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

RENAN SOARES DE ANDRADES

Prediction of Cancer Driver Genes with Graph Neural Networks: A Comparative Analysis and a Graph Convolutional Network-based Model

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

Advisor: Prof[a]. Dr[a]. Mariana Recamonde-Mendoza

Porto Alegre
May 2023

CIP — CATALOGING-IN-PUBLICATION

Andrades, Renan Soares de

Prediction of Cancer Driver Genes with Graph Neural Networks: A Comparative Analysis and a Graph Convolutional Network-based Model / Renan Soares de Andrades. – Porto Alegre: PPGC da UFRGS, 2023.

173 f.: il.

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

1. Cancer driver genes. 2. Prediction model. 3. Machine learning. 4. Bioinformatics. 5. Graph-based learning. 6. Graph neural networks. I. Recamonde-Mendoza, Mariana. II. Título.

“No tenemos miedo a las ruinas, llevamos
un mundo nuevo en nuestros corazones.
Y ese mundo está creciendo en este instante.”

— Buenaventura Durruti

AGRADECIMENTOS

Devo minha mais profunda gratidão aos meus pais, Rosângela e Paulo Ricardo, pelo amor e apoio que depositaram em mim ao longo de toda minha jornada acadêmica, por acreditarem em mim e encorajarem minhas decisões. Um punhado de caracteres jamais chegariam perto de mensurar o quanto vocês significam para mim, mas que estas palavras aqui publicadas fiquem à eternidade registradas que eu amo vocês.

Quero agradecer também aos meus irmãos, Victor e Vinícius, pelo carinho e pelo forte laço de amizade. Vocês dois talvez nem imaginem o quanto foram e ainda são importantes para mim, torço muito pelo sucesso de vocês pois são pessoas inestimáveis.

Sou grato também aos meus avós pelo amor que têm comigo, e aos meu tios Oraci Junior e Andrea, que sempre torcem muito por mim, que me apoiaram incondicionalmente no começo desta minha trajetória, e que são pessoas valorosas que contribuíram muito para que eu seja o que eu sou hoje.

Sou imensamente grato à minha orientadora Mariana Recamonde Mendoza, pela dedicação, ensinamentos e apoio ao longo da minha pesquisa. Uma professora excepcional que sempre engrandeceu meu trabalho e soube motivar sempre que algo parecia difícil demais para ser vencido. Agradeço seu carinho e paciência comigo ao longo desta jornada, a senhora é uma inspiração de profissional que tenho um carinho enorme.

Gostaria de agradecer a todos que passaram e contribuíram de alguma forma ao longo desta etapa, meu muito obrigado a todos por fazerem parte da minha vida e por me ajudarem a alcançar este marco significativo.

ABSTRACT

Identifying cancer driver genes (CDGs) is crucial for improving the understanding of cancer biology and developing effective diagnostic and treatment strategies. However, accurately identifying CDGs from a vast array of somatic mutations remains a challenge despite the substantial amount of genomic data available. Recent developments in graph-based machine learning (ML) methods, such as Graph Neural Networks (GNNs), have made them powerful tools for analyzing protein-protein interaction (PPI) networks and performing predictions at the node level of biological networks. However, the use of GNNs for identifying candidate CDGs is still underexplored. This study aims to explore the predictive power of GNNs and develop a practical approach for predicting CDGs by integrating PPI networks and multi-omics data across several cancer types. We investigate data-centric and algorithmic decisions involved in model training to understand the potential of GNNs for this prediction task and to identify a robust methodology for classifying genes as cancer-causing or neutral in 16 types of cancer. Three primary decision levels are addressed: (i) node feature definition, (ii) class imbalance mitigation, and (iii) choice of the learning algorithm. We extensively analyze different GNN models trained through a semi-supervised approach, using six different PPI networks and four types of omics data: single nucleotide variant, copy number variation, DNA methylation, and gene expression. These models are contrasted with the performance achieved by traditional ML algorithms using regular structured data for model development. Following the experimental comparative analysis, we explore ensemble learning strategies and hyperparameter tuning to improve the predictive power of the top-performing model. Our results demonstrate that GNNs outperform traditional ML approaches in predicting CDGs, and that adding node centrality measures as node features improves learning outcomes even for graph-based learning methods. We also highlight the significant contribution of ensemble learning methodologies in improving performance metrics by aggregating predictions of models trained on multiple PPI networks. Finally, using the proposed approach, we provide predictions for unlabeled genes regarding their potential role as CDGs. Overall, this study provides relevant insights into using GNNs to predict CDGs and highlights Graph Convolutional Networks as an effective algorithm for this task.

Keywords: Cancer driver genes. prediction model. machine learning. bioinformatics. graph-based learning. graph neural networks.

Predição de genes causadores de câncer com Graph Neural Networks: uma análise comparativa e um modelo baseado em Graph Convolutional Networks

RESUMO

Identificar os genes causadores de câncer (CDGs, de *cancer driver genes*) é crucial para melhor compreender a biologia do câncer e desenvolver estratégias eficazes de diagnóstico e tratamento. No entanto, a identificação precisa de CDGs a partir de uma vasta gama de mutações somáticas continua sendo um desafio, apesar da quantidade substancial de dados genômicos disponíveis. Desenvolvimentos recentes em métodos de aprendizado de máquina (AM) baseados em grafos, como *Graph Neural Networks* (GNNs), tornaram-se ferramentas poderosas para analisar redes de interação proteína-proteína (PPI) e realizar previsões em nível de nós. No entanto, o uso de GNNs para identificar CDGs candidatos ainda é pouco explorado. Este estudo visa explorar o poder preditivo de GNNs no contexto de predição de CDGs, desenvolvendo uma abordagem prática baseada na integração de redes PPI e dados multi-ômicos em vários tipos de câncer. Investigamos decisões centradas em dados e algorítmicas envolvidas no treinamento de modelos para entender o potencial de GNNs para essa tarefa de predição e para identificar uma metodologia robusta para classificar genes como CDGs ou neutros em 16 tipos de câncer. Três níveis primários de decisão são abordados: (i) definição de atributos do nó, (ii) mitigação do desequilíbrio de classes e (iii) escolha do algoritmo de aprendizado. Analisamos extensivamente diferentes modelos de GNNs treinados por meio de uma abordagem semi-supervisionada, usando seis redes PPI diferentes e quatro tipos de dados ômicos: variantes de nucleotídeos únicos, variação do número de cópias, metilação do DNA e expressão gênica. Esses modelos são comparados com o desempenho alcançado pelos algoritmos tradicionais de AM, treinados sobre dados estruturados regulares. Após a análise comparativa experimental, exploramos estratégias de aprendizado *ensemble* e ajuste de hiperparâmetros para melhorar o poder preditivo do modelo de melhor desempenho. Nossos resultados demonstram que as GNNs superam as abordagens tradicionais de AM na previsão de CDGs e que a adição de medidas de centralidade de nós como atributos dos nós no grafo melhora os resultados de aprendizado, mesmo para métodos de aprendizado baseados em grafos. Também destacamos a contribuição significativa das metodologias de aprendizado *ensemble* na melhoria das métricas de desempenho, agregando previsões de modelos treinados em várias redes PPI. Finalmente, usando a abordagem proposta, fornecemos previsões

para genes não marcados em relação ao seu papel potencial como CDGs. No geral, este estudo fornece informações relevantes sobre o uso de GNNs para prever CDGs e destaca as Redes Convolucionais de Grafos como um algoritmo eficaz para esta tarefa.

Palavras-chave: Genes causadores de câncer, modelo preditivo, aprendizado de máquina, bioinformática, aprendizado baseado em grafos, rede neural de grafo.

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
ACC	Accuracy
ANN	Artificial Neural Network
AUC-PR	Area Under the Precision-Recall Curve
AUC-ROC	Area Under the ROC Curve
CE	Cross-Entropy
CV	Cross-Validation
CDG	Cancer Driver Gene
CNA	Copy Number Alteration
CNN	Convolutional Neural Network
CNV	Copy Number Variation
CPDB	Consensus Path DB
DL	Deep Learning
DT	Decision tree
DNA	Deoxyribonucleic Acid
FL	Focal loss
GAT	Graph Attention Network
GBT	Gradient Boosting Tree
GCN	Graph Convolutional Network
GNN	Graph Neural Network
HGNC	HUGO Gene Nomenclature Committee
IARC	International Agency for Research on Cancer
ICGC	International Cancer Genome Consortium
ML	Machine Learning

MLP	Multilayer Perceptron
mRNA	Messenger RNA
OG	Oncogene
OMIM	Online Mendelian Inheritance in Man
PPI	Protein-protein interaction
PREC	Precision
RF	Random forests
REC	Recall
RNA	Ribonucleic acid
ROC	Receiver Operating Characteristic
SNV	Single Nucleotide Variant
TSG	Tumor Supressor Gene
TSS	Transcription Start Site
TCGA	The Cancer Genome Atlas

LIST OF FIGURES

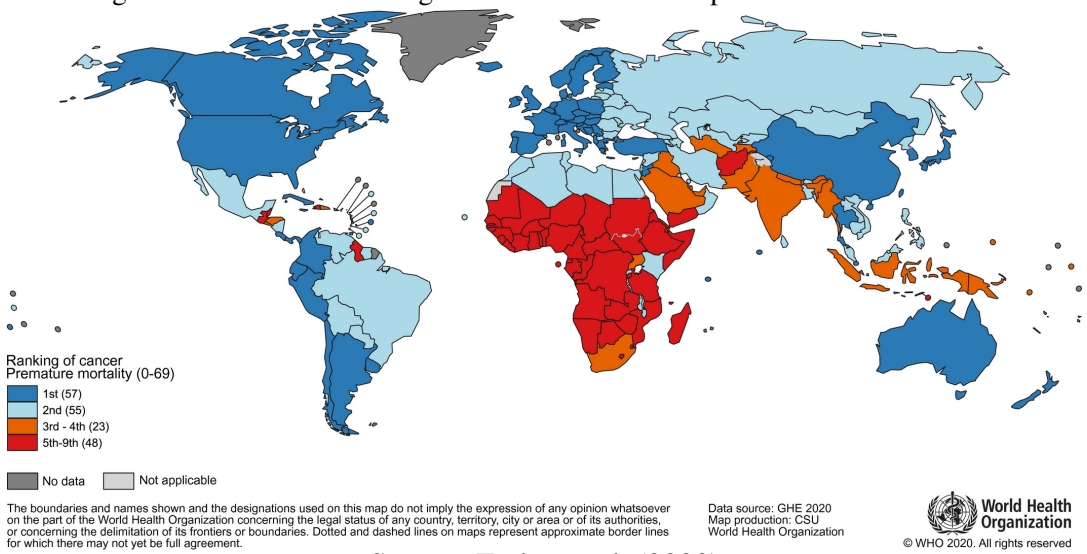
LIST OF TABLES

CONTENTS

1 INTRODUCTION

Cancer is a complex and heterogeneous disease that arises from a combination of genetic and environmental factors. According to the International Agency for Research on Cancer (IARC)[1], approximately one in five individuals worldwide develop cancer during their lifetime. Unfortunately, the incidence and mortality of cancer are rapidly increasing (FERLAY et al., 2020), with the GLOBOCAN 2020 projecting 28.4 million new cases of cancer globally by 2040, which represents a 47% increase from 2020 (BRAY et al., 2018). Moreover, cancer is a leading cause of deaths across the globe, accounting for nearly 10 million deaths in 2020. A recent study showed that cancer was the first or second cause of premature death (*i.e.,* under the age of 70) in 112 of 183 countries analyzed, and ranked third or fourth in other 23 countries (FERLAY et al., 2020), as shown in Figure 1.1. Thus, early detection and treatment of cancer lesions are crucial in reducing cancer mortality rates and the burden of this major public health problem.

Figure 1.1 – National ranking of cancer as a cause of premature death in 2019.



Source: Ferlay et al. (2020)

As a multifactorial disease, several factors may impact cancer risk. The risk factors include environmental and lifestyle factors, such as UV radiation, air pollution, cigarette smoking, hormone therapy, physical activity, and dietary patterns (WU et al., 2018). Nonetheless, cancer is largely the result of acquired genetic and epigenetic changes that culminate in abnormal and uncontrolled cell growth (TOMASETTI; VOGELSTEIN, 2015; ANANDAKRISHNAN et al., 2019). Somatic mutations, which are genetic alterations acquired during an individual's lifetime, are the most significant contributors to

[1] <https://www.iarc.who.int/>

tumorigenesis (MARTÍNEZ-JIMÉNEZ et al., 2020). Specifically, the somatic mutations that confer a selective growth advantage to a tumor cell, called "driver" mutations, reside in a subset of genes known as cancer driver genes (CDGs) (STRATTON; CAMPBELL; FUTREAL, 2009). However, the majority of somatic mutations are neutral or "passenger" mutations that have no impact on cancer initiation and progression (STRATTON; CAMPBELL; FUTREAL, 2009; VOGELSTEIN et al., 2013). Thus, identifying all genes carrying mutations that can drive carcinogenesis is a major goal in cancer research.

Distinguishing between driver mutations and passenger mutations is a challenging task, despite the increasing availability of cancer genomic datasets. The heterogeneity of mutation rate and mutation types across tumors pose the need for rigorous analytical methods that can effectively deal with these issues and also handle the natural challenges of genomic data analysis (MARTÍNEZ-JIMÉNEZ et al., 2020). Various computational approaches have been proposed to improve the detection of driver mutations and CDGs from genomics data. Machine learning (ML) algorithms have been particularly successful in predicting CDGs, as noted in a recent survey (ANDRADES; RECAMONDE-MENDOZA, 2022). However, advanced deep learning (DL) techniques are gaining momentum in the field by enabling the construction of end-to-end models that can handle complex data.

Among DL methods, there has been a growing interest in the use of graph-based learning algorithms for prediction tasks in bioinformatics due to their ability to learn patterns from graph-structured data modeling the complex relationship and manifold interactions among biological entities (GEORGOUSIS; KENNING; XIE, 2021). Specifically, graph neural networks (GNN) are emerging as key tools since they have shown promise in learning in the presence of irregular geometric shapes commonly found in biology, such as in protein-protein interaction (PPI) networks. Moreover, these algorithms allow integrating node-level features into the learning process, creating rich node embeddings that encompass information from nodes' properties and local structure in the graph.

However, there is limited exploration of GNNs for predicting CDGs across different cancer types. Existing research has shown the potential of GNNs to integrate PPI networks and high-throughput omics data, such as transcriptomic and epigenetic data from healthy and tumor tissues, to identify CDGs more accurately (SCHULTE-SASSE et al., 2019; SCHULTE-SASSE et al., 2021). Nevertheless, several data-centric strategies in GNN-based models' development, such as node feature vectors, node centrality measures, and class imbalance mitigation methods, remain unexplored. Additionally, pre-

vious works have not investigated the concept of ensemble learning alongside GNNs for CDGs prediction. Therefore, further research is needed to determine the effectiveness of these strategies in improving CDGs prediction.

In light of this, the present study aims to propose an effective approach for predicting cancer driver genes (CDGs) by exploring graph neural network (GNN) algorithms that integrate protein-protein interaction (PPI) networks and multi-omics data. Using a semi-supervised approach, we conducted a thorough investigation towards data-centric and algorithm-centric decisions during model training to better understand the potential of GNNs and identify a robust methodology capable of classifying genes as cancer drivers or passengers across 16 cancer types. Moreover, we selected one top-performing GNN algorithm, namely Graph Convolution Network (GCN), and carried out an extensive hyperparameter optimization process with six distinct PPI networks, comparing the performance among these models and also against ensemble-based approaches proposed in our work. Our contributions can be summarized as follows:

- Evaluation of three GNN algorithms and four traditional ML algorithms for three PPI networks in controlled experiments;
- Comparative analysis among different node feature vectors defined from four types of omics data;
- Investigation of the impact of including node centrality measures as node features in the performance of GNN models;
- Comparison among three strategies to handle class imbalance in GNNs;
- Proposal and evaluation of ensemble-based GCN approaches for CDG prediction.

The remainder of the work is organized as follows: Chapter 2 provides a theoretical foundation for the biological and computational aspects underlying this work. Chapter 3 conducts a comprehensive and critical literature review, which has also been published as a survey paper in the *Briefings in Bioinformatics* journal (ANDRADES; RECAMONDE-MENDOZA, 2022). Chapter 4 details the materials and methods employed in the study, for both the comprehensive experimental investigation and the development of GNN-based prediction models for CDGs. The results are presented in two chapters. Chapter 5 reports a series of experiments to compare algorithms and data strategies for CDGs prediction, and Chapter 6 provides a detailed analysis of fine-tuned GCN-based models for this task. Finally, the conclusions and perspectives for future works are summarized in Chapter 7.

2 THEORETICAL BACKGROUND

This chapter provides a theoretical background with the main concepts needed for the understanding of this work. The chapter is divided into two parts: the first part provides a brief biological summary that aims to situate the reader into the research problem, its relevance, and types of data involved; the second part presents an overview of the computational aspects of the work, describing the algorithms and techniques employed.

2.1 Biological concepts

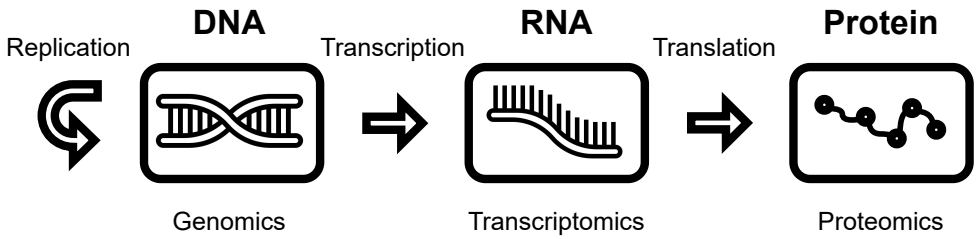
Human cells contain a complete set of molecular instructions for life that are encoded in deoxyribonucleic acid (DNA) molecules (ALBERTS et al., 2002). DNA serves as the storage material for genetic information and is the chemical structure that transmits these instructions from one generation to the next. Each DNA molecule is composed of a sequence of nucleotides that are linked end to end, forming a long chain. These nucleotides are represented by a single-letter code that stands for the nitrogen-containing base they carry: adenine (A), guanine (G), cytosine (C), and thymine (T).

The human genome, which is made up of a relatively small number of long DNA molecules arranged into structures called chromosomes, contains specific regions called genes that are known as the functional units of DNA (SIMPSON, 2022). Genes play a crucial role in encoding the instructions necessary to produce functional ribonucleic acid (RNA) molecules (noncoding RNA genes) or proteins (protein-coding genes) (SALZBERG, 2018). The production of functional products from instructions carried by a gene is a process known as gene expression, and was formalized more than 50 years ago, when Francis Crick enunciated the central dogma of molecular biology explaining the flow of genetic information within a biological system (CRICK, 1970).

The first step of gene expression occurs when the genetic information encoded in the DNA is used to produce an RNA molecule through a transcription process (Figure 2.1). When this molecule is a messenger RNA (mRNA), it can be decoded into a protein in a second step known as translation. Although, initially, only proteins were recognized as the final functional product of gene expression, nowadays it is known that noncoding RNA molecules may also play a vital role for organisms functioning and diseases, being thus recognized as functional products of noncoding genes. Besides transcription and translation, the central dogma of molecular biology also includes a DNA replication pro-

cess, in which two identical replicas of DNA are produced from a single DNA molecule.

Figure 2.1 – Central dogma of molecular biology



Source: Prepared by the author

Proper gene function is essential for life, but when the gene sequence or activity (*i.e.,* gene expression) is disrupted by biological or environmental factors, it can lead to diseases. Genomic instability, which refers to an increased propensity for changes in the genome during the life cycle of cells, is influenced by genetics to varying degrees, as noted by Shen (2011). Although all diseases are to some extent influenced by the genetic constitution of individuals, making them more or less susceptible to disorders, it is challenging to identify the genetic changes associated with a particular disease due to the vast number of variations between human genomes (JACKSON et al., 2018). Moreover, a disease may result from multiple variants in the DNA sequence involving one or a few nucleotides and also from environmentally induced heritable changes in gene expression that do not result from a change in the DNA sequence (known as epigenetic dysregulation).

One disease that arises from genetic and epigenetic dysregulation is cancer, a condition characterized by uncontrolled cell growth and metastasis. As there are many potential causes of cancer, the field of molecular biology is continually developing genome sequencing and computational tools to better understand cancer phenotypes and the underlying genotypic or molecular changes (SANT'ANNA et al., 2018). The next sections review relevant concepts related to the cancer genome, including the genomic changes related to the development of the disease and the types of genome-wide biological data useful for their investigation.

2.1.1 The cancer genome

Cancer is a group of complex diseases caused by genetic changes that result in abnormal and uncontrolled cell growth (STRATTON; CAMPBELL; FUTREAL, 2009). The genetic changes may be due to inherited genetic variation or acquired somatic mu-

tations, that is, DNA alterations that occur after conception. I classical tumorigenesis model, cancer is described as multiple and successive clonal expansions driven by the accumulation of genomic alterations that are preferentially selected by the tumor environment (YATES; CAMPBELL, 2012). These alterations may include various forms of mutations on specific genes, amplifications, deletions or rearrangements of chromosome segments, copy number changes that cause gain or loss of an entire chromosome(s), among others (SHEN, 2011). As a consequence, the functioning of protein-coding genes or regulatory components of the genome can be affected, particularly with regard to the control of cell growth and division (STRATTON; CAMPBELL; FUTREAL, 2009). Ultimately, cancer cells acquire the ability to invade neighboring tissues and metastasize, which makes cancer such a dangerous disease.

Since the first sequencing of the cancer genome in 2008, research has shown promising potential to advance prevention, diagnosis, prognosis, and treatment by promoting a better understanding about the fundamental biology of cancer (MWENIFUMBO; MARRA, 2013). An important advance was the description of several features common to most cancer cells that affect a handful of essential cellular functions and contribute to abnormal and uncontrolled cell growth, referred to as “cancer hallmarks”.

According to Hanahan (2022), these hallmarks comprise “the acquired capabilities for sustaining proliferative signaling, evading growth suppressors, resisting cell death, enabling replicative immortality, inducing/accessing vasculature, activating invasion and metastasis, reprogramming cellular metabolism, and avoiding immune destruction”.

However, genome-wide analyses of tumors also revealed that cancer genome is characterized by a significant heterogeneity between and even within tumor types (VOGELSTEIN et al., 2013). The mutation load in cancer is abundant and complex - although many variations may be found between different genomes, not all of them are implicated in cancer. Moreover, it has been observed that most common cancers are associated with several genetic mutations that occur at low frequencies (YATES; CAMPBELL, 2012). Thus, one of the great challenges of cancer research is precisely discovering the “drivers” of the disease, such as mutations that induce tumorigenesis, and the genes whose mutated forms affect the normal functioning of a ser of essential cellular processes - known as “cancer driver genes” (CDGs) (MARTÍNEZ-JIMÉNEZ et al., 2020).

Cancer driver genes can be of two types based on their role in cancer progression: (i) oncogenes (OGs), which promote uncontrolled cell growth and division, or (ii) tumor supressor genes (TSGs), which prevent the cell from undergoing uncontrolled division

(CHANDRASHEKAR et al., 2020). Therefore, the abnormalities in the cell cycle processes that control the tumor formation and development are due to genomic alterations that cause gain-of-function of OGs together with the loss-of-function of TSGs.

2.1.2 Driver and passenger mutations

As discussed in the previous section, the task of characterizing and understanding somatic genomic alterations and their relation with disease represents a significant challenge in cancer biology. According to Stratton, Campbell and Futreal (2009), the DNA sequence of most human cells acquire a set of differences from its progenitor fertilized egg through somatic mutations. Somatic mutations in cancer genome may include distinct classes of DNA sequence changes (STRATTON; CAMPBELL; FUTREAL, 2009):

- **Single nucleotide variant (SNV)** refers to a simple substitution of a single nucleotide for another. When this point mutation affects a coding region but does lead to amino acid substitution, the SNV is referred to as synonymous (or silent) mutation. When the nucleotide substitution alters the encoded amino acid, the SNV is called a nonsynonymous mutation. Nonsynonymous mutations can be further classified into missense or nonsense, depending on whether the nucleotide substitution results in the codon[1] sequence for a different amino acid that affect protein function in varying degrees (missense), or in a stop codon (nonsense) that generates truncated, incomplete, and nonfunctional protein products.
- **Insertion or deletions (indels)** refer to a small length of DNA (usually less than 50 base pairs) that is inserted into or deleted from the genome. When the length of the indel is not a multiple of three (*i.e.,* the length of a codon), this causes a frameshit that may drastically alter the mRNA sequence and, thus, the product of RNA translation.
- **Copy number alteration (CNA)** refers to gain or loss of large (greater than 50 base pairs) DNA sequences due to genomic rearrengements such as deletions or duplications. CNAs may encompass gene sequences, resulting in gene duplication or deletion. It is sometimes also referred to as copy number variation (CNV).

Occasionally, due to the mutations accumulated over the lifetime, a single cell may gain the capability to proliferate and survive more effectively than its neighbours.

[1] Codon is a sequence of three consecutive nucleotides that codes for a specific amino acid.

The so-called “driver” mutations can confer cells the advantage of growing uncontrollably, thus driving tumorigenesis. Although it is straightforward to define a cancer driver gene as one that carries a driver mutation that provides a selective growth advantage to a cell, a plethora of somatic abnormalities present in the cell is not necessarily linked to the cancer initiation or progression. When a mutation does not provide an advantage in clonal growth and therefore does not contribute to cancer development, it is referred to as a “passenger” mutation (VOGELSTEIN et al., 2013). In summary, a driver mutation induces uncontrolled cell proliferation and tumor growth, while a passenger mutation does not (STRATTON; CAMPBELL; FUTREAL, 2009). Since passenger mutations frequently arise during cell division, they are also present in cancer genomes, but they have no functional impact.

Therefore, the key analytical challenge in the study of cancer genome is to differentiate driver mutations from passenger mutations. Several computational strategies have emerged with the increasing availability of mutation profile data for thousands of cancer patients through initiatives such as The Cancer Genome Atlas (TCGA) or more recently, the International Cancer Genome Consortium (ICGC) (HIRANO et al., 2020). The vast amount of genomic data generated in the last years and the development of more powerful computational methods provide an excellent opportunity to predict cancer driver genes from molecular data. The next section introduces the main types of datasets explored in the prediction of cancer driver genes.

2.1.3 Genome-wide biological data

The identification of cancer drivers has been greatly advanced by high-throughput technologies that allow genome-wide measurements of molecular components that comprise human biological systems in any level - from DNA to proteins. This big data approach in the context of biology has generated a new field of “omics” science, which refers to the extensive quantification (*i.e.,* profiling) of molecular-level information spanning the whole cell or tissue of an organism (LI; CHEN, 2014). Depending on the nature of the molecular component being measured, a different type of omics data is generated: genomics, transcriptomics, epigenomics, proteomics, etc. Moreover, because every component of the human system behave as part of a highly interconnected and dynamic network, multifactorial diseases like cancer are the result of disruptions in several biological and pathological processes interacting in a complex network (BARABÁSI; GULBAHCE;

LOSCALZO, 2011). Thus, one way to interconnect different omics data is through biological networks, which are theoretical models that aim to represent the inherent complexity of biological systems (LIU et al., 2020). In what follows, we review the types of omics data employed in the current work and the concept of protein-protein interaction networks. Whereas here we focus in the theoretical aspects of the biological data, in Chapter 4 we explain in more detail how the datasets were collected and pre-processed to be more informative for a computational prediction method.

2.1.3.1 Omics data

Since the completion of the Human Genome Project in 2003, high-throughput technologies have continued to progress and improve the rate and quality of biological data collection (ROBINSON; NIELSEN, 2016). Nowadays, genome-scale measurements often replace molecule-scale measurements, allowing for investigation of the entire complement of a specific type of molecule that composes biological systems, in various levels of interrogation (KARCZEWSKI; SNYDER, 2018). The so-called "omics" data are crucial for elucidating the genomics of cancer and provide an unprecedented understanding of the molecular features underlying tumors. The Cancer Genome Atlas (TCGA) and the PanCancer Atlas have generated and analyzed omics profiling for more than 11,000 tumor samples across 33 cancer types. Among the several types of existing omics data (KARCZEWSKI; SNYDER, 2018), this work is interested in the following:

- **Genomics**: genome sequencing assays help to determine the complete DNA sequence of organisms and to identify DNA-level changes between cancer and healthy samples. It is through genomics that different types of mutations can be characterized in their totality, such as SNVs and CNAs.
- **Transcriptomics**: DNA microarrays and high-throughput sequencing technologies provide the ability to measure the gene expression for the entirety of genes in an organism at the transcriptional level. Although mRNA is not the final functional product of a gene, but rather an intermediary step in gene expression, the information about transcription levels is necessary for understanding the regulatory mechanisms controling gene expression and for finding dysregulated genes associated with diseases (BRAZMA; VILO, 2000). In transcriptomics, the complete set of RNA transcripts (*i.e.,* transcriptome) is quantified and may comprise both coding RNAs (*i.e.,* mRNA) and noncoding RNAs.

- **Epigenomics**: refers to the study of heritable changes in the genome that are not caused by DNA sequence mutations but still influence gene expression. The most common epigenetic mechanism is DNA methylation. DNA methylation involves the transfer of a methyl group to the C5 position of cytosine to form 5-methylcytosine (MOORE; LE; FAN, 2013). DNA methylation is considered an important regulator of gene expression, causing gene silencing by recruiting proteins involved in gene repression or inhibiting the binding of the transcription factor (KULIS; ESTELLER, 2010).

2.1.3.2 Protein-protein interaction networks

Proteins are important building components of cells and carry out the majority of their functions. However, proteins do not function independently within a biological system; they interact with other functional components in a complex network to maintain the stability of the internal environment of cells (KOH et al., 2012). Following this principle, the phenotypic impact of a disease is determined by the network context of a mutated or disregulated gene, including not only the gene itself and its function through the encoded functional products, but also its interaction partners (BARABÁSI; GULBAHCE; LOSCALZO, 2011).

There are various types of biological networks that differ in the types of macromolecules and interactions they describe. Generally, the macromolecule is represented as a "node" of the network and a large number of interactions, whether physical, biochemical, or functional, are represented as "edges" between nodes. This modeling technique provides an excellent understanding of subcellular systems on a global scale by representing the human interactome (LIU et al., 2020). In this study, we focus solely on protein-protein interaction (PPI) networks.

PPI networks offer a valuable framework for comprehending the functional organization of the proteome, and has been frequently applied in the study of human conditions such as cancer, infectious diseases, and neurodegenerative diseases (LU et al., 2020). They map physical links between two or more proteins, which are represented by the nodes of the biological network (LIU et al., 2020). To map human PPIs, there are a variety of experimental strategies, such as the yeast two-hybrid assay (Y2H), which measures direct physical interactions in cells, and affinity purification mass spectrometry, which measures the composition of protein complexes. This powerful tool can be applied in a high-throughput manner to detect these interactions in a genome-wide fashion,

although at high costs (STELZL et al., 2005). Additionally, literature-curation may be employed to systematically collect interactions from thousands of published small-scale studies targeting a single or few specific hypotheses (DAS; YU, 2012).

2.2 Computational concepts

Advancements in cancer genomics projects have generated massive amounts of data, providing opportunities to leverage computational approaches for more accurate prediction of cancer driver genes. A number of computational strategies have been proposed in literature, exploring several types of biological data. In Chapter 3, we provide a comprehensive review of related works, summarizing these efforts under the perspective of computational methods and data employed. This section aims to present theoretical concepts that are important for building a prediction model, including the learning algorithms and the evaluation metrics adopted.

2.2.1 Machine learning and learning paradigms

While various methods exist for prediction tasks, it is imperative to address the fundamental concept of machine learning. Machine learning (ML) is an artificial intelligence (AI) branch that allows computer algorithms to learn patterns from dat a (NAQA; MURPHY, 2015). The primary goal of ML is to develop a predictive model based on statistical associations between features of a given dataset. Through this approach, ML algorithms can be explored to learn an optimal mapping between genomic features and the capacity of a given mutation or mutated gene to drive tumorigenesis. Once these predictive patterns of cancer-driving genomic variations are learned, the model can be applied in classification tasks such as discerning if a somatic mutation is a driver mutation or a passenger mutation, or if a given gene is responsible for or does not contribute to cancer.

Considering the scope of classification tasks, ML-based models are trained to specify which of k categories some input sample belongs to. The models are generated based on a given set of input data (*i.e.,* training data) that usually consists of "features" and "labels" across a set of samples. Features are the measurements across all samples, raw or mathematically transformed, while labels are the categories the model aims to predict, *i.e.,* the output of the model (CAMACHO et al., 2018). When the labels of the

training data are known and the model aims to estimate the unknown mapping between input (*i.e.,* features) and output (*i.e.,* label) data, the ML algorithm is said to follow a supervised learning approach. The opposite of supervised learning is the unsupervised learning approach, in which only input samples are provided to the learning system and the labels are unknown.

In some situations, the labels are incomplete, that is, only a portion of the training data was previously labeled with the correct or expected outputs. Semi-supervised learning deals with problems in which the data is partially labeled. A common approach consists in using the labeled part to infer labels for the unlabeled part, and then proceed with a traditional supervised learning using the complete set of (now labeled) data (NAQA; MURPHY, 2015). According to Camacho et al. (2018), the use of semi-supervised learning may help surpass the model performance that can be achieved using only the labeled data (conducting a fully supervised learning), or ignoring the labels and considering all samples as unlabeled data in an approach known as unsupervised learning. In Chapter 4, we discuss in more detail how semi-supervised learning represents an interesting perspective for predicting cancer driver genes, since many human genes still lack proper annotation for their role in tumorigenesis.

The following sections describe specific ML algorithms commonly used for classification tasks, where the model output is in the form of categories. It is worth noting that we use an organization commonly seen in the literature, grouping algorithms into traditional machine learning and Deep learning (DL) methods. Moreover, we highlight algorithms that are important for the experiments conducted in the upcoming chapters.

2.2.2 Traditional machine learning algorithms

In this section, we present some traditional ML algorithms that are commonly explored in bioinformatics and are mostly aimed at supervised learning (KOTSIANTIS; ZAHARAKIS; PINTELAS, 2006). Traditional ML relies on manual feature engineering to craft features for the model. Thus, as complex as they may seem, they still demand domain expertise and human intervention. In what follows, we review support vector machine, tree-based learning algorithms, and artificial neural networks.

2.2.2.1 Support Vector Machine

Support Vector Machine (SVM) is a powerful method for classification tasks that creates a decision boundary between two classes to predict the labels of new samples, which are comprised of one or more features. This decision boundary is a hyperplane that is oriented to be as far as possible from the closest data points of each class, which are known as the support vectors (HUANG et al., 2018b).

One of the main advantages of SVM is its ability to handle non-linearly separable data using a kernel function to transform the low-dimensional space into a higher-dimensional feature space in which the problem becomes linearly separable (MAMMONE; TURCHI; CRISTIANINI, 2009). This makes it easier to find a decision boundary that separates the classes. In essence, the kernel function is a mathematical trick that enables SVM to transform lower dimensional data into higher dimensional data with the help of new features created, and separate the points in a higher dimension.

While there are different variations and extensions of the SVM algorithm, they all maintain the main properties that characterize the algorithm, such as the separation hyperplane, maximum margin hyperplane, and kernel function. These features allow SVM to find linear constraint problems without local maxima efficiently (MAMMONE; TURCHI; CRISTIANINI, 2009).

2.2.2.2 Tree-based learning algorithms

Tree-based methods are a popular class of supervised learning algorithms for classification and regression tasks. These algorithms have the structure of a decision tree (DT), which divides the data recursively based on the most discriminative features and assigns a label to each region of the input feature space. Decision trees are more interpretable than other classifiers because they combine simple questions about the data in an understandable way. Other advantages of decision trees include scalability, flexibility, and less requirements regarding data preparation (KOTSIANTIS, 2013).

Complex decision trees are prone to overfitting and, in this case, may exhibit poor generalization performance on new data. To address this challenge, pre-pruning and post-pruning techniques can be employed to control the growth of decision trees. Pre-pruning halts tree growth when there is insufficient data while post-pruning removes subtrees with inadequate data after tree construction. Nevertheless, to further address this problem, several variants of decision trees have been proposed, such as random forests and gradient

boosting (KINGSFORD; SALZBERG, 2008).

Random forests (RF) consist of a set of decision trees, each trained on a random subset of the training data and a random subset of the features. Random samples are generated independently and with the same distribution for all trees in the forest (BREIMAN, 2001). That is, the training set is is created based on sampling with replacement through a bootstrap process to create a modified training set of the same size as the original. Furthermore, during the construction of each tree, only a small, randomly selected subset of the available resources is considered when choosing the best split at each node (KINGSFORD; SALZBERG, 2008). To obtain the final prediction, the algorithm aggregates the predictions of all individual trees using the majority vote (BREIMAN, 2001). The main advantages of the random forest algorithm over a single decision tree include built-in randomness that reduces overfitting and increases the diversity of individual trees. This leads to a more accurate and robust model.

Another widely used tree-based algorithm is Gradient Boosting Trees (GBT), which combines multiple weak learners into a single stronger model. The main difference between GBT and the previous techniques is that GBT applies optimization by repeatedly re-weighting the training examples to focus on the most problematic ones. This includes a learning procedure where the objective is to construct base learners so that they are maximally correlated with the negative gradient of the loss function associated with the entire model (*i.e.,* the set of base learners) (SAGI; ROKACH, 2018).

Specifically, in GBT, a sequence of regression trees is computed, where the final prediction is obtained by aggregating these predictions from all trees in an additive manner, with each added model trained to minimize the loss function. It is a powerful and versatile algorithm that can be customized and optimized for different problem domains (KINGSFORD; SALZBERG, 2008).

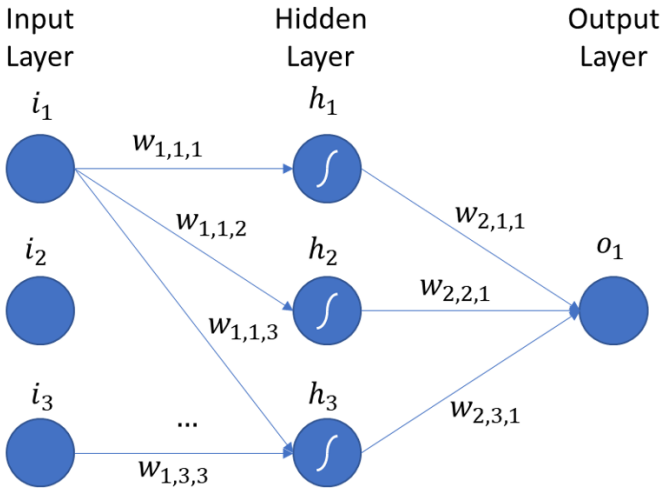
2.2.2.3 Artificial Neural Networks

Artificial Neural Networks (ANN) have become widely accepted as a powerful computational tool to solve complex real-world classification problems. ANNs are capable of modeling complex nonlinear problems, making them a popular choice in machine learning (ZOU; HAN; SO, 2009). A key property of ANNs is that they are inspired by early models of sensory processing in the human brain. Thus, just as the brain is made up of an elementary unit called a neuron, artificial neurons are the fundamental building blocks of all ANN models. In essence, an artificial neuron is a mathematical function

that takes in the linear combination of inputs, applies the function to them, and returns an output value (KROGH, 2008). This neural network unit is reffered to as the perceptron, which is also considered a single-layer neural network able to model only linear problems.

The most common architecture of ANNs is the Multilayer Perceptron (MLP), which is illustrated in Figure 2.2. Neurons are arranged in sequential fully connected layers. The leftmost layer in this network is called the input layer and the neurons within the layer are called input neurons. The rightmost or output layer contains the output neurons or, as in this case, a single output neuron. The intermediary layers between the input and the output ones are known as the hidden layers. All connections among nodes in the network have their corresponding weight represented by w in the figure. When designing the ANN topology, one must consider mainly , the number of hidden layers in the network and the number of nodes in each layer.

Figure 2.2 – Example of a Multilayer perceptron model.



Source: Trlin (2023)

Except for the nodes in the input layer, which propagate the original values of the input data for the subsequent layer, all other nodes implement a mathematical function that applies weights to the inputs and directs them through an activation function. These weights can be thought of as analogous to the strength of synapses in the brain. This mathematical function can be represented as follows:

$$y = \Phi\left(\sum_{i=1}^{N}(w_i x_i) + b\right) \tag{2.1}$$

where N represents the number of inputs to be processed, x_i represents a single variable or input feature, w_i represents a learnable weight for that input, b represents a learnable bias term, and Φ represents anactivation function that takes a linear combination of the

input values and returns a single output (ZOU; HAN; SO, 2009).

The activation function introduces non-linearity in the modeling process. While the sigmoid function was a traditional choice for early ANN models, other functions, such as the *rectified linear unit* (ReLU), are now commonly applied due to their superior performance (GOODFELLOW; BENGIO; COURVILLE, 2016). The ReLU activation function sets negative input values to zero, while passing input values that are equal to or greater than zero. This activation function allows faster learning compared to alternatives like the sigmoid function.

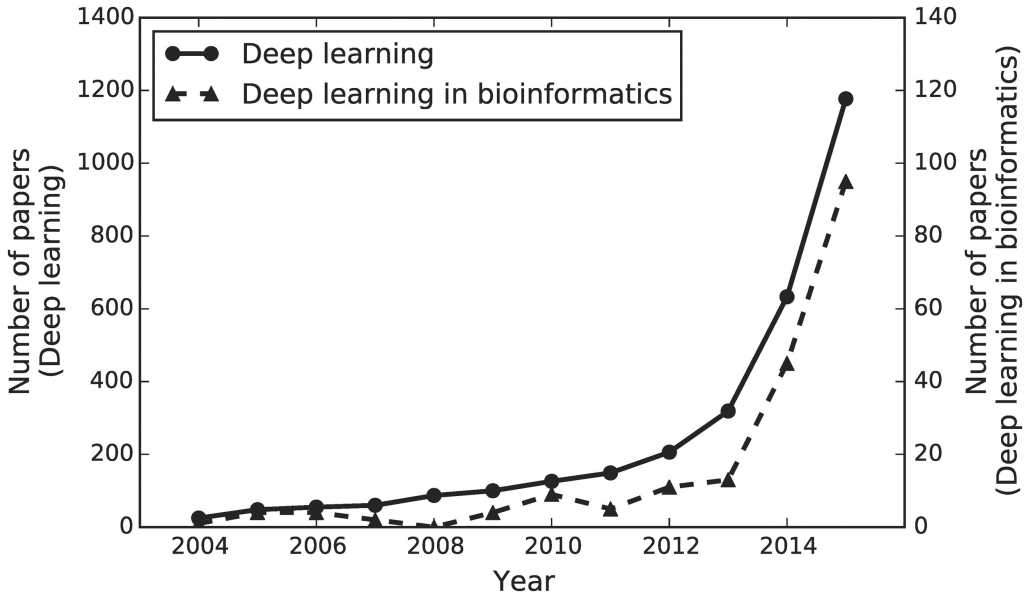
During the training process, the weights of an ANN are adjusted to minimize the total error between its output and the desired output (*i.e.,* the labeled data). These weights are typically initialized with small random values, and the training involves measuring the square difference between the ANN output and the desired output for each input, with the goal of minimizing the sum of these differences as much as possible. Backpropagation, a gradient-based optimization algorithm, is commonly used to adjust the weights by propagating the errors back through the network (KROGH, 2008).

2.2.3 Deep learning and graph-based learning

The field of bioinformatics is experiencing an increase in the accumulation of omics data through experimental studies. While traditional ML algorithms, such as ANNs, have been widely used to extract knowledge from this data, the emergence of big data has led to the development of more advanced and complex model architectures. Although these architectures are still based on ANNs, there has been a shift in focus towards deep learning (DL), which has shown promising results in achieving higher accuracy and performance compared to traditional ML algorithms. Consequently, DL has generated significant interest in academic and research communities. According to Min, Lee and Yoon (2017), this growth can be attributed to the advances made in deep learning, which have contributed to the growth of both sub-areas, as illustrated in Figure 2.3.

The success of deep learning can be attributed to the significant algorithmic detail involved in both constructing and training its architectures. Typically composed of multilayer artificial neural networks that leverage non-linear transformations, these architectures take on various forms depending on the characteristics of the input data and research objectives. Deep learning, as a subfield of ML, employs hierarchical architectures to learn high-level abstractions in data, enabling more accurate and sophisticated predictions and

Figure 2.3 – Number of published deep learning articles by year.



Source: Min, Lee and Yoon (2017)

insights to be made from the data (GOODFELLOW; BENGIO; COURVILLE, 2016).

One of the most representative network architectures in the field of deep learning is the convolutional neural network (CNN), which is designed to process data in matrix format. CNNs can be seen as a sequence of convolution layers and pooling layers. The convolution operation is implemented through the application of filters in the input. The filters are defined as a matrix of weights with the same values repeated in a specific order that is multiplied by the input. During this operation, specific characteristics are retained from the input while others are ignored, depending on the filter applied. The result of the convolution is usually passed through a non-linear activation function such as ReLU (LI et al., 2021; LECUN; BENGIO; HINTON, 2015). The convolution layer can be followed by a pooling layer that applies the poolin operation. In this step, groups of adjacent values in the input data (for example, groups of 2x2 pixels in an image) are aggregated by taking the maximum value among the (*i.e.,* Max Pooling) or the average among them (*i.e.,* Average Pooling). Pooling operations have the utility of intensifying important characteristics or features identified in the convolution layer.

Despite the effectiveness of CNNs in modeling regularly structured information, several domains produce data with irregular structure in non-Euclidean space. Graphs, which provide an effective representation of entities and their relationships, are among the many irregular geometric shapes commonly encountered in these domains (GEORGOUSIS; KENNING; XIE, 2021). Graphs are well-suited to modeling complex structures such as protein-protein interactions, which are natural networks found in the biolog-

ical domain, as discussed in Section 2.1.3.2. In what follows we provide basic definitions of graph structures and we introduce adaptations in deep learning algorithms to allow learning from graphs.

2.2.3.1 Basics of graph theory and node centralities

A graph is a mathematical structure comprising a set of vertices (or nodes) and edges that connect them. We represent a graph as a pair of vertices (V) and edges (E), denoted as $G = (V, E)$. An edge that links two vertices i and j may optionally have a weight $w_{ij} \in \mathbb{R}$ assigned to it. Depending on whether each edge has a weight assigned to it or not, a graph can be either weighted or unweighted. Likewise, a graph can be either directed or undirected, depending on whether its edges have a specific direction. For instance, protein-protein interactions can be represented as graphs, where nodes depict proteins and edges depict physical or functional connections among them. When a graph is weighted, the weights can signify the confidence level of the existence of a given connection between proteins.

Among the many utilities in using graph-based representation, the analysis of centrality measures are among the most widely used applications. Centralities are crucial features of graphs and help to identify the most influential or central nodes. Four centrality measures are of special interest in this work:

- **Degree**: is the number of neighbors of a node i in the network, or in other words, the number of edges that are incident on a given node, considering an undirected graph.
- **Betweenness**: quantifies the number of times that a given node is needed for any node to reach any other node in the graph. Nodes that appear more frequently along the shortest paths between other nodes will receive higher betweenness centrality scores. It is a way of detecting the influence that a node has on the flow of information in a graph based on the extent to which it lies on the shortest paths between pairs of other nodes.
- **Closeness**: computes the sum of distances from a given node to all other nodes, based on the shortest paths between all pairs of nodes. The resulting sum is then inverted to determine the closeness centrality score for that node. In the case of disconnected graphs, where there may not be a path between every pair of nodes, the number of nodes is used as a substitute for the length of the geodesic, which

always results in a longer path than the longest possible geodesic.

- **Clustering coefficient**: also called transitivity, is defined as the ratio of the number of edges between the node's neighbors over the maximum possible number of such edges. The transitivity of a graph is then defined as the average of the clustering coefficients of all its nodes. In case of the local transitivity, this probability is calculated separately for each node.

These different types of centralities can be used in various contexts to understand the structure and behavior of networks and to identify important nodes for distinct applications.

2.2.3.2 Introduction to graph neural networks

To extend the application of convolution and pooling operations from regular grids to arbitrary graphs, it is necessary to recognize that the connectivity patterns in graphs, specially those deriving from biological networks, are typically irregular. Graphs do not exhibit the regular structure found in images, thus presenting a challenge for deep learning algorithms (CHEN et al., 2020).

Graph Neural Networks (GNNs) are a promising approach to address the complexity of such patterns by extending the structural principles of CNNs to non-Euclidean data. In GNNs, a filter is applied to each node in the graph and its neighboring nodes, allowing the recognition of patterns in the local neighborhood of a node and the extraction of local structural information from the graph. This enables GNNs to learn representations of graph-structured data and perform various tasks such as node classification, link prediction, and graph-level prediction (ZHANG et al., 2019).

In the process of selecting a GNN model, some crucial steps must be considered. One such step is determining the type of graph structure, which can be divided into two scenarios: structural and non-structural. In structural scenarios, the graph's structure is explicit in the applications, whereas in non-structural scenarios, the graph is implicit, and it must be built from the task at hand. After obtaining the graph structure, the type of graph must also be determined. As previously mentioned, graphs are typically categorized as either directed or undirected, with directed graphs having edges that are directed from one node to another. Additionally, graphs can be classified as either homogeneous or heterogeneous, with homogeneous (heterogeneous) graphs having nodes and edges of the same (different) types. Moreover, it is also important to define the loss function based on the

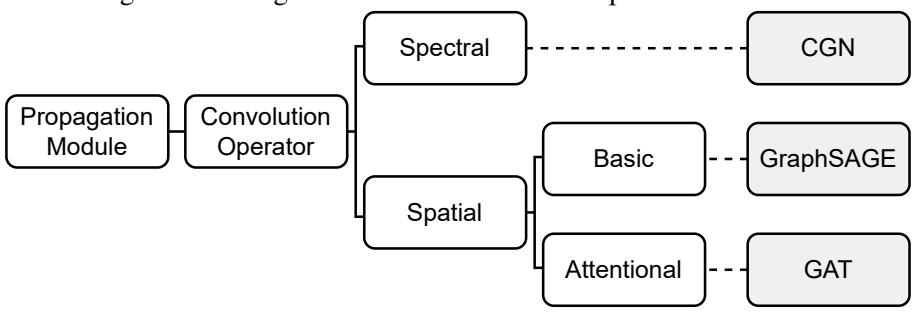
type of learning task addressed (*i.e.,* at node-level, edge-level, or graph-level prediction).

