \times n$ similarity matrix; $\mathbf{W}_{(i,j)}$ as the similarity between nodes x_i and x_j; $\rho(x_i, x_j)$ as the euclidean distance between nodes x_i and x_j; μ as a hyperparameter, which the authors recommend to be set between $[0.3, 0.8]$; mean$(\rho(x_i, N_i))$ the average distance between x_i and each of its neighbors; and $\varepsilon_{(i,j)}$ calculated by Equation 2.3.

Figure 2.3 – Illustration of the SNF method applied to a scenario in which the samples are patients and the omics data are composed of mRNA and DNA methylation profiles. In **a** the original data is represented; **b** shows the resulting similarity matrix for each data type; **c**, the adjacency matrices represented as graphs, in which the nodes are the patients and the edges, their similarities; **d** represents the iterative fusion process happening; and **e** shows the final SNF fused graph. The edge colors represent their origin.

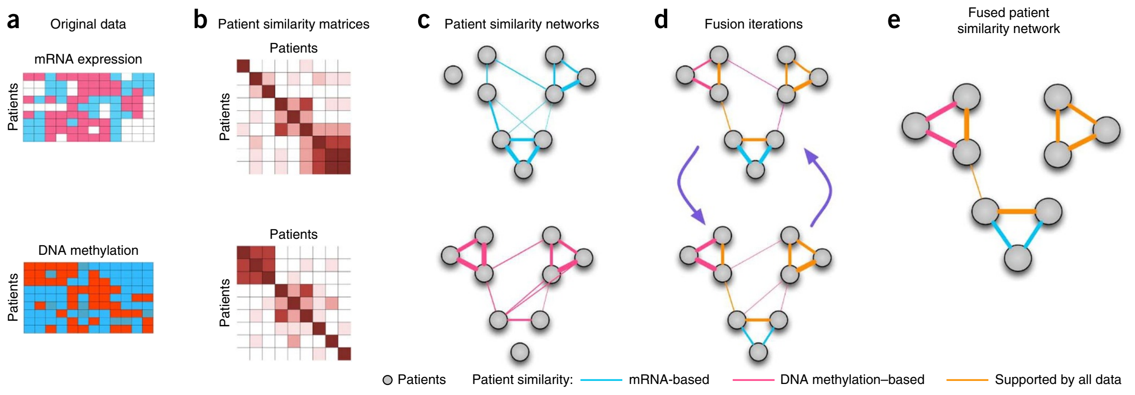


Image from Wang et al. (2014).

$$\mathbf{W}_{(i,j)} = exp\left(-\frac{\rho^2(x_i, x_j)}{\mu\varepsilon_{i,j}}\right) \tag{2.2}$$

$$\varepsilon_{i,j} = \frac{\text{mean}(\rho(x_i, N_i)) + \text{mean}(\rho(x_j, N_j)) + \rho(x_i, x_j)}{3} \tag{2.3}$$

A full kernel, which is represented by the matrix $\mathbf{P}_{(i,j)}$, is calculated for computing the fused matrix. $\mathbf{P}_{(i,j)}$ is defined by Equation 2.4. This matrix applies a normalization that is free of the scale of the diagonals filled with ones (self-similarity) and has the property $\sum_j \mathbf{P}(i,j) = 1$.

$$\mathbf{P}_{(i,j)} = \begin{cases} \frac{\mathbf{W}_{(i,j)}}{2\sum_{k\neq i}\mathbf{W}_{(i,k)}}, & j \neq i \\ 1/2, & j = i \end{cases} \tag{2.4}$$

While the matrix **P** carries the full information about the similarity of each pair of patients, a new matrix **S** is defined for encoding the similarity to the K most similar patients for each patient. **S** is used as the sparse kernel matrix in the fusion process, measuring the local affinity, and it is calculated as defined by Equation 2.5, in which N_i is the set of neighbor nodes of x_i.

$$\mathbf{S}_{(i,j)} = \begin{cases} \frac{\mathbf{W}_{(i,j)}}{\sum_{k\in N_i}\mathbf{W}_{(i,k)}}, & j \in N_i \\ 0, & \text{otherwise} \end{cases} \tag{2.5}$$

Finally, for computing the SNF matrix in a problem with m data types, where $v = 1, 2, ..., m$, we start calculating the matrices **P** and **S** for each v, indicated with a superscript, and iteratively recompute the m matrices $\mathbf{P}^{(v)}$ using Equation 2.6 until convergence or a number of predefined iteration steps. In the end, all the $\mathbf{P}^{(v)}$ matrices are combined into a single final matrix, the SNF network result, by applying a simple average to the m matrices' values.

$$\mathbf{P}^{(v)} = \mathbf{S}^{(v)} \times \left(\frac{\sum_{k \neq v} \mathbf{P}^{(k)}}{m-1} \right) \times (\mathbf{S}^{(v)})^T, v = 1, 2, ..., m \tag{2.6}$$

2.3 Spectral clustering

Given a similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$, where n represents the number of samples, and k the number of clusters, the goal of the Spectral Clustering (SC) algorithm is to assign each node of the graph (represented by **S**) to one of the k clusters. While there are several possible variations for the SC algorithm (LUXBURG, 2007), we focus on the normalized spectral clustering according to Shi and Malik (2000), which is the guiding reference for the SC implementation we utilized in our analyzes.

The algorithm starts by constructing a similarity graph, deriving the weighted adjacency matrix **A**. The unnormalized graph Laplacian **L** is then computed, following Equation 2.7, in which $\mathbf{D} \in \mathbb{R}^{(n \times n)}$ represents the degree matrix where $\mathbf{D}_{(i,i)} = \sum_j \mathbf{A}_{(i,j)}$. After that, the algorithm computes the first k generalized eigenvectors $u_1, u_2, ..., u_k$ from the problem $\mathbf{L}u = \lambda \mathbf{D}u$. Assuming $\mathbf{U} \in \mathbb{R}^{(n \times k)}$ as the matrix containing $u_1, u_2, ..., u_k$ as columns, and $y_i \in \mathbb{R}^k$ as the vector corresponding to the ith row of **U**, the algorithm clusters the points $(y_i)_{i=1,2,...,n}$ with the k-means algorithm (HASTIE; TIBSHIRANI; FRIEDMAN, 2001) into clusters $\mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_n$. Finally, it outputs $\{\mathbf{Y}_1, \mathbf{Y}_2, ..., \mathbf{Y}_k\}$, with $Y_i = \{j | y_j \in C_i\}$.

$$\mathbf{L} = \mathbf{D} - \mathbf{W} \tag{2.7}$$

2.4 Multi-layer perceptron and the fundamentals of artificial neural networks

Multi-layer perceptron (MLP), also called feed-forward network (GOODFELLOW; BENGIO; COURVILLE, 2016), is a particular architecture of Artificial Neural

Networks (ANN) in which all the neurons in one layer are connected to all the neurons of the next layer. Assuming a typical classification problem, the goal of such structures, which can be thought of mathematical functions $f(\overrightarrow{x}, \mathbf{W}, \overrightarrow{b})$, is to approximate real functions f^* that map sample features $\overrightarrow{x}$ to a specific label. To do that, the ANNs utilize a matrix of learnable weights **W** and a learnable bias vector $\overrightarrow{b}$ that are optimized during training through a process called backpropagation.

When the network has more than one layer, it maps the input features into another feature space, which maps this second feature space to the output space or keeps using additional layers in between, called hidden layers, before mapping those hidden features to the actual final output space. For example, assuming an MLP with three layers, this network can be thought of as a chain of functions $f^{(3)}(f^{(2)}(f^{(1)}(\overrightarrow{x})))$, in which each one has its own weight matrix. The actual computations made by each layer are a multiplication between the features vector and the weight matrix. Because this type of computation can not model non-linear functions, the resulting vector or scalar from this multiplication can be submitted to what is called an activation function a. The first layer linear combination is mathematically defined by Equation 2.8, in which F represents the number of dimensions of the feature space; $w_{i,j}^{(1)}$ represents the value in the ith column and jth row of the first layer weight matrix $\mathbf{W}_1$; and j represents the jth component of the output vector produced by $f_j^{(1)}(\overrightarrow{x})$ function application, which could be a single scalar as well.

$$f_j^{(1)}(\overrightarrow{x}) = a\left(b_j^{(1)} + \sum_{i=1}^{F} w_{i,j}^{(1)} x_i\right) \tag{2.8}$$

Another equivalent mathematical representation is given by Equation 2.9.

$$f^{(1)}(\overrightarrow{x}) = a\left(\overrightarrow{b}^{(1)} + \mathbf{W}_1 \overrightarrow{x}\right) \tag{2.9}$$

Of note, traditional deep learning books and other references like Goodfellow, Bengio and Courville (2016) may denote the linear combination $\mathbf{W}\overrightarrow{x}$ with a transpose symbol ^{T} in **W** or $\overrightarrow{x}$. Here, we can simply directly assume that $\overrightarrow{x} \in \mathbb{R}^F$ and $\mathbf{W} \in \mathbb{R}^{F' \times F}$, where F' represents the size of the output space.

The presented theory only accounts for the feed-forward process. To actually learn the weights and biases, the model (an instanced MLP architecture) needs to be trained. Usually, this training happens in a supervised fashion, in which some data samples $\overrightarrow{x}$ are provided as examples along with their respective expected output $f^*(\overrightarrow{x})$. The network starts with random weights, and feed-forwards the input $\overrightarrow{x}$, producing an output that is compared with the expected $f^*(\overrightarrow{x})$. This comparison is computed by a differentiable cost/loss function, like the mean squared error, and the gradient with respect to the learnable parameters is calculated. This gradient indicates in which direction the parameters must change in order for the loss function to increase. Thus, the negative gradient is used in order to lower the loss. A learning rate scalar value is applied to adjust the weights in a controllable manner towards the reduction of the loss function. There are several sophisticated algorithms for performing such a process. We refer the reader to Abdolrasol et al. (2021) as a starting point for acquiring detailed information about these optimization algorithms.

When the trained model has a lot less error rate for the training data compared to the test data, it is said to be overfitting (BILBAO; BILBAO, 2017). The overfitting phenomenon is undesirable since it means that the model could not generalize well the problem, appropriately approximating the real function. This could happen due to several reasons, and techniques known as regularization can be used for trying to solve the issue (GOODFELLOW; BENGIO; COURVILLE, 2016). A simple technique for avoiding overfitting is the early stopping, in which a patience term p is defined. When a performance improvement in the validation data is not observed for p epochs, the model training halts. Another simple yet very effective regularization technique is the dropout, in which random input signals of each layer are discarded (zeroed) during the network training. A given dropout rate defines the percentage of units to drop from the network, which are selected at random at each training step. This technique was found to improve the performance of ANNs in a wide range of problem domains, including in the analysis of biology computational data (SRIVASTAVA et al., 2014).

2.5 Graph Convolutional Networks

The Graph Convolutional Network (GCN) algorithm proposed by Kipf and Welling (2017) works based on the assumption that a node's properties in a network are highly influenced by its neighbor nodes. It works as a features smoothing, which is based on an aggregation of the node and its neighbors that undergoes through a standard neural network function. The proposed algorithm explicitly uses an $\tilde{\mathbf{A}}$, which is simply the adjacency matrix $\mathbf{A}$ with all the nodes having a self-connection, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, where $\mathbf{I}_N$ is the identity matrix. $\tilde{\mathbf{D}}$ is the degree matrix of $\tilde{\mathbf{A}}$ defined analogously to $\mathbf{D}$ of Section 2.3.

Before going through any further detail, we simplify the other equations by defining the matrix $\hat{\mathbf{A}}$ with Equation 2.10. This matrix plays the role of normalizing the nodes' features considering the degrees of $\tilde{\mathbf{A}}$ and is indicated in the original GCN paper (KIPF; WELLING, 2017) as a pre-processing step.

$$\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}} \tag{2.10}$$

The central idea here is that low-degree nodes have more impact on their neighbors than high-degree nodes, as these high-degree nodes scatter their influence at many nodes. The matrix multiplication $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}$ scales by row, and its result multiplied by another $\tilde{\mathbf{D}}^{-\frac{1}{2}}$ scales by column. $\tilde{\mathbf{D}}^{-\frac{1}{2}}$ is used instead of simply $\tilde{\mathbf{D}}^{-1}$ because the normalization process happens twice, so $\sqrt{\tilde{\mathbf{D}}_{(i,i)}\tilde{\mathbf{D}}_{(j,j)}}$ is used for rebalancing purposes.

While $\hat{\mathbf{A}}$ is an $N \times N$ matrix, given that we have N nodes, we also have to define a matrix (or collection of vectors) for the nodes' features. For this purpose, we define $\mathbf{X} \in \mathbb{R}^{N \times F}$ as the features matrix, in which F is the number of features. With the matrix multiplication $\hat{\mathbf{A}}\mathbf{X}$ we achieve a balanced scaled features vector for each node, which is the average of all the neighbors' feature vectors (including itself). The resulting matrix $\in \mathbb{R}^{N \times F}$ represents each node's scale-averaged features vector on each row of the matrix.

We define an architecture of L stacked GCN layers by Equation 2.11, assuming $\mathbf{W}^{(l)}$ as the layer l's matrix of learnable weights, σ as an activation function, $\mathbf{H}^{(l)}$ as the matrix of hidden features of the lth layer, and $\mathbf{H}^{(0)} = \mathbf{X}$. As more GCN layers are added, the nodes being classified receive influencing signals (features) from higher-order neighborhoods. For example, a 2-layer GCN architecture classifies node v_i considering the features of its neighbors and the neighbors of its neighbors. Equation 2.12 exemplifies a possible 2-layer GCN architecture feed forward activation calculation, using softmax (WANG et al., 2018) and LeakyReLU (MAAS et al., 2013) as activation functions.

$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right) \tag{2.11}$$

$$f(\mathbf{X}, \mathbf{A}) = softmax\left(\hat{\mathbf{A}}LeakyReLU(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(0)})\mathbf{W}^{(1)}\right) \tag{2.12}$$

2.6 Graph Attention Networks

Proposed by Veličković et al. (2018), the Graph Attention Network (GAT) architecture implements a self-attention mechanism to learn which neighbors will be the most influential in a node's classification. The original paper specifically applies a single-layer feed-forward network as its attention mechanism, and a multi-head attention procedure to stabilize the learning process (VASWANI et al., 2017).

The GAT layer transforms a set of node features **h** $= \{\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_N}\}, \overrightarrow{h_i} \in \mathbb{R}^F$ into a set of node features **h'** $= \{\overrightarrow{h'_1}, \overrightarrow{h'_2}, ..., \overrightarrow{h'_N}\}, \overrightarrow{h'_i} \in \mathbb{R}^{F'}$, where N is the number of nodes, F is the number of features for each node and F' is the number of features for the output node features. In case this F' is representing the output of an ANN model for, let's say a four-classes classification problem, this F' could be exactly four (the model's attributed probability for the node being each one of the problem classes).

For learning how to make these transformations, the GAT layer utilizes a learnable weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$ and a learnable weight vector $\overrightarrow{\mathbf{a}} \in \mathbb{R}^{2F'}$, which is used in the defined attention mechanism with an additional non-linearity achieved by the LeakyReLU function application. Considering $||$ as the concatenation operation and $\mathcal{N}_i$ as the set of all negihbor nodes of i, the following equation is utilized for computing the attention coefficient α_{ij} of a neighbor node j over a particular node i (including i itself):

$$\alpha_{ij} = \frac{exp\left(LeakyReLU\left(\overrightarrow{\mathbf{a}}^T[\mathbf{W}\overrightarrow{h_i}||\mathbf{W}\overrightarrow{h_j}]\right)\right)}{\sum_{k \in \mathcal{N}_i} exp\left(LeakyReLU\left(\overrightarrow{\mathbf{a}}^T[\mathbf{W}\overrightarrow{h_i}||\mathbf{W}\overrightarrow{h_k}]\right)\right)} \tag{2.13}$$

The computed α_{ij} coefficients are used to obtain the $\overrightarrow{h'_i}$ classification output. To do that, another non-linearity σ is applied to the average result of the K multi-head attention learned outputs, which is the summation of the j nodes' classification contributions as defined by the innermost summation in Equation 2.14. Considering K multi-head attention executions, α^k_{ij} as the kth learned α_{ij} and $\mathbf{W}^k$ as the kth learned weight matrix, the following equation formalizes the GAT layer output for node i:

$$\overrightarrow{h'_i} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j \in \mathcal{N}_i} \alpha^k_{ij}\mathbf{W}^k\overrightarrow{h_j}\right) \tag{2.14}$$

Figure 2.4 extracted from the original GAT paper (VELIČKOVIĆ et al., 2018) illustrates how the algorithm operates.

Figure 2.4 – GAT illustration. On the left, the $LeakyReLU$ is applied over the learned weights and attention mechanism linear combination with the nodes' features. On the right, the multi-head attention process with $K = 3$ considering six neighbor nodes for the classification of node 1.

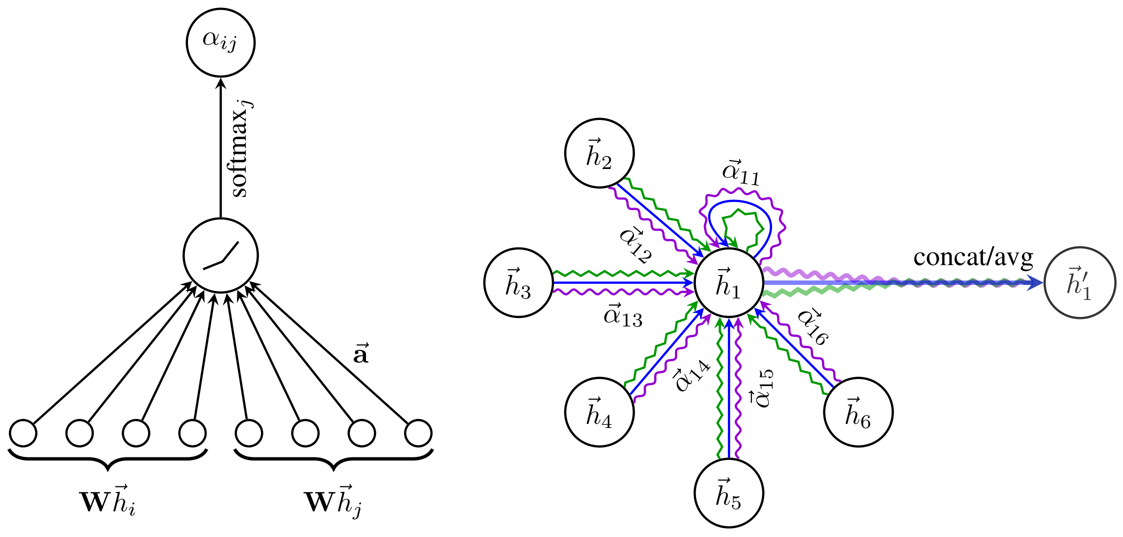


Image from Veličković et al. (2018).

2.7 Focal loss

The focal loss (FL) cost function was originally tested in the context of object detection (LIN et al., 2017), but its proposal contemplates ANNs in general, with a focus on unbalanced classification problems. In these types of problems, traditional loss functions can bias the machine learning models toward the correct classification of the majority class, as their successful classification have a larger impact on the loss function output reduction. Because there is a significant class imbalance in our data (Section 4.1), it is imperative to adopt strategies such as the application of the FL as a cost function for the classification using deep learning models.

The original FL is defined for binary classification, having the cross-entropy function as its basis. As Lin et al. (2017) argue, simply weighting the cross-entropy with an α parameter does balance the importance of positive/negative examples, but does not take into account the notion of hard/easy to classify examples. To solve this issue, the authors propose a modulating factor $(1 - p_t)^\gamma$ that down-weights easy examples. Thus, the α-weighted FL for binary classification is defined as shown by the equation below.

$$\mathrm{FL}(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \tag{2.15}$$

Where p_t is the model's estimated probability p for the positive class if the observed class is actually positive, and $1 - p$ if the observed class is negative. Analogously, α_t is a weighting parameter that establishes a weight for the positive class. Additionally, while γ is a tunable parameter, Lin et al. (2017) found that $\gamma = 2$ worked best for their experiments, so in this work we follow their recommendation, submitting the models to other two possible choices for γ (Section 4.3).

If, instead of binary, the classification problem is multi-class, α, γ and p are transformed into vectors of length m, and the FL can be straightforwardly defined by simply adding the $(1 - p_t)^\gamma$ parameter to Equation 2.16 (CHANG et al., 2018), as defined below. Consider m as the number of classes in the problem and y as a one-hot encoded vector for the true class of the sample corresponding to the model's outputted $\overrightarrow{p}$.

$$\text{FL}(\overrightarrow{p}) = \sum_{i=1}^{m} y_i \alpha_i (1 - p_i)^{\gamma_i} log(p_i) \tag{2.16}$$

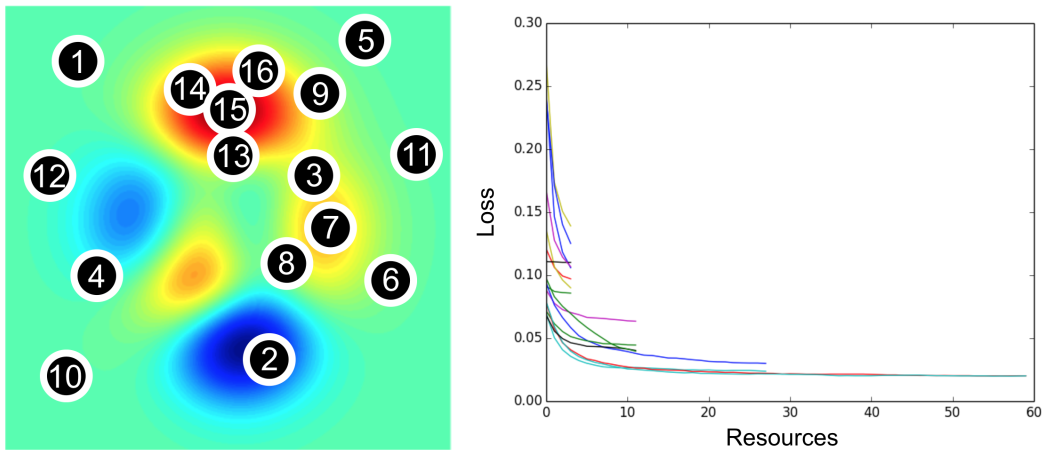
2.8 Hyperparameter optimization with the hyperband algorithm

ANNs have a lot of configurable hyperparameters, including the number of layers, units (or neurons) in each layer, learning rate, loss function, optimizer algorithm, dropout rate, among others. Choosing the exact optimal network setup can be a very challenging task, demanding lots of experiments and error-prone empirical analyzes. To properly deal with such a challenging task, several algorithms for hyperparameter optimization/tuning were proposed (YANG; SHAMI, 2020). The goal of these algorithms is to efficiently search in the hyperparameters' choices space for the optimal configuration. In this study, we utilize the hyperband optimization algorithm due to its computational efficiency when executed in parallel, and its good performance compared to other traditional hyperparameter optimization methods (LI et al., 2017).

The hyperband algorithm extends the Successive Halving (JAMIESON; TALWALKAR, 2016), utilizing it as a subroutine. Given a budget B and a maximum number of hyperparameters configurations n, the Successive Halving uniformly allocates the budgets for each participating configuration on each round of the tournament and takes only the half best ones to the next round, until only one remains. For example, considering $B = 300$ epochs and $n = 8$, there would be three rounds r before deciding the best configuration, each one with 8, 4, and 2 configurations respectively. On each round, B/r

Figure 2.5 – Visual comparison of configuration selection and configuration evaluation hyperparameter tuning methods.

(a) Configuration selection (b) Configuration evaluation



Images from Li et al. (2017).

epochs (100) are uniformly distributed amongst the configurations. For example, in the second round, each configuration has 25 epochs.

The hyperband algorithm defines a reduction factor (provided as input) instead of halving the configurations on each round, and it works with a maximum number of resource allocation R that can be allocated to a single configuration, instead of a maximum budget, B. With this value, it randomly samples the budget allocation for each configuration, instead of uniformly distributing them. Li et al. (2017) shows that such a process allows the exploration of different convergence behaviors. The authors also report a $5\times$ to $30\times$ faster execution of the hyperband optimization compared to other popular Bayesian optimization algorithms (WU et al., 2019).

Figure 2.5 shows a visual comparison of methods based on configuration selection, like Bayesian optimization, against configuration evaluation methods like the hyperband algorithm. In the configuration selection scenario, a sequential process is applied. This process is represented by Figure 2.5a, in which the heatmap shows a 2D search space with lower error rates assigned to warmer colors, and the numbers represent the sequence of configuration explorations. The configuration evaluation technique, on the other hand, is adaptative in computation, allocating more resources to more promising hyperparameter configurations. Figure 2.5b illustrates this process, where the lines represent different configurations plotted for an evaluation metric (loss) against resources allocation.

