Machine Learning for Cancer Survival Prediction

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

Anna Kristina Thedinga

Berlin 2024

Erstgutachter: Prof. Dr. Martin Vingron
Zweitgutachter: Prof. Dr. Tobias Scheffer

Tag der Disputation: 05. Juli 2024

Preface

Contributions and Publications

This dissertation is based on a project that had the aim of investigating machine learning for cancer survival prediction. Based on extensive experimentation and fruitful discussions, two different aspects of machine learning for cancer survival prediction were explored. The first aspect was to develop a machine learning approach based on XGBoost tree ensemble learning and network propagation to predict cancer survival and to derive the biological plausibility of the survival prediction method by using network propagation. The idea for this approach arose from valuable discussions with my supervisor Ralf Herwig. The approach, in particular single-cohort and pan-cancer survival prediction trained on gene expression data and the identification and analysis of a pan-cancer survival network, was published in iScience[170] and a corresponding protocol detailing the steps necessary to reproduce the results from the first publication was published in STAR Protocols[169]. In this dissertation, the approach was further extended beyond the published version, in particular by integrating additional molecular data types and by considering the tumor status of patients as additional information. The second aspect of this work was the exploration of transfer learning for cancer survival prediction, where we explored the transferability of knowledge learned by neural networks for different tasks to cancer survival prediction. The idea for this part of the work developed through discussions with my second reviewer and thesis advisory committee (TAC) member Tobias Scheffer and my supervisor Ralf Herwig. This part of the dissertation is unpublished.

In the first part of this work, network propagation was used to add biological plausibility to the cancer survival prediction method. Network propagation leverages prior knowledge from a network, commonly a protein-protein interaction network, to gain insights into underlying biological mechanisms[43] and has been successfully used on a variety of biological problems. Recently, I performed network propagation on time-resolved gene expression profiles of *Leishmania major* infected bone marrow-derived macrophages from mice that were susceptible or resistant to the disease. This contributed to a publication, which investigated how host M-CSF-induced gene expression affects the immune response to *Leishmania major* infection and was published in Frontiers in Immunology[20]. In addition, I contributed to a paper that is currently in revision at Nature Communications by performing network propagation. In this paper, the time-resolved insulin-regulated phosphoproteome was analyzed to gain a better understanding of insulin intracellular signaling.

Furthermore, I participated in the collaborative ML-Med project, which had the goal of developing a machine learning approach for drug sensitivity prediction. Results from this project were published in NAR Genomics and Bioinformatics[138] and Cancers[139]. The first publication[138] proposed a drug sensitivity prediction method that uses a ranking loss to identify the most effective anti-cancer drugs for unseen cancer cell lines or the most sensitive cancer cell lines for new drugs, whereas the second publication[139] introduced an approach that uses transfer learning to transfer knowledge learned from cancer drug sensitivity prediction on in vitro data to patient-derived data, such as patient-derived cell cultures, xenografts, and organoids.

Acknowledgements

First and foremost, I would like to thank my supervisor Ralf Herwig for giving me the opportunity to work as a PhD candidate in his group at the Max Planck Institute for Molecular Genetics in Berlin and for his supervision and support. I would furthermore like to thank my supervisor Martin Vingron for accepting me into the Department of Computational Biology, supporting and advising my work, and reviewing this dissertation. Many thanks also to Tobias Scheffer for his helpful advice and constructive feedback and for reviewing this dissertation, and to his group members Paul Prasse and Pedro Alonso Campana for fruitful discussions. Lastly, I would like to thank Marie-Laure Yaspo for being part of my thesis advisory committee (TAC) and for her advice during my TAC meetings.

I am grateful for the opportunity to be part of the International Max Planck Research School for Biology And Computation (IMPRS-BAC), which supported me in my work, gave me the opportunity to participate in scientific and soft skill seminars, and provided many opportunities to meet fellow PhD students. Special thanks go to the PhD coordinators Kirsten Kelleher and Anne-Dominique Gindrat for organizing so many nice events and workshops and for helping with all PhD-related issues. I would also like to thank my former and current group members Matthias Lienhard, Gal Barel, Abeera Fatima, and Yalan Bi, as well as all the other members of the Computational Biology department for welcoming me into the group and the department, for the interesting scientific and non-scientific discussions in the office, at seminars and retreats, and for the fun times that we spent together.

Moreover, I would profoundly like to thank Andreas Buchheit and Jan Thedinga for proofreading this dissertation and giving me much appreciated feedback.

Finally, I want to express my deepest gratitude to my partner Steffen and my whole family for always supporting me throughout my PhD, for encouraging me in difficult times, and for always being there for me.

Contents

List of Figures

List of Tables

List of Abbreviations

Adam	adaptive moment estimation
AP-MS	affinity purification mass spectrometry
ATP	adenosine triphosphate
CCLE	Cancer Cell Line Encyclopedia
cDNA	complementary DNA
CGCI	Cancer Genome Characterization Initiative
CHF	cumulative hazard function
CNV	copy number variation
CPDB	ConsensusPathDB
CPTAC	Clinical Proteomic Tumor Analysis Consortium
DNA	deoxyribonucleic acid
FDR	false discovery rate
fMRI	functional MRI
FPKM	fragments per kilobase of transcript per million fragments mapped
FWER	family-wise error rate
GDC	Genomic Data Commons
GDSC	Genomics of Drug Sensitivity in Cancer Database
GR	glucocorticoid receptor
GTEx	Genotype-Tissue Expression
IPA	Ingenuity Pathway Analysis
KEGG	Kyoto Encyclopedia of Genes and Genomes
lncRNA	long non-coding RNA
MAF	mutation annotation format
MKL	multiple-kernel learning
MMP	matrix metalloprotease
MRI	magnetic resonance imaging
mRNA	messenger RNA
mTOR	mechanistic target of rapamycin
NCG	Network of Cancer Genes
NGS	next-generation sequencing
ORA	over-representation analysis
PCA	principal component analysis

PID	Pathway Interaction Database
PPI	protein-protein interaction
RF	random forest
RNA	ribonucleic acid
RNA-seq	RNA-sequencing
ROI	R Optimization Infrastructure
RPPA	reverse-phase protein array
rRNA	ribosomal RNA
RWR	random walk with restart
SEM	standard error of the mean
SNP	single nucleotide polymorphism
SNV	single nucleotide variant
SV	structural variant
SVM	support vector machine
TCGA	The Cancer Genome Atlas
TIL	tumor-infiltrating lymphocyte
TME	tumor microenvironment
TPE	Tree-structured Parzen Estimator
TPM	transcripts per million
tRNA	transfer RNA
VAE	variational autoencoder
XGBoost	extreme gradient boosting
Y2H	yeast two-hybrid

1

Introduction

This chapter aims to motivate the work and to explain the research objective of this dissertation. In addition, a brief outline of the following chapters of this dissertation is given.

1.1 Motivation

With approximately 10 million deaths in 2020, cancer is one of the leading causes of death worldwide[55]. In Germany alone, 231,533 people have died from cancer in 2022, accounting for 22.4% of all deaths and making it the second leading cause of death[163] in the country. To reduce these numbers and improve the survival of cancer patients is the primary goal of cancer therapy. However, therapies with the aim of complete remission are often relatively aggressive and can be accompanied by severe side effects. When a patient has a poor prognosis and complete remission is not possible or at least highly unlikely, therapy can also have the goal of merely prolonging life or improving life quality for the remainder of the patient's life[27]. Accordingly, the choice of therapy is heavily influenced by patient prognosis. Computational models for cancer survival prediction can help estimate prognosis and quantify individual patient risk to guide therapy decisions. These models are typically based on either clinical data such as age or tumor stage, molecular data such as mutations or gene expression, or imaging data such as hematoxylin and eosin (H&E, where hematoxylin stains cell nuclei in the tissue slide blue and eosin stains the extracellular matrix and cytoplasm pink[28]) stained whole slide images (WSIs) or magnetic resonance images (MRI) of the tumor and use machine learning to detect relationships between these data and the survival of the corresponding cancer patients.

1.2 Research Objective

The research objective of this dissertation is twofold. The first goal is the development of a machine learning approach for cancer survival prediction based on molecular patient data and the investigation of the approach's biological plausibility. The second goal is to answer the question of whether cancer survival prediction can be improved by transferring knowledge from pre-trained machine learning models through transfer learning.

To achieve the first goal, we propose a survival prediction method that applies XGBoost tree ensemble learning on molecular data such as gene expression to predict cancer survival for 25 cancer types from The Cancer Genome Atlas (TCGA). The survival prediction method is evaluated for cancer-type-specific training, where a separate prediction model is trained for each cancer type, and pan-cancer training, where molecular data from patients of all 25 cancer types is combined to train a shared survival prediction model. We show that pan-cancer training yields improved prediction performance over cancer-type-specific training and gene expression is the most informative of the evaluated molecular datatypes, which include mutation, copy number variation, gene expression, and protein expression data. The biological plausibility of the proposed approach is investigated by applying network propagation on feature importance scores learned by the pan-cancer survival prediction model trained on gene expression data. This way, a pan-cancer survival network is inferred and further analyzed with respect to biological pathways and mechanisms.

To answer the second research question, we explore transfer learning, where a machine learning model is pre-trained on a source domain and then knowledge from the pre-trained model is transferred to a target domain to improve prediction performance on this target domain. We use neural networks as the machine learning model of choice for most transfer learning tasks and investigate two different settings of transfer learning: In the first setting, a neural network for pan-cancer survival prediction is pre-trained on 25 different cancer types from TCGA and knowledge learned by this model is transferred to survival prediction on smaller, independent cancer datasets. In the second setting, neural networks are trained on data from the Genotype-Tissue Expression (GTEx) project for auxiliary tasks like tissue type classification and age prediction. Knowledge from the pre-trained models is then transferred to the task of cancer survival prediction on the TCGA dataset and the effect of this knowledge transfer is evaluated.

1.3 Outline of This Dissertation

This dissertation is structured as follows:

In Chapter 2, the biological terminology and concepts necessary to understand this dissertation are outlined.

Chapter 3 describes the fundamental mathematical principles and key methodology underlying the work of this dissertation.

Chapter 4 briefly introduces existing work within the thematic scope of this dissertation and explains its relevance for this dissertation. The literature introduced in this chapter includes work from the field of cancer survival prediction as well as relevant publications related to XGBoost, which is a key machine learning method of this dissertation (cf. Chapter 5), and applications of transfer learning, which is further explored with regard to cancer survival prediction in Chapter 6.

In Chapter 5, we introduce our approach for the identification of a pan-cancer survival network with gradient tree boosting and network propagation. The approach has two key steps: In the first step, an XGBoost machine learning method is trained to predict cancer survival from patient molecular data. In the second step, the biological plausibility of this survival prediction method is investigated by applying network propagation to the genes identified as important survival features in the first step. This way, a pan-cancer survival prediction network is identified, which is then further analyzed with respect to biological mechanisms and molecular pathways.

In Chapter 6, transfer learning as a means to improve cancer survival prediction is explored. To this end, different transfer learning scenarios are evaluated, in which neural networks are pre-trained on different tasks, and then knowledge from these pre-trained models is transferred to the task of cancer survival prediction to improve prediction performance.

The final Chapter 7 summarizes the results of this dissertation and gives an outlook on how challenges of cancer survival prediction on currently available cancer datasets could be resolved and how cancer survival prediction could be further advanced in the future if more comprehensive data becomes available. In addition, a brief conclusion is provided.

2

Biological Background

This chapter introduces the biological background knowledge that is fundamental for understanding the following chapters.

2.1 Cancer

Cancer describes a group of diseases that can affect any body part and is characterized by the rapid creation of abnormally growing cells [185]. These abnormal cells can also spread to other body parts and initiate new tumors, a process called metastasis [185,68].

2.1.1 The Hallmarks of Cancer

Despite cancer being a heterogeneous group of diseases rather than a single disease with uniform phenotype, several characteristics are shared between cancer types. Hanahan and Weinberg [67] summarized these characteristics as “The Hallmarks of Cancer”, which originally comprised six and were later updated to eight [68] functional capabilities shared between all types of cancer cells (Figure 2.1). These hallmarks are acquired by the cells during their development from a healthy normal cell to a cancer cell and encompass the following capabilities: Sustaining proliferative signaling, evading growth suppressors, resisting cell death, enabling replicative immortality, inducing angiogenesis, activating invasion and metastasis, reprogramming cellular metabolism, and avoiding immune destruction. In the following paragraphs, these eight hallmark capabilities will be explained in more detail.

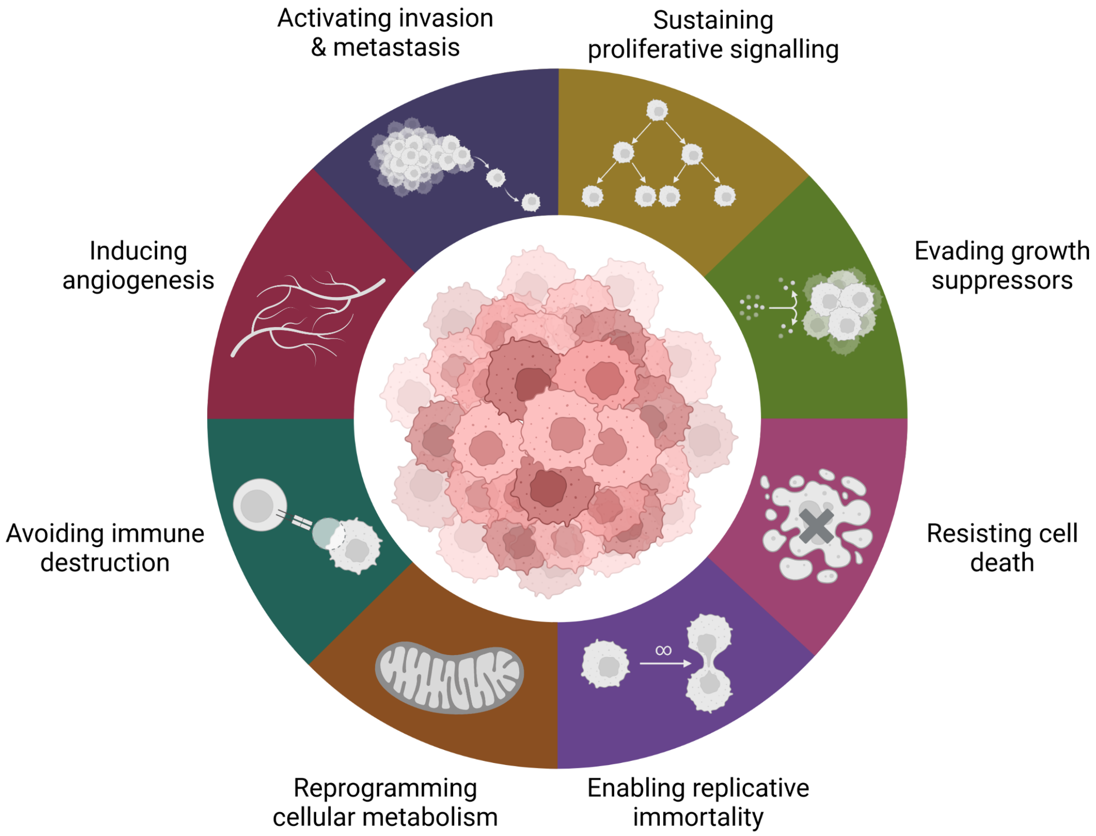

Activating invasion & metastasis
Sustaining proliferative signalling
Inducing angiogenesis
Evading growth suppressors
Avoiding immune destruction
Resisting cell death
∞
Reprogramming cellular metabolism
Enabling replicative immortality

Figure 2.1: Hallmarks of cancer. The eight hallmarks of cancer proposed by Hanahan and Weinberg in 2011[68] summarize the functional characteristics shared by all cancer cells. Inspired by[68] and created with `BioRender.com`.

Sustaining Proliferative Signaling

While normal, non-cancerous tissues preserve homeostasis of cell number and thus maintain tissue function and architecture by controlling the production and release of growth-promoting signals, cancer cells deregulate these signals and can thus sustain proliferation[68].

Evading Growth Suppressors

In addition to sustaining proliferative signaling, cancer cells can also evade growth suppressors, which in normal tissues limit cell growth and proliferation[68]. Many of the genes encoding proteins involved in the negative regulation of cell proliferation, such as *TP53*, have been identified as tumor suppressors that are frequently inactivated in human cancers.

Resisting Cell Death

Furthermore, cancer cells can resist cell death by limiting or circumventing apoptosis[68]. In normal cells, apoptosis is triggered by extracellular and intracellular signals as a response to physiologic stress such as abnormal signaling or DNA damage. Cancer cells however can evade apoptosis, for instance by loss of TP53 tumor suppressor function, an increase of the expression of antiapoptotic regulators or survival signals, or by downregulation of proapoptotic factors.

Enabling Replicative Immortality

Another hallmark of cancer is the enabling of replicative immortality[68]. To form tumors, cancer cells need the ability to replicate without the normal limitations that restrict the number of growth and division cycles a cell can go through. Once the cell reaches this limit, senescence, which is a viable but nonproliferative state of the cell, or apoptosis, which leads to the death of the cell, is induced. Cancer cells can reach a state of immortalization with unlimited replicative potential by maintaining long telomeres that protect the chromosome ends, while in normal cells, the telomeres shorten with every cell cycle and eventually become so short that they lose their protective function and senescence or apoptosis is triggered.

Reprogramming Cellular Metabolism

To fuel cell growth and replication, the cancer cells also adjust their energy metabolism[68]. Normal cells under aerobic conditions use glycolysis in the cytosol to convert glucose to pyruvate and then transmit the pyruvate to the mitochondria, which use oxidative phosphorylation to produce adenosine triphosphate (ATP), thereby consuming oxygen and producing carbon dioxide. Under anaerobic conditions, normal cells limit their energy metabolism largely to glycolysis to produce more lactate via fermentation and only transmit little pyruvate to the mitochondria. In contrast to normal cells, cancer cells reprogram their energy metabolism and favor producing lactate over transmitting pyruvate to the mitochondria and producing ATP even under aerobic conditions, a phenomenon that is called the "Warburg effect" after its discoverer Otto Warburg. One hypothesis that could explain the Warburg effect is that increased glycolysis facilitates the incorporation of nutrients into biomass, which is needed to assemble new cells during replication, because when glycolysis is increased and only a little pyruvate is transmitted to the mitochondria, the glycolytic intermediates can be used for the biosynthesis of macromolecules like nucleotides, amino acids, and lipids that are required for cell proliferation and replication, instead of mainly producing ATP[68,81].

Avoiding Immune Destruction

Tumors can only grow if the cancer cells are able to avoid destruction through the immune system[68]. According to the well-established theory of immune surveillance, the immune system constantly monitors tissues and cells and eliminates most cancer cells. Thus, the immune system functions as a barrier to tumor formation and progression. However, some cancer cells can evade being detected by the immune system or limit immune destruction and ultimately form tumors.

Inducing Angiogenesis

When a tumor forms by replication of cancer cells, it needs to be supplied with nutrients and oxygen. Moreover, the metabolic waste and carbon dioxide produced by the cancer cells must be transported away from the tumor[68]. To this end, the tumor induces a process called angiogenesis. During angiogenesis, tumor-associated neovasculature that helps to sustain tumor growth is generated by the formation of new blood vessels from existing ones. However, this newly formed vasculature is typically aberrant with distorted and enlarged vessels, vessels branching in a convoluted and excessive manner, capillary sprouting prematurely, leakiness of vessels, erratic blood flow, and more.

Activating Invasion and Metastasis

Carcinomas, cancers arising from epithelial tissues, can, as they progress, invade tissue in the vicinity of the tumor or metastasize to distant tissues[68]. Invasion and metastasis can be viewed as a multi-step process, starting with the local invasion of cancer cells, which then also invade nearby blood and lymphatic vessels and transit through the lymphatic and hematogeneous systems to distant tissues, where they escape from the lumina of the vessels into the parenchyma of the tissue and form small nodules of cancer cells called micrometastases. As a last step of this invasion-metastasis cascade, these micrometastases then grow into macroscopic tumors, a process that is called colonization.

2.1.2 Cancer Survival

Metastasis is a common cause of cancer death[185]. For instance, two-thirds (66.7%) of deaths from solid tumor cancers registered in the Cancer Registry of Norway in 2015 had metastases as a contributing cause of death[52]. However, there was substantial variation between different cancer types: While for nose sinus and testicular cancers, metastasis was present in 100% of registered cancer deaths, only 9.3% of central nervous system cancer deaths were associated with metastasis.

This heterogeneity between different cancer types is also reflected in the overall death rates in different cancers. While the 5-year relative survival across all cancer types was 68% for patients diagnosed between 2012 and 2018 in the United States of America, it was substantially lower in pancreas (12%), liver (21%), and esophagus (21%) cancers. For other cancer types such as melanoma, testis, prostate, and thyroid cancers, on the other hand, only a small proportion of patients died from the disease within the first five years after diagnosis and 5-year relative survival rates were as high as 94%, 95%, 97% and 98%, respectively[158].

Worldwide, cancer is one of the leading causes of premature death, i.e. death between ages 30 to 69, according to the World Cancer Report 2020[57]. In 134 of 183 countries, it is either the first or second leading cause of premature death. In 2016, 40.5 million people worldwide died from noncommunicable diseases, accounting for 72% of global deaths and 15.2 million of these deaths were premature. Of these 15.2 million premature deaths from noncommunicable diseases, 4.5 million (29.8%) were attributed to cancer, making cancer the second leading cause of premature death worldwide.

2.2 Omics—Different Aspects of Molecular Biology

Cancer cells and tumors have abnormal function and structure compared to normal cells and tissues, as reflected in "The Hallmarks of Cancer"[68]. To identify, quantify, and characterize the biological molecules involved in the function and structure of cells, tissues, and organisms is the objective of a scientific field called omics[175]. The term omics summarizes different aspects of molecular biology that end with the suffix *-omics*, including gen*omics*, transcript*omics*, and prote*omics*, among others.