A GNN model is usually built based on the combination of computational modules, which commonly include the propagation module, the sampling module, and the pooling module (ZHOU et al., 2020; GEORGOUSIS; KENNING; XIE, 2021). The propagation module is particularly important for our work, as it enables the dissemination of information between nodes, allowing for the aggregation of both features and topological information to create embeddings. The sampling module is typically used in conjunction with the propagation module when the graphs are large and need to be separated into modules. Finally, the pooling module is employed to extract information from nodes when representations of subgraphs or high-level graphs are required.

The propagation modules in GNNs typically consist of three subcomponents: the convolution operator, the recurrent operator, and the jump connection, as discussed by Zhou et al. (2020). In this text, we will focus on the most commonly used subcomponent for GNN models, the convolution operator, which aims to generalize convolutions from other domains to the graph domain.

Graph convolutions have shown successful applications in various fields, including bioinformatics, where they are particularly useful for analyzing data with no clear and visible structure, such as protein-protein interactions (ERASLAN et al., 2019). Existing graph convolution models can be broadly classified into two categories: spectral-based and spatial-based. Spectral-based models utilize principles from graph signal processing, such as the Laplacian and Fourier transform, to map the irregular structure of a graph onto a regular Euclidean space, enabling convolutional operations. In contrast, spatial-based models directly define the convolution operation using the information dissemination mechanism inherent in the graph, with propagation methods similar to those of standard CNNs.

Figure 2.4 – Algorithms selected under computational module



Source: Adapted from Zhou et al. (2020)

The GNN algorithms selected for exploration in this work are presented in Fig-

ure 2.4, adapted from Zhou et al. (2020). These algorithms are part of the propagation modules, which are responsible for propagating information between nodes, allowing for the capture of topological and feature-related data. This aligns with the goals of our work, which aims to leverage the structure of PPI networks and the information contained in omics data for identification of CDGs, which can be modeled as a node prediction task. As such, these algorithms are ideal constructs for the design of our predictive model. The next sections briefly explain these algorithms

2.2.3.3 Spectral-based models: Graph Convolutional Networks

Spectral graph theory provides the foundation for graph convolution in the spectral domain, which involves transforming graph signals from the nodes domain to the spectral domain. By applying a Fourier or wavelet transform to the signal using the graph Laplacian, spectral approaches can convert graph signals to the graph's frequency domain or spectrum, allowing local convolutions using shared weights similar to traditional CNNs (GEORGOUSIS; KENNING; XIE, 2021).

Graph Convolutional Networks (GCN), proposed by Kipf and Welling (2016), are based on the idea that the properties of a node within a network are substantially influenced by its neighboring nodes' attributes (*i.e.,* features). The model aims to learn a function of signals/features in a graph $G = (V, E)$ that takes as input a feature matrix $X = N \times D$ and an adjacency matrix A representing the graph structure. The output is a node-level matrix Z of size $N \times F$, where F is the number of output features per node. Each neural network layer can be expressed as a non-linear function:

$$H^{(l+1)} = f(H^{(l)}, A) \tag{2.2}$$

where $H^{(0)} = X$ and $H^{(L)} = Z$ (where L represents the number of layers). The different models vary only in how the function f is selected and parameterized.

The layer propagation rule can be represented by:

$$f(H^{(l)}, A) = \sigma(AH^{(l)}W^{(l)}) \tag{2.3}$$

where W is a weight matrix for the l-th neural network layer, and σ is a non-linear activation function like ReLU.

Enforcing self-loops in the graph is a common approach to guarantee that each node propagates its own feature vector. This is done by simply adding the identity matrix to A. However this A is typically not normalized and therefore multiplying with will

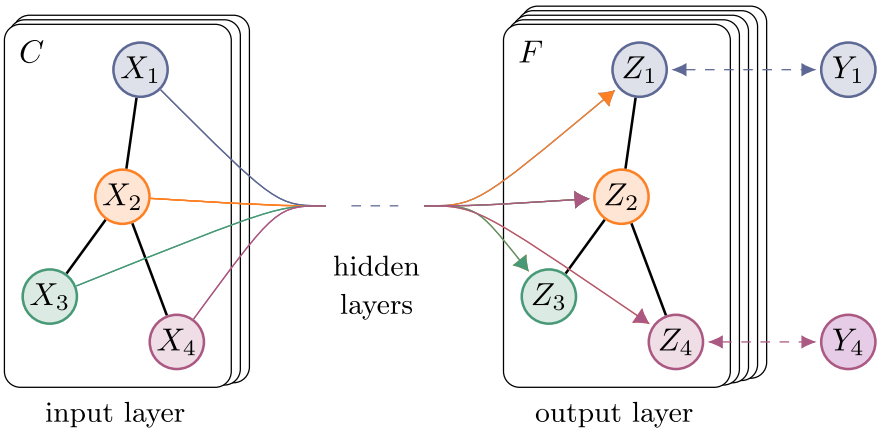
completely change the scale of the feature vectors. To solve this problem, it is necessary to guarantee the normalization of A so that all lines add up to one, that is, $D^{-1}A$, where D is the diagonal node degree matrix. Multiplying with $D^{-1}A$ corresponds to taking the average of neighboring node features. And when we use a symmetric normalization $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ we essentially arrive at the propagation rule introduced by Kipf and Welling (2016):

$$f(H^{(l)}, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{2.4}$$

with $\hat{A} = A + I$, where I is the identity matrix and $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$.

Figure 2.5 illustrates the input channels (C) and the feature maps (F) in the output layer of a spectral-based graph convolution (ZHANG et al., 2021).

Figure 2.5 – Schematic depiction of a graph convolution model.



Source: Kipf and Welling (2016)

2.2.3.4 Spatial-based models: Graph Attention Networks and GraphSage

Spatial convolution on a graph is similar in concept to convolution on a regular domain. Spatial techniques define convolutions directly on the graph, leveraging its topology. However, the challenge lies in devising a convolution operation that can accommodate varying neighborhood sizes while preserving the local invariance of Convolutional Neural Networks (CNNs) (ZHOU et al., 2020).

To address the limitations of Graph Convolutional Network (GCN) and other structurally similar models, Graph Attention Network (GAT) was proposed by Veličković et al. (2017). GAT incorporates an attention mechanism during the graph propagation stage to learn the edge weights between connected nodes. The GAT model takes the node

feature set $x = \{x_1, x_2, ..., x_n\}$ as input to an attention layer, which outputs a new set of learned node features $h = \{h_1, h_2, ..., h_n\}$. The attention coefficient for the edge (i, j) is denoted by α_{ij} and is defined by the following equation:

$$\alpha_{ij} = \frac{exp(LeakyReLU(a^T[Wx_i||Wx_j]))}{\sum_{k \in N_i} exp(LeakyReLU(a^T[Wx_i||Wx_k]))} \tag{2.5}$$

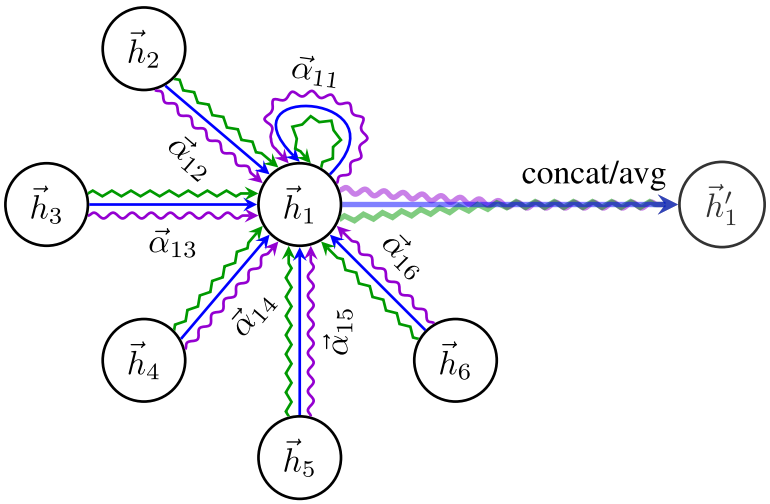
here, N_i refers to the set adjacent nodes of node i, a denotes the learnable weight vector, and W is the shared linear transformation weight matrix. The output features for each node are computed using the following equation:

$$h_i = \sigma(\sum_{j \in N_i} \alpha_{ij} W x_j) \tag{2.6}$$

The technique of multi-head attention extends the attention layer to K separate attention mechanisms, thereby promoting the stability of the self-attention learning process. Figure 2.6 illustrates how the algorithm operates. The final expression is provided as follows:

$$h_i = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in N_i} \alpha_{ij}^k W^k x_j) \tag{2.7}$$

Figure 2.6 – Multi-head attention (with $K = 3$ heads) by node 1 on its neighborhood.



Source: Veličković et al. (2017)

Hamilton, Ying and Leskovec (2017) proposed another spatial-based model that uses a conceptually simpler GCN, referred to as GraphSAGE (SAmple and aggreGatE). This model is a general inductive framework that leverages node attribute information to generate node embeddings for previously unseen data efficiently. Instead of training

separate embeddings for each node, a function is learned that generates embeddings by performing sampling and aggregation of features from a node's immediate neighborhood.

Each layer of the GraphSAGE network consists of two principal procedures: sampling and aggregation. The aggregation procedure for a layer l also consists of two steps. The first step is the aggregation of the signals from a node's immediate neighborhood:

$$\mathrm{h}_{N_i}^{l+1} = aggregate_{l+1}(\{h_j^l, \forall j \in N_i\}) \tag{2.8}$$

The second step is the concatenation of the aggregated signals with the target node signal, which is passed through a dense layer:

$$\mathrm{h}_i^{l+1} = \sigma(W^{l+1}.[h_j^l || h_{N_i}^{l+1}]) \tag{2.9}$$

where $W_l \in \mathbb{R}^{d \times 2c}$ is a matrix of learned weights for the lth layer, and σ is a non-linear activation function, and the first layer h_0 is the imput signal.

2.2.4 Ensemble learning

Ensemble learning is a popular approach in machine learning that involves combining multiple individual models to improve overall predictive performance, particularly in supervised tasks. Once the individual models, also known as base learners or base models, are trained, their combined predictions can be used to label new, unseen data. The central idea behind ensemble learning is that introducing diversity among models, whether through variations in data or algorithms, can compensate for errors in individual models and improve overall performance more effectively than a single model (SAGI; ROKACH, 2018; DONG et al., 2020).

Building an ensemble model involves, in general, applying a methodology for training the individual models in order to introduce some degree of diversity among them, and choosing a suitable process for combining the outputs of these models into a single output. Diversity can be introduced, for example, by random resampling the data or by employing different learning algorithms or different hyperparameters configuration. Regarding the combination of the results from the base models, although several approaches exist, they can be organized into two main categories:

- **Strategies that combine the predicted class labels**: each base model provides a class label that is meaningful to the problem domain. A simple, yet very effective

approach to combine these predictions, is by voting. The majority voting method, for instance, involves predicting the class label with the most votes.

- **Strategies that combine the predicted class probabilities**: most ML algorithms are able to assign a probability for each class, which summarizes the likelihood of an instance belonging to that class. Thus, combination rules for ensemble models can also be applied over the predicted probabilities. One common way to merge probabilities is simply to take the average among values predicted for a given class. Then, the ensemble class label can be extracted based on the class that maximizes the merged probabilities.

We note that the algorithms Random Forest and Gradient Boosting Trees, presented in Section 2.2.2, are examples of ensemble-based learning methods. RF introduces diversity by applying a repeated resampling without replacement (*i.e.,* bootstrap), while GBT uses a boosting strategy that resamples data based on weights that are adjusted throughout the process according to previous classification hits and errors.

2.2.5 Class imbalance

Class imbalance occurs when there is an uneven distribution of instances across the target classes in a classification problem. The majority class has the highest proportion of instances, while the minority class has the fewest, resulting in limited representation for the concept underlying the minority class in the dataset (HE; GARCIA, 2009). In many cases, the minority class is of particular interest, such as in clinical diagnosis with ML predictive models, where only a small fraction of patients are expected to present the disease, or when predicting cancer driver mutations, where most sequenced mutations are passenger rather than driver mutations.

Although several real world problems are characterized by class imbalance in their data distribution, when the training data is imbalanced, ML algorithms struggle to distinguish effectively between the minority and majority class. This happens because most standard ML algorithms expect balanced class distributions or assume equal misclassification costs (*i.e.,* the cost of a false positive prediction is the same as of a false negative one) (HE; GARCIA, 2009). Thus, class imbalance presents a longstanding challenge for ML, as models tend to be biased towards the majority class, leading to high error rates for the minority class (LEEVY et al., 2018).

Several class imbalance learning solutions have been proposed in literature (HE; GARCIA, 2009). In what follows, we briefly review those employed in the current work:

- **Undersampling Majority Class**: a widely used approach that involves a direct subsampling technique to balance class distributions. The technique randomly excludes instances of the majority class, usually until this class has the same number of instances as the minority class. This technique is only suitable for large datasets as it can potentially discard crucial data points for the model.
- **Balanced Cross-Entropy**: a popular technique to tackle class imbalance that involves introducing a weighting factor, denoted as $\alpha \in [0, 1]$ for the positive class and $1 - \alpha$ for the negative class. This weight factor represents an attempt to readjust the misclassification costs. In practice, α can be determined by the inverse class frequency or treated as a hyperparameter and determined through cross-validation. The α-balanced cross-entropy (CE) loss can be expressed as follows:

$$CE(p_t) = -\alpha_t log(p_t) \tag{2.10}$$

  where p_t is the model's estimated probability p for the positive class if the observed class is actually positive
- **Focal Loss**: a newer approach proposed by Lin et al. (2017), which is a dynamically scaled cross-entropy loss encompassing a scaling factor that gradually decreases to zero as the confidence of the model in the correct class prediction increases. This scaling factor $(1 - p_t)^\gamma$ can effectively reduce the influence of simpler examples during the training phase and facilitate the model to concentrate on more complex examples. Thus, the α-weighted FL for binary classification is defined as shown by the equation below:

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma log(p_t) \tag{2.11}$$

  where α is the same weighting parameter used in balanced cross-entropy, while γ is a new adjustable parameter that controls the degree of focusing. When $\gamma = 0$, the function is equivalent to the balanced cross-entropy approach.

2.2.6 Model evaluation

Model evaluation is a critical step in a machine learning pipeline and is typically applied to three main sub-tasks in model development (RASCHKA, 2018): (i) estimate the generalization performance of a trained model, that is, assess the predictive performance of the model on future (unseen) data, (ii) increase the predictive performance of a learning algorithm by exploring the hypothesis space more broadly with different hyperparameter settings (*i.e.,* conduct a fine-tuning of hyperparameters), and (iii) select the ML algorithm that is best-suited for our prediction problem (*i.e.,* compare different algorithms). The step of model evaluation entails two main decisions: how to split the original dataset to allow model training and performance assessment, and how to evaluate the predictive performance. The next sections review concepts related to these aspects.

2.2.6.1 Data splitting strategies

The holdout method is a simple and commonly used data splitting strategy in machine learning. It involves randomly dividing a labeled dataset into two parts: a training set and a test set, typically in a 75:25 ratio (other common ratios are 70:30 and 80:20). The training set is used to fit the model, while the test set is used to evaluate its performance on unseen data. To ensure that the resulting subsets are representative of the original dataset, random splitting is often conducted in a stratified manner, preserving the original class proportions in the resulting subsets. Although widely adopted for model evaluation, holdout method presents some limitations: (i) it is very sensitive to how we split the data into training and test sets, (ii) it does not allow us to estimate how performance varies according to differences in the test set, (iii) it does not support the hyperparameter optimization process, which must be done with independent data to avoid the effect of data leakage (RASCHKA, 2018).

To address these limitations and provide a more robust estimate of performance, k-fold cross-validation (CV) has become the most common technique for model evaluation and selection in ML. The idea of k-fold CV is to split the dataset into k disjoint parts (*i.e.,* folds) of similar size: one part is used for validation, and the remaining $k - 1$ parts are merged into a training set in an iterative process. Although k is a hyperparameter of the algorithm, $k = 5$ and $k = 10$ are commonly applied values. At the end, the performance is usually estimated as the arithmetic mean (and standard deviation) over k validation sets. Through this process, k-fold CV guarantes that all instances are used for training

and testing, reducing the risks of a highly pessimistic or optimistic performance estimate.

Holdout and k-fold CV are often used together for model evaluation and selection. In this approach, holdout is applied as a first step to split the original dataset into training and test sets. The test set is kept aside as an independent dataset from the process of model training and selection to avoid data leakage. The training set is then used in the k-fold CV to experiment with one or more algorithms and various hyperparameter settings. The resulting performance estimates from k-fold CV are used to select the best hyperparameter setting for the algorithm, which is then used to fit the model with the complete training data. Finally, the independent test set withheld in the first step is used for the final evaluation of model performance.

2.2.6.2 Performance metrics for classification tasks

Several performance metrics can be adopted to assess distinct types of correct predictions (*i.e.,* true positive (TP) and true negative (TN)) and incorrect predictions (*i.e.,* false positive (FP) and false negative (FN)) in binary classification tasks. Here, we summarize the metrics analyzed in the current work.

- **Accuracy** (**Acc**): reflects the proportion of instances in the test set that were predicted correctly by the model (*i.e.,* the hit rate). Accuracy alone may not properly reflect the model performance for datasets with class imbalance, especially in severily skewed data distribution, as it may be biased towards the majority class. The metric is computed as follows:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.12}$$

- **Precision** (**Prec**): estimates the proportion of positive instances predicted correctly by the model. Precision is also called Positive Predictive Value (PPV) and is a measure of classifiers exactness. Low precision indicates a high number of false positives. Precision is computed according to the following equation:

$$Pre = \frac{TP}{TP + FP} \tag{2.13}$$

- **Recall** (**Rec**): measures the ability of a model to correctly identify positive in-

stances. Recall is also called Sensitivity or True Positive Rate (TPR), and a low recall indicates a high number of false negatives. Recall is estimated as follows:

$$Rec = \frac{TP}{TP + FN} \tag{2.14}$$

- **Specificity** (**Spe**): measures the proportion of true negatives that are correctly identified by the model. Specificity is also called True Negative Rate (TNR), and is computed according to the following equation:

$$Spe = \frac{TN}{TN + FP} \tag{2.15}$$

The sum of specificity and false positive rate (FPR) is always 1. Thus, it is common estimate the FPR as:

$$FPR = 1 - Spe \tag{2.16}$$

- **Area under the ROC curve** (**AUC-ROC**): the Receiver Operator Characteristic (ROC) curve shows the trade-off between true positive rates (*i.e.,* sensitivity, in the y-axis) and false positive rates (in the x-axis) across different thresholds of prediction probability[2]. The area under the generated ROC curve (AUC-ROC) is often used as an estimate of model performance and represents the probability that a randomly chosen positive instance will be ranked higher than a randomly chosen negative instance. Thus, the higher the AUC-ROC value, the better the predictive performance of the model.
- **Area under the Precision-Recall curve** (**AUC-PR**): the Precision-Recall (PR) curve shows the tradeoff between precision (*i.e.,* in the y-axis) and recall (in the x-axis) for distinct thresholds of prediction probabilities. The area under the PR curve (AUC-PR) is very useful to summarize the performance, with high scores meaning that the model is able to return precise results (high precision), as well as to correctly predict the majority of the positive instances (high recall). The AUC-

[2]Most performance metrics evaluate classifiers based on a specific probability threshold, usually 0.5, meaning that an instance is assigned to the positive class if the predicted probability is equal or above 0.5. This is the case, for instance, for accuracy, precision, recall, and specificity. When using approaches based on performance curves, the skill of a model is assessed across all thresholds of probability.

PR is a more appropriate measure of classification success under the presence of class imbalance.

3 RELATED WORKS

This chapter reviews computational approaches proposed in previous works to identify driver mutations and cancer driver genes. Due to the scope of this work, we focus on machine learning-based solutions, discussing how interactions between data types and ML algorithms have been explored in related works. We also describe current analytical limitations identified through literature review, some of which served as motivation for the present work. The chapter is based on a survey paper published in the journal *Briefings in Bioinformatics* (ANDRADES; RECAMONDE-MENDOZA, 2022), as follows:

> ANDRADES, R.; RECAMONDE-MENDOZA, M. Machine learning methods for prediction of cancer driver genes: a survey paper. **Briefings in Bioinformatics**, Volume 23, Issue 3, May 2022, bbac062. Available: <https://doi.org/10.1093/bib/bbac062>.

3.1 Initial considerations

Previous works have dedicated efforts to summarize and organize literature regarding computational approaches for prediction of CDGs and driver mutations, with different focuses (ZHANG et al., 2014; CHEN; SUN; SHEN, 2015; CHENG; ZHAO; ZHAO, 2016; DIMITRAKOPOULOS; BEERENWINKEL, 2017; CHEN et al., 2020; ROGERS; GAUNT; CAMPBELL, 2020). Although some ML-based methods were covered by some of these works, they were mostly developed for a more general problem of distinguishing disease-related SNVs from common polymorphisms (ZHANG et al., 2014; CHEN; SUN; SHEN, 2015). Other surveys have categorized methods according to their major feature types (CHENG; ZHAO; ZHAO, 2016) or prediction strategy (DIMITRAKOPOULOS; BEERENWINKEL, 2017), without emphasis on ML. Recent works also focused on comparing the predictive performance of distinct computational methods for cancer drivers prediction (CHEN et al., 2020; PHAM et al., 2021). Finally, Rogers, Gaunt and Campbell (2020) reviewed generic ML-based tools to predict the pathogenic impact of human genome variants, further concentrating their discussion on a specific set of tools to predict cancer drivers. Nonetheless, the theoretical background and methodological details of ML were not discussed by the authors.

A table summarizing the articles mentioned, with a brief description of their scope,

is provided in the appendix (Table A.1). All these previous works, jointly, have covered a large body of the literature regarding computational methods for cancer driver discovery and have been crucial to elucidate the particular niche targeted by each solution, as well as their potential in solving the task. We emphasize, however, that aspects entailed in the development of ML models for CDG prediction were not the focus of previous discussions. This chapter provides a comprehensive analysis of ML-based computational approaches to identify driver mutations and CDG, constructing an integrated, panoramic view of data and ML techniques within this domain.

3.2 Search methodology

Two main bibliographic repositories were used as search sources, PubMed and DBLP. While the former is most focused on biomedical literature, the latter is specialized in computer science bibliography. These repositories were chosen because, together, they cover a wide range of papers from high-impact international scientific journals and conferences in both the fields of medicine and computer science. The query for the PubMed database was constructed using keywords related to "cancer driver genes", "prediction", and "machine learning" as terms of interest. For DBLP, the search was conducted using only terms related to the domain of interest since all indexed articles are within the computer science field. These terms included "cancer driver genes", "cancer", "disease-associated genes", and "oncogenes".

Our initial search returned 355 related papers found in PubMed and 84 papers retrieved from DBLP. Duplicates were removed, resulting in 420 papers. Each paper was individually evaluated regarding its relevance to the research topic covered. As inclusion criteria, we only considered papers that explicitly mention any ML algorithm as part of its methodological approach. All titles and abstracts were screened to identify papers that employed machine learning techniques to predict cancer driver genes. When necessary, we carried out a diagonal reading of the papers to confirm their suitability for this chapter's scope. After this initial analysis, 45 (12.68%) papers from PubMed and 7 (8.33%) papers from DBLP were considered to be significantly related to the research question addressed by this work. Interestingly, many discarded papers focused on network-based computational methods to identify cancer drivers. Here, we only selected those that addressed the use of ML techniques in their methodology in conjunction with network-based approaches.

The 52 eligible papers were read in full for a more in-depth analysis of their methodology. We verified aspects such as which specific ML techniques were adopted in the proposed solution and the operating steps in which they were used. After this second round of paper examination, 36 papers were confirmed to be relevant, as they applied ML algorithms as a central part of the proposed computational solution for cancer drive gene prediction. Finally, to guarantee that no significant contribution to the field of cancer driver gene prediction was left behind, we manually revised the references of the selected papers to search for relevant works that our search strategy has not retrieved. Besides, we used Google Scholar to search the scientific literature for studies that have cited the selected papers and thus increase the range of articles included in this review chapter. Five papers were selected after these additional searches, totaling 41 relevant papers for our work (Table 3.1). Our search was concluded on April 1st, 2021.

The acronyms used in Table 3.1 represent the categorization regarding the data used by the authors, Functional Genomics (FG), Functional Integration (FI), Genomic variation (GV), Ontology-based (Ob) and Network-based (Nb). As for the algorithm used, Artificial neural network (ANN), Probabilistic methods (Pm), Regression-based (Rb), SVM-based (SVM), Tree-based (Tb), Supervised learning (others) (So), Deep learning (DL), Traditional unsupervised learning (UL) and evolutionary algorithm (EA). These concepts will be explored in detail in the following sections. In addition, the performance metrics column also includes a list of acronyms used to simplify the table, accuracy (Acc), F1-measure (F1), Matthew's correlation coefficient (MCC), precision (Pre), area under the Precision-Recall curve (AUC-PR), area under the ROC curve (AUC-ROC) and recall (Rec).

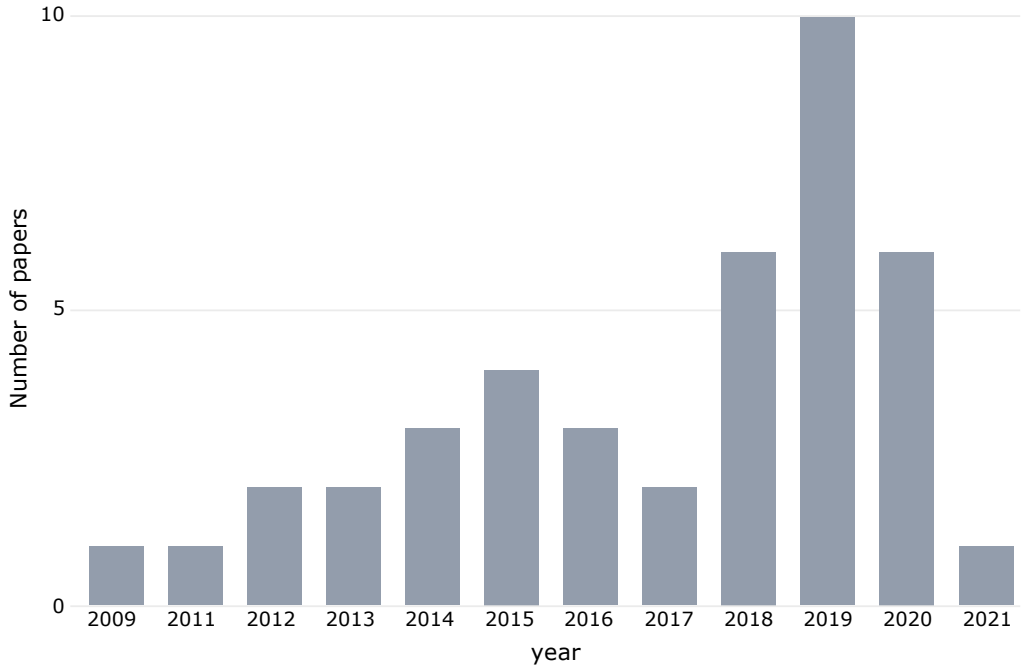
Table 3.1 – Main characteristics of selected papers.

Reference	Data categories	Data integration	Algorithms	Feature selection	Performance metrics	Validation strategy
Carter et al. (2009)	GV;FI	Y	Tb	Y	AUC-ROC; Acc	k-fold CV
Capriotti and Altman (2011)	GV;FI;Ob	Y	SVM	N	AUC-ROC; Acc	k-fold CV
Fu et al. (2012)	FG	N	So	N		
Tan, Bao and Zhou (2012)	GV;FI;Ob	Y	SVM	Y	Acc	k-fold CV
Davoli et al. (2013)	GV;FI	Y	Rb	Y	Acc	k-fold CV
Mao et al. (2013)	GV;FI	Y	SVM	Y	AUC-ROC	k-fold CV
Manolakos et al. (2014)	FG;Ob	Y	UL	N		Holdout (70%/30%) + k-fold CV
Schroeder et al. (2014)	GV;FI	Y	Rb;Tb;Pm	N	Acc	k-fold CV
U et al. (2014)	FI	N	SVM;Tb;ANN;Pm;So	Y	Acc; F1; Pre; Rec	k-fold CV
Anoosha et al. (2015)	FI	N	SVM	Y	Acc	k-fold CV
Gnad et al. (2015)	GV;FI;FG	Y	Rb	Y	Acc	k-fold CV
Park et al. (2015)	GV;FG;Nb	Y	Rb	Y	Acc	
Soliman et al. (2015)	GV;FI	Y	SVM;Rb	Y	Acc; F1; MCC	k-fold CV
Dong et al. (2016)	GV;FI;Ob	Y	SVM;Rb	Y	AUC-ROC	k-fold CV
LI et al. (2016)	GV	N	EA	N		
Tokheim et al. (2016)	GV;FI;FG;Nb	Y	Tb	N		
Park et al. (2017)	GV;FG	Y	Rb	Y	Acc	
Tavanaei et al. (2017)	FI	N	DL	N	AUC-ROC; Acc; Pre; Rec	Holdout (85%/15%)
Agajanian et al. (2018)	FI	N	Rb;Tb	Y	Acc; F1; Pre; Rec	Holdout (80%/20%) + k-fold CV
Celli, Cumbo and Weitschek (2018)	FG	N	Tb	N	F1	Holdout (70%/30%)
Guan et al. (2018)	GV;FG	Y	SVM	Y		k-fold CV
Lu et al. (2018)	GV;FG	Y	UL	N	Acc; F1; Rec	k-fold CV
Wang et al. (2018b)	GV;FI;FG	Y	Pm	N	AUC-ROC	k-fold CV
Zhou, Gao and Skolnick (2018)	FI;Ob;Nb	Y	Tb	Y	F1; Rec; MCC	k-fold CV
Agajanian, Oluyemi and Verkhivker (2019)	GV;FI	Y	Tb;DL	Y	AUC-ROC; Acc; F1	Holdout (80%/20%) + k-fold CV
Althubaiti et al. (2019)	Ob	N	ANN	N	AUC-ROC; F1	k-fold CV
Collier, Stoven and Vert (2019)	GV;FI;Nb	Y	SVM	N	AUC-ROC	k-fold CV
Han et al. (2019)	GV	N	So	N		
Jiang et al. (2019)	GV;FG	Y	Tb	N		
Luo et al. (2019)	GV;FG	Y	DL	N	AUC-ROC	
Nicora et al. (2019)	GV;FI	Y	SVM;Rb;Tb	N	Acc; Rec	Holdout (70%/30%) + k-fold CV
Schulte-Sasse et al. (2019)	GV;FG;Nb	Y	DL	N	AUC-ROC; AUC-PR; MCC	k-fold CV
Xi et al. (2019)	GV	N	UL	N	Pre; Rec	k-fold CV
Zhu et al. (2019)	FI	N	-	N		
Chandrashekar et al. (2020)	GV;FI	Y	Tb	N	Acc	k-fold CV
Colaprico et al. (2020)	GV;FI;FG;Ob;Nb	Y	Tb	N		
Cutigi et al. (2020)	GV;Nb	Y	SVM;Tb	N	AUC-ROC; Acc; F1; Pre; Rec	k-fold CV
Gumpinger et al. (2020)	FI;Nb	Y	SVM;Rb;Tb	N	F1; Pre; Rec; AUC-PR	k-fold CV
Lyu et al. (2020)	GV;FI;FG;Ob	Y	SVM;Rb;Tb	Y	Acc; Pre; Rec; AUC-PR	k-fold CV
Wang et al. (2020)	FI	N	SVM;Tb;ANN;So	N	AUC-ROC; Acc; F1; Pre; Rec	k-fold CV
Nulsen et al. (2021)	GV;FI;FG;Ob;Nb	Y	SVM	Y	AUC-ROC	k-fold CV

3.3 Overview of selected papers

Papers distribution according to the publication year (Figure 3.1) suggests a growing interest in the research topic. Most papers (56.09%) were published after 2018 and the highest number of publications were found in 2019 (10). The low number of related papers published in 2021 is justified by the short period covered by our analysis. Papers were published mainly in scientific journals, with only five (FU et al., 2012; TAVANAEI et al., 2017; NICORA et al., 2019; SCHULTE-SASSE et al., 2019; CUTIGI et al., 2020) appearing in conference proceedings. The top three journals in terms of number of papers were Bioinformatics (6), Journal of Computational Biology (3), and Plos One (3).

Figure 3.1 – Number of selected papers per year of publication.



Source: Prepared by the author

In terms of target prediction problem, 32 papers (*i.e.,* 78.04%) concentrated in running predictions at the gene level. Among these, six papers (DAVOLI et al., 2013; GNAD et al., 2015; TAVANAEI et al., 2017; CHANDRASHEKAR et al., 2020; COLAPRICO et al., 2020; LYU et al., 2020) aimed to distinguish oncogene and tumor suppressor gene (TSG) (*i.e.,* the two subclasses of CDGs), whereas the others focused on classifying a given gene as CDG or not. Seven papers targeted predictions on mutation level (MAO et al., 2013; U et al., 2014; SOLIMAN et al., 2015; AGAJANIAN et al., 2018; ZHOU; GAO; SKOLNICK, 2018; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; WANG et al., 2020), most of which restricted the analysis for missense mutations.

We also found one paper aiming at identifying cancer modules to discover cancer driver genes (MANOLAKOS et al., 2014) and other focusing on the prediction of false positive CDGs (CUTIGI et al., 2020). Also, while most papers addressed cancer drivers in general, some focused on specific types of cancer, such as colon adenocarcinoma (FU et al., 2012; LUO et al., 2019), breast cancer (CELLI; CUMBO; WEITSCHEK, 2018; LU et al., 2018; LUO et al., 2019), lung adenocarcinoma (LUO et al., 2019), thyroid (CELLI; CUMBO; WEITSCHEK, 2018), and kidney (CELLI; CUMBO; WEITSCHEK, 2018). We also observed predictive models for cancer-related mutations in human protein kinases (U et al., 2014) and, more specifically, in the Epidermal Growth Factor Receptor (ANOOSHA et al., 2015).

To better map the current state-of-the-art in terms of available resources and adopted methodologies, two main analyses were made. The first one focused on the data types used as model's feature, while the second focused on summarizing the computational aspects of the proposed predictive model, such as type of learning and specific ML algorithms adopted. The next sections describe and organize the selected papers in terms of data categories and computational strategies.

3.4 Data categories

A core component for ML-based predictive models is the training data due to the common sense that much of the model's success depends on the fed data. A given prediction problem may have its concept represented in many different ways, which is especially true in the genomics domain. Some of these representations may be better than others in revealing the patterns we sought to learn. Thus, a clear comprehension of the possible types of features for instances representation within our domain may provide insights into current limitations and new analytical opportunities. We observed a large variety of information used as models' input features for cancer drivers prediction. After careful analysis of the selected papers, we classified them into five categories based on the properties evaluated as predictors (Table 3.2): (1) *Genomic Variation* (GV), (2) *Functional Impact* (FI), (3) *Functional Genomics* (FG), (4) *Network-based* (Nb), and (5) *Ontology-based* (Ob). Subcategories were defined for some categories to better organize the corresponding features based on their semantics.

A general overview of papers in terms of the data categories employed is shown in Figure 3.2. *Genomic variation* was the most common feature category used (29 papers,

Table 3.2 – Data categories and subcategories adopted to classify selected papers according to types of features employed.

Category	Subcategory	Description
Genomic Variation	Mutations	properties of somatic single nucleotide variants (SNV) and frameshift insertion/deletion (*e.g.,* estimate of the mutation frequency, mutation ratio, mutation hotspots)
	Copy number alteration (CNA)	information related to amplifications, deletions, and duplication of segments of DNA that changes the number of copies of a particular DNA segment within the genome (*e.g.,* deletion and amplification frequency)
	DNA sequence	raw nucleotide sequence
Functional Impact	Functional impact scores	outputs of *in silico* variant effect predictors concerning the probability of deleterious changes on protein function
	Protein-based	properties of protein sequence and structure, from amino acids to protein's tertiary structure
	Evolution-based	evolutionary conservation scores, amino acids substitution rate, gene age, gene damage index, number of human paralogs, etc.
Functional Genomics	Transcriptomics	large-scale gene expression profiles or statistics derived from differential gene expression analyses
	Epigenomics	DNA methylation, histone modifications, chromatin accessibility, DNA replication time
	Proteomics	protein expression data, mainly expressed as categorical features that indicate whether or not a protein is expressed in a given human tissue
Ontology-based	-	functional, cellular, or phenotypic annotations obtained from bioinformatics databases or related works
Network-based	-	node properties from structural analysis of molecular networks (*e.g.,* PPI or gene-gene networks)

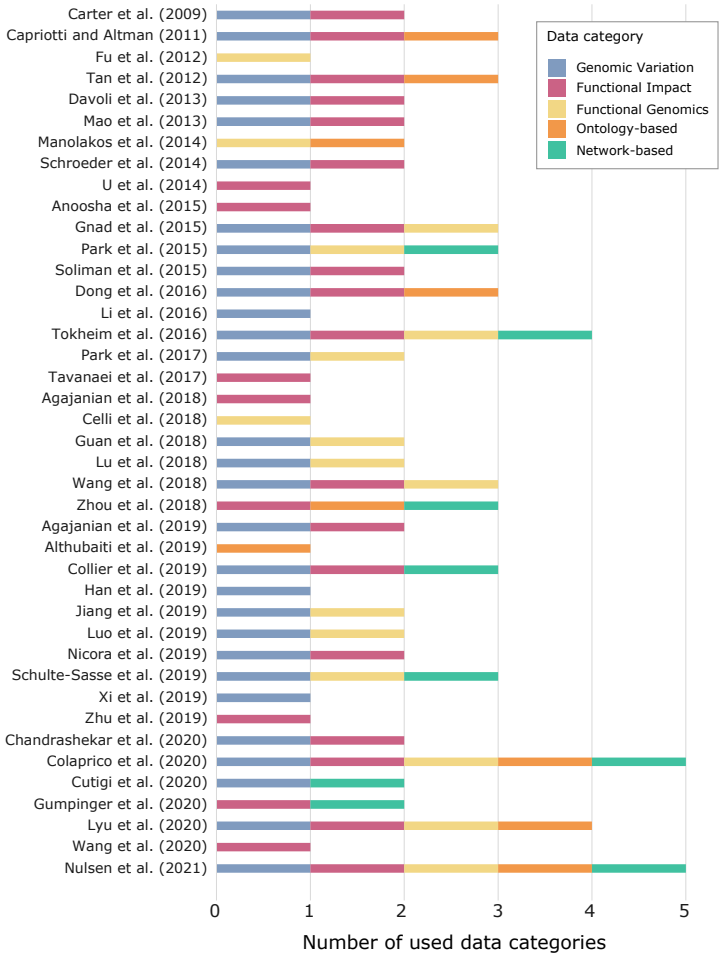
i.e., 70.73%), followed by *Functional Impact* (26 papers) and *Functional Genomics* (16 papers). *Network-based* and *Ontology-based* features were each employed in 9 papers (Figure 3.2a). Since some data categories contain multiple subcategories, we analyzed the number of distinct feature datasets adopted per paper and found that it varied from 1 to 8 ((NULSEN et al., 2021)), with an average of 2.87 ($\pm$ 1.17) datasets.

One of our main findings regarding the features aspect of ML models is that many works adopted more than one data category. Twenty-nine papers (*i.e.,* 70.73%) used two or more data categories and were classified in a *Data integration* category. The distribution of the number of occurrences of data categories by years of publication does not suggest any association between these factors (Figure 3.2b).

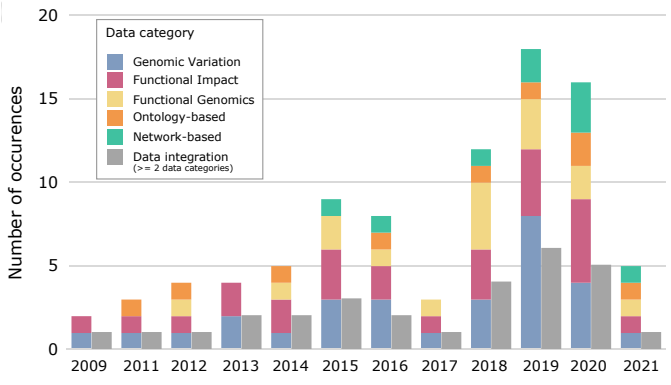
Moreover, integrating different data categories is not necessarily a recent tendency, as papers published in 2009 and 2011 already proposed such strategy. Nonetheless, most of the papers using features from three or more data categories were published from 2015, and those that used four (TOKHEIM et al., 2016; LYU et al., 2020) or five (COLAPRICO et al., 2020; NULSEN et al., 2021) data categories were published mainly in 2020 and 2021, which may indicate a trend for increasing data diversity in newer models.

The Venn diagram in Figure 3.2c summarizes the intersections among distinct data categories. The most recurrent combination was the integration of *Genomic Variation* and *Functional Impact* (8 papers), followed by *Genomic Variation* and *Functional Genomics* (5 papers). Also, while *Functional Impact* was the most common data category used as a single source of features in the proposed models, the *Network-based* category was not exclusively used by any ML model among the revised papers. In what follows, we review

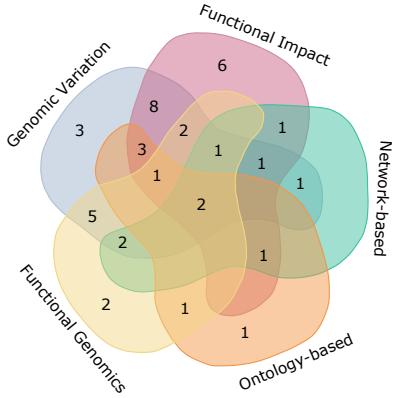
Figure 3.2 – Analysis of data categories used as model features by selected papers.
(a) Relation of data categories per paper, organized by year of publication in ascending order.



(b) Distribution observed in the use of the data categories per year of publication.



(c) Venn diagram showing the intersection of studies in terms of employed data categories.



Source: Prepared by the author

the particularities of each data category.

3.4.1 Genomic variation

Three papers (LI et al., 2016; HAN et al., 2019; XI et al., 2019) used *Genomic Variation* data as the single source of feature, whereas 23 (*i.e.,* 56.09%) combined it with other categories. A long-standing hypothesis in the discovery of cancer drivers is that driver genes are mutated more frequently than expected as compared to a background mutation rate (BMR) estimated from cancer samples for a given cancer type (TAMBORERO et al., 2013). Nonetheless, the mutational landscape of cancer consists of 'mountains' of very frequently mutated genes and of 'hills' of significantly but less frequently mutated genes (VOGELSTEIN et al., 2013). Thus, simply characterizing candidate driver genes with frequency-based methods poses the challenge of robust BMR estimation: a low BMR may lead to many spurious findings, whereas a high BMR may miss the driver genes mutated at very low frequency. ML-based methods aim to allow the algorithm to learn the mutational patterns related to cancer drivers from training data. Here, we considered three subcategories (Table 3.2): i) mutations, ii) copy number alteration (CNA); and iii) raw DNA sequence, which was included since information regarding DNA variants would be implicitly available.

Mutation-related properties was the most common feature type, employed by 26 papers (*i.e.,* 63.41%). Fifteen papers explored mutations' properties as the single source of genomic variation features, and one paper (HAN et al., 2019) exclusively relied on mutation-based features for model development. Mutation frequency was widely used by selected papers, usually estimated from databases such as Catalogue of somatic mutations in cancer (COSMIC) (TATE et al., 2018), The Cancer Genome Atlas (TCGA), and HapMap (GIBBS et al., 2003). Some works (DAVOLI et al., 2013; GNAD et al., 2015; LYU et al., 2020) distinguished between potentially high (HiFI) or low (LoFI) functional impact mutations when computing mutation frequency, adopting *in silico* predictors of mutations' functional impact. Moreover, silent mutations and LoFI missense mutations, or a combination of both (GNAD et al., 2015; LYU et al., 2020), were often taken as a measure for genes' BMR. Davoli et al. (2013) proposed an entropy-based mutation selection score that reflects the spatial distribution of these features, measuring the occurrence of 'mutation hotspots'. This measure of positional clustering was adopted by several papers (GNAD et al., 2015; TOKHEIM et al., 2016; COLLIER; STOVEN; VERT, 2019; LUO et al., 2019; COLAPRICO et al., 2020; NULSEN et al., 2021). Mutation hotspots were also detected using scores computed by OncoDriveCLUST (TAM-

BORERO; GONZALEZ-PEREZ; LOPEZ-BIGAS, 2013; SCHROEDER et al., 2014; NULSEN et al., 2021) and applying density estimates to aggregate closely-spaced missense mutations into peaks and compute mutation fraction inside the highest peak (CHANDRASHEKAR et al., 2020).

The normalized number of single nucleotide variants (SNVs) in the mutation's exon (MAO et al., 2013), mutations' distance to closest Transcribed Sequence Start (TSS) and closest Transcribed Sequence End (TSE) (AGAJANIAN; OLUYEMI; VERKHIVKER, 2019), and a gene-level binary matrix summarizing mutation occurrence across all samples (LI et al., 2016; XI et al., 2019) were also adopted.

Fourteen papers (*i.e.,* 34.15%) (DAVOLI et al., 2013; SCHROEDER et al., 2014; GNAD et al., 2015; PARK et al., 2015; DONG et al., 2016; LI et al., 2016; PARK et al., 2017; GUAN et al., 2018; LU et al., 2018; JIANG et al., 2019; SCHULTE-SASSE et al., 2019; XI et al., 2019; LYU et al., 2020; NULSEN et al., 2021) adopted CNA data as models' features, which was first introduced in this domain by Davoli et al. (2013).

Their model analyzed the deletion and amplification frequency to distinguish among neutral genes, oncogenes, and TSGs. Two papers (GNAD et al., 2015; LI et al., 2016) employed the Genomic Identification of Significant Targets In Cancer (GISTIC) (BEROUKHIM et al., 2007; MERMEL et al., 2011) algorithm to obtain DNA amplification and deletion regions. Gnad et al. (2015) used as features the sums of amplification or deletion frequencies across 13 TCGA cancer types. Eleven papers that fall within the CNA subcategory also adopted mutation data. However, in most cases, the features from the different subcategories were used independently rather than combined into a single input feature. One exception is the work by Schulte-Sasse et al. (2019), in which the sum of SNVs and CNA is averaged across all samples of a given cancer type to estimate a cancer mutation rate per gene. Furthermore, among the three papers that apply only CNA features from the *Genomic variation* category, two (PARK et al., 2017; GUAN et al., 2018) also adopted gene expression data in their analyses, since CNA is known to mediate phenotypic changes through their impact on expression. Park et al. (2017) quantified the association between CNA and gene expression using a previously proposed method (YUAN et al., 2011), differentiating between cis-effects (*i.e.,* when gene expression is influenced by CNA in proximal genes within a several Mb window) and trans-effects (*i.e.,* when gene expression is influenced by remote alterations throughout the genome). Guan et al. (2018) used gene-level summarized cell-lines CNA and expression profiles obtained from the Cancer Cell Line Encyclopedia (CCLE).

Finally, only one paper (AGAJANIAN; OLUYEMI; VERKHIVKER, 2019) used raw DNA sequence as model's features. Mutations were represented by stacking two nucleotide sequences on top of each other, one with the original nucleotide for a given position and the other with the mutated version, and a convolutional neural network (CNN) was trained to automatically extract driver-related sequence patterns.

3.4.2 Functional impact

The functional impact (FI) of SNVs on protein function was analyzed by 63.41% of papers. Their goal is to better identify lowly recurrent mutated driver genes or driver genes that are mutated late during tumor development, which are more challenging cases for methods that exclusively analyze genomic variation features (GONZALEZ-PEREZ; LOPEZ-BIGAS, 2012). This has become a widely applied strategy even outside the scope of ML-based papers, as reflected by the several computational methods developed for pathogenicity prediction of SNVs (CHEN et al., 2020; WU et al., 2016). Three subcategories were created (Table 3.2): i) functional impact scores provided by *in silico* predictors for variant effect; ii) protein-based features; and iii) evolution-based features. Although most FI predictors rely on features related to proteins' properties or evolutionary conservation (CHEN et al., 2020), we did not consider this in our classification.

FI scores was the largest subcategory, employed by 18 papers. Table 3.3 summarizes the FI predictors used. Most works adopted the predicted scores or p-values as models' features, while others (*e.g.,* (COLLIER; STOVEN; VERT, 2019)) estimated the number of damaging missense mutations per gene using a pre-defined score threshold. Several works integrated the output of multiple FI predictors in their feature vectors (MAO et al., 2013; DONG et al., 2016; AGAJANIAN et al., 2018; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; ZHU et al., 2019; WANG et al., 2020). Dong et al. (2016) used 11 tools for point coding mutations and one tool, FunSeq2 (FU et al., 2014), to annotate non-coding variants. Wang et al. (2020) explored the largest diversity of tools: 23 conservation-base, ensemble-based, and function-prediction methods.