3 RELATED WORKS

As discussed in Section 2.1, there are a number of different omics data available, which characterize the organism under investigation from different biological perspectives. Although the gene expression is the most frequently analyzed one (MATURANA et al., 2019), the integration and joint analysis of the different types of omics has been also subject of investigation by the scientific community (HUANG; CHAUDHARY; GARMIRE, 2017; MANZONI et al., 2018), which coined the term multi-omics. As reviewed by Menyhárt and Győrffy (2021), besides multivariate methods like the non-negative matrix factorization (ZHANG et al., 2012), statistical methods such as iCluster (SHEN; OLSHEN; LADANYI, 2009) and iClusterBayes (MO et al., 2018), and network-based methods like iOmics (KOH et al., 2019), similarity-based methods for data integration, like the SNF algorithm reviewed in Section 2.2, have also been studied.

In fact, networks naturally occur in several complex biological investigation settings (MUZIO; O'BRAY; BORGWARDT, 2021; YI et al., 2022). With the increasing reports of GNNs' success for graph representation learning, such algorithms have also conquered a substantial space for biology-related tasks that models their data using graphs, demonstrating promising results. These algorithms have been used for molecule analysis in tasks like molecule representation learning (LI et al., 2021), molecule properties prediction (WIEDER et al., 2020), molecule graph generation (MAHMOOD et al., 2021), and drug-target prediction (MANOOCHEHRI; PILLAI; NOURANI, 2019). Another popular biology-related use for GNNs is in the healthcare field, in which the applications vary from the classification of Alzheimer's disease based on brain resonance imaging (SONG et al., 2019) to prediction tasks on electronic health records (CHOI et al., 2020).

Network data modeling strategies with graph representation learning using GNNs are also present in the multi-omics context. The most straightforward application of such techniques is in the realms of protein-protein interaction networks – for example, for link prediction (LIU et al., 2019) and protein function prediction (YOU et al., 2021). There are also some works investigating genomics graph analysis, like in the study by Han et al. (2019) in which a GCN combined with matrix factorization method was employed for discovering gene-disease associations, and in Rhee, Seo and Kim (2017), where the authors apply a GCN for the breast cancer subtyping task, using gene expression data combined with protein-protein interaction networks.

As can be seen, GNNs and other forms of graph analysis permeate several biologi-

Table 3.1 – Summary of the considered most similar works to ours. The PSN column indicates whether or not the study utilizes patient/sample similarity networks.

Work	Multi-omics data	Cancer staging	GNN	PSN
Yu et al. (2020)	no	yes	no	no
Pai et al. (2019)	yes	no	no	yes
Lu et al. (2022)	no	no	no	yes
Lu and Uddin (2021)	no	no	yes	yes
Zhang et al. (2022)	yes	no	yes	yes
Baul et al. (2022)	no	no	yes	yes

cal investigations and have been helping to advance a myriad of research problems within this field. Because of that, in the remaining text of this chapter, we focus on discussing with greater detail only the works that we consider most similar to ours (summarized by Table 3.1), reasoning about their strengths and limitations, which may pose promising research opportunities and investigation gaps to cover. An exception is the work by Yu et al. (2020), which, despite not using samples as nodes for their base graphs subject to graph analysis, is one of the most successful methods for cancer staging we could find in the literature (considering the number staging system[1] for the classes of the problem).

The method proposed by Yu et al. (2020) firstly selects only the differentially expressed genes, then divides the samples of each cancer stage into 30% for reference and the remaining for training and testing. With the reference samples, the method computes a co-expression gene network for each cancer stage, in which genes are nodes and two genes were linked if their Pearson correlation coefficient in terms of expression was above 0.7. For each sample in the train/test set, the method computes a perturbed co-expression network for each cancer stage and then extracts from the network a vector subject to classification, comparing the performance of six ML algorithms: Naive Bayes, treebag, C5.0, random forests, random ferns, and weighted subspace random forests. With exception of the Naive Bayes and the random ferns, all the classifiers achieved similar very impressive performances. The C5.0, for instance, reached an accuracy of 0.9504 for the colon adenocarcinoma (COAD) dataset, 0.9452 for the kidney renal clear cell carcinoma (KIRC) dataset, and 0.7933 for the lung adenocarcinoma (LUAD) dataset. Despite these very impressive results, the authors used only gene expression data, no GNN was applied for the graph classification task and we could not identify any reasoning behind the chosen correlation threshold. It would also be interesting to investigate closer the influence of the number of samples selected for computing the base co-expression network, as more

[1]https://www.nhs.uk/common-health-questions/operations-tests-and-procedures/what-do-cancer-stages-and-grades-mean

samples would result in a network more robust to perturbation from a single sample, and fewer samples, in a more sensible network, thus, greatly influencing in the input data subject to classification.

While Yu et al. (2020) approach shows very promising results, our investigation focus lies in sample (patient) similarity networks. According to (PAI; BADER, 2018), these networks have the advantage of easily integrating heterogeneous data, and naturally handling missing information. Authors report only two methods exploring this type of network modeling approach by the time they published their review: the SNF (already covered in Section 2.2) and the netDx (PAI et al., 2019). Based on selected genes and a similarity metric, such as Pearson correlation or euclidean distance, patient networks are built by netDx using a sparsification technique, which includes three parameters: the minimum edge weight to include, how many top interactions to include per node (*topX*), and the upper bound on the number of edges in a network (*maxE*). The authors report different optimal values for the different investigated cancer datasets, where the *topX* varied from 30 to 50, and the *maxE* from 3,000 to 6,000. Label propagation is then applied to classify the samples. The authors benchmarked their method against classical ML methods for the cancer survivability task and found that netDx outperformed the other algorithms.

Lu and Uddin (2021) compared the usage of GCN and GAT for predicting chronic diseases – cardiovascular disease (CVD) and chronic pulmonary disease (CPD) – from administrative claim data. They propose to model the data as a weighted patient network that can capture patients' latent relationships. Two patients are connected in the network if they were diagnosed with one or more diseases in common. The weights in the edges represent how many shared diagnosed diseases the patients had. As node features, the authors use age, gender, and smoking history. The authors also compare the GNNs with classical ML algorithms, whose input data corresponds to the node features and network engineered features based on previous study (LU et al., 2022).The GNN models largely outperformed the classical ML algorithms. For CVD, GAT achieved 93.49% of accuracy while the GCN was a little worse with 90.04% of accuracy. This was also observed for the CPD classification task (89.15% for GAT, and 87.26% for GCN). Lu et al. (2022) investigated a very similar approach for predicting type 2 diabetes mellitus, but they did not apply any GNN for the classification task.

Zhang et al. (2022) proposed a DeepGCN (LI et al., 2019) for diagnosing liver cancer using multi-omics data – RNA-Seq, DNA methylation and copy number variation (CNV). The network was built using the SNF method, and a denoising autoencoder (DAE) (VINCENT et al., 2008) was applied for learning features from the samples that were used as node features. The authors successfully demonstrated that using all the three selected omics data as opposed to just one or a combination of two was the most effective setting. Additionally, they performed some experiments demonstrating that the presence of the SNF network was indeed vital to the model's performance, showing up to 12.2% of gains in accuracy. Their method, pDeepGCN, was the best compared to other state-of-the-art and classic machine learning algorithms, with an accuracy of 98.57%, 1.31% better than the second best method, XGBoost-AD (MA et al., 2020). Although the authors found impressive classification results, we note that the binary classification task for cancer disease is much simpler than the multi-class cancer staging. Additionally, only liver cancer was explored in their work, and a traditional MLP was not used to compare if such a simpler deep learning algorithm would have been more (or comparably) successful. We also could not find any mention of a sparsification procedure in the resulting SNF adjacency matrix, neither the authors executed their experiments with sufficient repetitions, enabling a statistical comparison.

Baul et al. (2022) proposed the omicsGAT method for cancer subtype prediction and cancer patient stratification using RNA-Seq data. They apply a GAT algorithm for learning hidden features from a sample similarity network (built using correlation scores between the samples) and a three-layers MLP for classifying the hidden features. For the classification task, the authors compare their method with classical ML algorithms, as well as an MLP and a GCN. Although the task was tumor subtyping, we do not consider it comparable to the multi-class cancer staging we are dealing with, since they separate it into two tasks with only positive or negative possible outcomes. Their method was the best among the tested ones, but no statistically significant differences were found for the estrogen receptor classification task when compared to the SVM and random forest algorithms. Once again, we could not find any indication of a sparsification procedure in the manuscript. The correlation-based sample similarities would produce a dense adjacency matrix, but, by inspecting their source code[2], a binary sparse adjacency matrix is expected as input for their method.

[2]https://github.com/compbiolabucf/omicsGAT

4 METHODOLOGY

For assessing if sample similarity networks can be used to improve the classification of tabular data, particularly for high-quality cancer staging, we defined three strategies for building these networks based on our literature review (see Section 3). To take advantage of this relational data structure, we utilized two recently proposed and already successful algorithms for node label prediction: the GCN and GAT graph neural network architectures. Additionally, as a graph analysis baseline algorithm, we used the Spectral Clustering directly into the SNF matrix, as proposed by Wang et al. (2014). On the other hand, as a baseline strategy to compare with the sample similarity network ones, we used a traditional MLP directly with the tabular data.

4.1 Analyzed data

The target datasets used by the employed methods for cancer staging were acquired from and pre-processed by Duan et al. (2021). This pre-processed data was originally provided freely and publicly by The Cancer Genome Atlas Program (TCGA), a popular program that has already characterized over 20000 cancer and matched normal samples across 33 types of cancer[1]. Particularly, we selected the omics datasets containing only the statistically significant features as evaluated by the authors that processed and published these datasets[2].

Among the provided types of omics data, we selected transcriptomics, miRNomics, and methylomics as they are the most informative and used omics for related classification problems (DUAN et al., 2021). While the authors analyzed nine types of cancer, we selected only the three types with the larger amount of samples of the minority class, which turned out to be the COAD (colon adenocarcinoma), KIRC (kidney renal clear cell carcinoma) and LUAD (lung adenocarcinoma) cohorts.

For the acquisition of the cancer stage information of each sample, we collected clinical data from FireBrowse (CENTER, 2016), which is an initiative that compiles and facilitates the access of TCGA data, in addition to offering them in a pre-processed format that simplifies analyses. By using the FireBrowse data with the multi-omics datasets, it was possible to build a traditional multi-class classification dataset for each of the selected

[1]https://cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga
[2]https://github.com/GaoLabXDU/MultiOmicsIntegrationStudy

Table 4.1 – Datasets summarization

Cancer Type	# Stage 1	# Stage 2	# Stage 3	# Stage 4	# Features
COAD	44	112	85	41	4200
KIRC	152	31	72	58	4200
LUAD	244	109	73	19	4200

Figure 4.1 – TCGA barcode structure.

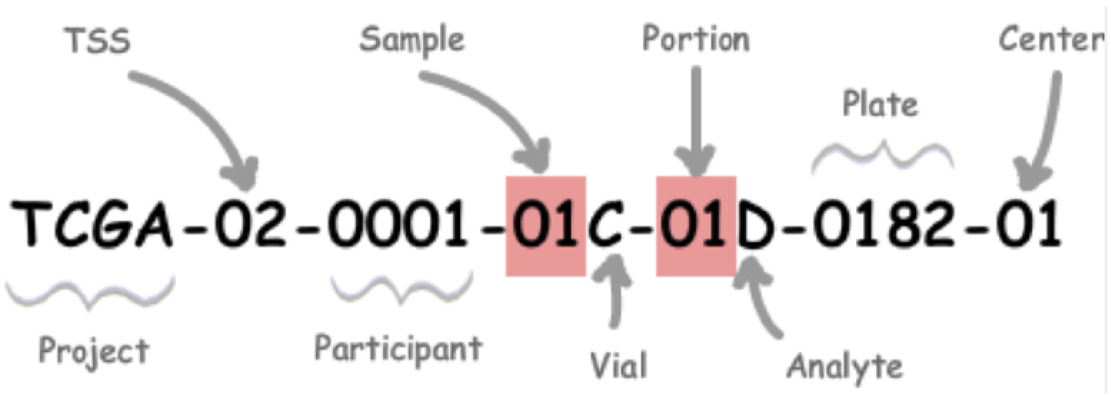


Image from GDC's documentation[3].

cancer types. The summarization of these datasets is provided in Table 4.1. The 4200 features of each dataset correspond to 2000 features from mRNASeq data, 200 features from miRNASeq data, and 2000 features from DNA methylation data.

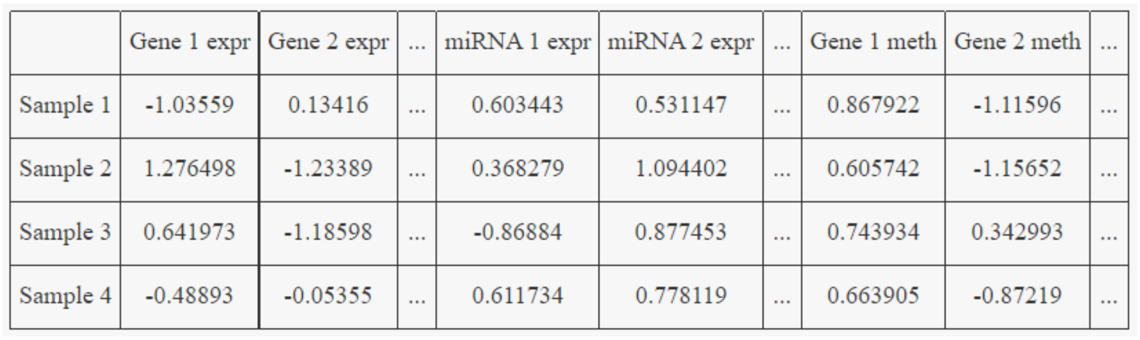
To generate the class information for the samples inside the multi-omics datasets, their index was used. Each sample is named with a TCGA barcode that contains basic information as shown by Figure 4.1.

The samples in the multi-omics datasets did not have the complete barcode, they included information about the *Project* (only TCGA samples), *TSS* (tissue source site), *Participant* (the subject from whom the samples were collected) and the *Sample* (the type of the sample that was collected). Because we are interested in the cancer staging problem, we added a data processing step for eliminating all possible tissue samples that were labeled as control or tumor-adjacent (normal, non-cancerous samples). For doing that, we followed the Genomic Data Commons (GDC) documentation[3] and isolated only the samples having a code number ranging from *01* to *09* in the *Sample* TCGA barcode field.

By making sure we were dealing only with tumor-tissue samples, we could properly integrate the resulting data with the clinical information dataset downloaded from FireBrowse. This enabled the generation of the class dataset, which was a simple look-up

[3]https://docs.gdc.cancer.gov/Encyclopedia/pages/TCGA_Barcode/

Figure 4.2 – Illustration example of a final processed dataset of features. Columns "Gene *X* expr" and "miRNA *Y* expr" represent gene *X* and miRNA *Y* expression levels, respectively. Columns "Gene *X* meth" represent the methylation score of gene *X*.



	Gene 1 expr	Gene 2 expr	...	miRNA 1 expr	miRNA 2 expr	...	Gene 1 meth	Gene 2 meth	...
Sample 1	-1.03559	0.13416	...	0.603443	0.531147	...	0.867922	-1.11596	...
Sample 2	1.276498	-1.23389	...	0.368279	1.094402	...	0.605742	-1.15652	...
Sample 3	0.641973	-1.18598	...	-0.86884	0.877453	...	0.743934	0.342993	...
Sample 4	-0.48893	-0.05355	...	0.611734	0.778119	...	0.663905	-0.87219	...

table that linked the barcodes with their respective class (one of the four possible cancer stages). Subtypes of specific stages (*e.g* "Stage 3 A", "Stage 1 II", *etc*) were agglutinated in one type ("Stage 3", "Stage 1", *etc*) and "Stage X" samples were discarded as they are uncharacterized tumor samples regarding the adopted cancer staging system.

Each sample – a row in the tabular data for the MLP algorithm, and a node in the sample similarity networks for the graph-based algorithms – had the same set of features with no faulty values. The original downloaded data was composed of three separate datasets, one for each of the adopted omics. These datasets represented the features in the rows and the samples in the columns. Thus, a transposition step followed by an inner join-based concatenation process for the three was needed to produce a single tabular dataset representing the sample features. Figure 4.2 shows a partial example of a dataset of features, illustrating its final format.

4.2 Networks generation

Our main goal is to investigate if the multi-omics cancer staging task can benefit from the sample similarity network modeling. We take advantage of such non-euclidean relational structures by utilizing a class of newly developed and successful ML algorithms for this kind of data, graph neural networks. To properly address this research question, we investigate three sample similarity network generation strategies for our data:

- **CGEN**: correlation network based on gene expression levels
- **CMON**: correlation network based on multi-omics data
- **SNF**: Similarity Network Fusion

The three strategies produce a dense adjacency matrix with sample similarity values for all pairs of samples. Thus, in order to produce a sparse adjacency matrix, we apply thresholds in the sample similarities, zeroing all the values that were below the defined thresholds. Additionally, the diagonals are filled with ones, producing an auto-connection for every node, since, without this auto-connection, the GNN models would not be able to consider the features of the node subject to classification.

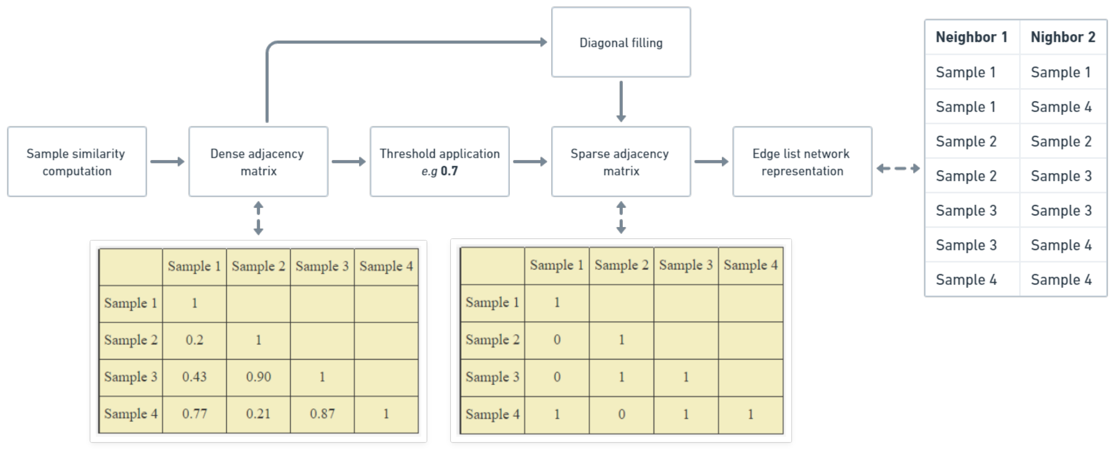
To avoid applying arbitrary thresholds to sample similarity values generated from different strategies, we chose to work with percentiles, particularly $\{1\%, 5\%, 10\%, 25\%, 50\%, 75\%, 90\%, 95\%, 99\%\}$. These percentile-oriented thresholds are related to the top biggest values for the sample-sample connections. A percentile of 75% indicates that only the top 25% most similar sample-sample connections would be kept. Thus, the greater the percentile, the more strict the network is with its connections, meaning that this network would have fewer connections. We acknowledge that such a strategy has its limitations, since we are enforcing the network to have at least a specified number of connections, and there could be a case in which a very sparse network would be a more proper representation of the similarity relations between the data samples. This way, an automatic and more sophisticated method for selecting the appropriate threshold should be used. Nonetheless, we argue that such a challenge is a research problem of its own and beyond the scope of this work, leaving its investigation for future research efforts.

Since miRNA expression and DNA methylation act directly and indirectly in the regulation of gene expression levels characterized by the RNASeq data, this data can be considered as a more immediate representation of the relation between the collected samples and the state of the organisms under investigation. Considering this, the CGEN strategy calculates the pairwise Pearson correlation coefficients, ***r***, of only the samples' gene expression levels, ignoring the other features. The ***r*** coefficient is then used as a measure of similarity for the samples.

On the other hand, the CMON strategy considers the three omics as indispensable information for describing the samples' similarity. Thus, it calculates the correlations separately for each omic, which left all pairwise samples with three correlation values. CMON considers as the samples' similarity the biggest ***r*** value among the three computed ones. For instance, if sample A and sample B have $r = 0.3$ considering the mRNASeq data, $r = 0.8$ considering the miRSeq data, and $r = 0.01$ considering the methylation data, the actually quantified similarity between A and B according to CMON is 0.8.

Finally, the SNF strategy also considers all the three omics, but, unlike CMON, it

Figure 4.3 – Illustration example of how the sample similarity networks are generated given a similarity computation strategy. If the produced dense adjacency matrix does not have its diagonal filled with ones, an extra step for this procedure is performed.


Diagonal filling
Sample similarity computation
Dense adjacency matrix
Threshold application e.g 0.7
Sparse adjacency matrix
Edge list network representation

	Sample 1	Sample 2	Sample 3	Sample 4
Sample 1	1			
Sample 2	0.2	1		
Sample 3	0.43	0.90	1	
Sample 4	0.77	0.21	0.87	1

	Sample 1	Sample 2	Sample 3	Sample 4
Sample 1	1			
Sample 2	0	1		
Sample 3	0	1	1	
Sample 4	1	0	1	1

Neighbor 1	Nighbor 2
Sample 1	Sample 1
Sample 1	Sample 4
Sample 2	Sample 2
Sample 2	Sample 3
Sample 3	Sample 3
Sample 3	Sample 4
Sample 4	Sample 4

fuses the three adjacency matrices into one, as explained in Section 2.2. The final connections are drawn by the same threshold logic followed by CMON and CGEN. Figure 4.3 illustrates how this process works, abstracting the way the similarity matrix is calculated, which depends on the adopted strategy. Note that the final network representation is in the format of an edge list. However, this is an irrelevant implementation decision that does not affect the experiments themselves.

4.3 Algorithms and metrics

To investigate the potential of the sample similarity network modeling, we apply two top-performative GNN algorithms: Graph Attention Networks and Graph Convolutional Networks. We work with StellarGraph (DATA61, 2018) implementation of both and compare with an MLP, whose implementation choice was Keras (CHOLLET et al., 2015), and the Scikit-learn's Spectral Clustering unsupervised learning algorithm (PEDREGOSA et al., 2011). The ANN algorithms were also subject to hyperparameter tuning.

The hyperparameter tuning process was executed for tuning the ANN models in the first experiments' repetition using only the training and validation data (the data split is detailed in Section 4.4). After that, the tuned hyperparameters were fixed and used in the remaining experiments' repetitions. Because the set of possible choices for hyperparameters can increase exponentially, we tried to select the most relevant ones that were hard to intuitively be set for our problem; could produce a lot of impact on the performance; and, whenever possible, were shared across the different ANN architectures. We also considered those that could generate interesting discussions depending on the tuning results.

Thus, the selected hyperparameters and their respective possible choices were the following:

- Number of hidden layers: {1, 2, 3}
- Number of neurons/units on each layer: {32, 64, 128}
- Focal loss γ: {0, 1, 2}
- Dropout: {0, 0.1, 0.2, 0.3}
- Learning rate: {0.0001, 0.0005, 0.001, 0.005}

The only hyperparameter selected for tuning that was not shared across the algorithms was the GAT's number of attention heads, which could be set by choosing from the following set of possible values: {2, 4, 8}. Additionally, as explained in Chapter 2, the number of hidden layers for the GNN architectures defines the number of hops in the neighborhood that will influence the classification of a given node.

As our work represents early efforts for the application of the sample similarity networks modeling with GNNs for node classification, we do not investigate thoroughly the classification performance from different perspectives, discussing particular difficult stages to classify, for example. Rather, we concentrate efforts on analyzing the multiple algorithms and problem-modeling setups, comparing each other, and reasoning about their strengths and limitations. Because our data is significantly unbalanced (see Table 4.1), the AUC-ROC and AUC-PR were adopted as evaluation metrics for the assessments. Particularly, the AUC-PR was selected as the guiding metric for the hyperparameter tuning optimization process. An exception to that is the SC algorithm, which is unable to provide probabilities for applying a classification threshold in its standard implementation (see Section 2.3). In this case, the greatest accuracy obtained from each possible cluster-class assignment was collected and used as its evaluation metric. Hence, for comparing the other classification algorithms with SC, the accuracy was also collected.