In this section, different types of omics that are relevant to this work will be introduced. However, this section does not intend to give a comprehensive list of all omics types, but focuses only on the omics modalities used in this work for cancer survival prediction.

2.2.1 Genomics

Genomics is concerned with the study of the genome, that is, the entirety of deoxyribonucleic acid (DNA) in an organism[175]. Its aim is to identify genetic variants[175,168], such as short insertions and deletions (indels), single nucleotide variants (SNVs) or single nucleotide polymorphisms (SNPs), which are SNVs with an abundance of at least 1% in the population[24], but also more complex structural variants (SVs), which are DNA variations larger than 50 base pairs and are thought to account for 50–95% of the sequence variation between human samples and the reference genome[42]. To identify genetic variants in the DNA, the genome or parts of it are sequenced. The development of next-generation sequencing (NGS) technologies has made it possible to sequence entire genomes and thus detect genetic variants on a genome-wide scale, which was previously not possible in a reasonable amount of time using the Sanger sequencing technology[14]. In NGS, millions of small DNA fragments are sequenced in parallel and then mapped to a reference genome (Figure 2.2). In this way, NGS can identify different types of mutations and other genetic variations, such as base substitutions, and indels, but also more complex variations like large genomic deletions, genome

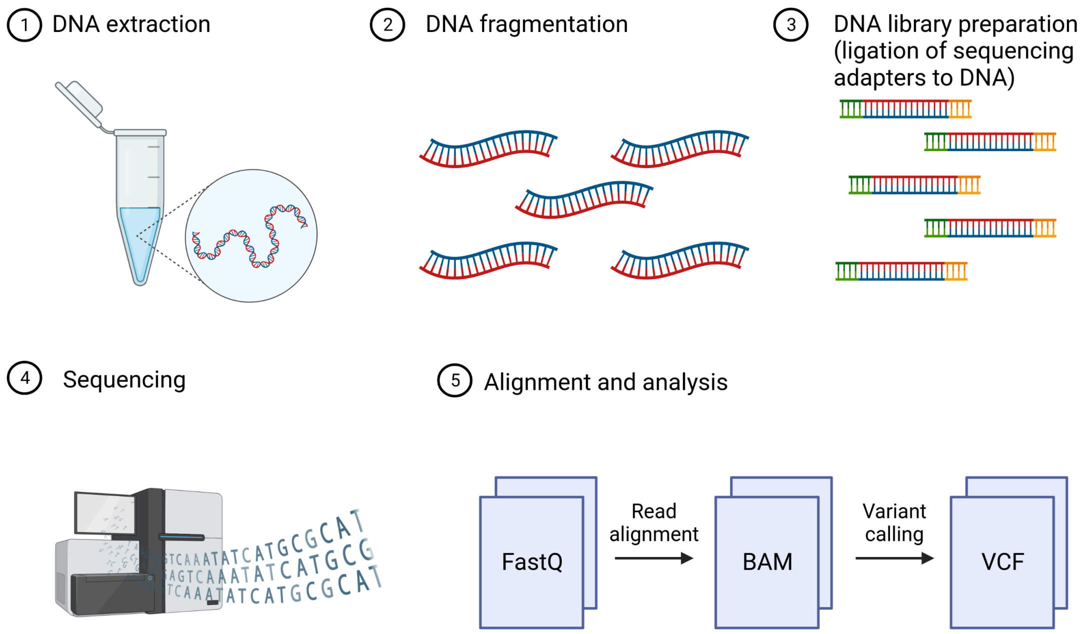


Figure 2.2: Next-generation DNA-sequencing. Illustration of a typical NGS workflow for DNA-sequencing. First, DNA is extracted and fragmented. Next, a sequencing library is prepared by ligating sequencing adapters to both ends of the DNA fragments and the DNA library is sequenced. Finally, the sequenced reads are aligned to a reference genome and genomic variants are called. Created with BioRender.com.

rearrangements, and translocations, while Sanger sequencing is only able to identify substitutions and indels.

Genetic variants are typically stored in the variant call format (VCF)[168], in which each variant is represented by, at minimum, the chromosome it is located on, the exact base position, a reference allele sequence, and at least one alternative allele. Optionally, the variant entry can also contain additional information such as a quality score, for known SNPs a dbSNP identifier, or any other additional information about the variant.

2.2.2 Transcriptomics

Transcriptomics is the study of the transcriptome, which is the total ribonucleic acid (RNA) expressed in a cell type or tissue[175,53]. The transcriptome comprises different types of RNA, including for example messenger RNA (mRNA), transfer RNA (tRNA), ribosomal RNA (rRNA), and long non-coding RNA (lncRNA). According to the central dogma of molecular biology, which was first formulated by Francis Crick in 1958[47] and further explained in 1970[46], the genetic information of a cell is carried by DNA and can be transferred into RNA, which serves as a template for protein (Figure 2.3). However, Crick[47,46] emphasizes that while genetic information can be transferred from nucleic acid (i.e., DNA or RNA) to nucleic acid or from nucleic acid to protein, the transfer of genetic information between proteins or from protein to nucleic acid is not possible[47,46].

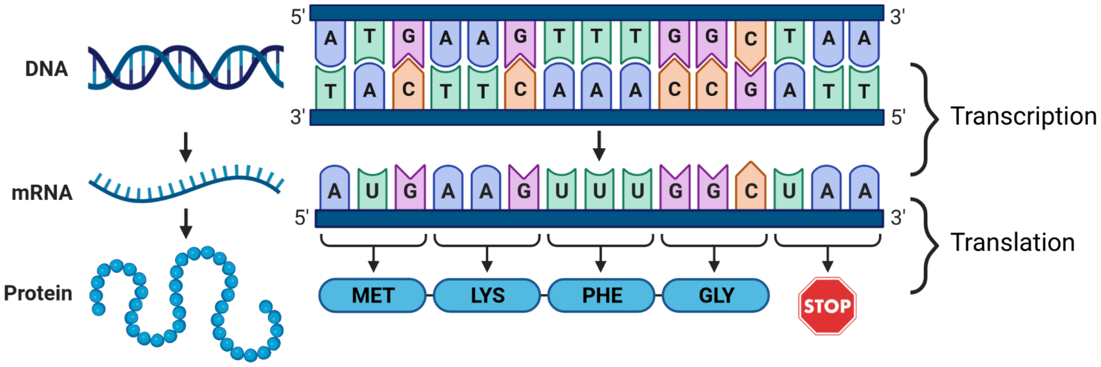

DNA
mRNA
Protein
5'
A T G A A G T T T G G C T A A
3'
T A C T T C A A A C C G A T T
3'
5'
Transcription
A U G A A G U U U G G C U A A
5'
3'
Translation
MET
LYS
PHE
GLY
STOP

Figure 2.3: The central dogma of molecular biology. According to the central dogma of molecular biology[47,46], DNA contains the genetic information of the cell and can be transcribed into mRNA, which in turn can be translated into an amino acid sequence that constitutes a protein. Created with `BioRender.com`.

However, since then many more types of RNA besides the protein-coding mRNA have been discovered. In fact, while the vast majority (more than 90%) of the human genome is transcribed into RNA[72], only a small fraction (less than 3%) of this RNA is translated into protein[110]. While some types of non-coding RNA, such as tRNA and rRNA, fulfill infrastructural roles, others like lncRNA have gene regulatory functions[110]. RNA microarrays and the NGS technology RNA-sequencing (RNA-seq) are the two principal methods for the quantification of gene expression[119]. The microarray technology is based on probes—short nucleotide oligomers complementary to the RNA transcripts—that are fixed to a solid substrate such as a glass array. Gene expression is quantified by measuring the abundance of fluorescently labeled transcripts to the microarray, which can be detected by the intensities of fluorescence at the individual probe locations on the microarray. RNA-seq, on the other hand, uses high-throughput sequencing to quantify gene expression, whereby complementary DNA (cDNA) synthesized from the RNA transcripts is sequenced and gene expression is quantified by counting the number of reads from each transcript (Figure 2.4). More precisely, in RNA-seq, after they have been isolated from tissue, long RNAs are either fragmented into short segments and then converted into cDNA (steps 2 and 3 in Figure 2.4) or first converted into cDNA, which is then fragmented[181]. Next, sequencing adaptors are ligated to each cDNA fragment and the cDNA fragments are sequenced. After sequencing, the resulting sequence reads are mapped to a reference genome or reference transcriptome and the number of reads mapping to each gene are counted. RNA-seq, which has replaced microarrays as the predominant gene expression quantification method by now, has some advantages over microarrays: Firstly, to perform a microarray experiment, it is necessary to know the sequences of the transcripts in advance in order to be able to generate the set of complementary probes, while for RNA-seq, such prior knowledge is not required. Additionally, RNA-seq has a much larger dynamic range than microarrays, which can suffer from signal saturation for highly abundant transcripts[85], and the amount of RNA required as input is much higher for microarrays than for RNA-seq (micrograms vs. nanograms).

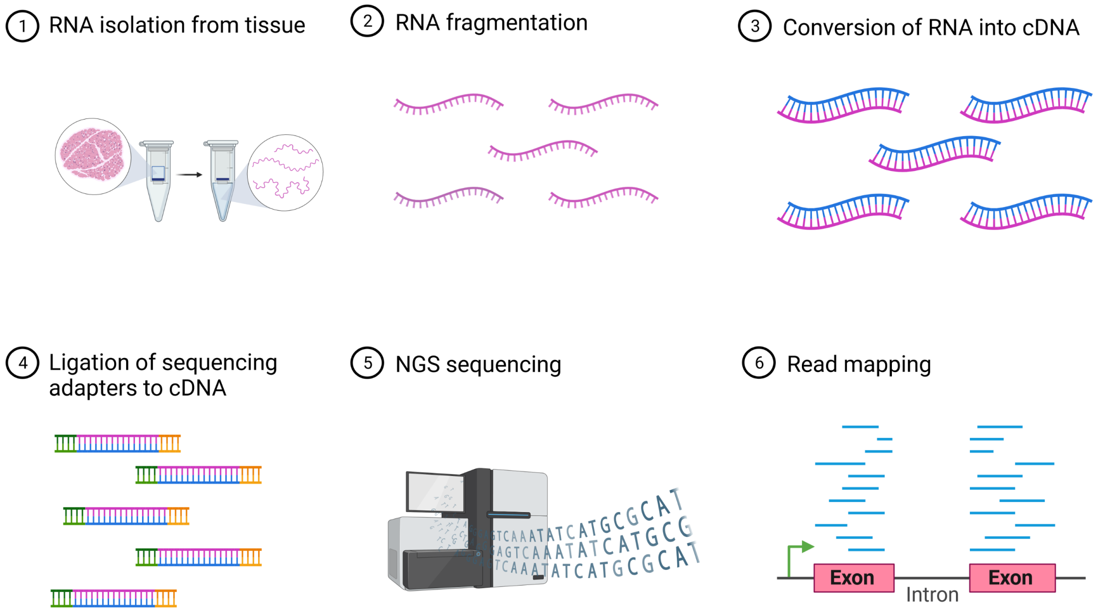


Figure 2.4: RNA-sequencing. Illustration of a typical RNA-seq workflow. RNA is first isolated from tissue and then fragmented into short segments. Next, cDNA is synthesized from the short RNA segments and sequencing adapters are ligated to each cDNA fragment. The cDNA fragments are then sequenced by NGS and the resulting sequence reads are mapped to a reference genome or transcriptome. Created with BioRender.com.

2.2.3 Proteomics

Proteomics focuses on studying the proteome, the set of all proteins in a cell, tissue, or organ at a certain point in time[175]. The aim of proteomics is to identify and quantify proteins and to resolve protein structure and function[9]. Protein expression levels are not only dependent on the expression of their corresponding mRNA, but also on translational regulation and are thus more informative for the characterization of a biological system than genomics and transcriptomics. Methods for the quantification of protein expression include the high-throughput techniques mass spectrometry and protein microarrays. In mass spectrometry, the proteins are first transformed into gas-phase ions and then separated in a mass analyzer by an electric or magnetic field based on their mass-to-charge ratios. To quantify protein expression, the amount of ions with a protein-specific charge ratio is measured[9]. The reverse-phase protein array (RPPA) is a type of protein microarray[9]. In the RPPA method, cell lysates, which contain proteins, are in the first step placed on a slide coated with nitrocellulose. Then the slide is probed with antibodies that are targeted against specific proteins and protein levels are quantified by first using fluorescent, chemiluminescent, and colorimetric assays to detect the antibodies and then comparing the antibody levels with those of a microarray with reference peptides.

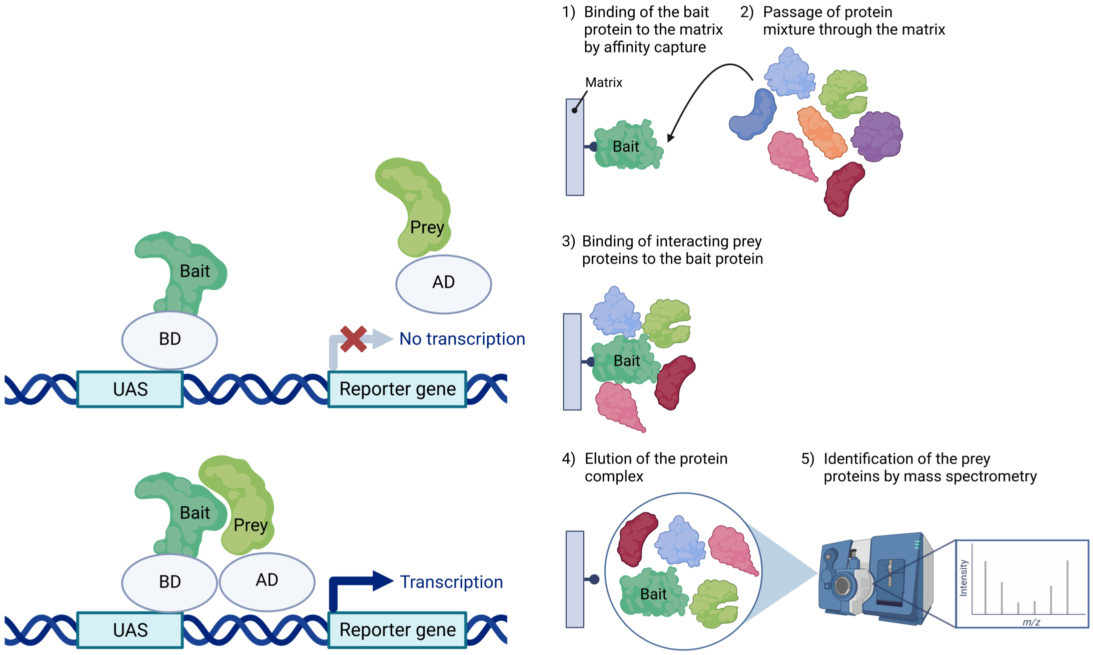
2.3 Protein-Protein Interaction Networks

Proteins are essential for cells to function and are involved in key cellular processes such as metabolism, cell signaling, transport, cellular decision making, and cellular organization[22]. These protein functions are mediated by molecular interactions, including protein-protein interactions (PPIs), where proteins physically interact with each other and thus perform functions like environmental sensing, signal transduction, regulation of metabolic and signaling enzymes, conversion of energy into motion, or maintenance of the cell's structural organization. Two common high-throughput methods for identifying protein-protein interactions are yeast two-hybrid (Y2H) and affinity purification mass spectrometry (AP-MS)[102] (Figure 2.5).

In the yeast two-hybrid method, the DNA-binding domain of a transcription factor is attached to a protein of interest, called bait, and the activation domain of the same transcription factor is attached to another protein, called prey, that is potentially interacting with the first protein. To detect whether there is a protein-protein interaction between bait and prey, both proteins are expressed in a yeast cell. If both proteins bind to each other, the two attached transcription factor components form a functional transcription factor, which then activates a reporter gene.

The affinity purification mass spectrometry method, on the other hand, does not use a bait protein paired with a single prey protein, but can simultaneously detect multiple prey proteins interacting with a bait protein. To this end, the bait protein is isolated in a matrix by affinity capture, and a protein mixture containing multiple prey proteins that potentially interact with the bait protein is passed through the matrix. During this passage, proteins that interact with the bait protein are retained in the matrix because they bind to the bait, while other proteins that do not interact with the bait can pass through the matrix. The prey proteins that have bound to the bait and have thus been retained in the matrix can then be identified from their peptide signatures by mass spectrometry.

The entirety of detected PPIs, involving a plethora of different proteins, can be assembled into a PPI network. Commonly, PPI networks are formulated as undirected graphs, in which proteins are represented by nodes and PPIs are represented by edges that connect the nodes of interacting proteins[102]. There are multiple databases providing PPI networks of different completeness and size, some of which extend the experimentally identified PPIs by interactions that are for instance predicted computationally or inferred from homology with other species[142]. Examples of widely used PPI databases are BioGRID[132] and STRING[167],


Bait
Prey
AD
BD
No transcription
UAS
Reporter gene
Bait
Prey
BD
AD
Transcription
UAS
Reporter gene
1) Binding of the bait protein to the matrix by affinity capture
2) Passage of protein mixture through the matrix
Matrix
Bait
3) Binding of interacting prey proteins to the bait protein
Bait
4) Elution of the protein complex
5) Identification of the prey proteins by mass spectrometry
Bait
Intensity
m/z

(a) yeast two-hybrid (Y2H) method.

(b) Affinity purification mass spectrometry method

Figure 2.5: Protein-protein interaction detection methods. **(a)** In the yeast two-hybrid (Y2H) method, the bait protein is attached to the DNA-binding domain (BD) of a transcription factor and the prey protein is attached to the activation domain (AD) of the same transcription factor. The BD binds to the upstream activating sequence (UAS) and if the bait and prey proteins interact, the AD localizes the reporter gene and activates gene expression. Thus, gene expression of the reporter gene serves as a measure of protein-protein interaction between the bait protein and the prey protein. Created with `BioRender.com` **(b)** In affinity purification mass spectrometry (AP-MS), the bait protein is bound to a matrix by affinity capture and a protein mixture is passed through the matrix. Interacting prey proteins bind to the bait protein and are retained in the matrix, while other proteins pass through. The protein complex of proteins bound to the bait protein is then eluted and the prey proteins are identified by mass spectrometry. Created with `BioRender.com`.

but also the meta-database ConsensusPathDB (CPDB)[82,93], which combines PPI data from multiple resources.

2.3.1 ConsensusPathDB

ConsensusPathDB (CPDB)[82,93] is a meta-database containing data on PPIs as well as other types of molecular interactions, such as drug-target and biochemical interactions, for human, mouse, and yeast obtained from over 30 different public interaction databases, including PPI databases such as BioGRID[132]. In addition to molecular interaction data, CPDB also provides molecular pathway gene sets from well-known databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG)[94]. The 2016 version of the CPDB database contained 261,085 human protein interactions, including both binary PPIs and protein complexes, and 4,593 pathway gene sets[82], which increased to 616,304 protein interactions and 5,578 pathway gene sets in the most recent CPDB version from 2022[93]. Binary PPIs in the CPDB database are annotated with a score, which is in the range $[0, 1]$ and indicates the confidence associated with the respective interaction.

2.4 Molecular Pathways

Cellular functions are rarely conferred by single molecules, but rather by sets or pathways of interacting molecules and their respective genes[74,60]. Hartwell et al.[74] define such pathways, which they call "modules", as discrete functional entities, composed of many molecule types, whose function is separable from that of other modules and arises from interactions between the module's components. The molecules comprising a module can be DNA, RNA, proteins, and other small molecules, which in isolation cannot confer the same function as the module they are members of. Rather than defining pathways merely as sets of interacting molecules without further specifying the exact nature of the interactions involved, another definition of molecular pathways comes from a network perspective, where a pathway is viewed as a network of molecular interactions connected by the molecules taking part in these interactions[149]. In this pathway definition, there are different types of interactions, including gene regulation, which comprises transcription and translation, transport of molecules, also known as translocation, reactions involving the conversion of small molecules, PPIs, as well as so-called macroprocesses, whose internal organization is unknown. Furthermore, whole pathways of interacting molecules can be part of other, often more generalized pathways. However, while the network-based definition of a pathway is arguably more biologically precise than the gene set-based definition, for many downstream analyses such as overrepresentation analysis (ORA), the simpler view of a pathway as a set or functional entity of interacting molecules is usually sufficient. There are several public databases providing comprehensive information on molecular pathways. Some of the most prominent pathway databases include KEGG[94], a database of manually curated pathways, Reactome[61], which besides the molecules constituting a pathway also contains information about the relations between these molecules, and Wikipathways[125], a community-driven pathway database project to which experts from different sub-fields of biology can contribute their knowledge.

3

Mathematical Principles and Methodological Background

This chapter introduces some basic mathematical principles that lay the foundation for this dissertation as well as the key statistical methods and algorithms that are used in this work.

3.1 Mathematical Notation

Firstly, we will introduce some basic mathematical notation used throughout this dissertation. Natural numbers are denoted by $\mathbb{N} = 0, 1, 2, \ldots$ and include 0, while natural numbers excluding 0 are denoted by $\mathbb{N}^+$. Real numbers are denoted by $\mathbb{R}$. The notation $(x_1, x_2, \ldots, x_n)$ denotes a row vector with n elements and the superscript $\top$ indicates the transpose of a matrix or vector, such that $(x_1, x_2, \ldots, x_n)^\top$ would be the corresponding column vector. The notation $[a, b]$ is used for closed intervals, indicating that a and b are both included in the interval, while (a, b) denotes an open interval excluding a and b. The capital letter P denotes probability with $P(A)$ being the probability of an event A and $P(A|B)$ being the conditional probability of event A, given another event B. A binomial coefficient for two integers n and k with $0 \leq k \leq n$ is denoted by $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ and describes the number of ways to choose an unordered subset of k elements from a set of n elements.