The protein-based subcategory was adopted by 11 papers. Some works (CARTER et al., 2009; TAN; BAO; ZHOU, 2012; MAO et al., 2013) evaluated the changes in residues' charge, volume, polarity, the hydrophobicity resulting from the mutation, the predicted residue solvent accessibility and backbone flexibility, the mutation's effect on protein stability, and the probability that the secondary structure of the wild type residue's

Table 3.3 – *In silico* tools for functional impact prediction

Reference	Tools
Capriotti and Altman (2011)	PANTHER (THOMAS et al., 2006)
Davoli et al. (2013)	Polyphen-2 (ADZHUBEI et al., 2010)
Mao et al. (2013)	SIFT (NG; HENIKOFF, 2003), PolyPhen-2 (ADZHUBEI et al., 2010), CONDEL (GONZÁLEZ-PÉREZ; LÓPEZ-BIGAS, 2011), MutationAssessor (REVA; ANTIPIN; SANDER, 2011), PhyloP (POLLARD et al., 2010), GERP++ (DAVYDOV et al., 2010), and LRT (CHUN; FAY, 2009)
Schroeder et al. (2014)	OncodriveFM (GONZALEZ-PEREZ; LOPEZ-BIGAS, 2012)
Gnad et al. (2015)	MutationAssessor (REVA; ANTIPIN; SANDER, 2011)
Dong et al. (2016)	SIFT (NG; HENIKOFF, 2003), PolyPhen-2 (ADZHUBEI et al., 2010), LRT (CHUN; FAY, 2009), MutationTaster (SCHWARZ et al., 2010), MutationAssessor (REVA; ANTIPIN; SANDER, 2011), FATHMM (SHIHAB et al., 2013), GERP++ (DAVYDOV et al., 2010), PhyloP (POLLARD et al., 2010), CADD (KIRCHER et al., 2014), VEST (CARTER et al., 2013), SiPhy (GARBER et al., 2009), FunSeq2 (FU et al., 2014)
Tokheim et al. (2016)	VEST (CARTER et al., 2013)
Agajanian et al. (2018)	SIFT (NG; HENIKOFF, 2003), PolyPhen-2 (ADZHUBEI et al., 2010), LRT (CHUN; FAY, 2009), MutationAssessor (REVA; ANTIPIN; SANDER, 2011), MutationTaster (SCHWARZ et al., 2010), FATHMM (SHIHAB et al., 2013), MSRV (JIANG et al., 2007), SinBaD (LEHMANN; CHEN, 2013), GERP++ (DAVYDOV et al., 2010), SiPhy (GARBER et al., 2009), PhyloP (POLLARD et al., 2010), Grantham (GRANTHAM, 1974), CADD (KIRCHER et al., 2014), GWAVA (RITCHIE et al., 2014), MetaLR (DONG et al., 2015), and MetaSVM (DONG et al., 2015)
Wang et al. (2018b)	SIFT (NG; HENIKOFF, 2003) and GERP++ (DAVYDOV et al., 2010)
Zhou, Gao and Skolnick (2018)	ENTPRISE (ZHOU; GAO; SKOLNICK, 2016)
Agajanian, Oluyemi and Verkhivker (2019)	SIFT (NG; HENIKOFF, 2003), PolyPhen-2 (ADZHUBEI et al., 2010), LRT (CHUN; FAY, 2009), MutationAssessor (REVA; ANTIPIN; SANDER, 2011), MutationTaster (SCHWARZ et al., 2010), FATHMM (SHIHAB et al., 2013), MSRV (JIANG et al., 2007), SinBaD (LEHMANN; CHEN, 2013), GERP++ (DAVYDOV et al., 2010), SiPhy (GARBER et al., 2009), PhyloP (POLLARD et al., 2010), Grantham (GRANTHAM, 1974), CADD (KIRCHER et al., 2014), GWAVA (RITCHIE et al., 2014), MetaLR (DONG et al., 2015), and MetaSVM (DONG et al., 2015)
Collier, Stoven and Vert (2019)	Polyphen-2 (ADZHUBEI et al., 2010)
Zhu et al. (2019)	20/20+ (TOKHEIM et al., 2016), MutSigCV (LAWRENCE et al., 2013), OncodriveFM (GONZALEZ-PEREZ; LOPEZ-BIGAS, 2012), OncodriveCLUST (TAMBORERO; GONZALEZ-PEREZ; LOPEZ-BIGAS, 2013), DrGaP (HUA et al., 2013), and MUFFINN (CHO et al., 2016)
Colaprico et al. (2020)	VEST (CARTER et al., 2013)
Gumpinger et al. (2020)	MutsigCV (LAWRENCE et al., 2013)
Lyu et al. (2020)	VEST (CARTER et al., 2013), PolyPhen-2 (ADZHUBEI et al., 2010)
Wang et al. (2020)	GERP++ (DAVYDOV et al., 2010), PhastCons, PhyloP (POLLARD et al., 2010), LRT (CHUN; FAY, 2009), SiPhy (GARBER et al., 2009), FATHMM (SHIHAB et al., 2013), fitCons (GULKO et al., 2015), MutationAssessor (REVA; ANTIPIN; SANDER, 2011), MutationTaster (SCHWARZ et al., 2010), PolyPhen2-HDIV (ADZHUBEI et al., 2010), PolyPhen2-HVAR (ADZHUBEI et al., 2010), PROVEAN (CHOI; CHAN, 2015), SIFT (NG; HENIKOFF, 2003), VEST3 (CARTER et al., 2013), CADD (KIRCHER et al., 2014), DANN (QUANG; CHEN; XIE, 2015), Eigen (IONITA-LAZA et al., 2016), FATHMM-MKL (SHIHAB et al., 2015), GenoCanyon (LU et al., 2015), M-CAP (JAGADEESH et al., 2016), MetaLR (DONG et al., 2015), MetaSVM (DONG et al., 2015), and REVEL (IOANNIDIS et al., 2016)

region is helix, loop, or strand. Anoosha et al. (2015) considered 49 physical, chemical, energetic, and conformational parameters comparing wild type and mutant residues, as well as neighboring residue information at different window lengths. Feature vectors encoding the mutated sequence and the mutation's local sequence environment (CAPRIOTTI; ALTMAN, 2011) or the biochemical properties of the atomic coordinates of proteins' PDB structure (TAVANAEI et al., 2017) were also proposed. Other characteristics considered were the fraction of the affected protein structure (ZHOU; GAO; SKOLNICK, 2018), amino acid composition of the residues in contact with the mutation or of its domain (ZHOU; GAO; SKOLNICK, 2018), amino acid's position in the codon or protein

(AGAJANIAN; OLUYEMI; VERKHIVKER, 2019), number of complexes containing the protein (NULSEN et al., 2021), and background probability for observing the wild type or mutant residue in the first, middle, or last position of an amino acid triple, or at the center of a window of 5 amino-acid residues (CARTER et al., 2009; TAN; BAO; ZHOU, 2012). Finally, amino acids substitution scores were employed by several studies (CARTER et al., 2009; TAN; BAO; ZHOU, 2012; MAO et al., 2013; U et al., 2014; ANOOSHA et al., 2015), most of which integrated distinct substitution scoring matrices.

The evolution-based subcategory was employed in 14 studies, most of which computed evolutionary conservation scores using distinct strategies or tools (CARTER et al., 2009; CAPRIOTTI; ALTMAN, 2011; MAO et al., 2013; U et al., 2014; ANOOSHA et al., 2015; TOKHEIM et al., 2016; AGAJANIAN et al., 2018; ZHOU; GAO; SKOLNICK, 2018; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; COLAPRICO et al., 2020; LYU et al., 2020). MGAentropy, for instance, was employed by three papers (MAO et al., 2013; TOKHEIM et al., 2016; LYU et al., 2020). The rationale is that the more an amino-acid residue is functionally or structurally important, the more conserved is over evolution. Chandrashekar et al. (2020) computed the mean substitution rate of all protein's positions as well as of positions under the highest peak of closely-spaced mutations. Lyu et al. (2020) employed the gene age, gene damage index, and the number of human paralogs for each gene, among others. Finally, Nulsen et al. (2021) included the indication of genes' evolutionary origin (*i.e.,* pre-metazoan, metazoan, vertebrate, or post-vertebrate).

We observed that six studies (U et al., 2014; ANOOSHA et al., 2015; TAVANAEI et al., 2017; AGAJANIAN et al., 2018; ZHU et al., 2019; WANG et al., 2020) in the *Functional Impact* category only used this data type for developing their predictive models (Figure 3.2a). Among these, Zhu et al. (2019) and Wang et al. (2020) defined their features exclusively based on scores from FI predictors. This is not surprising, as this feature carries rich underlying information provided by the various properties analyzed by each FI predictor.

3.4.3 Functional genomics

About 39% of papers used *Functional Genomics* features, which we divided in three subcategories (Table 3.2): i) transcriptomics; ii) epigenomics; and iii) proteomics. The motivation comes from the observation that mutation frequency is strongly cor-

related with transcriptional activity and DNA replication timing (JIANG et al., 2019; LAWRENCE et al., 2013), and that driver gene mutations are tightly tied to DNA methylation landscape in multiple types of cancer (CHEN et al., 2017). Moreover, mutation rates vary among individual genes and are influenced by many factors, including the chromatin state (SCHUSTER-BÖCKLER; LEHNER, 2012) and the aforementioned ones (*i.e.,* gene expression, replication timing, DNA methylation). Thus, integrating these properties in the ML models may help differentiate cancer genes from the rest of human genes (NULSEN et al., 2021).

Transcriptomic data was employed in 15 papers, representing 93.75% of papers classified as *Functional Genomics*. Gene expression levels were provided as model inputs in several works (FU et al., 2012; MANOLAKOS et al., 2014; PARK et al., 2015; GUAN et al., 2018). Other papers summarized gene expression profiles by computing differential gene expression scores (GNAD et al., 2015; TOKHEIM et al., 2016; LU et al., 2018; WANG et al., 2018b; SCHULTE-SASSE et al., 2019; COLAPRICO et al., 2020; LYU et al., 2020) or average expression in cancer tissues or cell lines (JIANG et al., 2019). In Nulsen et al. (2021), authors quantified the number of tissues expressing each gene, as well as binary features indicating whether the gene is expressed in a certain number of tissues. We observed that 11 studies used only transcriptomic data, while three (JIANG et al., 2019; SCHULTE-SASSE et al., 2019; LYU et al., 2020) combined transcriptomics and epigenomics, and one (NULSEN et al., 2021) combined transcriptomic- and proteomic-based features.

Five papers (TOKHEIM et al., 2016; CELLI; CUMBO; WEITSCHEK, 2018; JIANG et al., 2019; SCHULTE-SASSE et al., 2019; LYU et al., 2020) used features from the epigenomic domain. Despite the low number of papers, a wide range of information was registered. In Tokheim et al. (2016), authors used as features the DNA replication time and the 3D chromatin interaction capture (HiC) statistic, which is a measure of open vs. closed chromatin state, obtained from the MutSigCV webpage (LAWRENCE et al., 2013). Jiang et al. (2019) complemented these features with chromatin accessibility by ATAC-Seq data and beta values obtained from DNA methylation data. In Schulte-Sasse et al. (2019), authors analyzed 16 types of cancer from TCGA and represented each gene by a 16×3-dimensional vector, containing estimates of differential DNA methylation from the gene's promoter region, differential gene expression, and gene's mutation rate in each type of cancer analyzed. Lyu et al. (2020) explored several types of epigenetic properties, including early replication timing quantified by the S50 score, promoter

and gene-body cancer–normal methylation difference, and histone modifications from the ENCODE project. Finally, one paper (CELLI; CUMBO; WEITSCHEK, 2018) focused exclusively on DNA methylation data, using the beta values as model's predictors.

Lastly, only one paper (NULSEN et al., 2021) was classified in the proteomics subcategory, using features that reflect the number of healthy human tissues expressing a protein and whether a protein is expressed in 0 to 8 tissues, or in 41 or more tissues according to the Protein Atlas v18.

3.4.4 Ontology-based

In the *Ontology-based* category, annotations varied from Gene Ontology (GO) terms to specific categorization of genes' role in organisms' functioning and phenotype (*e.g.,* essentiality, diseases involvement, cellular localization). The rationale is that prior knowledge regarding genes associated with diseases or molecular processes implicated in cancer may help improve gene prioritization in the search for CDGs. Nine papers (21.95%) were classified within this subcategory. While most papers used ontology-based features in combination to other data types, one paper employed exclusively background knowledge about gene functions, cellular locations, and cellular and organism phenotypes obtained from Cellular Microscopy Phenotype Ontology (CMPO), Gene Ontology (GO), and Mammalian Phenotype Ontology (MP) databases to learn an embedding for each gene using a neuro-symbolic approach (ALTHUBAITI et al., 2019).

Information from biological processes was integrated into two frameworks (CAPRIOTTI; ALTMAN, 2011; COLAPRICO et al., 2020), either as features encoding the number of GO terms associated with a given gene (CAPRIOTTI; ALTMAN, 2011) or as prior knowledge about biological processes linked to cancer to identify their mediators (COLAPRICO et al., 2020). Prior knowledge regarding disease involvement of protein under altered function was also explored by selected works, using, for instance, the databases Phenolyzer (DONG et al., 2016) and GeneCards (ZHOU; GAO; SKOLNICK, 2018). Moreover, two papers adopted protein essentiality annotations based on the rationale that functional changes in essential proteins are more likely to be associated with diseases (ZHOU; GAO; SKOLNICK, 2018; NULSEN et al., 2021).

Annotations about regulatory roles or interactions were integrated into the models proposed by two works (MANOLAKOS et al., 2014; NULSEN et al., 2021). In the first (MANOLAKOS et al., 2014), authors explored transcription factors obtained from

the Human Protein Reference Database (HPRD) (VAQUERIZAS et al., 2009) to identify cancer-related modules formed by regulatory genes and their downstream targets. In the second (NULSEN et al., 2021), authors included the number of microRNAs targeting a given gene (NULSEN et al., 2021) registered at miRTarBase (CHOU et al., 2018) and miRecords (XIAO et al., 2009) databases, motivated by the hypothesis that genes related to canonical driver are targeted by more microRNAs. Finally, one work (LYU et al., 2020) used annotations regarding super-enhancer from the dbSUPER database (KHAN; ZHANG, 2016) and cell proliferation scores as a proxy for essentiality.

3.4.5 Network-based

The *Network-based* category appeared in nine papers. In biological systems, the interactions between proteins are essential for the comprehension of cell physiology since most proteins interact with others for proper biological activity. This also implies that any abnormality in a given gene or protein may spread through its links in the molecular network and impact the activity of other elements. Thus, the hypothesis that a disease phenotype is rarely a consequence of a defect on a single gene or protein but rather of alterations in the biological processes that interact in a complex network has motivated network-based approaches to study human diseases (BARABÁSI; GULBAHCE; LOSCALZO, 2011).

The predictive features in this domain were mainly related to centrality measures obtained from Protein-protein interaction (PPI) networks (Table 3.4), such as degree, betweenness, and clustering coefficient (TOKHEIM et al., 2016; ZHOU; GAO; SKOLNICK, 2018; COLAPRICO et al., 2020; CUTIGI et al., 2020; NULSEN et al., 2021). The hypothesis is that proteins encoded by canonical drivers tend to have higher centrality than other proteins (NULSEN et al., 2021). Cutigi et al. (2020) considered several other node properties: closeness, eigenvector, coreness, average of neighbors' degree, leverage, information, and bridging. Nulsen et al. (2021) defined a binary feature indicating whether a protein is a hub (*i.e.,* top 25% of degree distribution) or not in the PPI network.

Park et al. (2015) leveraged gene networks constructed based on comprehensive genome-scale information, including PPIs (unspecified sources), gene expression, SNVs, and CNA. Collier, Stoven and Vert (2019) quantified the gene-gene similarity using an integrated kernel function that combines prior information about mutations and PPI net-

Table 3.4 – Protein-protein interaction (PPI) networks adopted by selected papers

Reference	**PPI Networks**
Tokheim et al. (2016)	BioGRID (STARK et al., 2006)
Zhou, Gao and Skolnick (2018)	HIPPIE (SCHAEFER et al., 2012)
Collier, Stoven and Vert (2019)	HPRD (PRASAD et al., 2009)
Schulte-Sasse et al. (2019)	ConsensusPathDB (KAMBUROV et al., 2011)
Cutigi et al. (2020)	ReactomeFI (JASSAL et al., 2020), HINT (DAS; YU, 2012), HPRD (PRASAD et al., 2009), HuRI (LUCK et al., 2020)
Gumpinger et al. (2020)	InBio Map (LAGE et al., 2007)
Nulsen et al. (2021)	BioGRID (STARK et al., 2006), MIntAct(ORCHARD et al., 2014), DIP (SALWINSKI et al., 2004), HPRD (PRASAD et al., 2009)

work. In Schulte-Sasse et al. (2019), instead of exploring handcrafted network-based features, authors adopted the Graph Convolutional Network (GCN) algorithm that can directly analyze graph-structured data and recognize patterns in a local neighborhood of a node, using the PPI network as the input graph. Similarly, Gumpinger et al. (2020) generated node embeddings that integrate the PPI network structure with nodes' MutSig p-values. Their findings suggest that nodes' context in the network introduces valuable information for CDGs prediction.

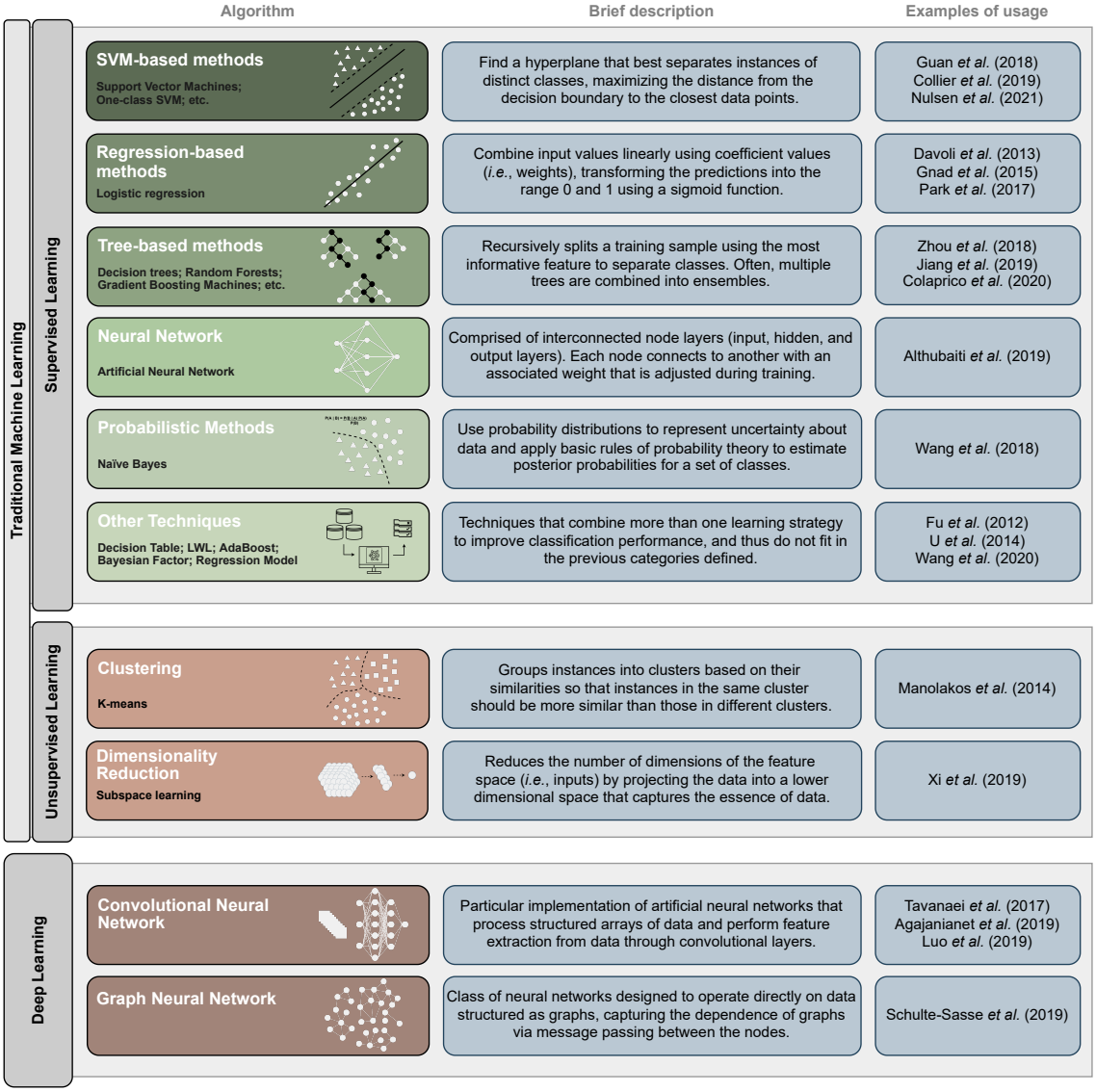
3.5 Machine learning strategies

Regarding computational methods, we categorized the selected papers into traditional ML and deep learning (DL). Traditional ML was further subdivided into supervised and unsupervised methods in our analysis (Figure 3.3), while DL may comprise supervised, unsupervised and semi-supervised approaches, which were considered here as a single category. A theoretical background for ML was provided in Chapter 2.

Two papers follow distinct methodologies: the first uses a genetic algorithm (LI et al., 2016), which is a population-based nature-inspired learning algorithm, and the second is a web-based consensus CDG caller that performs rank-based aggregation of FI predictors' scores, including ML-based tools (ZHU et al., 2019). We will not give emphasis to these works in our review.

Most papers (80%) adopted a traditional supervised learning approach, exploring the known examples of true cancer drivers that are available in specialized databases (Table 3.5) as training data. The COSMIC database (TATE et al., 2018) and its CGC resource (SONDKA et al., 2018) were the most common sources for positive labels (*i.e.,* drivers), followed by TCGA (significantly mutated genes) and NCG (REPANA et al.,

Figure 3.3 – Classes and examples of machine learning algorithms identified through this work with applications in cancer driver gene prediction.



Source: Prepared by the author

2019). Some works restricted the selection for CDGs verified by at least two sources (COLAPRICO et al., 2020). High-quality negative examples (*i.e.,* non-drivers) are, in general, more challenging to define when applying ML to omics data (LIBBRECHT; NOBLE, 2015; SCHUBACH et al., 2017) since the choice is highly prone to biased predictions or false negatives depending on filtering criteria applied (ROGERS; GAUNT; CAMPBELL, 2020), and no comprehensive ground truth datasets are available. Selected papers explored mainly neutral variants from public genomic variation databases (CAPRIOTTI; ALTMAN, 2011; TAN; BAO; ZHOU, 2012; U et al., 2014) and synthetically generated passenger missense mutations (CARTER et al., 2009; SOLIMAN et al., 2015). Another adopted approach is to start from all genes and recursively remove positive la-

beled genes according to a compendium of annotation databases (SCHULTE-SASSE et al., 2019; WANG et al., 2020). Many papers (SOLIMAN et al., 2015; TOKHEIM et al., 2016; AGAJANIAN et al., 2018; COLLIER; STOVEN; VERT, 2019; LUO et al., 2019; NICORA et al., 2019; CHANDRASHEKAR et al., 2020; COLAPRICO et al., 2020) also collected labeled datasets from the scientific literature (*e.g.,* (CARTER et al., 2009; TAMBORERO et al., 2013; VOGELSTEIN et al., 2013; MARTELOTTO et al., 2014; BAILEY et al., 2018)).

Table 3.5 – Databases adopted in the selected papers for constructing labeled datasets of positive and negative examples of cancer drivers

Sources for positive examples	
Database	**References**
COSMIC (TATE et al., 2018) or Cancer Gene Census (CGC) (SONDKA et al., 2018)	(CARTER et al., 2009; CAPRIOTTI; ALTMAN, 2011; TAN; BAO; ZHOU, 2012; MAO et al., 2013; SCHROEDER et al., 2014; U et al., 2014; ANOOSHA et al., 2015; DONG et al., 2016; TAVANAEI et al., 2017; JIANG et al., 2019; ZHU et al., 2019; LUO et al., 2019; COLAPRICO et al., 2020; CHANDRASHEKAR et al., 2020; GUMPINGER et al., 2020; LYU et al., 2020)
TCGA	(MAO et al., 2013; DONG et al., 2016; WANG et al., 2018b; ZHU et al., 2019; CHANDRASHEKAR et al., 2020; NULSEN et al., 2021)
Network of Cancer Genes (NCG) (REPANA et al., 2019)	(SCHULTE-SASSE et al., 2019; CUTIGI et al., 2020; NULSEN et al., 2021)
OMIM (HAMOSH et al., 2005)	(FU et al., 2012; ZHU et al., 2019)
intOGen (GONZALEZ-PEREZ et al., 2013)	(ALTHUBAITI et al., 2019; HAN et al., 2019)
ClinVar (LANDRUM et al., 2018)	(ZHOU; GAO; SKOLNICK, 2018)
CBioPortal (CERAMI et al., 2012)	(AGAJANIAN; OLUYEMI; VERKHIVKER, 2019)
DriverDBv2 (CHUNG et al., 2016)	(HAN et al., 2019)
OncoKB (CHAKRAVARTY et al., 2017)	(WANG et al., 2020)
FASMIC (NG et al., 2018)	(WANG et al., 2020)
Sources for negative examples	
Database	**References**
Synthethic examples	(CARTER et al., 2009; CAPRIOTTI; ALTMAN, 2011)
Swiss-Prot Variant Pages (YIP et al., 2008)	(CAPRIOTTI; ALTMAN, 2011; TAN; BAO; ZHOU, 2012)
CCLE (BARRETINA et al., 2012)	(MAO et al., 2013)
SNP@Domain (HAN et al., 2006)	(U et al., 2014)
COSMIC (TATE et al., 2018)	(ANOOSHA et al., 2015)
CBioPortal (CERAMI et al., 2012)	(AGAJANIAN; OLUYEMI; VERKHIVKER, 2019)
intOGen (GONZALEZ-PEREZ et al., 2013)	(ALTHUBAITI et al., 2019)
OncoKB (CHAKRAVARTY et al., 2017)	(WANG et al., 2020)
FASMIC (NG et al., 2018)	(WANG et al., 2020)
gnomAD (KARCZEWSKI et al., 2020)	(WANG et al., 2020)

Three papers (MANOLAKOS et al., 2014; LU et al., 2018; XI et al., 2019)

adopted traditional unsupervised learning and four papers explored DL (TAVANAEI et al., 2017; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; LUO et al., 2019; SCHULTE-SASSE et al., 2019).

To provide a more granular discussion about traditional supervised learning approaches, we further divided them into six subcategories: SVM-based methods, regression-based methods, tree-based methods, neural network, probabilistic methods, and other approaches that do not fall into any of the previous classes. Figure 3.4a shows the algorithm categories distribution across the year of paper publication. In Figure 3.4b, we summarize the observed associations between ML strategies and data categories.

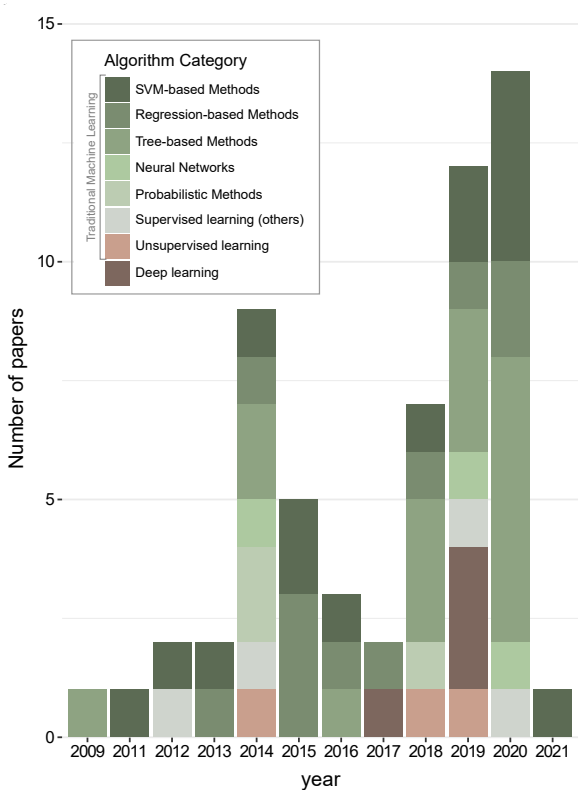
3.5.1 Methods based on traditional supervised learning

Tree-based and SVM-based methods were the most common supervised learning approaches, observed in 39.02% and 36.59% of selected papers. The first proposals by Carter et al. (2009) and Capriotti and Altman (2011) were based on these algorithms. Among the SVM-based approaches, whereas most papers adopted the traditional SVM algorithm (CAPRIOTTI; ALTMAN, 2011; TAN; BAO; ZHOU, 2012; MAO et al., 2013; U et al., 2014; SOLIMAN et al., 2015; DONG et al., 2016; GUAN et al., 2018; CUTIGI et al., 2020; GUMPINGER et al., 2020; LYU et al., 2020; WANG et al., 2020), we observed three papers using OneClass SVM (COLLIER; STOVEN; VERT, 2019; NICORA et al., 2019; NULSEN et al., 2021) and one paper using Sequential Minimal Optimization (SMO) (ANOOSHA et al., 2015). SVM is a popular and consolidated technique in the field, as it continues to be largely applied since 2011.

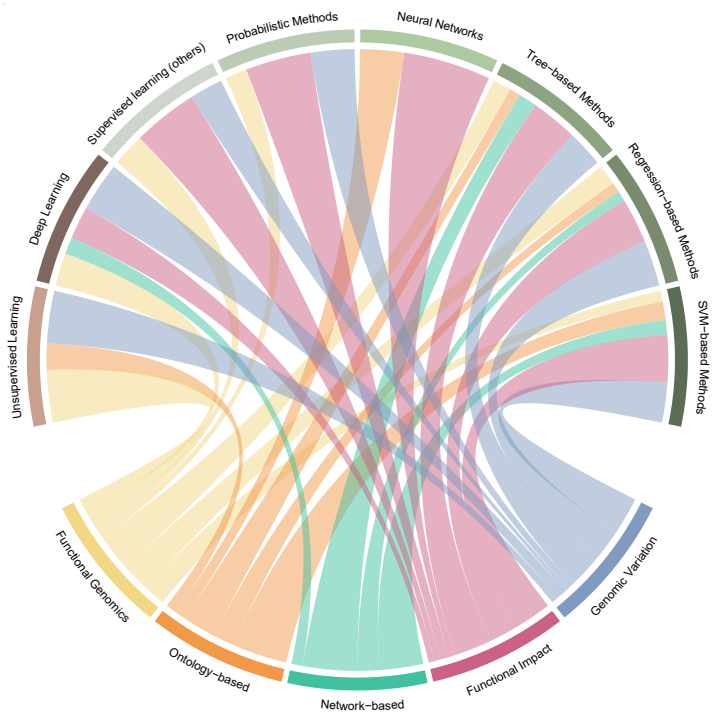
Considering tree-based methods, although some papers explored the traditional decision tree algorithm (SCHROEDER et al., 2014; U et al., 2014; WANG et al., 2020), the vast majority used tree-based ensemble classifiers. Ensemble methods train multiple weak classifiers, such as decision trees, and combine their output (*e.g.,* with majority voting) to achieve a better predictive performance. We observed a frequent use of Random Forests (RF) (CARTER et al., 2009; SCHROEDER et al., 2014; U et al., 2014; TOKHEIM et al., 2016; AGAJANIAN et al., 2018; CELLI; CUMBO; WEITSCHEK, 2018; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; NICORA et al., 2019; CHANDRASHEKAR et al., 2020; COLAPRICO et al., 2020; CUTIGI et al., 2020; GUMPINGER et al., 2020; LYU et al., 2020; WANG et al., 2020), as well as Gradient Boosting Trees (GBT) (ZHOU; GAO; SKOLNICK, 2018; AGAJANIAN et al., 2018), and eXtreme Gra-

Figure 3.4 – Analysis of types of algorithms used for model development in the selected papers. (a) Distribution of the number of occurrences found for each algorithm category per year of publication. (b) Association between data categories and algorithm categories.

(a)



(b)



Source: Prepared by the author

dient Boosting (XGBoost) (LYU et al., 2020; WANG et al., 2020). Other less common tree-based variants were also explored, including Conditional trees, NB trees, and Functional trees (SCHROEDER et al., 2014).

Regression-based methods appeared in 11 selected papers, most of which adopted logistic regression (SCHROEDER et al., 2014; GNAD et al., 2015; SOLIMAN et al., 2015; DONG et al., 2016; AGAJANIAN et al., 2018; GUMPINGER et al., 2020; LYU et al., 2020). We also found papers using Ridge (NICORA et al., 2019) and Lasso (DAVOLI et al., 2013) regularized regressions.

The works by Park *et al.* (PARK et al., 2015; PARK et al., 2017) introduced new approaches based on linear regression and regularization schemes. Authors proposed a sparse overlapping group Lasso to perform group selection while identifying crucial genes (*i.e.,* potential CDGs) within each group (PARK et al., 2015), and an interaction-based feature-selection strategy with adaptive regularization that adjusts the amount of the L1-type penalty imposed on each gene proportionally to the degree to which gene expression alteration is explained by CNAs (PARK et al., 2017).

Probabilistic methods and artificial neural networks (ANN) were less frequent among selected papers. Naïve Bayes was applied in two studies (SCHROEDER et al., 2014; U et al., 2014), while a Bayesian network was adopted only in one (U et al., 2014). U et al. (2014) used both the traditional implementation of naïve Bayes, as well as the DTNB algorithm that combines naïve Bayes with induction of decision tables (HALL; FRANK, 2008). Wang et al. (2018b) proposed a Bayesian hierarchical modeling approach to identify mutations that best predict alterations in gene expression, which represent candidate drivers. Furthermore, ANNs were used in three works (U et al., 2014; ALTHUBAITI et al., 2019; WANG et al., 2020).

Althubaiti et al. (2019) and Wang et al. (2020) trained an ANN with two hidden layers using a Rectified Linear Unit (ReLU) as an activation function for the hidden layers. In the former, the network receives as input embedding vectors generated from different ontologies, while in the latter, the input vectors are built from the pathogenicity prediction scores provided by *in silico* tools.

We highlight three works that explored the greatest number of algorithms. U et al. (2014) compared 11 algorithms, including SVM-based, tree-based, ANN, and probabilistic methods. SVM presented the best performance among the classifiers, but a weighted voting approach among all models achieved the most robust results. Schroeder et al. (2014) compared six regression-based, probabilistic, and tree-based methods, and ob-

served that RF produced the highest predictive performance. Wang et al. (2020) analyzed seven algorithms, including SVM-based, tree-based, and ANN, and pointed XGBoost as the top-performing method.

We also identified the use of less conventional supervised learning methods, including the Bayesian Factor Regression Modeling (BFRM), a sparse statistical model for high-dimensional data analysis (FU et al., 2012), Decision Tables, and Locally Weighted Learning (LWL) (U et al., 2014).

Finally, Han et al. (2019) applied a ML principle, but the weight parameters in the proposed score are learned by maximizing their global weighted score stats from previous genes in the training mutation data.

3.5.2 Methods based on traditional unsupervised learning

Three papers applied unsupervised methods to CDGs prediction. Manolakos et al. (2014) proposed an algorithm for detecting gene drivers using clustering to find genes with similar expression profiles across cancer patients. Gene clusters are created using K-means, and then sparse representations of genes in a given cluster are identified as a linear combination of a small number of regulatory genes, pointed as potential cancer drivers. An Expectation-Maximization technique was used by Lu et al. (2018) in their module-based framework integrating transcriptome and genome. To identify CDGs, authors proposed an iterative approach that determines the best modulators to explain gene expression profiles of genes in a given module and re-assigns each gene to the module whose associated regulation program best predicts its behavior. In Xi et al. (2019), a subspace learning framework was employed to obtain a low-dimensional vectorized representation of unannotated genes using a binary mutation matrix as input. Driver genes can be discriminated evaluating the distances between the output vectors and the origin in the low-dimensional subspace, with the top-ranked genes being promising driver candidates.

3.5.3 Methods based on deep learning

Despite the increasing use of DL in Bioinformatics, only four (TAVANAEI et al., 2017; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; LUO et al., 2019; SCHULTE-

SASSE et al., 2019) papers adopted this class of algorithms. Convolutional neural networks (CNN) were used in three papers (TAVANAEI et al., 2017; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; LUO et al., 2019) following a supervised approach. Tavanaei et al. (2017) developed a parallel CNN with three branches followed by a multi-layer fully connected neural network. Each branch receives a projection of a 3-D protein structure to a 2-D feature map set for 16 features associated with the atomic coordinates $< x, y, z >$. The input data is processed by four convolution and pooling layers adopting the ReLU activation function and three fully connected layers. Luo et al. (2019) trained a 1-D CNN with a mutation-based feature matrix as input. Authors varied the number of convolutional layers (1-4) and the number of fully connected layers (1-3), determining the best hyperparameters by grid search.

Agajanian, Oluyemi and Verkhivker (2019) trained a CNN model that processes encoded raw nucleotide sequences, using grid search to optimize its architecture. The authors evaluated label encoding (*i.e.,* each nucleotide receives a unique integer ID), word2vec embedding (*i.e.,* each nucleotide receives a numeric representation in a vector space based on the analysis of its sequential context), and one-hot encoding (*i.e.,* each nucleotide receives a bit encoded string), with the latter achieving best predictive performance. Their approach has the attractive property of combining high-level DNA-derived scores computed by the CNN with 32 features obtained from FI predictors as inputs for an RF model, resulting in performance improvement.

Finally, Schulte-Sasse et al. (2019) used an extension of the CNN framework designed to handle graph-structured data as input, named Graph Convolutional Network (GCN) (KIPF; WELLING, 2016). The GCN directly classifies the nodes of a network based on the network structure (*e.g.,* PPI network) and nodes' characteristics (*e.g.,* gene mutation rates, gene expression, and DNA methylation) using semi-supervised learning. Their optimal GCN was composed of two graph convolutional layers with 50 and 100 filters, respectively.

3.5.4 Feature selection

Feature selection (FS) are dimensionality reduction approaches to reduce the input feature space by removing irrelevant, redundant, or noisy features, as comprehensively reviewed by Li et al. (2017). FS helps in decreasing the chance of overfitting, improving performance, and interpreting the patterns learned by models. Several papers explored

FS in their methodology (CARTER et al., 2009; TAN; BAO; ZHOU, 2012; DAVOLI et al., 2013; MAO et al., 2013; U et al., 2014; ANOOSHA et al., 2015; GNAD et al., 2015; PARK et al., 2015; SOLIMAN et al., 2015; DONG et al., 2016; PARK et al., 2017; TAVANAEI et al., 2017; AGAJANIAN et al., 2018; GUAN et al., 2018; ZHOU; GAO; SKOLNICK, 2018; LYU et al., 2020; NULSEN et al., 2021). Among these, we observed the use of mutual information (CARTER et al., 2009), DX score (TAN; BAO; ZHOU, 2012), Lasso regression (DAVOLI et al., 2013), Mann–Whitney U test (MAO et al., 2013), and Chi-square (SOLIMAN et al., 2015).

U et al. (2014) applied five FS algorithms (*i.e.,* oneR, reliefF, Chi-square, gain ratio, and correlation-based) and retained all the top features indicated by at least three algorithms. Mao et al. (2013) evaluated all possible combinations with fewer than four features using k-fold cross-validation and the best feature subset was then expanded using a hill-climbing strategy to iteratively include the remaining features into the combination. Recursive feature elimination was employed by two papers (AGAJANIAN et al., 2018; ZHOU; GAO; SKOLNICK, 2018). Also, some papers analyzed feature's importance separately for oncogenes and TSGs (DAVOLI et al., 2013; GNAD et al., 2015; LYU et al., 2020). Davoli et al. (2013) identified the LOF/benign ratio and the missense entropy as the best features to discriminate among oncogenes and TSGs in their model. Lyu et al. (2020) grouped correlated features using hierarchical clustering prior to FS, identifying three and five relevant feature groups for TSGs and OGs.

3.5.5 Model validation protocols

We analyzed the model validation protocols among traditional supervised learning and DL methods. This is crucial in ML methodology since an improper validation may lead to over-optimistic expectations regarding the model's performance. Two papers (TAVANAEI et al., 2017; CELLI; CUMBO; WEITSCHEK, 2018) used the basic holdout method, *i.e.,* a simple division of the original dataset into train and test sets. A drawback of the holdout method is that the measured performance can highly depend on the instances included in the train and test datasets. Tavanaei et al. (2017) repeated the holdout process three times using different random seeds to observe performance variation; nonetheless, standard deviations were not reported. About 70.7% of papers used the k-fold cross-validation process, which despite demanding more computational power and time to run, it is the most recommended approach for model validation under limited

data. Besides providing a more robust performance assessment based on k evaluations, it ensures that every observation from the original dataset has the chance of appearing in the train and test set. Finally, some papers (MANOLAKOS et al., 2014; AGAJANIAN et al., 2018; AGAJANIAN; OLUYEMI; VERKHIVKER, 2019; LUO et al., 2019; NICORA et al., 2019) combined the holdout and the k-fold cross-validation methods: the k-fold cross-validation is run over the train set derived from the holdout, and the best model built during this process is further applied over the test set. This is an interesting approach, as the performance obtained for the test set is more reliable since it is guaranteed to be free from data leakage.

Additionally, some papers evaluated their predictive models with independent test sets (TAN; BAO; ZHOU, 2012; MAO et al., 2013; SCHROEDER et al., 2014; ANOOSHA et al., 2015; SOLIMAN et al., 2015; DONG et al., 2016; TAVANAEI et al., 2017; ZHOU; GAO; SKOLNICK, 2018; COLLIER; STOVEN; VERT, 2019; HAN et al., 2019; LUO et al., 2019; COLAPRICO et al., 2020; LYU et al., 2020; WANG et al., 2020; NULSEN et al., 2021). For instance, Collier, Stoven and Vert (2019) used a subset of cancer genes from the Cancer Gene Census (CGC) v86 that was not included in the development of previous models. Han et al. (2019) analyzed the ability of their model to recover a set of high confidence cancer genes manually curated from literature. Nulsen et al. (2021) evaluated their method with an independent cancer cohort data of osteosarcoma, a rare bone cancer, and found that it was able to identify reliable cancer drivers in individual patients even for cancer types not used for training. Interestingly, some papers carried out experimental analyses of selected findings to characterize mutations and their functional impact (U et al., 2014; WANG et al., 2018b; HAN et al., 2019).

3.5.6 Predictive performance of ML strategies

The performance of traditional supervised learning and DL methods was assessed mainly using accuracy, the area under the ROC curve (AUC-ROC), recall (*i.e.,* sensitivity), and precision. Some papers also summarized the trade-off between precision and recall reporting the area under the precision-recall curve (AUC-PR) and the F1-score (*i.e.,* the harmonic mean between precision and recall, attaching equal weights to both false positives and false negatives). Moreover, three papers reported the Matthew's Correlation Coefficient (MCC) (SOLIMAN et al., 2015; ZHOU; GAO; SKOLNICK, 2018; SCHULTE-SASSE et al., 2019) and one paper (TOKHEIM et al., 2016) proposed the

mean absolute log2 fold change (MLFC) metric, which quantifies the deviation between the theoretically expected p-values and the p-values generated by a method. Although some unsupervised approaches were evaluated with the above metrics (LU et al., 2018; XI et al., 2019), others within this category used Jaccard index, R^2, and Pearson correlation (MANOLAKOS et al., 2014).

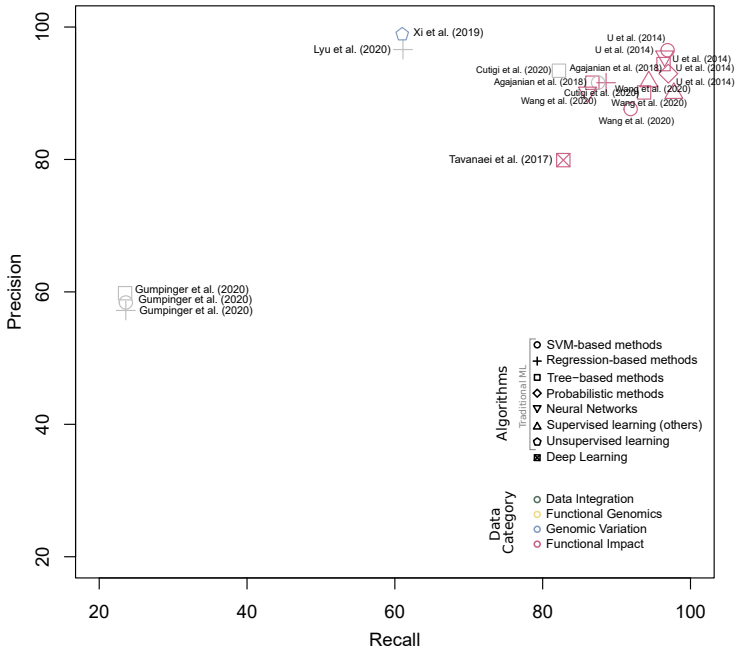
We summarized methods' predictive performance for common metrics, as reported by the authors in their original publication (Figure 3.5). Since these models were evaluated on different benchmarks and for different cancer types, it is improper to compare them directly. Thus, our analysis aims to provide an overview of the predictive power achieved by ML-based methods. Although it may guide methodological choices in future works, it should not be used for drawing hard conclusions about top-performing approaches. Observing the trade-off between precision and recall (Figure 3.5a), the predictive performance does not seem to be segregated by algorithm type but rather by the features type. The work by U et al. (2014) is a good example for this observation. Authors used five algorithm classes to train models with the same feature vector and found a slight variation among their performances. However, this hypothesis should be confirmed in a more profound analysis using the same experimental settings for different methods.

A summary of accuracy and AUC-ROC values (Figure 3.5b) suggests that SVM-based and Tree-based methods had consistent performance, with most models surpassing the 90.00 mark. ANN and other supervised learning techniques, although not as frequent as others, also resulted in models with high predictive performance. Orienting the analysis for learning paradigms, the AUC-ROC values varied between 87.90 (TAVANAEI et al., 2017) and 98.00 (LUO et al., 2019) for DL, and between 87.00 (WANG et al., 2018b) and 99.07 (WANG et al., 2020) for supervised learning. The average accuracy was 89.68 and 89.90 for traditional supervised and unsupervised learning, respectively, and 83.96 for DL. However, AUC-ROC and accuracy might be deceptive and lead to incorrect conclusions with class imbalance. In general, the AUC-PR values varied from 41.65 (GUMPINGER et al., 2020) to 83.00 (SCHULTE-SASSE et al., 2019), and the F1-score from 33.40 (GUMPINGER et al., 2020) to 99.00 (CELLI; CUMBO; WEITSCHEK, 2018), reflecting a much larger variability.

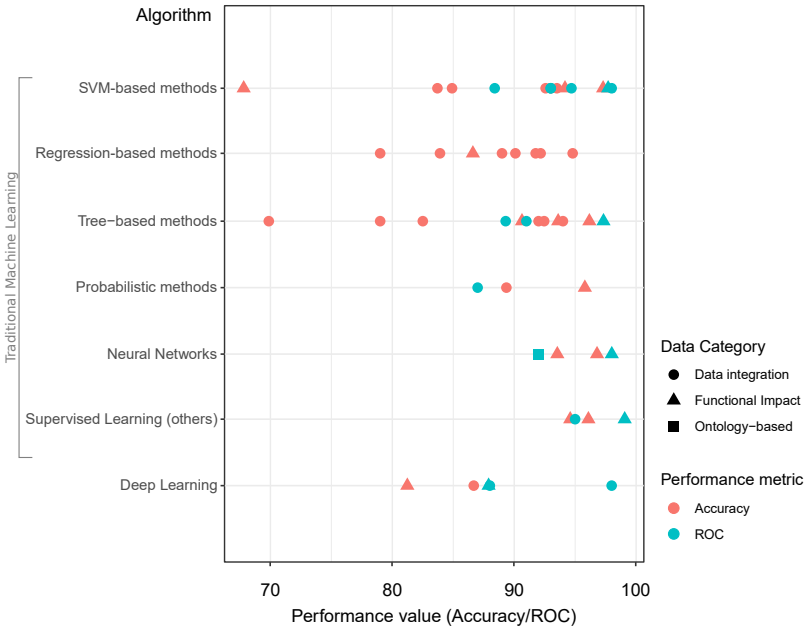
Considering predictive performance, we did not observe great difference among supervised and unsupervised learning, although for the latter we only evaluated three papers. Since collecting experimentally validated driver genes may be difficult and there is currently no agreement on gold standard datasets to train and validate models (RAI-

Figure 3.5 – Summary of predictive performance reported by selected papers. (a) Trade-off between precision and recall. (b) Summary of accuracy and AUC-ROC values.

(a)



(b)



Source: Prepared by the author

MONDI et al., 2021), the main advantage of using unsupervised over supervised learning (either shallow or deep) is that it eliminates the need to curate positive and negative examples for training. It also alleviates the class imbalance problem and the uncertainty in the definition of non-drivers. However, as shown by Gumpinger et al. (2020), the exist-

ing knowledge on well-established cancer genes, although limited, is valuable to improve learning results when appropriately exploited.

3.6 Methodological challenges

Undoubtedly, the use of ML algorithms to predict CDGs and driver mutations has enabled critical scientific advances and has an excellent potential to go even further. In this chapter, we have shed light on the broad data and algorithmic landscape behind this problem, summarizing the myriad of possibilities of combining them into novel solutions and their achievements. Nonetheless, to accelerate the computational discovery of new cancer drivers with ML, several challenges remain to be explored. In what follows, we highlight open problems identified through our review.

3.6.1 Class imbalance

The prediction of cancer driver mutations constitutes an inherent class imbalance issue, given the significantly lower number of known driver mutations in comparison to passenger mutations. Despite the negative impact on the model development process, few selected papers using traditional supervised or deep learning approaches have discussed strategies to deal with this issue. We observed the use of undersampling of the majority class (CAPRIOTTI; ALTMAN, 2011; TAN; BAO; ZHOU, 2012; GNAD et al., 2015; LUO et al., 2019; CUTIGI et al., 2020; GUMPINGER et al., 2020), weighted SVM (MAO et al., 2013), cost-sensitive classifier (U et al., 2014), and weighted cross-entropy as loss function (SCHULTE-SASSE et al., 2019). One-class SVM, although not originally designed to deal with class imbalance, may be particularly useful in this scenario (COLLIER; STOVEN; VERT, 2019; NULSEN et al., 2021). A better understanding of the effectiveness of existing strategies, as well as an elaboration of new ways of mitigating class imbalance, are relevant points to investigate. Semi-supervised and self-supervised learning paradigms can also be promising directions to explore.

Another related issue is the choice of appropriate performance metrics. We observed a prominent use of accuracy and AUC-ROC values for performance assessment, which under skewed class distributions do not provide adequate assessment for the minority class (*i.e.,* driver genes or mutations). Both metrics are less sensitive to false positives

as the size of the negative class grows and can produce misleadingly high values. Thus, it is crucial to use metrics that can pay proper attention to methods' performance for the minority class, such as precision, recall, F1 score, AUC-PR, and MCC. This methodological aspect is crucial for model assessment and should not be neglected in future efforts.