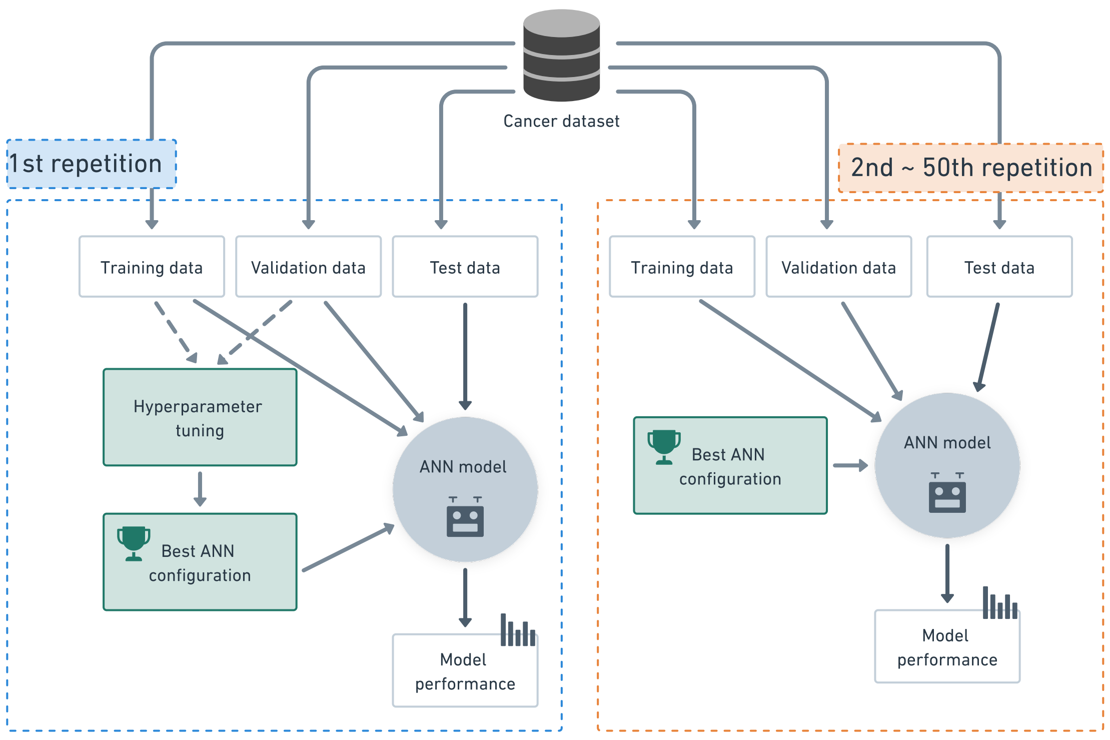
4.4 Models Evaluation

For each cancer dataset, each network generation strategy with a particular threshold application represented a single GAT or GCN experiment. These experiments were executed 50 times each with 50 different stratified sets of training (80% of samples), validation (10% of samples), and test (10% of samples) data. The MLP experiment was based on a single 50-repetitions experiment for each cancer dataset, as the network modeling strategies and the varying thresholds did not affect anything in the data used for training and testing the MLP. In addition, for ensuring a fair comparison among the ANN algorithms, a seed was used at the beginning of each experiment, which guarantees the same train-validation-test split for each one of the 50-repetitions experiments.

On the other hand, the SC algorithm experiment consisted of 50 executions using all the data for each dataset considering its SNF affinity matrix, as proposed by (WANG et al., 2014). The lack of train/validation/test split in the SC experiments is due to its unsupervised learning nature, which can enable the usage of all the data samples. Besides, as explained in Section 4.3, no AUC metrics were collected for this algorithm and the metric used for comparing the performances was the highest accuracy considering all possible clustering-class association combinations possible. This approach could also jeopardize the cancer staging application in real-life scenarios since it would demand the correct classification of the clusters as an additional human analysis intervention process.

The hyperparameter tuning was executed only in the first repetition of each experiment of the ANNs, using the validation data for evaluating the hyperparameters choices in the tuning optimization process. However, the validation data was also utilized in the remaining repetitions for the early stopping strategy, which intends to avoid models' overfitting during the training. The early stopping patience was set to 10 for the hyperparameter tuning process and 50 for the training of the selected model configuration, both with a maximum of 300 epochs. Figure 4.4 illustrates what is the execution flow of a single ANN experiment with its 50 repetitions.

Figure 4.4 – Illustration of a single ANN experiment process. The experiment has 50 repetitions, choosing different sets of training, validation, and test data at each one. In the first repetition, the training and validation data are also used by a hyperparameter tuner, which defines the best ANN configuration to be used throughout all the experiment's repetitions.



5 RESULTS

As presented in Chapter 1, among the investigation questions our study aims to address, we want mainly to understand if the network modeling approach improves the standard tabular one for the cancer staging multi-class classification problem, and how well GNNs perform with such data. An MLP model learning with the tabular data was used as a baseline for evaluating if the network modeling approach improved its performance. As a graph baseline, we chose the SC algorithm explained in earlier chapters.

The following sections present our findings, firstly discussing the varying thresholds for each sample network generation strategy. The best threshold for each strategy is then used for comparing the strategies against each other and the baselines. We proceed the analysis of the results with a discussion about the best hyperparameters found by the implemented tuning technique and if they changed throughout different network generation strategies and thresholds. Finally, we provide some insights into why the lowest thresholds produced the worst results, proposing informative network visualizations for enabling this discussion.

5.1 Thresholds

As discussed in Section 4.2, nine quantile-based thresholds were selected for investigating the performance of the GNN algorithms. Tables 5.1, 5.2 and 5.3 show the mean AUC-ROC across the 50 repetitions for each algorithm and each strategy executed on the KIRC, COAD and LUAD datasets, respectively.

From the first inspection of the results tables it is possible to see that, with some exceptions, the higher thresholds yielded better performances, especially for the GAT model. Perhaps the most notable exception, in this case, was GAT's performance for the LUAD dataset using CMON strategy and a threshold of 95%, in which the performance drop was substantially visible. These results indicate that more strict thresholds, *i.e.* networks with fewer connections, tend to work better. Figures 5.1, 5.2 and 5.3 reinforce that there is also a tendency to improve the AUC-PR performance of the networks as the threshold increases, although some performance fluctuations are also observed. The performance growth plots in the Appendix A.1 (Figures A.1–A.6) show a similar behavior for the other strategies. In Section 5.4, we discuss with greater detail a possible explanation for why this is happening. Additionally, the best results observed for each pair

Table 5.1 – AUC-ROC results (mean) in KIRC dataset for each network generation strategy and each GNN algorithm. Cells in grey highlight the best GNN result within the same strategy and threshold. Cells in green highlight the best result for each strategy.

KIRC		**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	GAT	0.52	0.47	0.51	0.56	0.62	0.62	0.63	0.63	0.69
	GCN	0.54	0.61	0.60	0.59	0.60	0.60	0.62	0.64	0.71
CMON	GAT	0.63	0.53	0.58	0.51	0.59	0.64	0.67	0.67	0.73
	GCN	0.68	0.67	0.64	0.67	0.66	0.70	0.69	0.66	0.71
SNF	GAT	0.51	0.50	0.45	0.61	0.62	0.56	0.68	0.68	0.71
	GCN	0.57	0.63	0.64	0.65	0.68	0.71	0.70	0.71	0.70

Table 5.2 – AUC-ROC results (mean) in COAD dataset for each network generation strategy and each GNN algorithm. Cells in grey highlight the best GNN result within the same strategy and threshold. Cells in green highlight the best result for each strategy.

COAD		**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	GAT	0.50	0.50	0.49	0.54	0.50	0.60	0.56	0.58	0.65
	GCN	0.52	0.51	0.55	0.56	0.56	0.59	0.64	0.60	0.62
CMON	GAT	0.50	0.51	0.55	0.50	0.52	0.52	0.55	0.53	0.65
	GCN	0.55	0.56	0.56	0.57	0.56	0.56	0.59	0.56	0.63
SNF	GAT	0.48	0.52	0.51	0.51	0.52	0.57	0.51	0.59	0.63
	GCN	0.44	0.50	0.56	0.53	0.55	0.53	0.54	0.61	0.63

dataset-strategy were, with only one exception (because of a tie), in the highest tested threshold: 99%; however, as we will discuss later, there is a caveat to this threshold.

By inspecting the tables, it is also possible to notice that GCN tends to surpass GAT across the different tested thresholds, especially in the lowest ones, which can indicate a more robust nature of the GCN model in terms of the network modeling quality. Nonetheless, in terms of the best performance for each dataset-strategy, GAT surpasses GCN in all strategies for LUAD, all strategies for COAD (just tying in SNF), and also when using CMON for KIRC, losing in CGEN and tying in SNF. The GCN algorithm shines in the KIRC dataset, in which it dominates GAT in most of the thresholds, in particular as the best strategy using CGEN.

Table 5.3 – AUC-ROC results (mean) in LUAD dataset for each network generation strategy and each GNN algorithm. Cells in grey highlight the best GNN result within the same strategy and threshold. Cells in green highlight the best result for each strategy.

LUAD		**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	GAT	0.50	0.52	0.51	0.58	0.54	0.53	0.51	0.63	0.70
	GCN	0.52	0.53	0.53	0.58	0.59	0.57	0.61	0.64	0.68
CMON	GAT	0.52	0.51	0.48	0.51	0.55	0.63	0.61	0.51	0.70
	GCN	0.55	0.57	0.55	0.55	0.55	0.55	0.60	0.61	0.68
SNF	GAT	0.52	0.50	0.52	0.55	0.58	0.56	0.55	0.61	0.70
	GCN	0.50	0.53	0.53	0.56	0.57	0.57	0.60	0.64	0.67

Figure 5.1 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (KIRC) using the correlation network based on multi-omics data (CMON) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC (b) AUC-PR

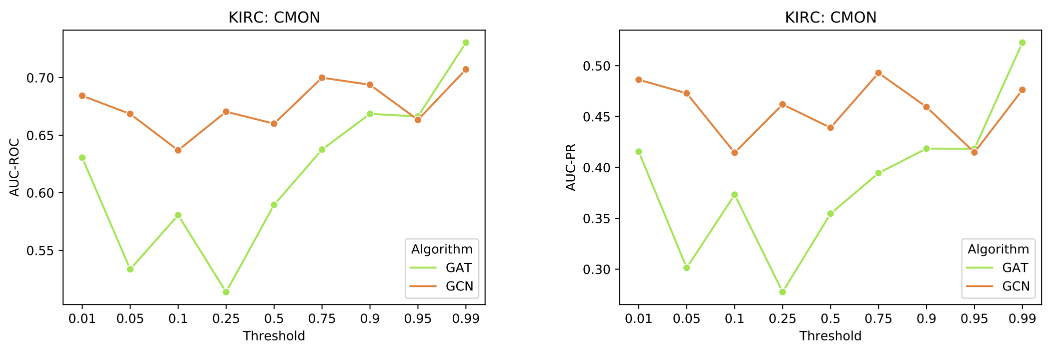


Figure 5.2 – Performance comparison of GAT and GCN across the investigated thresholds for the colon cancer data (COAD) using the correlation network based on multi-omics data (CMON) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC (b) AUC-PR

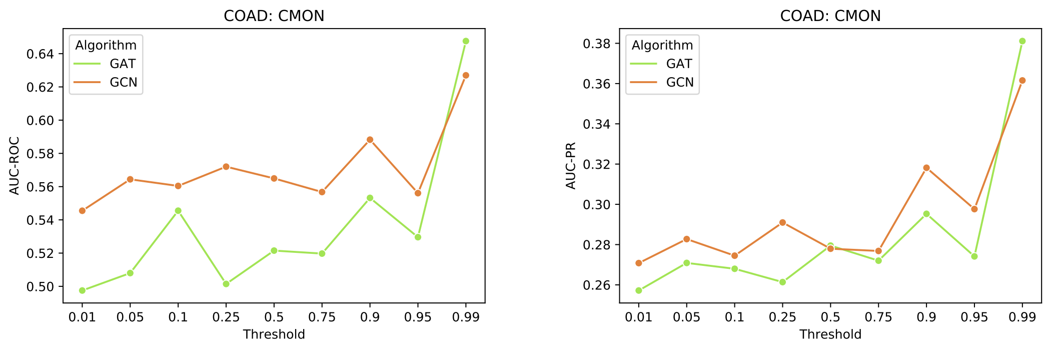
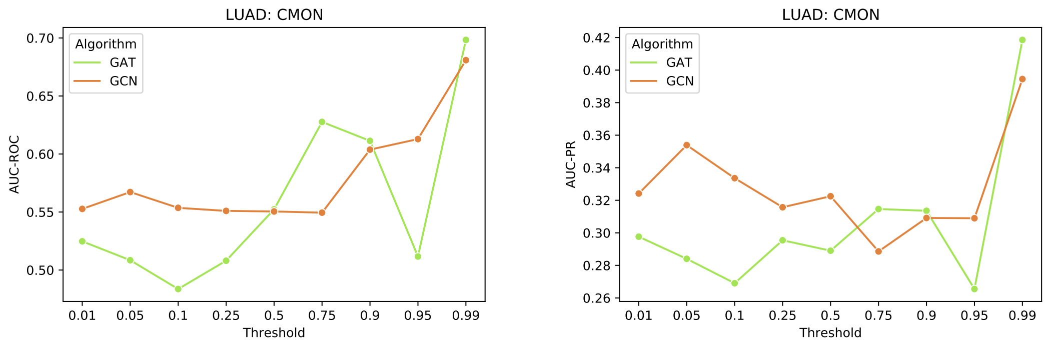


Figure 5.3 – Performance comparison of GAT and GCN across the investigated thresholds for the lung cancer data (LUAD) using the correlation network based on multi-omics data (CMON) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC (b) AUC-PR



While higher thresholds seemed to be linked with better performances, it still remains as an open question the searching for the best possible threshold for each dataset-strategy. Systematic and automatic approaches should be more closely investigated for preventing biases in the threshold choices.

As we will discuss in Section 5.4, the resulting networks from the application of the threshold 99% have a lot of nodes without connections, which can put into question whether such a threshold is appropriate to decide the best GNN model for the tested domain. In fact, if we disregard this threshold, the GCN model becomes the best GNN for all strategies in KIRC, all strategies in COAD, and two strategies in LUAD, which would completely invert the conclusion of the analysis.

Regarding the network generation strategies, CMON seems to work best when observing the best results (rounded with two decimal places) for each strategy, but again, when disregarding the last threshold this is not true anymore. So we consider that there is no strong indication that one is more appropriate than the others, which puts the adopted SNF method in advantage since it is the most efficient algorithm, at least when using the default euclidean distance metric. It is worth mentioning, though, that there are lots of other possible affinity equations to test, even correlation ones, so it remains as an open question how the other SNF parameters could further affect the observed results.

For clarity, Table 5.4 explicitly presents what setups were considered the best ones dataset-strategy-wise, considering AUC-ROC metric with more than two rounded decimal places. Additionally, it is convenient to elect a single best dataset setup for other analyzes where it would be too complex to discuss each setup. These best setups for each dataset are highlighted in green by the same table.

Table 5.4 – Best network generation strategies' setup for each dataset. The rows highlighted in green are considered the best setup for a particular dataset.

Dataset	**Setup**	**Algorithm**	**Threshold**
KIRC	CGEN-Best	GCN	99%
	CMON-Best	GAT	99%
	SNF-Best	GAT	99%
COAD	CGEN-Best	GAT	99%
	CMON-Best	GAT	99%
	SNF-Best	GCN	99%
LUAD	CGEN-Best	GAT	99%
	CMON-Best	GAT	99%
	SNF-Best	GAT	99%

5.2 Baselines comparison

The following comparison with the baselines considers only the best setup for each network generation strategy (threshold-algorithm). We explore some visualizations for analyzing the results of the three ANN algorithms and constructing a better understanding of their performances considering the fluctuations of the 50 different executions of each experiment. After that, we chose to compare quantitatively these three GNNs for each cancer dataset against the MLP and the SC (with SNF) approaches.

For a better understanding of how the GNN methods compare to the MLP and each other in each scenario, it is useful to compare their distributions side-to-side visually. Figures 5.4, 5.5 and 5.6 compare the AUC-ROC for the best threshold-network on each dataset as defined by Table 5.4. Since the MLP uses tabular data, its performance is independent of threshold choice and only changes for the different cancer datasets.

Analyzing Figure 5.4, it is possible to notice that the medians are really close to each other, with GCN having a slight disadvantage. In Figures 5.5 and 5.6, this median difference between MLP and GAT to GCN is more clearly visible. Additionally, the MLP for KIRC showed a more consistent AUC-ROC performance, centering the observations

Figure 5.4 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 99% threshold.

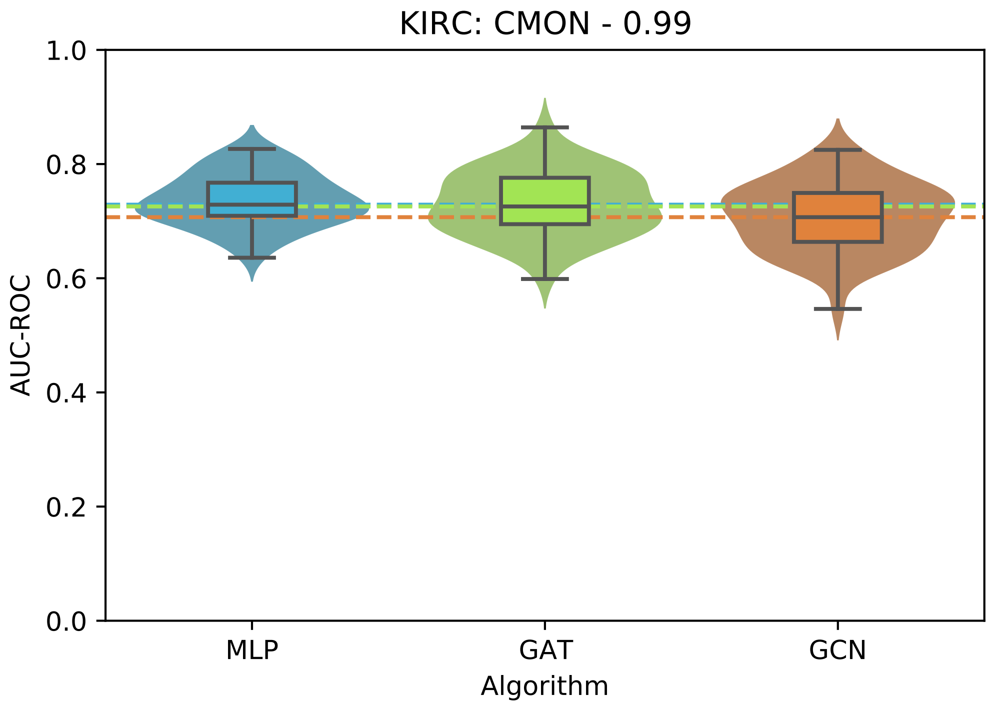


Figure 5.5 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 99% threshold.

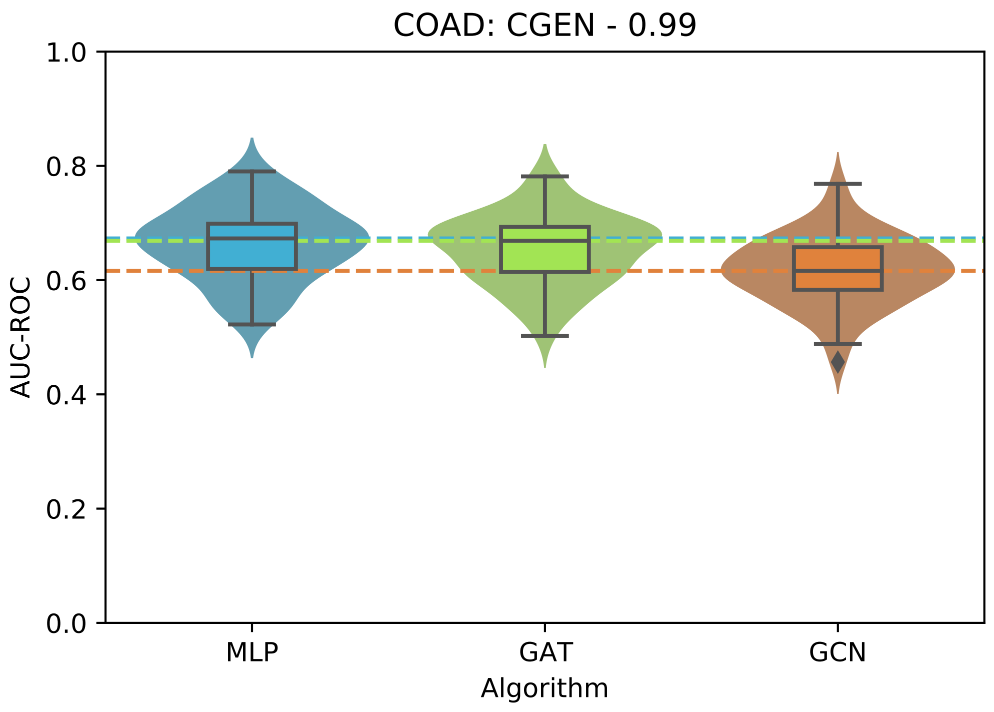


Figure 5.6 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 99% threshold.

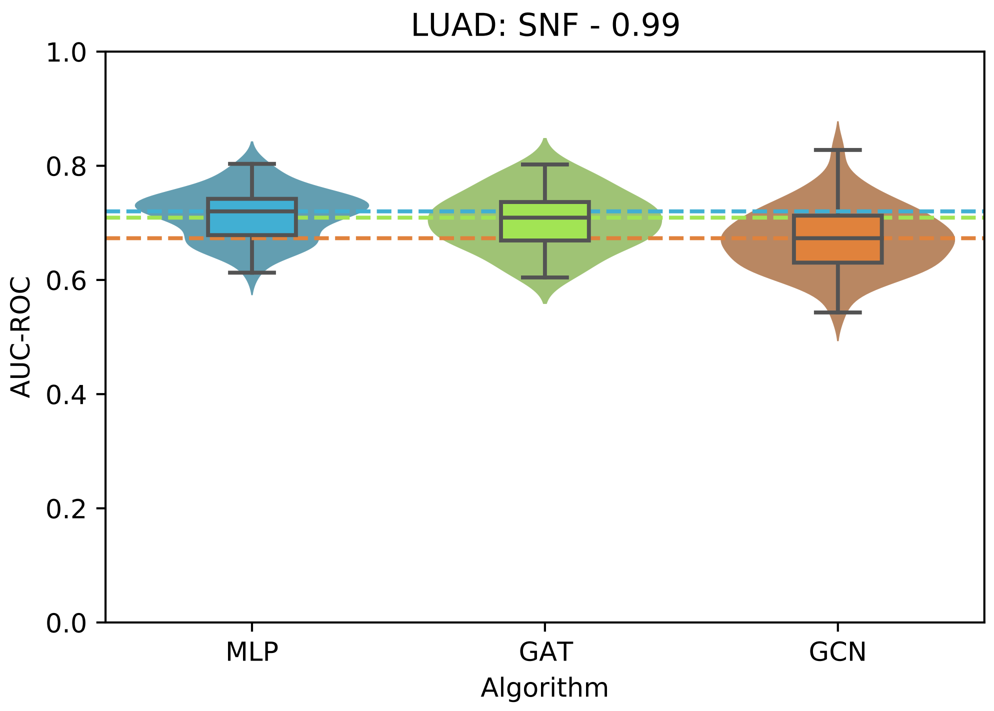


Table 5.5 – Performances comparison in KIRC dataset of the best setups for each network generation strategy and each baseline. Highlights in green indicate the best performance for each metric and results in bold indicate no statistically significant difference ($p < 0.05$) compared to the best performance, according to a Wilcoxon signed-rank test with Bonferroni correction.

KIRC	Accuracy	AUC-ROC	AUC-PR
SC	0.38 ± 0.00	-	-
MLP	0.51 ± 0.06	0.74 ± 0.05	0.54 ± 0.08
CGEN-Best	0.48 ± 0.06	0.71 ± 0.04	0.50 ± 0.07
CMON-Best	**0.51 ± 0.08**	**0.73 ± 0.06**	**0.52 ± 0.08**
SNF-Best	**0.48 ± 0.07**	0.71 ± 0.05	0.49 ± 0.07

Table 5.6 – Performances comparison in COAD dataset of the best setups for each network generation strategy and each baseline. Highlights in green indicate the best performance for each metric and results in bold indicate no statistically significant difference ($p < 0.05$) compared to the best performance, according to a Wilcoxon signed-rank test with Bonferroni correction.

COAD	Accuracy	AUC-ROC	AUC-PR
SC	0.30 ± 0.00	-	-
MLP	**0.41 ± 0.09**	0.66 ± 0.06	**0.39 ± 0.09**
CGEN-Best	0.41 ± 0.08	**0.65 ± 0.06**	0.39 ± 0.07
CMON-Best	**0.41 ± 0.09**	**0.65 ± 0.07**	**0.38 ± 0.08**
SNF-Best	0.37 ± 0.10	0.63 ± 0.07	**0.36 ± 0.07**

around its mean with a smaller interquartile range. This is not the case for COAD and LUAD datasets when comparing the MLP to the other algorithms.

No outliers were observed and most violin plots seemed to represent a Gaussian distribution. An exception to that is the MLP performance for the LUAD dataset, which has a little bump in its left tail, which could maybe be more properly modeled by a bi-modal distribution. This shape has also appeared in some other scenarios and this is why we preferred to perform the statistical tests with a Wilcoxon signed-rank test, which is a non-parametric test (WOOLSON, 2007). In Appendix A.2, the other distributions for each network generation strategy considering the thresholds 95% and 99% are presented.

Tables 5.5, 5.6 and 5.7 show how the best GNN for each network generation strategy compares with the two adopted baselines in terms of accuracy, AUC-ROC and AUC-PR. All values were rounded to two decimal places and the cells of each table represent the mean value for the 50 executions ± their standard deviation for that specific metric.

The first thing that can be noticed is that the SC approach, adopted by Wang et al. (2014), is the worst among all the tested methods. Despite its computational efficiency, there seems to be no reason to adopt such an approach, since it is the least accurate one and has also all the limitations discussed in Section 4.4.