3.2 Machine Learning

The term machine learning was coined by Arthur L. Samuel, who in 1959 described the concept of machine learning as "the programming of a digital computer to behave in a way which, if done by human beings or animals, would be described as involving the process of learning"[147]. More specifically, machine learning refers to the capability of a system to acquire knowledge by extracting patterns from raw data rather than relying on hard-coded rules or knowledge[63].

There are two major categories of machine learning—supervised learning and unsupervised learning—and most machine learning algorithms fall into one of these two categories[63]. In supervised learning, each sample or data point in the input data is associated with an output label or target and the goal of the machine learning algorithm is to predict this target from the features of the input data[63,75]. One can think of supervised learning as a student-teacher relationship, where the machine learning algorithm as the student tries to learn from the input data and is provided with the correct result or target by a teacher who functions as a supervisor and teaches the student what to learn[63]. In unsupervised learning, on the other hand, there is no teacher supervising the student's learning process and there are no target variables that the machine learning algorithm should learn to predict. Instead, the goal of unsupervised learning is to infer properties of the input's probability distribution from the input data samples alone[63,75].

Classification and regression are common tasks from the category of supervised learning, while a typical example of unsupervised learning is clustering[63]. The goal of classification is to predict to which of a predefined number k of categories or classes an input sample belongs to[63]. To this end, the classifier typically either tries to learn a function $f : \mathbb{R}^n \to \{1, \ldots, k\}$ that for an input sample $x \in \mathbb{R}^n$ with n input features outputs the number of the class y the sample belongs to, i.e. $f(x) = y$, or a function $f : \mathbb{R}^n \to \mathbb{R}^k$, which outputs a probability distribution over the k different classes[63]. In the latter case, the predicted class $\hat{y}$ for input sample x is the class with the highest probability in the predicted output probability distribution. In regression, in contrast to classification, the target variable is numerical rather than categorical, and the machine learning algorithm thus tries to learn a function $f : \mathbb{R}^n \to \mathbb{R}$, which outputs a real number $y \in \mathbb{R}$ for each input sample $x \in \mathbb{R}^n$[63].

In order to be able to learn anything, supervised machine learning algorithms rely on a performance measure. This performance measure is usually specific to the task the machine learning algorithm is trying to learn and quantitatively measures how well the algorithm performs in predicting target variables from input data[63]. Performance measures are used in supervised learning in two different ways: During training and after training. During training, a loss function is used to compute the prediction error on the training data. The machine learning algorithm then tries to minimize this training error by iteratively adapting the algorithm's learnable parameters. After the training phase, a performance metric is typically used to evaluate the algorithm's performance on a test dataset that was not used for training in order to determine how well the algorithm generalizes to new data. In classification tasks, for instance, prediction accuracy is a commonly used performance metric to measure how well the machine learning algorithm has learned its classification task. How well a machine learning model generalizes to new data is closely related to the two concepts of underfitting and overfitting. If the model is not complex enough and its capacity to fit different functions is low, underfitting occurs and the model is not able to fit the training data sufficiently, resulting in a high training error[63]. On the other hand, if the model's capacity to fit different functions is too high, it may overfit the training data, resulting in a small training error, but a high test error[63]. Hence, a good machine learning model is characterized by a capacity that allows it to neither underfit nor overfit the training data and achieve a small training error while keeping the gap between training and test error small as well[63].

3.2.1 Tree Boosting

One widely used and effective type of machine learning is tree boosting[31]. The basic idea underlying the concept of boosting is that multiple weak predictors can together constitute a strong predictor[150,75]. In this context, a weak predictor can have an error rate that is only marginally better than random, but by sequentially applying and then combining multiple weak predictors, it is possible to compose a stronger predictor. In tree boosting, decision trees are used as weak predictors. Decision trees are named after their tree-like structure, where a sample is passed to the root node at the top of the tree and is then moved down through the tree according to a sequence of decisions or splits[58]. Each node in the decision tree except for the leaf nodes has an associated splitting attribute, which is usually an input feature, and when the sample passes through the node, it is forwarded to one of the child nodes through the branch that corresponds to a certain value of this attribute, i.e., that is either below or above a threshold value. In contrast to the inner nodes of the tree, the childless leaf nodes do not have splitting attributes associated with them. Instead, each leaf node $j = 1, 2, ..., J$ corresponds to a constant leaf score ω_j, and the decision tree will output that score as a prediction for all samples reaching that specific leaf node[58,75]. From a feature space perspective, a decision tree can also be viewed as a method that partitions the input feature space into J disjoint regions $R_j, j = 1, 2, ..., J$, corresponding to the J leaf nodes of the decision tree, and then assigns a leaf score ω_j to each region[75]. Figure 3.1 illustrates a decision tree with two features and six leaf nodes and shows the corresponding feature space partition.

Mathematically, a decision tree with parameters $\Theta = \{R_j, \omega_j\}_{j=1}^{J}$ can be formulated as

$$T(x; \Theta) = \sum_{j=1}^{J} \omega_j I\left(x \in R_j\right), \tag{3.1}$$

where $x = (x_1, x_2, ..., x_n)$ is an input sample with n features and $I(\cdot)$ refers to the indicator function, which outputs 1 if its input term evaluates to *true* and 0 otherwise[75].

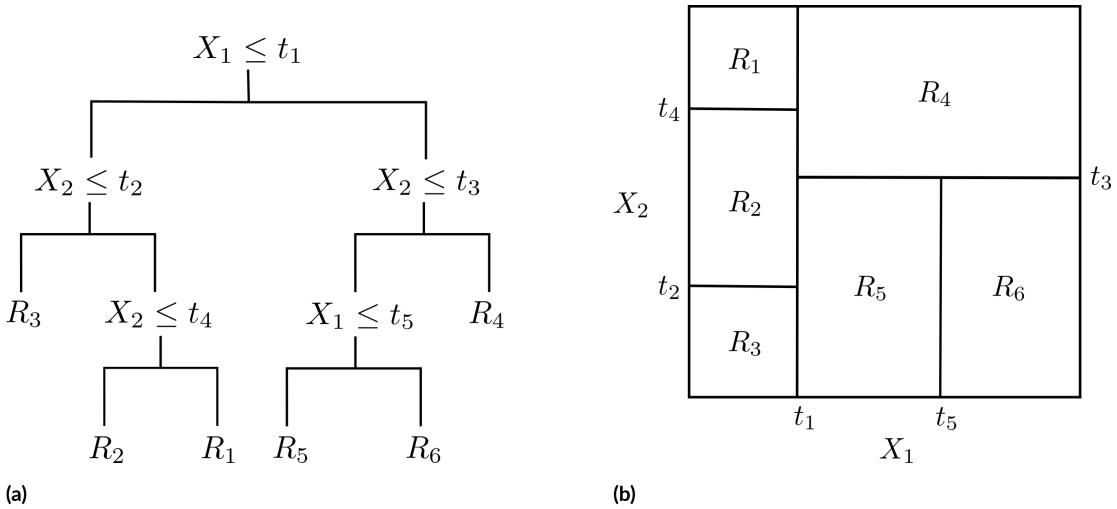


Figure 3.1: Decision tree. **(a)** Decision tree with two features (X_1 and X_2) and six leaf nodes, which partition the feature space into regions $R_1, ..., R_6$. Each inner node corresponds to one decision threshold $t_1, ..., t_5$ on one of the features. **(b)** Partition of the feature space by the decision tree from (a). Each rectangle corresponds to one leaf node i of the decision tree and the corresponding region of the feature space R_i. Adapted from[89].

A boosted tree is an additive combination of M such simple decision trees, that is,

$$f^{(M)}(x) = \sum_{m=1}^{M} f_m(x), \tag{3.2}$$

with $f_m(x) = T(x; \Theta_m)$. In each tree construction step $m \in \{1, ..., M\}$, a new tree $T(x; \Theta_m)$ is added to the model and its parameters $\Theta_m = \{R_{jm}, \omega_{jm}\}_{j=1}^{J_m}$, where J_m is the number of leaves of the tree, R_{jm} are the regions of the corresponding feature space, and ω_{jm} are the leaf scores, are optimized based on the previous model $f^{(m-1)}(x)$ with $f^{(0)}(x) = 0$ by solving the following optimization problem:

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} l\left(y_i, f^{(m-1)}(x_i) + T(x_i; \Theta_m)\right) \tag{3.3}$$

Here, $\hat{\Theta}_m$ are the optimal parameters for the m-th tree, y_i is the target output for sample $i \in \{1, ..., N\}$, and $l : \mathbb{R}^2 \to \mathbb{R}$ is a task-specific differentiable loss function. Accordingly, the objective that the model attempts to minimize in the m-th training iteration then becomes

$$\mathcal{L}^{(m)} = \sum_{i=1}^{N} l\left(y_i, f^{(m-1)}(x_i) + f_m(x_i)\right). \tag{3.4}$$

Notably, the leaf scores ω of boosted trees are continuous[31], that is $\omega_j \in \mathbb{R}$ for all $j =$

$1, 2, ..., J$.

Extreme Gradient Boosting

Extreme gradient boosting (XGBoost) is a scalable tree boosting method[31]. As a tree boosting method, it uses a tree ensemble of additive functions to predict an output given an input sample (cf. Equation 3.2). However, in contrast to the objective from Equation 3.4, which the model tries to minimize, XGBoost uses a regularized objective

$$\mathcal{L}^{(m)} = \sum_{i=1}^{N} l\left(y_i, f^{(m-1)}(x_i) + f_m(x_i)\right) + \Omega\left(f_m\right) \tag{3.5}$$

with $\Omega(f) = \gamma J + \frac{1}{2}\lambda \left\|\omega\right\|^2$ as a regularization term that penalizes model complexity to reduce overfitting. J is the number of leaves in the tree f, ω are the associated leaf scores, and γ and λ are model hyperparameters. The objective is optimized by second-order approximation:

$$\mathcal{L}^{(m)} \simeq \sum_{i=1}^{N} \left[l\left(y_i, f^{(m-1)}(x_i)\right) + g_i f_m(x_i) + \frac{1}{2} h_i f_m^2(x_i) \right] + \Omega(f_m), \tag{3.6}$$

where $g_i = \partial_{f^{(m-1)}(x_i)} l\left(y_i, f^{(m-1)}(x_i)\right)$ is the first order derivative of the loss function l and $h_i = \partial^2_{f^{(m-1)}(x_i)} l\left(y_i, f^{(m-1)}(x_i)\right)$ is the second order derivative of the loss function. This approximation can be derived from Taylor's theorem, which states that for a function f that is $(n+1)$ times differentiable in the interval I and $x, x_0 \in I$:

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}}{k!}(x - x_0)^k + R_n(x - x_0) \tag{3.7}$$

with remainder $R_n(x - x_0) = \frac{f^{(n+1)}(\vartheta)}{(n+1)!}(x - x_0)^{n+1}$ for a ϑ between x and x_0[73].

When removing the constant terms from the approximation in Equation 3.6, a simplified model objective $\tilde{\mathcal{L}}^m$ at training iteration m can be derived as

$$\tilde{\mathcal{L}}^m = \sum_{i=1}^{N} \left[g_i f_m(x_i) + \frac{1}{2} h_i f_m^2(x_i) \right] + \Omega(f_m) \tag{3.8}$$

$$= \sum_{i=1}^{N} \left[g_i f_m(x_i) + \frac{1}{2} h_i f_m^2(x_i) \right] + \gamma J_m + \frac{1}{2} \lambda \sum_{j=1}^{J_m} \omega_{jm}^2 \tag{3.9}$$

$$= \sum_{j=1}^{J_m} \left[\left(\sum_{i \in I_{jm}} g_i \right) \omega_{jm} + \frac{1}{2} \left(\sum_{i \in I_{jm}} h_i + \lambda \right) \omega_{jm}^2 \right] + \gamma J_m, \tag{3.10}$$

where $I_{jm} = \{i | q_m(x_i) = j_m\}$ is the instance set of leaf j_m of tree f_m, which contains the indices of the samples assigned to leaf j_m, and q_m is the tree structure of f_m, which maps samples to leaf indices. For a fixed tree structure $q_m(x)$, the optimal score w_{jm}^* of leaf j_m, which minimizes the objective, can be computed as

$$w_{jm}^* = -\frac{\sum_{i \in I_{jm}} g_i}{\sum_{i \in I_{jm}} h_i + \lambda}. \tag{3.11}$$

By substituting w_{jm} by w_{jm}^* in Equation 3.10, the corresponding objective term $\tilde{\mathcal{L}}^m(q_m)$ becomes

$$\tilde{\mathcal{L}}^m(q_m) = -\frac{1}{2} \sum_{j=1}^{J_m} \frac{\left(\sum_{i \in I_{jm}} g_i \right)^2}{\sum_{i \in I_{jm}} h_i + \lambda} + \gamma J_m. \tag{3.12}$$

This objective term $\tilde{\mathcal{L}}^{\,m}(q_m)$ can be used to evaluate how good the tree structure q_m is for the prediction task. Because this optimization strategy uses the gradient of the tree functions, this type of tree boosting is called gradient tree boosting.

Since it is computationally infeasible to test all possible tree structures q and compute their corresponding objective, XGBoost uses a greedy algorithm to construct a tree structure q_m starting from a single leaf and growing the tree by iteratively adding new split nodes and their corresponding branches and leaves. To this end, candidate splits on all features are constructed and each candidate split is evaluated in terms of the loss reduction by the split:

$$\Delta \mathcal{L}_{split} = \frac{1}{2} \left[\frac{\left(\sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i \right)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma, \tag{3.13}$$

where I_L and I_R with $I = I_L \cup I_R$ are the instance sets of left and right child nodes after the split and the candidate split with the largest $\Delta \mathcal{L}_{split}$ is selected.

3.2.2 Neural Networks

A neural network is a type of machine learning architecture that tries to simulate the learning mechanism in the brain of biological organisms[4]. It consists of multiple computational units, which are called neurons and are interconnected by edges, each of which is associated with a weight that is used to scale the input passing through the edge[4]. The neural network then computes a function of the inputs by propagating them through the network from the input neurons to the output neuron(s)[4]. More specifically, the neural network computes the function $f(x;\theta) = \hat{y}$ with inputs x, learnable parameters θ, which in a neural network are weights and biases, and output $\hat{y}$[63]. Learning occurs by adapting the parameters θ such that the function f computed by the neural network becomes as similar as possible to the target function $f^*(x) = y$, where y is the true outcome associated with x[63].

If the neurons of a neural network are arranged in a layer-wise fashion with one layer of input neurons, one or multiple intermediate layers, which are called hidden layers, and one output layer, the neural network is called a feedforward neural network, or sometimes also multilayer perceptron (MLP)[4,63] (Figure 3.2).

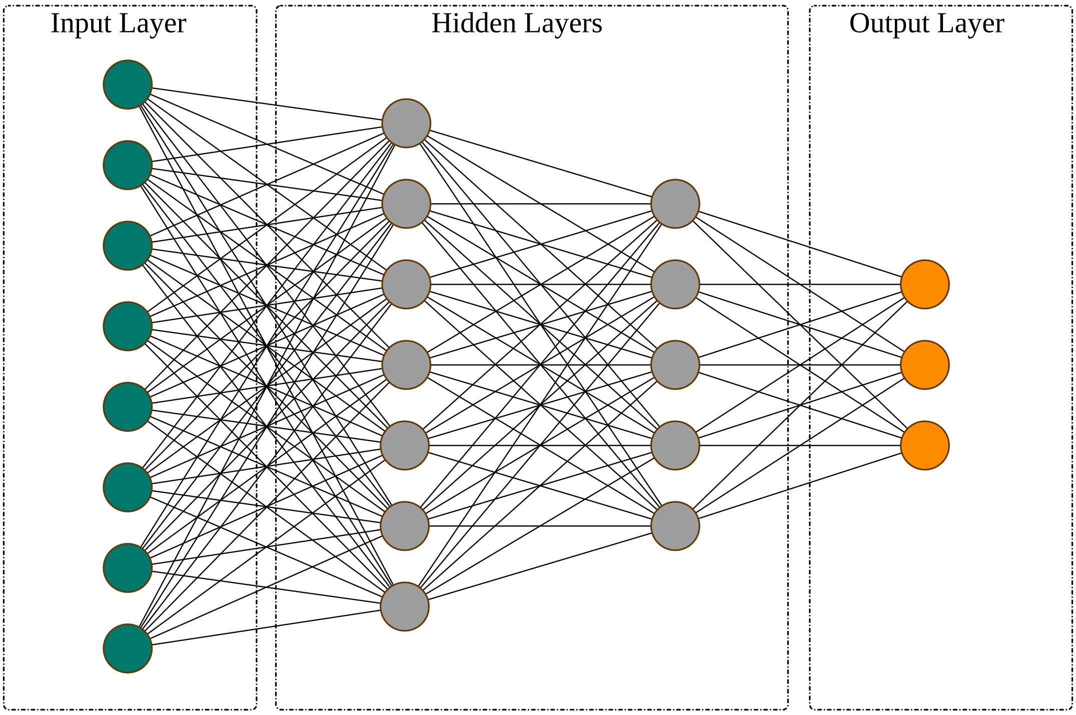


Figure 3.2: Feedforward neural network. General architecture of a feedforward neural network with one input layer, one or multiple hidden layers, and one output layer. In a feedforward neural network, each neuron in one layer is connected to all neurons in the next layer and information is propagated in a forward direction from the input layer through the hidden layers to the output layer. Created with `https://www.yworks.com/yed-live/`.

The name 'feedforward neural network' originates from the characteristic of this type of neural network that each node in one layer is connected to all nodes in the next layer by weighted edges and each layer 'feeds' its computed activation values to the next layer in a forward direction from input layer to output layer[4]. That is, the first layer of a feedforward neural network computes activation values by the following function:

$$h^{(1)} = g^{(1)}\left({W^{(1)}}^{\top} x + b^{(1)}\right), \tag{3.14}$$

where x is a vector of inputs, W is a layer-specific weight matrix, b is a bias vector, and g is a nonlinear activation function. Each subsequent layer i then computes the function

$$h^{(i)} = g^{(i)}\left({W^{(i)}}^{\top} h^{(i-1)} + b^{(i)}\right), \tag{3.15}$$

based on the activations $h^{(i-1)}$ of the previous layer[63]. While in early neural network architectures, the activation function g was usually the sigmoid or tanh function[4], in modern feedforward neural networks it is often recommended to use the rectified linear unit (ReLU) instead, which is defined as $g(z) = \max\{0, z\}$[63].

As in other machine learning architectures, learning in neural networks relies on a task-specific loss function, which measures the prediction error made by the model on the training data (cf. Section 3.2). The training algorithm is usually based on using the gradient to descend the loss function by iteratively adjusting the weight and bias parameters, driving the loss to a very low value[63]. To compute the gradient, the backpropagation algorithm is normally used. During backpropagation, the error computed by the loss function is propagated backward through the network from the output layer to the input layer to compute $\nabla l(\theta)$, which is the gradient of the loss function with respect to the parameters θ[63]. In this context, θ are the weights and biases of the neural network. Basically, backpropagation computes the chain rule of calculus, which can be used to calculate the derivatives of functions composed of other functions, given that the derivatives of the other functions are known[63]. The chain rule of calculus states the following[63]: Given $x \in \mathbb{R}$ and two functions $f : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ with $y = g(x)$ and $z = f(g(x)) = f(y)$, then

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}. \tag{3.16}$$

In the non-scalar case, if $x \in \mathbb{R}^m, y \in \mathbb{R}^n, g : \mathbb{R}^m \to \mathbb{R}^n$, and $f : \mathbb{R}^n \to \mathbb{R}$ with $y = g(x)$ and $z = f(y)$, the chain rule generalizes to

$$\frac{\partial z}{\partial x_i} = \sum_{j=1}^{n} \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i}. \tag{3.17}$$

Equation 3.17 can also be rewritten in vector notation using the $n \times m$ Jacobian matrix $\frac{\partial y}{\partial x}$ of g, which is a matrix containing all partial derivatives of g:

$$\nabla_x z = \left(\frac{\partial y}{\partial x}\right)^\top \nabla_y z \tag{3.18}$$

with $\nabla_y z$ being the gradient of z with respect to y and $\nabla_x z$ being the gradient of z with respect to x. Hence, backpropagation becomes the recursive computation of such a product of the Jacobian matrix and the gradient for each operation in the computational graph of the neural network[63].

Once the gradient of the loss function with respect to the model's parameters has been computed using backpropagation, this gradient can be used for learning by adapting the network's weights and biases using a gradient descent algorithm such as stochastic gradient descent[63]. Gradient descent is based on the notion that the derivative $f'(x)$ of a univariate function $f(x)$ provides the slope of the function at point x and thus indicates how a small change in the input x affects the output of the function[63]. Thus, $f(x)$ can be reduced by moving x in small steps into the opposite direction of the derivative[63]. If $f(x)$ is a multivariate function with $x = (x_1, ..., x_n)$, $n \in \mathbb{N}^+$, as is usually the case in neural networks, its gradient is a vector that contains all partial derivatives of f, where element i of the gradient is the partial derivative of f with respect to x_i, and the negative gradient points into the direction of steepest descent of f[63]. Thus, by moving into the direction of the negative gradient of f by a small step, $f(x)$ is decreased[63]. The step size is specified by the learning rate ε, which is a positive scalar[63]. Given an initial point x and a learning rate ε, a new point x' can be determined, which reduces $f(x')$ compared to $f(x)$ and is defined as follows[63]:

$$x' = x - \varepsilon \nabla_x f(x). \tag{3.19}$$

If f is the loss function of a neural network and x are the weights and biases that should be learned, this equation instructs the neural network how to change these weights and biases in order to reduce the prediction error. This step of adjusting weights and biases according to the gradient and learning rate is repeated multiple times during gradient descent and the procedure converges when all elements of the gradient are zero, or in practice rather very close to zero[63].