3.6.2 Data leakage

Data leakage happens when information from the test set is accidentally used to develop the model. For instance, pre-processing the complete dataset with normalization, standardization, or data imputation techniques before data partitioning causes instances from the test set to influence the train set. Similarly, performing feature selection in the complete dataset that will further be divided into independent partitions for model training and validation produces the same unintentional effect. Even under the use of cross-validation, data leakage may occur if the same test fold used to optimize the model with hyperparameter tuning or feature selection, for instance, is applied for model evaluation and selection. These methodological flaws were identified in several revised papers. Taking steps to prevent data leakage is imperative to guarantee better model generalization. Data preparation pipelines should be modeled based on training data and then applied to all partitions (*e.g.,* train, test, and validation set). Nested cross-validation is a promising alternative for comparing and selecting ML models with size-limited datasets, reducing the bias in the error estimate when both hyperparameter tuning and evaluation are carried out (VARMA; SIMON, 2006; VABALAS et al., 2019). Moreover, FS protocols designed for high-dimensional data may control optimistic performance estimates (KUNCHEVA; RODRÍGUEZ, 2018), especially under low-sample-size as it is the case for some cancer types.

3.6.3 Data representativeness

In ML, guaranteeing that all the variability associated with a prediction task is represented in the training dataset is as essential as data volume. In the context of cancer, the high intra- and inter-tumor heterogeneity pose additional challenges to data representativeness. Data may be unevenly distributed across distinct cancer types, tumor stages, patients' clinical profiles, or even distinct demographic groups (*e.g.,* gender and

ethnicity), which certainly interferes with model generalization power. For instance, sex-associated differences in tumor molecular profiles and mutation frequency were previously reported (LI et al., 2018). Likewise, mutational processes vary widely among cancer types (BROWN et al., 2019). Thus, any underrepresented subpopulation may suffer from biased prediction results when building models without considering this inequality.

This rationale applies not only to a sample-oriented perspective but also to a gene-oriented perspective. Here, we highlight two possible underrepresentation issues. First, some driver genes are commonly mutated across cancer types, while others are tumor-specific (POULOS; WONG, 2019). Thus, tumor-specific drivers may not be identified in pan-cancer models due to low statistical power arising from low-sample size. Patterns from local samples tend to be diluted among patterns from the population, hindering their identification by the ML algorithm.

Second, known cancer driver mutations tend to occur in a subset of human genes, while negative samples are spread across the genome. As discussed by Raimondi et al. (2021), this may cause ML models to learn the simpler gene-level patterns instead of the more intricate molecular-level functional effects of driver variants, achieving an unrealistic good performance for variant-level prediction. Thus, we stress the importance of defining high-quality, unbiased negative class samples for supervised learning methods. Finally, to continue advancing, new computational strategies are needed to overcome the underrepresentation issues aforementioned. Some promising directions are developing patient-level and cohort-level methods and exploring multi-task learning for simultaneously optimizing pan-cancer and cancer-specific models.

3.6.4 Model interpretability

Although some works have investigated the most relevant features to predict CDGs, model interpretability is still in its infancy within this domain. Model interpretability helps increase the predictive model's trust and provides valuable biological insights about molecular differences among driver and passenger mutations that may fill current knowledge gaps. Although interpreting deep learning and graph-based learning models is especially challenging, recent works have proposed strategies to comprehend the decision-making process, exploring their applications to bioinformatics (SCHULTE-SASSE et al., 2021; TALUKDER et al., 2021). We advocate the use of model interpretability tools in future works as they may explain the discovered patterns and, consequently, ensure these

patterns are significant and consistent with the target task (*i.e.,* avoid *Clever Hans* effect (LAPUSCHKIN et al., 2019)), elucidate properties associated to subgroups of drivers, and point to particular cases that deserve further attention from experts. However, we raise caution that pitfalls in model interpretability may produce incorrect conclusions (MOLNAR et al., 2020), thus demanding careful use.

3.6.5 Non-coding drivers

Mutations occurring in both coding and non-coding DNA regions may play a crucial role in cancer development. For instance, highly recurrent mutations in the promoter region of *TERT* gene have been found in more than 50 tumor types, prompting efforts to identify additional non-coding driver events (ELLIOTT; LARSSON, 2021; BELL et al., 2016). However, there are few computational tools specifically tailored for detecting drivers in non-coding regions (*e.g.,* (FU et al., 2014; LAWRENCE et al., 2013; GUO; CHANG; SKANDERUP, 2020)). Identifying signals of positive selection in non-coding DNA is even more challenging because the non-coding region is about 50 times larger than the coding exome, and the number of known cancer genes with non-coding mutations is much more limited (BELKADI et al., 2015; ELLIOTT; LARSSON, 2021). The more frequent use of whole-genome sequencing (WGS) tends to produce more and more data that could be analyzed for this purpose. Given the several ways in which variations in non-coding DNA may contribute to tumor emergence, exploring ML learning (especially unsupervised algorithms) and the increasing volume of WGS data represents an important research opportunity to advance in the field.

3.6.6 Graph-based machine learning

Networks are pervasive in biology, representing the existing interactions between genes, gene products, and other molecules. Network-based analysis has attracted considerable attention in the prediction of disease genes in general (ATA et al., 2021), and cancer drivers specifically (ZHANG; ZHANG, 2016; DIMITRAKOPOULOS; BEERENWINKEL, 2017). Thus, a natural and still underexplored direction in this domain is the use of graph-based ML algorithms, which can directly process graph-structured data in an end-to-end manner without the need for handcrafted feature engineering.

Graph Neural Networks (GNNs), as reviewed by Wu et al. (2020), can better explore network's topological information for node-level or edge-level prediction tasks in contrast to traditional network analyses, thus finding several fruitful applications in Bioinformatics (ZHANG et al., 2021). Although only one selected paper adopted this approach, recent papers (SCHULTE-SASSE et al., 2021; PENG et al., 2021) further explored GNNs and their variants (*e.g.,* Graph Convolutional Networks) for cancer driver gene identification, obtaining outstanding performance compared with the state-of-the-art methods. We believe that a deeper exploration of GNNs with multimodal data (*e.g.,* multi-omics), proposing strategies to improve robustness to structural noise and class imbalance, may enable a more precise and comprehensive identification of CDGs.

3.7 Summary

In recent years, researchers have made significant progress in identifying genomic changes that contribute to the development of tumors and in distinguishing driver mutations from passenger mutations. This chapter highlights efforts that use machine learning algorithms, including traditional ML and deep learning algorithms, and summarizes the different datasets and approaches used to define features for predictive models. However, despite these advances, the field is still facing many biological and technical challenges that hinder our ability to comprehensively detect cancer drivers. To continue making progress, it is important to review significant advances and methodological details from previous research, identify gaps that need to be addressed, and propose new perspectives to advance our understanding of cancer drivers. This work focuses on one particular trend identified, which is the integration of data, and aims to address two specific challenges among those outlined in Section 3.6: treating class imbalance and better exploring PPI network structures during learning with graph-based learning algorithms.

4 MATERIALS AND METHODS

In the previous chapter, it was possible to note that despite the important computational advancements in predicting CDGs, there are still several methodological challenges that require attention. One area that remains underexplored is the use of graph-based learning algorithms, such as graph neural networks. However, given the increasing success of GNNs in bioinformatics applications as reviewed by Zhang et al. (2021), exploring the potential of these algorithms for predicting CDGs is a promising research direction.

Thus, the primary focus of this work is to investigate the potential of GNNs for predicting CDGs and develop a model that can efficiently leverage information contained in PPI networks and multi-omics data to learn patterns related to CDGs. Our approach is motivated by findings from previous research highlighting that a node context within a biological network provides valuable information for predicting cancer drivers (GUMPINGER et al., 2020).

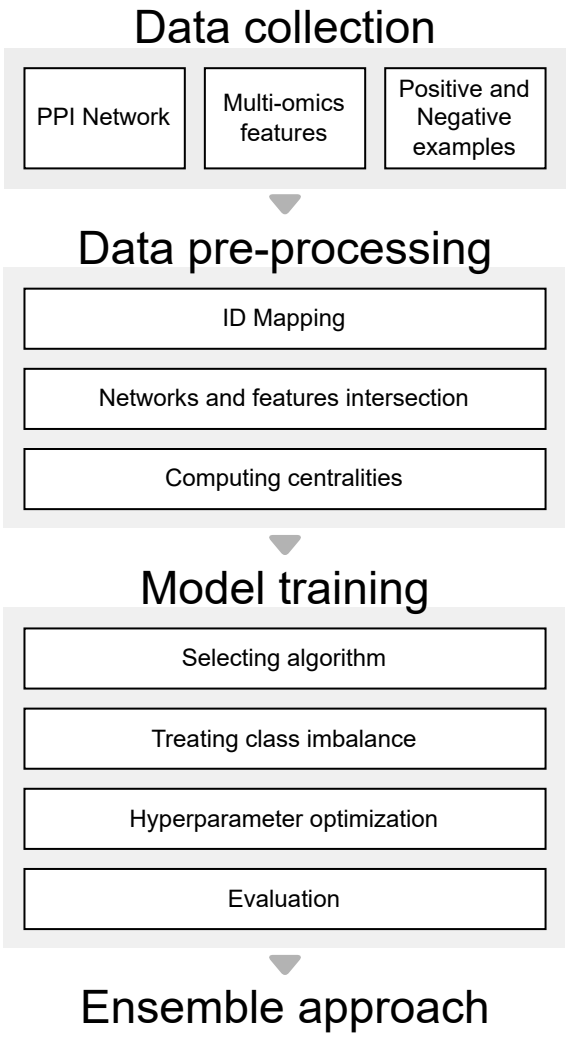
To develop a graph-based predictive model for CDG discovery, we conducted a comprehensive analysis of several important aspects related to model development, including the choice of PPI networks and omics data employed as node features, the use of distinct GNN algorithms, and the strategies adopted for handling class imbalance. Furthermore, we proposed ensemble-based approaches to improve the predictive performance of individual models. A summary of our methodology is provided in Figure 4.1.

This chapter details the materials and methods utilized in this work, providing an overview of each methodological step involved in both the comprehensive experimental investigation of learning algorithms and data representation (reported in Chapter 5) and the development of a GNN model for CDGs prediction (reported in Chapter 6).

4.1 Data collection

All the data employed in this work derive from publicly available sources. The following sections detail our data collection process to obtain protein-protein interaction networks (*i.e.,* our graphs used as the basis for learning), omics data (*i.e.,* our node features), and gene annotations regarding their role in causing cancer (*i.e.,* our node labels). Figure 4.2 illustrates how these different types of data are integrated, summarizing the multi-omics data collected.

Figure 4.1 – Methodology employed in the current work for model development.



Source: Prepared by the author

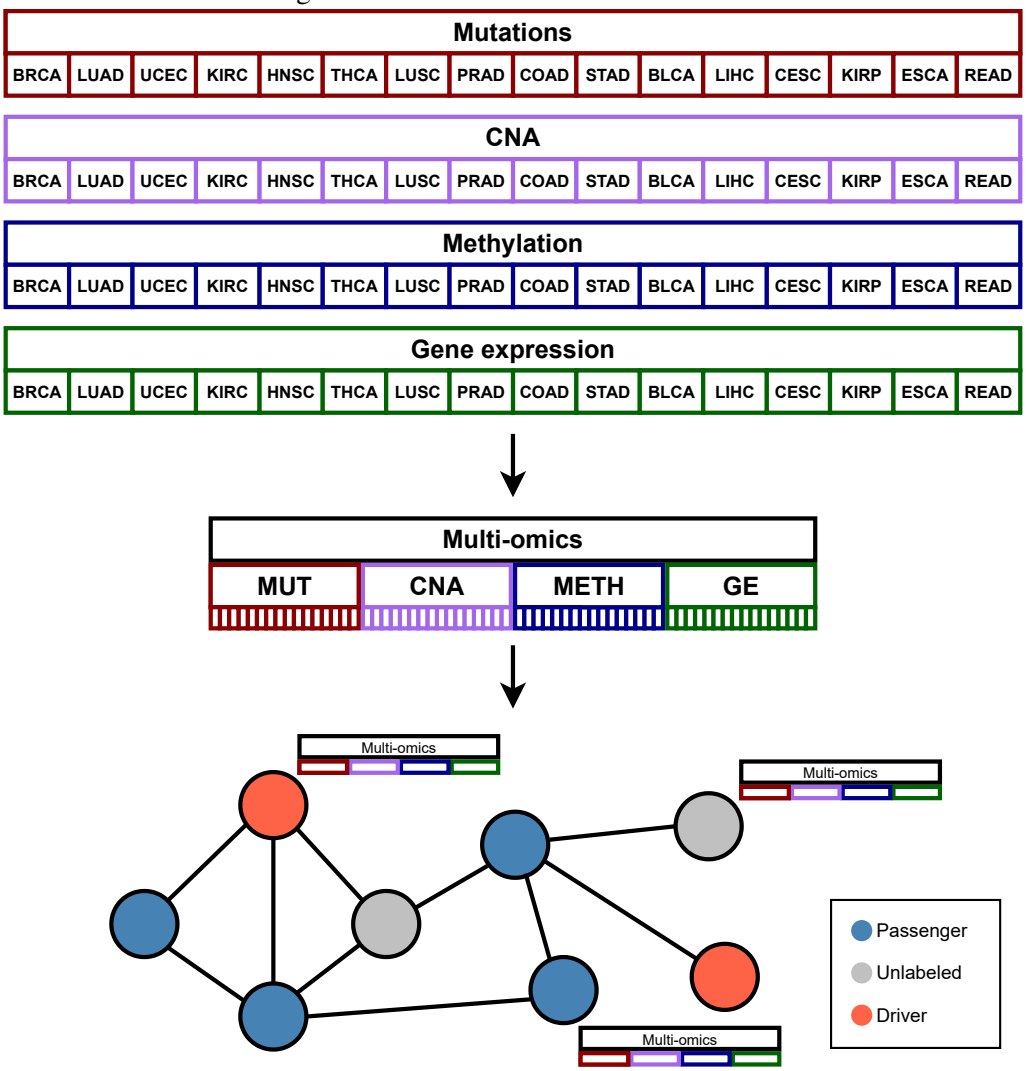
4.1.1 Protein-protein interaction networks

As explained in Section 2.1.3.2, PPI networks depict the physical or functional relationship between proteins (*i.e.,* the nodes of the networks). Several databases provide organized and structured information about PPI networks. In this work, we adopted a reference database for the retrieval of molecular networks in human diseases research, named the Network Data Exchange (NDEx) (PRATT et al., 2015; PILLICH et al., 2017; PRATT et al., 2017). The NDEx Project[1] is a web-based resource that allows for the storage, sharing, manipulation, and publication of biological network models in human disease research (PRATT et al., 2015; PILLICH et al., 2017; PRATT et al., 2017).

From the NDEx repository, we selected six pre-processed PPI networks that were

[1]Network Data Exchange (NDEx): <https://www.ndexbio.org/>

Figure 4.2 – Multi-omics Data Collection



Source: Prepared by the author

deposited by Huang et al. (2018a) in a study that performed a systematic evaluation of molecular networks regarding their ability to recover gene sets that characterize human diseases. Specifically, we downloaded the following networks: Human Protein Reference Database (HPRD), MultiNet, IRefIndex (IREF), Consensus Path DB (CPDB), Parsimonious Composite Network (PCNET), and Search Tool for Recurring Instances of Neighboring Genes (STRING). These networks were chosen because they vary in size, comprising from 30,000 (HPRD) to 5 million (STRING) molecular interactions. Moreover, PCNET showed the highest performance in retrieving the disease-associated gene sets among a total of 21 networks evaluated in the original work (HUANG et al., 2018a).

PPI networks are typically represented as undirected and unweighted graphs (similar to the citation networks used in original GCN applications (KIPF; WELLING, 2016)). Although the NDEx repository provides edge weights for some PPI networks, which rep-

resent the total weight of evidence for an interaction between two proteins, we discarded this information for all PPI networks used in this study. The PPI networks had already undergone a normalization process to enable comparison among them. In the original study (HUANG et al., 2018a), only interactions among protein-coding genes were kept, and all nodes (proteins) were mapped to their corresponding Gene Symbols provided by HUGO Gene Nomenclature Committee (HGNC). However, because this data was compiled in 2018, PPI data underwent another normalization process for nomenclature in our study, which will be later explained in Section 4.2.1.

Finally, we defined a large consensus PPI network by merging the six selected PPI networks into a single network and removing duplicated edges. This PPI network is the most comprehensive among all collected interaction datasets and it aims to evaluate how the algorithms perform for larger and denser networks. We call this merged network the UNION PPI network, which is composed of more than 7.5 million interactions. Details about the original number of nodes and edges in each PPI network are provided in Table 4.1.

Table 4.1 – Number of nodes and edges in PPI networks

	HPRD	MULTINET	IREF	CPDB	PCNET	STRING	UNION
Edges	36,867	109,595	133,548	1,648,426	2,724,724	5,135,768	7,515,970
Nodes	9,453	14,445	14,667	16,301	19,781	18,266	21,968

4.1.2 Omics data

Integrating different types of omics data in the discovery of CDGs is a common approach, as discussed in Chapter 3, since each type of omics carries a distinct, partial view about the molecular changes associated with cancer. In this work, we followed this trend and collected four types of omics data interrogating SNVs, CNAs, DNA methylation, and gene expression in a large scale (see Section 2.1.3 for theoretical background). The datasets were originally produced by the TCGA project and comprise over 8,000 samples from 16 different cancer types. We downloaded a version of the data that was pre-processed by a previous study (SCHULTE-SASSE et al., 2021) to create informative features from this genome-wide molecular data. The authors defined computational features as relative values comparing a quantity of a biological feature in tumor sample compared to a matched normal sample from the same cancer type. Due to this definition, the analysis was limited to those cancer types with data available for both cancer and

normal samples, as listed in Table 4.2.

Table 4.2 – List of cancer types included in the omics data analyzed.

Abbreviation	Cancer Type Description
BRCA	Breast invasive carcinoma
LUAD	Lung adenocarcinoma
UCEC	Uterine Corpus Endometrial carcinoma
KIRC	Kidney renal clear cell carcinoma
HNSC	Head and Neck squamous cell carcinoma
THCA	Thyroid carcinoma
LUSC	Lung squamous cell carcinoma
PRAD	Prostate adenocarcinoma
COAD	Colon adenocarcinoma
STAD	Stomach adenocarcinoma
BLCA	Bladder Urothelial Carcinoma
LIHC	Liver hepatocellular carcinoma
CESC	Cervical squamous cell carcinoma and endocervical adenocarcinoma
KIRP	Kidney renal papillary cell carcinoma
ESCA	Esophageal carcinoma
READ	Rectum adenocarcinoma

In what follows, we briefly mention the pre-processing steps conducted by Schulte-Sasse et al. (2021); more details can be obtained in the original work.

- **SNVs:** the somatic mutation provided in Mutation Annotation Format (MAF) was subject to statistical analysis for variants calling following the pipeline from HotNet2 (LEISERSON et al., 2015). Silent mutations were removed from data to concentrate on SNVs that have an effect on the mRNA or protein. A normalized mutation rate (*i.e.,* SNV frequency) was computed for each gene in each cancer type, by dividing the number of non-silent SNVs in that gene its by exonic length.
- **CNAs:** copy number alterations detected through DNA sequencing were analyzed with GISTIC2 (MERMEL et al., 2011) to identify the exact amplified or deleted target genes. Ultra-mutated samples from TCGA were removed. The copy number rate of a certain gene in a TCGA cohort was defined as the number of times a gene was amplified or deleted in that specific cohort.
- **DNA methylation**: the DNA methylation status was analyzed for the promoter region of genes, defined as the region of ± 1000 base pairs around the Transcription Start Site (TSS) of that gene. Probes annotation was carried out with Gencode. Within the promoter region, the average $beta$ value (*i.e.,* DNA methylation signal) across all probes was used to determine the DNA methylation status of the gene. The $beta$ value ranges between 0 and 1 and describes the portion of cells that are

methylated in the sample. For each gene, the average differential DNA methylation was computed by averaging the difference between the $beta$ values for a cancer sample and its matched normal sample across all samples in a given TCGA cohort.

- **Gene expression**: authors used the data from Wang et al. (2018a), in which gene expression levels for TCGA cancer samples were combined with gene expression data for normal samples of the same tissue produced by the Genotype-Tissue Expression (GTEx) consortium in a controlled protocol to ensure similar distributions among cancer and normal samples. The differential expression for a gene in a given cancer sample was calculated as a log2-fold change between the expression values in the cancer versus a matched normal sample. Finally, for each gene, an average differential expression was computed across all samples from the same cancer type.

After individual pre-processing, each type of omics data results in a feature matrix in the form of $N \times 16$, in which N denotes the number of genes and 16 refers to the number of TCGA cancer cohorts analyzed (as illustrated in Figure 4.2). Thus, for each gene, we obtained 16 features corresponding to its molecular profile in terms of SNVs, CNAs, differential DNA methylation or differential gene expression across 16 types of cancer. Each feature matrix was passed through min-max normalization to ensure similarity among the scale of the different types of omics data in joint analyses. Besides the individual features matrices, we also generated a multi-omics feature matrix by simply concatenating the four individual feature matrices for the individual omics levels, generating a $22,193 \times (4 * 16)$ matrix.

4.1.3 Positive and negative examples of CDGs

To create a reliable list of labeled genes, we examined catalogs of cancer-causing genes available in previous studies or public databases. We defined two main sets of genes: the positively labeled genes, which are *driver genes*, and the negatively labeled genes, which are *non-cancer genes* or *passenger genes*. However, many genes do not fall into either of these categories because they have some evidence of involvement in oncogenesis but lack definitive annotation to be considered as true drivers, or because they have been associated with other diseases, increasing their chance to be cancer-related genes. These genes are referred in our work as the *unlabeled* set, but we note that they may still represent potential candidate driver genes. The data collection process for these

gene sets is elaborated in the following paragraphs.

We obtained six lists of genes to be processed as necessary for creating node labels for our networks: (i) driver and candidate drivers from the Network of Cancer Genes (NCG) version 7.0 (REPANA et al., 2019); (ii) Tier 1 and Tier 2 lists from the Cancer Gene Census (CGC) available in COSMIC v95 (TATE et al., 2018); (iii) the OncoKB Cancer Gene List (version of July 1st, 20222) (CHAKRAVARTY et al., 2017); (iv) a compilation of cancer genes organized by Bushman Lab[2] (v5, from June 2021); (v) genes related to cancer pathways in the Consensus Path DB (CPDB) (Release 35, updated on 05/06/2021) (KAMBUROV et al., 2011); and (vi) disease-associated genes registered in the Online Mendelian Inheritance in Man (OMIM) (HAMOSH et al., 2000). All gene lists were downloaded at January 14th, 2022.

For positive examples of CDGs, we collected a list of canonical cancer drivers from NCG 7.0 (591 genes), the Tier 1 list from CGC (578 genes) that contains genes with cancer gene mutation patterns and documented activity relevant to cancer, and a list of TSGs and OGs from OncoKb (500 genes). Combining the three lists and removing the duplicated genes, we arrive at 907 positive genes.

Next, we compiled a list of disease candidate genes, which are those gene implicated in other diseases or that have indications of an oncogenic role (*e.g.,* through computational prediction) but the body of scientific evidence supporting their role is still emerging, thus, they lack strong experimental evidence to be labeled as positive examples of cancer drivers. We collected the list of candidate cancer drivers from NCG 7.0 (3177 genes), the Tier 2 list from CGC (151 genes), the cancer-related gene list provided by the Bushman Lab (2580 genes), and the cancer gene list from OncoKB excluding annotated TSGs and OGs (500 genes). We also obtained a list of genes annotated in cancer pathways according to CPDB (4333 genes) and a list of genes previously associated with other human diseases according to OMIM (2143 genes). Again, as redundancy may exist when integrating the lists obtained from different data sources, many duplicates were found and removed. Moreover, genes that were classified as drivers in any of the sources previously mentioned were removed from the list. We obtained a total of 5777 unique disease candidate genes that were grouped as the *unlabeled* gene set. These genes were treated as unlabeled nodes in our computational prediction approach, since the uncertainty about their role in oncogenesis prevents us from considering true cancer drivers, but it is risky to label them as passenger ones due to existing evidence of association with cancer or

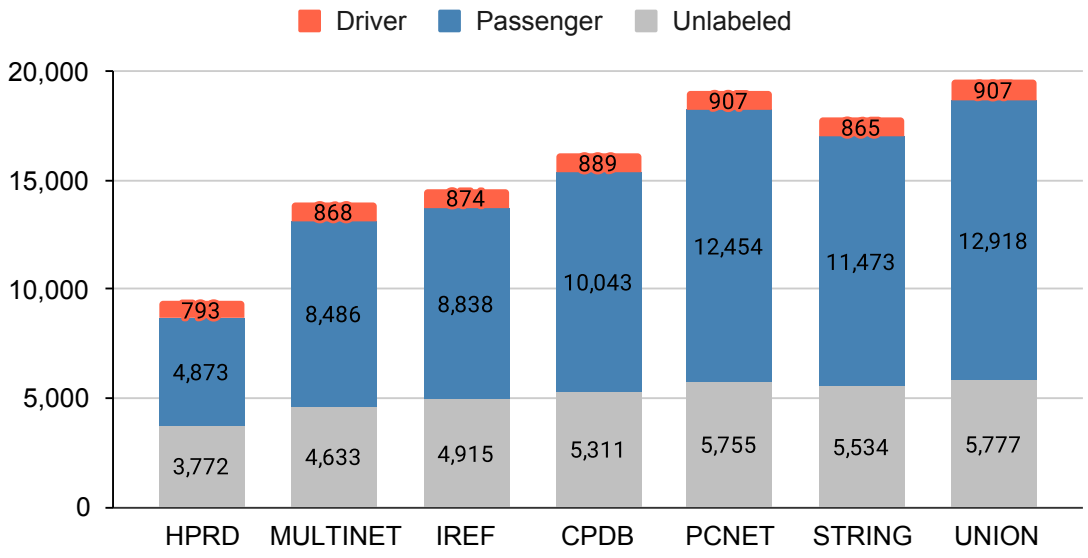
[2]Available at <http://www.bushmanlab.org/links/genelists>

other diseases.

Given that the definition of robust negative examples of CDGs is more challenging, as mentioned in Chapter 3, we followed a similar strategy applied in previous works and filtered the list of genes available in the collected PPI networks and omics data to remove all genes with previously reported association with cancer or other diseases (including strong and weak evidences). Thus, we considered as negative examples (*non-cancer genes*), all the genes in our data that were not included in any of the following lists: the compiled list of positive examples of CDGs, the candidate cancer genes list, the cancer pathways gene list from CPDB, and the disease-related genes list from OMIM. In summary, negative examples of CDGs are those that do not appear in the defined *driver* gene set or in the defined *unlabeled* gene set.

With the complete list of positive and negative examples of CDGs, it is possible to cross this information with the genes present in each PPI network to define node labels distribution per network. This process and other pre-processing steps will be detailed in the next section. However, the final class label distribution per PPI network is shown in Figure 4.3. We can observe the high class imbalance in the dataset, with the driver genes representing the minority class.

Figure 4.3 – Final distribution of class labels per PPI network.



Source: Prepared by the author

4.2 Data pre-processing

All collected data were subject to further pre-processing to allow creating a consistent training data for the learning algorithms explored in this work. In this section, we describe the steps involved in data pre-processing, which comprise (i) mapping gene IDs, (ii) intersecting genes among PPI networks and omics data, and (iii) extracting node centralities from PPI networks.

4.2.1 Gene ID mapping

To carry out the experiments for model training with ML or DL, the nodes IDs need to be standardized across the various data sources that are collected in this work. As mentioned earlier, the genes of the PPI networks are standardized in Gene Symbols provided by HUGO Gene Nomenclature Committee (HGNC), however the networks were last updated in 2018. The HGNC is a committee within the Human Genome Organization (HUGO), which frequently update its databases and has a curated online repository of HGNC-approved gene nomenclature.

To ensure that we have the most up-to-date gene IDs, we used the HGNC Multi-symbol checker. This tool allows the user to upload a list of gene symbols in a search field to verify if they are HGNC approved symbols. The results include a "match type" column that shows how each queried symbol matches the returned HGNC symbol – if either with approved HGNC symbols, alias symbols, updated symbols, or if the queried gene symbol has been withdrawn from database or if it did not return any match. Based on these results, we created a dictionary with the list of query genes and their corresponding approved symbols. We conducted the query on November 26th, 2021, and downloaded a complete dictionary of gene ID mappings that was used throughout all experiments.

Table 4.3 shows the original number of nodes in each of the seven PPI networks used and the resulting number of nodes after gene ID mapping. It is important to note that some nodes, and their connected edges, may have been removed from the network if their original ID could not be mapped to the updated HGNC gene symbols. This may lead to a reduction in the number of edges as well.

Table 4.3 – Original and modified number of nodes and edges in PPI networks as a result of data pre-processing steps.

PPI Networks	Original		Gene ID mapping		Omics intersection	
	Edges	Nodes	Edges	Nodes	Edges	Nodes
HPRD	36,867	9,453	36,852	9,444	36,844	9,438
MULTINET	109,595	14,445	108,568	13,987	108,568	13,987
IREF	133,548	14,667	133,100	14,632	133,095	14,627
CPDB	1,648,426	16,301	1,641,909	16,251	1,641,849	16,243
PCNET	2,724,724	19,781	2,712,881	19,116	2,712,881	19,116
STRING	5,135,768	18,266	5,028,709	18,044	5,002,412	17,872
UNION	7,515,970	21,968	7,391,452	19,792	7,365,082	19,602

4.2.2 Intersection of PPI networks and omics data

In the proposed approach, a prediction model is built from the joint use of PPI networks and omics data through GNNs. The PPI network provide the fundamental graph structure for the graph-based learning, whereas the omics data provide node features that enrich node embeddings generated for the classification task. Thus, it is imperative to ensure a correspondence among genes in the PPI networks and genes present in the multi-omics dataset. For each PPI network, we compared the genes in the network and the genes in the muli-omics data to keep in the PPI network only genes that are in the intersection among both datasets. After this step, the number of nodes and edges were updated again, as shown in Table 4.3. This step allowed all nodes from all PPI networks to have their own set of features.

4.2.3 Extraction of node centralities

In Section 2.2.3.1, we provided definitions of four measures of node centralities: degree, betweenness, closeness, and clustering coefficient. In the present section, we provide a brief overview of the extraction and normalization procedures used to obtain these measures. We employed a network analysis package, called *igraph* (CSARDI; NEPUSZ, 2006), which is an open-source tool that can be implemented using various programming languages, including Python, R, Mathematica, and C/C++. The igraph provides a versatile network analysis that enables mining information from graphs, ranging from simple operations such as adding and removing nodes to more complex theoretical constructions

such as community detection. The Python interface of igraph (version 0.9.8) is utilized in this work to compute the four node-level properties, as detailed below:

- **Degree**: accepts either a singular node ID or a list of node IDs as an input parameter and returns the number of edges incident on a node of the graph. The output is a single integer or a list, based on the input parameter. Degree values were normalized by the min-max method to keep it contained in an interval from 0 to 1, similar to the range of values for the omics features.
- **Betweenness**: assigns higher betweenness centrality scores for nodes that appear more frequently along the shortest paths computed. The score returned by the function is not standardized, so the same min-max normalization is also used.
- **Closeness**: when assessing how many steps is required to reach every other node from a given node, a normalized value is obtained by the inverse average distance to all reachable nodes.
- **Clustering coefficient**: the function returns a score value between 0 and 1, that means the probability that the adjacent nodes of a given node i are connected. Thus, it was not necessary to apply a normalization. However, when a node has a degree lower than two, there is no transitivity, so a parameter was defined to consider these cases as zero (by default they are defined as NaN).

The analysis of centrality values adds four features to each node in the PPI networks. One of the goals of our experiments is to evaluate the effect of using centrality measures as features in GNNs, as will be later discussed in Chapter 5. We emphasize that the centrality measures for a given gene vary among PPI networks because they are highly dependent on the graph structure. Unlike omics-based features, which remain the same for a given gene across PPI networks, the centralities-based features differ among PPI networks.

4.3 Model training and evaluation

After data collection, two classes of algorithms were applied for model development: graph neural network algorithms, which are the focus of our work, and traditional machine learning algorithms, which served as baselines in our experimental comparison. While graph neural networks were applied following a semi-supervised learning paradigm, traditional ML algorithms were trained in a pure supervised learning approach.

This section aims to clarify the technical details underlying the application of the learning algorithms to the collected data.

4.3.1 Graph-based learning algorithms

One of the goals of our study is to conduct a comparison among graph-based learning algorithms. To achieve this, we utilized a library called StellarGraph (DATA61, 2018), which offers implementations for state-of-the-art GNNs. This Python library is built on TensorFlow 2 and Keras, as well as Pandas and NumPy. We selected StellarGraph for its ease of use, modularity, extensibility, and range of algorithms that solve many ML tasks on graphs, including node-level predictions, which enables a fairer comparison among algorithms using the same data representation format and library.

Graph-based model training was conducted primarily with StellarGraph, but also employing TensorFlow, Keras, Pandas, and scikit-learn for auxiliar functions. Three GNN algorithms were selected for evaluation: Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), and GraphSAGE (see Section 2.2.3 for technical details). Since the algorithms were selected for a categorical prediction task, the categorical cross-entropy loss function is used as the standard loss function. While some of our experiments were based on default hyperparameters for the applied algorithms (see Chapter 5 for details), other experiments addressed hyperparameters optimization in an extensive evaluation of selected methods (see Chapter 6 for details). Section 4.3.2 will elaborate more on the optimization of hyperparameters.

When employing GNN algorithms, the task of predicting CDGs is modeled as a semi-supervised node classification task. Semi-supervised learning on graphs leverages the propagation of information in a per-layer fashion to obtain more advanced node representations, thereby avoiding the need for labeled data for all nodes while making the best use of available prior knowledge. In our domain, there are positively and negatively labeled genes, as well as a large set of unlabeled genes due to the lack of definitive evidence regarding their role in oncogenesis. However, by propagating feature information among neighboring nodes in a layer-wise manner, GNNs can still enable the model to learn patterns that can help discriminate CDGs. After training, the model can be used to predict the class of the unlabeled nodes. This setup is particularly well-suited for discovering associations between entities in biological networks, such as cancer driver genes, given the limited knowledge about biological and molecular mechanisms underlying diseases.

4.3.2 Hyperparameters in GNNs: definitions and optimization

The success of ML and DL algorithms, including GNNs, is partly attributed to the configuration of their hyperparameters, which enable them to achieve superior results on a given dataset (YUAN; WANG; PANG, 2021). Therefore, it is crucial to comprehend the hyperparameters involved in the chosen algorithms and conduct hyperparameter optimization when feasible. Despite the high similarity among the GNN algorithms applied in this work (Section 4.3.1) in term of their propagation module, each algorithm has particular hyperparameters on their own that are, in general, introduced as a way to help the algorithm to stand out from previous approaches. Table 4.4 presents a brief description for hyperparameters involved in GNNs training.

Table 4.4 – Hyperparameters of the selected GNN algorithms

Hyperparameter	Description
Layer sizes	The number of hidden convolutional layers of the GNN used for training and the feature dimensions for each hidden layer.
Activation function	The non-linear transformation function that is applied to the input signal and outputs a value that is passed on to the next layer.
Dropout rate	The percentage of units in the hidden layers that are randomly dropped during training.
Learning rate	The step size for a model's weight updates to reach the minimum loss function.
Number of epochs	The length of the training or, in other words, the number of times the training data will pass through the neural network.
Loss function	The evaluation function to estimate how well the algorithm can model the input data, measuring the difference between the predicted output and the actual output.
Attention heads	The number of operations of the layer that are independently replicated K times, used for regularization process to avoid fitting.
Attention dropout	The dropout rate applied to attention coefficients.
Batch size	The number of random samples that will be propagated through the network.
Number of samples	Number of neighboring nodes to sample at each level of the model

However, the high number of hyperparameters in GNN algorithms is a primary challenge in developing effective predictive models, as several hyperparameters can significantly impact model performance. Furthermore, hyperparameters optimization is often computationally expensive, and there is no guarantee that the chosen configuration is the optimal choice without experimental comparison. However, it is possible to obtain suitable hyperparameters value guidelines based on prior research.

For instance, Kipf and Welling (2016) reported that two to three convolutional layers are sufficient for learning patterns with GCNs, such that we start our experiments with only two layers. As for the number of units per layer, it depends on the specific model, but we decided to initially employ 64 units. The optimal dropout rate for improving model performance can range from 0 to 1, with a 10% dropout already able to enhance the performance. Dropout is implemented in GNNs by randomly removing edges from the graph during training. Thus, our experiments started with a dropout rate of 0.1. Meanwhile, the learning rate usually falls within the range of 0.1 to 0.0001, according to Koutsoukas et al. (2017). We adopted, initially, a learning rate of 0.001 with ADAM optimizer. Additionally, the number of epochs should be chosen based on the local computing capacity. Our exploratory experiments used 300 epochs for GNN model training.

Activation functions are used in all GNN components to incorporate the feature-topology coupling into the network. The activation function ReLU has been frequently used due to its superior performance in many prediction tasks (GOODFELLOW; BENGIO; COURVILLE, 2016). For our classification task, the recommended loss function by StellarGraph is categorical cross-entropy. Finally, batch size is the only hyperparameter common to all algorithms that receives different recommendations for distinct algorithms according to StellarGraph default values. GCN and GAT are full-batch models, which means the batch size is one, while GraphSAGE trains the model with a batch size of 50.

The hyperparameters discussed so far are shared among the three selected GNN algorithms. Nonetheless, some algorithms introduce other hyperparameters specific to their functioning. The GAT algorithm has two hyperparameters of its own, the attention heads and attention dropout (as described in Table 4.4), which we initially configured with the default values recommended by the library: 8 and 0.01, respectively. GraphSAGE includes a hyperparameter related to the number of nodes to sample, for which we also used the standard setup of 10 sampled nodes in the first layer and 5 sampled nodes in the second layer for a two-layer network.

In our study, we performed two rounds of experiments to evaluate the performance

of GNN algorithms for CDGs prediction based on PPI networks. In the first round, we used the default hyperparameter values, as outlined in the previous paragraphs, while in the second round, we fine-tuned a GNN model using a hyperparameter optimization process based on a grid search and k-fold cross-validation. The specific hyperparameter grid used is described in detail in Chapter 6. However, we note that due to the computational complexity of the PPI networks, a large number of hyperparameter combinations can result in long execution times, demanding an adaptation to the methodology of training and evaluating models. To address this, we adopted a strategy where we first ran hyperparameter optimization for the three smallest PPI networks with less than 140,000 edges (HPRD, MULTINET, and IREF). Based on an analysis of these results, we simplified the hyperparameters grid and used a reduced number of combinations for the three largest PPI networks. Further details on this analysis can be found in Chapter 6.

4.3.3 Traditional machine learning algorithms

Four traditional ML algorithms were chosen as baseline methods in our experiments, namely Support Vector Machines (SVM), Random Forests (RF), Gradient Boosting Trees (GBT), and Artificial Neural Networks (ANN). A theoretical background for these algorithms was provided in Section 2.2.2. For these methods, the prediction of CDGs was modeled as a conventional supervised classification task, in which labeled positive and negative examples are used to learn the data patterns and fit a predictive model that can be latter applied to classifiy unseen, unlabeled data.

The traditional ML algorithms were applied in our pipeline for model development through implementations provided by the scikit-learn package (PEDREGOSA et al., 2011) version 1.1.1, for SVM, RF, and GBT, and by the Keras library (CHOLLET et al., 2015) for ANN, using default hyperparameters in most cases, as listed below:

- **SVM**: C = 1, kernel = “rbf”, degree = 3, gamma = “scale”, coef0 = 0, shrinking = True, probability = False, tol = 0.001, cache size = 200, class weight = None, verbose = False, max iter = -1, decision function shape = “ovr”, break ties = False, and random state = None
- **RF**: n estimators = 100, criterion = “gini”, max depth = None, min samples split = 2, min samples leaf = 1, min weight fraction leaf = 0, max features = “sqrt”, max leaf nodes = None, min impurity decrease = 0, bootstrap = True, oob score = False,

n jobs = None, random state = None, verbose = 0, warm start = False, class weight = None, ccp alpha = 0, max samples = None

- **GBT**: loss = "log_loss", learning rate = 0.1, n estimators = 100, subsample = 1, criterion = "friedman_mse", min samples split = 2, min samples leaf = 1, min weight fraction leaf = 0, max depth = 3, min impurity decrease = 0, init = None, random state = None, max features = None, verbose = 0, max leaf nodes = None, warm start = False, validation fraction = 0.1, n iter no change = None, tol = 0.0001, ccp alpha = 0

Only ANN had the hyperparameters configured to allow a more direct comparison with the GNNs models, as will be detailed in Chapter 5.

4.3.4 Class imbalance strategies

The issue of class imbalance is a frequent problem in the context of predicting CDGs with ML, and it can be addressed by applying different strategies that prevent models from biasing their predictions for the majority class. In this work, we applied some popular strategies to correct or compensate the class imbalance as reviewed in Section 2.2.5: Random Undersampling, Balanced Cross-Entropy, and Focal Loss.

Undersampling and oversampling strategies are quite common in ML with class imbalance and relatively simple to apply in tabular data. However, because we are dealing with graph structures, none of the traditional oversampling strategies satisfied our needs. Moreover, a recent algorithm proposed for synthetic oversampling from minority class in graph-based learning, named GraphSMOTE (ZHAO; ZHANG; WANG, 2021), could not be applied to our framework using the same data structures adopted for other approaches, preventing a fair comparison in our methodological analyses. Thus, we decided to use only a random undersampling method by removing excessive negative labels and their corresponding nodes from the PPI network to balance the amount of positively and negatively labeled nodes. This operation causes a decrease in the PPI networks used for training. Nonetheless, turning negatively labeled nodes into unlabeled nodes could introduce unintentional effects in the learning process, which we aimed to avoid.

Despite the use of StellarGraph library to train the GNN models, we still worked with the conventional Keras prediction values, allowing us to directly employ Keras loss functions. We used the Balanced Cross-Entropy and the Focal Loss functions, which are

straightforward methods to apply to GNNs and were introduced in Section 2.2.5. The use of Focal Loss involves the configuration of hyperparameters α and γ. We explored different values for these hyperparameters in two ways: using pre-defined combinations of values extracted from the original work (as will be explained in Chapter 5) and running a hyperparameter optimization process that included these hyperparameters in the search grid (as will be later discussed in Chapter 6).

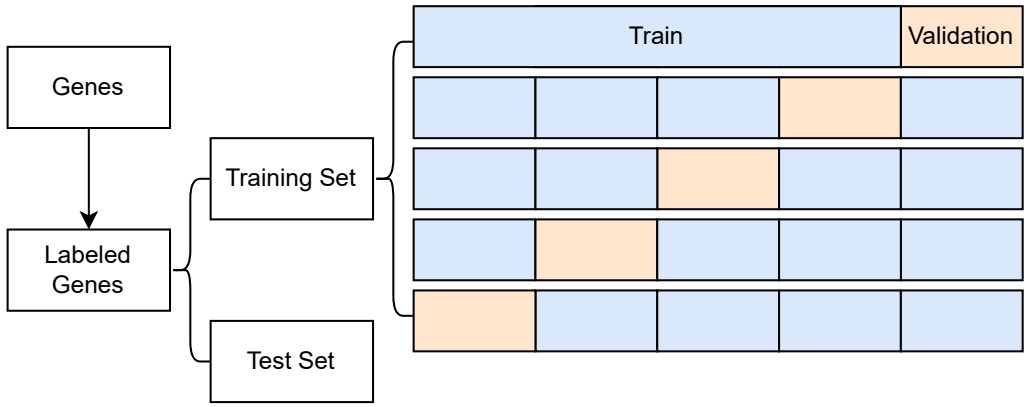
4.3.5 Performance evaluation

An important step in the pipeline for ML model development is the performance assessment of the generalization power of the model, which usually involves the definition of the data splitting strategy and of the performance metrics. In this work, model development was conducted in two stages, starting with an initial comparison of a set of algorithms, followed by hyperparameters optimization of the best performing algorithm. The decision to explore in-depth only a selected algorithm was motivated by the high computational costs involved in our experiments. Thus, while the general approach for model validation is the same for the two stages, some differences exist between them and also among GNNs and traditional ML methods, which will be explained next.

In all experiments, GNN models were evaluated using a Holdout data split to generate stratified training and test sets in a 80:20 ratio, followed by a 5-fold cross-validation in the training set, as shown in Figure 4.4 (see Section 2.2.6 for detailed explanation). Since the training process is based on semi-supervised learning, performance evaluation is limited to the labeled data. Therefore, unlabeled genes are not used for creating training and test sets and are saved for a final prediction of the model, while the labeled genes are passed through the data splitting strategy to allow model training and validation. Our performance comparison is based mainly on the prediction for the test set, although the performance for the training and validation sets in the 5-fold CV process is also monitored and analyzed to better understand the results and investigate the risk of model overfitting.

Ensuring fairness and reproducibility is crucial for comparing GNN models across different PPI networks. To achieve this, we aimed to maintain a consistent test set throughout the experiments. In the first stage, we accomplished this by setting the *random_state* parameter of the function used to divide the labeled data into training and test sets. This guarantees that the same test set is employed for each PPI network during the comparative performance evaluation carried out. In the second stage, on the other hand, we aimed to

Figure 4.4 – Strategy for GNNs model training and validation based on a Holdout followed by a 5-fold CV in the training set.



Source: Prepared by the author

compare not only model predictive performance but also actual predictions for the same set of genes across different PPI networks. Thus, for this stage of our methodology, we manually separated a test set that is common to all PPI networks and used it as a fixed independent test set in the experiments. This process is further discussed in Section 6.1.1.

For traditional ML algorithms, the performance evaluation was based only on a 5-fold CV process in the complete labeled data. Since in the stage of algorithms comparison, we did not included a step of tuning ML models using hyperparameters optimization, the k-fold CV procedure is a reliable approach to conduct model evaluation and extract robust performance estimates.

All models were evaluated based on the Area under the Precision-Recall Curve (AUC-PR), Area under the ROC curve (AUC-ROC), and accuracy. These metrics were defined in Section 2.2.6. Due to the severe class imbalance found in our data and to the negative impact it causes in the model evaluation through accuracy and AUC-ROC, we employed AUC-PR as our main metric.

4.4 Construction of ensemble models

Two ensemble models were constructed based on the strategies outlined in Section 2.2.4. Our aim was to investigate whether the consensus among a set of predictive models, each of which trained with a specific PPI network as the base graph for learning, can introduce robustness and performance gain to our approach.

The trained GNN models, regardless of the algorithm used, are able to provide for a given test instance (*i.e.,* an unlabeled node referring to an unannotated gene) a predicted

class and predicted class probabilities. Since our task is binary classification, we obtain two probabilities for each gene, one for the *Driver* class (positive) and one for the *Passenger* class (negative), with both probabilities summing to one. A higher probability for the positive class indicates a greater likelihood of the gene being a driver gene, according to the model. Thus, two strategies were adopted to obtain ensemble-based predictions:

- **Ensemble-Majority**: combines predicted class labels using majority voting. A gene is classified as a driver if it is predicted as such by at least three PPI networks (*i.e.,* models), with ties considered positive due to the even number of networks.
- **Ensemble-Average**: combines class probabilities by taking the average across all six PPI networks (*i.e.,* models). A gene is classified as a driver if its average probability for the *Driver* class is equal to or greater than 0.5.

Finally, we note that for the purpose of discussing results, we will refer to the GCN trained with UNION PPI network, with the Ensemble-Majority, and with the Ensemble-Average as our ensemble-based prediction models.

4.5 Summary

This chapter introduced the datasets and the methods explored in this work to build the framework for our comparative analysis among learning algorithms and to develop a graph-based predictive model for identifying CDGs. The selected pre-processed multi-omics datasets, derived from TCGA, encompass 16 different types of cancer and four levels of omics data, including genomics (SNVs and CNAs), transcriptomics, and epigenomics. Furthermore, six different PPI networks of different dimensions were collected and pre-processed to extract centrality measures and to fit the needs for the application of GNNs. From these networks, a UNION PPI network was generated based on a simple merging of all collected PPI networks. A careful curation of databases was conducted to define the most possible reliable sets of positive and negative examples of CDGs to serve as node labels for model training. Finally, the methodology for model development included strategies for handling class imbalance and for building ensemble models as means to improve the prediction results. The next chapters will detail the application of these materials and methods for a comprehensive experimental investigation of learning algorithms and data representation and for the and the development of a GNN model for CDGs prediction.

5 COMPARATIVE EVALUATION OF ALGORITHMS AND DATA STRATEGIES FOR CANCER DRIVER GENE PREDICTION

In this chapter, we present and discuss a series of experiments designed to evaluate and compare various methodological strategies for predicting CDGs. Specifically, we focus on three decision levels: (i) the types of features used to describe genes (*i.e.,*, instances) for the prediction task, (ii) the strategies employed to address class imbalance, and (iii) the specific algorithm used for CDG prediction. In the latter, we include a comparative analysis between two classes of learning algorithms, traditional (*i.e.,* "shallow") ML algorithms and graph neural networks, to better understand the potential benefits of using graph-based learning for discovering CDGs.

The following sections present the experimental setup and our results for this set of experiments. Our discussion will be oriented towards five main research questions:

- *RQ1* – How does the choice of omics data used as node features impact the predictive performance of GNNs?
- *RQ2* – Can GNNs benefit from the inclusion of node centrality measures as node features?
- *RQ3* – What is the best strategy to handle class imbalance in the node prediction problem addressed?
- *RQ4* – How do traditional ML algorithms compare to GNNs when applied to the same omics data?
- *RQ5* – Which graph neural network model performs best for CDG prediction?

We conducted the analyses described in this chapter using the three smallest PPI networks presented in Section 4.1.1, namely HPRD, MULTINET, and IREF, due to the high computational costs involved. In total, we carried out 468 experiments testing variations in terms of features, algorithms, class imbalance strategies, and networks. For the sake of brevity, we focus our comparison mainly on the performance of the test set, and we present only the most meaningful results in this chapter. However, additional results for these experiments can be found in Appendix B, specifically in Tables B.1, B.2, and B.3 for GNNs, and Tables B.4, B.5, and B.6 and Figure B.4 for traditional ML algorithms.