Nonetheless, the graph-based algorithms could also not overcome the performance of a traditional MLP with statistical significance, considering a Wilcoxon signed-rank test

Table 5.7 – Performances comparison in LUAD dataset of the best setups for each network generation strategy and each baseline. Highlights in green indicate the best performance for each metric and results in bold indicate no statistically significant difference ($p < 0.05$) compared to the best performance, according to a Wilcoxon signed-rank test with Bonferroni correction.

LUAD	Accuracy	AUC-ROC	AUC-PR
SC	0.30 ± 0.01	-	-
MLP	**0.45 ± 0.06**	0.71 ± 0.04	0.46 ± 0.07
CGEN-Best	0.46 ± 0.07	**0.70 ± 0.04**	0.43 ± 0.07
CMON-Best	0.43 ± 0.07	0.70 ± 0.04	0.42 ± 0.06
SNF-Best	**0.45 ± 0.07**	**0.70 ± 0.05**	**0.44 ± 0.07**

with Bonferroni correction and $p < 0.05$. Some interesting points to mention are that the CGEN strategy was significantly worse than MLP regardless of the metric in the KIRC dataset and that the CMON strategy was significantly worse than the best method for all metrics in the LUAD dataset. While in all cases there was always a strategy comparable to the MLP, with no statistically significant difference, the baseline is way easier to implement, and the input data is also easier to process for accomplishing the integration of the multi-omics. Thus, when only reasoning about the convenience and performance for the investigated classification task in the tested datasets, there seems to be no advantage for using the network modeling approach with the tested GNNs, contrary to the findings of Zhang et al. (2022) and Baul et al. (2022).

It is important to notice that one of the mentioned studies did not use multi-omics data, and neither of the works tried to tackle the challenging task of cancer staging. Additionally, Zhang et al. (2022) utilized DAE for learning hidden features for the nodes, instead of using all the data features; and applied a DeepGCN model, instead of the original GCN. However, given the great impact on performance observed by changing the network generation strategy and the threshold application, we argue that this could be the greatest focal investigation point, and, as reviewed in Chapter 3, both works did not mention their similarity matrix sparsification methodology.

An additional observation is that the performance results of the investigated methods by our work are not even close to the ones reported by Yu et al. (2020), who also studied the cancer staging classification task, although with gene co-expression networks and under the perspectives of a graph classification task. A possible explanation for the poor performances obtained in our experiments may be due to the absence of feature selection or other dimensionality reduction approaches, contrary to what the related work did. With low sample size and large feature spaces, deep learning algorithms as applied in our study may suffer more with performance drops. Thus, this extra data processing step could be important to improve the final observations.

Even with all the presented negative arguments against the usage of the patient similarity network data modeling technique with GNNs, there is still room for their usage, since, for all metrics in all datasets, there was always a GNN with comparable performance. As pointed out by Pai and Bader (2018), there are some advantages to modeling the problem with patient similarity networks. Among them, the authors mention that these networks can naturally handle heterogeneous data by converting each data type into a similarity network; can naturally handle missing data, as a patient missing in one network may be in another and could still be used; and also provides greater interpretability, as this representation can convert the data into network views where the decision boundary can be visually evident, besides being conceptually analogous to clinical diagnosis, which often involves a physician relating a patient to a mental database of similar patients they have seen. In addition to that, we note that such a modeling approach can easily enable semi-supervised learning, which is the original application scenario thought for GCN, and also have the ability to propagate neighbors features through the network for handling missing information, as proposed recently by Rossi et al. (2021).

Finally, we also observe that the node classes themselves could be used as features of the training nodes. This would not only provide significantly more information to the patients network, helping in the classification of similar patients, but also would not be naturally available to the tabular data modeling approach used by algorithms such as the investigated MLP. In fact, the netDx method proposed by Pai et al. (2019), utilizes only label propagation as its classification technique over biological networks and reported improved performance compared to other machine learning algorithms, which indicates a very promising research opportunity that was not investigated here.

With the analyzes made in these two first sections of the results chapter, we answered four out of the five questions posed in Chapter 1. We demonstrated 1) how well can GNNs applied to multi-omics patient similarity networks perform for the cancer staging task; 2) that these GNNs did not outperform a simpler and cheaper MLP model applied to corresponding tabular data; 3) what were the performance impacts of changing the network generation strategy and selected sparsification threshold; and 4) that, despite some expected performance differences, the results' conclusions were comparable among the three different selected cancer datasets. In the following sections, we discuss the hyperparameters tuning results for answering the final fifth question; and analyze the resulting patient similarity networks for investigating further how the thresholds and network generation strategies impact the final network.

5.3 Hyperparameters

As explained in Chapter 4, there were a number of hyperparameters subjected to hyperparameter tuning. This process is important for analyzing the applicability of the whole methodology with less interference from users' arbitrary choices since the ANNs' tuner can select different combinations of hyperparameters for achieving a better performance. It is important to remember when analyzing the number of layers for the GNNs, that this parameter tells how many hops in the neighborhood the models should consider. Thus, if only one layer is used, it means only the first-order neighbors will influence a node's classification. Additionally, because there were too many variations of experiments and approaches, we chose to closely analyze and visually explore a subset of them. We selected the CGEN strategy for the first visual analysis since it is the most commonly used approach for generating sample similarity networks, and the 99% threshold for the second visual analysis, since it is the best one in terms of classification performance, as discussed previously.

To recap, the tuned hyperparameters and their respective possible choices were:

- Number of hidden layers: {1, 2, 3}
- Number of neurons on each layer: {32, 64, 128}
- Focal loss γ: {0, 1, 2}
- Dropout: {0, 0.1, 0.2, 0.3}
- Learning rate: {0.0001, 0.0005, 0.001, 0.005}
- GAT's number of attention heads: {2, 4, 8}

5.3.1 Number of layers and units on each layer

In Figure 5.7, we show the final optimal number of layers and neurons found by the Hyperband Algorithm for KIRC dataset using the CGEN network generation strategy. Figures 5.8 and 5.9 show the same information for COAD and LUAD, respectively. The MLP results were only plotted for comparison reasons, but its hyperparameters do not vary across thresholds and network generation strategies since it uses tabular data directly.

Figure 5.7 – Final optimal number of layers and neurons found by the Hyperband Algorithm for KIRC dataset using CGEN with the different tested thresholds.

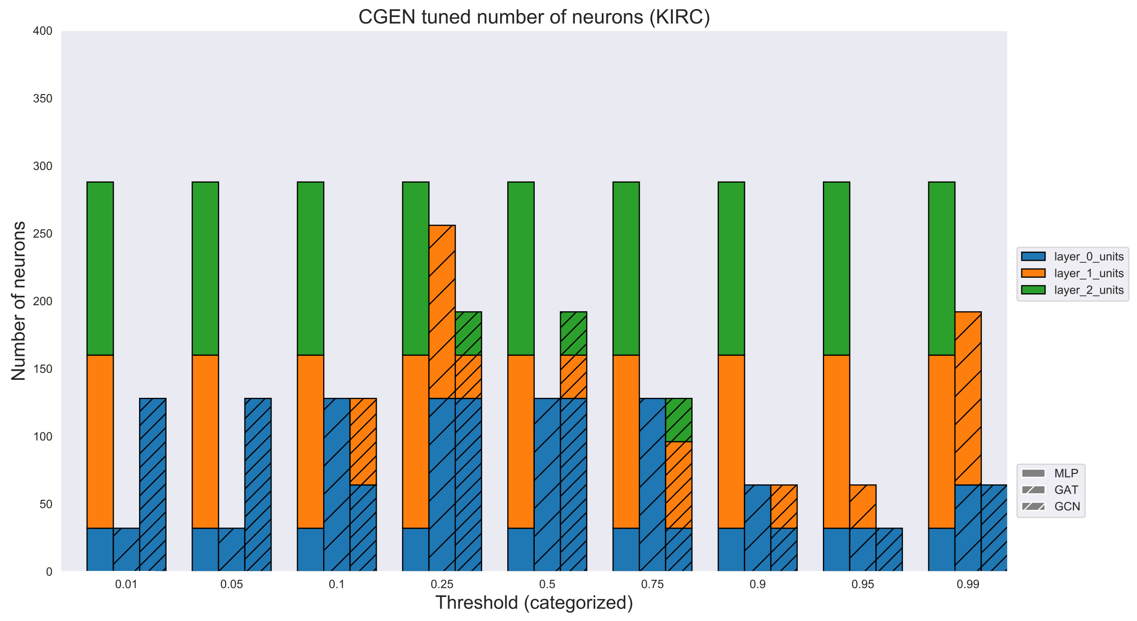
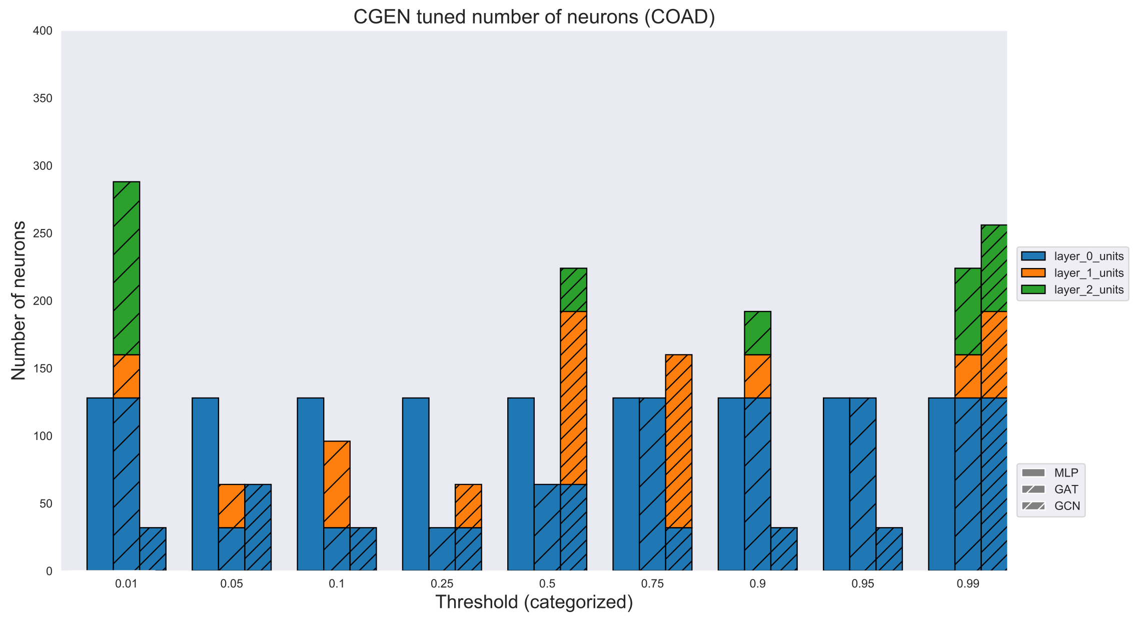
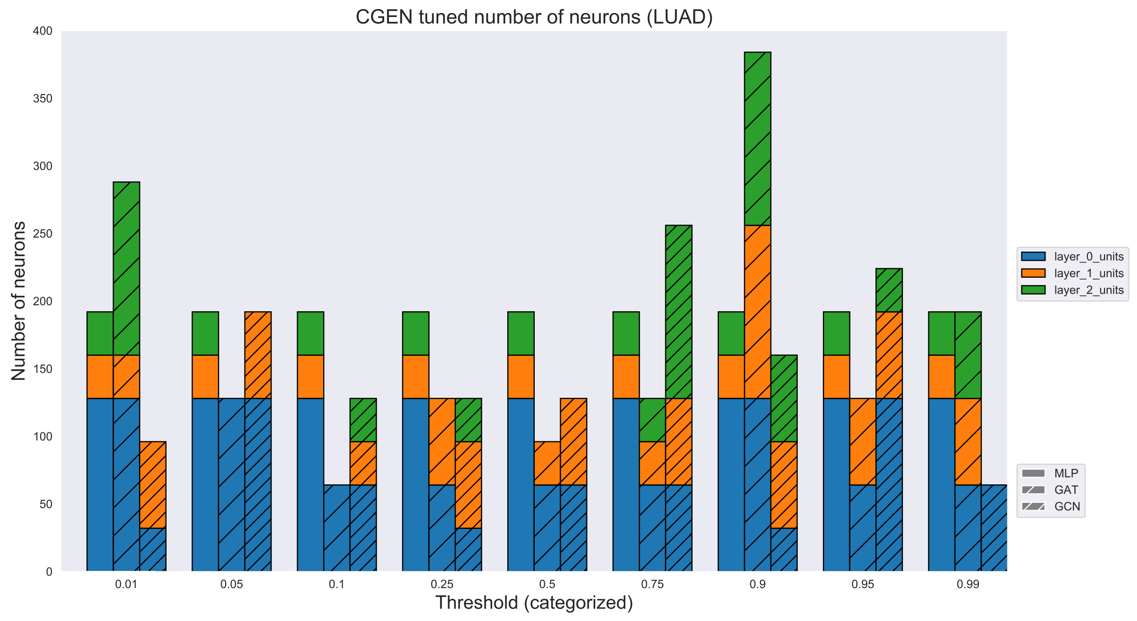


Figure 5.8 – Final optimal number of layers and neurons found by the Hyperband Algorithm for COAD dataset using CGEN with the different tested thresholds.



It is interesting to notice in the KIRC dataset that the tuner selected only 32 layer units and a single layer in the GAT's 1% and 5% thresholds, despite its performance limitations. For the 10% threshold, the chosen number of units for its single layer quadrupled, and yet it achieved the second lowest GAT-CGEN performance. In this dataset, no threshold caused GAT to use three layers of neurons, and the best setup did not use the maximum number of units. In fact, the maximum number of units and layers were only

Figure 5.9 – Final optimal number of layers and neurons found by the Hyperband Algorithm for LUAD dataset using CGEN with the different tested thresholds.



selected by GAT's tuner in the LUAD dataset using a CGEN threshold of 90%, and by GCN's tuner also in the LUAD dataset using a CMON threshold of 75%. Interestingly, in the KIRC dataset, the best GCN threshold, which vastly increased the GCN performance compared to the other thresholds, only used one layer with 64 units.

Out of all the performed experiments, there were only two cases in which a GNN-strategy did not achieve the best performance using the 99% threshold. In the KIRC dataset using a GCN with SNF, the best thresholds were the 75% and 95%. In both of these thresholds, the tuner algorithm chose only one GCN layer with 128 units in the 75% threshold, and also only one GCN layer but with 32 units for the 95% threshold. In contrast, both the 90% and the 99% thresholds used three GCN layers. This could indicate that the poorer performance could be related to an inadequate choice of hyperparameters from the utilized algorithm. This phenomenon also happens in the COAD dataset using a GCN with CGEN, in which the best threshold was the 90%, and the model had only one layer with 32 units, while for the 99% threshold the model had three layers with 128, 64 and 64 units respectively.

In the best CGEN threshold for the LUAD dataset, while GAT's preferred number of layers was three, GCN, again, worked better using only one layer. This pattern of GCN's best threshold selecting only one layer happened in all cases, except for COAD modeled with SNF (2 layers) and LUAD modeled with CMON (3 layers). For GAT's best threshold, the chosen number of layers and units on each layer varied a lot, but as opposed to GCN there was no case in which only one layer was selected, which can indicate that GAT's attention mechanism could work better with higher order neighborhoods.

5.3.2 Focal loss

One of the things we were interested in investigating with the hyperparameters tuning process was whether the focal loss (FL) function would be preferred for our classification task. A simple way for enabling this investigation is to set different possible choices for the γ parameter, including 0. Since we are using a weighted categorical version of the FL, a $\gamma = 0$ would mean that the traditional weighted categorical cross-entropy would be utilized instead, *i.e.* the focusing parameter for modulating factor $(1 - p)$ would turn this multiplying term into 1, the multiplicative identity.

Figures 5.10, 5.11 and 5.12 show the tuned γ for each threshold. Again, it is important to notice that the MLP value is only plotted for comparison since the network generation strategies and their different thresholds do not affect the MLP.

The first interesting aspect to notice is that a decent amount of cases fell into the $\gamma = 0$ scenario. Since our problem deals with highly unbalanced data and we optimized the hyperparameters choices by utilizing the AUC-PR metric, we expected the FL to be of particular interest to our problem. This is because the γ parameter helps to focus on the classes that are difficult for the model to classify. However, a $\gamma = 0$ means that this focusing mechanism was not the best option when it comes to achieving greater AUC-PRs. Specifically, there were seven cases in which the **best** performance of the GNN

Figure 5.10 – Tuned focal loss γ parameter across the tested sparsification thresholds applied to CGEN strategy in KIRC dataset.

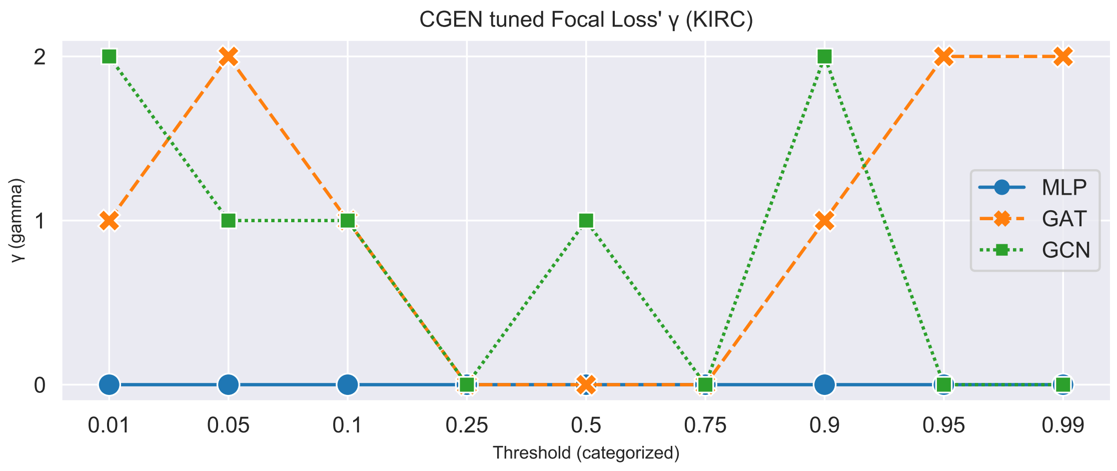


Figure 5.11 – Tuned focal loss γ parameter across the tested sparsification thresholds applied to CGEN strategy in COAD dataset.

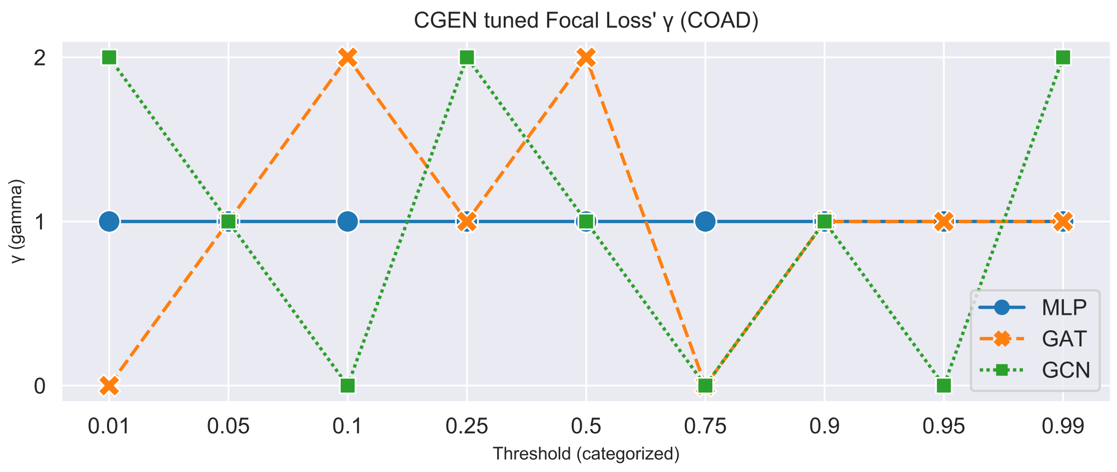
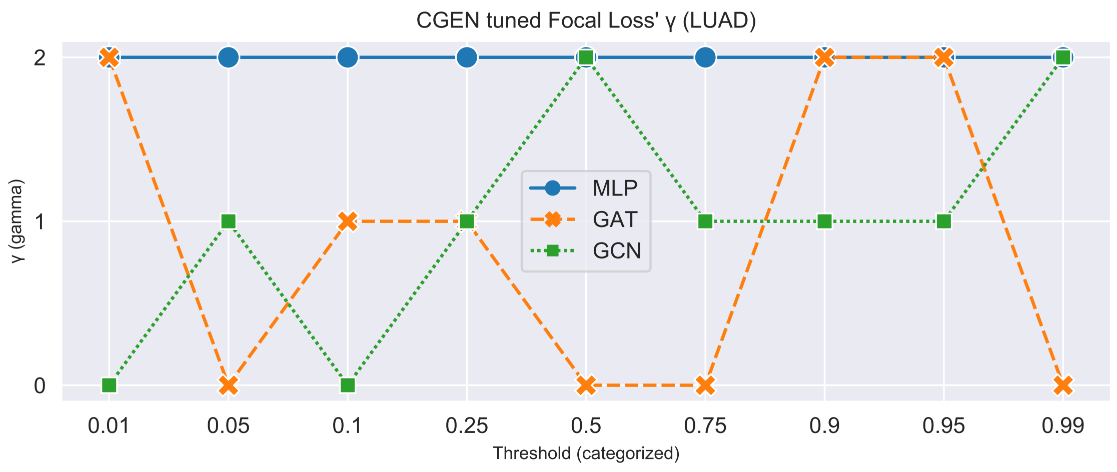


Figure 5.12 – Tuned focal loss γ parameter across the tested sparsification thresholds applied to CGEN strategy in LUAD dataset.



algorithm used a $\gamma = 0$. This corresponds to 35% of the best results (dataset-GNN-strategy wise). The cases in which such a phenomenon happens are the following:

- On KIRC dataset: GCN-CGEN-99%, GAT-CMON-99%, GAT-SNF-90%
- On COAD dataset: GAT-CMON-99%, GAT-SNF-99%, GCN-SNF-99%
- On LUAD dataset: GAT-CGEN-99%

Besides, this tuned choice for γ was also present in the MLP experiment executed in the KIRC dataset. While this was expected to occur in a few cases, we were surprised by the number of hyperparameter-tuned settings that included such choice, particularly in those with the best performance. These results cannot but cast some doubt on whether the FL can actually be useful for our kind of data, highlighting its generalization limitation

Table 5.8 – Tuned hyperparameters for MLP and the best GNN-based setup in each dataset. The LR column refers to the learning rate, and the AH to the number of attention heads used by GAT.

Dataset	Setup	Layers	Neurons	γ	Dropout	LR	AH
KIRC	GAT CMON-99%	3	[32, 64, 32]	0	0	0.0005	2
	MLP	3	[32, 128, 128]	0	0	0.0005	-
COAD	GAT CGEN-99%	3	[128, 32, 64]	1	0	0.0005	4
	MLP	1	[128]	1	0	0.0005	-
LUAD	GAT SNF-99%	2	[64, 32]	2	0	0.001	4
	MLP	3	[128, 32, 32]	2	0	0.0005	-

for different unbalanced datasets than the investigated by (LIN et al., 2017).

5.3.3 Best hyperparameters

Finally, we aim to address the fifth question our study proposes to address: what are the best hyperparameters for each investigated algorithm? For simplicity, we chose to document here only the tuned parameters for the best setups of each dataset (see Table 5.4), as well as the MLP versions. Table 5.8 shows the resulting tuned hyperparameters.

It is interesting to notice that none of the top performative GNNs used only one layer, suggesting that the 2- and 3-order neighbors are also important and actually contributed to the classification task. The number of neurons varied and no pattern such as "farthest neighbor features yielded a layer with less number of neurons" was observed. The preferred learning rate was 0.0005, although in one GNN scenario a learning rate two times higher was chosen.

A more peculiar scenario depicted by Table 5.8 is the ANNs' dropout avoidance. All the experiments showcased in Table 5.8 used a dropout of zero. Since this is a very popular regularization strategy to avoid high-variance models, one could make a case arguing that maybe the ANNs were oversimplifying the assumption. However, this is most unlikely given that the tuner algorithm could choose more complex networks with more learning parameters. Another explanation for this is that either the tuner algorithm did not have enough epochs to decide whether a specific setup choice was better than the other, or it chose a close to perfect ANN complexity architecture given that it 1) did not overfit the data, since no dropout was applied; and 2) did not underfit the data, since the maximum number of layers/neurons were not selected. An exception to 2) is the GAT CGEN-99% setup, in which the maximum number of neurons for its first layer was in fact selected by the tuner.

5.4 Networks visualization

In order to reason about the classification limitations in the networks with lower thresholds, we propose to analyze these networks visually. In our plots, we represent each node with a specific color, indicating its class. For the edges, we chose to depict in black the ones connecting nodes of the same class, and in orange the ones connecting nodes of different classes. Additionally, we removed all auto-connections for achieving a cleaner visualization, but the reader has to keep in mind that all nodes have an auto-connection.