3.2.3 Transfer Learning

Transfer learning is a concept based on the idea that the performance of a machine learning model on a target domain—often with a limited number of labeled samples—can be improved by leveraging knowledge from a related source domain[191] (Figure 3.3).

A domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ consists of a feature space $\mathcal{X}$ and a marginal distribution $P(X)$

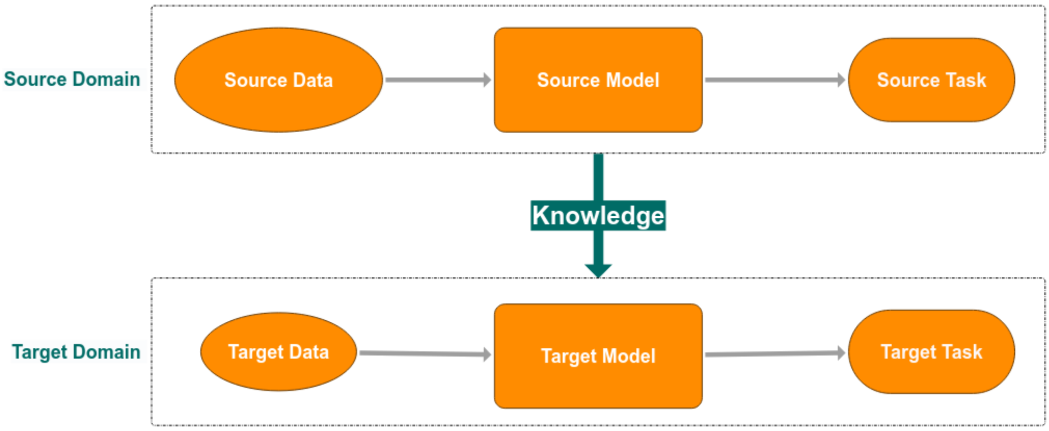


Figure 3.3: Transfer Learning. In transfer learning, knowledge learned on a source domain is transferred to a target domain to improve the performance of the target model. Created with `https://www.yworks.com/yed-live/`.

with instance set $X = \{x_i | x_i \in \mathcal{X}, i = 1, ..., n\}$ [191,188]. This means that every input instance x_i in the instance set X is contained in the domain's feature space $\mathcal{X}$ and $P(X)$ captures the distribution of the instance set. In practice, a domain is usually observed by a set of instances that belong to the domain and can either be labeled or unlabeled [191]. If two domains are different, this means that they can have different feature spaces or different marginal distributions [188]. In supervised machine learning, the goal usually is to learn a task $\mathcal{T}$ given some labeled training data from a domain $\mathcal{D}$. $\mathcal{T} = \{\mathcal{Y}, f\}$ has two components: the label space $\mathcal{Y}$ and a decision or prediction function $f : \mathcal{X} \rightarrow \mathcal{Y}$, which is an implicit function that is learned from the training data and can then be used to make predictions on unseen instances [191,188]. With these definitions in mind, transfer learning can be defined more formally: Given a set of n_S instance-label pairs $\{(x_i, y_i) | x_i \in \mathcal{X}_S, y_i \in \mathcal{Y}_S, i = 1, ..., n_S\}$ from a source domain $\mathcal{D}_S$ with feature space $\mathcal{X}_S$ and label space $\mathcal{Y}_S$ and a set of n_T instance-label pairs $\{(x_i, y_i) | x_i \in \mathcal{X}_T, y_i \in \mathcal{Y}_T, i = 1, ..., n_T\}$ from a target domain $\mathcal{D}_T$ with feature space $\mathcal{X}_T$ and label space $\mathcal{Y}_T$, and $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$ (i.e., the source and target domains are different or the source and target tasks are different), transfer learning uses knowledge from the source domain $\mathcal{D}_S$ and task $\mathcal{T}_S$ to improve the learning and performance of the prediction function $f^T : \mathcal{X}_T \rightarrow \mathcal{Y}_T$ on the target domain [191,179,188]. This two-domain scenario, where knowledge is transferred from one source domain to one target domain, is the most common transfer learning scenario [188]. However, it is also possible to extend the definition of transfer learning to multiple source and target domains and tasks. In this case, transfer learning uses knowledge learned from $m_S \in \mathbb{N}^+$ source domains and tasks $\{(\mathcal{D}_{S_j}, \mathcal{T}_{S_j}) | j = 1, ..., m_S\}$ to improve the prediction functions $f_{T_j}, j = 1, ..., m_T$, on $m_T \in \mathbb{N}^+$ different target domains and tasks $\{(\mathcal{D}_{T_j}, \mathcal{T}_{T_j}) | j = 1, ..., m_T\}$ [191].

Transfer learning can be categorized based on different criteria: For instance, transfer learning can be divided into homogeneous transfer learning and heterogeneous transfer learning based on the similarity of the source and target domains[191,188]. While in homogeneous transfer learning, the label spaces must be the same and the domains must have the same feature space, in heterogeneous transfer learning, the label spaces can be different and the domains can have different feature spaces, making it necessary to adapt the feature space during transfer learning and thus making heterogeneous transfer learning more complex than homogeneous transfer learning[191]. Another criterion by which transfer learning can be categorized is the availability of label information. In inductive transfer learning, label information is available for the target domain, while in transductive transfer learning, label information is only available for the source domain, but not for the target domain, and in unsupervised transfer learning, label information is available for neither domain[191]. The two types of transfer learning categorization described so far are based on the type of transfer learning problem at hand, but it is also possible to categorize transfer learning based on how knowledge is transferred. Instance-based, feature-based, parameter-based, and relation-based are four categories related to the question of "how to transfer"[191,188]. Instance-based approaches are usually based on an instance weighting strategy[191], where source-domain instances are weighted and the transferred knowledge consists of the instances with large weights[188]. Feature-based approaches try to learn a good feature representation—either by transforming the source features to match the target features or by learning a common latent feature space for the source and target domains—such that the source domain data can be used for training on the target domain[191,188]. Parameter-based approaches—also termed model-based approaches—transfer knowledge from the source domain to the target domain through the learned parameters of a model that was trained on the source domain[191,188]. The motivation behind this type of approach is the assumption that a model that has been well trained on the source domain will have captured much useful general structure, which can be transferred to the target domain, thereby benefiting target model performance[188]. Lastly, relation-based approaches are based on the idea that at least some relationships between instances are similar in the source and target domains and that rules regarding these relationships between entities can be transferred between domains[191,188].

In the context of neural networks and deep learning, pre-training approaches from the parameter- or model-based category are widely used[188]: To transfer knowledge from the source domain to the target domain, a (deep) neural network is first trained to solve a source task on the source domain. Once training is completed, the parameters of the pre-trained neural network are transferred to the target domain and task, for example by freezing some layers of the pre-trained neural network and fine-tuning the parameters of the last few layers based on labeled instances from the target domain[191,188].

Transfer learning has been shown to be successful in many cases. For instance, Zoph et al.[193] pre-trained a neural machine translation model on a large bilingual dataset and used the trained model to translate between languages with little bilingual data available, Phan et al.

used transfer learning for automatic sleep staging by pre-training a neural network on a large source dataset and fine-tuning it on smaller target datasets[136], and Maqsood et al.[123] transferred knowledge from the AlexNet[105] image classification method to detect Alzheimers's disease from MRI images. However, transfer learning does not always positively affect the prediction performance on new tasks[191]. Instead, transferring knowledge from a source domain to the target domain can even negatively impact model performance on the target domain and task[191]. This phenomenon is called *negative transfer* and can happen, for example, if the source and target domains are not related to each other closely enough or if the model is not able to find the part of the knowledge that is transferable to the target domain and would benefit performance[191,188].

3.2.4 Feature Selection

If machine learning methods are trained on high-dimensional data—as in the case of most omics data such as gene expression—a phenomenon termed the "curse of dimensionality" can make training difficult and cause overfitting. The term "curse of dimensionality" was first introduced by Richard Bellman[15] and refers to the circumstance that the number of observations required to specify a function in an n-dimensional space grows exponentially with the number of dimensions n and thus in higher dimensions data becomes more sparse[15,111]. Possible approaches for alleviating the overfitting problem caused by high data dimensionality are the application of dimensionality reduction techniques such as principal component analysis (PCA)[84,134] to the data before training or the use of feature selection techniques to select a subset of features to train the machine learning model on. While dimensionality reduction techniques alter the original representation of the features, potentially exacerbating interpretability, feature selection methods select a subset of the original features, leaving feature representations intact and thus allowing for better interpretability[146].

Feature selection methods applied to supervised learning problems can be categorized into filter methods, wrapper methods, and embedded methods[146]. Filter methods have in common that they select features based on intrinsic properties of the data, such as the correlation between features or the χ^2 test[146]. In contrast to filter methods, which do not consider dependencies between the features and the target variable, wrapper methods do take such dependencies into account by evaluating the same machine learning method as used for the prediction task on different subsets of features and selecting the feature subset that yields the best performance[146]. However, wrapper methods are computationally expensive, especially for datasets with many features, because the number of feature subsets that need to be evaluated by the machine learning method grows exponentially with the number of features[146]. Embedded methods, on the other hand, also take relations between the features and the target into consideration and use the same machine learning method as used for the prediction task to evaluate features, but in contrast to wrapper methods, the evaluation is not done on different subsets of features, but by leveraging built-in feature importance measures of the selected machine learning method[146]. While embedded feature selection methods are thus only feasible with machine learning methods implementing such feature importance measures, they are computationally far less expensive than wrapper methods[146] because they don't rely on evaluating different feature combinations separately.

3.2.5 Hyperparameter Optimization

In addition to learnable parameters, which are estimated from training data, many machine learning models also have parameters that cannot be directly learned from the data[106]. These parameters are called hyperparameters, also referred to as tuning parameters[106,89]. Hyperparameters control many important properties of the machine learning model such as model complexity and poorly adapted hyperparameters can negatively impact the prediction performance[106]. Therefore, for most machine learning models, it is crucial to optimize the model's hyperparameters.

Grid Search

Grid search is the most basic hyperparameter optimization method[56]. Given a finite set of discrete values for each hyperparameter, grid search evaluates the machine learning model for every possible combination (the Cartesian product) of these sets of hyperparameter values[56], usually on a set of validation samples that are neither used for estimating the model's learnable parameters nor for evaluating the performance of the final model. Afterwards, the combination of hyperparameters that yielded the best model performance can be selected for training a final machine learning model. However, the main disadvantage of grid search is that the number of model evaluations grows exponentially with the number of optimized hyperparameters and the number of candidate values per hyperparameter[56]. Thus, for machine learning models with many hyperparameters or fine-grained hyperparameter configuration spaces, grid search becomes computationally expensive and sometimes infeasible due to its high time complexity.

Random Search

Random search is a less runtime-intensive, yet simple alternative to grid search. In contrast to grid search, which evaluates all possible hyperparameter configurations from a finite set of hyperparameter values, random search only evaluates a pre-defined number of randomly sampled hyperparameter configurations[56]. Importantly and as opposed to grid search, random search does not need to be supplied with a set of discrete candidate values for each hyperparameter, but can also sample hyperparameter values from a continuous distribution, allowing it to evaluate more different values of a hyperparameter than grid search typically does.

Bayesian Optimization

Instead of treating the evaluated hyperparameter configurations independently from each other, as grid search and random search do, Bayesian optimization iteratively selects new hyperparameter configurations to evaluate based on the results of previously evaluated hyperparameter configurations. In each iteration, a probabilistic surrogate model, which can for example be a Gaussian process or a Tree-structured Parzen Estimator (TPE), is fitted to the results of all hyperparameter configurations that have been evaluated up to this point[56]. Then, an acquisition function is used to determine the utility of different hyperparameter configurations based on the predictive distribution of the surrogate model, balancing exploration and exploitation[56]. In this context, exploration means selecting hyperparameter configurations in yet relatively unexplored regions of the hyperparameter space, while exploitation means selecting hyperparameters in regions that are most likely to yield a good performance according to the current surrogate model.

3.3 Survival Analysis

Survival analysis refers to a set of problems where individuals from one or more groups may experience a defined event, often called failure, that occurs after a certain period of time, often called failure time[45]. The failure event must occur at a discrete time point and each individual can experience the event only once[45]. Furthermore, the time origin must be defined for each individual and the different individuals should be as comparable as possible at their time origin[45]. The time to the event or failure time of each individual is then measured with respect to the individual's time origin[45]. Examples of survival analysis tasks are survival times of patients enrolled in a clinical study, but also lifetimes of machines or machine components in industry settings, or duration of unemployment or strikes in economics[45]. In the first case of patient survival in a clinical study, for instance, the time origin could be the entry date at which an individual was enrolled in the clinical study and the failure that an individual might experience would be death in general or death from a certain cause like lung cancer. Failure time would then accordingly be measured as time from study enrollment to death.

3.3.1 Censoring

A peculiarity of survival data is that some individuals may not have been observed until they experienced the event of failure[45]. In the case of a clinical study, some patients might have survived until the end of the study or some patients might have dropped out of the study at some point and might thus have been lost to follow-up. Alternatively, if the failure event is not death in general, but death from a certain cause such as lung cancer, a patient who has died from another cause (e.g. cardiovascular disease), would not experience the event of interest either. If failure cannot be observed in a patient, the patient is called censored and the event that made it impossible to observe failure is called censoring[45]. Like failure, censoring also occurs at a discrete time point and the time period until censoring occurs is called censoring time[45]. Figure 3.4 illustrates the survival times of ten different patients, of which five experience failure and the other five are censored at different time points, in real time (Figure 3.4a) and in time relative to entry into the study as the time origin (Figure 3.4b). For an individual i, the observation can be formalized in terms of its failure time t_i and the time period of observation c_i, which is the censoring time if the individual has not experienced failure by that time[45]. Then, the observation for individual i consists of the time $y_i = \min(t_i, c_i)$ and an indicator variable $\delta_i \in \{0, 1\}$, which indicates the censoring status of the individual with $\delta_i = 0$ if $t_i > c_i$ (i is censored) and $\delta_i = 1$ if $t_i \leq c_i$ (i is uncensored)[45].

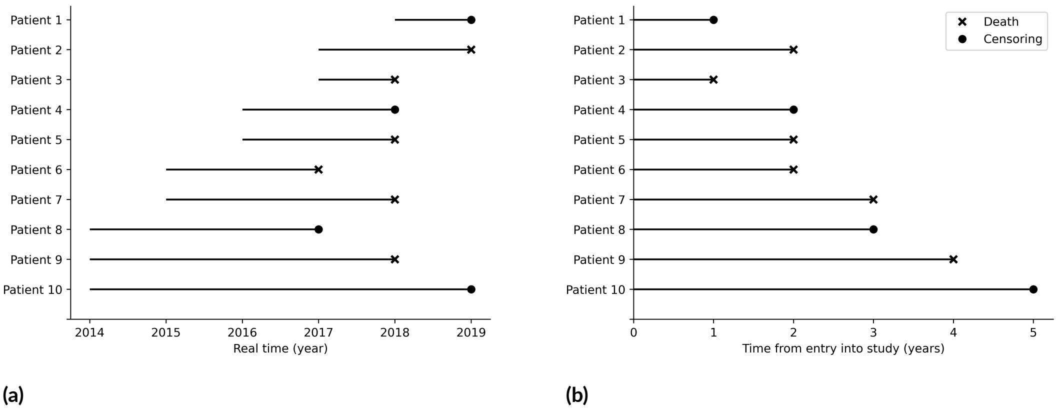


Figure 3.4: Survival time. **(a)** Real survival times of ten patients. Lines start at the year of entry into the study of the respective patient and end at the year of death (failure) or censoring. **(b)** Time to failure or time to censoring for the same ten patients in years with study entry as the time origin. Events are indicated with ✖ for death (failure) and ● for censoring. Adapted from[45].

3.3.2 Cox Regression

In conventional regression, the task is to learn a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and predict a numerical target value from a given input[63]. In survival analysis problems, however, where some of the samples are censored before they experience failure, conventional regression is not suited to predict survival in terms of failure time, because the failure time of censored patients is unknown. To resolve this problem, David Cox developed a novel type of regression that can handle censored samples[44] and is also known as Cox regression. Cox regression considers the following problem setting: Given a population of n_0 individuals for n of which failure time and for the rest censoring time is observed, the failure time is represented by a random variable T, which can be either discrete or continuous, with $k \leq n$ different failure times $t_{(1)} < t_{(2)} < ... < t_{(k)}$ and $k = n$ in the continuous case. Then, the age-specific failure rate or hazard is defined as

$$\lambda(t) = \lim_{\Delta t \to 0_+} \frac{P(t \leq T \leq t + \Delta t | t \leq T)}{\Delta t}. \tag{3.20}$$

That is, $\lambda(t)$ is a function of time, which for a given time t yields the rate of failure in the infinitesimal time interval $[t, t+\Delta t)$. Furthermore, $\lambda(t)\Delta t$ can be regarded as the approximate probability that an individual that has survived until time t fails in the next instant[100].

If for each individual $i \in \{1, ..., n_0\}$, m covariates x are available and the ith individual is associated with the covariate values $x_i = (x_{1i}, ..., x_{mi})$, the hazard function can be expressed as a function of time t and covariates x and becomes:

$$\lambda(t; x) = \exp(x\beta)\lambda_0(t), \tag{3.21}$$

where $\lambda_0(t)$ is an unknown baseline hazard function under the standard conditions $x = 0$ and β is a $m \times 1$ vector of unknown parameters[44]. Cox[44] allows $\lambda_0(t)$ to be arbitrary with the reasoning that the main interest of survival analysis is in the regression parameters β and if $\lambda_0(t)$ is left arbitrary, usually only little information about β is lost.

In the case of continuous failure times, the conditional probability that an individual j dies or fails at time point t_i, given that $i - 1$ individuals $j_1, ..., j_{i-1}$ have failed before t_i and one individual from the remaining population that has not experienced failure or censoring before t_i fails at t_i can be computed as[45]

$$P_i(j|j_1, ..., j_{i-1}) = \frac{\exp(x_j\beta)\lambda_0(t_i)}{\sum_{l:t_l \geq t_i}(\exp(x_l\,\beta)\lambda_0(t_i))} \tag{3.22}$$

$$= \frac{\exp(x_j\,\beta)}{\sum_{l:t_l \geq t_i}\exp(x_l\,\beta)}, \tag{3.23}$$

with x_i being the covariate set of individual i. According to the chain rule for conditional probabilities, the joint probability distribution for all failures $j_1, ..., j_n$ can be computed as[45]

$$p(j_1, ..., j_n) = \prod_{i=1}^{n} P_i(j_i|j_1, ..j_{i-1}) \tag{3.24}$$

$$= \prod_{i=1}^{n} \frac{\exp(x_i\,\beta)}{\sum_{l:t_l \geq t_i}\exp(x_l\,\beta)} \tag{3.25}$$

and the corresponding partial log-likelihood is[45,44]

$$L_{\text{Cox}}(\beta) = \sum_{i=1}^{n} \left(x_i\,\beta - \log \sum_{l:t_l \geq t_i} \exp(x_l\,\beta) \right). \tag{3.26}$$

For machine learning-based survival prediction, the negative partial log-likelihood is commonly used as a loss function[34,95,107], which takes the form:

$$l_{Cox} = -\sum_{i=1}^{n} \left(\hat{h}_\beta(x_i) - \log \sum_{l:t_l \geq t_i} \exp(\hat{h}_\beta(x_l)) \right), \tag{3.27}$$

where l_{Cox} denotes the loss function and the predicted log-risk $\hat{h}_\beta(x)$ is the output of the machine learning model given an input x. To make the loss independent of the size of the analyzed dataset, it is often averaged over the number of uncensored samples n[95,107] and thus becomes:

$$l_{Cox} = -\frac{1}{n} \sum_{i=1}^{n} \left(\hat{h}_\beta(x_i) - \log \sum_{l:t_l \geq t_i} \exp(\hat{h}_\beta(x_l)) \right) \tag{3.28}$$

$$= -\frac{1}{n} \sum_{i:\delta_i=1} \left(\hat{h}_\beta(x_i) - \log \sum_{l:t_l \geq t_i} \exp(\hat{h}_\beta(x_l)) \right), \tag{3.29}$$

where δ_i indicates the censoring status of individual i with $\delta_i = 1$ if i is uncensored and $\delta_i = 0$ otherwise.

3.3.3 Concordance Index

The concordance index or C-Index[71] is a performance metric commonly used to evaluate the performance of survival prediction methods. It is suited for populations of individuals where part of the individuals experience failure and the other part of individuals are censored. The C-Index calculates the ratio of the number of pairs of individuals whose failure times and predictions are concordant with the number of comparable pairs. In this context, a concordant pair denotes a pair of individuals where the individual with the shorter failure time has the smaller predicted failure time or the larger predicted risk score, and a comparable pair is a pair of individuals where either both individuals are uncensored or one individual is censored and its censoring time is larger than the failure time of the other individual, thus allowing to determine which of the two individuals survived longer. Therefore, the C-Index can be viewed as a type of rank correlation between observed failure times and predicted failure times or risks[71].

For this work, the C-Index implementation of Dereli et al.[51] was adapted as follows to the prediction of risk instead of failure time:

$$\text{C-Index} = \frac{\sum_{i=1}^{N} \sum_{j \neq i} \Delta_{ij} I\left(\left(y_i - y_j\right)\left(\hat{y}_j - \hat{y}_i\right) > 0\right)}{\sum_{i=1}^{N} \sum_{j \neq i} \Delta_{ij}} \tag{3.30}$$

$$\text{with } \Delta_{ij} = \begin{cases} 1 & \text{if } (\delta_i = 1 \text{ and } \delta_j = 1) \text{ or } (\delta_i = 1 \text{ and } \delta_j = 0 \text{ and } y_i < y_j) \\ 0 & \text{otherwise,} \end{cases} \tag{3.31}$$

where y_i is the failure time of individual i, $\hat{y}_i$ is the corresponding predicted risk score, $\delta_i \in \{0, 1\}$ denotes the censoring status with $\delta_i = 1$ if i is uncensored and $\delta_i = 0$ if i is censored, Δ_{ij} indicates whether the pair of individuals i and j is comparable, and $I(\cdot)$ is the indicator function.