5.1 Experiments setup

To ensure a fair comparison between the selected GNN algorithms, we maintained fixed hyperparameter values across all experiments described in this chapter. This approach allowed us to focus on comparing the impact of other decisions related to the methodology employed in model development. Table 5.1 shows the hyperparameter configuration for the three GNN algorithms, with most of the values defined as default by the StellarGraph library. The cross-entropy loss function was used in all experiments unless loss function variations are mentioned in the text, *e.g.,* when assessing the class imbalance solutions. Furthermore, since we used the same Python library to train GNN models in our experiments (as described in Chapter 4), the data structure used was the same for all algorithms, making performance comparisons more straightforward.

The models based on SVM, RF, and GBT were built using the Scikit-learn library (PEDREGOSA et al., 2011) and the ANN was trained using the Keras library (CHOLLET et al., 2015). For the ANN, we configured the hyperparameters similar to the values presented in Table 5.1. We used two hidden layers with 64 neurons in each, the cross-entropy loss function, and a dropout rate and learning rate of 0.01 and 0.001, respectively. The hyperparameters for the remaining algorithms (*i.e.,* RF, GBT, and SVM), were kept as the default values provided by the Scikit-learn library (version 1.1.1).

Model evaluation was conducted as explained in Chapter 4, applying the Holdout method followed by 5-fold CV in the training data for GNNs, and using a 5-fold CV over the complete dataset for traditional ML algorithms. Although accuracy and AUC-ROC were collected through the experiments, we concentrate our discussion in the AUC-PR metric, as it represents a better compromise for evaluating problems with class imbalance in which the other metrics tend to be highly optimistic.

5.2 Results

5.2.1 How does the choice of omics data used as node features impact the predictive performance of GNNs?

Similarly to traditional ML algorithms, GNNs can leverage the information provided by node feature vectors to learn patterns associated to CDGs. By integrating this information with the local structure of the nodes in the graph, the GNN learns to predict

Table 5.1 – Hyperparameters configuration used for GNNs in the comparative evaluation experiments.

Hyperparameter	Algorithm		
	GAT	GCN	GraphSAGE
Number of layers	2	2	2
Number of units on each layer	64	64	64
Activation function	ReLU	ReLU	ReLU
Loss function	Cross-entropy	Cross-entropy	Cross-entropy
Dropout rate	0.01	0.01	0.01
Learning rate	0.001	0.001	0.001
Number of epochs	300	300	300
Attention heads	8	-	-
Attention dropout	0.01	-	-
Batch size	1	1	50
Number of samples per layer	-	-	10 and 5

their labels. Therefore, it is reasonable to assume that the use of different types of omics data as node feature vectors may impact on the performance of the model.

To evaluate the effects of applying different types of omics data as node features on the performance of GNN models, we carried out five experiments with each of the smallest PPI networks (*i.e.,* HPRD, MULTINET, and IREF). First, we analyzed the model performance using a single type of omics data (*e.g.,* Mutations, CNA, DNA Methylation, or Gene expression) as node feature vectors during training. Next, we trained models concatenating the four types of omics data as node features, which is known as a multi-omics approach. We note that these experiments used the traditional cross-entropy loss function, thus, they do not address the issue of class imbalance.

Table 5.2 summarizes the results for the AUC-PR metric for the test set. We highlight in boldface the best results for each algorithm and PPI network. It is worth noting that the overall performance of all GNN models falls short of what is expected from a good classifier. The low AUC-PR values suggest that the models are not performing well in correctly identifying all true CDGs and avoiding false positive predictions, which means classifying a gene as a cancer driver when it is not. Nevertheless, given the significant class imbalance in our dataset, such difficulty is expected, and the performance metrics still provide a basis for comparison.

Our experiments showed that the choice of node feature vector can have a significant impact on the performance of GNN models. For example, in the HPRD PPI network, performance ranged from 0.2761 (using *CNA* node features) to 0.3844 (using *DNA Methylation*) for GAT, from 0.2967 (using *Gene Expression*) to 0.4984 (using *Multi-omics*) for GCN, and from 0.2397 (using *Gene Expression*) to 0.4089 (using *Mutations*)

Table 5.2 – AUC-PR performance comparison among GNNs for variations of types of omics data used as node features.

Features	PPI Network	Algorithm		
		GAT	GCN	GraphSAGE
Mutations	HPRD	0.3565	0.4470	**0.4089**
CNA		0.2761	0.3992	0.3134
DNA Methylation		**0.3844**	0.3708	0.2402
Gene Expression		0.2806	0.2967	0.2397
Multi-Omics		0.3258	**0.4984**	0.3317
Mutations	MULTINET	0.2142	0.3523	**0.3842**
CNA		**0.2865**	0.3277	0.1996
DNA Methylation		0.2219	0.3185	0.1770
Gene Expression		0.2562	0.2506	0.1981
Multi-Omics		0.2182	**0.3839**	0.2671
Mutations	IREF	0.2606	**0.4344**	**0.3550**
CNA		0.3027	0.2844	0.2036
DNA Methylation		**0.3088**	0.2509	0.1482
Gene Expression		0.2211	0.2699	0.1850
Multi-Omics		0.2144	0.4315	0.2001

for GraphSage. GraphSage consistently showed the largest performance variations between minimum and maximum values across the three PPI networks (HPRD: 70.58%, MULTINET: 117.06%, and IREF: 139.54%), while GAT showed the smallest (HPRD: 39.22%, MULTINET: 33.75%, and IREF: 44.02%), indicating that GraphSage is more sensitive to changes in node features than GAT and GCN in our domain.

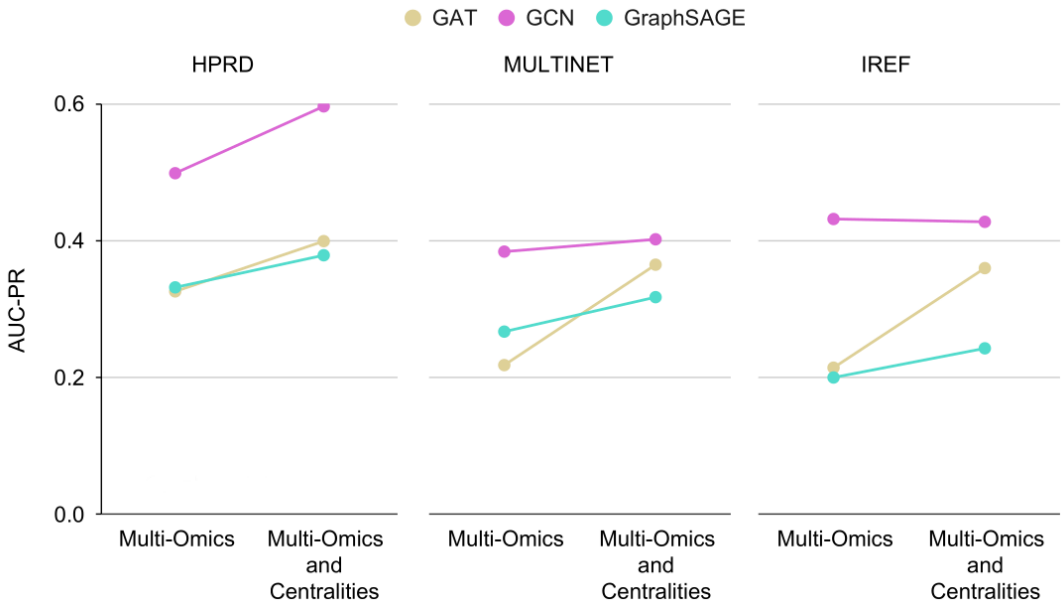
Additionally, the ranking of feature types based on performance is not consistent across all networks and algorithms. For instance, while *Mutation* yielded the highest performance in four out of the nine combinations of PPI network and GNN algorithm, *Multi-omics* and *DNA methylation* achieved the best performance in two combinations each. *CNA*, on the other hand, was the top-performing feature set in only one combination. This result is biologically plausible, since DNA mutations are the primary cause of most cancers. Nonetheless, integrating the four types of omics in the multi-omics approach still seemed advantageous as it was able to improve performance in some cases, especially for the GCN algorithm. Thus, for the next experiment, we select the multi-omics strategy as our default configuration for the node features vector.

5.2.2 Can GNNs benefit from the inclusion of node centrality measures as node features?

Node centrality measures, such as degree and betweenness, are commonly employed as handcrafted graph features for ML approaches in disease gene prediction. This was the traditional approach to explore the information contained in biological networks (e.g., PPI networks) in node prediction tasks before the advent of graph-based learning algorithms (ATA et al., 2021). Given that node centrality measures were relatively successful in capturing the global and local graph structure of nodes in previous efforts, we aimed to investigate whether they can increase model reliability and improve performance metrics when used as features along with multi-omics data.

We extracted four node centrality measures - degree, betweenness, closeness, and clustering coefficient (see Section 2.2.3 for definitions) - from each of the PPI networks using the igraph package (CSARDI; NEPUSZ, 2006). These centrality measures were used exclusively as additional features for the PPI network from which they were computed since they are node features that rely on the graph structure. We then conducted a new round of model training and evaluation using a node feature vector composed of multi-omics data and centrality measures.

Figure 5.1 – Analysis of the impact of the inclusion of centrality measures as node features in the performance of GNNs.



Source: Prepared by the author

Figure 5.1 shows the results of this experiment for the test set, compared against the results when considering only multi-omics features (Section 5.2.1). We observed

that centrality measures provided important information for training the model since they improved the performance in most scenarios. The only exception was the GCN model for the IREF network, which did not benefit from the additional features. The predictive performance increased up to 67.81% for the GAT model applied to the IREF network. In general, the GAT model was the most benefited when adding centrality measures to the node features vector, although the GCN model also significantly improved the predictive performance after the inclusion of centrality features in the HPRD network.

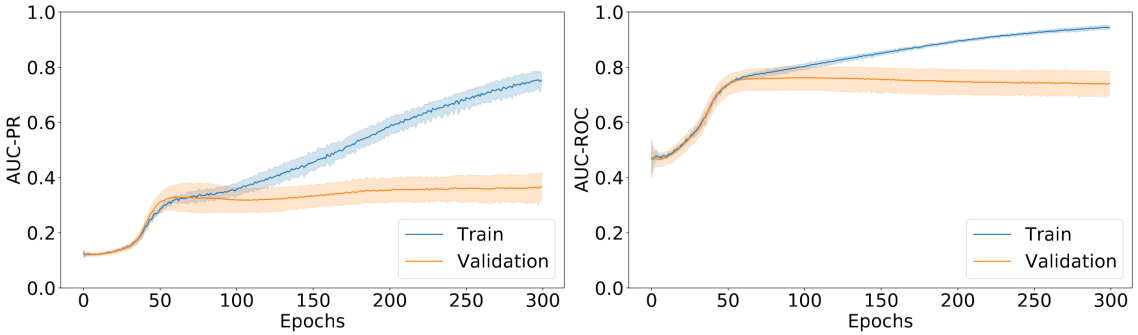
Whereas Figure 5.1 presents the results for the independent test set, which is our main interest for performance comparison, we also provide the learning curves for the HPRD network when using multi-omics and centralities as node features in Figure 5.2 (the same analyses can be seen for MULTINET and IREF networks in the Appendix B, Figures B.1 and B.2). These figures allow us to evaluate the training and validation performance history of the algorithms through 300 epochs. Values represent the mean and standard deviation across 5-fold CV. We note that GAT (Figure 5.2a) and GraphSage (Figure 5.2c) tend to quickly overfit in the first 50 epochs when using default hyperparameters. Moreover, GAT shows the highest performance variation among the five repetitions.

Another interesting point in Figure 5.2 is that the ROC curves (right column) are all shifted upwards compared to the PR curves (left column). This means that the model evaluation through the AUC-ROC metric suggests higher predictive performance. However, this is an artifact of our dataset, which contains a low proportion of instances in the positive class (*i.e.,* true CDGs).

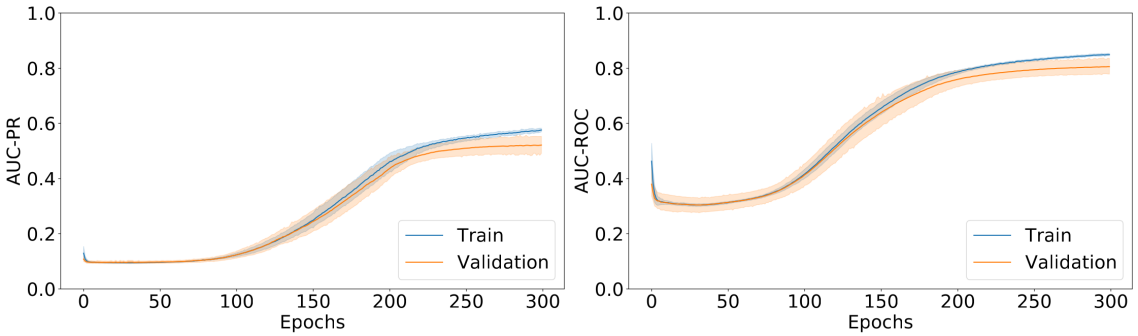
To better illustrate this issue and reinforce our choice to focus the comparative analysis on the AUC-PR values, we provide a direct comparison among accuracy, AUC-ROC, and AUC-PR for the test set (Table 5.3). We observed that accuracy and AUC-ROC are always higher than AUC-PR for every combination of GNN algorithm and PPI network. While the AUC-PR values range from 0.2427 to 0.5960 in these experiments, accuracy varies between 0.8289 and 0.9213, and AUC-ROC varies between 0.6629 and 0.8453. Finally, given that the use of centrality measures as node features improve the predictive performance for graph-based learning algorithms, from this point forward, we will always use the node feature vector composed of multi-omics data and centrality measures as the default feature set.

Figure 5.2 – Training and validation performance for the HPRD network in (a) GAT, (b) GCN, and (c) GraphSAGE in terms of AUC-PR (left column) and AUC-ROC values (right column).

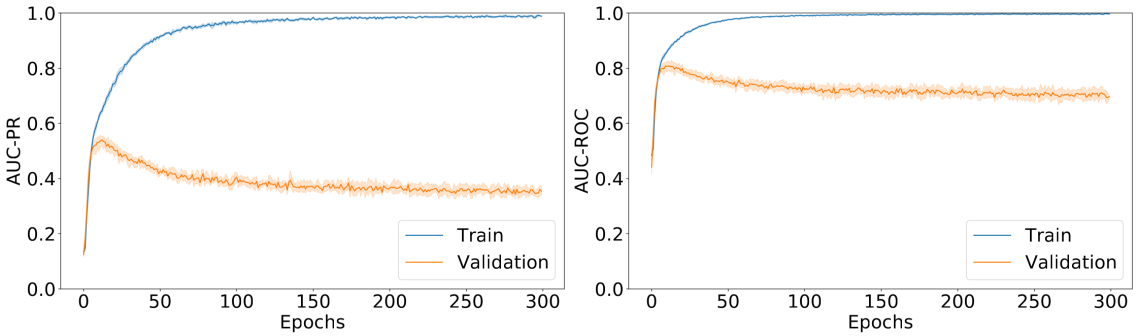
(a) GAT



(b) GCN



(c) GraphSAGE



Source: Prepared by the author

5.2.3 What is the best strategy to handle class imbalance in the node prediction problem addressed?

Class imbalance is a recurring problem when working with cancer genomics and cancer driver prediction, as the target factor, either a true driver mutation or a true CDG, is always the most rare phenomenon when compared to passenger mutations or neutral genes. As noted in Chapter 3, other authors also face this inherent obstacle, although only a few reviewed papers clearly discuss the strategy adopted to address this issue.

In Section 2.2.5, we presented the theoretical foundation for most common strate-

Table 5.3 – Different performance metrics for predictions on the test set using a combination of multi-omics and centrality measures as node features.

Algorithm	PPI Network	Performance metric		
		Accuracy	AUC-ROC	AUC-PR
GAT	HPRD	0.8580	0.7969	0.3992
	MULTINET	0.9043	0.8312	0.3648
	IREF	0.9120	0.8453	0.3598
GCN	HPRD	0.8889	0.8418	0.5960
	MULTINET	0.9123	0.8162	0.4019
	IREF	0.9213	0.8246	0.4274
GraphSAGE	HPRD	0.8289	0.7132	0.3786
	MULTINET	0.9011	0.7201	0.3174
	IREF	0.8827	0.6629	0.2427

gies to deal with class imbalance, which we experimentally compare in the current section: Random Undersampling, Balanced Cross-entropy, and Focal Loss. For each GNN algorithm and PPI network, we applied five different strategies to mitigate the class imbalance among positive and negative examples of CDGs. Besides Random Undersampling and Balanced Cross-entropy, we selected three variations of the Focal Loss function from

Table 5.4 – AUC-PR peformance comparison between strategies to address the class imbalance issue. The values refer to predictive performance on the test set.

Strategy	PPI Network	Algorithm		
		GAT	GCN	GraphSAGE
Cross-entropy	HPRD	0.3992	**0.5960**	0.3786
Undersampling		0.3275	0.5479	0.3332
Balanced cross-entropy (α=0.85)		**0.4215**	0.5918	0.4069
Focal loss (γ=0.5; α=0.50)		0.3873	0.5235	0.4230
Focal loss (γ=1; α=0.25)		0.3581	0.5756	0.3985
Focal loss (γ=2; α=0.25)		0.3792	0.5583	**0.4480**
Cross-entropy	MULTINET	0.3648	0.4019	0.3174
Undersampling		0.2817	0.4521	0.2188
Balanced cross-entropy (α=0.85)		0.3551	**0.4661**	**0.3929**
Focal loss (γ=0.5; α=0.50)		**0.4143**	0.4177	0.3264
Focal loss (γ=1; α=0.25)		0.3363	0.2978	0.3196
Focal loss (γ=2; α=0.25)		0.2753	0.4481	0.3822
Cross-entropy	IREF	0.3598	0.4274	0.2427
Undersampling		0.1714	0.4384	0.1887
Balanced cross-entropy (α=0.85)		0.3135	**0.4841**	**0.3013**
Focal loss (γ=0.5; α=0.50)		0.3546	0.4664	0.2296
Focal loss (γ=1; α=0.25)		0.2977	0.3776	0.2832
Focal loss (γ=2; α=0.25)		**0.3917**	0.4488	0.2928

the original article (LIN et al., 2017), which presents a series of experiments and arrives at a set of best combinations for the hyperparameters γ and α. We used γ = 0.5 and α = 0.50, γ = 1 and α = 0.25, and γ = 2 and α = 0.25. We note that the Balanced Cross-Entropy resembles a Focal loss function with $\gamma = 0$ and with the α parameter indicating the weight to balance the importance of positive/negative instances (here, we adopted $\alpha = 0.85$). The Cross-entropy cost function was used in two moments: in the initial experiments conducted previously (considered as our "baseline") and in the scenario where the Random Undersampling technique is applied to data. Table 5.4 presents the results of our experiments comparing strategies to deal with class imbalance. The best results for each algorithm and PPI network are highlighted in boldface.

Overall, we observed performance improvements in most cases where class imbalance correction was carried out. Balanced Cross-Entropy yielded the most promising results, while the undersampling strategy was generally less effective. Although the results are still below the desired level of predictive performance, they show potential for further improvement through hyperparamenter optimization in the case of Balanced Cross-Entropy and Focal Loss. Since the Balanced Cross-Entropy is a special case of a Focal Loss, we defined Focal Loss as the most promising approach to handle class imbalance in our domain.

5.2.4 How do traditional ML algorithms compare to GNNs when applied to the same omics data?

As part of our investigation, we aimed to understand the potential benefits of using the full network structure through graph-based learning methods compared to relying solely on handcrafted features such as omics data and centrality measures for model learning. To this end, we conducted a series of experiments with four traditional ML algorithms widely used in the bioinformatics domain: SVM, Random Forests, Gradient Boosting Trees, and Artificial Neural Networks. For more information on the theoretical background, please refer to Chapter 2.

For the traditional ML algorithms, we had a conventional classification task in which the algorithm was only provided with a dataframe containing instances (*i.e.,* genes) with their features vector and corresponding label. The structure of the PPI network was not used when the features were based solely on omics data. Therefore, the predictive performance was entirely associated with the informative power of omics data. Never-

Table 5.5 – AUC-PR peformance comparison among traditional ML algorithms and variations in the omics data used as features.

Features	PPI Network	Algorithm			
		SVM	RF	GBT	ANN
Mutations	HPRD	0.3272	0.3664	**0.3753**	0.2252
CNA		0.1950	0.1702	0.1846	0.1619
DNA Methylation		0.1435	0.1519	0.1487	0.1324
Gene Expression		0.1368	0.1559	0.1564	0.1563
Multi-Omics		0.2595	0.3566	0.3565	0.2119
Mutations	MULTINET	0.2819	0.3066	**0.3212**	0.1726
CNA		0.1247	0.1136	0.1169	0.1220
DNA Methylation		0.0951	0.1122	0.1089	0.0853
Gene Expression		0.0997	0.1071	0.1151	0.0946
Multi-Omics		0.2025	0.2976	0.3066	0.1614
Mutations	IREF	0.2598	0.2929	**0.3087**	0.1848
CNA		0.1204	0.1201	0.1268	0.1165
DNA Methylation		0.0969	0.1023	0.1044	0.1081
Gene Expression		0.0906	0.1016	0.1077	0.0981
Multi-Omics		0.1958	0.2869	0.3029	0.1519

theless, we performed experiments for each PPI network since the set of genes analyzed differed among networks. To ensure consistency, we kept the hyperparameters constant through all experiments and evaluated the impact of omics levels on each algorithm.

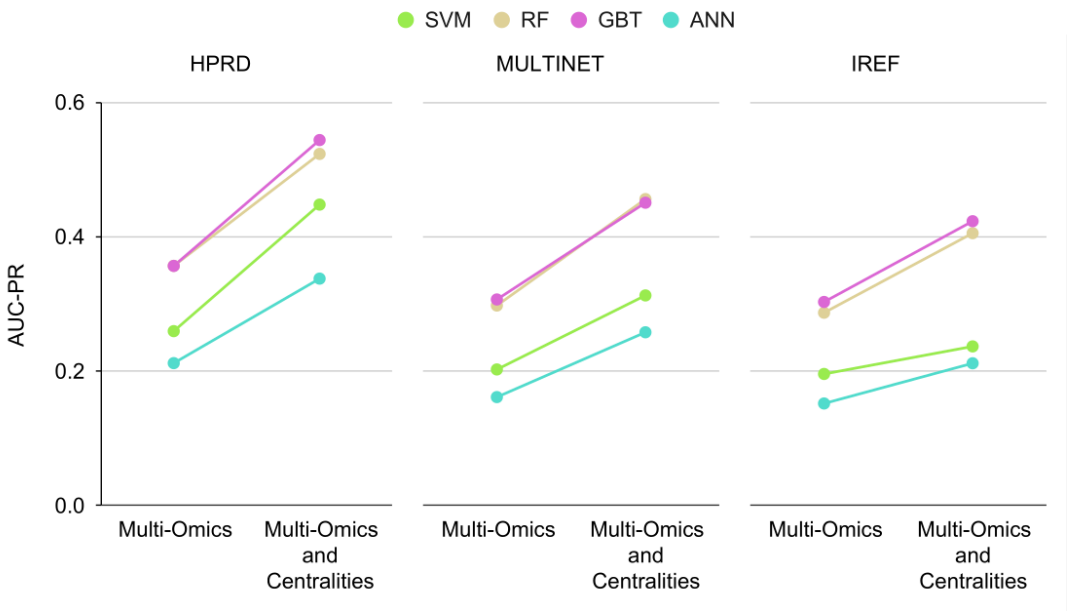
Table 5.5 summarizes the results for the test set. The best performance per PPI network is highlighted in boldface, and the values refer to the average performance across 5-fold CV. The traditional ML algorithms performed better on mutation data, but the overall performance of these models was worse than those achieved by GNN algorithms. Among the four traditional ML algorithms applied, GBT was the top-performing one for all PPI networks, followed by RF. Multi-omics data was the second-best feature set for all PPI networks.

We also assessed the effect of using class weights during training to adjust the cost of misclassifications and avoid concentrating the learning on the majority class, similar to the experiments with GNNs. However, these experiments did not yield any significant results since most of the models did not exceed the standard performance presented in Table 5.5 upon class imbalance mitigation with class weights. Therefore, we do not report these results here.

Finally, we assessed the impact of incorporating node centrality measures as features in traditional ML algorithms, inspired by the promising results achieved with GNNs.

We used the multi-omics features vector as our baseline and added measures for node degree, betweenness, closeness, and clustering coefficient for each PPI network. The results presented in Figure 5.3 demonstrate that the inclusion of centrality measures as an extra source of information improved the performance of all models, surpassing the results obtained with individual omics data or their combination (i.e., multi-omics). We can also clearly visualize that the magnitude of performance change upon addition of centrality measures was greater for traditional algorithms than for GNNs (Figure 5.1), which is expected, since GNNs by definition already have information about the graph (*i.e.,* PPI network) in the learning process. Another interesting observation is that the decision tree-based algorithm had a very close performance in this prediction task, and also very similar behavior with the inclusion of centrality features.

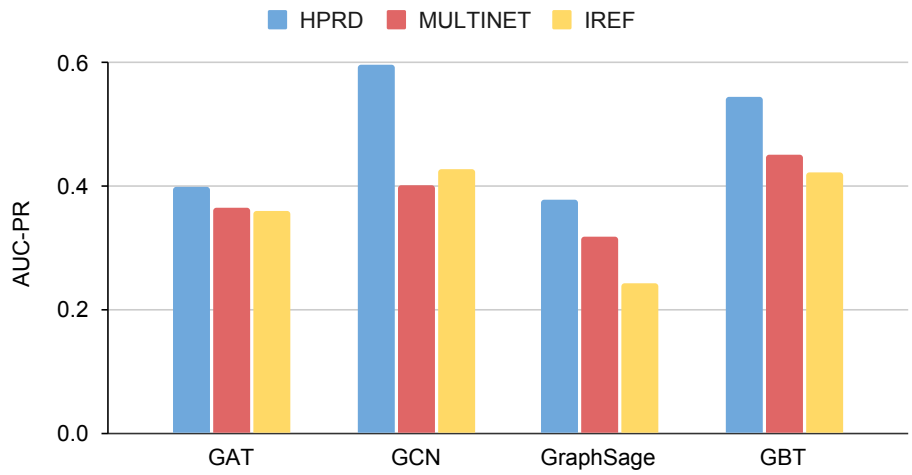
Figure 5.3 – Analysis of the impact of including node centrality measures as features in the traditional ML algorithms.



After comparing the results of experiments using multi-omics data and node centralities among the best-performing traditional ML algorithm (*i.e.,* GBT) and GNNs, as depicted in Figure 5.4, we can conclude that GBT had a competitive performance in relation to graph-based models. GBT achieved better results than GAT and GraphSAGE for all PPI networks. When we compared GCN and GBT, we noticed that GCN outperformed GBT in two of the analyzed networks, while GBT achieved the highest score among all the evaluated approaches for MULTINET.

Overall, our experiments indicate that the task of predicting CDGs is, in general, more challenging for traditional ML algorithms than for GNNs. However, traditional

Figure 5.4 – Comparison between GNNs and GBT using multi-omics and node centralities as features.



ML algorithms stand out in some cases, as in the use of tree-based methods with mutation data. Additionally, our analyses including centrality measures highlighted their relevance as a complement to multi-omics data. This was particularly evident in this set of experiments, where centralities played a significant role in enhancing the predictive performance obtained with multi-omics data for algorithms that learn from tabular data. These observations underscore the importance of considering the network structure for CDG prediction and suggest that centralities may play a critical role in such scenarios.

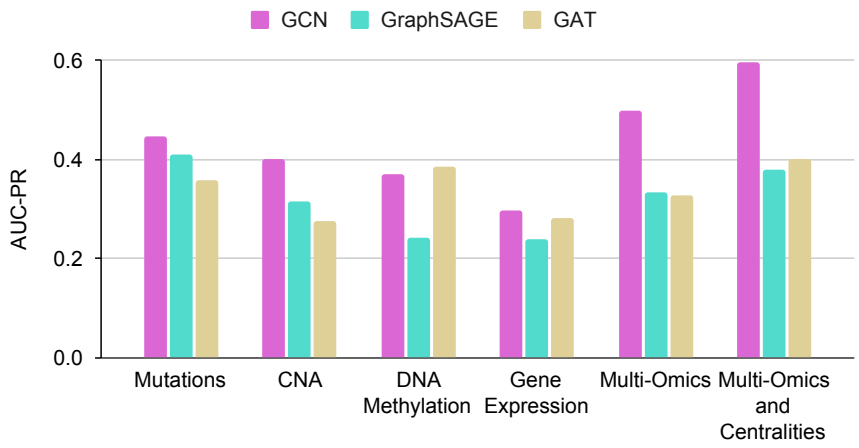
5.2.5 Which graph neural network algorithm performs best for CDG prediction?

The last and main question raised in this chapter refers to investigating which of the GNN algorithms is the most promising for the task of predicting CDGs. After conducting several exploratory experiments comparing various approaches, we aimed to select one algorithm for further development through a more detailed methodological process, including hyperparameter optimization, in an effort to improve its predictive performance for this task.

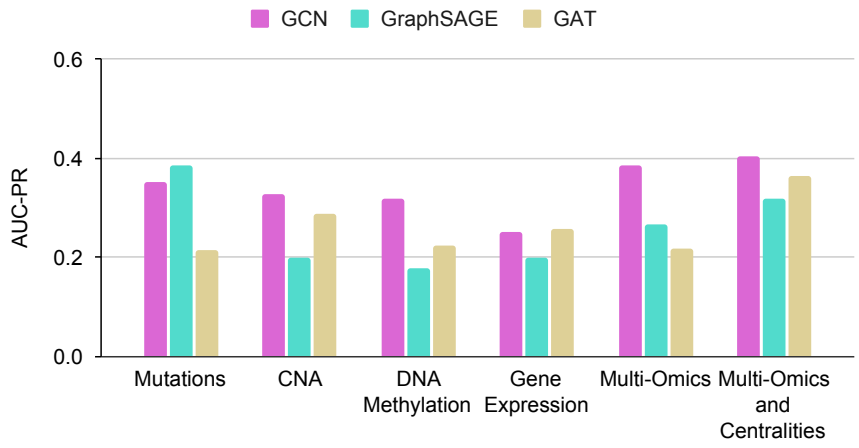
Figure 5.5 provides a comparison of the test performance measures of the three evaluated GNN algorithms. Based on our analysis of these results, as well as the findings discussed in Sections 5.2.1 to 5.2.4 in this chapter, we concluded that GCN is the most promising algorithm for accurately predicting CDGs. GCN demonstrated superior performance in nearly all scenarios tested, in terms of both PPI networks and the composition of the node features vector. Additionally, the highest performance achieved in this

Figure 5.5 – Test set cross entropy performance of (a) HPRD, (b) MULTINET, and (c) IREF networks.

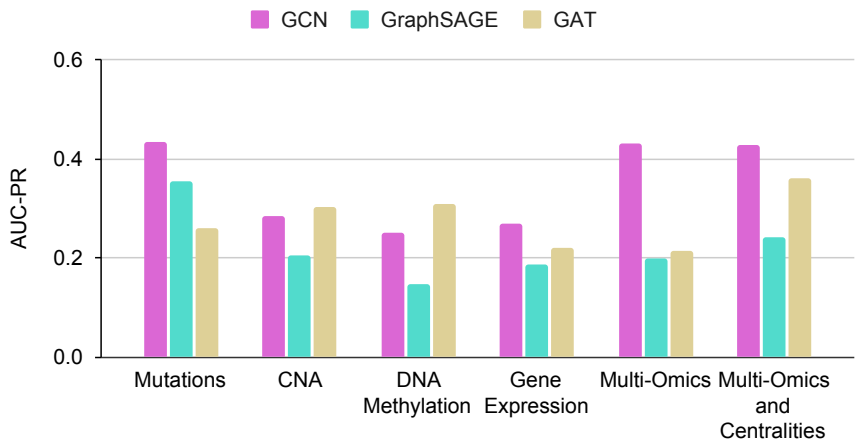
(a) HPRD



(b) MULTINET



(c) IREF



experimental comparison was associated with the use of GCN in conjunction with multi-omics and centralities. Consequently, we have chosen the GCN model and the feature set comprising multi-omics and centrality measures as our standard approach for the learning algorithm and data representation. In the next chapter, we will provide a more in-depth investigation of this strategy, including efforts to optimize hyperparameters.

5.3 Summary

The primary focus of this chapter was to investigate the potential of GNN algorithms and distinct feature definition strategies for predicting CDGs. A series of experiments were carried out to extract an understanding about how, in general, GNN algorithms perform in this task. Besides comparing three graph-based models for three PPI networks, we experimentally compared different strategies to define instances features vector and to mitigate class imbalance. Moreover, we also compared the GNN algorithms with four traditional ML algorithms. Based on the results of our extensive experiments, we have concluded that graph-based learning algorithms generally outperform other approaches in the prediction task by utilizing information about node connectivity for model training and classification. Among the GNNs applied, GCN demonstrated the most promising results, displaying consistent and robust performance across all evaluated scenarios. Our findings also indicate that traditional ML algorithms are highly sensitive to the type of omics data used, and tend to perform better when incorporating mutation information. Furthermore, we have observed that node centrality measures calculated from PPI networks can be highly informative and beneficial for the prediction task, even for graph-based algorithms. Given that GCN proved to be the most effective in detecting CDGs in different scenarios, this algorithm will be the subject of a more detailed study of hyperparameters optimization and ensemble prediction strategies using GCN in Chapter 6.

6 GCN-BASED MODELS FOR CANCER DRIVER GENE PREDICTION

In this chapter, we provide a deeper examination of the prediction of CDGs using the GCN algorithm, which showed the most promising performance in the experimental comparison presented in Chapter 5. We detail the investigation of the best set of hyperparameters for the three smaller PPI networks and the three larger PPI networks. We also report the performance of GCN models trained with each of the six PPI networks using a common independent test set. Finally, we compare the predictive performance of the models trained on the collected PPI networks with those trained on the UNION PPI network defined in our work (as explained in Section 4.1.1) and ensemble-based approaches (as detailed in Section 4.4).

6.1 Experiments setup

In this series of experiments, we aimed to optimize the hyperparameters of the GCN algorithm to try to achieve higher predictive performance. Specifically, we conducted hyperparameter optimization for each of the PPI networks collected from the literature, namely HPRD, MULTINET, IREF, CPDB, PCNET, and STRING.

To ensure a fair comparison between models for the same set of test genes, we randomly split the main dataset into training and test sets considering only positively and negatively labeled genes common to all PPI networks for the test set. With this, we generated a fixed test set used in these experiments. We then used 5-fold cross-validation in the training set to evaluate the performance during hyperparameter optimization. All models were trained using the multi-omics data and centrality measures as node features.

After hyperparameter optimization, we trained a final model using the best hyperparameter configuration and the complete training set, and applied it to predict the label for the independent test set and for the unlabeled nodes present in each PPI network. The first dataset allowed us to evaluate the final performance of the model, while the second dataset aimed to suggest potential candidate driver genes for further investigation.

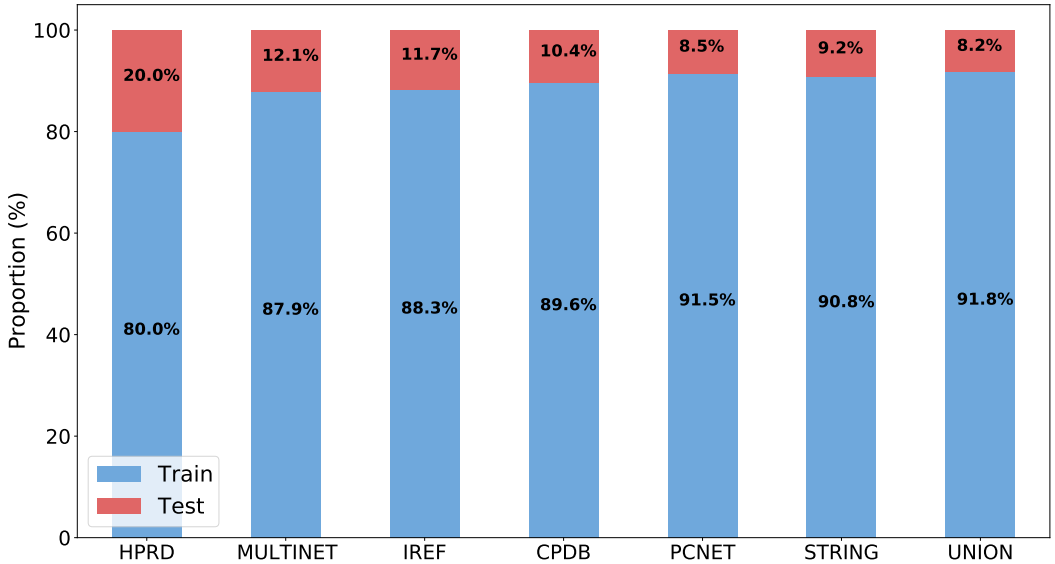
In the following sections, we describe the creation of the independent test set used in these experiments and the hyperparameter optimization strategy adopted.

6.1.1 Creation of an independent test set

To create the independent test set for this series of experiments, we searched for genes present in all six PPI networks collected. We identified a set of 8,068 genes, which represents 41.2% of the total number of genes included in all networks (*i.e.,* the union of genes among all PPI networks). Taking as reference the smallest PPI network, which is the HPRD network, we estimated the number of genes to withhold for the test considering an 80:20 ratio for training and test sets. We found that about 1,134 labeled genes should be reserved as a test set to follow this proportion of data split for the smallest network.

Then, we randomly selected 1,134 labeled genes among those that are common to all six PPI networks in a stratified fashion, generating an independent test set with 159 driver genes (*i.e.,* positive examples) and 975 passenger genes (*i.e.,* negative examples). This test set was fixed during our experiments in the sense that it was the same test set used for all PPI networks. The strategy adopted causes the proportion of the test set to vary among PPI networks, as shown in Figure 6.1. For example, the test set represents only 8.5% of the total number of genes in the PCNET network. However, reserving 20% of labeled data for the test set based on the number of genes comprised in the largest network would cause a considerable restriction of data for training the models from small PPI networks such as the HPRD. Therefore, in this chapter, all the results reported for the independent test set will refer to exactly the same set of 159 driver genes and 975 passenger genes.

Figure 6.1 – Proportion of training and test sets for all PPI networks considering a fixed set of labeled genes for testing.



Source: Prepared by the author

6.1.2 Strategy for hyperparameter optimization

Table 6.1 lists the hyperparameters that were chosen for the investigation conducted in this chapter and the values that were initially included in the grid search, selected based on the recommendations discussed in Chapter 4.

Table 6.1 – Hyperparameters evaluated for GCN training with HPRD, MULTINET, and IREF networks

Hyperparameter	Values tested
Number of layers	2
Number of neurons/units on each layer	64; 128; 256
Activation function	ELU; ReLU
Dropout	0.001; 0.01; 0.1
Learning rate	0.00001; 0.0001; 0.001
Number of epochs	500
Focal loss (gamma)	0; 0.5; 1; 2
Focal loss (alpha)	0.25; 0.50; 0.75; 0.90

The training process for the GCN algorithm involves a total of 864 combinations of hyperparameters related to the learning algorithm and to the focal loss function. Evaluating all these combinations can be computationally expensive due to the complexity of the PPI network, especially for the three largest PPI networks, resulting in long execution times. Therefore, we adopted a strategy to simplify the search grid in these cases. First, we evaluated all the 864 combinations of hyperparameters for the three smallest PPI networks, namely HPRD, MULTINET, and IREF, which have less than 140,000 edges. Based on the results obtained for this first round of experiments, we carried out a performance impact analysis of hyperparameter values.

For this analysis, we defined a set of thresholds for AUC-PR performance and counted the number of hyperparameter combinations that produced results higher than the threshold, for each cutoff defined. The thresholds $\{0.2, 0.4\}$ were used for the three PPI networks, but the set was extended with higher threshold values that were closer to the maximum AUC-PR performance obtained for each PPI network analyzed. The higher the performance threshold applied, the fewer the number of hyperparameter combinations attached to the training process that achieved that minimum performance mark. The intention was to extract a fine-grained analysis of the relation of hyperparameters values and AUC-PR performance for top-performing models.

Then, we analyzed the most frequent values of hyperparameters for each performance threshold. We investigated, for instance, the existence of specific hyperparameter

values that were never related to high performance, or, in contrast, that were very frequent among the models with outstanding results. We used the findings of this analysis to prune or to adjust the grid of hyperparameter values for the second round of experiments involving the three largest PPI networks.

This approach was implemented in response to the intricacy of GNN techniques. Specifically, the execution time of a training epoch for a small network, such as HPRD, is typically measured in milliseconds, whereas larger networks, such as STRING, require approximately 8 seconds to complete a single epoch, on average. Furthermore, even the GCN algorithm, which is the most expeditious of the three GNN algorithms examined in the preceding chapter, still necessitates a nontrivial amount of computational time.

6.2 Results

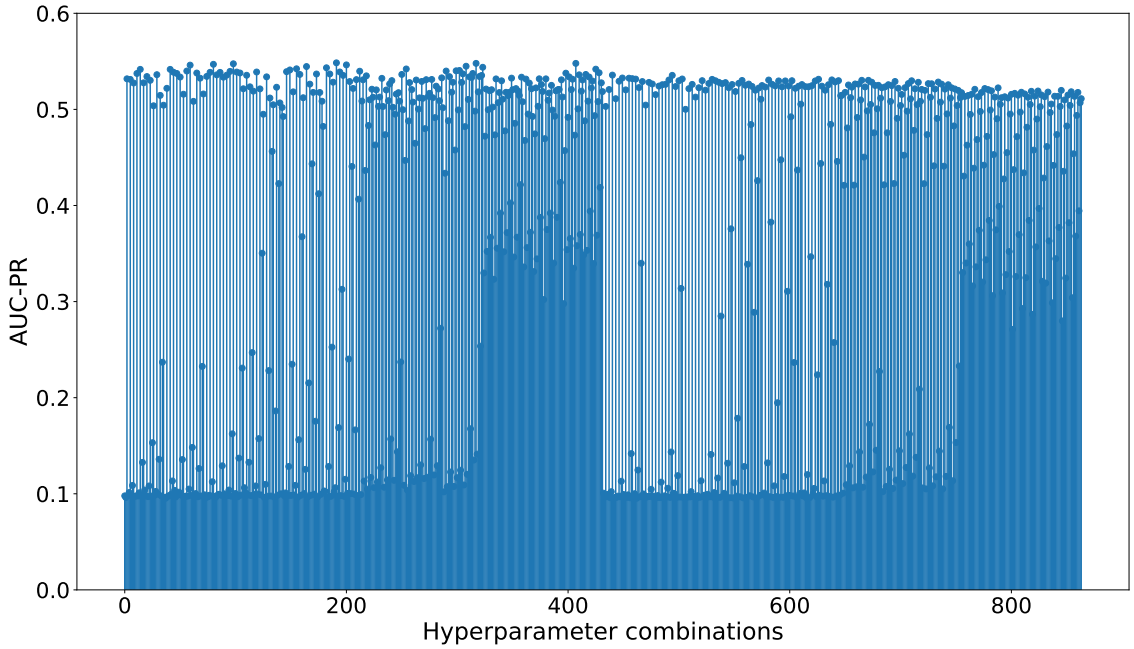
6.2.1 Hyperparameter optimization for the three smallest PPI networks

The hyperparameter values used for the HPRD, MULTINET, and IREF networks were presented in Table 6.1. As previously mentioned, a total of 864 configurations were assessed for the smallest PPI networks. The average performance of models in the validation set (*i.e.,* the independent folds in the 5-fold CV) was analyzed for each hyperparameter configuration. Observing the general panorama of this evaluation, as summarized in Figure 6.2 for the HPRD network, we can note that there are hyperparameters combinations that were clearly not promising in terms of the achieved performance, while others stood out in this aspect. In other words, there is a discernible trend in higher and lower performance according to the hyperparameters configuration adopted. This trend was also observed for the analysis of the results obtained for the two other smallest networks, MULTINET and IREF, as shown in the appendix (Figure C.1).

To illustrate how we conducted the impact analysis of hyperparameter combinations, we show in the Table 6.2 the results for the HPRD network considering an AUC-PR performance threshold of 0.53. Filtering the results by this threshold yielded a total of 97 hyperparameter combinations[1]. We observed, for instance, that higher learning rates occurred more frequently, to the extent that the lowest learning rate (*i.e.,* 0.00001) employed

[1]For comparison purpose, when considering the HPRD PPI network, using the threshold of AUC-PR > 0.2 we obtained 556 combinations of hyperparameter values, using the threshold AUC-PR > 0.4 we obtained 457 combinations, and using the threshold AUC-PR > 0.5 we obtained 350 combinations.

Figure 6.2 – Performance analysis of all hyperparameter combinations for the HPRD network.



Source: Prepared by the author

Table 6.2 – Occurrence frequency of hyperparameters values for the models trained with the HPRD network using AUC-PR threshold of 0.53. A total of 97 combinations met the criteria.

Activation function		Alpha (α)		Gamma (γ)	
ReLU	**84.5%**	0.25	**36.0%**	0	22.6%
ELU	15.4%	0.5	22.6%	0.5	**27.8%**
		0.75	23.7%	1	**27.8%**
		0.9	17.5%	2	21.6%
Number of neurons		**Learning rate**		**Dropout**	
64	**39.1%**	0.00001	0.0%	0.001	24.7%
128	30.9%	0.0001	9.2%	0.01	27.8%
256	29.8%	0.001	**90.7%**	0.1	**47.4%**

was never selected among the 97 combinations that met the AUC-PR threshold applied. In summary, higher learning rates seem to produce more effective classification results. Some hyperparameters are still inconclusive as to their impact on the predictive power, especially because their impact varies according to the network analyzed.

Only two hyperparameters had clear and similar patterns across the three PPI networks that were significant for planning the next round of hyperameraters optimization: the learning rate and the activation function. Thus, for the largest PPI networks, it was possible to reduce the number of parameter combinations by approximately one-third just by adjusting two hyperparameters, the activation function, for which we only maintained the "ReLU" function, and the learning rate. For the learning rate, we removed the two smallest values from the original grid, and used 0.001 and 0.005, where the latter was a new value included for training the largest PPI networks.

Figure 6.3 shows the performance variation for the top five combinations of hyperparameters in the experiments with (a) HPRD, (b) MULTINET, and (c) IREF networks. The validation performance in the last training epoch in each of the five folds is observed and used to report these results, with the white dot representing the mean. Additionally, the best hyperparameter combination according to the average performance in the validation sets is highlighted in green. We observed that the median performance is around 0.60 for HPRD, and around 0.5 for MULTINET and IREF. Although the median performance if very close among the top 5 configurations, some specific hyperparameter combinations lead to larger variations in performance or to the occurrence of outliers.

6.2.2 Hyperparameter optimization for the three largest PPI networks

The CPDB, PCNET, and STRING networks belong to the group of largest PPI networks, having more than 1.5 million edges. Processing these networks requires significant computational power and longer execution time for each experiment. Therefore, to address this issue, we modified the grid of hyperparameter values based on the results described in the previous section. The modified grid is shown in Table 6.3 and it reduces the number of hyperparameter combinations to a total of 288 possible combinations.

Table 6.3 – Hyperparameters evaluated for GCN training with CPDB, PCNET, and STRING networks

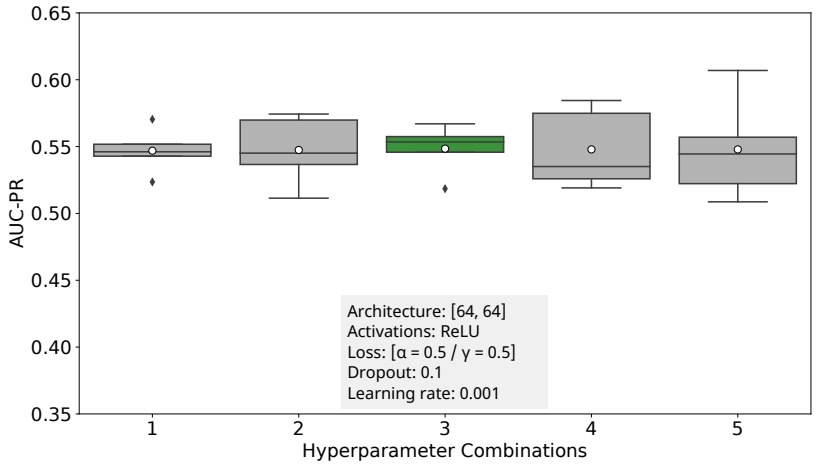
Hyperparameter	Parameter Ranges
Number of layers	2
Number of neurons/units on each layer	64; 128; 256
Activation function	ReLU
Dropout	0.001; 0.01; 0.1
Learning rate	0.001; 0.005
Number of epochs	500
Focal loss (gamma)	0; 0.5; 1; 2
Focal loss (alpha)	0.25; 0.50; 0.75; 0.90

Again, based on an overview of the results for the STRING network (Figure 6.4), we also noticed that the AUC-PR values varied according to the hyperparameter values used. However, the amplitude of changes was lower than those observed in the experiments for the smallest networks. In fact, most of the models trained using the STRING network achieved an AUC-PR value close to 0.5, with similar results obtained for the other PPI networks explored in this round of experiments (Figure C.2).

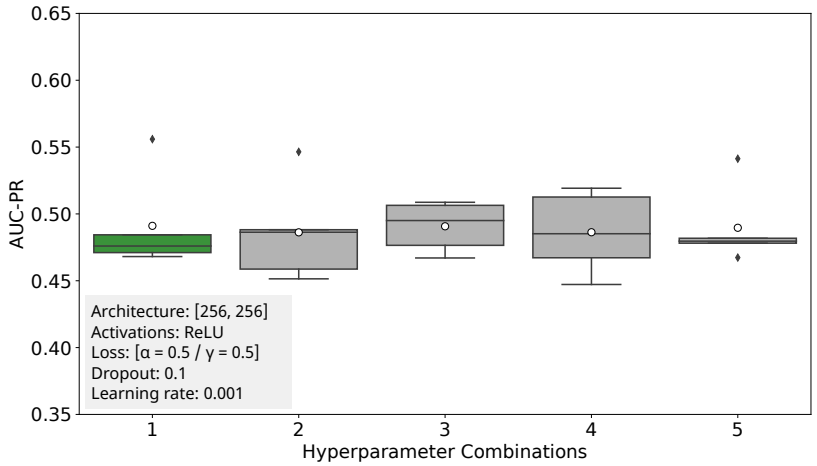
One interesting finding is that the PPI largest networks did not necessarily outper-

Figure 6.3 – Five best combinations of parameters for (a) HPRD, (b) MULTINET and (c) IREF.