Since all cancer datasets showed similar behaviors for each network generation strategy and threshold, we arbitrarily concentrate our discussion on KIRC. Figures 5.13 and 5.14 show the proposed network visualization for CGEN using the thresholds 75% and 90%, respectively. By looking at these plots, it becomes very clear why the GNNs can not properly classify the samples, having the worst results in lower thresholds. The 75% threshold produces a very chaotic network with lots of connections between nodes of different classes, which is certainly negatively impacting the learning task. Lower thresholds produce even messier networks and the classification only gets more challenging, since each node classification receives feature inputs from a lot of neighbors that do not have the same class. Thus, there is no doubt that a proper network modeling technique is of great importance for the node classification task.

Figures 5.15 and 5.16 show the KIRC networks for the two most strict investigated thresholds, 95% and 99%. Even the 95% network has lots of connections between nodes of different classes, although in this network we start to see better separation of the green and magenta nodes, *i.e.* stage I and stage III patients, respectively. Moving to the pickier threshold of 99%, we observe that most connections are lost and start questioning if the resulting structure is representative of a sample similarity network at all since the majority of the nodes do not have any edges, with exception of their own not plotted auto-connection. This can indicate that a network-free representation is indeed preferred and why the 99% threshold achieved better results: as fewer connections are added, more closely the problem becomes to a tabular classification one in which the only input for the classifier deciding a sample's class is this sample's own features. This could also explain why there were several cases of GNNs performance in the 99% threshold with no statistically significant difference compared to the MLP performances.

Figure 5.13 – KIRC patient similarity network generated using the CGEN strategy with a threshold of 75%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

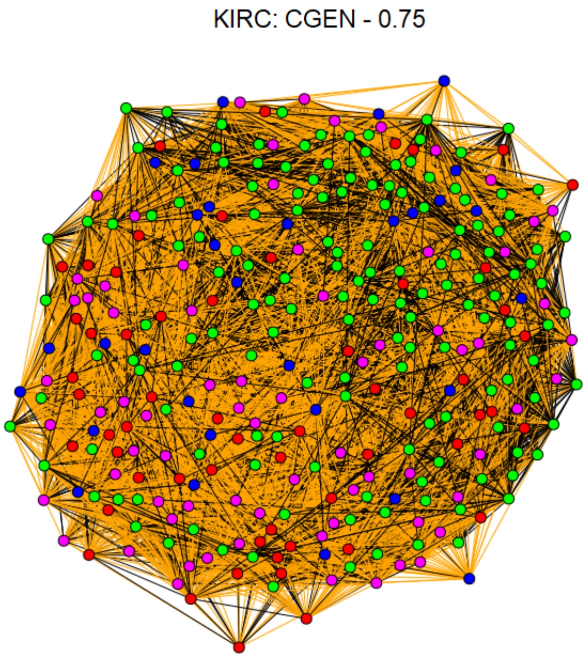


Figure 5.14 – KIRC patient similarity network generated using the CGEN strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

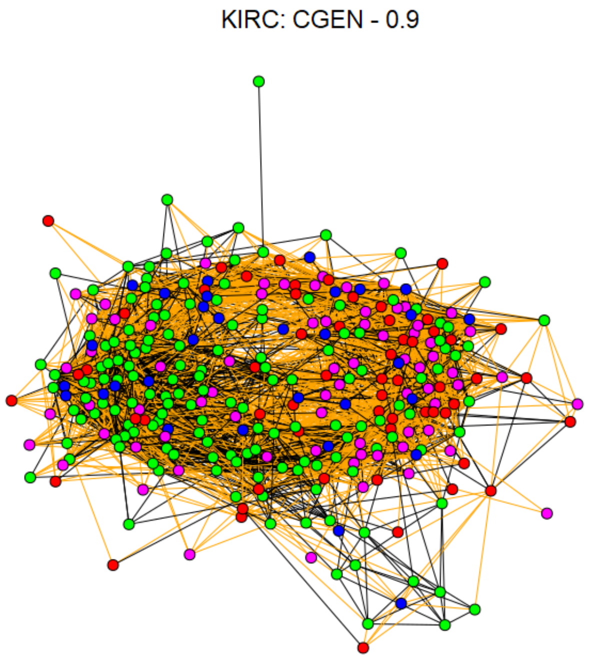
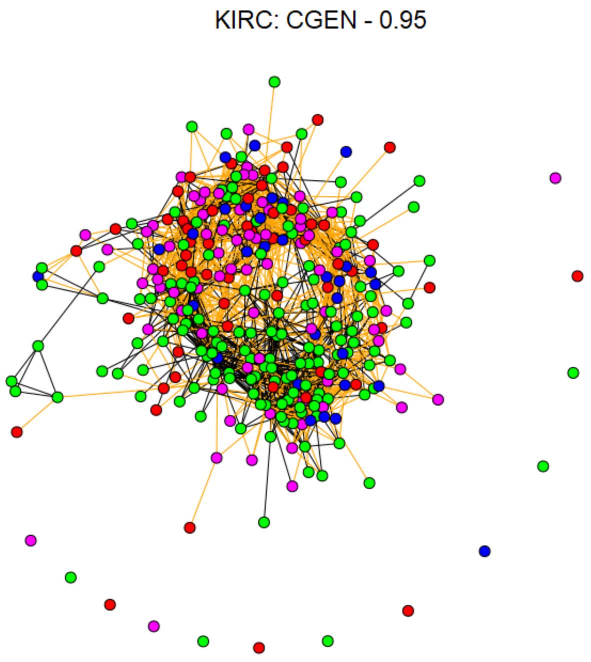
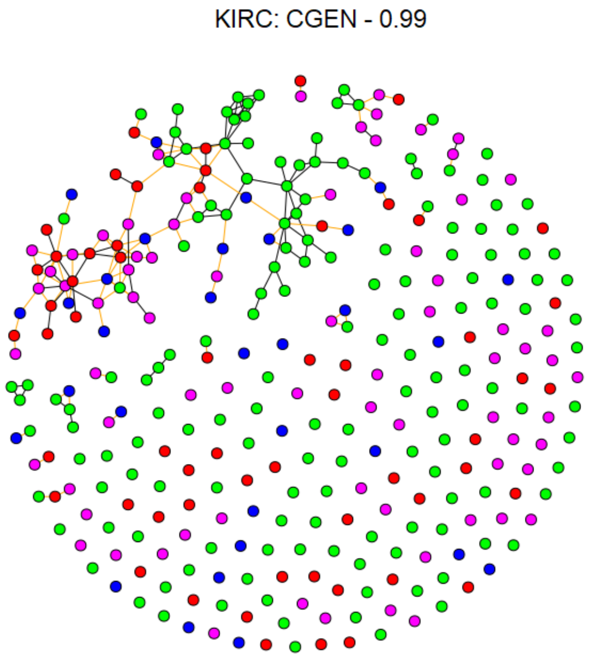


Figure 5.15 – KIRC patient similarity network generated using the CGEN strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.



Therefore, GNNs performances in the 99% threshold could be questioned, and more thresholds between 90% and 99% should be explored. The best case scenario would be to algorithmically decide the most appropriate threshold to use, although, as we showed, the sole GNN performance could not be an adequate heuristic for that. In Appendix A.3 we show other resulting patient similarity networks, and in Appendix B.1 we show a summary of how many nodes were connected only with themselves for each network generation strategy and threshold.

Figure 5.16 – KIRC patient similarity network generated using the CGEN strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.



6 CONCLUSIONS

Cancer staging with multi-omics integrated data is an investigation topic that has been gaining more attention recently, although gene expression alone applied to cancer diagnosis is still the most typical kind of research present in the literature. Some methods based on patient similarity have been proposed recently, with promising results. However, we could not find any study investigating this type of data representation for cancer staging, particularly utilizing multi-omics data. Additionally, some of the few studies investigating sample similarity networks do not even apply recent more advanced methods for graph representation learning, such as GNNs. Here, we investigated the applicability of GNNs to sample similarity networks generated with multi-omics data for the multi-class classification problem of cancer staging.

Contrary to related literature, our findings did not show classification performance advantages for the network modeling approach compared to a simpler multilayer perceptron applied to corresponding tabular integrated data. We show that the sparsification method applied to the similarity matrix and the computation of these similarities among pair-wised samples are of great relevance for a proper classification using GNNs, *i.e.* the modeling quality of the networks is key to achieving better classification performance. Even though these sample-sample networks have been studied by the cancer scientific community, a more comprehensive and systematic investigation about them is required. Our findings contribute to the understanding of such networks and the justification for more research efforts in this direction.

Nonetheless, as discussed in Chapter 5, there are still relevant application spaces for such networks. Among the applications, we mentioned its usage for semi-supervised learning and the ability to easily integrate and deal with heterogeneous and missing data. Since there was always a GNN-strategy setup capable of achieving similar performance results as the MLP adopted as baseline, this makes the approach suitable for the same task, although the network methods are more complex to configure and use. Besides, there were some promising research opportunities pointed out in the results discussions, such as the utilization of a dimensionality reduction technique and the usage of training samples' classes as features for the classification of the other samples. Additionally, for ML researchers and practitioners interested in the same classification task, we provide a comprehensive study of some important hyperparameters utilized by the investigated algorithms, which could guide future research efforts.

Therefore, Graph Neural Networks are applicable to cancer staging using multi-omics sample similarity networks. However, the method could not achieve similar performance to the state-of-the-art (YU et al., 2020) and more investigation is needed to reason if this is due to the utilized data (DUAN et al., 2021) or lack of additional/different modeling techniques and classification algorithms.

REFERENCES

ABDOLRASOL, M. G. et al. Artificial neural networks based optimization techniques: A review. **Electronics**, MDPI, v. 10, n. 21, p. 2689, 2021.

AHMAD, F. B.; ANDERSON, R. N. The leading causes of death in the us for 2020. **Jama**, American Medical Association, v. 325, n. 18, p. 1829–1830, 2021.

ALVAREZ, H. et al. Widespread hypomethylation occurs early and synergizes with gene amplification during esophageal carcinogenesis. **PLoS genetics**, Public Library of Science San Francisco, USA, v. 7, n. 3, p. e1001356, 2011.

BARABÁSI, A.-L. Network science. **Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences**, The Royal Society Publishing, v. 371, n. 1987, p. 20120375, 2013.

BAUL, S. et al. omicsgat: Graph attention network for cancer subtype analyses. **bioRxiv**, Cold Spring Harbor Laboratory, 2022.

BERTSIMAS, D.; WIBERG, H. Machine learning in oncology: methods, applications, and challenges. **JCO Clinical Cancer Informatics**, American Society of Clinical Oncology, v. 4, 2020.

BHAWE, K. M.; AGHI, M. K. Microarray analysis in glioblastomas. In: **Microarray Data Analysis**. [S.l.]: Springer, 2015. p. 195–206.

BILBAO, I.; BILBAO, J. Overfitting problem and the over-training in the era of data: Particularly for artificial neural networks. In: IEEE. **2017 eighth international conference on intelligent computing and information systems (ICICIS)**. [S.l.], 2017. p. 173–177.

CENTER, B. I. T. G. D. A. Analysis-ready standardized tcga data from broad gdac firehose 2016_01_28 run. **Broad Institute of MIT and Harvard. Dataset.**, 2016.

CHANG, J. et al. Brain tumor segmentation based on 3d unet with multi-class focal loss. In: IEEE. **2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)**. [S.l.], 2018. p. 1–5.

CHEAIB, J. G. et al. Stage-specific conditional survival in renal cell carcinoma after nephrectomy. In: ELSEVIER. **Urologic Oncology: Seminars and Original Investigations**. [S.l.], 2020. v. 38, n. 1, p. 6–e1.

CHOI, E. et al. Learning the graphical structure of electronic health records with graph convolutional transformer. In: **Proceedings of the AAAI conference on artificial intelligence**. [S.l.: s.n.], 2020. v. 34, n. 01, p. 606–613.

CHOLLET, F. et al. **Keras**. 2015. <https://keras.io>.

COLOMBELLI, F.; KOWALSKI, T. W.; RECAMONDE-MENDOZA, M. A hybrid ensemble feature selection design for candidate biomarkers discovery from transcriptome profiles. **Knowledge-Based Systems**, p. 109655, 2022. ISSN 0950-7051. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0950705122008383>.

DATA61, C. **StellarGraph Machine Learning Library**. [S.l.]: GitHub, 2018. <https://github.com/stellargraph/stellargraph>.

DUAN, R. et al. Evaluation and comparison of multi-omics data integration methods for cancer subtyping. **PLoS computational biology**, Public Library of Science San Francisco, CA USA, v. 17, n. 8, p. e1009224, 2021.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. [S.l.]: MIT press, 2016.

HAN, P. et al. Gcn-mf: disease-gene association identification by graph convolutional networks and matrix factorization. In: **Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining**. [S.l.: s.n.], 2019. p. 705–713.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. The elements of statistical learning. springer series in statistics. **New York, NY, USA**, 2001.

HUANG, S.; CHAUDHARY, K.; GARMIRE, L. X. More is better: Recent progress in multi-omics data integration methods. **Frontiers in Genetics**, Frontiers, v. 8, p. 84, 2017.

JAMIESON, K.; TALWALKAR, A. Non-stochastic best arm identification and hyperparameter optimization. In: PMLR. **Artificial intelligence and statistics**. [S.l.], 2016. p. 240–248.

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

KOH, H. W. et al. iomicspass: network-based integration of multiomics data for predictive subnetwork discovery. **NPJ systems biology and applications**, Nature Publishing Group, v. 5, n. 1, p. 1–10, 2019.

KOUROU, K. et al. Machine learning applications in cancer prognosis and prediction. **Computational and structural biotechnology journal**, Elsevier, v. 13, p. 8–17, 2015.

LI, C. et al. 3dmol-net: learn 3d molecular representation using adaptive graph convolutional network based on rotation invariance. **IEEE Journal of Biomedical and Health Informatics**, IEEE, 2021.

LI, G. et al. Deepgcns: Can gcns go as deep as cnns? In: **Proceedings of the IEEE/CVF international conference on computer vision**. [S.l.: s.n.], 2019. p. 9267–9276.

LI, L. et al. Hyperband: A novel bandit-based approach to hyperparameter optimization. **The Journal of Machine Learning Research**, JMLR. org, v. 18, n. 1, p. 6765–6816, 2017.

LI, S. et al. Prenatal epigenetics diets play protective roles against environmental pollution. **Clinical epigenetics**, BioMed Central, v. 11, n. 1, p. 1–31, 2019.

LI, S.; TOLLEFSBOL, T. O. Dna methylation methods: Global dna methylation and methylomic analyses. **Methods**, Elsevier, v. 187, p. 28–43, 2021.

LI, X.; WANG, C.-Y. From bulk, single-cell to spatial rna sequencing. **International Journal of Oral Science**, Nature Publishing Group, v. 13, n. 1, p. 1–6, 2021.

LIN, T.-Y. et al. Focal loss for dense object detection. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 2980–2988.

LIU, L. et al. Integrating sequence and network information to enhance protein-protein interaction prediction using graph convolutional networks. In: IEEE. **2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)**. [S.l.], 2019. p. 1762–1768.

LOCKE, W. J. et al. Dna methylation cancer biomarkers: translation to the clinic. **Frontiers in genetics**, Frontiers Media SA, v. 10, p. 1150, 2019.

LOWE, R. et al. Transcriptomics technologies. **PLoS computational biology**, Public Library of Science San Francisco, CA USA, v. 13, n. 5, p. e1005457, 2017.

LU, H.; UDDIN, S. A weighted patient network-based framework for predicting chronic diseases using graph neural networks. **Scientific reports**, Nature Publishing Group, v. 11, n. 1, p. 1–12, 2021.

LU, H. et al. A patient network-based machine learning model for disease prediction: The case of type 2 diabetes mellitus. **Applied Intelligence**, Springer, v. 52, n. 3, p. 2411–2422, 2022.

LUXBURG, U. V. A tutorial on spectral clustering. **Statistics and computing**, Springer, v. 17, n. 4, p. 395–416, 2007.

MA, B. et al. Diagnostic classification of cancers using extreme gradient boosting algorithm and multi-omics data. **Computers in biology and medicine**, Elsevier, v. 121, p. 103761, 2020.

MAAS, A. L. et al. Rectifier nonlinearities improve neural network acoustic models. In: CITESEER. **Proc. icml**. [S.l.], 2013. v. 30, n. 1, p. 3.

MAHMOOD, O. et al. Masked graph modeling for molecule generation. **Nature communications**, Nature Publishing Group, v. 12, n. 1, p. 1–12, 2021.

MANOOCHEHRI, H. E.; PILLAI, A.; NOURANI, M. Graph convolutional networks for predicting drug-protein interactions. In: IEEE. **2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)**. [S.l.], 2019. p. 1223–1225.

MANZONI, C. et al. Genome, transcriptome and proteome: the rise of omics data and their integration in biomedical sciences. **Briefings in Bioinformatics**, Oxford University Press, v. 19, n. 2, p. 286–302, 2018.

MATURANA, E. López de et al. Challenges in the integration of omics and non-omics data. **Genes**, MDPI, v. 10, n. 3, p. 238, 2019.

MENYHÁRT, O.; GYŐRFFY, B. Multi-omics approaches in cancer research with applications in tumor subtyping, prognosis, and diagnosis. **Computational and structural biotechnology journal**, Elsevier, v. 19, p. 949–960, 2021.

MO, Q. et al. A fully bayesian latent variable model for integrative clustering analysis of multi-type omics data. **Biostatistics**, Oxford University Press, v. 19, n. 1, p. 71–86, 2018.

MUZIO, G.; O'BRAY, L.; BORGWARDT, K. Biological network analysis with deep learning. **Briefings in bioinformatics**, Oxford University Press, v. 22, n. 2, p. 1515–1530, 2021.

NICORA, G. et al. Integrated multi-omics analyses in oncology: a review of machine learning methods and tools. **Frontiers in oncology**, Frontiers Media SA, v. 10, p. 1030, 2020.

PAI, S.; BADER, G. D. Patient similarity networks for precision medicine. **Journal of molecular biology**, Elsevier, v. 430, n. 18, p. 2924–2938, 2018.

PAI, S. et al. netdx: interpretable patient classification using integrated patient similarity networks. **Molecular systems biology**, v. 15, n. 3, p. e8497, 2019.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PRIOR, F. W. et al. Tcia: an information resource to enable open science. In: IEEE. **2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)**. [S.l.], 2013. p. 1282–1285.

QIN, H.; NIU, T.; ZHAO, J. Identifying multi-omics causers and causal pathways for complex traits. **Frontiers in genetics**, Frontiers Media SA, v. 10, p. 110, 2019.

REDDY, K. B. Microrna (mirna) in cancer. **Cancer cell international**, Springer, v. 15, n. 1, p. 1–6, 2015.

REUTER, J. A.; SPACEK, D. V.; SNYDER, M. P. High-throughput sequencing technologies. **Molecular cell**, Elsevier, v. 58, n. 4, p. 586–597, 2015.

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.

ROSSI, E. et al. On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features. **arXiv preprint arXiv:2111.12128**, 2021.

SCHÜBELER, D. Function and information content of dna methylation. **Nature**, Nature Publishing Group, v. 517, n. 7534, p. 321–326, 2015.

SHEN, R.; OLSHEN, A. B.; LADANYI, M. Integrative clustering of multiple genomic data types using a joint latent variable model with application to breast and lung cancer subtype analysis. **Bioinformatics**, Oxford University Press, v. 25, n. 22, p. 2906–2912, 2009.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. **IEEE Transactions on pattern analysis and machine intelligence**, Ieee, v. 22, n. 8, p. 888–905, 2000.

SONG, T.-A. et al. Graph convolutional neural networks for alzheimer's disease classification. In: IEEE. **2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)**. [S.l.], 2019. p. 414–417.

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.

SUPPLITT, S. et al. Current achievements and applications of transcriptomics in personalized cancer medicine. **International Journal of Molecular Sciences**, v. 22, n. 3, 2021. ISSN 1422-0067. Available from Internet: <https://www.mdpi.com/1422-0067/22/3/1422>.

TREVIZAN, B.; RECAMONDE-MENDOZA, M. Ensemble feature selection compares to meta-analysis for breast cancer biomarker identification from microarray data. In: SPRINGER. **International Conference on Computational Science and Its Applications**. [S.l.], 2021. p. 162–178.

VASWANI, A. et al. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017.

VELIČKOVIĆ, P. et al. Graph attention networks. In: **International Conference on Learning Representations**. [S.l.: s.n.], 2018.

VINCENT, P. et al. Extracting and composing robust features with denoising autoencoders. In: **Proceedings of the 25th international conference on Machine learning**. [S.l.: s.n.], 2008. p. 1096–1103.

WANG, B. et al. Similarity network fusion for aggregating data types on a genomic scale. **Nature methods**, Nature Publishing Group, v. 11, n. 3, p. 333–337, 2014.

WANG, M. et al. A high-speed and low-complexity architecture for softmax function in deep learning. In: IEEE. **2018 IEEE asia pacific conference on circuits and systems (APCCAS)**. [S.l.], 2018. p. 223–226.

WANG, Z.; WU, X.; WANG, Y. A framework for analyzing dna methylation data from illumina infinium humanmethylation450 beadchip. **BMC bioinformatics**, Springer, v. 19, n. 5, p. 15–22, 2018.

WEINBERG, R. A. **The Biology of Cancer**. [S.l.]: Garland Science, 2013.

WIEDER, O. et al. A compact review of molecular property prediction with graph neural networks. **Drug Discovery Today: Technologies**, Elsevier, v. 37, p. 1–12, 2020.

WOOLSON, R. F. Wilcoxon signed-rank test. **Wiley encyclopedia of clinical trials**, Wiley Online Library, p. 1–3, 2007.

WU, J. et al. Hyperparameter optimization for machine learning models based on bayesian optimization. **Journal of Electronic Science and Technology**, Elsevier, v. 17, n. 1, p. 26–40, 2019.

YANG, L.; SHAMI, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. **Neurocomputing**, Elsevier, v. 415, p. 295–316, 2020.

YI, H.-C. et al. Graph representation learning in bioinformatics: trends, methods and applications. **Briefings in Bioinformatics**, Oxford University Press, v. 23, n. 1, p. bbab340, 2022.

YOU, R. et al. Deepgraphgo: graph neural network for large-scale, multispecies protein function prediction. **Bioinformatics**, Oxford University Press, v. 37, n. Supplement_1, p. i262–i271, 2021.

YU, X. et al. Co-expression based cancer staging and application. **Scientific reports**, Nature Publishing Group, v. 10, n. 1, p. 1–10, 2020.

ZHANG, G. et al. A novel liver cancer diagnosis method based on patient similarity network and densegcn. **Scientific Reports**, Nature Publishing Group, v. 12, n. 1, p. 1–10, 2022.

ZHANG, S. et al. Discovery of multi-dimensional modules by integrative analysis of cancer genomic data. **Nucleic acids research**, Oxford University Press, v. 40, n. 19, p. 9379–9391, 2012.

ZHANG, X.; JONASSEN, I.; GOKSØYR, A. Machine Learning Approaches for Biomarker Discovery Using Gene Expression Data. In: NAKAYA, H. I. (Ed.). **Bioinformatics**. [S.l.]: Exon Publications, 2021. p. 53–64.

ZHAO, Y. et al. Tpm, fpkm, or normalized counts? a comparative study of quantification measures for the analysis of rna-seq data from the nci patient-derived models repository. **Journal of translational medicine**, BioMed Central, v. 19, n. 1, p. 1–15, 2021.