3.4 Network Propagation

As described in Section 2.3, the entirety of PPIs can be formulated as an undirected graph, where nodes represent proteins and edges represent interactions between proteins. Network propagation is a type of method in which information is diffused or propagated over a graph or network to amplify biological signal and at the same time reduce noise[43]. This concept can be imagined as fluid flowing through the graph or network, where each node is filled with a node-specific amount of fluid that is proportional to its initial importance and this fluid then flows from one node to the neighbors of this node, and then in the next step from the neighbors to the neighbors' neighbors and so on, until either the diffusion process is halted after a few steps or an equilibrium is reached, where all liquid is evenly distributed over all nodes[43]. In the first case, where the liquid diffusion or network propagation process is stopped after a few steps, some nodes in the neighborhood of nodes that were initialized with a large amount of fluid will have received more fluid than other nodes, which is proportional to their importance in the network after network propagation and can be used to prioritize nodes. However, the other case, where the liquid is evenly distributed over all nodes of the network at the end of the network propagation, is not informative, because all nodes have the same amount of liquid at the end and no prioritization of nodes is not possible anymore. Another approach to gain information from network propagation and prioritize nodes besides halting the network propagation after a certain number of steps is the concept of random walk with restart (RWR)[43]. In contrast to the approach described before, where each node is initialized once with a certain amount of fluid and this fluid is then diffused over the network, RWR returns the liquid to the initial nodes at each step with a certain probability[43]. This way, the fluid is kept somewhat close to the initial nodes and propagation to distant nodes through long paths becomes less likely, thus making it possible for the network propagation to continue for many steps and reach a steady state, where the amount of fluid in each node at each step changes only marginally, while the fluid is still largely confined to the local node neighborhoods of the initial nodes[43].

More formally, the amount of fluid or weight associated with each node after step k of RWR can be computed as

$$p_k = \alpha p_0 + (1-\alpha)\,Wp_{k-1}, \tag{3.32}$$

where p_0 is a weight vector, which assigns an initial weight to each node of the network, p_{k-1} are the node weights in step $k-1$ of the network propagation, W is a normalized version of the adjacency matrix A of the network, where A represents which nodes are connected to which other nodes in the network, and α is a restart probability, specifying with which probability the node weight or fluid is returned to the respective initial node at each step[43]. α can also be viewed as a network smoothing parameter, where a larger value of α corresponds to less network smoothing[43]. Thus, the network propagation over k steps can be computed iteratively starting from p_0. A common normalization for the adjacency matrix A is the degree normalization, where A is normalized by the diagonal degree matrix D, which on the diagonal contains the degree, that is the number of neighbors, of each node, and the normalized adjacency matrix is computed as $W = AD^{-1}$ or $W = D^{-1/2}AD^{-1/2}$[43].

If the eigenvalues of the normalized adjacency matrix W are less than or equal to one and the network is connected, meaning each node can be reached from each other node in the network by traversing the edges, RWR converges to a steady-state distribution p, which can be calculated as

$$p = \alpha\,(I-(1-\alpha)\,W)^{-1}\,p_0, \tag{3.33}$$

with I being the identity matrix[43]. Figure 3.5 illustrates RWR on a small graph.

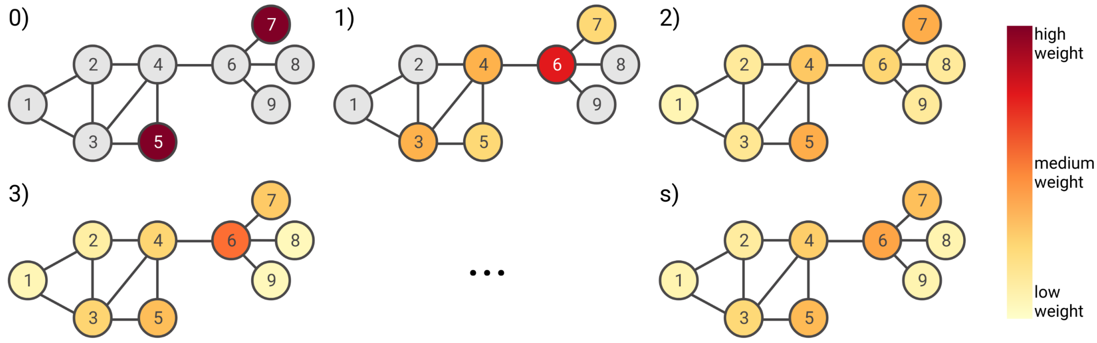


Figure 3.5: Random walk with restart (RWR). Illustration of RWR on a small graph with nine nodes. In 0) two nodes (5 and 7) are initialized with a high weight, while all other nodes are initialized with zero weight. During the subsequent network propagation steps 1)-3), the weight is propagated over the network, until a steady-state s) is reached. Created with BioRender.com.

In the biological context, network propagation is often used to propagate information over PPI networks, which were introduced in Section 2.3. Here, the underlying assumption is that proteins encoded by genes that give rise to similar phenotypes have a tendency to interact with each other and this tendency can be exploited by network propagation to identify proteins (and their encoding genes) that are associated with a phenotype of interest but were not initially known to be associated with this phenotype by leveraging prior information on genes with a known association to this phenotype.

3.4.1 NetCore

NetCore[13] is a network propagation method, which is based on RWR, but in contrast to normalizing the adjacency matrix of the underlying graph by the node degree as described above, NetCore applies a normalization based on node core to the adjacency matrix.

PPI networks are believed to be scale-free[12]. That is, they follow a power-law distribution, where a small number of nodes, the so-called hubs, have a high degree and are connected to a large number of interaction partners, while most nodes only participate in a small number of interactions. A possible biological explanation for the scale-free nature of PPI networks is gene duplication[12]: When a gene is duplicated during cell division, resulting in two identical genes and consequently two identical proteins in the daughter cells, the proteins interacting with the protein that is being duplicated will also interact with the duplication of this protein and thus each gain an interaction partner. If gene duplication is equally likely for each protein-coding gene, proteins with many interaction partners are more likely to be connected to a duplicated protein than proteins with few interaction partners and will therefore gain interactions through duplication events at a higher probability. This concept is called preferential attachment and promotes the scale-free characteristic of PPI networks when these networks are growing. However, besides this biological explanation for the power-law distribution that is observed in PPI networks, there are also experimental reasons for this observation. As described in Section 2.3, PPIs are commonly detected by experimental methods like Y2H or AP-MS, where one protein of interest is used as bait protein and interactions of other proteins with this bait protein are measured. The selection of the bait proteins is done by the investigator and hence is biased towards certain proteins of interest, which are thus heavily studied and have many detected PPIs, while other proteins are studied less and thus have fewer known interactions[62]. Additionally, the selection of a protein as bait can introduce further experimental bias stemming from proteins behaving differently in Y2H experiments if they are used as bait as compared to being used as prey[164].

NetCore tries to reduce this degree bias by using a normalization strategy based on node core during network propagation instead of the commonly used degree normalization[13]. The core value of a node can be computed from node degree in an iterative process and reflects the influence of a node on the spreading of information in the network. More precisely, the node core measures how central a node is in the network, where a high node core indicates that the node is located in a densely connected part of the network, while nodes in the periphery of the network have a low core value, even if they have a high degree. To compute the core value of a node, the k-shell decomposition method[59] can be used. In the first step, all nodes with degree $k = 1$ are recursively removed from the network such that only nodes with degree $k \geq 2$ remain in the network. Recursive removal of nodes with degree k means that removing nodes with degree k from the network is repeated until no nodes with degree k are left in the network after the removal step. A core value of 1 is assigned to all nodes that could be removed from the network in this first step. In the second step of the k-shell decomposition, all nodes with degree $k = 2$ are recursively removed from the network until only nodes with degree $k \geq 3$ remain and a core value of 2 is assigned to the removed nodes. This node removal step is repeated iteratively with increasing values of k until all nodes in the network have been assigned a core value. The node core values are used by NetCore to normalize the adjacency matrix A of the PPI network as follows:

$$A^{core}_{i,j} = \frac{k_i}{\sum_{l:A_{l,j} \neq 0} k_l}, \tag{3.34}$$

where k_i is the core value of node i in the network[13]. Thus, during the network propagation, neighbors of a node that have a high core value will get more weight than neighbors with a low core value that are located in the network's periphery. Network propagation is then performed by computing the steady-state distribution of an RWR according to Equation 3.33 with $W = A^{core}$[13].

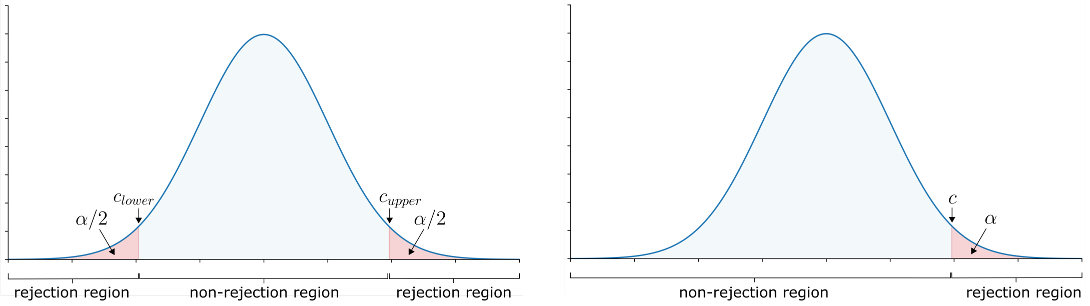
In addition to re-weighting nodes by network propagation, NetCore implements semi-supervised module identification, where network propagation results are combined with a set of seed genes, which can be either user-defined or inferred from the initial node weights of the network propagation input, to extract a biologically relevant sub-network and network modules from the PPI network[13]. To this end, initially, only seed genes are included in the sub-network and the sub-network is then extended by intermediate nodes that are direct neighbors of at least one seed node, have a weight that is above a pre-defined minimum weight after network propagation and have been identified as significant according to a permutation test[13]. If the user does not define a list of seed genes, the top 100 genes with the highest initial weights before network propagation are used as seed genes[13]. After constructing the sub-network, the sub-network is split into connected components, which are called modules[13].

3.5 Hypothesis Testing

Hypothesis testing is a statistical concept used to decide if a statistical hypothesis about a population should be either accepted or rejected based on experimental sample values from this population[137]. In hypothesis testing, there are two types of hypotheses, which are complementary to each other. The first type of hypothesis is the null hypothesis H_0, also called the statement of "no difference", which is the hypothesis the researcher wants to investigate through the experiment. The second hypothesis type is the alternative hypothesis H_1, which states the opposite of the null hypothesis.

For instance, the researcher might have a coin and would like to find out whether this coin is fair or biased. In this case, the null hypothesis could be "The coin is fair" and the alternative hypothesis could be formulated as "The coin is biased".

To decide whether to accept or reject the null hypothesis, a test statistic is necessary[137]. The test statistic is a function of the experimental sample values and follows a distribution that depends on a parameter θ with $\theta = \theta_0$ under the null hypothesis[70]. The distribution of possible outcomes of the test statistic is divided into two disjoint regions, the non-rejection region (sometimes also called acceptance region[137]), which includes all outcomes that are consistent with the null hypothesis at a predefined level of confidence and thus do not lead to a rejection of the null hypothesis, and the rejection region, which is also called critical region and comprises all outcomes that lead to the rejection of the null hypothesis[70]. The boundary values dividing non-rejection and rejection regions are called critical values and can be computed based on the probability distribution of the test statistic given a significance level α[70]. Thus, the significance level α controls when the null hypothesis is rejected, which is the case when, under the null hypothesis, the probability that a sample comes from the hypothesized probability distribution is less than or equal to α[70]. In practice, α is typically set to a small value of 0.01, 0.05, or 0.10[70]. If the null hypothesis is rejected despite being true, this is called a Type I Error and the probability for committing a Type I Error is α[70]. Conversely, not rejecting the null hypothesis even though the alternative hypothesis is true is called Type II Error[70].



(a) Two-sided hypothesis test

(b) One-sided hypothesis test

Figure 3.6: Hypothesis test. **(a)** In a two-sided hypothesis test, the rejection region consists of two parts and is separated from the non-rejection region by two critical values c_{lower} and c_{upper}. The total area of the rejection region is equal to the significance level α. **(b)** In a one-sided hypothesis test, only one critical value c separates non-rejection and rejection regions. Here, a right-sided test is shown for illustration.

There are two types of test statistics, two-sided tests and one-sided tests: In two-sided tests, the rejection region consists of two parts on both sides of the non-rejection region, which are defined by the two critical values c_{lower} and c_{upper}, and outcomes that fall either below c_{lower} or above c_{upper} lead to the rejection of the null hypothesis[70] (Figure 3.6a). On the other hand, in one-sided tests, there is only one critical value that separates non-rejection and rejection regions[70]. If the rejection region lies on the left side of the non-rejection region and the researcher is interested in outcomes smaller than the critical value, this is called a left-sided test, while in a right-sided test, the rejection region lies on the right side of the non-rejection region and values larger than the critical value are of interest[70] (Figure 3.6b).

3.5.1 *P*-Value

The *p*-value is the probability of obtaining an outcome by chance that is as extreme or more extreme than the observed outcome under the assumption that the null hypothesis is true[101]. To determine if the *p*-value of an observed outcome is statistically significant, it can be compared to the significance level α. If the *p*-value is smaller than or equal to α, this means that the corresponding test statistic value falls within the rejection region of the test statistic and thus the null hypothesis can be rejected[70]. Conversely, if the *p*-value is larger than α, this is reflective of the test statistic value falling within the non-rejection region and hence the null hypothesis is not rejected.

3.5.2 Multiple Hypothesis Testing

Multiple hypothesis testing refers to the setting where two or more statistical hypotheses are tested simultaneously[152]. In that case, the probability of committing a Type I Error accumulates in proportion to the number of tested hypotheses[152].

This can be illustrated by an example: Consider the setting where ten different hypotheses are tested simultaneously at a significance level of 0.05. Then, the probability of obtaining at least one significant result purely by chance would be approximately 40%, as shown by the following calculation:

$$\begin{aligned} P(\text{at least one significant result}) &= 1 - P(\text{no significant result}) \\ &= 1 - (1 - 0.05)^{10} \\ &\approx 0.401 \end{aligned}$$

This probability of committing at least one Type I Error is called the family-wise error rate (FWER)[83]. If not only 10, but 50 hypotheses were tested simultaneously at the same significance level of 0.05, the FWER would rise from 40% when testing 10 hypotheses to above 90% when testing 50 hypotheses. Thus, when multiple hypotheses are tested simultaneously on the same data, all of the obtained p-values need to be adjusted for multiple testing, for instance with the Bonferroni method or the less conservative Benjamini-Hochberg method for multiple testing correction.

Bonferroni Method

The Bonferroni method is a multiple testing correction method that controls the FWER by adjusting the significance level α proportionally to the number of tested observations[77]. That is, instead of comparing the p-value of an observation to the significance level α, it is compared to α/n, where n is the number of tested observations and the null hypothesis is rejected only if the p-value is smaller than α/n.

Benjamini-Hochberg Method

Instead of directly controlling the FWER like the Bonferroni method, the Benjamini-Hochberg method controls the false discovery rate (FDR), which is the likelihood that an incorrect rejection of the null hypothesis occurs[76]. Controlling for the FDR instead of the FWER is less conservative, allowing for the rejection of more null hypotheses in a family of tested hypotheses.

In the Benjamini-Hochberg multiple testing correction method[16], the m tested hypotheses are sorted by their p-values $p_{(i)}$ in ascending order such that $p_{(1)} \leq p_{(2)} \leq \ldots \leq p_{(m)}$. Then k is determined, which is the largest index i for which

$$p_{(i)} \leq \frac{i}{m} q^*,$$

where q^* is the desired FDR. Once k has been determined, the null hypotheses corresponding to p-values $p_{(i)}$ with $i = 1, 2, \ldots, k$ are rejected.

3.5.3 Wilcoxon Rank Sum Test

The Wilcoxon rank sum test is a nonparametric test that is used to assess whether the difference between two groups of samples is significant[79]. To this end, the samples from both groups are sorted by magnitude and a rank is assigned to each sample based on this sorting, where the sample with the largest magnitude gets a rank of 1. If two or more samples have the same magnitude, the average rank of these samples in the sorted magnitude list is assigned to these samples (e.g., if three samples have the same magnitude and are located at ranks 2, 3, and 4 in the sorted magnitude list, all of the three samples are assigned a rank of 3). Then, for each of the two sample groups that should be compared, the sum of ranks is computed over all samples belonging to the respective group and the rank sum value of the smaller sample group is compared to a rank sum score-associated p-value table to assess statistical significance.

The Wilcoxon rank sum test is related to the Student's t-test in that both tests are used to compare two sample groups and assess whether both groups are significantly different from each other, but in contrast to the parametric Student's t-test, which requires the two sample groups to be normally distributed and to have approximately equal variance, the Wilcoxon rank sum test is nonparametric and does not have these requirements[78,79].

3.6 Over-Representation Analysis

Over-representation analysis (ORA) is a type of pathway analysis that can be used for the functional interpretation of genes associated with a phenotype of interest[183]. More precisely, given a collection of gene sets or biological pathways and a list containing genes of interest (e.g., genes that are associated with a phenotype of interest), ORA can be used to identify those gene sets or pathways that are significantly over-represented or enriched for the genes of interest[183]. To this end, ORA uses the hypergeometric distribution to calculate a p-value, which reflects the probability that at least as many genes of interest as observed are contained in a given gene set or pathway by chance, and can be calculated as follows[21]:

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{i}}, \tag{3.35}$$

where N is the number of genes in the background distribution (e.g., all genes measured in an experiment), M is the size of the input list of genes (e.g., genes that are associated with a phenotype of interest), n is the size of the gene set or pathway, and k is the number of genes from the input list that are contained in the gene set or pathway (Figure 3.7).

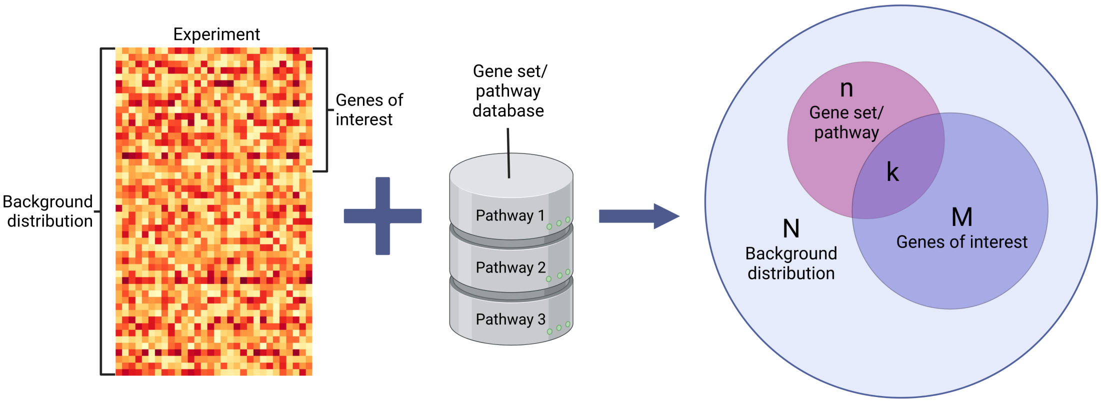


Figure 3.7: Over-representation analysis (ORA). Given a list containing genes of interest from a background distribution (e.g., all genes measured in an experiment) and a database of gene sets or pathways, ORA uses the hypergeometric distribution to identify gene sets or pathways significantly enriched for genes of interest. The Venn diagram represents the ORA parameters from Equation 3.35. Created with `BioRender.com`.

This type of test to compute the p-value is also known as right-tailed Fisher's exact test[183]. Since this p-value is typically calculated for each of multiple gene sets or pathways separately, multiple testing correction is commonly applied to the computed p-values.

4

Related Work

This chapter aims to place this dissertation in context with existing work in the field by highlighting selected publications related to different aspects of this dissertation.

4.1 Cancer Survival Prediction

The focus of this dissertation is cancer survival prediction. Cancer survival prediction is a challenging task that has been addressed in several works. In this section we will introduce a selection of these works. Most publications in the field of cancer survival prediction address the task of single-cancer survival prediction, where the aim is to predict the survival of patients suffering from a specific type of cancer. Relevant examples of single-cancer survival prediction methods will be explained in Subsection 4.1.1. However, there are also some works addressing pan-cancer survival prediction, where survival models are trained on patients from multiple cancer types simultaneously. This pan-cancer training allows the models to leverage knowledge from a larger number of samples from the different cancer types instead of being limited to the samples available for only one specific cancer type. Two methods representing this category of cancer survival prediction will be introduced in Subsection 4.1.2.

4.1.1 Single-Cancer Survival Prediction

Random survival forests[88] are a well-known and widely used survival prediction method. They are based on the popular random forest (RF) method[23] and extend it by the ability to handle right-censored survival data. This is achieved by introducing splitting rules adapted to survival data when growing trees and modifying the random forest to predict the ensemble cumulative hazard function, which is computed by first calculating a cumulative hazard function for each tree of the survival random forest and then averaging over all trees. The cumulative hazard function, in turn, is estimated by the Nelson–Aalen estimator, which computes the cumulative proportion of deaths among individuals at risk summed over time. Random survival forests are commonly applied to predict cancer survival. For instance, Zhang et al.[190] compared random survival forests and Cox regression trained on clinical data of spindle cell carcinoma patients and identified survival prognostic features, while Dereli et al.[51] used random survival forests to predict cancer survival for different cancer types based on gene expression data and compared the results to their proposed cancer survival prediction method.

Survival support vector machines[156] are another established survival prediction method that can be used for single-cancer survival prediction. They extend classic support vector machines by regression on censored targets. To incorporate censored samples into the model, survival support vector machines disregard predicted survival times of censored samples that are larger than the respective censoring times when computing the loss during model training.