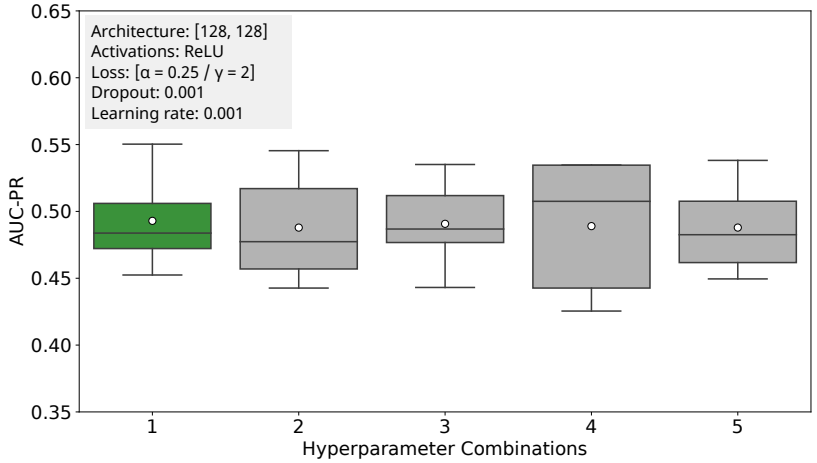
(a) HPRD



(b) MULTINET



(c) IREF



form all the smallest networks, although, theoretically, they carry out more information about the molecular aspects of genetic interactions. The best performance for validation data was obtained with PCNET (*i.e.,* AUC-PR = 0.5367). Figure 6.5 shows the five best

combinations of hyperparameters for each of the largest networks analyzed. Although this average performance of around 0.5 seems low, it shows the complexity that the studied algorithms face in dealing with class imbalance, even when applying mitigation strategies. Therefore, higher values for metrics such as accuracy and AUC-ROC may not represent the prediction goal we aim to achieve.

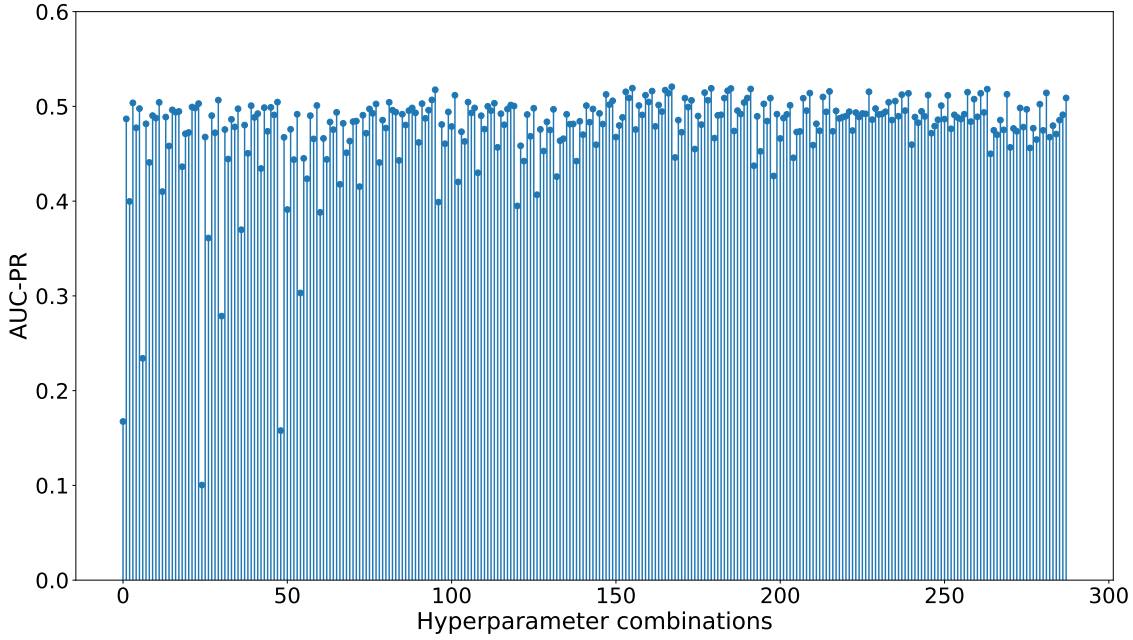
Leveraging the results of the hyperparameters optimization for all the PPI networks, the next sections will present the results for model training and evaluation using the fixed independent test set defined in Section 6.1.1.

6.2.3 Predictive performance of individual GCN-based models for the test set

Following hyperparameters optimization, the best hyperparameter values obtained for each PPI network were applied to train GCN-based models for the prediction of CDGs as explained in Section 6.1. These values are summarized in Table 6.4. We investigate the performance of the models during training and validation through the learning curves, but we focus the analysis on the performance for the fixed test set described in Section 6.1.1.

We selected three PPI networks that obtained the best results, regardless of their dimension, to present the training and validation performance in this section. Figure 6.6 presents the main metrics observed in the experiments (*i.e.,* AUC-PR and AUC-ROC)

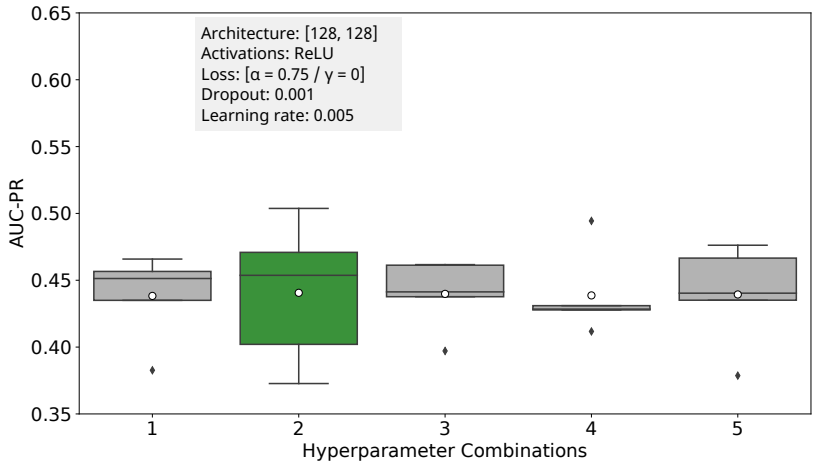
Figure 6.4 – Performance analysis of all hyperparameter combinations for the STRING network.



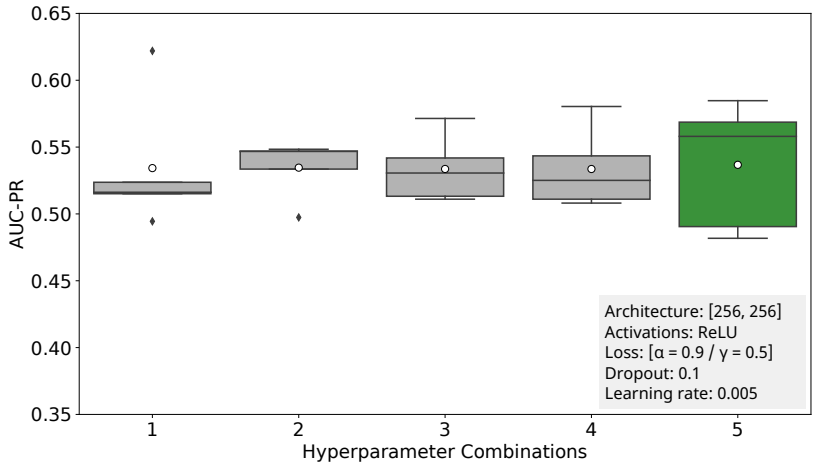
Source: Made by the author

Figure 6.5 – Five best combinations of parameters for (a) CPDB, (b) PCNET and (c) STRING.

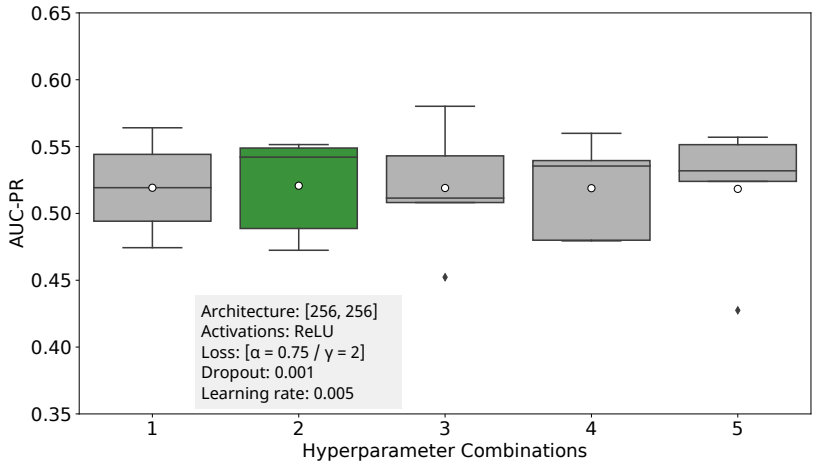
(a) CPDB



(b) PCNET



(c) STRING



for MULTINET, PCNET, and STRING. The other PPI networks are analyzed in the Appendix, Figure C.3. Furthermore, because the focal loss function has its own hyperparameters and each PPI network has been optimized with different combinations of the

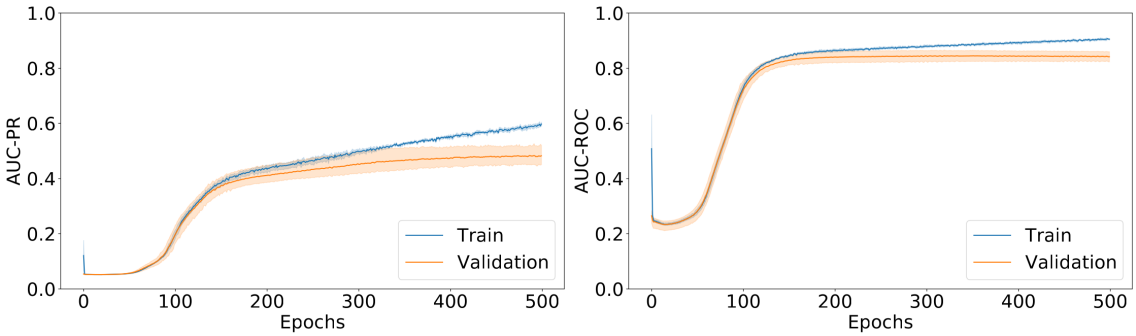
Table 6.4 – Best hyperparameter configuration found for each PPI network.

	HPRD	MULTINET	IREF	CPDB	PCNET	STRING
Layer sizes	64; 64	256; 256	128; 128	128; 128	256; 256	256; 256
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Dropout	0.1	0.1	0.001	0.001	0.1	0.001
Learning rate	0.001	0.001	0.001	0.005	0.005	0.005
Loss (alpha)	0.5	0.5	0.25	0.75	0.9	0.75
Loss (gamma)	0.5	0.5	2	0	0.5	2

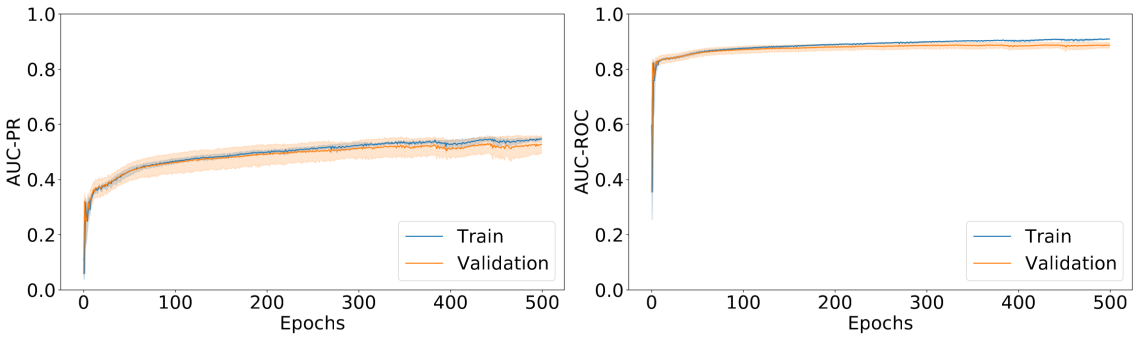
focal loss hyperparameters, the loss curves for these experiments are also provided in the Appendix (Figure C.4).

Figure 6.6 – AUC-PR and AUC-ROC performance for GCN models trained with (a) MULTINET, (b) PCNET, and (c) STRING networks.

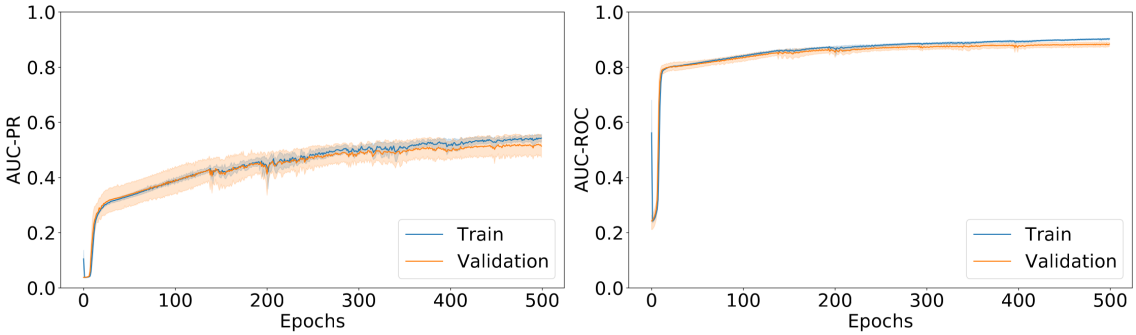
(a) MULTINET



(b) PCNET



(c) STRING



In Figure 6.6, we can note that in some cases, such as for MULTINET, the model

shows signs of overfitting from 200 epochs onwards. This was also observed for HPRD, IREF, and CPDB (Figure C.3). Despite this, we still noted an improvement in the performance of models for the test data, compared with the metrics without hyperparameter optimization reported in Chapter 5. We remind that the common test set used across the PPI networks, as presented in Section 6.1.1, contains 1134 labeled genes: 159 positives and 975 negatives examples.

Table 6.5 summarizes the performance metrics for the independent test set of the GCN models trained on each of the PPI networks. AUC-PR values varied from 0.5646 for the CPDB network to 0.6358 for the STRING network. Accuracy and AUC-ROC values, as expected, are higher, reaching values above 0.80, except for the accuracy of the PCNET network. For the three smallest PPI networks, there was an increase of 10.42%, on average, for the AUC-PR performance.

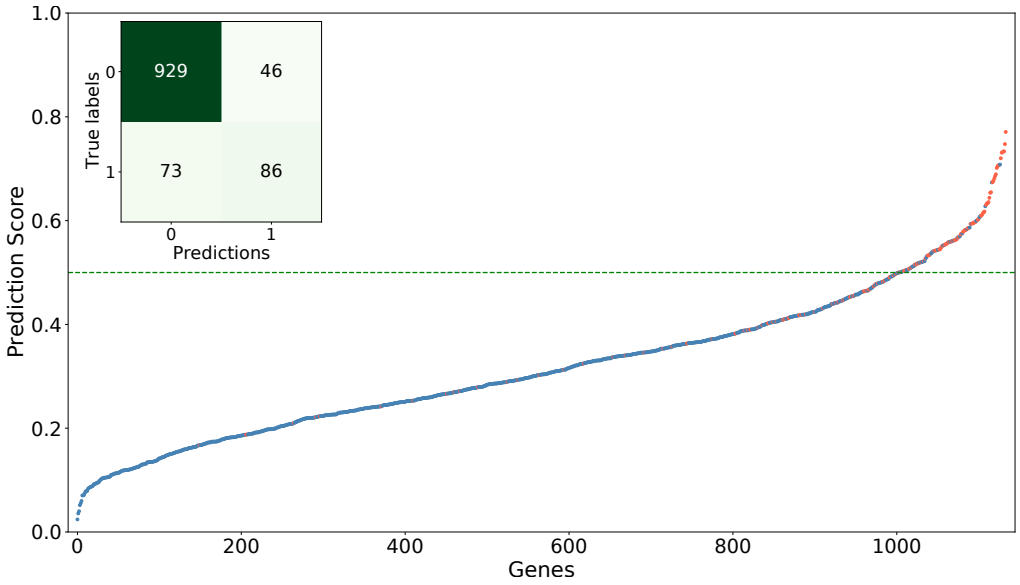
Table 6.5 – Performance of individual GCN-based models on the test set

PPI Network	Metrics		
	Accuracy	AUC-ROC	AUC-PR
HPRD	0.8889	0.8433	0.5768
MULTINET	0.8915	0.8303	0.5947
IREF	0.8827	0.8063	0.5663
CPDB	0.8642	0.8314	0.5646
PCNET	0.7795	0.8527	0.5925
STRING	**0.8951**	**0.8798**	**0.6358**

However, even with hyperparameters optimization, the performance of the GCN-based models, in general, did not exceed an AUC-PR of 0.6, such that compared to other related works, the performance of our models are still limited. Schulte-Sasse et al. (2021), for example, considered the state-of-the-art among related works, reported AUC-PR values of 0.74 and 0.68 with the GCN algorithm and the MULTINET and PCNET PPI networks, respectively. However, the authors use a different cost-sensitive loss function, distinct sets of positively and negatively labeled data, and also different data pre-processing steps – all of which can impact the results.

In comparison to other relevant studies, we can include two cancer-specific approaches, namely MutSigCV (LAWRENCE et al., 2013) and 20/20+ (TOKHEIM et al., 2016), which employed different networks and achieved AUCPR average results of 0.36 and 0.63, respectively. Additionally, the author of the EMOGI model (SCHULTE-SASSE et al., 2021) proposed a baseline approach that utilizes a Random Forest and achieved an average AUCPR of 0.58 across the analyzed networks, relying only in the extracted features, as we did in Section 5.2.4.

Figure 6.7 – Predicted probabilities for the test set using the STRING PPI network for training the GCN model. The red dots represent driver genes and the blue dots represent passenger genes. The confusion matrix is extracted considering a probability threshold of 0.5.



Source: Prepared by the author

Figure 6.7 presents a scatterplot with the predicted probability (y-axis) for each gene (x-axis) in the test set using the STRING PPI network to train the GCN model. The colors indicate the true class for the gene: the red dots represent driver genes, and the blue dots represent passenger genes. The confusion matrix considering a probability threshold of 0.5 is computed and provided at the top left corner. There were a total of 132 genes predicted as drivers by the model, 86 of them correctly assigned to the positive class.

6.2.4 Predictive performance of ensemble-based prediction models for the test set

Continuing our analysis of GCN for CDGs prediction, we investigated the collective performance of all PPI networks in the test set. Our objective is to determine whether combining multiple PPI networks can lead to superior predictions. We employed two main approaches for combining the networks: unifying the nodes and edges of the six PPI networks into a single network, which we refer to as the UNION PPI network (as explained in Section 4.1.1), and applying aggregation methods to the outputs of six individual GCN models, each trained with a specific PPI network. The latter approach utilized ensemble learning strategies outlined in Section 4.4, resulting in the Ensemble-Majority and the Ensemble-Average models.

We note that due to the large dimension and complexity of the UNION PPI net-

work (*i.e.,* 7,365,082 edges and 19,602 nodes), instead of conducting a hyperparameter optimization for a GCN trained with this specific network, we used the best hyperparameters found for the STRING PPI network (*i.e.,* the largest network collected from literature). The training and validation performance for the UNION PPI network, shown in Figure 6.8, achieved marks similar to those obtained with the other networks. We obtained an average AUC-PR value of 0.4969 in the validation data, corroborating the difficulty in predicting true positives faced in our domain. The average AUC-ROC value was 0.8854, still consistent with the results obtained individually from the other networks.

However, for the test set, we obtained an AUC-PR value of 0.5579, indicating that the simple merging of the nodes and edges of all PPI networks was not effective in improving the results compared to the individual PPI networks. Figure 6.9 shows the predicted probabilities for the labeled genes in the test set and also the confusion matrix for a probability threshold of 0.5. The GCN model based on the UNION PPI network predicted 119 genes as cancer drivers, where only 68 of them are true positives.

Finally, we compared another interesting approach to combining the knowledge contained in distinct PPI networks inspired by the ensemble learning paradigm. We analyzed the results for the Ensemble-Majority and Ensemble-Average. We note that while the former provides only predicted class labels due to the type of aggregation based on class votes, the latter provides predicted class labels and predicted class probabilities.

The general performance of these models in predicting the genes of the test set is summarized in Figure 6.10. For both models, we applied the standard probability threshold of 0.5 to extract class labels. A clear impact of the two ensemble approaches, as compared to the confusion matrices shown for the STRING (Figure 6.7) and the UNION (Figure 6.9) PPI networks, is the reduction in the number of false positives. The Ensemble-Majority predicted a total of 110 driver genes, from which 80 are true positives and 30 are false positives. The Ensemble-Average predicted a total of 92 driver genes, from which

Figure 6.8 – Learning curves for the UNION PPI network in terms AUC-PR and AUC-ROC.

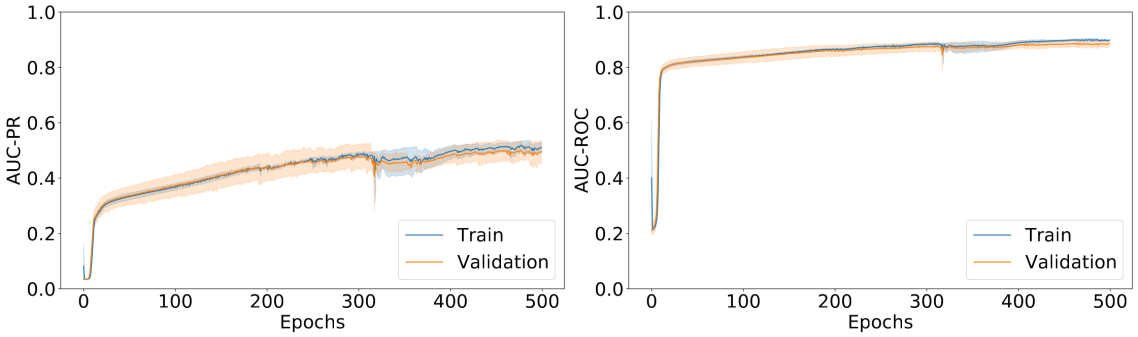
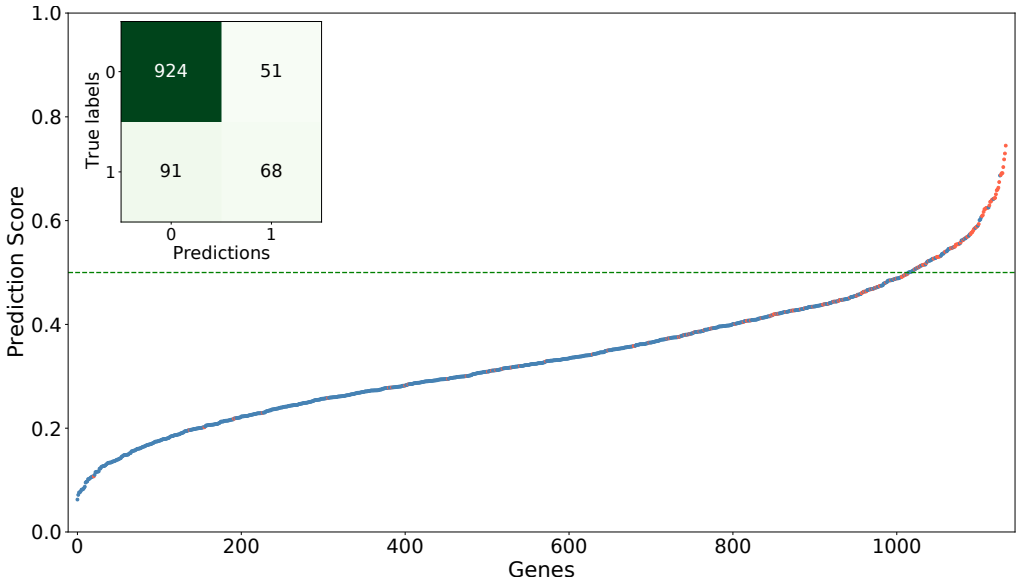
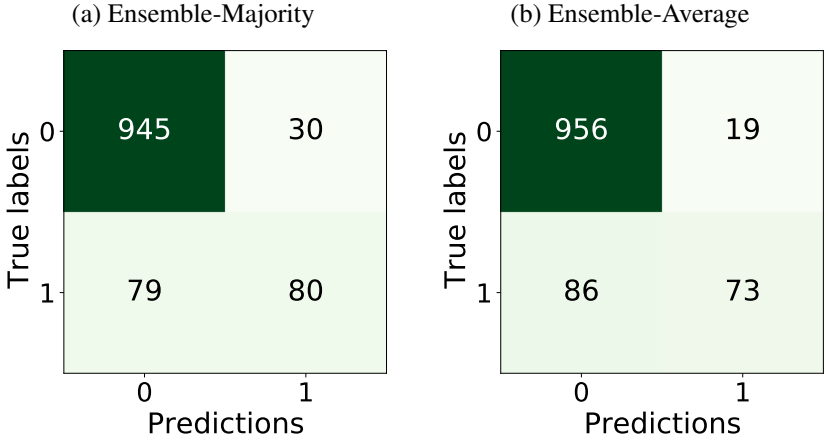


Figure 6.9 – Predicted probabilities for the test set using the UNION PPI network for training the GCN model. The red dots represent driver genes and the blue dots represent passenger genes. The confusion matrix is extracted considering a probability threshold of 0.5.



73 are true positives and 19 are false positives. However, the ensemble methods failed to identify 79 and 86 drivers (*i.e.,* false negatives) at the probability threshold used.

Figure 6.10 – Confusion matrix for the test set, analyzing the ensemble models (a) Ensemble-Majority and (b) Ensemble-Average.



We compared the predictions of individual models and ensemble-based models using a common test set. We summarized the number of predictions per class, as well as the values of TP, FP, TN, and FN from their corresponding confusion matrices. Table 6.6 shows the results, with the first line displaying the reference numbers for the test set and the following lines reporting the prediction results for each discussed model. For easier comparison, we also included the precision (prec) and recall (rec) for each model.

We observed that while the IREF-based prediction model had the highest preci-

sion, its recall was very low, correctly identifying only 28 true drivers from the test set. On the other hand, the PCNET-based prediction model had the highest recall at the expense of lower precision. The Ensemble-Average model had the most balanced performance in terms of precision and recall.

Table 6.6 – Summary of models predictions for the independent test set.

	#Drivers	#Passengers	TP	FP	TN	FN	Prec	Rec
Test Set	159	975	-	-	-	-	-	-
HPRD	85	1049	59	26	949	100	0.6941	0.3711
MULTINET	74	1060	55	19	956	104	0.7432	0.3459
IREF	30	1104	28	2	973	131	0.9333	0.1761
CPDB	173	961	89	84	891	70	0.5145	0.5597
PCNET	337	797	123	214	761	36	0.3650	0.7736
STRING	132	1002	86	46	929	73	0.6515	0.5409
Predicted UNION	119	1015	68	51	924	91	0.5714	0.4277
Ensemble-Majority	110	1024	80	30	945	79	0.7273	0.5031
Ensemble-Average	92	1042	73	19	956	86	0.7935	0.4591

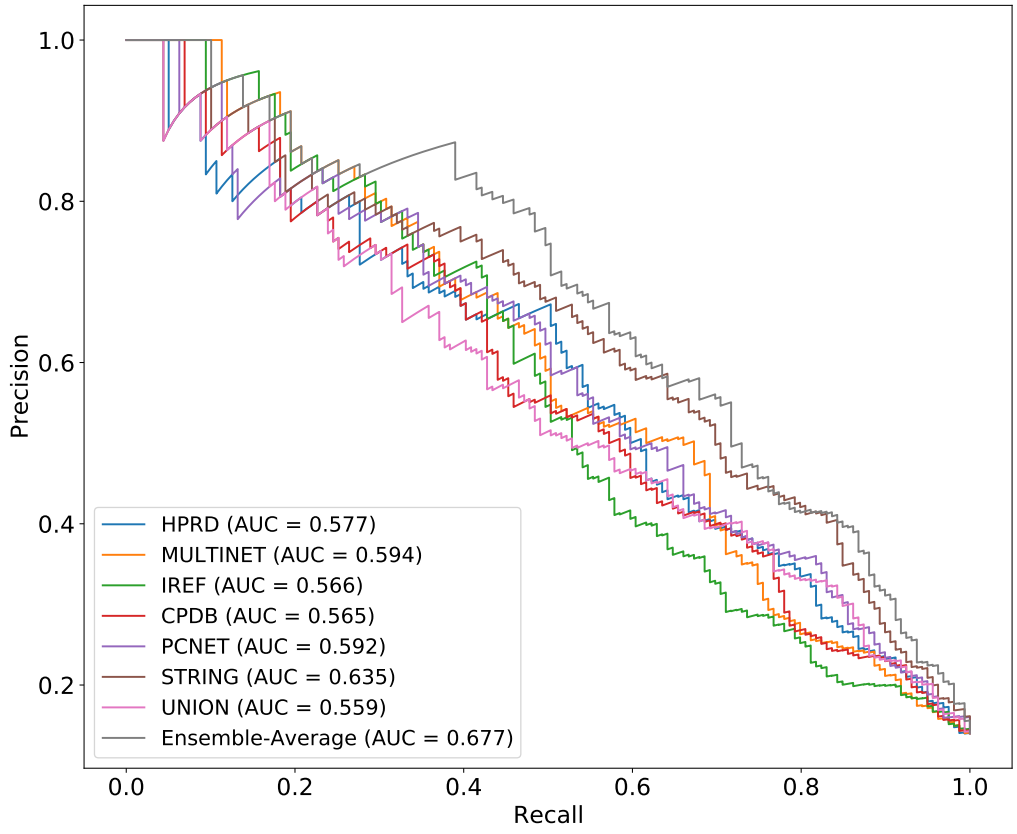
Finally, we also plot the precision-recall curves (Figure 6.11) for all proposed GCN models except the Ensemble-Majority one. This model was not included in the analysis because it does not provide predicted class probabilities needed to analyze the curve. This analysis corroborates the fact that the Ensemble-Average GCN model tends to perform better than the other models in this classification task, as it presented the highest AUC-PR score (AUC-PR = 0.677).

6.2.5 Analysis of GCN-based predictions for unlabelled nodes

The main goal of our work is to develop and evaluate a GCN-based model for the prediction of CDGs. However, we are also interested in identifying potential driver genes among the unlabeled genes of the PPI networks. This could be interesting since it would enable pointing out candidate driver genes for further experimental investigation. In this section, we conduct and discuss this analysis.

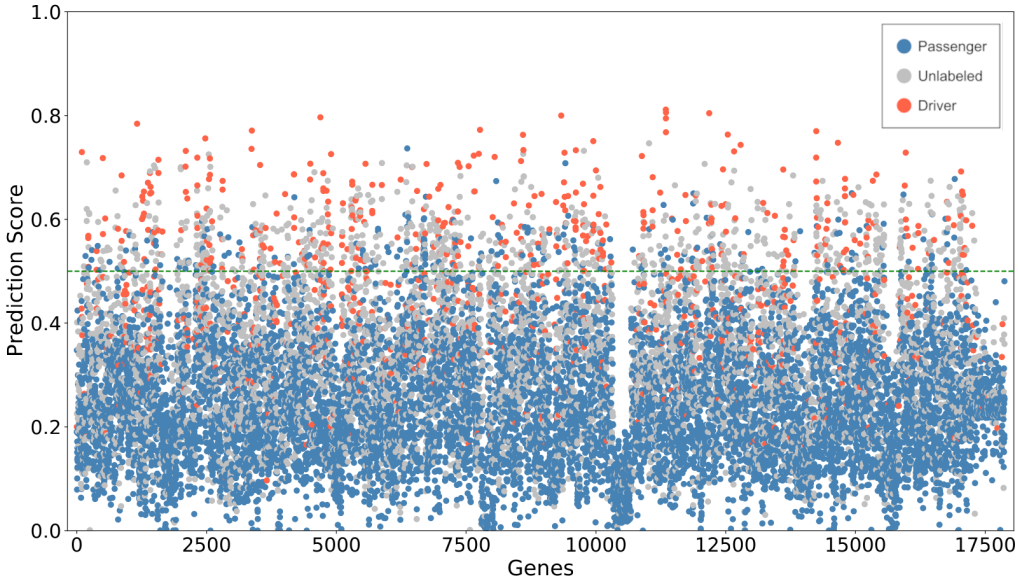
The GCN models were applied for the classification of the unlabeled nodes of their underlying PPI network (or the multiple networks, in case of the ensemble approaches). As an example, Figure 6.12 displays a scatterplot for the predicted probabilities of all nodes in the STRING PPI network. Red dots represent true driver genes, blue dots represent true passenger genes, and gray dots represent unlabeled genes. Using a probability threshold of 0.5 represented by the green dashed line, several of the unlabeled genes are

Figure 6.11 – Precision-Recall curves for GCN models predictions for the test set.



classified as cancer driver genes by our GCN model based on the STRING PPI network. The same visualization is provided for the other PPI networks in the Appendix C, Figure C.7 and Figure C.8.

Figure 6.12 – Predicted probabilities for all genes in the STRING PPI network using the trained GCN model.



By examining the prediction for the complete set of nodes comprised by a PPI net-

work, we can assess the behavior of unlabeled genes that received the same predictions in different models. Table 6.7 presents in the first column a subset of 10 genes that were unanimously predicted as cancer driver genes by the six GCN models trained in the PPI networks collected from the literature. The complete list of the 33 candidate cancer driver genes predicted by the six individual GCN-based models is provided in the Appendix C (Table C.4); however, we choose to show in the Table the 10 genes with the highest predicted probabilities for the *driver* class across individual models. In addition, 1477 genes were unanimously predicted as passengers by the six individual models, from which we select a sample of 33 to present in Table C.4, also prioritizing those with higher predicted probabilities.

Table 6.7 – Top ten candidate driver genes with the highest predicted probabilities according to some selected approaches.

Predicted by models for the six PPI networks	Predicted by model for the UNION network	Predicted by the Ensemble-Average model
HDAC2	ALB	GRB2
SP1	APP	HDAC2
PTK2	ELAVL1	HDAC3
HDAC3	ETS1	KAT2B
UBE2I	FOS	NR3C1
NR3C1	GAPDH	PRKCA
CDK1	NFKB1	PTK2
KAT2B	**SP1**	RELA
NCK1	TBP	SHC1
PRKCA	TCF4	**SP1**

Table 6.7 also includes the top ten genes according to the probabilities for the *driver* class predicted by the UNION-based GCN model and the Ensemble-Average GCN model. Six genes are present in both the unanimous predictions among individual models (first column) and in top 10 predictions for the Ensemble-Average model (third column). Only one gene, SP1 (Sp1 Transcription Factor), is predicted as candidate in all three cases analyzed in the table. Further investigation of these genes may elucidate new drivers of tumorigenesis.

Another analysis we carried out was to investigate false positives indicated with high probabilities as CDG by the proposed models. We include similarobservations as in Table 6.7, but now considering false positive genes instead of candidates genes (Table 6.8). The first column represents genes that are originally labeled as negative genes and that were predicted as positive by five or more different networks, considering the highest probability in all networks. Only three genes were unanimously predicted as false

positives for this case YWHAZ, SPTBN1 and H2BC21. The second and third column represent the original negative genes predicted as CDGs with the highest probability considering the UNION network model and the Ensemble-Average method, respectively.

Table 6.8 – Top ten false positives driver genes with the highest predicted probabilities according to some selected approaches.

False positive predicted by separated models of PPI networks	False positives predicted by model for the UNION network	False positives predicted by the Ensemble-Average model
SUMO2	MYH14	**YWHAZ**
H2BC21	VCAM1	YWHAG
SUMO3	PI3	SFN
YWHAQ	TF	YWHAH
SPTBN1	UTRN	SRF
YWHAG	MAFF	IRAK1
PTPRF	SST	MBP
SRF	**YWHAZ**	PIN1
KHDRBS1	SMARCA5	SMARCA5
YWHAZ	MBP	PPFIA1

Observing the high-confidence false positives predictions of our models, we may extract insights about genes that have strong indication of involvement with cancer, perhaps as a CDG, but that were not experimentally verified as drivers of carcinogenesis. This is the case for gene YWHAZ, which was unanimously predicted as a driver by all analyzed networks, as well as by the UNION network and the Ensemble-Average model. Interestingly, prior works have discussed that YWHAZ is frequently up-regulated in multiple types of cancers and may act as an oncogene by promoting malignant properties of cancer cells (GAN; YE; HE, 2020). Thus, further investigation of false positives associated with high CDG probability is interesting to review original annotations and expand the current knowledge about CDGs.

6.3 Summary

With the best GNN algorithm selected in the previous chapter, in this chapter we seek to optimize the prediction results of the GCN model, we perform a search for better combinations of hyperparameters in each of the selected networks, to meet the different particularities of each one of them.

With this, we identified a set of genes common to all networks that could be used as test data. We then apply a new training round to each of the networks, with the optimal parameters found and with a common test set, from this we can analyze the prediction tendency of each of the networks separately by looking at this fixed list that is evaluated

in the same way for all networks.

We implemented a majority voting system and utilized the average prediction scores of the networks to identify genes that were highly likely to be classified as driver genes. In addition, we devised a new network architecture by combining the nodes and edges of all six networks and removing duplicate gene links. This new network was also subjected to prediction analysis using the same ensemble, but was found to be less effective in terms of metrics than the initial ensemble network analysis.

Despite this, we successfully identified a list of candidate genes that were not previously classified as drivers or passengers but were classified as drivers in different contexts. We also identified a list of 33 candidate genes that were unanimously predicted by all selected networks to be cancer driver genes.

7 CONCLUSION AND FUTURE WORK

Identifying cancer driver genes (CDGs) accurately remains a critical challenge in cancer research. These genes play a crucial role in cancer development and progression. Despite extensive work in the field, predicting CDGs remains complex due to the heterogeneity of underlying biological mechanisms. In recent years, the use of graph-based learning methods, particularly Graph Neural Networks (GNNs), has shown great potential to improve the accuracy of CDG prediction. Schulte-Sasse et al. (2021), for example, produced an approach capable of integrating pan-cancer data into a GCN-based model to identify new drivers of cancer. Our study expanded on this approach by evaluating and comparing different methodological strategies to predict CDGs using three different GNN algorithms. We focused on adding useful information to describe genes and dealing with class imbalance. Centralities added relevant information to node features vector, increasing performance in graph-based learning methods as well as for traditional ML algorithms. We also observed that graph-based learning algorithms generally outperformed other approaches in the prediction task.

Among the applied GNNs, GCN demonstrated the most promising results, showing consistent and robust performance in all evaluated scenarios. We carried out an extensive search for hyperparameter optimization to optimize prediction results in each of the six PPI networks selected for the study. Moreover, we successfully implemented two ensemble learning strategies, a majority voting system, and a system using the average predicted clas probabilities among models networks. The overall performance of these models in predicting test set genes showed a clear impact of the two ensemble approaches in reducing the number of false positives. We compared these results with a comprehesinve PPI network defined in this work, which combines the nodes and edges of all six selected PPI networks and removes duplicate gene links. The ensemble models outperformed the other approaches used for prediction, surpassing both the individual PPI networks and the unified PPI network. Our best approach, represented by the Ensemble-Average GCN-based model, reached an AUC-PR score of 0.67 in an independent test set of drivers and passenger genes. Although this result is promising, our model still failed to improve over the performance of previously proposed approaches (SCHULTE-SASSE et al., 2021).

Further research in this area could lead to a better understanding of the molecular mechanisms underlying cancer and pave the way for more precise models that, in

turn, will help in the development of more effective diagnostic and therapeutic strategies. Areas for growth and development include new approaches for handling class imbalance in graph-based learning, such as GraphSMOTE (ZHAO; ZHANG; WANG, 2021), and multi-task learning on GNNs to be able to simultaneously train models capable of predicting cancer-specific and pan-cancer driver genes. Understanding driver mutation profiles of different cancer types is crucial, as driver mutations can differ between cancer types, and individual tumors have their own mutation profiles. Finally, improving the strategies to create labeled data for positive and negative examples of CDGs can have a great impact on the overall performance of prediction model.

REFERENCES

ADZHUBEI, I. A. et al. A method and server for predicting damaging missense mutations. **Nature Methods**, Nature Publishing Group, v. 7, n. 4, p. 248–249, 2010.

AGAJANIAN, S. et al. Machine Learning Classification and Structure-Functional Analysis of Cancer Mutations Reveal Unique Dynamic and Network Signatures of Driver Sites in Oncogenes and Tumor Suppressor Genes. **Journal of Chemical Information and Modeling**, v. 58, n. 10, p. 2131–2150, 2018. ISSN 15205142.

AGAJANIAN, S.; OLUYEMI, O.; VERKHIVKER, G. M. Integration of random forest classifiers and deep convolutional neural networks for classification and biomolecular modeling of cancer driver mutations. **Frontiers in Molecular Biosciences**, v. 6, n. JUN, 2019. ISSN 2296889X.

ALBERTS, B. et al. **Molecular Biology of the Cell**. 4th. ed. New York: Garland Science, 2002.

ALTHUBAITI, S. et al. Ontology-based prediction of cancer driver genes. **Scientific Reports**, v. 9, n. 1, p. 1–9, 2019. ISSN 20452322.

ANANDAKRISHNAN, R. et al. Estimating the number of genetic mutations (hits) required for carcinogenesis based on the distribution of somatic mutations. **PLoS Computational Biology**, Public Library of Science, v. 15, n. 3, p. e1006881, 2019.

ANDRADES, R.; RECAMONDE-MENDOZA, M. Machine learning methods for prediction of cancer driver genes: a survey paper. **Briefings in Bioinformatics**, v. 23, n. 3, 03 2022. ISSN 1477-4054. Bbac062. Available from Internet: <https://doi.org/10.1093/bib/bbac062>.

ANOOSHA, P. et al. Discrimination of driver and passenger mutations in epidermal growth factor receptor in cancer. **Mutation Research - Fundamental and Molecular Mechanisms of Mutagenesis**, Elsevier, v. 780, p. 24–34, 2015. ISSN 18792871. Available from Internet: <http://dx.doi.org/10.1016/j.mrfmmm.2015.07.005>.

ATA, S. K. et al. Recent advances in network-based methods for disease gene prediction. **Briefings in Bioinformatics**, Oxford University Press, v. 22, n. 4, p. bbaa303, 2021.

BAILEY, M. H. et al. Comprehensive characterization of cancer driver genes and mutations. **Cell**, Elsevier, v. 173, n. 2, p. 371–385, 2018.

BARABÁSI, A.-L.; GULBAHCE, N.; LOSCALZO, J. Network medicine: a network-based approach to human disease. **Nature Reviews Genetics**, Nature Publishing Group, v. 12, n. 1, p. 56–68, 2011.

BARRETINA, J. et al. The cancer cell line encyclopedia enables predictive modelling of anticancer drug sensitivity. **Nature**, Nature Publishing Group, v. 483, n. 7391, p. 603–607, 2012.

BELKADI, A. et al. Whole-genome sequencing is more powerful than whole-exome sequencing for detecting exome variants. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 112, n. 17, p. 5473–5478, 2015.

BELL, R. J. et al. Understanding TERT promoter mutations: a common path to immortality. **Molecular Cancer Research**, AACR, v. 14, n. 4, p. 315–323, 2016.

BEROUKHIM, R. et al. Assessing the significance of chromosomal aberrations in cancer: methodology and application to glioma. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 104, n. 50, p. 20007–20012, 2007.

BRAY, F. et al. Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. **CA: A Cancer Journal for Clinicians**, Wiley Online Library, v. 68, n. 6, p. 394–424, 2018.

BRAZMA, A.; VILO, J. Gene expression data analysis. **FEBS letters**, Elsevier, v. 480, n. 1, p. 17–24, 2000.

BREIMAN, L. Random forests. **Machine learning**, Springer, v. 45, p. 5–32, 2001.

BROWN, A.-L. et al. Finding driver mutations in cancer: Elucidating the role of background mutational processes. **PLoS Computational Biology**, Public Library of Science San Francisco, CA USA, v. 15, n. 4, p. e1006981, 2019.

CAMACHO, D. M. et al. Next-Generation Machine Learning for Biological Networks. **Cell**, Elsevier Inc., v. 173, n. 7, p. 1581–1592, 2018. ISSN 10974172. Available from Internet: <https://doi.org/10.1016/j.cell.2018.05.015>.

CAPRIOTTI, E.; ALTMAN, R. B. A new disease-specific machine learning approach for the prediction of cancer-causing missense variants. **Genomics**, v. 98, n. 4, p. 310–317, oct 2011. ISSN 08887543. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0888754311001601>.

CARTER, H. et al. Cancer-specific high-throughput annotation of somatic mutations: Computational prediction of driver missense mutations. **Cancer Research**, v. 69, n. 16, p. 6660–6667, 2009. ISSN 00085472.

CARTER, H. et al. Identifying mendelian disease genes with the variant effect scoring tool. **BMC Genomics**, BioMed Central, v. 14, n. 3, p. 1–16, 2013.

CELLI, F.; CUMBO, F.; WEITSCHEK, E. Classification of Large DNA Methylation Datasets for Identifying Cancer Drivers. **Big Data Research**, Elsevier Inc., v. 13, p. 21–28, 2018. ISSN 22145796. Available from Internet: <https://doi.org/10.1016/j.bdr.2018.02.005>.

CERAMI, E. et al. **The cBio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data**. [S.l.]: AACR, 2012. 401–404 p.

CHAKRAVARTY, D. et al. OncoKB: a precision oncology knowledge base. **JCO Precision Oncology**, American Society of Clinical Oncology, v. 1, p. 1–16, 2017.

CHANDRASHEKAR, P. et al. Somatic selection distinguishes oncogenes and tumor suppressor genes. **Bioinformatics**, v. 36, n. 6, p. 1712–1717, 2020. ISSN 14602059.

CHEN, H. et al. Comprehensive assessment of computational algorithms in predicting cancer driver mutations. **Genome Biology**, BioMed Central, v. 21, n. 1, p. 1–17, 2020.

CHEN, J.; SUN, M.; SHEN, B. Deciphering oncogenic drivers: from single genes to integrated pathways. **Briefings in Bioinformatics**, Oxford University Press, v. 16, n. 3, p. 413–428, 2015.

CHEN, M. et al. Simple and deep graph convolutional networks. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2020. p. 1725–1735.

CHEN, Y.-C. et al. Significant associations between driver gene mutations and dna methylation alterations across many cancer types. **PLoS Computational Biology**, Public Library of Science San Francisco, CA USA, v. 13, n. 11, p. e1005840, 2017.

CHENG, F.; ZHAO, J.; ZHAO, Z. Advances in computational approaches for prioritizing driver mutations and significantly mutated genes in cancer genomes. **Briefings in Bioinformatics**, Oxford University Press, v. 17, n. 4, p. 642–656, 2016.

CHO, A. et al. MUFFINN: cancer gene discovery via network analysis of somatic mutation data. **Genome Biology**, BioMed Central, v. 17, n. 1, p. 1–16, 2016.

CHOI, Y.; CHAN, A. P. PROVEAN web server: a tool to predict the functional effect of amino acid substitutions and indels. **Bioinformatics**, Oxford University Press, v. 31, n. 16, p. 2745–2747, 2015.

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

CHOU, C.-H. et al. miRTarBase update 2018: a resource for experimentally validated microRNA–target interactions. **Nucleic Acids Research**, Oxford University Press, v. 46, n. D1, p. D296–D302, 2018.

CHUN, S.; FAY, J. C. Identification of deleterious mutations within three human genomes. **Genome Research**, Cold Spring Harbor Lab, v. 19, n. 9, p. 1553–1561, 2009.

CHUNG, I.-F. et al. DriverDBv2: a database for human cancer driver gene research. **Nucleic Acids Research**, Oxford University Press, v. 44, n. D1, p. D975–D979, 2016.

COLAPRICO, A. et al. Interpreting pathways to discover cancer driver genes with Moonlight. **Nature Communications**, Springer US, v. 11, n. 1, 2020. ISSN 20411723. Available from Internet: <http://dx.doi.org/10.1038/s41467-019-13803-0>.

COLLIER, O.; STOVEN, V.; VERT, J. P. LOTUS: A single- And multitask machine learning algorithm for the prediction of cancer driver genes. **PLoS Computational Biology**, v. 15, n. 9, p. 1–27, 2019. ISSN 15537358.

CRICK, F. Central dogma of molecular biology. **Nature**, Nature Publishing Group, v. 227, n. 5258, p. 561–563, 1970.

CSARDI, G.; NEPUSZ, T. The igraph software package for complex network research. **InterJournal**, Complex Systems, p. 1695, 2006. Available from Internet: <https://igraph.org>.

CUTIGI, J. F. et al. Combining Mutation and Gene Network Data in a Machine Learning Approach for False-Positive Cancer Driver Gene Discovery. In: SETUBAL, J. C.; SILVA, W. M. (Ed.). **Advances in Bioinformatics and Computational Biology. BSB 2020. Lecture Notes in Computer Science**. [S.l.: s.n.], 2020. v. 12558, p. 81–92. ISBN 9783030657741. ISSN 16113349.

DAS, J.; YU, H. HINT: High-quality protein interactomes and their applications in understanding human disease. **BMC Systems Biology**, BioMed Central, v. 6, n. 1, p. 1–12, 2012.

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

DAVOLI, T. et al. Cumulative Haploinsufficiency and Triplosensitivity Drive Aneuploidy Patterns and Shape the Cancer Genome. **Cell**, v. 155, n. 4, p. 948–962, nov 2013. ISSN 00928674. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0092867413012877>.

DAVYDOV, E. V. et al. Identifying a high fraction of the human genome to be under selective constraint using gerp++. **PLoS Computational Biology**, Public Library of Science San Francisco, USA, v. 6, n. 12, p. e1001025, 2010.