APPENDIX A — SUPPLEMENTARY FIGURES

A.1 GNNs classification performance across thresholds

Figure A.1 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (KIRC) using the correlation network based on gene expression levels (CGEN) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC (b) AUC-PR

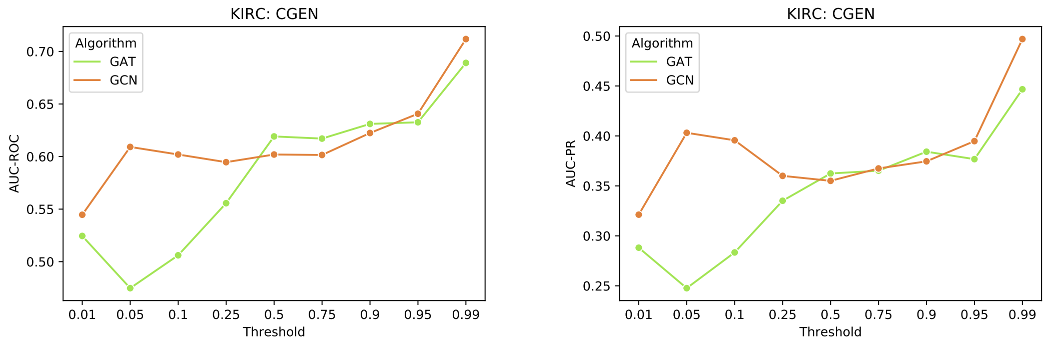


Figure A.2 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (COAD) using the correlation network based on gene expression levels (CGEN) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC (b) AUC-PR

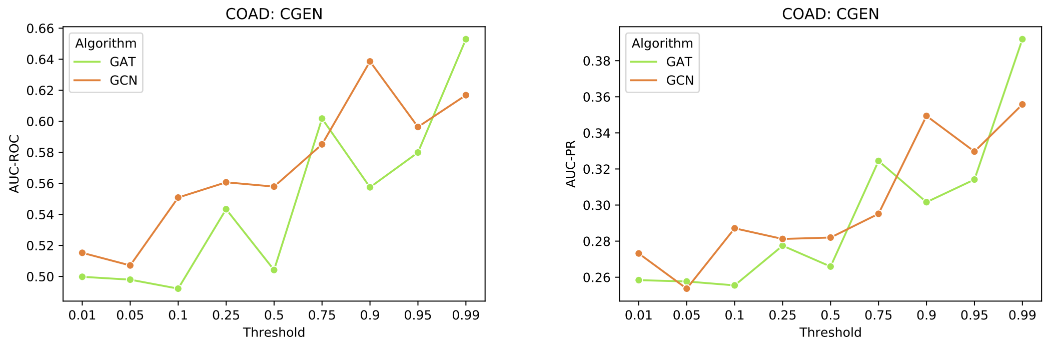
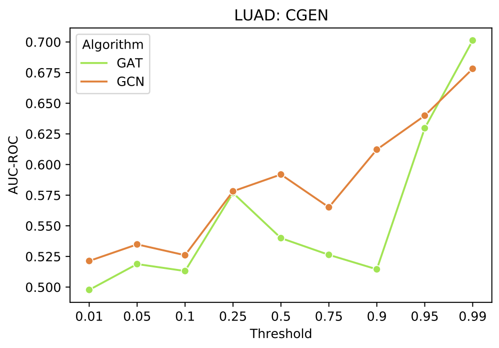


Figure A.3 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (LUAD) using the correlation network based on gene expression levels (CGEN) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC



(b) AUC-PR

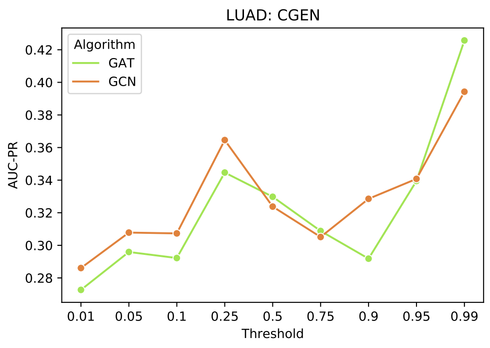
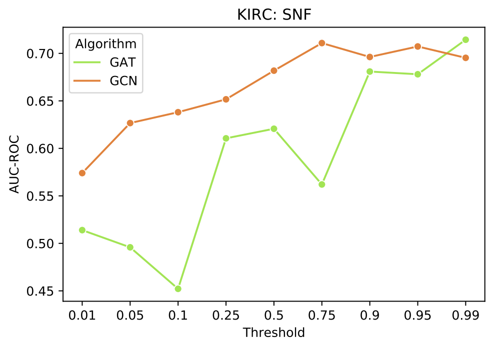


Figure A.4 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (KIRC) using the Similarity Network Fusion (SNF) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC



(b) AUC-PR

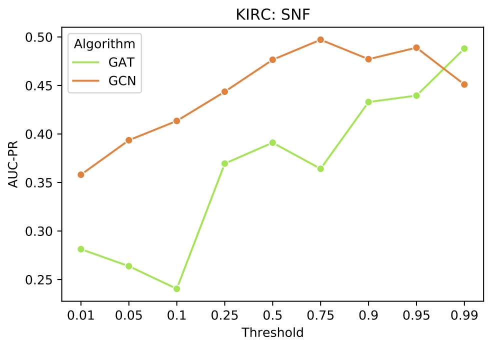
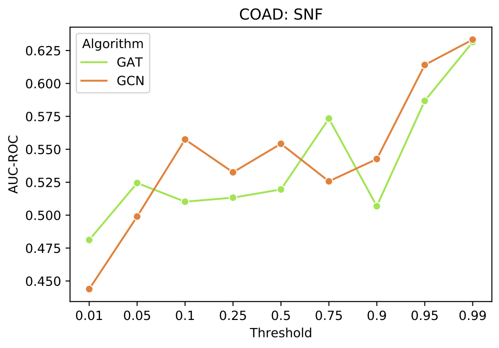


Figure A.5 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (COAD) using the Similarity Network Fusion (SNF) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC



(b) AUC-PR

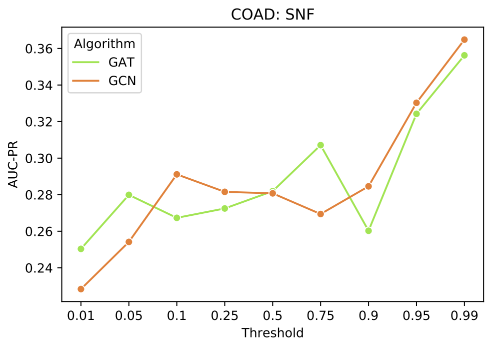
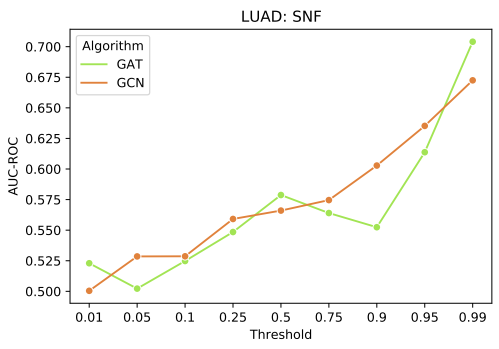
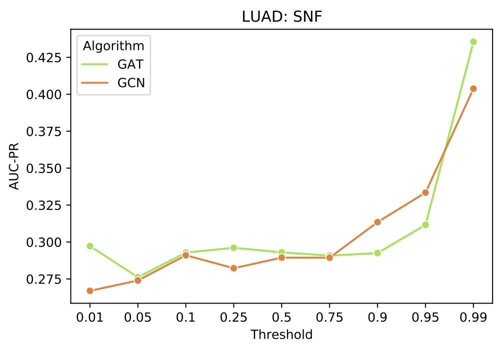


Figure A.6 – Performance comparison of GAT and GCN across the investigated thresholds for the kidney cancer data (LUAD) using the Similarity Network Fusion (SNF) strategy. Plot (a) shows the mean AUC-ROC and plot (b) shows the mean AUC-PR.

(a) AUC-ROC



(b) AUC-PR



A.2 GNNs and MLP classification performances

Figure A.7 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 99% threshold.

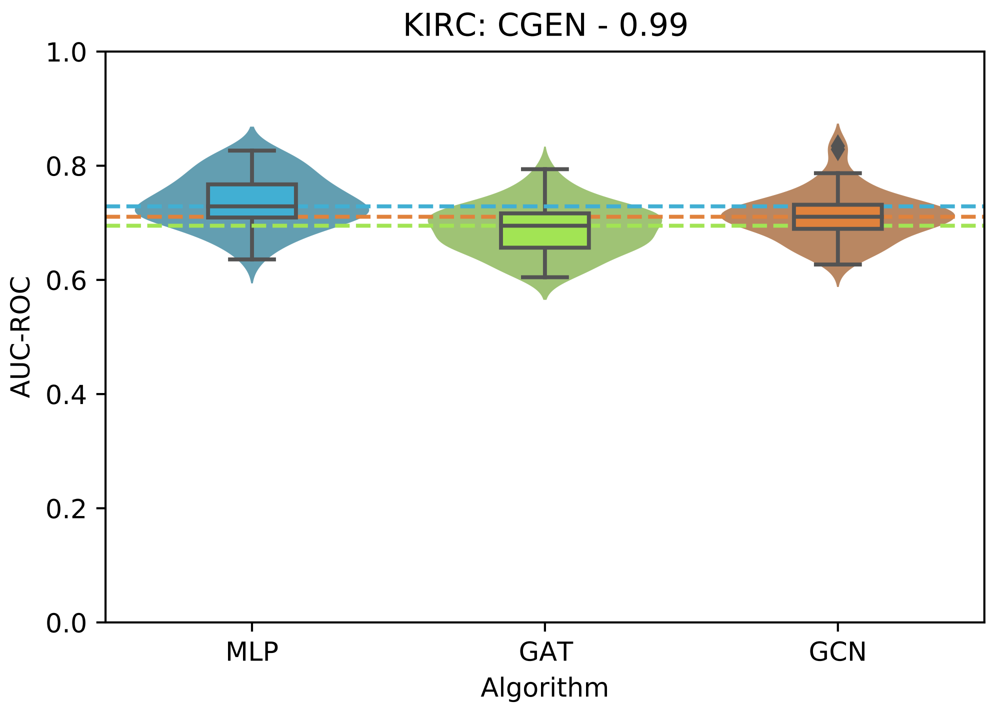


Figure A.8 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 95% threshold.

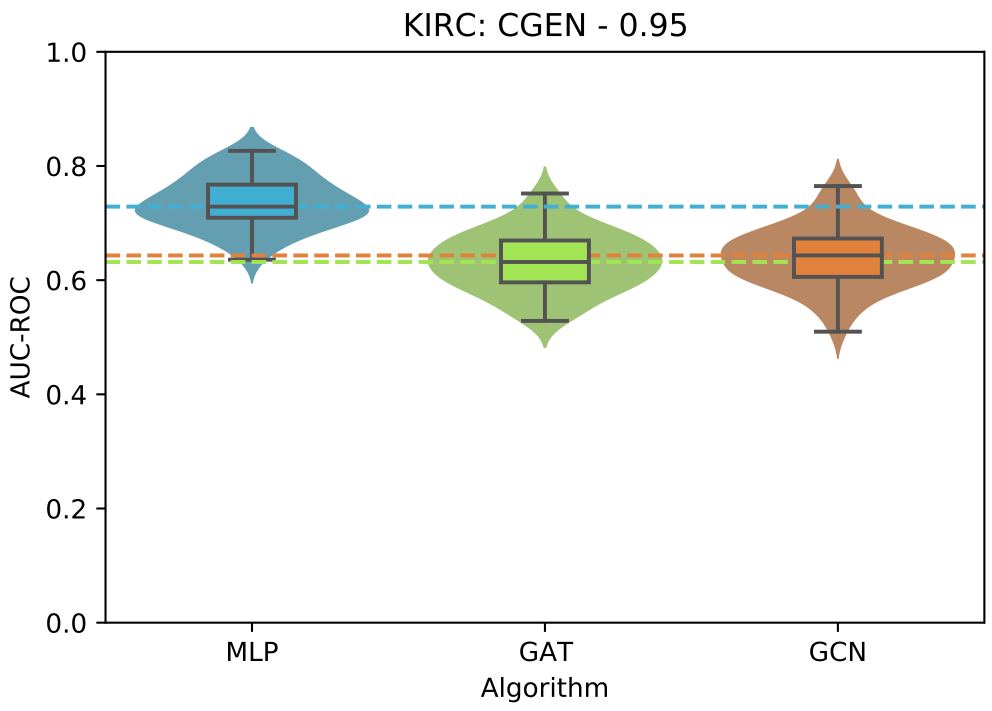


Figure A.9 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 95% threshold.

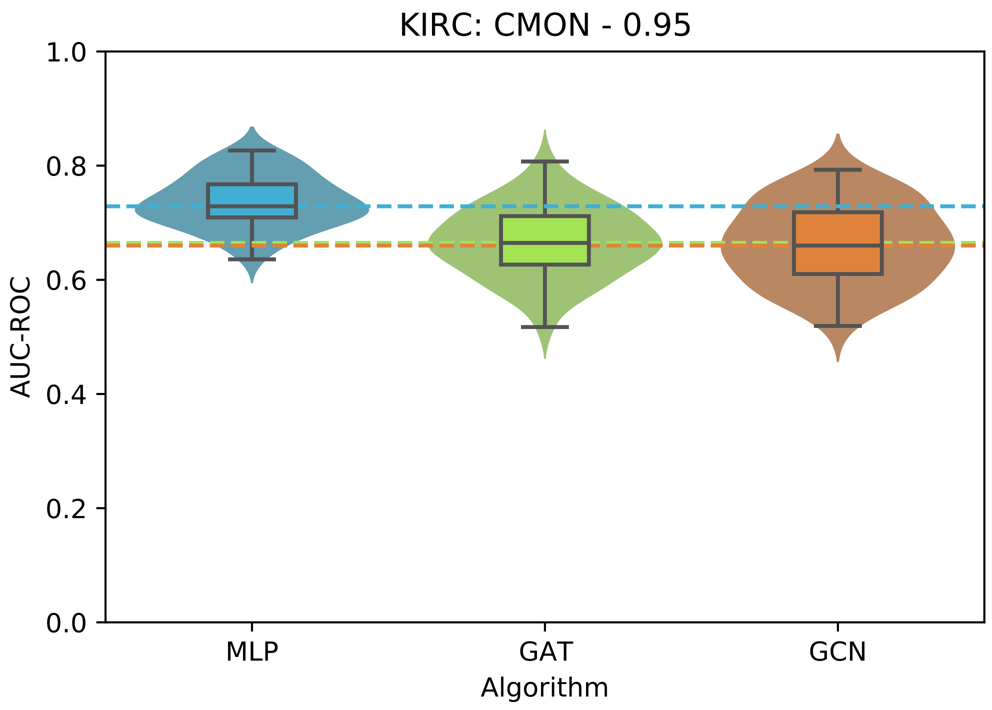


Figure A.10 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 99% threshold.

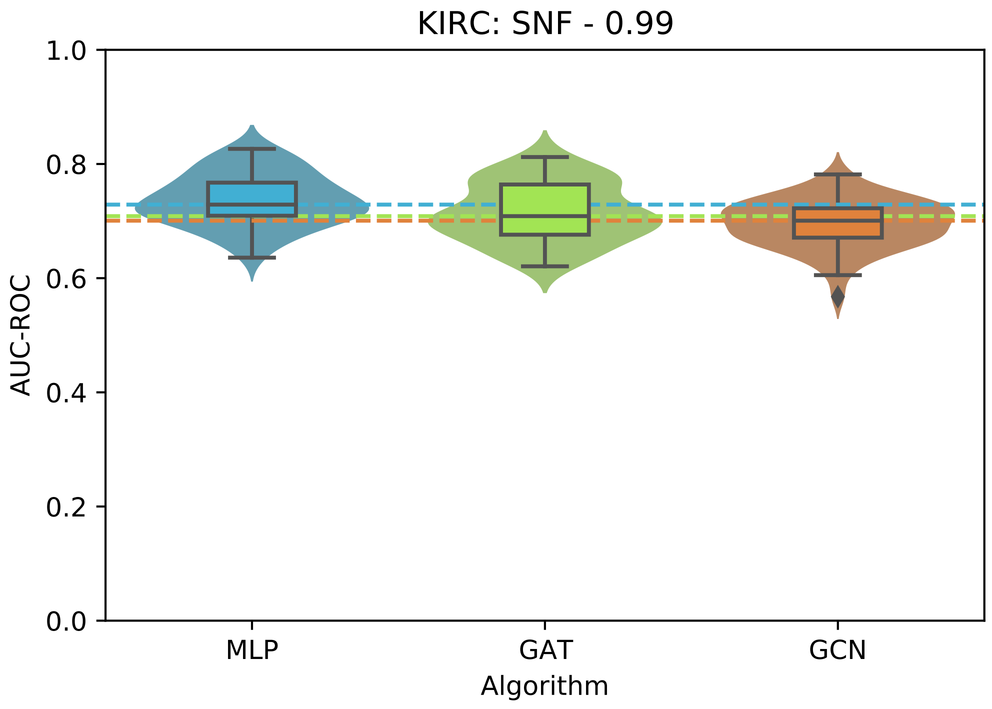


Figure A.11 – AUC-ROC performance comparison on KIRC dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 95% threshold.

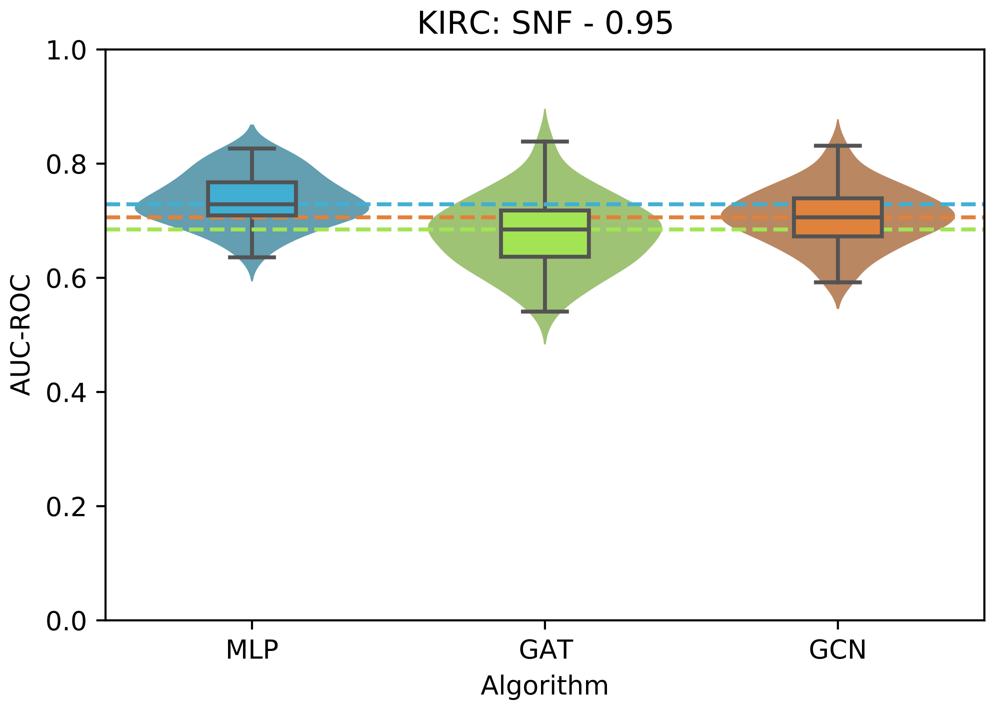


Figure A.12 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 95% threshold.

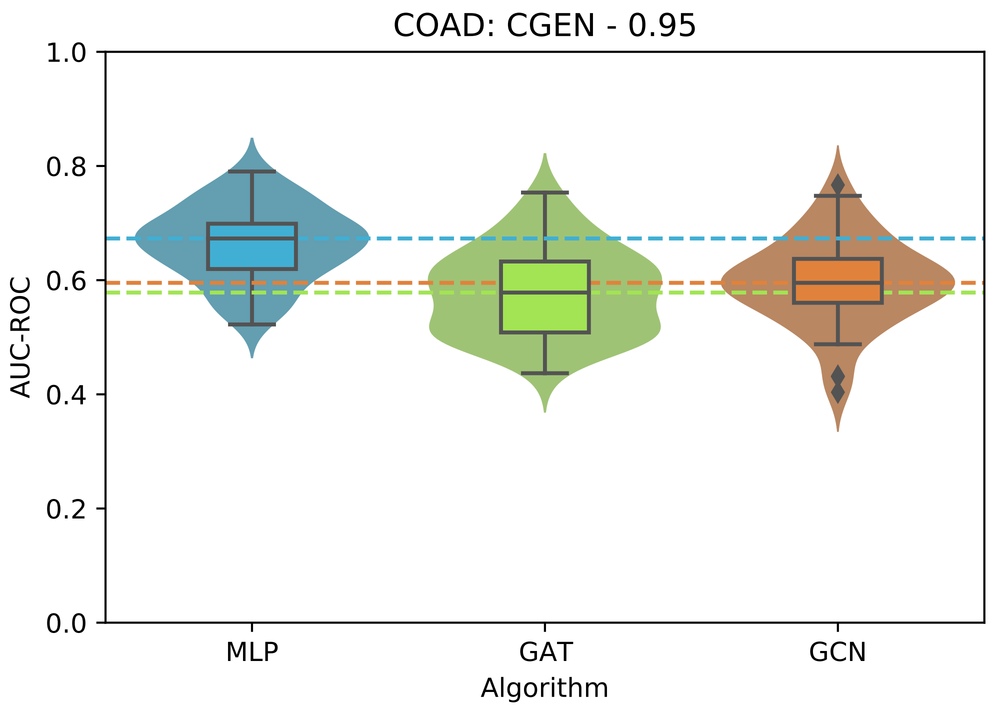


Figure A.13 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 99% threshold.

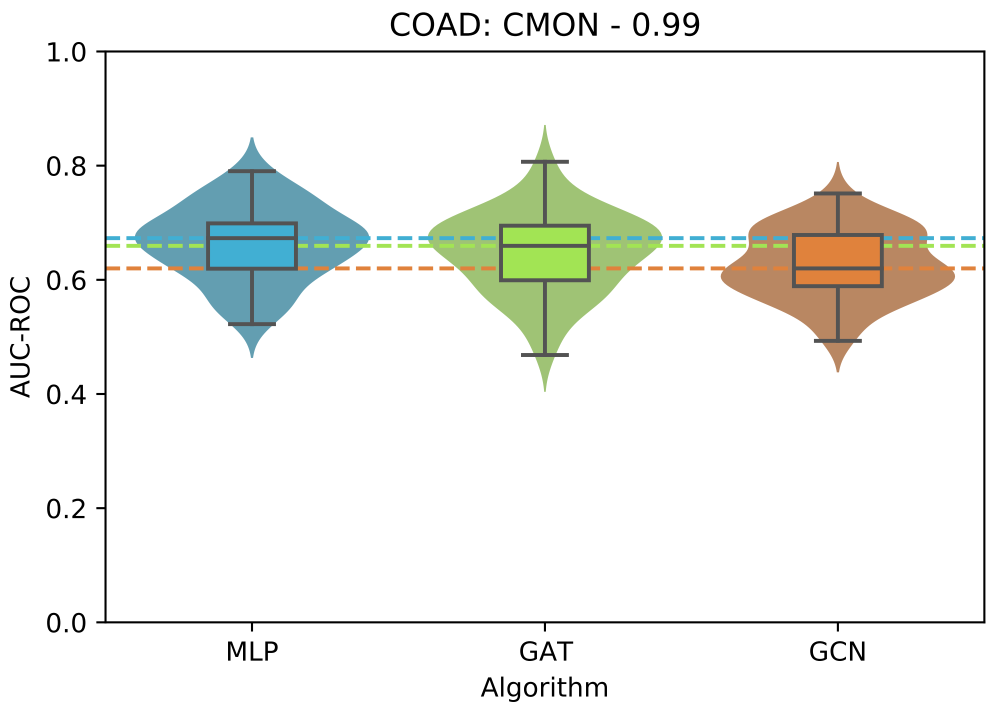


Figure A.14 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 95% threshold.

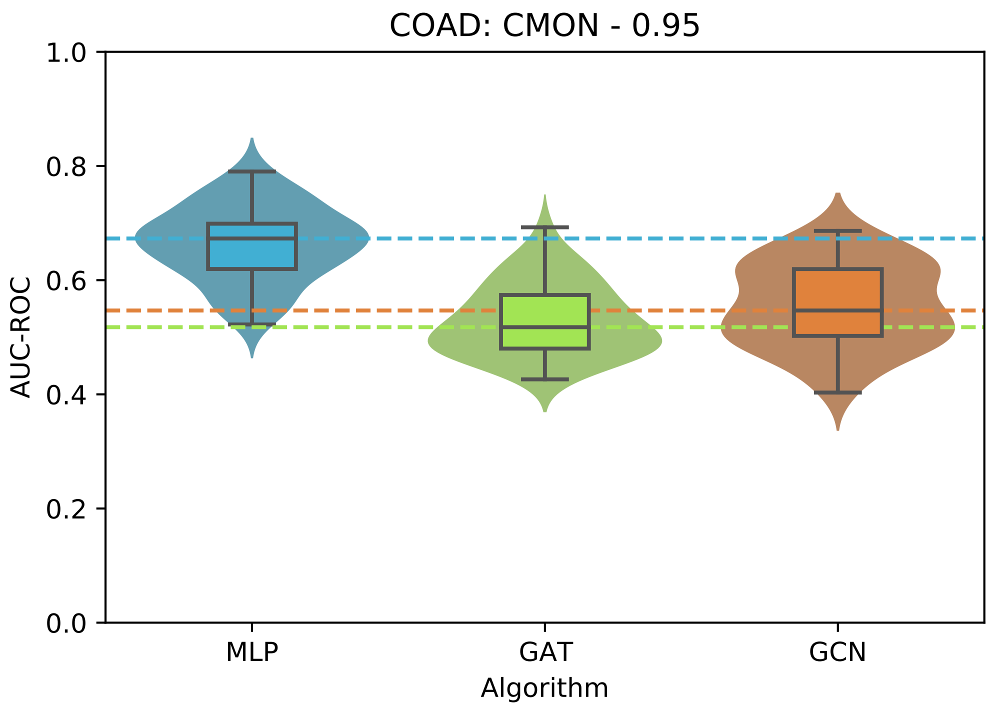


Figure A.15 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 99% threshold.