Dereli et al.[51] applied both random survival forests and survival support vector machines to compare their proposed Path2Surv survival prediction method. To this end, they trained and evaluated all three methods on gene expression data from 20 different cancer types. Their proposed cancer survival prediction method Path2Surv is a multiple-kernel learning method that is based on the aforementioned survival support vector machines. The main novelty of Path2Surv, which distinguishes it from survival support vector machines, is that it combines multiple kernels, each representing a molecular pathway or gene set, to predict cancer survival, thus incorporating prior biological knowledge into the learning process.

Another single-cancer survival prediction approach that incorporates prior knowledge into the learning process is the reweighted random survival forest[180]. This prior knowledge comes in the form of gene interaction information and is incorporated into the model by reweighting genes according to their topological importance. More specifically, reweighted random survival forests are trained on gene expression data, where topologically important genes receive high weights, thus biasing the model to select them as predictors with higher probability than topologically less important genes. According to the authors of reweighted random survival forests, the underlying assumption behind prioritizing topologically important genes in this way is that these genes often have important functions in disease development and show consistent gene expression variations across patients[180]. In the original publication introducing reweighted random survival forests[180], the method was applied to two cancer types, namely glioblastoma multiforme and esophageal squamous cell carcinoma, and topological importance values were derived from a global pathway network based on the KEGG[94] pathway database and a co-expression network based on the training gene expression data, respectively.

In 2016, Li et al.[112] proposed a somewhat different approach to survival prediction, where they formulated survival prediction as a multitask learning problem and tried to estimate the patients' survival times by predicting their survival status at predefined time intervals. This way, the survival prediction task is reformulated from a regression problem to a combination of multiple binary classification problems and the corresponding regularized optimization problem is solved to obtain survival coefficients. In the original publication[112], the multitask learning approach was applied to seven different cancer datasets, where each dataset comprised gene expression data of patients from one cancer type.

In addition to modifying traditional machine learning methods such as support vector machines or random forests to handle right-censored survival data, some works have also approached the problem of single-cancer survival prediction by applying neural networks. Three prominent examples of neural network-based cancer survival prediction methods are Cox-nnet, DeepSurv, and Cox-PASNet.

Cox-nnet[34] is a neural network consisting of two layers—one hidden layer with 143 nodes and one output layer with a single node—with the output layer implementing Cox regression. Thus, the output layer computes the relative risk compared to a non-parametric baseline for each patient. Cox-nnet uses the partial log-likelihood (cf. Chapter 3, Equation 3.26) to compute the loss and incorporates dropout to prevent overfitting. In the original publication[34], the method was applied for survival prediction on gene expression data of ten different cancer types from the TCGA.

DeepSurv[95] is a Cox proportional hazards deep feed-forward neural network, which, similar to Cox-nnet, implements Cox regression in its output layer and uses the negative partial log-likelihood with regularization as a loss function. In contrast to Cox-nnet, which has only one hidden layer, DeepSurv has up to three hidden layers depending on the dataset it is trained on and uses dropout in combination with L_2-regularization to prevent overfitting. Additionally, DeepSurv implements a treatment recommender system, where initially each patient is assigned to one treatment group and each treatment group is assumed to have an independent risk function. Then, after the model is trained, each patient can be passed through the network once in a treatment group i and again in a treatment group j, and the difference of log hazards for the different treatment options is computed. If this difference is positive, this means that treatment option i has a higher predicted risk of death than treatment option j and treatment j is recommended for the patient. Otherwise, if the difference is negative, treatment option i is recommended. DeepSurv was evaluated on simulated and real survival and treatment data[95], where the real survival data stemmed from three different studies: a study on heart attack survival and a study on survival of seriously ill hospitalized adults, both comprising clinical features, and one study on breast cancer survival with both clinical and gene expression features. However, from the last dataset, only a subset of four gene indicators was used as gene expression features and the remaining gene expression data was disregarded.

Cox-PASNet[69] combines neural networks with a priori biological information. More precisely, prior knowledge about biological pathways is explicitly incorporated into the architecture of the neural network by introducing sparse layers that represent genes and pathways. Thus, Cox-PASNet consists of an input gene layer, where each node corresponds to a gene and only genes belonging to at least one pathway are considered. The second layer of Cox-PASNet is the pathway layer, where each node represents a specific biological pathway and is only connected to nodes from the input layer that correspond with genes belonging to that pathway. The pathway layer is followed by multiple sparse hidden layers, whose output is combined with the output of a clinical layer that introduces clinical features into the model and forwarded to a single-node output layer, which—similar to DeepSurv—implements Cox regression with L_2-regularization. The sparsity in the hidden layers is achieved by initializing them to be fully connected in each training epoch, then using a dropout technique to randomly select and train a small sub-network, and applying sparse coding on the trained sub-network, where connections with an absolute weight that is below a layer-specific threshold are removed. To evaluate Cox-PASNet for cancer survival prediction, the method was applied to gene expression and clinical data from glioblastoma multiforme[69], which is an aggressive type of brain cancer.

4.1.2 Pan-Cancer Survival Prediction

In 2019, Cheerla and Gevaert introduced a multimodal neural network-based model for pan-cancer survival prediction[29]. Their method incorporates clinical data, gene and microRNA expression data, as well as histopathology whole slide images. A separate neural network per data modality is first used to extract feature vectors of length 512 for each modality. These modality-specific neural networks are trained using a representation learning framework. In this representation learning framework, a similarity loss is used to make the feature vectors extracted from the same patient but different data modalities similar, while driving the feature vectors corresponding to different patients apart. To predict cancer survival, the feature vectors extracted from the different data modalities are aggregated into a single representation vector of length 512, which is then fed to a prediction layer implementing Cox regression. The sum of similarity loss and Cox loss—computed as the negative partial log-likelihood—is used as an overall loss to train the model. During training, multimodal dropout is applied to make the model robust to missing data modalities. Multimodal dropout is a variation of the dropout technique where instead of dropping single neurons, whole feature vectors corresponding to one of the data modalities are randomly dropped with a pre-defined probability and the weights of the remaining modalities are scaled up accordingly. Cheerla and Gevaert evaluated their multimodal survival prediction model on single-cancer and pan-cancer data comprising 20 cancer types from TCGA and different combinations of data modalities, which always included clinical data. They found microRNA expression to be the most and gene expression to be the least informative data modality for pan-cancer survival prediction integrating all modalities. For single-cancer survival prediction, they found different combinations of data modalities to yield the best performance: for eight cancer types, the combination of all four considered data modalities was the most informative for survival, while for six cancer types, the combination of clinical data, microRNA, and histopathology whole slide images, excluding gene expression data, showed the best results. For all but one (KIRC) cancer type, they additionally found that pan-cancer training including all data modalities yielded superior results compared to single-cancer training on the same modalities.

In 2020, Vale-Silva and Rohr proposed MultiSurv[176], another multimodal deep learning method for pan-cancer survival prediction. MultiSurv integrates a pan-cancer dataset consisting of 33 different cancer types and multiple data modalities, including clinical data, histopathology microscopy slides, and different types of molecular data such as gene expression, microRNA expression, DNA methylation, and CNV data. Similar to the method proposed by Cheerla and Gevaert in 2019[29], the different data modalities are first individually fed to modality-specific sub-models, which are used for feature extraction. The outputs of all sub-models are then fused into a compact representation vector using a multimodal fusion procedure that is based on a multimodal keyless attention mechanism. Using this attention mechanism, the model can learn how much to focus on each of the modalities when fusing them into the representation vector. The representation vector is then fed to a six-layer fully connected neural network with one output node, which implements Cox regression and is optimized using the average negative partial log-likelihood as loss function, similar to the single-cancer neural network-based survival prediction methods. Analogously to Cheerla and Gevaert's multimodal survival prediction model[29], MultiSurv is also trained with multimodal input data dropout, where for every patient, each data modality is dropped with a certain probability during training and the values of the fused representation vector are scaled up to compensate for the missing data modality, allowing the model to handle missing data modalities. Vale-Silva and Rohr evaluated their proposed MultiSurv method on 33 cancer types from TCGA and different combinations of data modalities and found the combination of clinical, gene expression, and DNA methylation data to yield the best pan-cancer survival prediction results. Interestingly, and in stark contrast to Cheerla and Gevaert, who found gene expression to be the least informative modality[29], Vale-Silva and Rohr found gene expression to be the most predictive single data modality, followed by DNA methylation[176].

4.2 XGBoost

Our work is subdivided into two main parts: the identification of a pan-cancer survival network with gradient tree boosting and network propagation, and transfer learning for cancer survival prediction. In the first part, XGBoost[31] (cf. Section 3.2.1) is used to predict survival in individual cancer types and for a pan-cancer dataset from the TCGA database, followed by network propagation on the pan-cancer prediction results to identify a pan-cancer survival network. The XGBoost framework[31] has been successfully applied in a number of different biomedical prediction tasks. Here we introduce three interesting examples, namely the application of XGBoost in the diagnosis of chronic kidney disease[131], epilepsy detection based on language patterns identified from cerebral activity using XGBoost[174], and the prediction of the biological activity of molecular compounds[11].

Chronic kidney disease is characterized by a gradual loss of kidney function, including its ability to filter the bloodstream and dispose of metabolic waste[131]. It affects more than 10% of the population worldwide and 15% of the South African population[131]. Ogunleye and Wang from the University of Johannesburg, South Africa propose the use of XGBoost to diagnose chronic kidney disease from clinical features[131]. The prediction task is formulated as a binary classification problem, where patients can be classified as either healthy or ill. To solve this prediction task, Ogunleye and Wang considered different machine learning frameworks, including logistic regression, linear discriminant analysis (LDA), classification and regression tree (CART), support vector machine (SVM), k-nearest neighbor (KNN), and XGBoost, and found that without hyperparameter tuning, XGBoost showed the best prediction performance. Based on these results, they selected XGBoost for the disease diagnosis task and tuned its hyperparameters using grid search. To evaluate the model for diagnosing chronic kidney disease, a 10-fold cross-validation scheme was used. In addition to one XGBoost model using the full set of available features, Ogunleye and Wang trained and evaluated a second XGBoost model using only a subset of features, which had roughly half the size of the original feature set. To select the features from the full feature set, they applied three different feature selection methods, namely recursive feature elimination (RFE), extra tree classifier (ETC), and univariate selection (US), and retained features selected by at least two of the three methods. The evaluation of both models showed that the reduced model only using the subset of selected features matched the performance of the full model using the original feature set.

Another application of XGBoost in the biomedical field is the detection of epilepsy from language patterns based on cerebral activity. In 2017, Torlay et al. proposed the application of an XGBoost model on functional MRI (fMRI) data to identify atypical language patterns and classify subjects as healthy or patients with epilepsy[174]. The symptoms of focal epilepsy are caused by the lesion or dysfunction of a specific cerebral region, which is often located in the vicinity of language networks[174]. Additionally, brain networks involved in cognitive functions such as language show reorganization or plasticity in patients with focal epilepsy, leading to atypical language patterns, which can be mapped with fMRI[174]. Torlay et al. leveraged this characteristic and applied an XGBoost binary classification model to fMRI mappings of language networks to distinguish healthy individuals, who show typical language patterns, from epilepsy patients, who show atypical language patterns[174]. The XGBoost model was trained and evaluated using a random subsampling scheme, where in each of 12 replications, patients were randomly split into training and test sets and an inner 5-fold cross-validation for feature selection was performed on the training data to select the most predictive combination of 20 features derived from fMRI activation signal.

XGBoost has also been successfully applied in the field of drug discovery. This application was proposed by Mustapha and Saeed, who used XGBoost for bioactive molecule prediction[11] and showed that the method could outperform other machine learning methods like random forest, support vector machines, radial basis function neural network, and Naïve Bayes on the majority of evaluated datasets. They formulated the prediction of the biological activity of molecular compounds as a binary classification problem, where compounds were classified as either active or inactive. The prediction was based on quantitative descriptors of the compound's molecular structure, called molecular fingerprints, and was evaluated on seven datasets that had previously been used to validate molecular fingerprint-based molecule classification and activity prediction, showing very good validation accuracy of up to 98%.

4.3 Transfer Learning

In the second main part of this work, we seek to improve cancer survival prediction by the use of transfer learning. Transfer learning is based on the idea that knowledge from one domain with abundant data can be transferred to a different, but related domain to improve performance[191] (cf. Section 3.2.3). Here, we introduce three exemplary works that use the concept of transfer learning, either to build a general machine learning framework, such as incorporating transfer learning into the XGBoost algorithm[166], or to solve a specific prediction task, such as predicting drug sensitivity for anti-cancer compounds[139] or predicting cancer survival based on gene expression data[98].

TransBoost[166] is an extension of the XGBoost framework[31] implementing transfer learning for binary classification tasks with XGBoost boosting trees. For transfer learning, it combines a parallel tree structure with an instance weighting strategy. More precisely, TransBoost consists of two parallel XGBoost classification models, which are trained conjointly. They are constrained to share the same tree structure and split values, but can have different node weights. One of the two models, termed main boosting tree, is optimized on the combination of target domain and weighted source domain instances, while the other model, called ancillary boosting tree, is optimized on source domain instances only. To this end, the main model, which is also the final model that can be used to make predictions on the target domain, is trained using the sum of the loss on the target domain instances, a weighted loss on the re-weighted source domain instances, and a regularization term as the objective. The ancillary model, on the other hand, is trained to optimize a regularized loss on the unweighted source domain instances. The weights used for re-weighting the source domain instances in the main model are defined as the ratio between the joint distribution of the target domain and the joint distribution of the source domain and can be computed for each source domain instance at each training iteration based on the predictions of the main model and the ancillary model for the respective instance. Source domain instances that resemble the distribution of the target domain will receive high weights, while instances that differ more from the target domain distribution receive low weights. This way, the distribution discrepancy between source domain and target domain is minimized and knowledge from the source domain can be effectively transferred to improve model performance on the target domain. Thus, TransBoost implements an instance-based transfer learning approach (cf. Section 3.2.3).

Transfer learning has been successfully applied to several biomedical tasks, including drug sensitivity prediction and cancer survival prediction. Recently, Prasse et al. used a parameter-based (cf. Section 3.2.3) transfer learning approach for drug sensitivity prediction, where they first pre-trained a neural network on in vitro gene expression data and then fine-tuned the model on patient-derived gene expression data[139]. Large-scale drug sensitivity screening data, such as that provided by the Genomics of Drug Sensitivity in Cancer Database (GDSC), are typically generated by exposing cultured cancer cell lines to a variety of drug candidates[139]. However, these cancer cell lines have often been cultured for years or even decades under selective pressure in culture conditions and without interaction with other cell types, so they may no longer represent the molecular characteristics of the primary tumor well[139]. On the other hand, there are patient-derived model systems for assessing drug sensitivity, such as ex vivo cell cultures, patient-derived xenografts, or patient-derived organoids, which more closely resemble clinical tumors than cultured cell lines[139]. However, drug sensitivity screening is more complex in these model systems, resulting in a much lower availability of drug sensitivity data compared to cultured cell lines, which is not sufficient for training high-capacity machine learning models[139]. To address this problem, Prasse et al.[139] proposed a transfer learning approach, where first a neural network model was trained on gene expression data from the GDSC database to predict drug sensitivity, and then the pre-trained model was fine-tuned on a patient-derived drug sensitivity dataset. Three different neural network models were investigated for pre-training and fine-tuning: PaccMann, a state-of-the-art drug sensitivity prediction method, which uses prior knowledge of drug targets and network propagation on a PPI network for feature selection on the gene expression data and attention-based network modules to encode gene expression and drug information, tDNN, which is based on approximately 1,900 biologically relevant genes and uses two separate fully connected sub-networks for processing gene expression and drug information, respectively, before concatenating the output of both sub-networks and passing it through some additional fully connected layers, and a convolutional neural network, consisting of a single-layer gene expression sub-network and a convolutional drug sub-network, respectively, whose outputs are then concatenated and passed through some additional fully connected layers with batch normalization. The transfer learning approach was evaluated on four different target datasets used for fine-tuning, including cultured cell lines from the Cancer Cell Line Encyclopedia (CCLE), ex vivo cell lines from the Beat Acute Myeloid Leukemia program, a lung cancer xenograft dataset, and the Pancreatic Cancer Patient-derived Organoid dataset. For each model and each target dataset, two different settings were evaluated: the precision oncology setting, where the aim was to predict drug sensitivity of known drugs for new, previously unseen tumor cases, and the drug development setting, where drug sensitivity of known tumor cases to a new drug was predicted. Additionally, for each target dataset, it was investigated how the number of training samples would impact model performance by fine-tuning the pre-trained model on data subsets of different sizes. In the precision oncology setting, pre-training consistently improved prediction performance when up to 1,000 training samples from the target dataset were used for fine-tuning, while in most cases there was no significant improvement when more than 1,000 training samples were used. In the drug development setting, however, the benefit of pre-training was less dependent on the number of training samples from the target domain and improvements from pre-training were observed across the full range of sample sizes.

VAECox[98] is another parameter-based (cf. Section 3.2.3) transfer learning approach that is based on a neural network. It was introduced in 2020 by Kim et al. and uses transfer learning to transfer knowledge from pan-cancer RNA-seq gene expression data to predict survival in a specific cancer type. The training of VAECox consists of two steps: In the first step, a variational autoencoder (VAE) is trained on a pan-cancer gene expression dataset and in the second step, the trained weights from the encoder part of the VAE are transferred to a survival prediction model and fine-tuned to predict cancer survival for a single cancer type. The encoder consists of an input layer, one hidden layer, and a latent layer comprising a mean encoding component and a variance encoding component. Once trained on the pan-cancer pre-training dataset, the input layer, hidden layer, and mean encoding layer are combined with an additional hidden layer and a Cox-PH output layer for survival prediction, while the variance encoding layer is not transferred. The complete model is then trained on a single cancer type, using the negative partial log-likelihood as a loss function and fine-tuning the weights of the encoder layers. In the original publication, VAECox was evaluated on 10 different TCGA cancer types, while the gene expression VAE was pre-trained on a pan-cancer dataset comprising 20 TCGA cancer types. In addition to evaluating the survival prediction performance on the 10 cancer types, Kim et al. analyzed the hidden nodes of the VAECox model fine-tuned for breast cancer (BRCA) to find genes that were important for survival prediction. To this end, they extracted the hidden nodes with the highest variance from the second and third hidden layers of the model and computed the Pearson correlation between the extracted nodes and the expression of each gene across all breast cancer patients, deeming the genes with a high absolute correlation with the extracted hidden nodes as important.

5

Identification of a Pan-Cancer Survival Network with Gradient Tree Boosting and Network Propagation

This chapter introduces our approach for the identification of a pan-cancer survival network with gradient tree boosting and network propagation[*].

5.1 Motivation

The prediction of cancer survival is an important computational task in biomedical research and can be used to quantify patient risks and estimate prognoses. In fact, survival statistics are the most commonly used measure to estimate the prognosis of cancer patients[124], which has important implications for the choice of treatment. For instance, patients with a good prognosis might receive more aggressive therapies with the goal of remission, thereby accepting the occurrence of side effects, while patients with a poor prognosis might decide against aggressive therapies and favor palliative treatment to improve life quality instead[27]. Early methods for predicting survival include the Cox proportional hazards model[44], which can account for censored samples that often occur in survival data (cf. Section 3.3.2). However, the Cox proportional hazards model can only account for linear effects of covariates on the

[*] A major part of the work and results described in this chapter were published in[170] and[169]. This concerns in particular the single-cohort and pan-cancer survival prediction methods trained on gene expression data and the identification and analysis of the pan-cancer survival network. When describing this published work, we will refrain from repeatedly citing the aforementioned publications for the sake of readability.

log hazard function[153,165]. Nonlinear effects and interactions between covariates can only be considered if terms describing them are explicitly integrated into the model[153]. To overcome this limitation, we used the XGBoost machine learning framework in combination with the negative partial log-likelihood from the Cox proportional hazards model as a loss function to predict cancer survival. XGBoost[31] is a popular gradient tree boosting method (cf. Section 3.2.1) that has demonstrated good performance in various types of applications, including biomedical prediction tasks such as diagnosing chronic kidney disease[131] and identifying patients with epilepsy based on cerebral activity[174] (cf. Section 4.2). To the best of our knowledge, however, we were the first to apply XGBoost with negative partial log-likelihood to pan-cancer survival prediction based on gene expression data.

Modern machine learning methods like XGBoost or neural networks often show much better prediction performance than traditional machine learning methods like linear models or decision trees[2,115]. However, this improved prediction performance can often only be achieved by increased model complexity, which makes the model's decisions harder to understand and the predictions more difficult to interpret[115]. While several of the existing survival prediction methods introduced in Chapter 4 have addressed the issue of model interpretability either by analyzing the trained model with respect to important features or by directly integrating prior knowledge into the model, there are some shortcomings. For example, the authors of VAECox[98] (cf. Section 4.3) investigated their trained neural network by computing the correlation between hidden nodes and gene expression features, considering genes with high correlation to highly variable hidden nodes as important. However, hidden nodes can reflect more or less complex interactions between multiple features, where sometimes multiple hidden nodes together represent the same interaction or different interactions involve the same feature. In these cases, the correlation between a feature and any single hidden node might be rather moderate, even if this feature is highly important for the output of the neural network and contributes to several hidden nodes. On the other hand, another feature might be less important for the prediction of the neural network, but highly correlated to a single hidden node. In the model interpretation strategy applied to VAECox, the first feature would be considered less important than the second feature due to its smaller correlation with any single hidden node, even though it has a larger effect on the model's prediction. Instead of analyzing the trained model post hoc, other cancer survival prediction methods like Path2Surv[51], reweighted random survival forests[180], or Cox-PASNet[69] (cf. Section 4.1) directly incorporate prior biological knowledge to enable interpretation. While models following this approach are inherently more easily interpretable because their architecture directly reflects biological concepts such as pathways, this architecture might also lead to a loss of potentially valuable information: On the one hand, the model typically only contains features for which prior knowledge, e.g. in the form of pathway membership, is available and will completely disregard potentially informative features that are not contained in the respective dataset used as the source of prior knowledge. For instance, the KEGG pathway database used by reweighted random survival forest and Cox-PASNet contains only ∼8,000 unique genes and the Hallmark gene sets used by Path2Surv contain ∼4,000 unique genes, which is only a fraction of the total number of genes for which gene expression data is available in databases like TCGA. On the other hand, not only the number of features used by the model, but also the modeled interactions between features can be limited by integrating prior knowledge into the model architecture. For example, Path2Surv's[51] architecture is based on multiple kernels, each representing a separate pathway or gene set, preventing the model from learning interactions between genes that do not share a common pathway or gene set, even if they are actually interacting through another type of interaction not reflected in the pathway membership information, such as a protein-protein interaction.