DIMITRAKOPOULOS, C. M.; BEERENWINKEL, N. Computational approaches for the identification of cancer genes and pathways. **Wiley Interdisciplinary Reviews: Systems Biology and Medicine**, Wiley Online Library, v. 9, n. 1, p. e1364, 2017.

DONG, C. et al. iCAGES: Integrated CAncer GEnome Score for comprehensively prioritizing driver genes in personal cancer genomes. **Genome Medicine**, Genome Medicine, v. 8, n. 1, p. 1–22, 2016. ISSN 1756994X. Available from Internet: <http://dx.doi.org/10.1186/s13073-016-0390-0>.

DONG, C. et al. Comparison and integration of deleteriousness prediction methods for nonsynonymous snvs in whole exome sequencing studies. **Human molecular genetics**, Oxford University Press, v. 24, n. 8, p. 2125–2137, 2015.

DONG, X. et al. A survey on ensemble learning. **Frontiers of Computer Science**, Springer, v. 14, p. 241–258, 2020.

ELLIOTT, K.; LARSSON, E. Non-coding driver mutations in human cancer. **Nature Reviews Cancer**, Nature Publishing Group, p. 1–10, 2021.

ERASLAN, G. et al. Deep learning: new computational modelling techniques for genomics. **Nature Reviews Genetics**, Nature Publishing Group UK London, v. 20, n. 7, p. 389–403, 2019.

FERLAY, J. et al. **Global Cancer Observatory: Cancer Today**. 2020. <https://gco.iarc.fr/today>. Accessed April 2021.

FU, C. et al. Identification of oncogenic genes for colon adenocarcinoma from genomics data. In: **2012 IEEE 6th International Conference on Systems Biology (ISB)**. [S.l.: s.n.], 2012. p. 263–266. ISBN 9781467343985.

FU, Y. et al. Funseq2: a framework for prioritizing noncoding regulatory variants in cancer. **Genome Biology**, BioMed Central, v. 15, n. 10, p. 1–15, 2014.

GAN, Y.; YE, F.; HE, X.-X. The role of ywhaz in cancer: A maze of opportunities and challenges. **Journal of Cancer**, Ivyspring International Publisher, v. 11, n. 8, p. 2252, 2020.

GARBER, M. et al. Identifying novel constrained elements by exploiting biased substitution patterns. **Bioinformatics**, Oxford University Press, v. 25, n. 12, p. i54–i62, 2009.

GEORGOUSIS, S.; KENNING, M. P.; XIE, X. Graph deep learning: State of the art and challenges. **IEEE Access**, IEEE, v. 9, p. 22106–22140, 2021.

GIBBS, R. et al. The international hapmap project. **Nature**, Nature Publishing Group, v. 426, n. 6968, 2003.

GNAD, F. et al. Bioinformatics analysis of thousands of TCGA tumors to determine the involvement of epigenetic regulators in human cancer. **BMC Genomics**, v. 16, n. S8, p. S5, dec 2015. ISSN 1471-2164. Available from Internet: <https://bmcgenomics.biomedcentral.com/articles/10.1186/1471-2164-16-S8-S5>.

GONZÁLEZ-PÉREZ, A.; LÓPEZ-BIGAS, N. Improving the assessment of the outcome of nonsynonymous snvs with a consensus deleteriousness score, condel. **The American Journal of Human Genetics**, Elsevier, v. 88, n. 4, p. 440–449, 2011.

GONZALEZ-PEREZ, A.; LOPEZ-BIGAS, N. Functional impact bias reveals cancer drivers. **Nucleic Acids Research**, Oxford University Press, v. 40, n. 21, p. e169–e169, 2012.

GONZALEZ-PEREZ, A. et al. IntOGen-mutations identifies cancer drivers across tumor types. **Nature Methods**, Nature Publishing Group, v. 10, n. 11, p. 1081–1082, 2013.

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

GRANTHAM, R. Amino acid difference formula to help explain protein evolution. **Science**, American Association for the Advancement of Science, v. 185, n. 4154, p. 862–864, 1974.

GUAN, Y. et al. Prioritizing predictive biomarkers for gene essentiality in cancer cells with mRNA expression data and DNA copy number profile. **Bioinformatics**, v. 34, n. 23, p. 3975–3982, 2018. ISSN 13674811.

GULKO, B. et al. A method for calculating probabilities of fitness consequences for point mutations across the human genome. **Nature Genetics**, Nature Publishing Group, v. 47, n. 3, p. 276–283, 2015.

GUMPINGER, A. C. et al. Prediction of cancer driver genes through network-based moment propagation of mutation scores. **Bioinformatics**, v. 36, n. 1, p. i508–i515, 2020. ISSN 13674811.

GUO, Y. A.; CHANG, M. M.; SKANDERUP, A. J. Mutspot: detection of non-coding mutation hotspots in cancer genomes. **NPJ Genomic Medicine**, Nature Publishing Group, v. 5, n. 1, p. 1–5, 2020.

HALL, M. A.; FRANK, E. Combining naive bayes and decision tables. In: **FLAIRS conference**. [S.l.: s.n.], 2008. v. 2118, p. 318–319.

HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive representation learning on large graphs. **Advances in Neural Information Processing Systems**, v. 30, 2017.

HAMOSH, A. et al. Online mendelian inheritance in man (omim). **Human Mutation**, Wiley Online Library, v. 15, n. 1, p. 57–61, 2000.

HAMOSH, A. et al. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. **Nucleic Acids Research**, Oxford University Press, v. 33, n. suppl_1, p. D514–D517, 2005.

HAN, A. et al. SNP@Domain: a web resource of single nucleotide polymorphisms (SNPs) within protein domain structures and sequences. **Nucleic Acids Research**, Oxford University Press, v. 34, n. suppl_2, p. W642–W644, 2006.

HAN, Y. et al. DriverML: A machine learning algorithm for identifying driver genes in cancer sequencing studies. **Nucleic Acids Research**, Oxford University Press, v. 47, n. 8, 2019. ISSN 13624962.

HANAHAN, D. Hallmarks of cancer: new dimensions. **Cancer Discovery**, AACR, v. 12, n. 1, p. 31–46, 2022.

HE, H.; GARCIA, E. A. Learning from imbalanced data. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 21, n. 9, p. 1263–1284, 2009.

HIRANO, S. et al. Pan-cancer analysis of whole genomes. **Nature**, W190 Hochdurchsatz-Sequenzierung, v. 578, n. DKFZ-2020-01051, p. 82–93, 2020.

HUA, X. et al. DrGaP: a powerful tool for identifying driver genes and pathways in cancer sequencing studies. **The American Journal of Human Genetics**, Elsevier, v. 93, n. 3, p. 439–451, 2013.

HUANG, J. K. et al. Systematic evaluation of molecular networks for discovery of disease genes. **Cell Systems**, Elsevier, v. 6, n. 4, p. 484–495, 2018.

HUANG, S. et al. Applications of support vector machine (SVM) learning in cancer genomics. **Cancer Genomics & Proteomics**, International Institute of Anticancer Research, v. 15, n. 1, p. 41–51, 2018.

IOANNIDIS, N. M. et al. REVEL: an ensemble method for predicting the pathogenicity of rare missense variants. **The American Journal of Human Genetics**, Elsevier, v. 99, n. 4, p. 877–885, 2016.

IONITA-LAZA, I. et al. A spectral approach integrating functional genomic annotations for coding and noncoding variants. **Nature Genetics**, Nature Publishing Group, v. 48, n. 2, p. 214–220, 2016.

JACKSON, M. et al. The genetic basis of disease. **Essays in Biochemistry**, v. 62, n. 5, p. 643–723, 12 2018. ISSN 0071-1365. Available from Internet: <https://doi.org/10.1042/EBC20170053>.

JAGADEESH, K. A. et al. M-CAP eliminates a majority of variants of uncertain significance in clinical exomes at high sensitivity. **Nature Genetics**, Nature Publishing Group, v. 48, n. 12, p. 1581–1586, 2016.

JASSAL, B. et al. The reactome pathway knowledgebase. **Nucleic Acids Research**, Oxford University Press, v. 48, n. D1, p. D498–D503, 2020.

JIANG, L. et al. WITER: a powerful method for estimation of cancer-driver genes using a weighted iterative regression modelling background mutation counts. **Nucleic Acids Research**, v. 47, n. 16, p. e96, 2019. ISSN 13624962.

JIANG, R. et al. Sequence-based prioritization of nonsynonymous single-nucleotide polymorphisms for the study of disease mutations. **The American Journal of Human Genetics**, Elsevier, v. 81, n. 2, p. 346–360, 2007.

KAMBUROV, A. et al. ConsensusPathDB: toward a more complete picture of cell biology. **Nucleic Acids Research**, Oxford University Press, v. 39, n. suppl_1, p. D712–D717, 2011.

KARCZEWSKI, K. J. et al. The mutational constraint spectrum quantified from variation in 141,456 humans. **Nature**, Nature Publishing Group, v. 581, n. 7809, p. 434–443, 2020.

KARCZEWSKI, K. J.; SNYDER, M. P. Integrative omics for health and disease. **Nature Reviews Genetics**, Nature Publishing Group UK London, v. 19, n. 5, p. 299–310, 2018.

KHAN, A.; ZHANG, X. dbSUPER: a database of super-enhancers in mouse and human genome. **Nucleic Acids Research**, Oxford University Press, v. 44, n. D1, p. D164–D171, 2016.

KINGSFORD, C.; SALZBERG, S. L. What are decision trees? **Nature Biotechnology**, Nature Publishing Group US New York, v. 26, n. 9, p. 1011–1013, 2008.

KIPF, T. N.; WELLING, M. **Semi-Supervised Classification with Graph Convolutional Networks**. arXiv, 2016. Available from Internet: <https://arxiv.org/abs/1609.02907>.

KIRCHER, M. et al. A general framework for estimating the relative pathogenicity of human genetic variants. **Nature Genetics**, Nature Publishing Group, v. 46, n. 3, p. 310–315, 2014.

KOH, G. C. et al. Analyzing protein–protein interaction networks. **Journal of Proteome Research**, ACS Publications, v. 11, n. 4, p. 2014–2031, 2012.

KOTSIANTIS, S. B. Decision trees: a recent overview. **Artificial Intelligence Review**, Springer, v. 39, p. 261–283, 2013.

KOTSIANTIS, S. B.; ZAHARAKIS, I. D.; PINTELAS, P. E. Machine learning: A review of classification and combining techniques. **Artificial Intelligence Review**, v. 26, n. 3, p. 159–190, 2006. ISSN 02692821.

KOUTSOUKAS, A. et al. Deep-learning: investigating deep neural networks hyperparameters and comparison of performance to shallow methods for modeling bioactivity data. **Journal of Cheminformatics**, BioMed Central, v. 9, n. 1, p. 1–13, 2017.

KROGH, A. What are artificial neural networks? **Nature Biotechnology**, Nature Publishing Group US New York, v. 26, n. 2, p. 195–197, 2008.

KULIS, M.; ESTELLER, M. DNA methylation and cancer. **Advances in Genetics**, Elsevier, v. 70, p. 27–56, 2010.

KUNCHEVA, L. I.; RODRÍGUEZ, J. J. On feature selection protocols for very low-sample-size data. **Pattern Recognition**, Elsevier, v. 81, p. 660–673, 2018.

LAGE, K. et al. A human phenome-interactome network of protein complexes implicated in genetic disorders. **Nature Biotechnology**, Nature Publishing Group, v. 25, n. 3, p. 309–316, 2007.

LANDRUM, M. J. et al. ClinVar: improving access to variant interpretations and supporting evidence. **Nucleic Acids Research**, Oxford University Press, v. 46, n. D1, p. D1062–D1067, 2018.

LAPUSCHKIN, S. et al. Unmasking clever hans predictors and assessing what machines really learn. **Nature Communications**, Nature Publishing Group, v. 10, n. 1, p. 1–8, 2019.

LAWRENCE, M. S. et al. Mutational heterogeneity in cancer and the search for new cancer-associated genes. **Nature**, Nature Publishing Group, v. 499, n. 7457, p. 214–218, 2013.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015.

LEEVY, J. L. et al. A survey on addressing high-class imbalance in big data. **Journal of Big Data**, Springer, v. 5, n. 1, p. 1–30, 2018.

LEHMANN, K.-V.; CHEN, T. Exploring functional variant discovery in non-coding regions with SInBaD. **Nucleic Acids Research**, Oxford University Press, v. 41, n. 1, p. e7–e7, 2013.

LEISERSON, M. D. et al. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. **Nature Genetics**, Nature Publishing Group US New York, v. 47, n. 2, p. 106–114, 2015.

LI, C. H. et al. Sex differences in cancer driver genes and biomarkers. **Cancer Research**, AACR, v. 78, n. 19, p. 5527–5537, 2018.

LI, H. T. et al. Identification of driver pathways in cancer based on combinatorial patterns of somatic gene mutations. **Neoplasma**, v. 63, n. 01, p. 57–63, 2016. ISSN 13384317. Available from Internet: <http://www.elis.sk/index.php?page=shop.product_details&flypage=flypage.tpl&product_id=4460&category_id=128&option=com_virtuemart>.

LI, J. et al. Feature selection: A data perspective. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 50, n. 6, p. 1–45, 2017.

LI, Y.; CHEN, L. Big biological data: challenges and opportunities. **Genomics, Proteomics & Bioinformatics**, Elsevier, v. 12, n. 5, p. 187, 2014.

LI, Z. et al. A survey of convolutional neural networks: analysis, applications, and prospects. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, 2021.

LIBBRECHT, M. W.; NOBLE, W. S. Machine learning applications in genetics and genomics. **Nature Reviews Genetics**, Nature Publishing Group, v. 16, n. 6, p. 321–332, 2015.

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, C. et al. Computational network biology: data, models, and applications. **Physics Reports**, Elsevier, v. 846, p. 1–66, 2020.

LU, H. et al. Recent advances in the development of protein–protein interactions modulators: mechanisms and clinical trials. **Signal Transduction and Targeted Therapy**, Nature Publishing Group, v. 5, n. 1, p. 1–23, 2020.

LU, Q. et al. A statistical framework to predict functional non-coding regions in the human genome through integrated analysis of annotation data. **Scientific Reports**, Nature Publishing Group, v. 5, n. 1, p. 1–13, 2015.

LU, X. et al. The integrative method based on the module-network for identifying driver genes in cancer subtypes. **Molecules**, v. 23, n. 2, p. 1–15, 2018. ISSN 14203049.

LUCK, K. et al. A reference map of the human binary protein interactome. **Nature**, Nature Publishing Group, v. 580, n. 7803, p. 402–408, 2020.

LUO, P. et al. DeepDriver: Predicting cancer driver genes based on somatic mutations using deep convolutional neural networks. **Frontiers in Genetics**, v. 10, n. JAN, p. 1–12, 2019. ISSN 16648021.

LYU, J. et al. DORGE: Discovery of Oncogenes and tumoR suppressor genes using Genetic and Epigenetic features. **Science Advances**, v. 6, n. 46, p. eaba6784, nov 2020. ISSN 2375-2548. Available from Internet: <https://advances.sciencemag.org/lookup/doi/10.1126/sciadv.aba6784>.

MAMMONE, A.; TURCHI, M.; CRISTIANINI, N. Support vector machines. **Wiley Interdisciplinary Reviews: Computational Statistics**, Wiley Online Library, v. 1, n. 3, p. 283–289, 2009.

MANOLAKOS, A. et al. CaMoDi: A new method for cancer module discovery. **BMC Genomics**, v. 15, n. Suppl 10, p. 1–10, 2014. ISSN 14712164.

MAO, Y. et al. CanDrA: Cancer-specific driver missense mutation annotation with optimized features. **PLoS ONE**, v. 8, n. 10, 2013. ISSN 19326203.

MARTELOTTO, L. G. et al. Benchmarking mutation effect prediction algorithms using functionally validated cancer-related missense mutations. **Genome Biology**, BioMed Central, v. 15, n. 10, p. 1–20, 2014.

MARTÍNEZ-JIMÉNEZ, F. et al. A compendium of mutational cancer driver genes. **Nature Reviews Cancer**, Nature Publishing Group, v. 20, n. 10, p. 555–572, 2020.

MERMEL, C. H. et al. Gistic2. 0 facilitates sensitive and confident localization of the targets of focal somatic copy-number alteration in human cancers. **Genome Biology**, Springer, v. 12, n. 4, p. 1–14, 2011.

MIN, S.; LEE, B.; YOON, S. Deep learning in bioinformatics. **Briefings in Bioinformatics**, Oxford University Press, v. 18, n. 5, p. 851–869, 2017.

MOLNAR, C. et al. General pitfalls of model-agnostic interpretation methods for machine learning models. **arXiv preprint arXiv:2007.04131**, 2020.

MOORE, L. D.; LE, T.; FAN, G. DNA methylation and its basic function. **Neuropsychopharmacology**, Nature Publishing Group, v. 38, n. 1, p. 23–38, 2013.

MWENIFUMBO, J. C.; MARRA, M. A. Cancer genome-sequencing study design. **Nature Reviews Genetics**, Nature Publishing Group, v. 14, n. 5, p. 321–332, 2013.

NAQA, I. E.; MURPHY, M. J. What is machine learning? In: **Machine Learning in Radiation Oncology**. [S.l.]: Springer, 2015. p. 3–11.

NG, P. C.; HENIKOFF, S. SIFT: Predicting amino acid changes that affect protein function. **Nucleic Acids Research**, Oxford University Press, v. 31, n. 13, p. 3812–3814, 2003.

NG, P. K.-S. et al. Systematic functional annotation of somatic mutations in cancer. **Cancer Cell**, Elsevier, v. 33, n. 3, p. 450–462, 2018.

NICORA, G. et al. A semi-supervised learning approach for pan-cancer somatic genomic variant classification. In: RIAÑO, D.; WILK, S.; TEIJE, A. ten (Ed.). **Artificial Intelligence in Medicine (AIME 2019). Lecture Notes in Computer Science**. [S.l.: s.n.], 2019. v. 11526, p. 42–46. ISBN 9783030216412. ISSN 16113349.

NULSEN, J. et al. Pan-cancer detection of driver genes at the single-patient resolution. **Genome Medicine**, Genome Medicine, v. 13, n. 1, p. 1–14, 2021. ISSN 1756994X.

ORCHARD, S. et al. The mintact project—intact as a common curation platform for 11 molecular interaction databases. **Nucleic Acids Research**, Oxford University Press, v. 42, n. D1, p. D358–D363, 2014.

PARK, H. et al. Interaction-Based Feature Selection for Uncovering Cancer Driver Genes Through Copy Number-Driven Expression Level. **Journal of Computational Biology**, v. 24, n. 2, p. 138–152, 2017. ISSN 10665277.

PARK, H. et al. Sparse overlapping group lasso for integrative multi-omics analysis. **Journal of Computational Biology**, v. 22, n. 2, p. 73–84, 2015. ISSN 10665277.

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

PENG, W. et al. Improving cancer driver gene identification using multi-task learning on graph convolutional network. **Briefings in Bioinformatics**, Oxford University Press, p. bbab432, 2021.

PHAM, V. V. H. et al. Computational methods for cancer driver discovery: A survey. **Theranostics**, Ivyspring International Publisher, v. 11, p. 5553–5568, 2021.

PILLICH, R. T. et al. Ndex: a community resource for sharing and publishing of biological networks. In: **Protein Bioinformatics**. [S.l.]: Springer, 2017. p. 271–301.

POLLARD, K. S. et al. Detection of nonneutral substitution rates on mammalian phylogenies. **Genome Research**, Cold Spring Harbor Lab, v. 20, n. 1, p. 110–121, 2010.

POULOS, R. C.; WONG, J. W. Finding cancer driver mutations in the era of big data research. **Biophysical Reviews**, Springer, v. 11, n. 1, p. 21–29, 2019.

PRASAD, T. K. et al. Human protein reference database—2009 update. **Nucleic Acids Research**, Oxford University Press, v. 37, n. suppl_1, p. D767–D772, 2009.

PRATT, D. et al. Ndex 2.0: a clearinghouse for research on cancer pathways. **Cancer Research**, AACR, v. 77, n. 21, p. e58–e61, 2017.

PRATT, D. et al. Ndex, the network data exchange. **Cell Systems**, Elsevier, v. 1, n. 4, p. 302–305, 2015.

QUANG, D.; CHEN, Y.; XIE, X. Dann: a deep learning approach for annotating the pathogenicity of genetic variants. **Bioinformatics**, Oxford University Press, v. 31, n. 5, p. 761–763, 2015.

RAIMONDI, D. et al. Current cancer driver variant predictors learn to recognize driver genes instead of functional variants. **BMC Biology**, Springer, v. 19, n. 1, p. 1–12, 2021.

RASCHKA, S. **Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning**. arXiv, 2018. Available from Internet: <https://arxiv.org/abs/1811.12808>.

REPANA, D. et al. The Network of Cancer Genes (NCG): a comprehensive catalogue of known and candidate cancer genes from cancer sequencing screens. **Genome Biology**, Springer, v. 20, n. 1, p. 1–12, 2019.

REVA, B.; ANTIPIN, Y.; SANDER, C. Predicting the functional impact of protein mutations: application to cancer genomics. **Nucleic Acids Research**, Oxford University Press, v. 39, n. 17, p. e118–e118, 2011.

RITCHIE, G. R. et al. Functional annotation of noncoding sequence variants. **Nature Methods**, Nature Publishing Group, v. 11, n. 3, p. 294–296, 2014.

ROBINSON, J. L.; NIELSEN, J. Integrative analysis of human omics data using biomolecular networks. **Molecular BioSystems**, Royal Society of Chemistry, v. 12, n. 10, p. 2953–2964, 2016.

ROGERS, M. F.; GAUNT, T. R.; CAMPBELL, C. Prediction of driver variants in the cancer genome via machine learning methodologies. **Briefings in Bioinformatics**, v. 22, n. 4, 10 2020. Bbaa250.

SAGI, O.; ROKACH, L. Ensemble learning: A survey. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, Wiley Online Library, v. 8, n. 4, p. e1249, 2018.

SALWINSKI, L. et al. The database of interacting proteins: 2004 update. **Nucleic Acids Research**, Oxford University Press, v. 32, n. suppl_1, p. D449–D451, 2004.

SALZBERG, S. L. Open questions: How many genes do we have? **BMC Biology**, BioMed Central, v. 16, n. 1, p. 1–3, 2018.

SANT'ANNA, C. de C. et al. Molecular biology as a tool for the treatment of cancer. **Clinical and Experimental Medicine**, Springer, v. 18, n. 4, p. 457–464, 2018.

SCHAEFER, M. H. et al. Hippie: Integrating protein interaction networks with experiment based quality scores. **PLoS ONE**, Public Library of Science San Francisco, USA, v. 7, n. 2, p. e31826, 2012.

SCHROEDER, M. P. et al. OncodriveROLE classifies cancer driver genes in loss of function and activating mode of action. **Bioinformatics**, v. 30, n. 17, p. 549–555, 2014. ISSN 14602059.

SCHUBACH, M. et al. Imbalance-aware machine learning for predicting rare and common disease-associated non-coding variants. **Scientific Reports**, Nature Publishing Group, v. 7, n. 1, p. 1–12, 2017.

SCHULTE-SASSE, R. et al. Graph Convolutional Networks Improve the Prediction of Cancer Driver Genes. In: TETKO, I. V. et al. (Ed.). **Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions. Lecture Notes in Computer Science**. [S.l.: s.n.], 2019. v. 11731, p. 658–668. ISBN 9783030304928. ISSN 16113349.

SCHULTE-SASSE, R. et al. Integration of multiomics data with graph convolutional networks to identify new cancer genes and their associated molecular mechanisms. **Nature Machine Intelligence**, Nature Publishing Group, v. 3, n. 6, p. 513–526, 2021.

SCHUSTER-BÖCKLER, B.; LEHNER, B. Chromatin organization is a major influence on regional mutation rates in human cancer cells. **Nature**, Nature Publishing Group, v. 488, n. 7412, p. 504–507, 2012.

SCHWARZ, J. M. et al. MutationTaster evaluates disease-causing potential of sequence alterations. **Nature Methods**, Nature Publishing Group, v. 7, n. 8, p. 575–576, 2010.

SHEN, Z. Genomic instability and cancer: an introduction. **Journal of Molecular Cell Biology**, v. 3, n. 1, p. 1–3, 02 2011. ISSN 1674-2788. Available from Internet: <https://doi.org/10.1093/jmcb/mjq057>.

SHIHAB, H. A. et al. Predicting the functional, molecular, and phenotypic consequences of amino acid substitutions using hidden markov models. **Human Mutation**, Wiley Online Library, v. 34, n. 1, p. 57–65, 2013.

SHIHAB, H. A. et al. An integrative approach to predicting the functional effects of non-coding and coding sequence variation. **Bioinformatics**, Oxford University Press, v. 31, n. 10, p. 1536–1543, 2015.

SIMPSON, J. **Understanding Bioinformatics**. Murphy & Moore Publishing, 2022. ISBN 9781639875450. Available from Internet: <https://books.google.com.br/books?id=DoqzzgEACAAJ>.

SOLIMAN, A. T. et al. Driver missense mutation identification using feature selection and model fusion. **Journal of Computational Biology**, v. 22, n. 12, p. 1075–1085, 2015. ISSN 10665277.

SONDKA, Z. et al. The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers. **Nature Reviews Cancer**, Nature Publishing Group, v. 18, n. 11, p. 696–705, 2018.

STARK, C. et al. BioGRID: a general repository for interaction datasets. **Nucleic Acids Research**, Oxford University Press, v. 34, n. suppl_1, p. D535–D539, 2006.

STELZL, U. et al. A human protein-protein interaction network: a resource for annotating the proteome. **Cell**, Elsevier, v. 122, n. 6, p. 957–968, 2005.

STRATTON, M. R.; CAMPBELL, P. J.; FUTREAL, P. A. The cancer genome. **Nature**, Nature Publishing Group, v. 458, n. 7239, p. 719–724, 2009.

TALUKDER, A. et al. Interpretation of deep learning in genomics and epigenomics. **Briefings in Bioinformatics**, Oxford University Press, v. 22, n. 3, p. bbaa177, 2021.

TAMBORERO, D.; GONZALEZ-PEREZ, A.; LOPEZ-BIGAS, N. OncodriveCLUST: exploiting the positional clustering of somatic mutations to identify cancer genes. **Bioinformatics**, Oxford University Press, v. 29, n. 18, p. 2238–2244, 2013.

TAMBORERO, D. et al. Comprehensive identification of mutational cancer driver genes across 12 tumor types. **Scientific Reports**, Nature Publishing Group, v. 3, n. 1, p. 1–10, 2013.

TAN, H.; BAO, J.; ZHOU, X. A novel missense-mutation-related feature extraction scheme for 'driver' mutation identification. **Bioinformatics**, v. 28, n. 22, p. 2948–2955, 2012. ISSN 13674803.

TATE, J. G. et al. COSMIC: the Catalogue Of Somatic Mutations In Cancer. **Nucleic Acids Research**, v. 47, n. D1, p. D941–D947, 10 2018. ISSN 0305-1048. Available from Internet: <https://doi.org/10.1093/nar/gky1015>.

TAVANAEI, A. et al. A deep learning model for predicting tumor suppressor genes and oncogenes from PDB structure. In: **2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)**. IEEE, 2017. p. 613–617. ISBN 978-1-5090-3050-7. Available from Internet: <http://ieeexplore.ieee.org/document/8217722/>.

THOMAS, P. D. et al. Applications for protein sequence–function evolution data: mrna/protein expression analysis and coding snp scoring tools. **Nucleic Acids Research**, Oxford University Press, v. 34, n. suppl_2, p. W645–W650, 2006.

TOKHEIM, C. J. et al. Evaluating the evaluation of cancer driver genes. **Proceedings of the National Academy of Sciences of the United States of America**, v. 113, n. 50, p. 14330–14335, 2016. ISSN 10916490.

TOMASETTI, C.; VOGELSTEIN, B. Variation in cancer risk among tissues can be explained by the number of stem cell divisions. **Science**, American Association for the Advancement of Science, v. 347, n. 6217, p. 78–81, 2015.

TRLIN, G. **Multilayer Perceptron tutorial - building one from scratch in Python**. 2023. Available from Internet: <https://www.playandlearntocode.com/article/multilayer-perceptron-tutorial-building-one-from-scratch>.

U, M. C. et al. Prediction and Prioritization of Rare Oncogenic Mutations in the Cancer Kinome Using Novel Features and Multiple Classifiers. **PLoS Computational Biology**, v. 10, n. 4, 2014. ISSN 15537358.

VABALAS, A. et al. Machine learning algorithm validation with a limited sample size. **PLoS ONE**, Public Library of Science San Francisco, CA USA, v. 14, n. 11, p. e0224365, 2019.

VAQUERIZAS, J. M. et al. A census of human transcription factors: function, expression and evolution. **Nature Reviews Genetics**, Nature Publishing Group, v. 10, n. 4, p. 252–263, 2009.

VARMA, S.; SIMON, R. Bias in error estimation when using cross-validation for model selection. **BMC Bioinformatics**, Springer, v. 7, n. 1, p. 1–8, 2006.

VELIČKOVIĆ, P. et al. Graph attention networks. **arXiv preprint arXiv:1710.10903**, 2017.

VOGELSTEIN, B. et al. Cancer genome landscapes. **Science**, American Association for the Advancement of Science, v. 339, n. 6127, p. 1546–1558, 2013.

WANG, H. et al. AI-Driver: an ensemble method for identifying driver mutations in personal cancer genomes. **NAR Genomics and Bioinformatics**, Oxford Academic, v. 2, n. 4, dec 2020. ISSN 2631-9268. Available from Internet: <https://academic.oup.com/nargab/article/2/4/lqaa084/5922823https://academic.oup.com/nargab/article/doi/10.1093/nargab/lqaa084/5922823>.

WANG, Q. et al. Unifying cancer and normal RNA sequencing data from different sources. **Scientific Data**, Nature Publishing Group, v. 5, n. 1, p. 1–8, 2018.

WANG, Z. et al. Cancer driver mutation prediction through Bayesian integration of multi-omic data. **PLoS ONE**, v. 13, n. 5, p. 1–19, 2018. ISSN 19326203.

WU, J. et al. dbWGFP: a database and web server of human whole-genome single nucleotide variants and their functional predictions. **Database**, Oxford Academic, v. 2016, 2016.

WU, S. et al. Evaluating intrinsic and non-intrinsic cancer risk factors. **Nature Communications**, Nature Publishing Group UK London, v. 9, n. 1, p. 3490, 2018.

WU, Z. et al. A comprehensive survey on graph neural networks. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, v. 32, n. 1, p. 4–24, 2020.

XI, J. et al. Inferring subgroup-specific driver genes from heterogeneous cancer samples via subspace learning with subgroup indication. **Bioinformatics**, v. 36, n. 6, p. 1855–1863, oct 2019. ISSN 1367-4803. Available from Internet: <https://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/btz793/5593650>.

XIAO, F. et al. miRecords: an integrated resource for microRNA–target interactions. **Nucleic Acids Research**, Oxford University Press, v. 37, n. suppl_1, p. D105–D110, 2009.

YATES, L. R.; CAMPBELL, P. J. Evolution of the cancer genome. **Nature Reviews Genetics**, Nature Publishing Group, v. 13, n. 11, p. 795–806, 2012.

YIP, Y. L. et al. Annotating single amino acid polymorphisms in the UniProt/Swiss-Prot knowledgebase. **Human Mutation**, Wiley Online Library, v. 29, n. 3, p. 361–366, 2008.

YUAN, Y. et al. A sparse regulatory network of copy-number driven gene expression reveals putative breast cancer oncogenes. **IEEE/ACM Transactions on Computational Biology and Bioinformatics**, IEEE, v. 9, n. 4, p. 947–954, 2011.

YUAN, Y.; WANG, W.; PANG, W. A Systematic Comparison Study on Hyperparameter Optimisation of Graph Neural Networks for Molecular Property Prediction. In: **Proceedings of the Genetic and Evolutionary Computation Conference**. New York, NY, USA: Association for Computing Machinery, 2021. (GECCO '21), p. 386–394. ISBN 9781450383509. Available from Internet: <https://doi.org/10.1145/3449639.3459370>.

ZHANG, J. et al. Identifying driver mutations from sequencing data of heterogeneous tumors in the era of personalized genome sequencing. **Briefings in Bioinformatics**, Oxford University Press, v. 15, n. 2, p. 244–255, 2014.

ZHANG, J.; ZHANG, S. The discovery of mutated driver pathways in cancer: Models and algorithms. **IEEE/ACM Transactions on Computational Biology and Bioinformatics**, IEEE, v. 15, n. 3, p. 988–998, 2016.

ZHANG, S. et al. Graph convolutional networks: a comprehensive review. **Computational Social Networks**, SpringerOpen, v. 6, n. 1, p. 1–23, 2019.

ZHANG, X.-M. et al. Graph neural networks and their current applications in bioinformatics. **Frontiers in Genetics**, Frontiers Media SA, v. 12, 2021.

ZHAO, T.; ZHANG, X.; WANG, S. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In: **Proceedings of the 14th ACM international conference on web search and data mining**. [S.l.: s.n.], 2021. p. 833–841.

ZHOU, H.; GAO, M.; SKOLNICK, J. Entprise: an algorithm for predicting human disease-associated amino acid substitutions from sequence entropy and predicted protein structures. **PLoS ONE**, Public Library of Science San Francisco, CA USA, v. 11, n. 3, p. e0150965, 2016.

ZHOU, H.; GAO, M.; SKOLNICK, J. ENTPRISE-X: Predicting disease-associated frameshift and nonsense mutations. **PLoS ONE**, v. 13, n. 5, p. e0196849, may 2018. ISSN 1932-6203. Available from Internet: <https://dx.plos.org/10.1371/journal.pone.0196849>.

ZHOU, J. et al. Graph neural networks: A review of methods and applications. **AI open**, Elsevier, v. 1, p. 57–81, 2020.

ZHU, C. Y. et al. C3: Consensus Cancer Driver Gene Caller. **Genomics, Proteomics & Bioinformatics**, Elsevier B.V., v. 17, n. 3, p. 311–318, 2019. ISSN 22103244. Available from Internet: <https://doi.org/10.1016/j.gpb.2018.10.004>.

ZOU, J.; HAN, Y.; SO, S.-S. Overview of artificial neural networks. In: ____. **Artificial Neural Networks: Methods and Applications**. Totowa, NJ: Humana Press, 2009. p. 14–22. ISBN 978-1-60327-101-1. Available from Internet: <https://doi.org/10.1007/978-1-60327-101-1_2>.

APPENDIX A — MATERIAL OF LITERATURE REVIEW

Table A.1 – Summary of the scope and extent to which ML-based methods are covered by the main reviews related to the prediction of cancer driver genes.

Reference	**Brief description of review's scope**
Zhang et al. (2014)	Biology-based and ML-based approaches to identify driver mutations. Methods are categorized into three strategies: analysis of difference in mutation frequencies between driver and passenger mutations, prediction of functional impacts of mutation, and prediction provided by ML models. Only four ML-based methods are included, two of which were developed for the general problem of distinguishing disease-related mutations from common polymorphisms.
Chen, Sun and Shen (2015)	Tools to decipher oncogenic drivers, categorized according to the specific mutational level tackled, from nucleotide- to network-level. For nucleotide-level approaches, six ML methods are discussed, but only four were specifically designed for classifying somatic mutations as driver or passenger ones.
Cheng, Zhao and Zhao (2016)	Computational approaches to identify driver mutations and significantly mutated genes from next-generation sequencing data, categorized according to the types of features used: mutation frequency, functional impact, structural genomics, network or pathway, and data integration. Authors mention that ML underlie most tools reviewed, but do not provide methodological details regarding ML algorithms used and model training procedures adopted.
Zhang and Zhang (2016)	Brief summary of nine methods to predict driver genes, including two ML-based methods, and a more extensive review regarding methods for identifying driver pathways or modules in cancer. For the second part, the methods reviewed do not use ML algorithms but rather probability models or network-based approaches.
Dimitrakopoulos and Beerenwinkel (2017)	Three different classes of approaches for prediction of cancer genes and pathways: methods using known biological pathways, network-based methods, and methods detecting pathways *de novo*. Although ML algorithms may be eventually used in the methods reviewed, they were not the focus of the discussion carried out by the authors.
Chen et al. (2020)	Comprehensive assessment of computational methods for prediction of cancer driver mutations, in which 33 methods were analyzed in terms of predictive performance using five consistent and complementary benchmark datasets. Authors include some ML-based methods in their analyses, briefly describing the type of features used, but their emphasis is in comparing methods' performance rather than reviewing methodological details of their implementation.
Pham et al. (2021)	Computational methods for prediction of cancer drivers categorized according to their prediction target: single driver identification, cancer driver module identification, and personalized cancer driver identification. Twenty-four methods are discussed and eight were selected for a comparative study of predictive performance. The ML algorithms adopted by some of the reviewed methods are only cited, without further discussion about features used and other methodological details.
Rogers, Gaunt and Campbell (2020)	Seventeen generic ML-based tools to predict the pathogenic impact of human genome variants and eight tools specialized in cancer driver mutations. Most reviewed approaches are based on supervised learning techniques and data integration strategies. Although authors briefly mention the ML algorithms employed in these tools, the review is limited in the number of papers covered and in providing a detailed description about methodological aspects of tools' development, such as biological features used and how they are jointly explored with distinct ML approaches. Also, a substantial part of the review is dedicated to analyze the incidence of predicted high-confidence driver mutations obtained with one specific ML-based predictor for well-known cancer genes across distinct cancer types.

APPENDIX B — ADDITIONAL RESULTS FOR THE COMPARATIVE EVALUATION OF ALGORITHMS AND DATA FOR CDG PREDICTION

This appendix provides additional results for the experiments reported in Chapter 5, which aim to conduct an experimental comparison of learning algorithms and data-centric decisions for the prediction of CDGs.

Table B.1 – Complete experiments for the HPRD network, referring to the test data for the AUC-PR metric.

	GAT					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=1; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.3565	0.3324	0.3604	0.3262	0.2885	0.3158
CNA	0.2761	0.2616	0.3599	0.3737	0.3785	0.3358
DNA Methylation	0.3844	0.3030	0.3152	0.3539	0.3340	0.2451
Gene Expression	0.2806	0.2672	0.3578	0.3202	0.3205	0.2794
Multi-Omics	0.3258	0.2901	0.4020	0.4546	0.4153	0.3540
Multi-Omics and Centrality	0.3992	0.3275	0.4215	0.3873	0.3581	0.3792
	GCN					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.4470	0.4390	0.4783	0.4079	0.4230	0.5160
CNA	0.3992	0.4266	0.4722	0.4541	0.2098	0.4090
DNA Methylation	0.3708	0.3568	0.3995	0.2905	0.1992	0.3612
Gene Expression	0.2967	0.3989	0.4313	0.3793	0.1495	0.3600
Multi-Omics	0.4984	0.4988	0.4771	0.5371	0.5465	0.5524
Multi-Omics and Centrality	0.5960	0.5479	0.5918	0.5235	0.5756	0.5583
	GraphSAGE					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.4089	0.3346	0.4371	0.3682	0.3903	0.3840
CNA	0.3134	0.2780	0.2573	0.3029	0.2952	0.2807
DNA Methylation	0.2402	0.2073	0.2576	0.2931	0.2297	0.2620
Gene Expression	0.2397	0.2194	0.2608	0.2651	0.2308	0.2333
Multi-Omics	0.3317	0.2258	0.3052	0.3950	0.3543	0.3472
Multi-Omics and Centrality	0.3786	0.3332	0.4069	0.4230	0.3985	0.4480

Table B.2 – Complete experiments for the MULTINET network, referring to the test data for the AUC-PR metric.

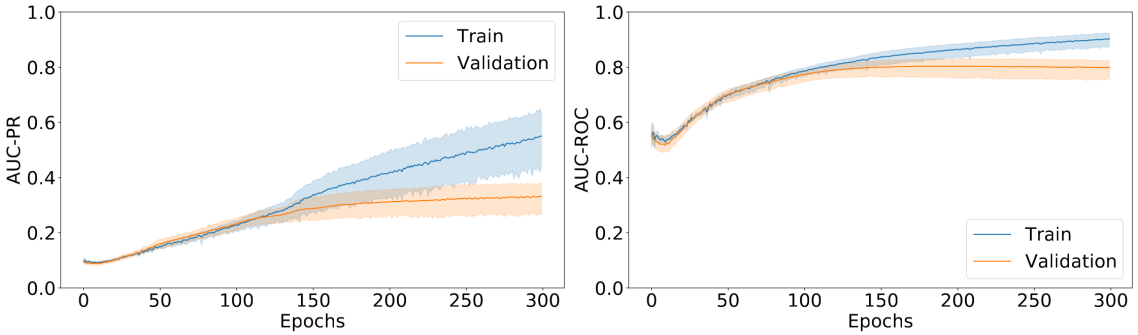
	GAT					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=1; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.2142	0.2492	0.2297	0.2857	0.1852	0.1988
CNA	0.2865	0.1982	0.2241	0.2044	0.2441	0.2442
DNA Methylation	0.2219	0.2741	0.2158	0.2495	0.2065	0.2599
Gene Expression	0.2562	0.1758	0.1746	0.2169	0.2303	0.1995
Multi-Omics	0.2182	0.3318	0.3687	0.4045	0.2117	0.3701
Multi-Omics and Centrality	0.3648	0.2817	0.3551	0.4143	0.3363	0.2753
	GCN					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.3523	0.3704	0.4261	0.3711	0.3672	0.3793
CNA	0.3277	0.3621	0.3808	0.3008	0.0735	0.2855
DNA Methylation	0.3185	0.3075	0.3499	0.3013	0.1824	0.2726
Gene Expression	0.2506	0.3451	0.3360	0.3324	0.2711	0.2921
Multi-Omics	0.3839	0.3153	0.3474	0.4133	0.2783	0.4272
Multi-Omics and Centrality	0.4019	0.4521	0.4661	0.4177	0.2978	0.4481
	GraphSAGE					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.3842	0.2907	0.3543	0.3835	0.3435	0.3196
CNA	0.1996	0.1821	0.2083	0.2029	0.2086	0.2146
DNA Methylation	0.1770	0.1276	0.1792	0.1612	0.1719	0.1918
Gene Expression	0.1981	0.1540	0.2021	0.1846	0.1782	0.1867
Multi-Omics	0.2671	0.2050	0.2685	0.3039	0.3121	0.2620
Multi-Omics and Centrality	0.3174	0.2188	0.3929	0.3264	0.3196	0.3822

Table B.3 – Complete experiments for the IREF network, referring to the test data for the AUC-PR metric.

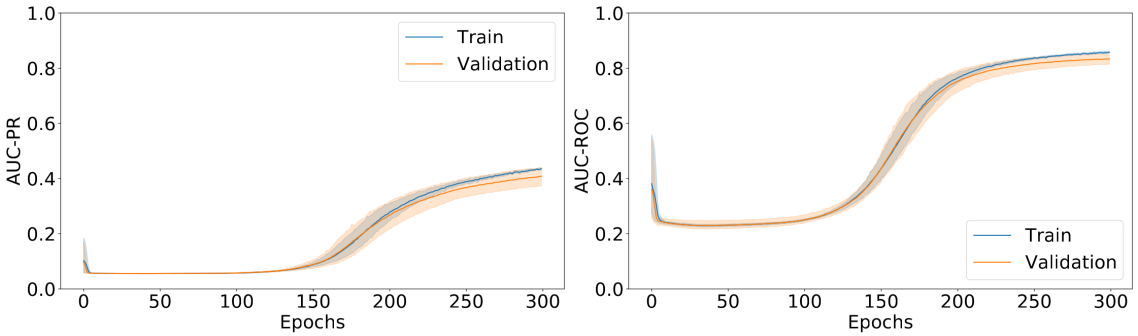
	GAT					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=1; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.2606	0.1668	0.2748	0.3085	0.2580	0.3240
CNA	0.3027	0.2270	0.3147	0.2950	0.2828	0.2929
DNA Methylation	0.3088	0.3135	0.3365	0.1831	0.2061	0.2437
Gene Expression	0.2211	0.2500	0.2342	0.2542	0.2900	0.2878
Multi-Omics	0.2144	0.2444	0.2727	0.3523	0.3169	0.3955
Multi-Omics and Centrality	0.3598	0.1714	0.3135	0.3546	0.2977	0.3917
	GCN					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.4344	0.3862	0.3964	0.3268	0.3289	0.3714
CNA	0.2844	0.3159	0.3855	0.3972	0.0864	0.3528
DNA Methylation	0.2509	0.2937	0.2901	0.2744	0.0883	0.2661
Gene Expression	0.2699	0.3221	0.3803	0.2760	0.0691	0.3898
Multi-Omics	0.4315	0.4834	0.4084	0.4813	0.4191	0.3723
Multi-Omics and Centrality	0.4274	0.4384	0.4841	0.4664	0.3776	0.4488
	GraphSAGE					
	Cross entropy	Undersampling	Weighted focal loss (γ=0; α=0.85)	Focal loss (γ=0.5; α=0.50)	Focal loss (γ=2; α=0.25)	Focal loss (γ=2; α=0.25)
Mutations	0.3550	0.2478	0.3239	0.2762	0.4245	0.3643
CNA	0.2036	0.1777	0.1880	0.1970	0.1854	0.1660
DNA Methylation	0.1482	0.1790	0.1697	0.1643	0.1835	0.2215
Gene Expression	0.1850	0.1389	0.1674	0.2232	0.1546	0.1796
Multi-Omics	0.2001	0.1597	0.2187	0.2208	0.2674	0.2985
Multi-Omics and Centrality	0.2427	0.1887	0.3013	0.2296	0.2832	0.2928

Figure B.1 – Training and validation for the MULTINET network in (a) GAT, (b) GCN and (c) GraphSAGE using 300 epochs and AUC-PR and AUC-ROC as performance metrics

(a) GAT



(b) GCN



(c) GraphSAGE

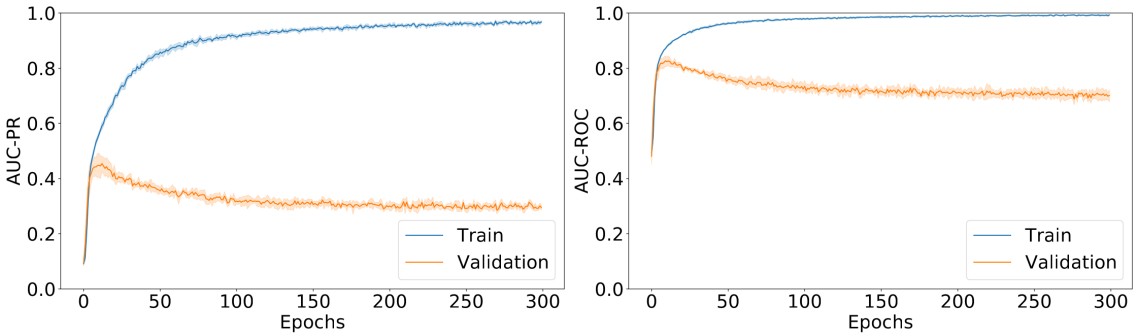
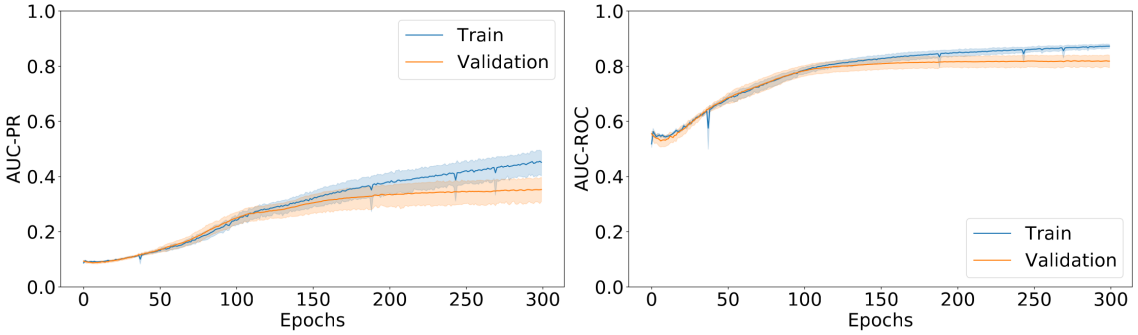
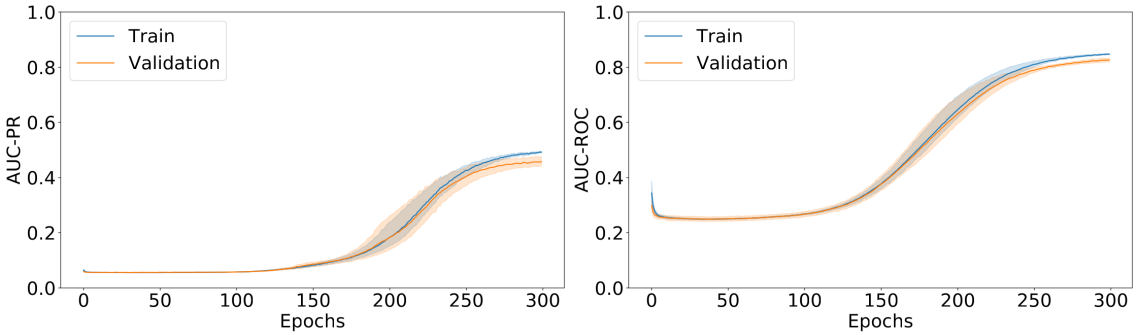


Figure B.2 – Training and validation for the IREF network in (a) GAT, (b) GCN and (c) GraphSAGE using 300 epochs and AUC-PR and AUC-ROC as performance metrics

(a) GAT



(b) GCN



(c) GraphSAGE

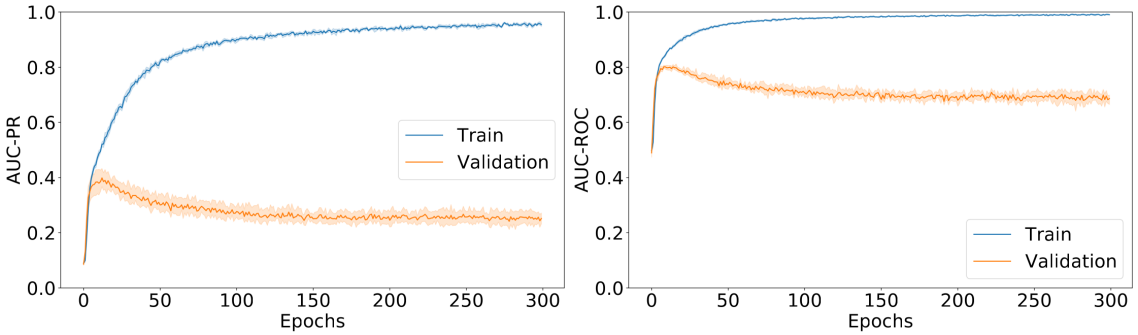
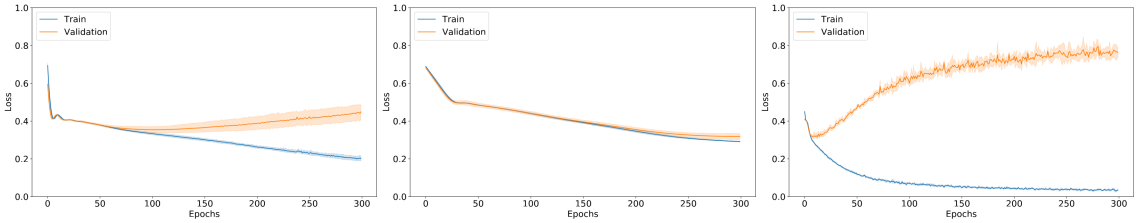
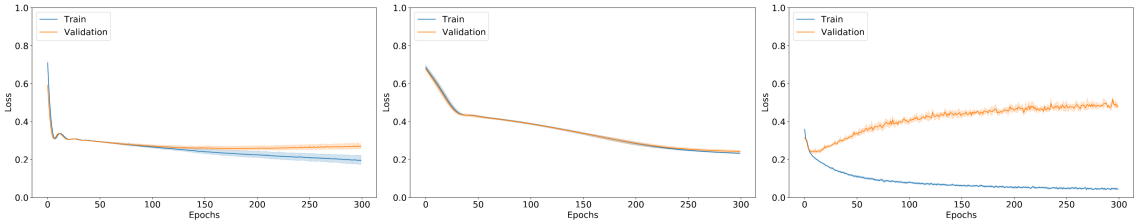


Figure B.3 – Loss curve for the GAT (left column), GCN (middle column), and GraphSAGE (right column) models in (a) HPRD, (b) MULTINET and (c) IREF PPI networks.