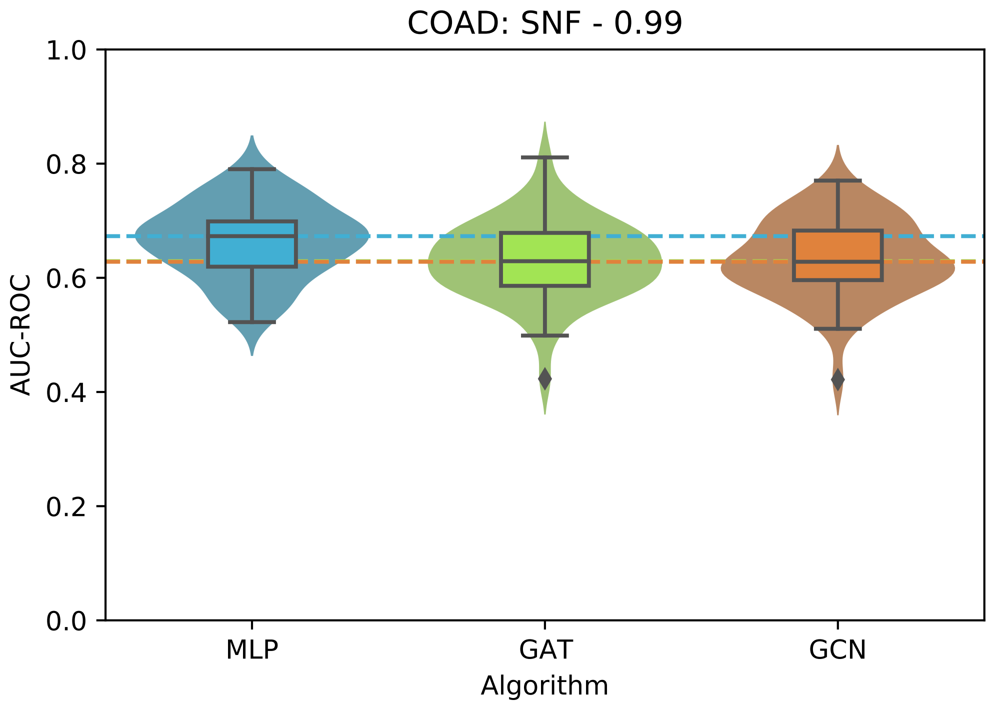


Figure A.16 – AUC-ROC performance comparison on COAD dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 95% threshold.

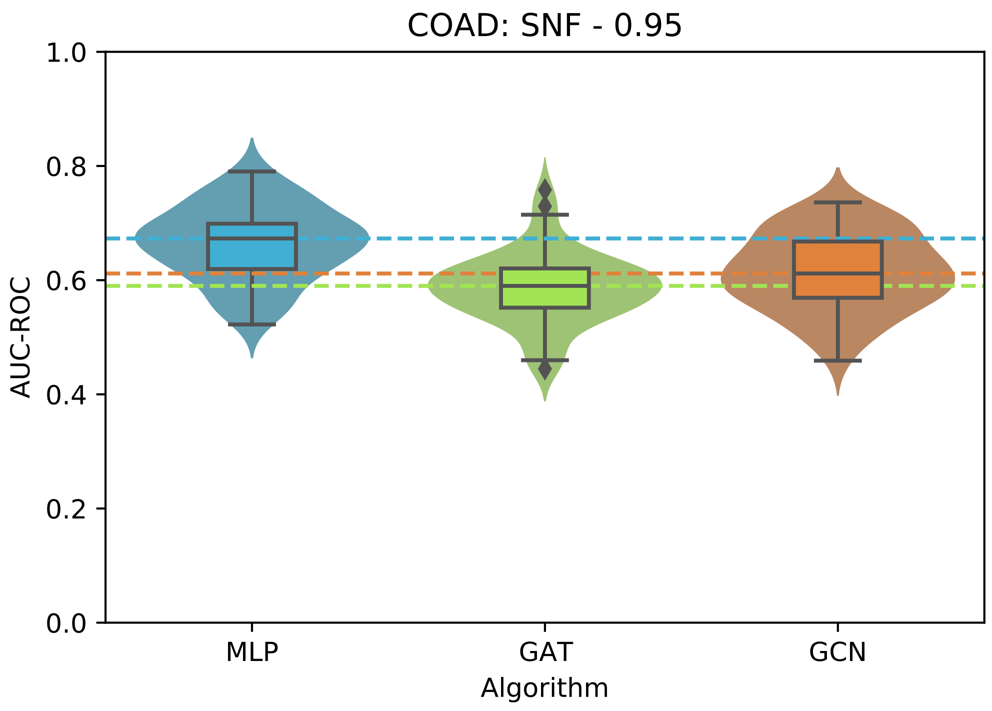


Figure A.17 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 99% threshold.

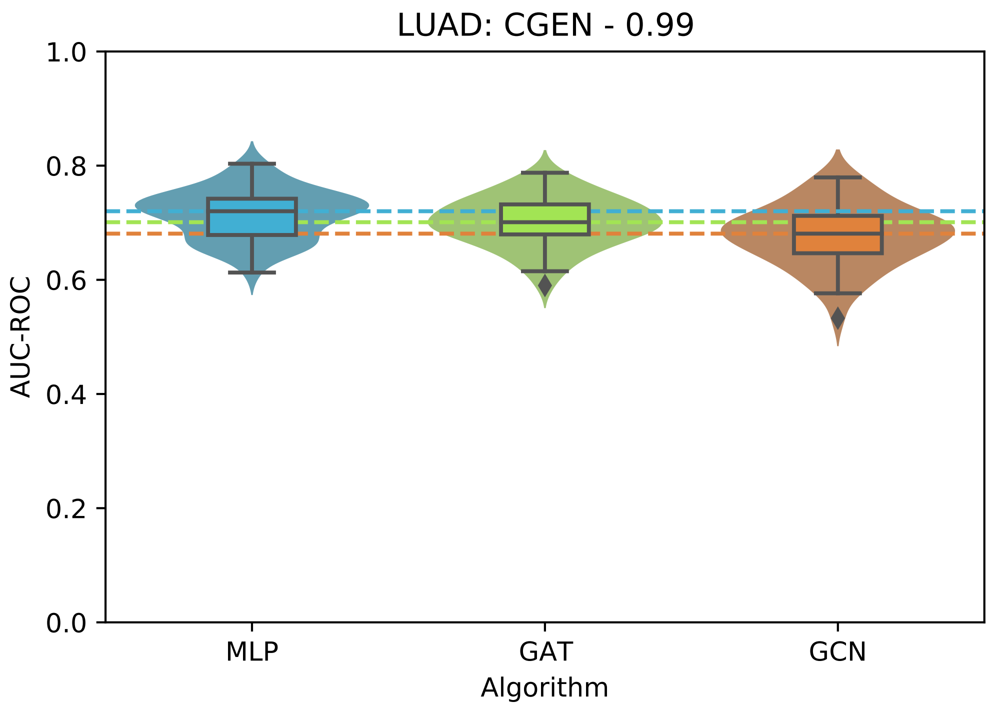


Figure A.18 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the CGEN network generation strategy and the 95% threshold.

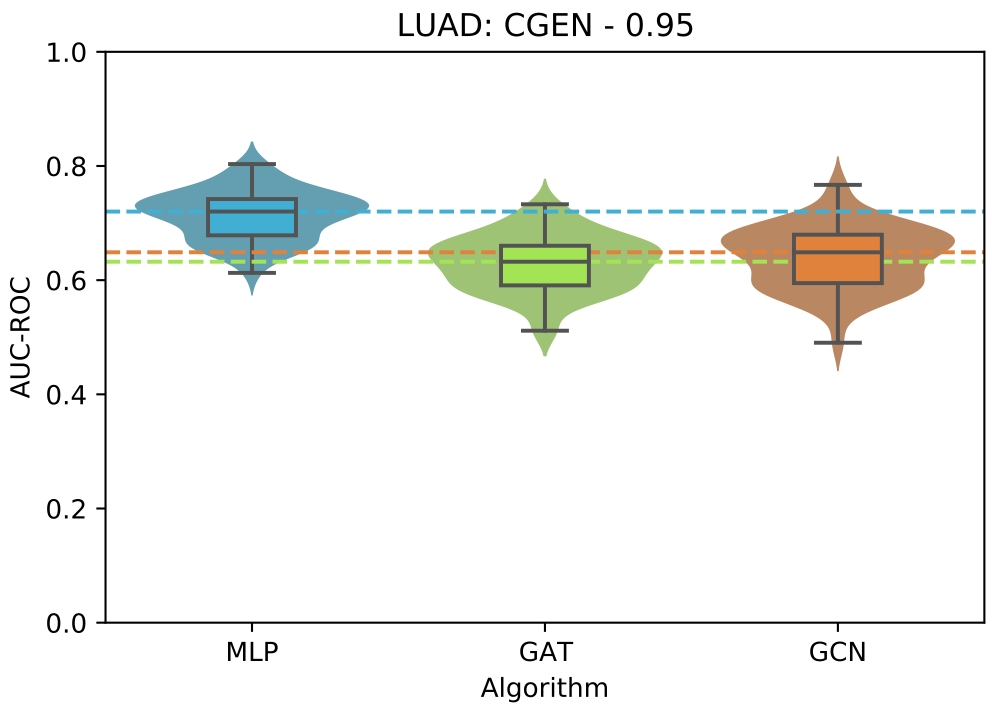


Figure A.19 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 99% threshold.

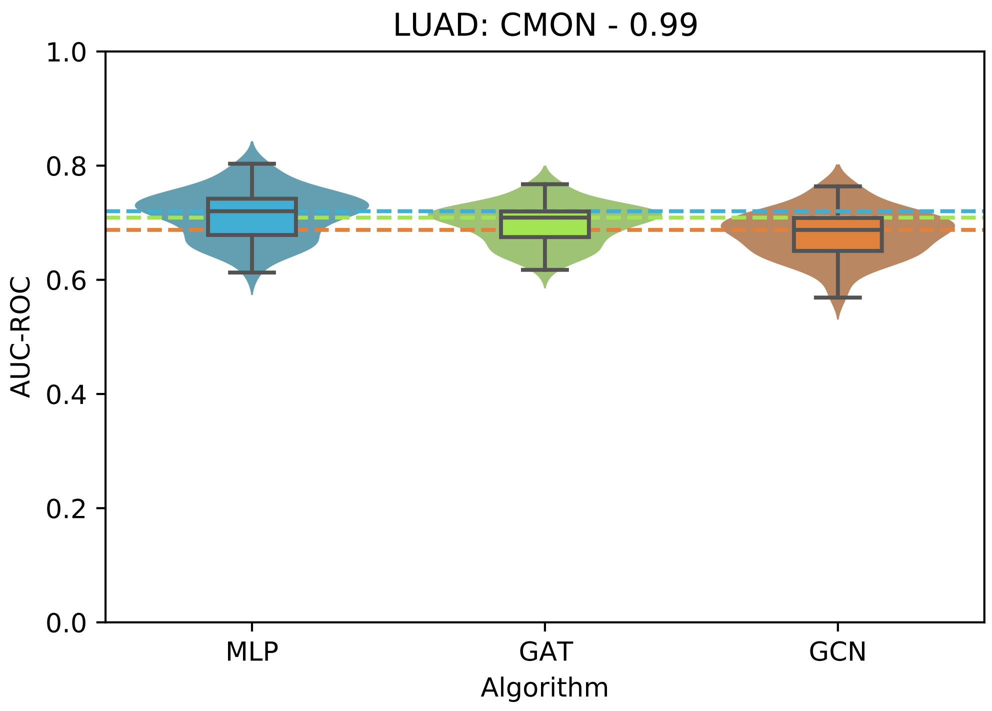


Figure A.20 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the CMON network generation strategy and the 95% threshold.

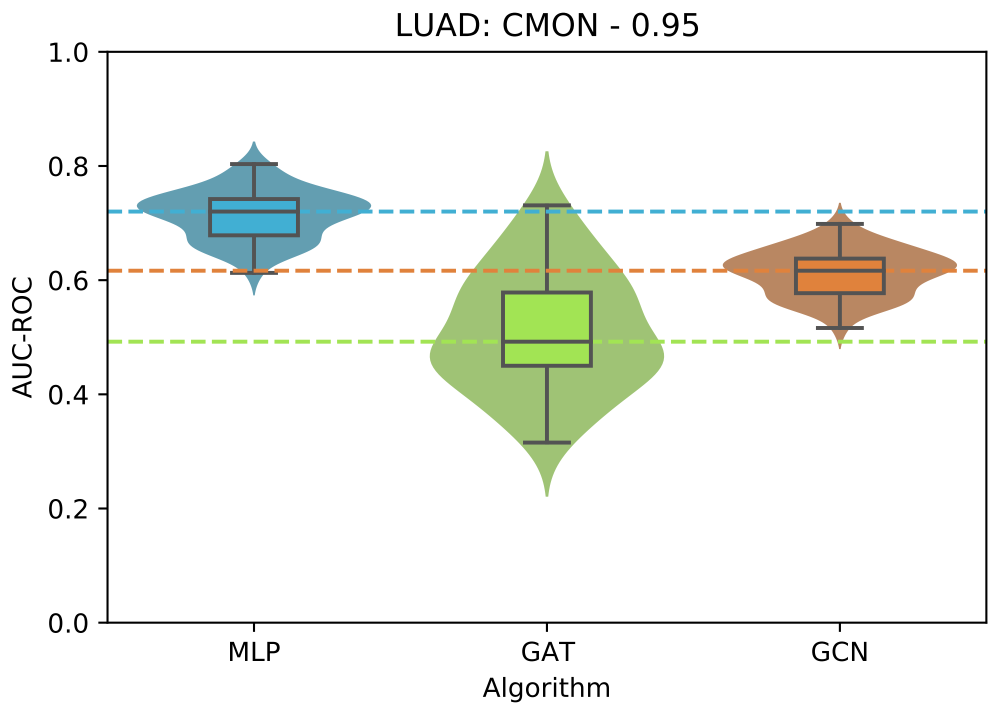
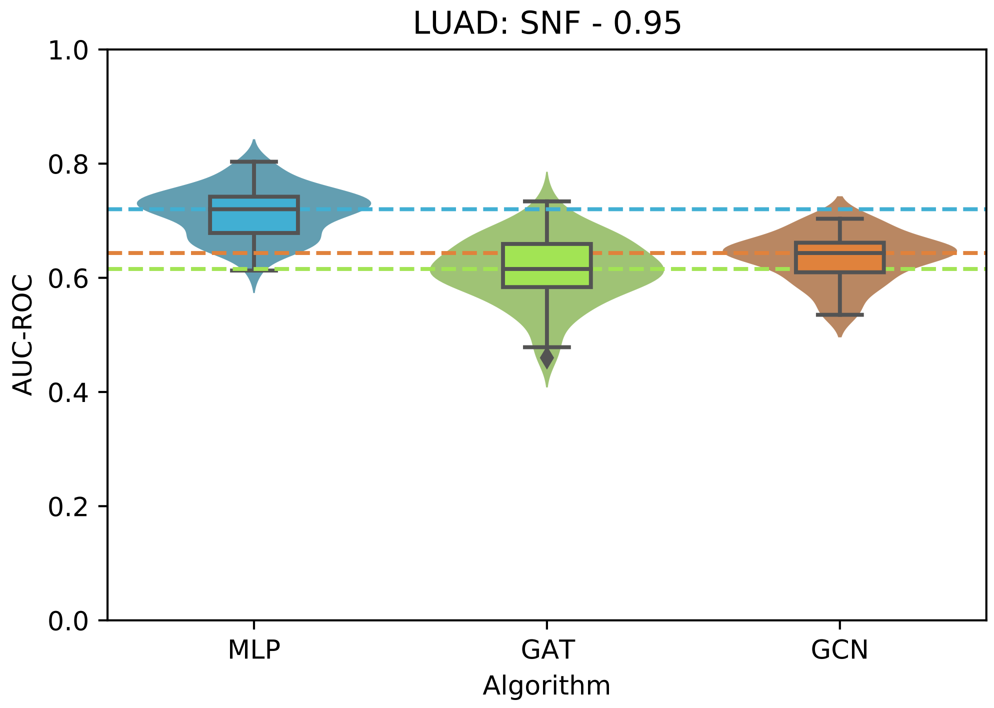


Figure A.21 – AUC-ROC performance comparison on LUAD dataset between the MLP approach and the GNNs using the SNF network generation strategy and the 95% threshold.



A.3 Networks visualization

Figure A.22 – KIRC patient similarity network generated using the CMON strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

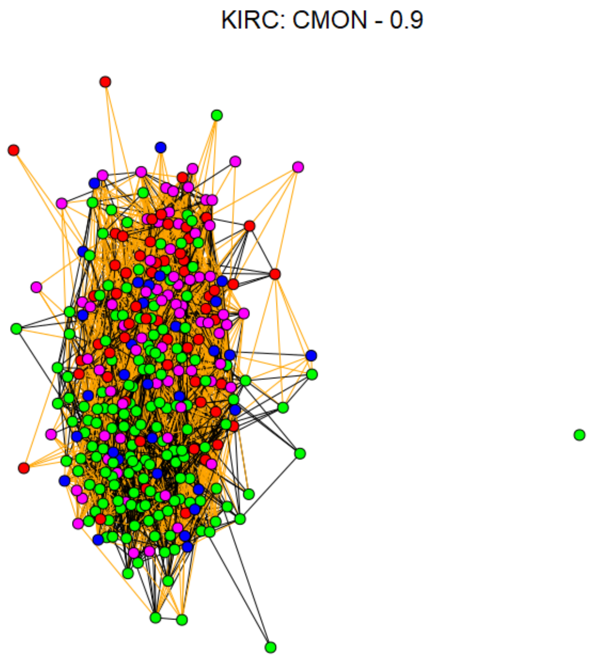


Figure A.23 – KIRC patient similarity network generated using the CMON strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

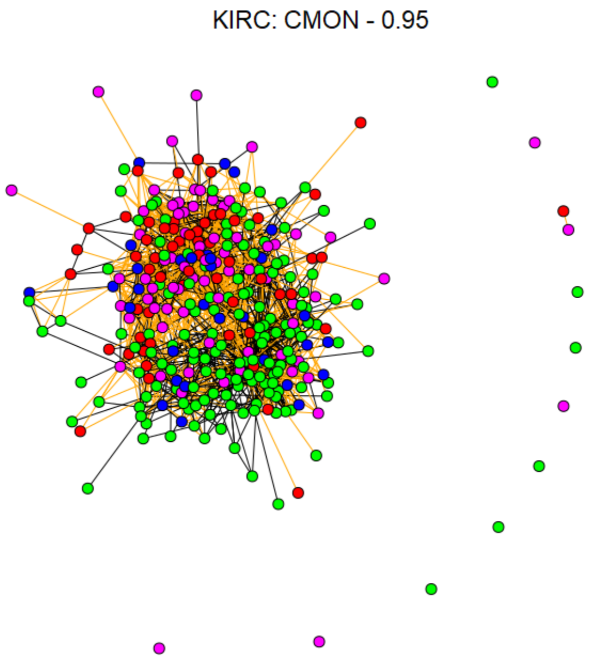


Figure A.24 – KIRC patient similarity network generated using the CMON strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

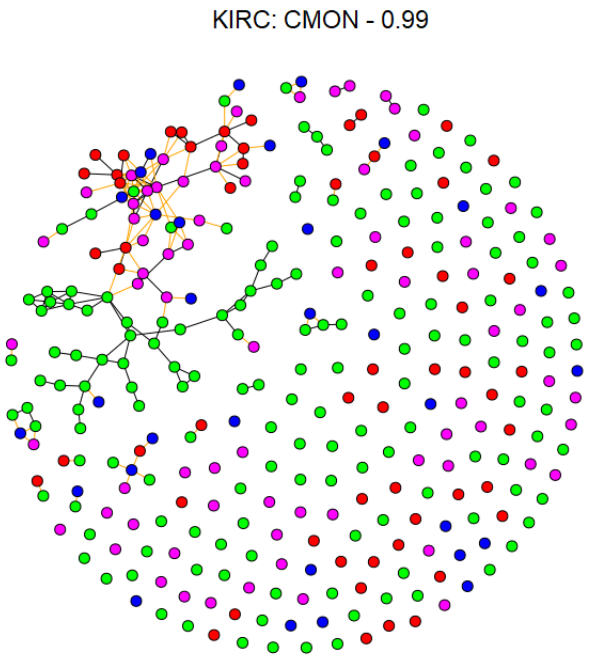


Figure A.25 – KIRC patient similarity network generated using the SNF strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

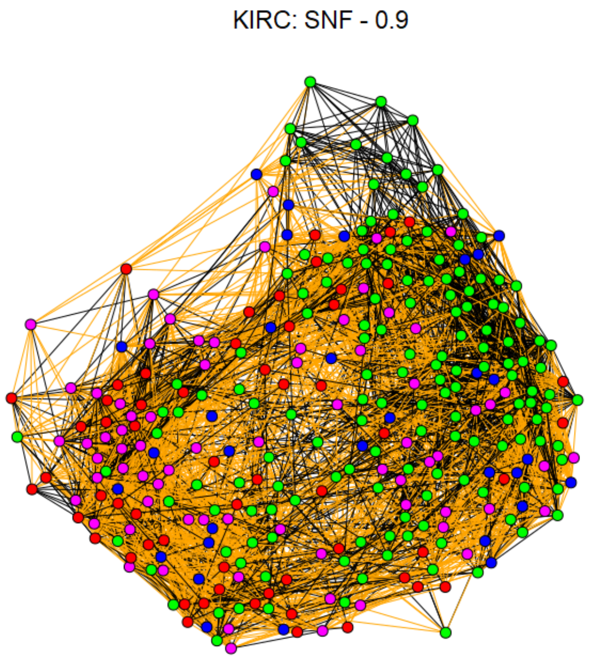


Figure A.26 – KIRC patient similarity network generated using the SNF strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

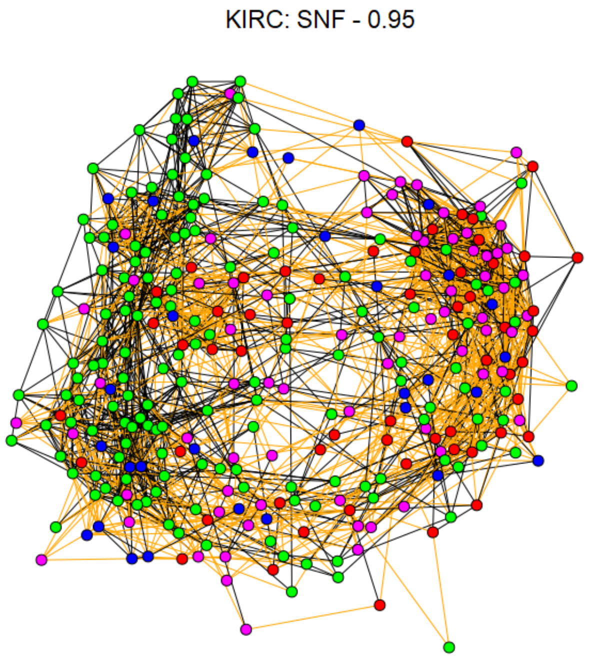


Figure A.27 – KIRC patient similarity network generated using the SNF strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

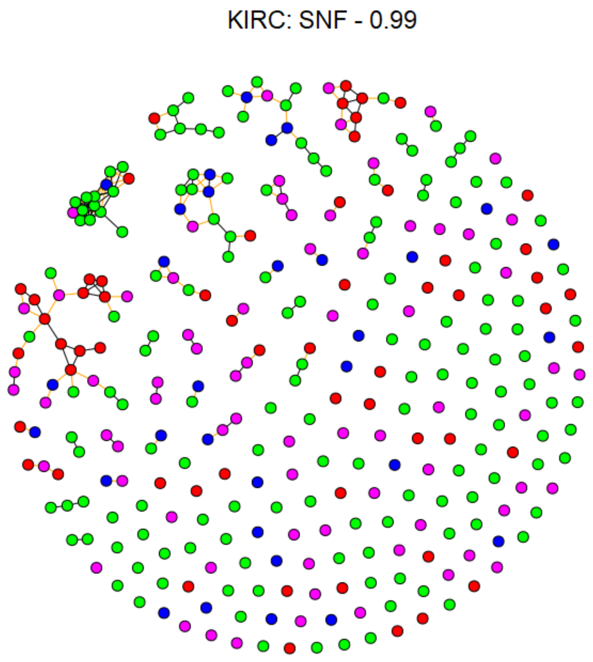


Figure A.28 – COAD patient similarity network generated using the CGEN strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