To show the biological plausibility of our cancer survival prediction method without a priori restricting the features to those contained in, for example, a specific pathway database, we combined the XGBoost model with post hoc network propagation on a comprehensive PPI network. More specifically, after training for cancer survival prediction, we extracted feature importance scores produced by the XGBoost model and used them as input to the NetCore[13] network propagation method. In this way, we hoped to identify a subnetwork of the PPI network with a high association with cancer survival and to gain a better understanding of the underlying biological mechanisms.

5.2 Methods

In this section, we describe the methodology used to predict cancer survival and to identify a pan-cancer survival network using network propagation based on the important features identified during survival prediction.

5.2.1 Data and Preprocessing

The survival prediction and network identification described in this chapter are based on molecular and clinical data from the TCGA consortium (https://www.cancer.gov/tcga). TCGA comprises molecular and clinical data from more than 10,000 cancer patients and for 33 different types of cancer, which originate from a wide variety of organ systems[133]. The data used in this work was retrieved from the Genomic Data Commons (GDC) data portal (https://portal.gdc.cancer.gov/). For the single-cohort and pan-cancer survival prediction and the identification of the pan-cancer survival network, RNA-seq gene expression data normalized as fragments per kilobase of transcript per million fragments mapped (FPKM) and corresponding clinical data, including survival or censoring time for each patient, was used (see Supplementary Table B.1 for more details on the used TCGA cancer cohorts). We decided to use FPKM-normalized gene expression data to ensure comparability between our survival prediction method, random survival forest, survival support vector machine, and the Path2Surv multiple-kernel learning (MKL) method, which is based on

FPKM-normalized gene expression data. For each cancer type, all HTSeq-FPKM files and corresponding clinical files were downloaded from the GDC data portal. For the cohorts TCGA-COAD, TCGA-LAML, TCGA-LUAD, and TCGA-LUSC, all data was obtained from GDC data release v22.0 (released January 16, 2020) and for the 29 remaining cohorts, all files were retrieved from GDC data release v24.0 (released May 7, 2020). For further analyses, which included evaluating the integration of additional data modalities like mutation, copy number variation, and protein expression data and the inclusion of information on tumor status into the survival prediction, we used gene expression data normalized as transcripts per million (TPM) instead of FPKM-normalized gene expression data. In TPM-normalized expression data, the sum of gene expression values over all genes is equal in each sample or patient and thus patients should—at least in theory—be more comparable in TPM-normalized data. Since the GDC database underwent a major update in 2022, in which gene expression files processed by HTSeq were replaced with files processed by STAR, STAR-TPM gene expression and corresponding clinical data from the GDC data release v32.0 (released March 29, 2022, downloaded with TCGAbiolinks R package[40,128,159]) were used for further analyses. For the evaluation of mutation, copy number variation, and protein expression data as additional data modalities for cancer survival prediction, all corresponding data for the analyzed cancer types was downloaded with the TCGAbiolinks R package from GDC data release v.32.0, analogously to the STAR-TPM gene expression data. Mutation data consisted of simple somatic nucleotide variations such as point mutations, missense mutations, nonsense mutations, and insertions and deletions (indels) and was downloaded from GDC as mutation annotation format (MAF) files containing masked somatic mutations, which are a filtered subset of somatic mutations with potential germline and lower quality variants removed (cf. `https://docs.gdc.cancer.gov/Data/File_Formats/MAF_Format/`). Copy number variation data contained integer gene level copy numbers computed as the weighted median of copy number values of all copy number segments overlapping with a gene (cf. `https://docs.gdc.cancer.gov/Data/Bioinformatics_Pipelines/CNV_Pipeline/`) and protein expression data was measured by RPPA (cf. Section 2.2.3 and `https://docs.gdc.cancer.gov/Data/Bioinformatics_Pipelines/RPPA_intro/`).

For training the survival prediction models in each of the evaluated setups, we only used TCGA cohorts with at least 20 uncensored patients, leaving 25 (ACC, BLCA, BRCA, CESC, COAD, ESCA, GBM, HNSC, KIRC, KIRP, LAML, LGG, LIHC, LUAD, LUSC, MESO, OV, PAAD, READ, SARC, STAD, SKCM, UCEC, UCS, and UVM) of the 33 TCGA cancer cohorts. We decided to exclude cohorts with less than 20 uncensored patients from model training since splitting these cohorts into 80% training and 20% test data would result in evaluating model performance on test data with no more than four uncensored patients, limiting the meaningfulness of the evaluation. The eight TCGA cohorts with less than 20 uncensored patients (CHOL, DLBC, KICH, PCPG, PRAD, TGCT, THCA, and THYM) were not used for survival model training, but only for evaluating the transferability of the pan-cancer XGBoost survival prediction model to new cancer types not seen during training.

For all analyses, we only used primary tumor and primary blood-derived cancer samples and excluded samples derived from normal tissues and metastatic tumors since these sample types are expected to have different molecular characteristics compared to primary cancer samples. In the case of multiple tumor samples from the same patient, we selected the sample with the lexicographically highest sample ID, assuming that this sample ID corresponded to the most recent sample and reasoning that there might have been reasons to re-sample from the same patient, making the most recent sample the most reliable. Furthermore, we excluded all patients for whom the molecular data modality of interest (e.g., gene expression) was not measured or for whom key clinical data like vital status, age, gender, or time to either death or censoring was missing or inconsistent. For the HTSeq FPKM-normalized gene expression data used in the first part of this work, this resulted in a total of 8,024 patients from the 25 different cancer cohorts that were used for model training and an additional 1,571 patients from the eight remaining cohorts that were not used for model training due to small numbers of uncensored patients. The gene expression data comprised 60,483 RNA molecules measured in TCGA for all cohorts—including protein-coding genes, processed pseudogenes[30], and lncRNAs[162], among others. From here on we will use the term 'gene' for all of these molecule types and not only for protein-coding genes. In the GDC update from 2022 (GDC data release v32.0), patient and gene numbers for the STAR TPM-normalized gene expression data increased slightly to 8,045 patients and 60,616 genes from the 25 cancer cohorts. Mutation data from GDC data release v32.0 comprised 17,975 genes mutated in a total of 7,975 patients. For copy number variation (CNV) data, in addition to the filtering steps described above, we removed CNVs located on the Y chromosome prior to survival prediction training, resulting in CNVs affecting 59,754 genes in 8,955 patients. Lastly, for protein expression data, TCGA data comprised 487 proteins measured in 6,256 patients from 24 of the 25 TCGA cohorts (there was no protein expression data available for TCGA-LAML), making protein expression the sparsest of the investigated data modalities.

5.2.2 Single-Cohort and Pan-Cancer Survival Prediction with XGBoost

We applied the XGBoost gradient tree boosting framework[31] to predict cancer survival in the form of Cox proportional hazards risk scores (cf. Section 3.3.2) from gene expression data in a single-cohort setting, where a separate model was trained for each of the 25 analyzed TCGA cancer cohorts, and in a pan-cancer setting, where the XGBoost model was trained on gene expression data from all 25 TCGA cancer cohorts jointly[170]. To this end, we used the Python XGBoost package (`https://xgboost.readthedocs.io`) with the learning objective set to Cox proportional hazards regression with negative partial log-likelihood (cf. Section 3.3.2). In both settings—single-cohort and pan-cancer—we repeated model training and evaluation 100 times with different splits of patients into training and test data to ensure a robust and reliable assessment of prediction performance in the respective setting. In each of these 100 replications, we randomly split the patients and their corresponding gene expression and survival data into 80% training and 20% test data using a stratified splitting strategy to ensure that the percentage of censored and uncensored patients in the training and test data was approximately the same in all replications. In the pan-cancer setting, we additionally ensured that the cohort composition of training and test data remained the same across replications by assigning 80% of patients from each cohort to the training data and 20% to the test data in each replication.

Then, in each replication, a survival prediction model was constructed and trained on the training data and evaluated on the test data. The training procedure (visualized in Figure 5.1) comprised three main steps: feature selection, hyperparameter tuning, and training of the XGBoost model.

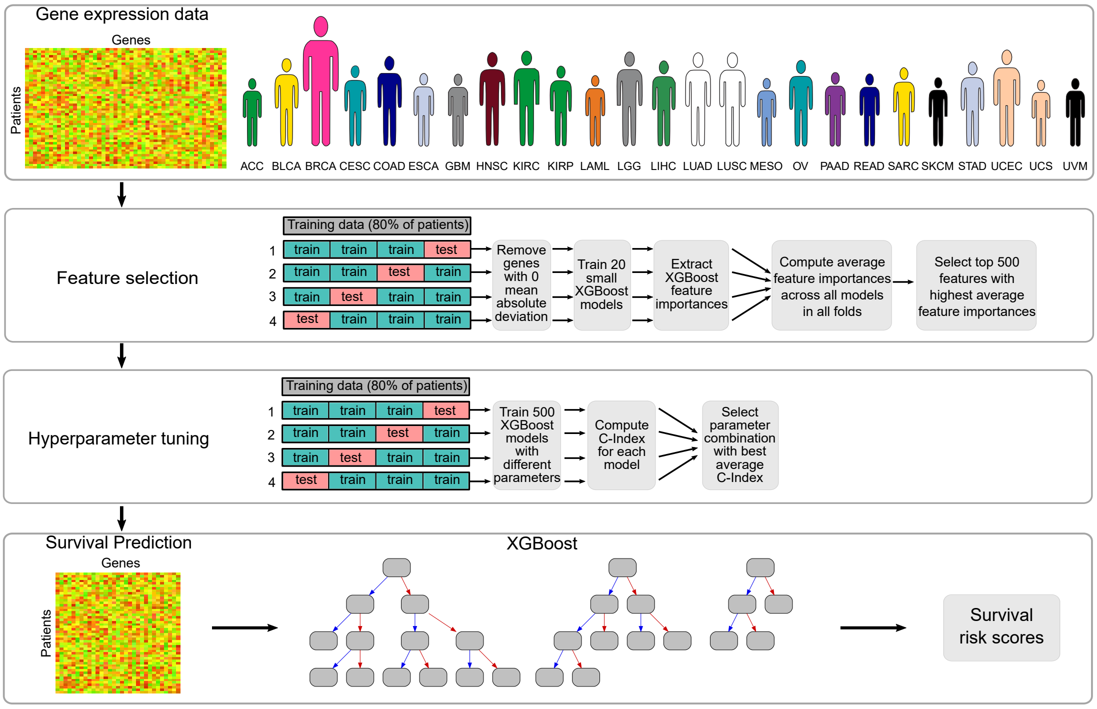

Gene expression data
Genes
Patients
ACC
BLCA
BRCA
CESC
COAD
ESCA
GBM
HNSC
KIRC
KIRP
LAML
LGG
LIHC
LUAD
LUSC
MESO
OV
PAAD
READ
SARC
SKCM
STAD
UCEC
UCS
UVM
Feature selection
Training data (80% of patients)
1 train train train test
2 train train test train
3 train test train train
4 test train train train
Remove genes with 0 mean absolute deviation
Train 20 small XGBoost models
Extract XGBoost feature importances
Compute average feature importances across all models in all folds
Select top 500 features with highest average feature importances
Hyperparameter tuning
Training data (80% of patients)
1 train train train test
2 train train test train
3 train test train train
4 test train train train
Train 500 XGBoost models with different parameters
Compute C-Index for each model
Select parameter combination with best average C-Index
Survival Prediction
Genes
Patients
XGBoost
Survival risk scores

Figure 5.1: Method outline of the XGBoost training procedure. The XGBoost training procedure is based on a *patient* × *gene expression* matrix comprising patients from either one cancer cohort (single-cohort approach) or multiple cancer cohorts (pan-cancer approach). From this gene expression matrix, a subset of 500 genes is selected in a feature selection step that includes a 4-fold cross-validation on the training data, in which small XGBoost models are trained on the complete set of features and the features with the highest average feature importance scores across models are selected. Next, another 4-fold cross-validation on the training data with reduced features is performed to tune the model hyperparameters and finally, a survival prediction model is trained on the reduced gene expression matrix of 500 genes and with the optimized model hyperparameters. This figure was published in[170].

In the feature selection step, an embedded feature selection approach (cf. Section 3.2.4) was implemented to reduce the number of gene expression features used to train the XGBoost survival prediction model in each training replication from 60,483 genes measured in TCGA to 500 genes that are informative for survival. Embedded feature selection uses the same machine learning method as chosen to solve the prediction task (XGBoost in this case) to evaluate and select features from a set of candidate features based on built-in feature importance measures[146]. To identify genes that are informative for survival more generally and not only for a specific training set composition, we integrated a stratified 4-fold cross-validation on the training data into the feature selection step. To this end, the training data was first split into four subsets or folds in a stratified manner such that the percentages of censored and uncensored patients were approximately equal in all folds. In each of the four cross-validation steps, three folds were used for training, while one fold was held out. Then, genes with zero mean absolute deviation (MAD) in the three training folds were removed because these genes are not informative for cancer survival. Next, 20 XGBoost survival prediction models with limited model size (maximum number of trees between 5 and 20, maximum tree depth between 1 and 3) and different sets of model hyperparameters were trained on the three training folds and feature importance was measured in terms of 'gain', which is the average improvement the respective feature adds to the evaluation metric across all decision tree splits in which it is used (cf. `https://xgboost.readthedocs.io/en/latest/python/python_api.html`) and hence measures the relative importance of this feature for the model's prediction. We limited model size in order to reduce runtime, which increases proportionally to the number of features and model size, to avoid overfitting on the large number of features of the training data, and to force each model to select only the most informative genes as features. Furthermore, we trained models with 20 different sets of randomly selected hyperparameter configurations in each cross-validation step to identify genes whose feature importance does not depend on the yet untuned model hyperparameters, but which have high feature importance under different sets of hyperparameters. Finally, we calculated a feature importance score for each candidate gene by averaging the computed feature importance scores across all 20 models per cross-validation step and across all four cross-validation steps and selected the top 500 genes with the highest average feature importance as features for training the final survival prediction model.

In the subsequent hyperparameter tuning step, the goal was to find XGBoost model hyperparameters—including the maximum tree depth, number of trees, and regularization parameters—that optimized the survival prediction performance. To this end, we first randomly generated 500 combinations of hyperparameters and introduced another 4-fold cross-validation scheme—analogous to the cross-validation in the feature selection step—on the training data to evaluate each hyperparameter combination. In each step of the 4-fold cross-validation, we trained one XGBoost model per hyperparameter combination on the three training folds using the 500 genes selected in the feature selection step as input features and evaluated the survival prediction performance of the model on the remaining fold in terms of concordance index (C-Index, cf. Section 3.3.3). To select the best hyperparameter combination, we then averaged the C-Indices obtained for each hyperparameter combination across the four cross-validation steps and selected the hyperparameter combination that showed the best average concordance index.

In the last step of the training procedure, the whole training data (80% of patients) was used to train a final XGBoost survival prediction model with the hyperparameters identified in the hyperparameter tuning step and based only on the 500 features selected in the feature selection step. The fully trained survival prediction model was evaluated on the held-out test data (20% of patients) by computing the C-Index on the test patients from each TCGA cancer cohort.

Thus, across the 100 model replications, we trained 100 independent models with training-data-specific sets of gene features and hyperparameters and obtained 100 C-Indices for each of the 25 analyzed TCGA cancer cohorts.

5.2.3 Comparison of the XGBoost Survival Prediction Method with Other Methods

In order to evaluate the survival prediction performance of our XGBoost-based method not only in isolation in terms of the C-Index, but also in comparison to other established methods, we compared the single-cohort XGBoost method against random survival forest[88], survival support vector machine[97,156], and the MKL method Path2Surv[51].

Random Survival Forest

Random survival forest[88] is a widely used random forest (RF) method specifically designed to handle right-censored survival data (cf. Section 4.1.1). RFs are ensemble tree methods, where a model consists of multiple decision trees and a prediction is generated by averaging over the trees[88]. Each tree is built based on a randomly drawn bootstrap sample of the data and for each tree node, a randomly drawn subset of features or covariates is selected as candidate variables to split samples on. To be able to handle right-censored survival data, random survival forest incorporates two key mechanisms into the RF method[88]: Firstly, when growing a tree, in each node, the covariate that maximizes the survival difference between the child nodes is selected as split variable, and tree growth is constrained in that each terminal node must contain at least $d_0 > 0$ unique deaths from the bootstrap sample of the data on which the tree is grown. Secondly, random survival forest calculates an ensemble cumulative hazard function (CHF) as the prediction output. The ensemble CHF is the average of the CHFs of all terminal nodes $\mathcal{T}$, where the CHF of a terminal node $h \in \mathcal{T}$ can be estimated by the Nelson-Aalen estimator. The Nelson-Aalen estimator represents the cumulative rate of expected deaths up to a time point t and for a terminal node $h \in \mathcal{T}$ takes the form[88]:

$$\hat{H}_h(t) = \sum_{t_{l,h} \leq t} \frac{d_{l,h}}{Y_{l,h}} \tag{5.1}$$

with l indexing time points $t_{l,h}$ that correspond to terminal node h and are smaller than or equal to the reference time point t, $d_{l,h}$ being the number of deaths at $t_{l,h}$ and $Y_{l,h}$ being the number of individuals at risk at $t_{l,h}$.

For the comparison between our XGBoost-based survival prediction method and random survival forest, we used the R implementation from Dereli et al.[51], which trains and evaluates the random survival forest and is based on the *randomForestSRC* package. Using this implementation, we evaluated the performance of random survival forests in 100 model replications for each TCGA cancer cohort. In each replication, the data of the respective cancer cohort (comprising all 60,483 genes measured in TCGA) was log2-transformed, genes with a standard deviation of zero were removed and the data was randomly split into 80% training and 20% test data. Next, the training data was normalized to zero mean and unit standard deviation, with the test data being normalized accordingly and the number of trees was tuned in a 4-fold cross-validation on the training data (range between 500 and 2,500 trees), while all other hyperparameters were kept as default. Then, the final survival prediction model was trained with the optimal number of trees and evaluated on the held-out test data using the C-Index.

Survival Support Vector Machine

Survival support vector machine[156] is a survival prediction method based on support vector machines (SVMs). SVMs are linear models that use mathematical kernel functions mapping input data to a higher-dimensional feature space to model not only linear, but also nonlinear relationships between input features and a target variable by encapsulating any nonlinearities in the kernel functions and thus transforming the nonlinear prediction problem to a linear problem[97]. In support vector regression, the SVM tries to learn the relationship between input features $x \in \mathbb{R}^p$ and a continuous target variable $y \in \mathbb{R}$[97,18]:

$$f(x) = w^\top \phi(x) + b, \tag{5.2}$$

where $f(x)$ is the function that best fits the training data, $k(x, x') = \phi(x)^\top \phi(x')$ is a kernel function with $\phi : \mathbb{R}^p \rightarrow \mathbb{R}^m$, $w \in \mathbb{R}^m$ is a weight vector, and $b \in \mathbb{R}$ is a bias. To find the optimal w and b, the following optimization problem is solved[97,18]:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} (\xi_i + \xi_i^*) \tag{5.3}$$

subject to

$$y_i - (w^\top \phi(x_i) + b) \leq \varepsilon + \xi_i \tag{5.4}$$

$$(w^\top \phi(x_i) + b) - y_i \leq \varepsilon + \xi_i^* \tag{5.5}$$

$$\xi_i, \xi_i^* \geq 0, \quad i = 1, \ldots, n \tag{5.6}$$

with n being the number of training samples, ξ_i and ξ_i^* being so-called slack variables for each data point $i = 1, ..., n$ that allow for model constraints to be violated and thus make it possible to solve otherwise unsolvable optimization problems, $C > 0$ being a regularization parameter that determines the trade-off between the minimization of the training error and the control for model complexity, and $\varepsilon > 0$ being a margin parameter that defines a threshold below which the prediction error is considered insignificant and thus influences model complexity.

To enable survival prediction with SVMs, this optimization problem can be modified to be able to handle censored patients[156,51]. To this end, a censoring indicator δ_i is introduced into Equation (5.3) of the optimization problem, with $\delta_i = 1$ indicating that patient i is censored and $\delta_i = 0$ meaning the patient is uncensored. This way, the model does not consider predicted survival times that are larger than the censoring times as errors for censored patients and the optimization problem becomes:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} (\xi_i + (1 - \delta_i)\xi_i^*) \tag{5.7}$$

subject to

$$y_i - (w^\top \phi(x_i) + b) \leq \varepsilon + \xi_i \tag{5.8}$$

$$(w^\top \phi(x_i) + b) - y_i \leq \varepsilon + \xi_i^* \tag{5.9}$$

$$\xi_i, \xi_i^* \geq 0, \quad i = 1, ..., n \tag{5.10}$$

For comparing our XGBoost-based survival prediction method with survival support vector machines, we adapted the R implementation from Dereli et al.[51]. Since the CPLEX optimization algorithm[86] used in the original implementation is not openly accessible, we used the R Optimization Infrastructure (ROI)[171] instead to solve the survival SVM optimization problem. According to the implementation of Dereli et al., we evaluated the performance of survival SVM in 100 model replications for each TCGA cancer cohort. In each replication, the data of the respective cancer cohort (comprising all 60,483 genes measured in TCGA) was log2-transformed, genes with a standard deviation of zero were removed and the data was randomly split into 80% training and 20% test data[51]. Then, the training data was normalized to zero mean and unit standard deviation, and the test data was normalized accordingly. The regularization parameter C was tuned in a 4-fold cross-validation on the training data (range between 1×10^4 and 1×10^5), ε was kept at 0 and the Gaussian kernel defined as

$$k_{\mathcal{G}}(x, x') = \exp\left(-\frac{(x - x')^\top (x - x')}{2\sigma^2}\right) \tag{5.11}$$

with kernel width parameter σ set to the mean of pairwise Euclidean distances between training samples was used as kernel function. At the end of each replication, the final survival prediction model was trained with the optimal C and evaluated on the test data using the C-Index.