(a) HPRD



(b) MULTINET



(c) IREF

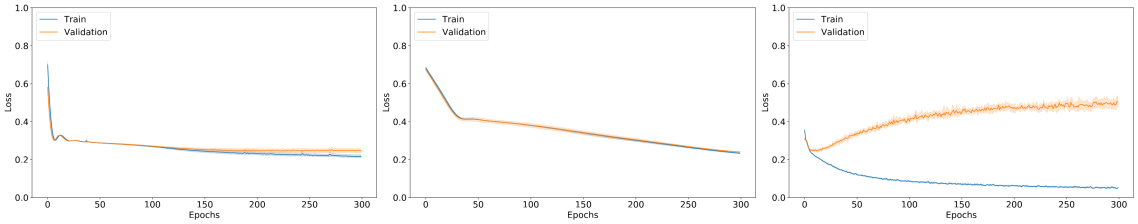


Table B.4 – Experiments for three traditional algorithms in HPRD network (mean used for the metrics presented, from test data)

	SVM				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.8701	0.7399	0.1172	0.5877	0.3272
CNA	0.8600	0.2000	0.0025	0.5297	0.1950
DNA Methylation	0.8600	0.0000	0.0000	0.4974	0.1435
Gene Expression	0.8600	0.0000	0.0000	0.4777	0.1368
Multi-Omics	0.8603	0.2000	0.0025	0.6346	0.2595
Multi-Omics and Centrality	0.8667	0.8512	0.0618	0.7608	0.4477
	Random Forest				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.8699	0.6835	0.1386	0.7031	0.3664
CNA	0.8538	0.3411	0.0466	0.5303	0.1702
DNA Methylation	0.8591	0.0000	0.0000	0.5208	0.1519
Gene Expression	0.8596	0.0000	0.0000	0.5362	0.1559
Multi-Omics	0.8674	0.7072	0.0945	0.7073	0.3566
Multi-Omics and Centrality	0.8840	0.7441	0.2672	0.8054	0.5231
	Gradient Boosting Tree				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.8681	0.6217	0.1525	0.6978	0.3753
CNA	0.8614	0.6783	0.0214	0.5249	0.1846
DNA Methylation	0.8582	0.0500	0.0012	0.5225	0.1487
Gene Expression	0.8579	0.1066	0.0025	0.5386	0.1564
Multi-Omics	0.8671	0.6178	0.1398	0.7015	0.3565
Multi-Omics and Centrality	0.8861	0.7079	0.3240	0.8174	0.5437

Table B.5 – Experiments for three traditional algorithms in MULTINET network (mean used for the metrics presented, from test data)

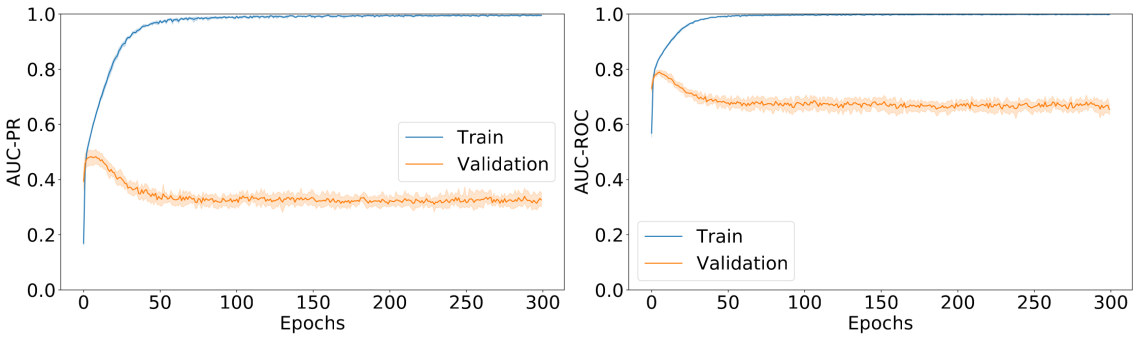
	SVM				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9129	0.7395	0.1025	0.5667	0.2819
CNA	0.9069	0.0000	0.0000	0.5072	0.1247
DNA Methylation	0.9072	0.0000	0.0000	0.5027	0.0951
Gene Expression	0.9072	0.0000	0.0000	0.4793	0.0997
Multi-Omics	0.9073	0.2000	0.0011	0.6506	0.2025
Multi-Omics and Centrality	0.9075	0.4000	0.0034	0.7523	0.3126
	Random Forest				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9121	0.6589	0.1152	0.7120	0.3066
CNA	0.9006	0.1393	0.0126	0.5357	0.1136
DNA Methylation	0.9070	0.0000	0.0000	0.5466	0.1122
Gene Expression	0.9070	0.0000	0.0000	0.5630	0.1071
Multi-Omics	0.9111	0.6618	0.0875	0.7238	0.2976
Multi-Omics and Centrality	0.9193	0.7331	0.2061	0.8193	0.4559
	Gradient Boosting Tree				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9114	0.6098	0.1325	0.7228	0.3212
CNA	0.9053	0.4438	0.0115	0.5469	0.1169
DNA Methylation	0.9062	0.1666	0.0023	0.5513	0.1089
Gene Expression	0.9065	0.0000	0.0000	0.5803	0.1151
Multi-Omics	0.9092	0.5534	0.1210	0.7294	0.3066
Multi-Omics and Centrality	0.9166	0.6299	0.2442	0.8295	0.4505

Table B.6 – Experiments for three traditional algorithms in IREF network (mean used for the metrics presented, from test data)

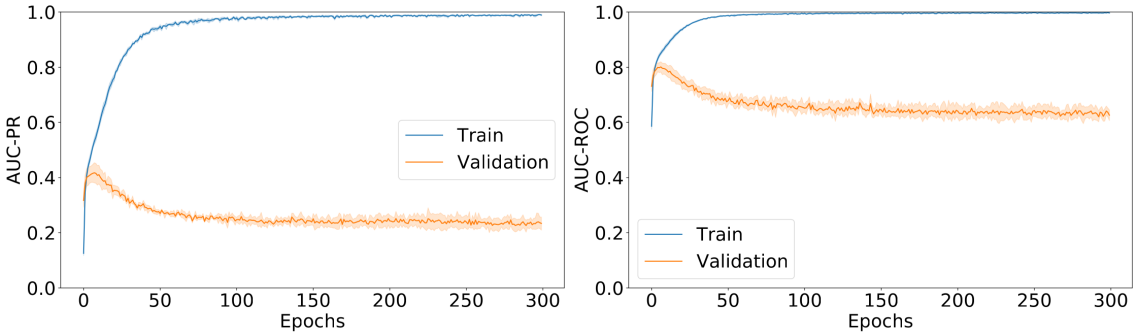
	SVM				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9148	0.7288	0.0938	0.5714	0.2598
CNA	0.9098	0.0000	0.0000	0.5131	0.1204
DNA Methylation	0.9100	0.0000	0.0000	0.5161	0.0969
Gene Expression	0.9100	0.0000	0.0000	0.4833	0.0906
Multi-Omics	0.9101	0.2000	0.0011	0.6433	0.1958
Multi-Omics and Centrality	0.9101	0.2000	0.0011	0.7060	0.2368
	Random Forest				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9139	0.6549	0.1064	0.7084	0.2929
CNA	0.9041	0.1731	0.0182	0.5714	0.1201
DNA Methylation	0.9100	0.0000	0.0000	0.5227	0.1023
Gene Expression	0.9098	0.0000	0.0000	0.5448	0.1016
Multi-Omics	0.9131	0.6950	0.0697	0.7160	0.2869
Multi-Omics and Centrality	0.9181	0.7262	0.1476	0.8006	0.4052
	Gradient Boosting Tree				
	Accuracy	Precision	Recall	AUC-ROC	AUC-PR
Mutations	0.9127	0.5832	0.1155	0.7245	0.3087
CNA	0.9103	0.6833	0.0114	0.5705	0.1268
DNA Methylation	0.9090	0.1066	0.0022	0.5398	0.1044
Gene Expression	0.9087	0.0000	0.0000	0.5607	0.1077
Multi-Omics	0.9129	0.5907	0.1167	0.7285	0.3029
Multi-Omics and Centrality	0.9187	0.6421	0.2231	0.8233	0.4230

Figure B.4 – Training and validation for the Neural Network algorithm in (a) HPRD, (b) MULTINET and (c) IREF using 300 epochs and AUC-PR and AUC-ROC as performance metrics

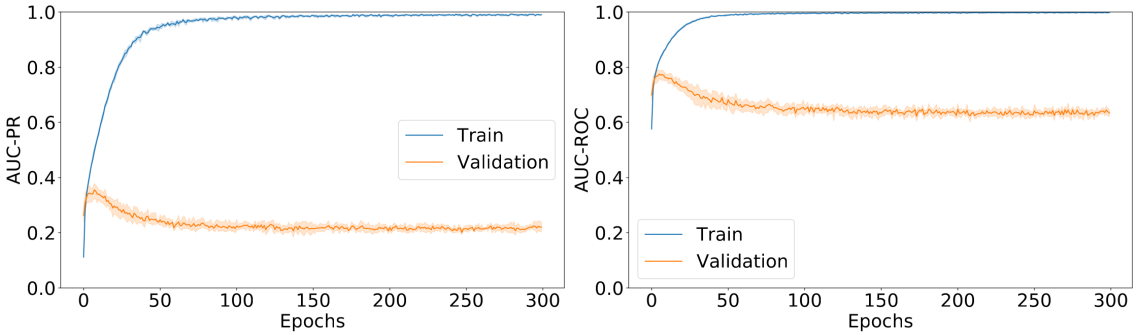
(a) HPRD



(b) MULTINET



(c) IREF



APPENDIX C — ADDITIONAL RESULTS FOR THE DEVELOPMENT OF GCN-BASED MODELS FOR PREDICTING CDGS

This appendix provides additional results for the experiments reported in Chapter 6, which aim to train predictive models based on the GCN algorithm for the identification of CDGs.

Table C.1 – Occurrence of hyperparameters in the results according to different thresholds for the HPRD network

AUC-PR >0.2 (556 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	49.6%	0.001	33.2%	0.25	14.0%	0	23.7%	64	30.7%	0.00001	14.0%
ELU	50.3%	0.01	33.2%	0.5	20.1%	0.5	23.9%	128	32.5%	0.0001	34.1%
		0.1	33.4%	0.75	26.9%	1	24.8%	256	36.6%	0.001	51.7%
				0.9	38.8%	2	27.5%				

AUC-PR >0.4 (457 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	50.1%	0.001	33.4%	0.25	15.7%	0	23.6%	64	31.5%	0.00001	0.8%
ELU	49.8%	0.01	33.2%	0.5	20.3%	0.5	24.2%	128	32.8%	0.0001	36.1%
		0.1	33.2%	0.75	31.5%	1	25.1%	256	35.6%	0.001	63.0%
				0.9	32.3%	2	26.9%				

AUC-PR >0.53 (97 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	84.5%	0.001	24.7%	0.25	36.0%	0	22.6%	64	39.1%	0.00001	0.0%
ELU	15.4%	0.01	27.8%	0.5	22.6%	0.5	27.8%	128	30.9%	0.0001	9.2%
		0.1	47.4%	0.75	23.7%	1	27.8%	256	29.8%	0.001	90.7%
				0.9	17.5%	2	21.6%				

Table C.2 – Occurrence of hyperparameters in the results according to different thresholds for the MULTINET network

AUC-PR >0.2 (506 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	49.2%	0.001	33.4%	0.25	14.2%	0	23.7%	64	31.4%	0.00001	14.2%
ELU	50.7%	0.01	33.4%	0.5	16.4%	0.5	24.3%	128	33.5%	0.0001	28.8%
		0.1	33.2%	0.75	26.6%	1	25.4%	256	34.9%	0.001	56.9%
				0.9	42.6%	2	26.4%				

AUC-PR >0.4 (162 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	89.5%	0.001	33.9%	0.25	22.8%	0	20.3%	64	27.1%	0.00001	0.0%
ELU	10.4%	0.01	33.3%	0.5	26.5%	0.5	23.4%	128	30.8%	0.0001	3.1%
		0.1	32.7%	0.75	25.3%	1	28.3%	256	41.9%	0.001	96.9%
				0.9	25.3%	2	27.7%				

AUC-PR >0.46 (89 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	100%	0.001	35.9%	0.25	17.9%	0	17.9%	64	12.3%	0.00001	0.0%
ELU	0.0%	0.01	34.8%	0.5	28.0%	0.5	22.4%	128	40.4%	0.0001	0.0%
		0.1	29.2%	0.75	34.8%	1	26.9%	256	47.1%	0.001	100%
				0.9	19.1%	2	32.5%				

Table C.3 – Occurrence of hyperparameters in the results according to different thresholds for the IREF network

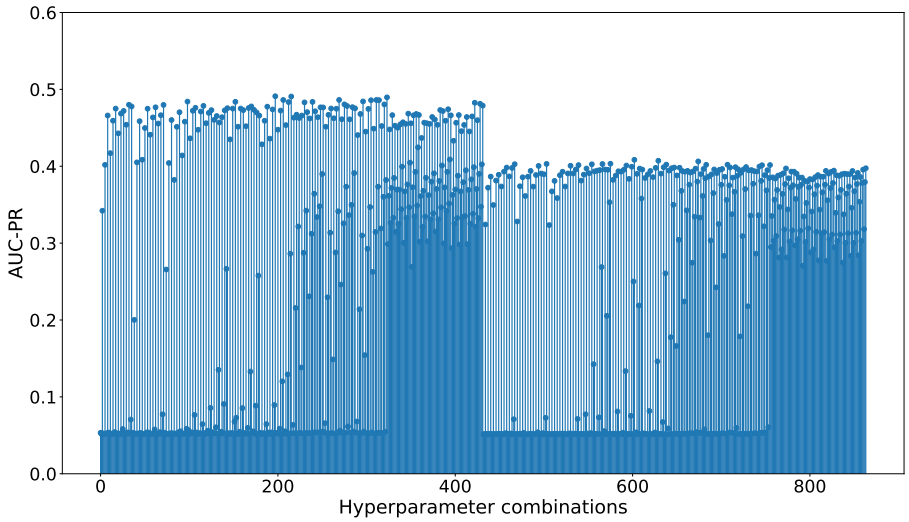
AUC-PR >0.2 (497 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	49.1%	0.001	33.4%	0.25	14.3%	0	23.5%	64	31.0%	0.00001	14.5%
ELU	50.9%	0.01	33.2%	0.5	16.3%	0.5	24.3%	128	33.4%	0.0001	27.8%
		0.1	33.4%	0.75	26.0%	1	25.6%	256	35.6%	0.001	57.7%
				0.9	43.5%	2	26.6%				

AUC-PR >0.4 (279 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	63.4%	0.001	34.1%	0.25	17.9%	0	21.1%	64	23.3%	0.00001	0.0%
ELU	36.6%	0.01	31.9%	0.5	24.0%	0.5	24.4%	128	35.5%	0.0001	12.9%
		0.1	34.1%	0.75	29.0%	1	25.4%	256	41.2%	0.001	87.1%
				0.9	29.0%	2	29.0%				

AUC-PR >0.474 (92 parameter combinations)											
Activation		Dropout		Loss (alpha)		Loss (gamma)		Layer sizes		Learning rate	
ReLU	100%	0.001	35.8%	0.25	21.7%	0	21.7%	64	21.8%	0.00001	0.0%
ELU	0.0%	0.01	32.6%	0.5	28.2%	0.5	27.1%	128	39.1%	0.0001	0.0%
		0.1	31.5%	0.75	29.3%	1	23.9%	256	39.1%	0.001	100%
				0.9	20.6%	2	27.1%				

Figure C.1 – Performance analysis of all hyperparameter combinations for the (a) MULTINET and (b) IREF networks.

(a) MULTINET



(b) IREF

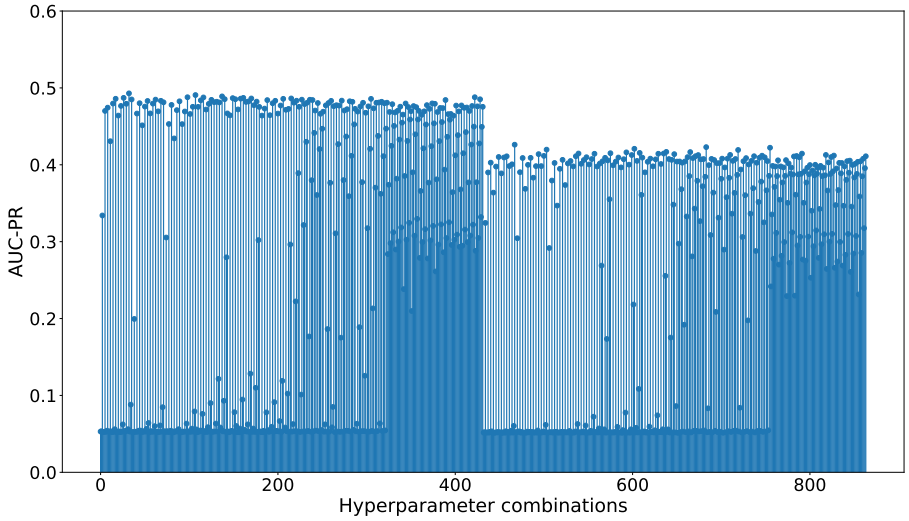
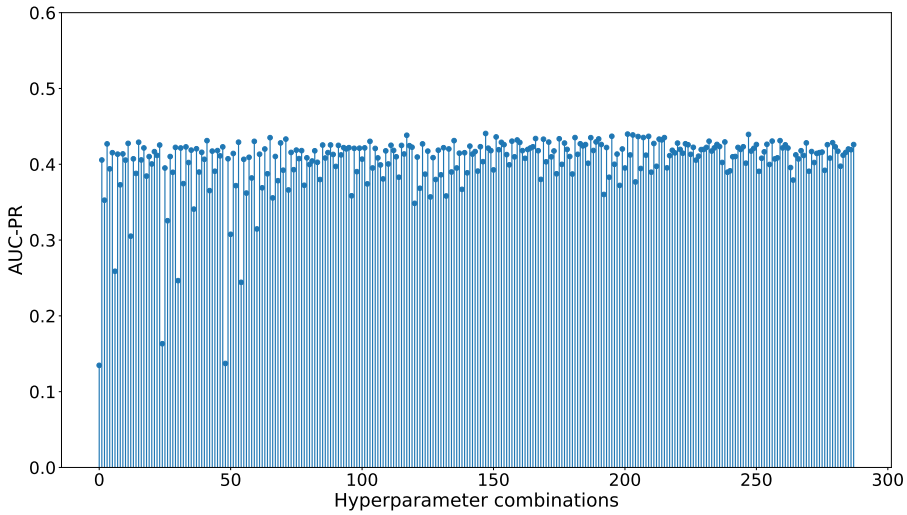


Figure C.2 – Performance analysis of all hyperparameter combinations for the (a) CPDB and (b) PCNET networks.

(a) CPDB



(b) PCNET

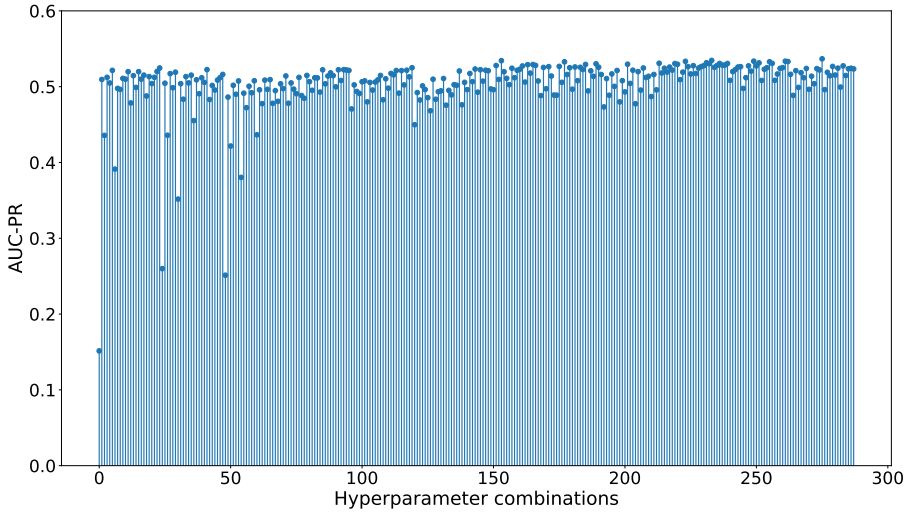
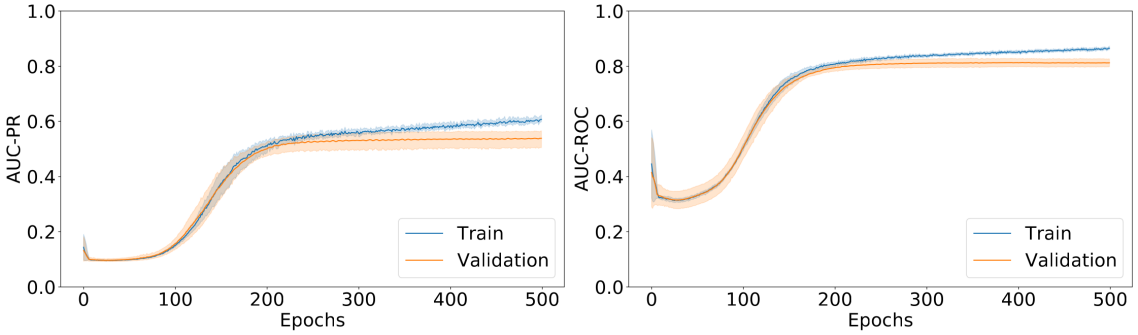
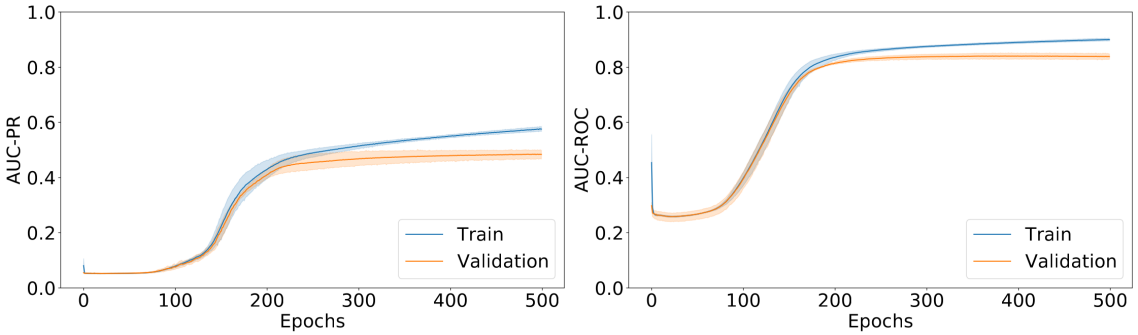


Figure C.3 – Training and validation performance for (a) HPRD, (b) IREF, and (c) CPDB. AUC-PR and AUC-ROC values are shown in the left and righ column, respectively

(a) HPRD



(b) IREF



(c) CPDB

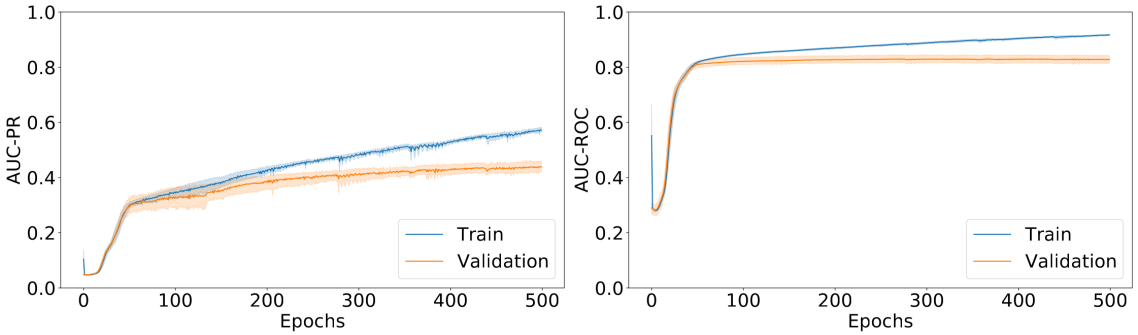
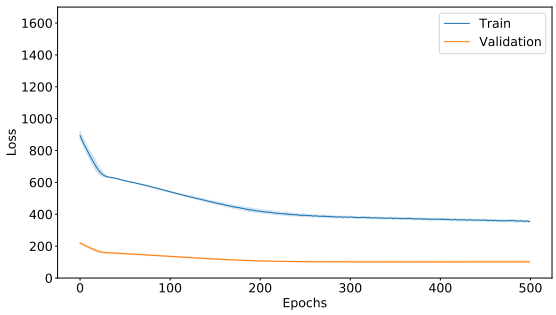
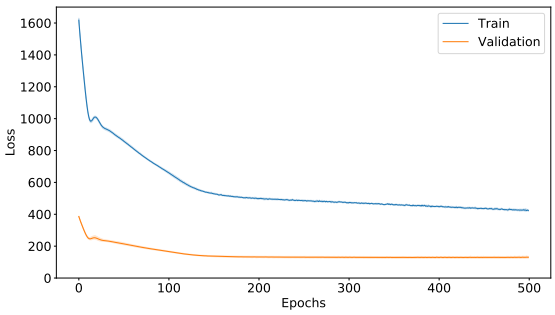


Figure C.4 – Loss curves for the six PPI networks.

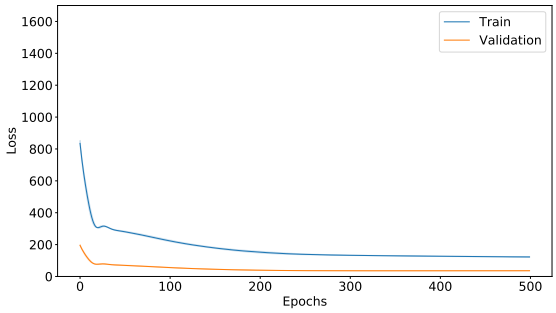
(a) HPRD



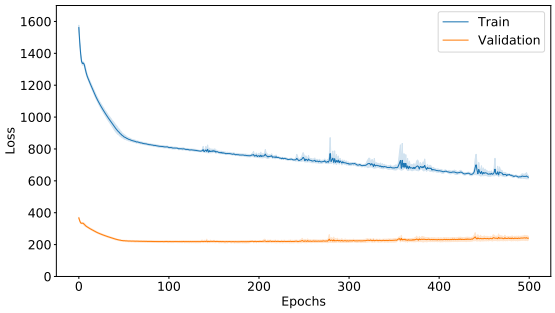
(b) MULTINET



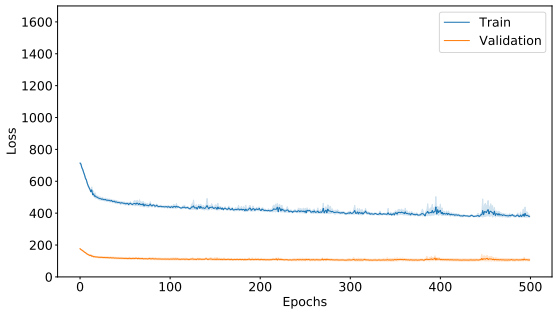
(c) IREF



(d) CPDB



(e) PCNET



(f) STRING

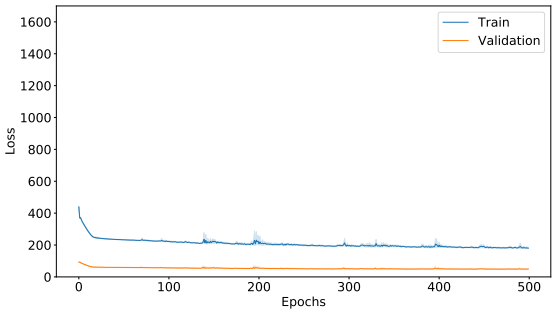
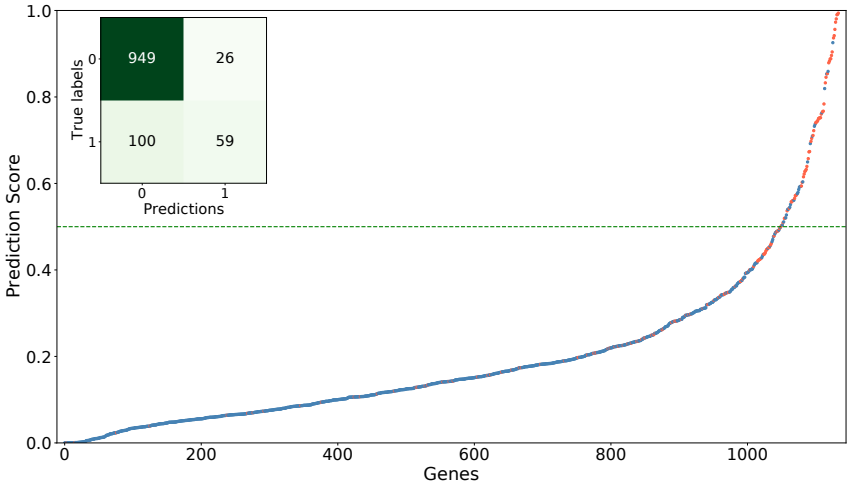
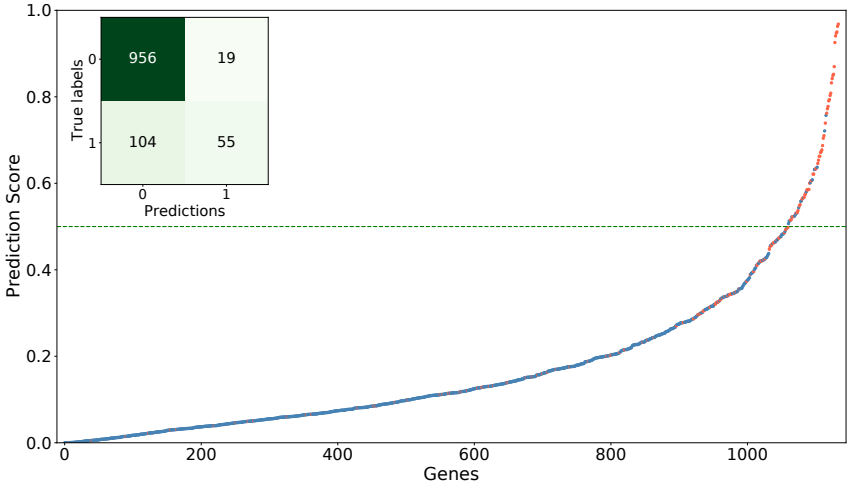


Figure C.5 – Prediction of GCN models trained with (a) HPRD, (b) MULTINET, and (c) IREF networks in the test set.

(a) HPRD



(b) MULTINET



(c) IREF

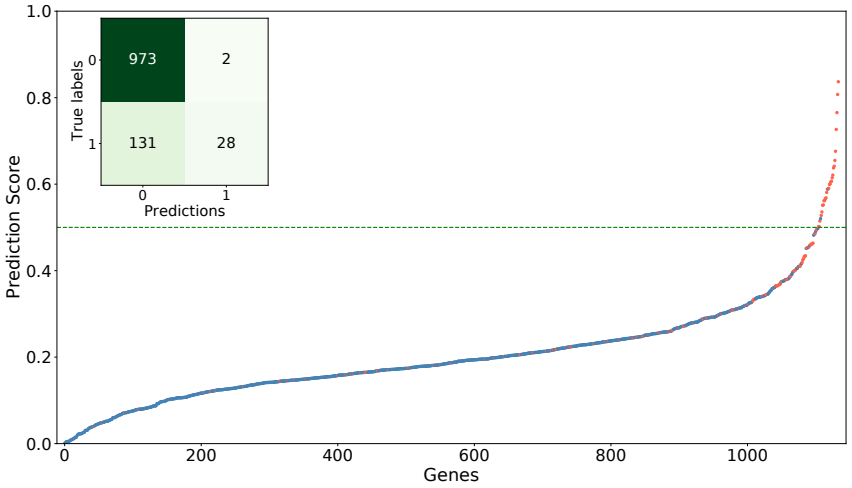
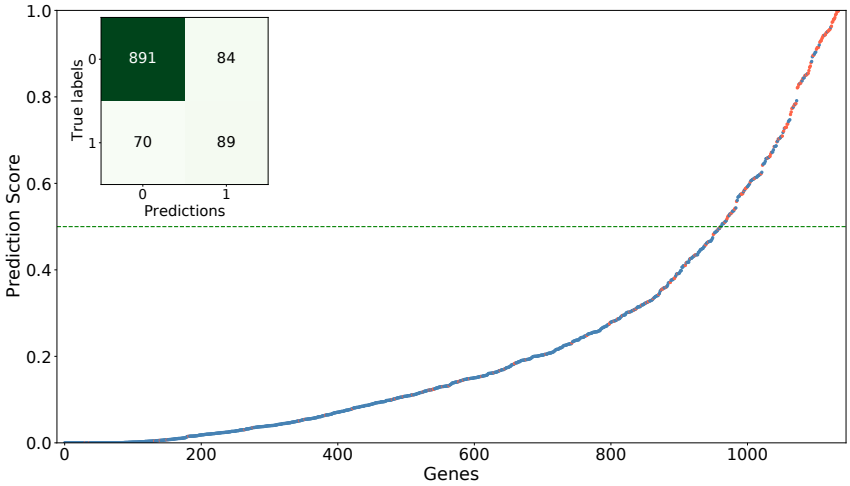


Figure C.6 – Prediction of GCN models trained with (a) CPDB and (b) PCNET networks in the test set.

(a) CPDB



(b) PCNET

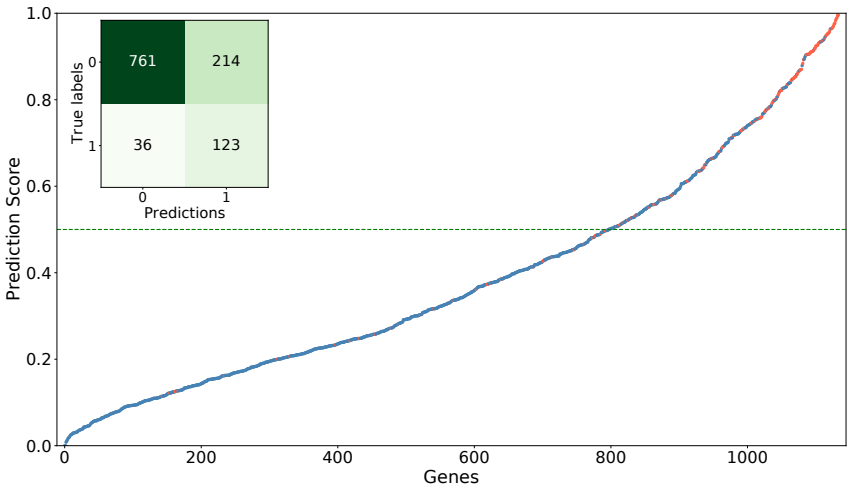
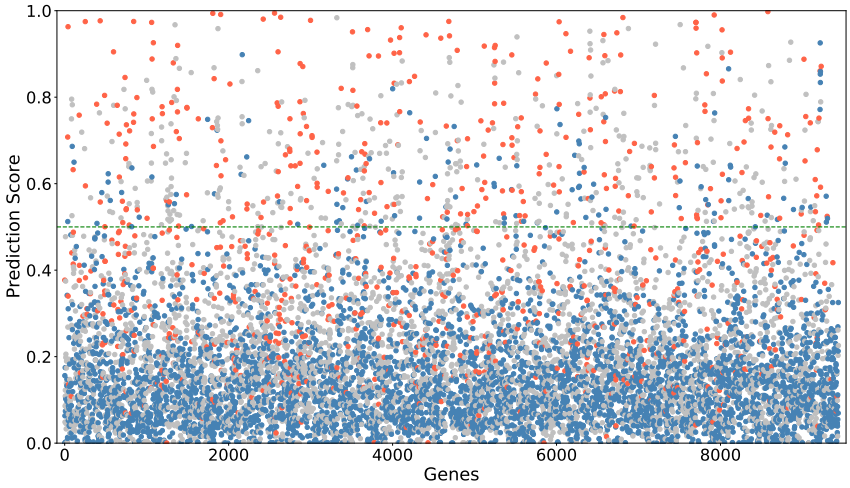
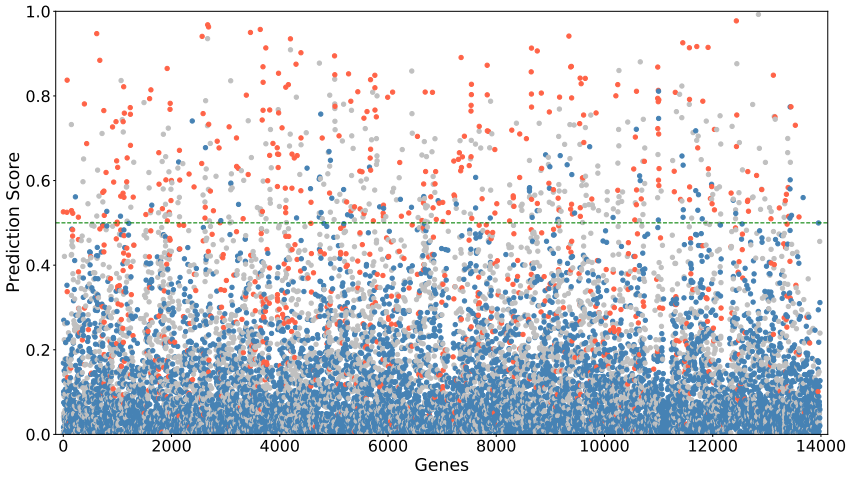


Figure C.7 – Cancer driver genes prediction of (a) HPRD, (b) MULTINET and (c) IREF networks

(a) HPRD



(b) MULTINET



(c) IREF

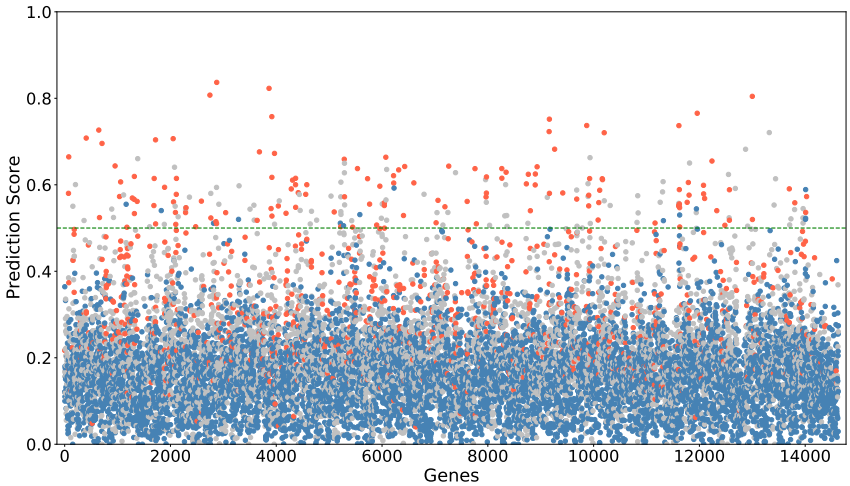
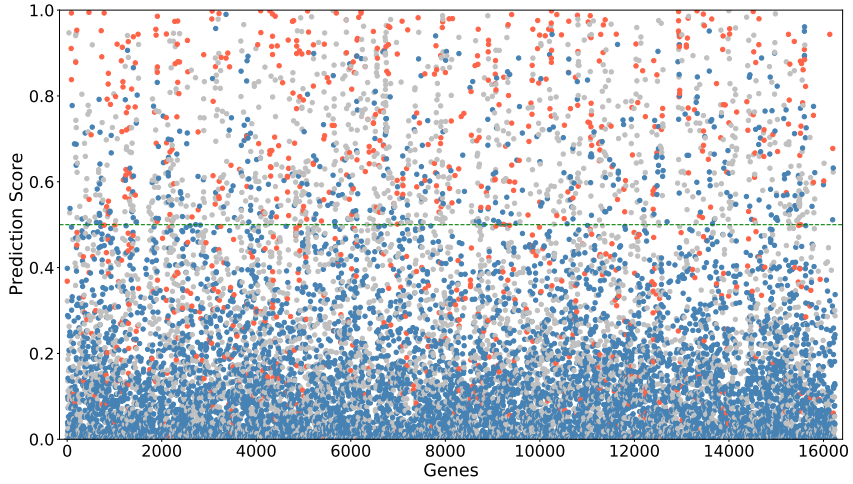


Figure C.8 – Cancer driver genes prediction of (a) CPDB and (b) PCNET networks

(a) CPDB



(b) PCNET

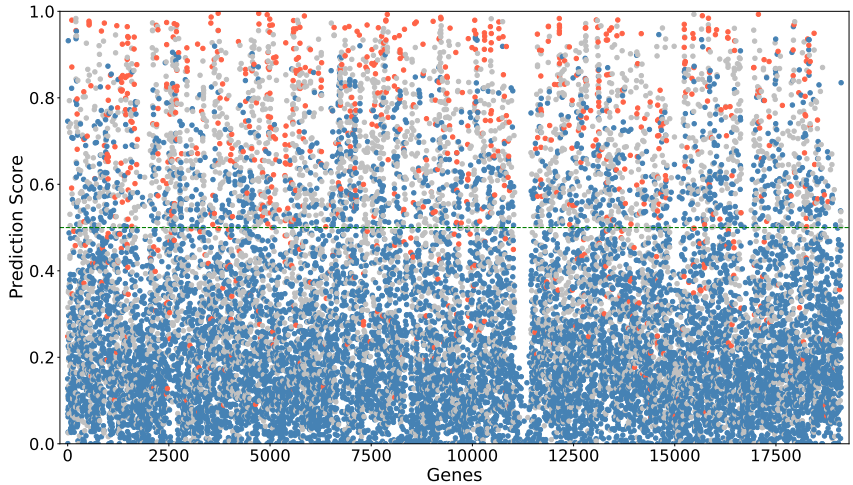


Table C.4 – List of candidate genes unanimously predicted as driver or passenger gene

Candidates predicted unanimously as driver gene	Candidates predicted unanimously as passenger gene
XRCC6	DISP3
SP1	DLST
TNFRSF1A	SLC22A5
KAT2B	CASKIN1
HMGB1	SRSF9
TTN	NPHP1
GRB2	DHDDS
VIM	APOBEC2
PTK2B	TMEM43
XRCC5	CDHR2
CHUK	YY1AP1
GHR	LHX4
SHC1	CALCR
PRKCA	C7
HDAC3	RHO
UBE2I	PEX19
KAT2A	CXADR
ACTB	GSTM4
GATA4	RBM7
CDK2	CCN3
ESR2	TPD52
FOS	GCKR
RUNX2	MPZ
PIAS1	PCBP2
RELA	GSTM2
JUP	ZNF41
NCK1	PLRG1
FN1	IDH3A
PTK2	CPNE1
NR3C1	APOL1
HDAC2	BMF
CTNNA1	ACADM
CDK1	MGST2

APPENDIX D — RESUMO EXPANDIDO

Identificar os genes e mutações específicas responsáveis pelo crescimento de tumores é um passo crítico para avançar nossa compreensão do câncer e explorar novos caminhos para diagnóstico e tratamento. Apesar da abundância de dados genômicos disponíveis, identificar com precisão genes causadores de câncer (CDG), entre milhões de possíveis mutações somáticas, apresenta um desafio significativo. Abordagens computacionais tornaram-se cada vez mais úteis na identificação de padrões genômicos ligados a fatores causadores de câncer e na construção de modelos para prever eventos causadores. O aprendizado profundo, um subcampo especializado do aprendizado de máquina, provou ser uma ferramenta valiosa em bioinformática, dada sua capacidade de criar e treinar arquiteturas semelhantes à estruturas biológicas. Coletivamente, essas duas subáreas oferecem oportunidades excepcionais para preencher as lacunas remanescentes no campo.

Nosso estudo começou com uma revisão abrangente dos métodos computacionais que empregam técnicas de aprendizado de máquina para detectar mutações e genes causadores de câncer. Esta revisão visa fornecer uma perspectiva integrada da vasta paisagem de dados e algorítmica que envolve esse desafio científico. Examinamos como as soluções anteriores exploraram as interações entre diferentes tipos de dados e algoritmos de aprendizado de máquina e também descrevemos as restrições analíticas atuais. Nosso objetivo é mapear os avanços significativos e detalhes metodológicos de pesquisas anteriores nesta área e identificar direções promissoras para trabalhos futuros.

Com base em nossa revisão de pesquisas anteriores sobre a previsão de CDGs, identificamos um desafio na área de aprendizado de algoritmos baseados em grafos, como as *graph neural networks* (GNN). Portanto, o objetivo principal deste estudo é investigar o potencial de GNNs na previsão de genes causadores de câncer e construir um modelo que possa efetivamente utilizar informações de redes de interação proteína-proteína (PPI) e dados multi-ômicos para identificar padrões relacionados ao CDGs. Para criar um modelo preditivo baseado em grafo a fim de prever CDGs, realizamos uma análise completa de vários aspectos importantes relacionados ao desenvolvimento do modelo. Isso incluiu a coleta de seis redes PPI distintas e dados multi-ômicos, abrangendo 16 tipos diferentes de câncer e quatro níveis de dados ômicos, que foram utilizados como features de nó. Também avaliamos o desempenho de diferentes algoritmos GNNs e estratégias para resolver problemas de desequilíbrio de classe. Além disso, propomos técnicas baseadas em *ensemble* para aumentar a capacidade preditiva de modelos individuais.

Conduzimos uma série de experimentos para obter informações sobre o desempenho geral dos algoritmos GNN na previsão de genes condutores. Além de comparar três modelos baseados em grafos para três redes PPI distintas, também comparamos diferentes estratégias para definir vetores de features de instância e mitigar o desequilíbrio de classes. Além disso, comparamos os algoritmos GNN com quatro algoritmos tradicionais de aprendizado de máquina.

Nossos resultados indicaram que entre as GNNs testadas, o algoritmo *graph convolutional network* (GCN) demonstrou os resultados mais promissores, exibindo desempenho consistente e robusto em todos os cenários avaliados. Nossas descobertas também indicam que os algoritmos tradicionais de aprendizado de máquina são altamente sensíveis ao tipo de dados ômicos usados e tendem a ter um desempenho melhor ao incorporar informações de mutação. Além disso, observamos que medidas de centralidade de nó calculadas a partir de redes PPI podem ser altamente informativas e vantajosas para a tarefa de previsão, mesmo para algoritmos que não são baseados em grafos. Com base nessas descobertas, selecionamos o algoritmo GCN para um estudo mais abrangente e como o foco principal de nosso estudo para prever CDGs, dada sua eficácia em diferentes cenários.

Por fim, conduzimos uma investigação completa para determinar as combinações ideais de hiperparâmetros para todas as seis redes selecionadas. Em seguida, relatamos o desempenho do treinamento usando um conjunto de teste compartilhado em todas as redes, permitindo-nos realizar diferentes estratégias de comparação. Primeiramente, comparamos uma rede que agrega todas as redes coletadas com seus respectivos desempenhos individuais. E posteriormente utilizamos duas estratégias de aprendizado *ensemble* que consiste em agregação dos rótulos preditos, por votação majoritária, e por combinação de probabilidades de predição, através da média. A rede unificada não superou as redes individuais, e as estratégias de aprendizado *ensemble* se mostraram eficazes ao superar o desempenho de todas as redes coletadas.

Nas predições finais, apresentamos uma lista de genes não rotulados que foram previstos como potenciais CDGs, que foram unanimemente preditos por todas as seis redes, a rede unificada e a estratégia de aprendizagem *ensemble*. Embora inúmeros desafios permaneçam, nossos resultados promissores sugerem direções valiosas para incorporar centralidades de rede como informações cruciais e a implementação de aprendizado *ensemble* em GNNs.