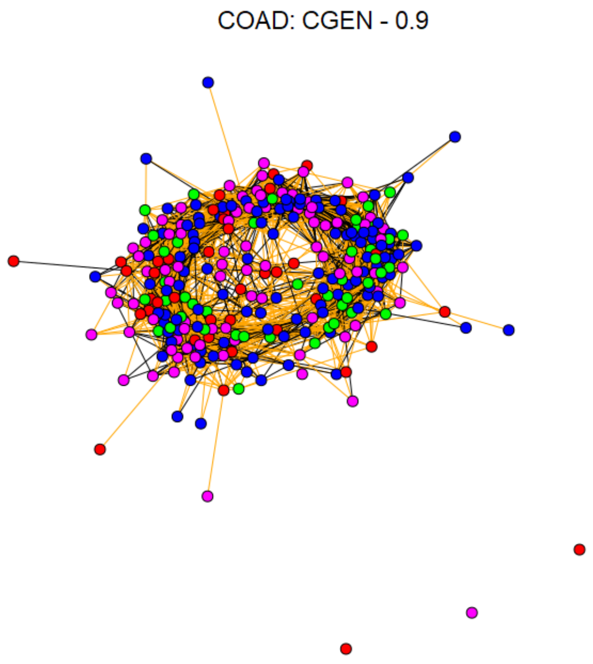


Figure A.29 – COAD patient similarity network generated using the CGEN strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

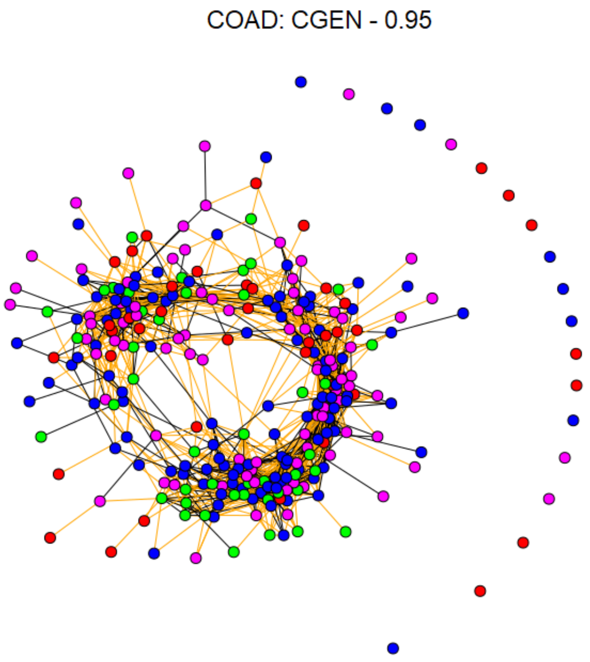


Figure A.30 – COAD patient similarity network generated using the CGEN strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

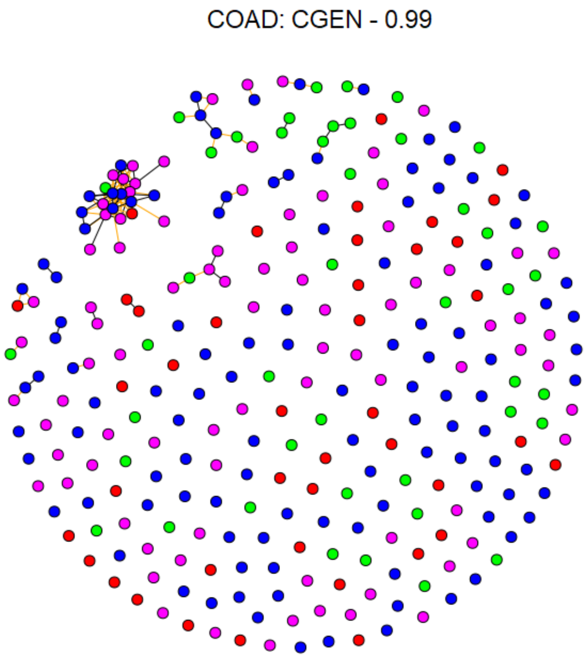


Figure A.31 – COAD patient similarity network generated using the CMON strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

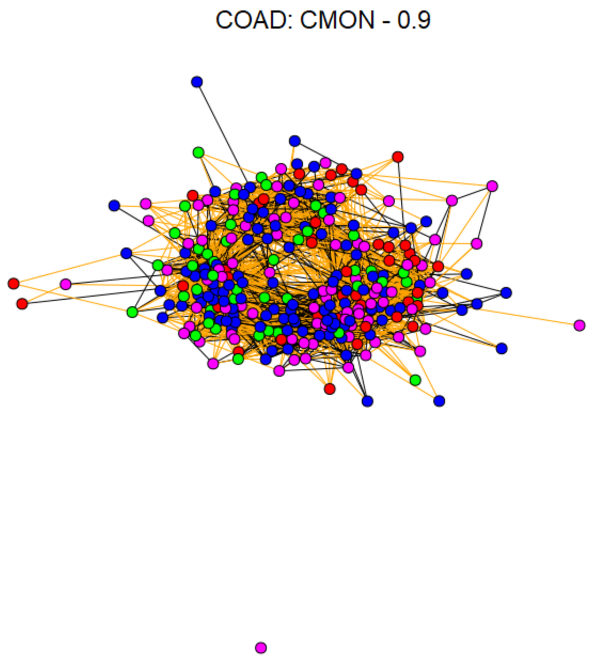


Figure A.32 – COAD patient similarity network generated using the CMON strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

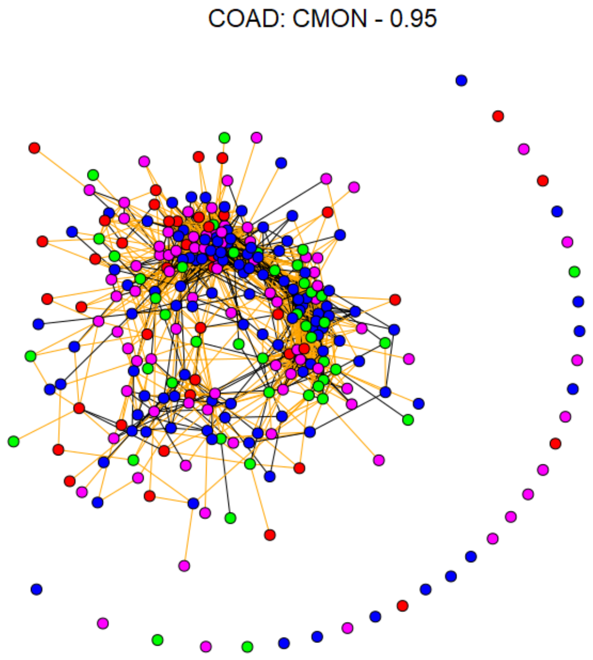


Figure A.33 – COAD patient similarity network generated using the CMON strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

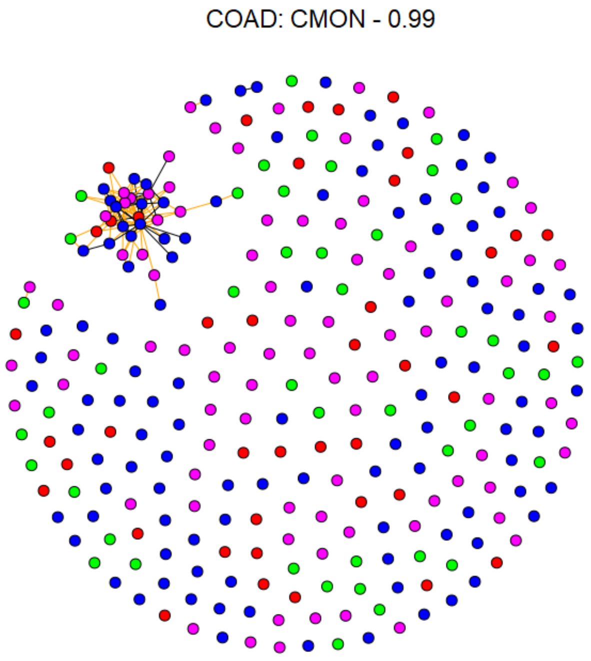


Figure A.34 – COAD patient similarity network generated using the SNF strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

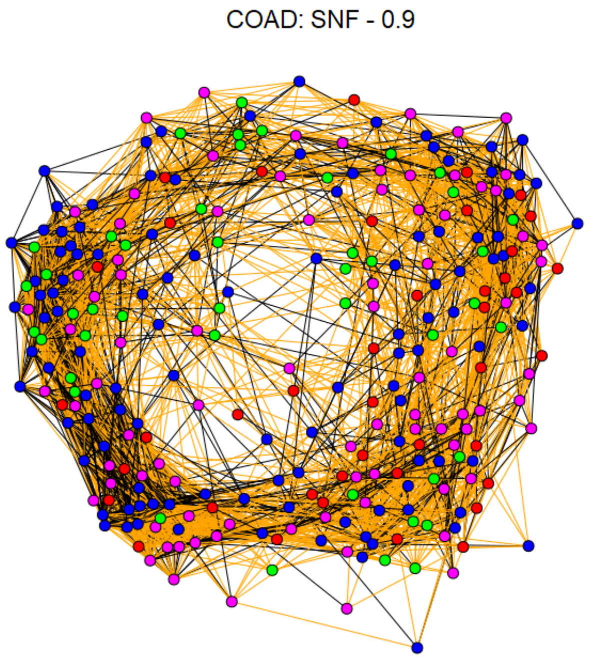


Figure A.35 – COAD patient similarity network generated using the SNF strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

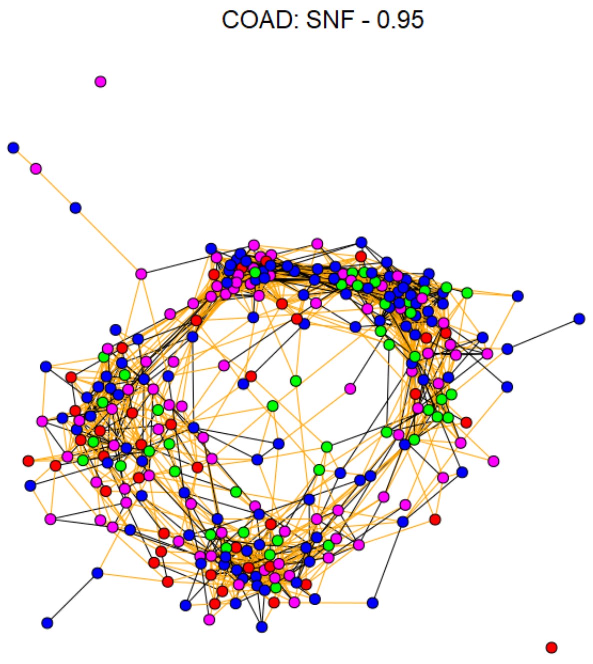


Figure A.36 – COAD patient similarity network generated using the SNF strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

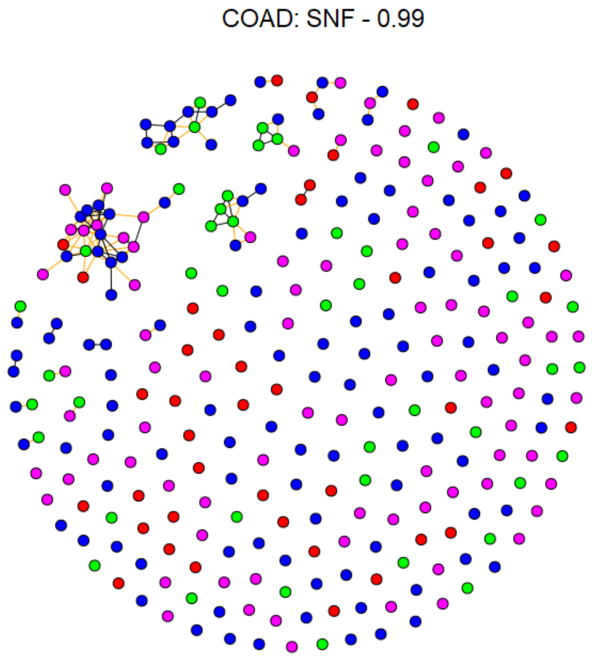


Figure A.37 – LUAD patient similarity network generated using the CGEN strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

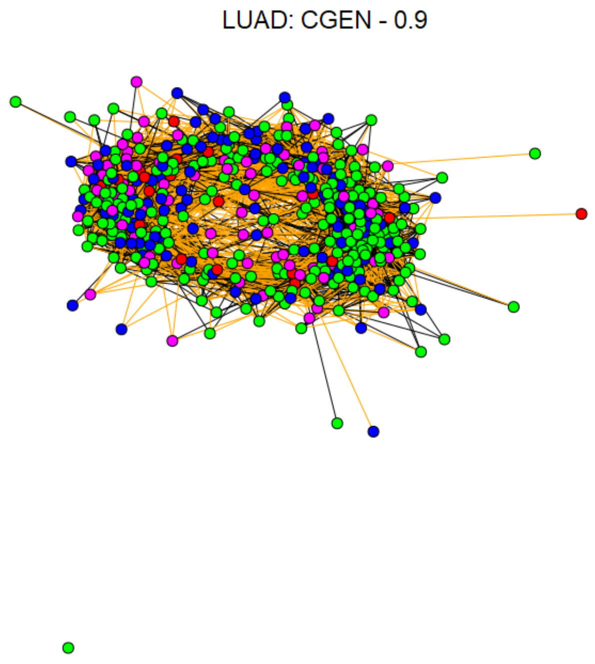


Figure A.38 – LUAD patient similarity network generated using the CGEN strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

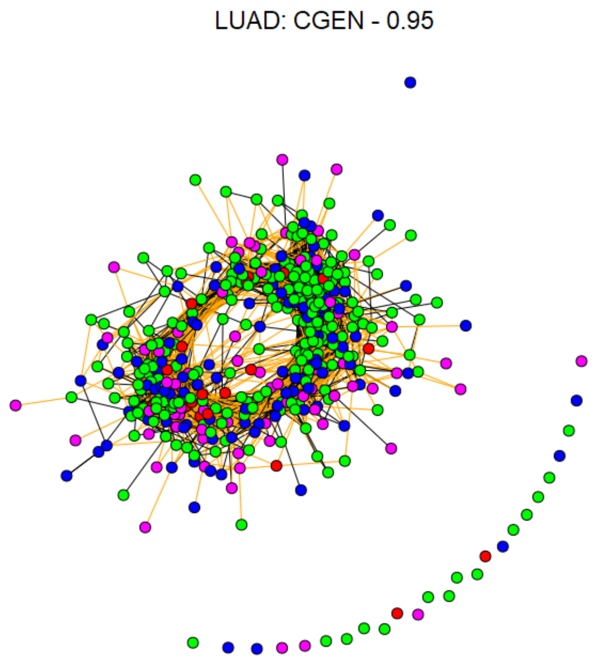


Figure A.39 – LUAD patient similarity network generated using the CGEN strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

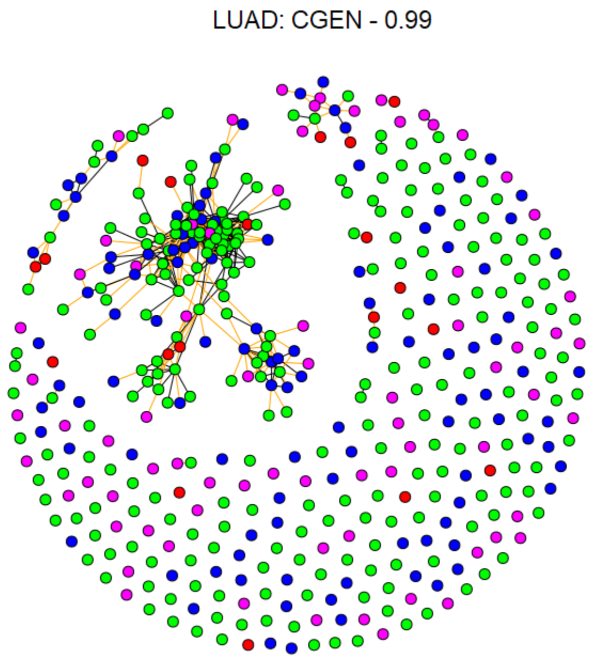


Figure A.40 – LUAD patient similarity network generated using the CMON strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

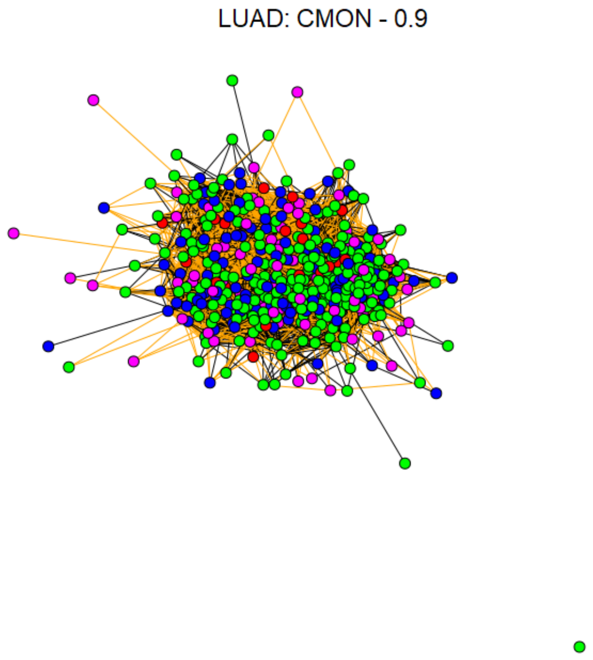


Figure A.41 – LUAD patient similarity network generated using the CMON strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

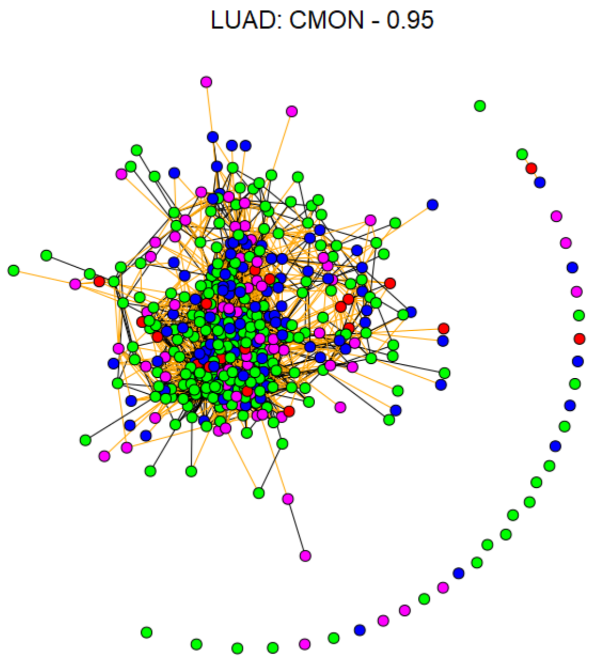


Figure A.42 – LUAD patient similarity network generated using the CMON strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

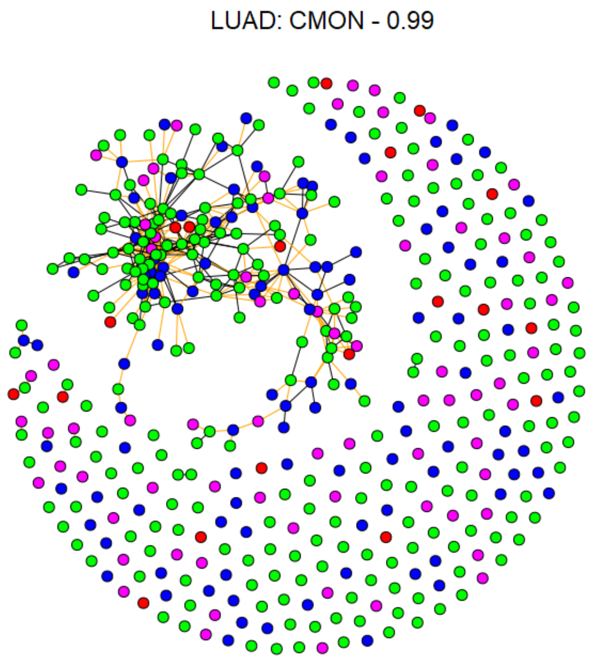


Figure A.43 – LUAD patient similarity network generated using the SNF strategy with a threshold of 90%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

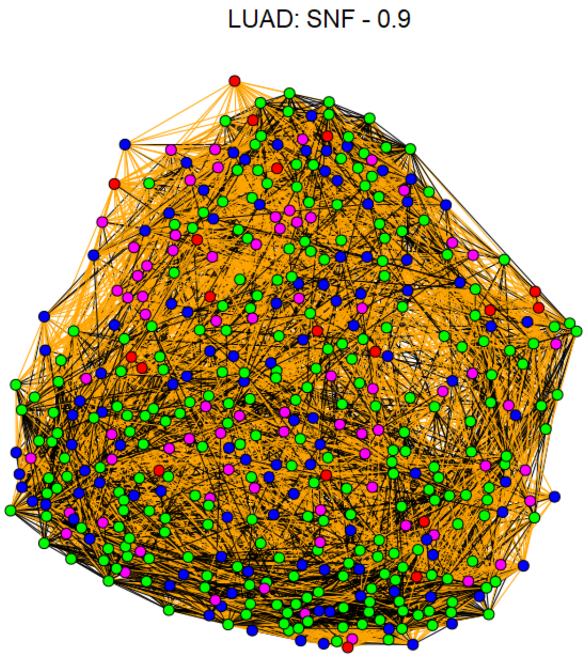


Figure A.44 – LUAD patient similarity network generated using the SNF strategy with a threshold of 95%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.

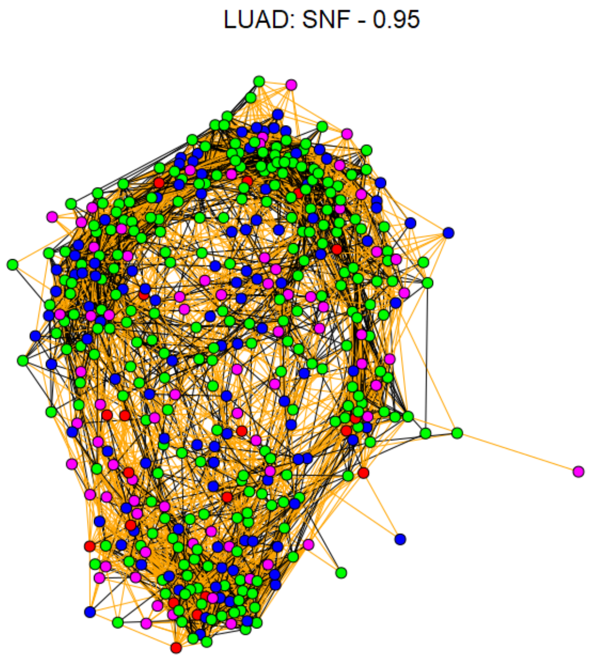
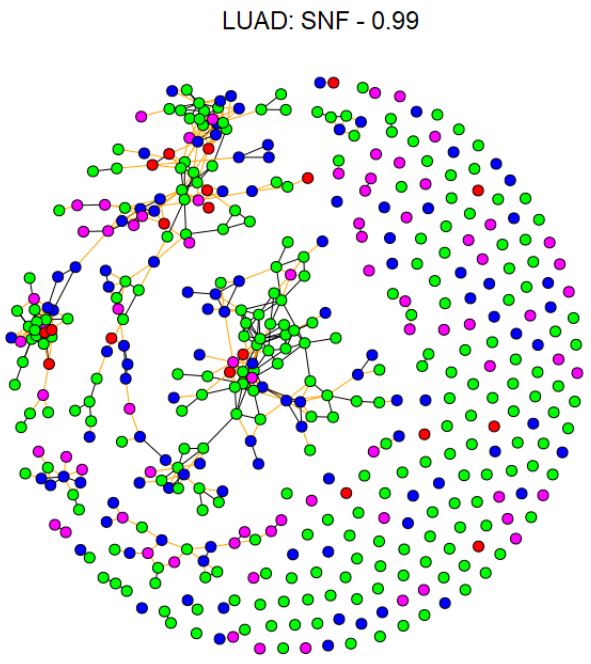


Figure A.45 – LUAD patient similarity network generated using the SNF strategy with a threshold of 99%. Black edges represent a connection between nodes of the same class, and orange edges represent a connection between nodes of a different class. Node colors represent patients' cancer stage: green represents stage I; blue, stage II; magenta, stage III; red, stage IV.



APPENDIX B — SUPPLEMENTARY TABLES

B.1 Networks summary

Table B.1 – Number (and percentage) of nodes connected only with themselves in the KIRC networks for each generation strategy and threshold.

KIRC	**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	0	0	0	0	0	0	0	13 (4%)	179 (57%)
CMON	0	0	0	0	0	0	1 (0%)	10 (3%)	180 (57%)
SNF	0	0	0	0	0	0	0	0	163 (52%)

Table B.2 – Number (and percentage) of nodes connected only with themselves in the COAD networks for each generation strategy and threshold.

COAD	**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	0	0	0	0	0	0	3 (0%)	19 (7%)	210 (74%)
CMON	0	0	0	0	0	0	1 (0%)	30 (11%)	239 (85%)
SNF	0	0	0	0	0	0	0	2 (0%)	202 (71%)

Table B.3 – Number (and percentage) of nodes connected only with themselves in the LUAD networks for each generation strategy and threshold.

LUAD	**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**
CGEN	0	0	0	0	0	0	1 (0%)	26 (6%)	259 (58%)
CMON	0	0	0	0	0	0	1 (0%)	31 (7%)	257 (58%)
SNF	0	0	0	0	0	0	0	0	169 (38%)