Path2Surv

Dereli et al.[51] developed the Path2Surv MKL method for survival prediction, which is based on survival support vector machines. However, instead of using a single kernel function like in survival support vector machines, Path2Surv combines multiple kernels in a weighted sum to learn the relationship between input features and the survival outcome. Each kernel $k_p(x, x'), p = 1, ..., P$ (with P being the total number of kernels) represents a molecular pathway or gene set and comprises all genes constituting that pathway or gene set as features. Path2Surv tries to learn a non-negative kernel weight η_p for each of these kernels, where the sum of all kernel weights is required to sum up to one and kernels can have a weight of zero to exclude pathways or gene sets that are not informative for survival from the prediction. This way, Path2Surv uses fewer gene expression features for survival prediction than random survival forest or survival support vector machine and also offers interpretability in terms of which pathways or gene sets are most relevant for survival prediction. Learning the kernel weights $\eta \in \mathbb{R}^P$ is considered as an outer optimization problem, while learning the weights and biases for each kernel as in the survival support vector machine formulation in Equations (5.7)–(5.10) is regarded as an inner optimization problem J. Accordingly, during model training, the following outer optimization problem is solved:

$$\min_{\eta} J(\eta) \tag{5.12}$$

subject to

$$\sum_{p=1}^{P} \eta_p = 1 \tag{5.13}$$

$$\eta_p \geq 0 \quad p = 1, ..., P, \tag{5.14}$$

where $J(\eta)$ represents the inner optimization problem with $\sum_{p=1}^{P} \eta_p k_p(x, x')$ replacing the kernel function $k(x, x')$.

In the publication of Path2Surv[51], the authors report results on two different gene sets and pathway databases, namely the Hallmark gene sets[113] and the Pathway Interaction Database (PID)[149]. We compared both versions of the Path2Surv method with our XGBoost-based survival prediction method. The comparison is analogous to comparing our XGBoost-based method with random survival forest and survival support vector machine. As for survival SVM, we replaced the CPLEX optimization algorithm[86] in the original implementation of Path2Surv by ROI[171] and evaluated the performance of Path2Surv in 100 model replications for each TCGA cancer cohort. In each replication, the data of the respective cancer cohort (comprising all 60,483 genes measured in TCGA) was log2-transformed, genes with a standard deviation of zero were removed and the data was randomly split into 80% training and 20% test data[51]. Then, the training data was normalized to zero mean and unit standard deviation, and the test data was normalized accordingly. The regularization parameter C was tuned in a 4-fold cross-validation on the training data (range between 1×10^4 and 1×10^5), ε was kept at 0 and the Gaussian kernel defined in Equation (5.11) with kernel width parameter σ set to the mean of pairwise Euclidean distances between training samples was used as kernel function for each gene set or pathway. At the end of each replication, the final survival prediction model was trained with the optimal C and evaluated on the test data using the C-Index.

5.2.4 Computation of Gene Weights for the Analysis of Important Features

To identify and analyze genes that are informative for cancer prognosis, we computed gene weights, which summarize the importance of genes for cancer survival prediction with XGBoost across model replications. To this end, feature importance scores, which reflect the relative importance of a gene for the prediction of cancer survival, were extracted in each replication of single-cohort or pan-cancer model training from the respective XGBoost model. These feature importance scores are directly provided by the XGBoost implementation and measure the 'gain' of each feature used by the trained XGBoost model, which is the average improvement the feature adds to the evaluation metric across all decision tree splits in which it is used. Using these gene-specific feature importance scores, we then computed a gene weight for each gene used in at least one model replication as the sum of feature importance scores over all 100 model replications, where a feature importance of zero is assumed for model replications in which the respective gene was not used in the model to predict survival.

5.2.5 Entropy Measurement for Cancer Type Specificity Analysis of Genes

In information theory, entropy measures the uncertainty or information content of a random variable[154]. It is often also referred to as Shannon entropy after Claude E. Shannon, who in 1948 introduced the following definition of entropy[154]:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2(P(x_i)), \qquad (5.15)$$

where $P(x_i)$ is the probability of outcome x_i of a random variable X with possible outcomes $x_1, \ldots, x_n$ and the logarithm has basis 2 for information measured in bits. That is, the entropy is maximal when all outcomes occur with the same probability, while it is minimal when only one of the outcomes occurs with certainty and all other possible outcomes have a probability of zero.

We have adopted the concept of entropy to evaluate how well genes identified as important features in our XGBoost approach generalize as features for predicting survival in different cancer types. In the XGBoost approach, we calculated gene weights by summing over feature importance scores from the 100 model replications for every gene. In the single-cohort XGBoost approach, this was done for each of the 25 TCGA cancer cohorts separately, such that each gene received one weight per cancer cohort, while in the pan-cancer approach, where all cancer types were combined for model training, only one weight per gene was computed. To compute the entropy of genes across cancer types, we constructed a gene weight matrix containing gene weights computed from the single-cohort approach for all genes and all 25 cancer cohorts and converted this gene weight matrix into a probability matrix by dividing each weight value by the sum of scores for this gene over all 25 cohorts. Then, we computed the entropy of each gene across cohorts according to Equation 5.15 by using the computed probability score. Thus, genes identified as important features for survival prediction with similar gene weights across all 25 cancer cohorts would have high entropy, while genes that were only predictive for cancer survival in one of the 25 cohorts would have a minimal entropy of zero. Therefore, the entropy of a gene with respect to the single-cohort XGBoost feature importance scores can be used to assess how well a gene generalizes as a survival prediction feature across cancer cohorts, in that genes with high entropy can be considered as predictive of cancer survival across different cancer types, while a low entropy means that a gene is likely to be cancer-type-specific.

5.2.6 Pan-Cancer Survival Network Identification

To further assess the biological plausibility of the important features identified in the pan-cancer XGBoost survival prediction approach, we used network propagation to infer a pan-cancer survival network (method outline shown in Figure 5.2). To this end, we applied the NetCore[13] network propagation method (cf. Section 3.4.1) over the high-confidence CPDB (version 34) PPI network[82,93], which we initialized with gene weights derived from the pan-cancer XGBoost approach. To map the gene weights onto the PPI network, we downloaded the high-confidence CPDB PPI network, which contained $114,341$ binary PPIs with interaction confidence > 0.95 and proteins mapped to $10,586$ Hugo Gene Symbols, from the NetCore GitHub repository (`https://github.molgen.mpg.de/barel/NetCore`) and converted Ensembl Gene Identifiers of the gene weights to Hugo Gene Symbols using the MyGene Python package (version 3.1, `http://mygene.info`)[186,187], removing gene entities that did not map to a Hugo Gene Symbol. Then, network propagation based on RWR was performed on the gene weight-initialized PPI network using NetCore with the default restart probability of 0.8. In addition to network propagation, NetCore also implements a subsequent semi-supervised module identification step, where phenotype-associated net-

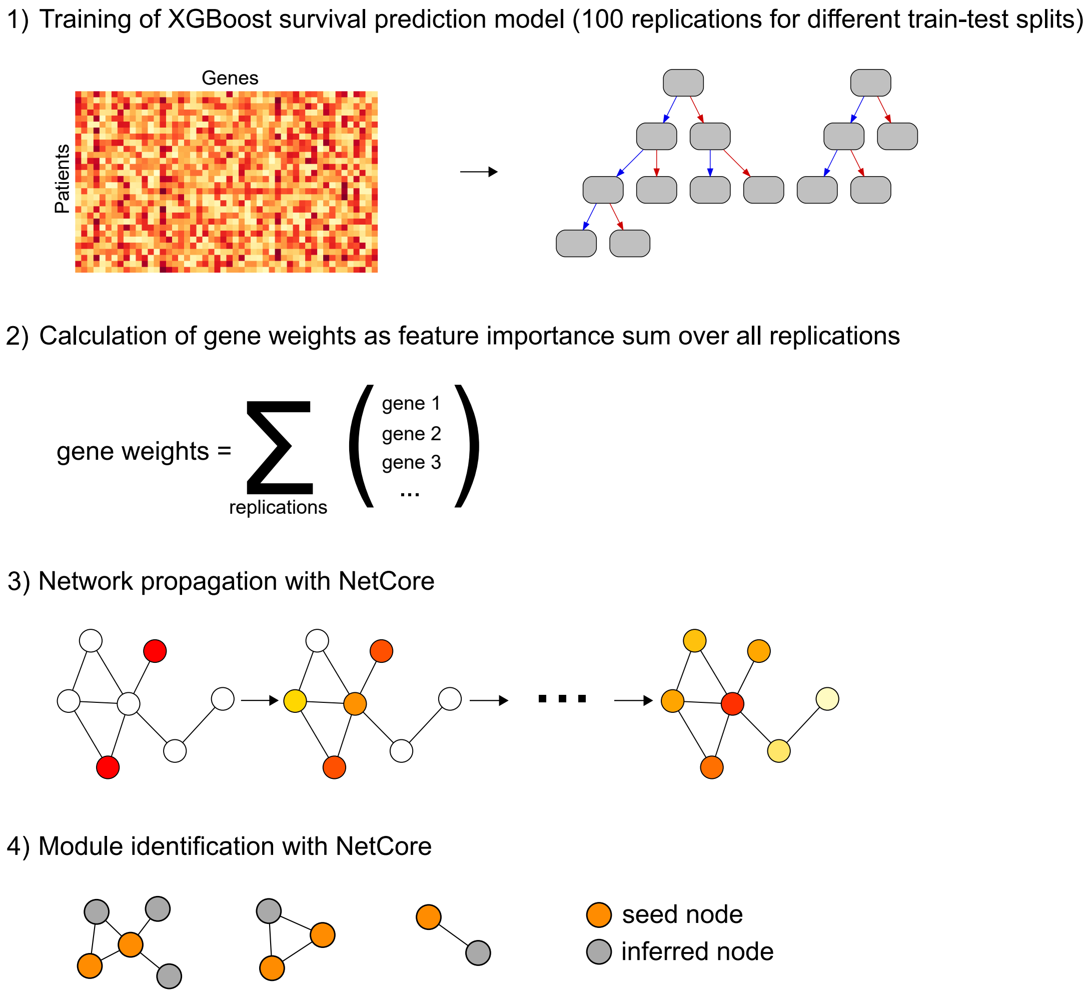

1) Training of XGBoost survival prediction model (100 replications for different train-test splits)
Genes
Patients
2) Calculation of gene weights as feature importance sum over all replications
gene weights =
replications
gene 1
gene 2
gene 3
...
3) Network propagation with NetCore
...
4) Module identification with NetCore
seed node
inferred node

Figure 5.2: Outline of the survival network identification. After 1) training 100 replications of the pan-cancer XGBoost survival prediction method on different train-test splits of the patients, 2) feature importance scores for each gene were extracted from each trained model and gene weights were computed as the sum of feature importance scores over the model replications. These gene weights were then used to 3) initialize a high-confidence PPI network and NetCore[13] was used to perform network propagation on the PPI network and 4) to identify network modules.

work modules are identified. These modules are sub-networks of the PPI network comprising seed nodes, which in this case are the top 100 genes with the highest initial gene weights represented in the PPI network, and inferred nodes, which are genes that function as links between seed nodes and have been identified as significant in the network propagation step. Taken together, all of the network modules identified by NetCore based on the XGBoost-derived pan-cancer gene weights form a pan-cancer survival network, i.e., a sub-network of the PPI network that is presumably associated with patient survival across different cancer types.

5.2.7 Over-Representation Analysis of the Pan-Cancer Survival Network

To further analyze the pan-cancer survival network identified by NetCore's network propagation and module identification with respect to biological function, we performed an ORA (cf. Section 3.6) using QUIAGEN's Ingenuity Pathway Analysis (IPA) software[104].

5.2.8 Implementation

Our cancer survival prediction method is based on the Python XGBoost package (`https://github.com/dmlc/xgboost/tree/master/python-package`). All steps of the method, including feature selection, hyperparameter tuning, and training of the final survival prediction model, were implemented in Python (release 3.7). Based on gene weights derived from the pan-cancer XGBoost survival prediction method trained on gene expression data, we conducted network propagation using NetCore[13] to identify a pan-cancer survival network. All experiments, including model training for survival prediction and network propagation, were performed on Linux servers. All corresponding code for the survival prediction based on gene expression data and the processing of the results to use with NetCore is available in the following GitHub repository: `https://github.molgen.mpg.de/thedinga/xgb_survival_network`. A protocol detailing all steps necessary to train the XGBoost pan-cancer survival prediction approach on gene expression data and to derive the pan-cancer survival network through network propagation was published in STAR Protocols[169].

5.3 Results

This section describes the results that were obtained for survival prediction with XGBoost in different settings and the pan-cancer survival network identified through network propagation based on the important features of the XGBoost method.

5.3.1 XGBoost Gradient Tree Boosting Predicts Cancer Survival in Different Cancer Types

As described in Section 5.2.3, we compared our single-cohort XGBoost survival prediction method, in which XGBoost survival prediction models were trained for each of the 25 analyzed TCGA cancer cohorts (cf. Section 5.2.1) separately, against three other survival prediction methods, which were also trained on one cancer cohort at a time, to assess the performance of our method in predicting cancer patient survival from gene expression data. Figure 5.3 shows the performances of the different survival prediction methods measured by C-Index (cf. Section 3.3.3). The evaluated methods are random survival forest (RF)[88], survival support vector machine (SVM)[156], the multiple-kernel learning (MKL) method Path2Surv[51] trained on the Hallmark gene sets[113] (MKL[H]), Path2Surv trained on the Pathway Interaction Database[149] (MKL[P]), and our proposed single-cohort XGBoost method[170] (XGB[SINGLE]). Our single-cohort XGBoost approach showed the best median C-Index of all evalu-

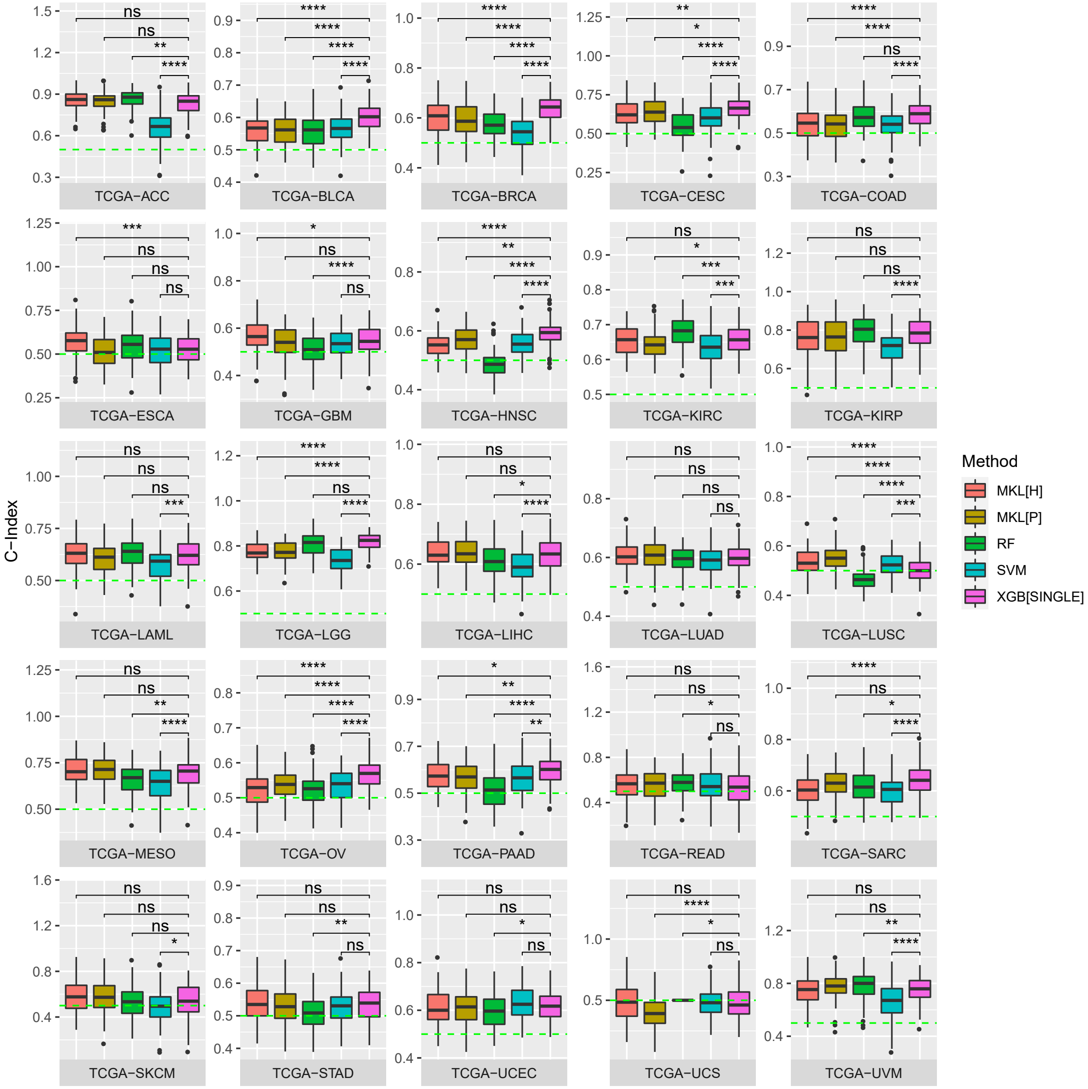


Figure 5.3: Single-cohort prediction performance. C-Index boxplots over 100 replications of model training for random survival forest (RF), survival support vector machine (SVM), the Path2Surv multiple-kernel learning on the Hallmark gene sets (MKL[H]) and the Pathway Interaction Database (MKL[P]), and the single-cohort XGBoost method (XGB[SINGLE]) on 25 different TCGA cancer cohorts. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05, * : p \leq 0.05, ** : p \leq 0.01, *** : p \leq 0.001, **** : p \leq 0.0001$. This figure was published in[170].

ated survival prediction methods for 10 of the 25 TCGA cohorts (BLCA, BRCA, CESC, COAD, HNSC, LGG, OV, PAAD, SARC, and STAD), while random survival forest was the best-performing method for 7 cohorts (ACC, KIRC, KIRP, LAML, READ, UCS, and UVM). Path2Surv outperformed the other methods in 4 cohorts (LIHC, LUAD, LUSC, and MESO) when it was trained on the PID and in 3 cohorts (ESCA, GBM, and SKCM) when it was trained on the Hallmark gene sets, while survival support vector machine showed the best median C-Index in only 1 of the TCGA cohorts (UCEC). In comparison with each of the other survival prediction methods individually, our single-cohort XGBoost method significantly outperformed random survival forest for 13, Path2Surv trained on the PID for 10, Path2Surv trained on the Hallmark gene sets for 9 and survival support vector machine for 17 of the 25 TCGA cohorts, where significance was evaluated by comparing mean C-Indices with Wilcoxon's unpaired rank-sum test and p-values ≤ 0.05 were considered significant.

For bladder urothelial carcinoma (TCGA-BLCA) and uveal melanoma (TCGA-UVM) as two example cancer types, we also assessed the Spearman correlation between the predictions of the different methods to investigate whether the different survival prediction methods make similar predictions for the same sets of cancer patients. To this end, we first split the patients of each cohort into 80% training and 20% test data and then trained each of the survival prediction models on the training data of the respective cohort. Then, we applied each of the trained methods to the TCGA-BLCA and TCGA-UVM test data, respectively, to predict the survival outcome of each test patient. The predicted survival outcome was either survival time (in survival support vector machine and Path2Surv) or a risk score (in random survival forest and single-cohort XGBoost), where a higher risk corresponded to shorter survival. Hence, survival predictions of survival support vector machine (SVM) and Path2Surv (MKL[H] and MKLP) were expected to be negatively correlated to the predictions of random survival forest (RF) and the single-cohort XGBoost method (XGB[SINGLE]). According to this expectation, predictions of different methods using the same output type (either survival time or risk score) were positively correlated, while survival time predictions were negatively correlated to risk predictions in both of the analyzed cancer cohorts (Figure 5.4a). However, the strength of correlation varied among cancer cohorts and between compared methods: In TCGA-UVM, Spearman correlations between the different methods were generally higher than in TCGA-BLCA. Furthermore, the predictions of MKL[P] and RF and of RF and XGB[SINGLE] were most highly correlated in TCGA-UVM, while in TCGA-BLCA, the correlation between XGB[SINGLE] and SVM was the highest and correlations between other methods were relatively weak ($R > -0.5$ or $R < 0.5$).

The likelihood of developing cancer is age-dependent with a probability below 6% (male 3.4%; female 5.5%) to develop cancer under the age of 50, but a probability over 25% (male 32.2%, 26% female) to develop cancer in the time span above an age of 70 years[157]. Therefore, age is an important indicator of tumor development. Indeed, we observed that the survival prediction performance of our single-cohort XGBoost method was at least to some degree dependent on the age distribution of the studied cohort. For instance, TCGA-ACC and TCGA-LGG—the two cohorts for which the single-cohort XGBoost method showed the

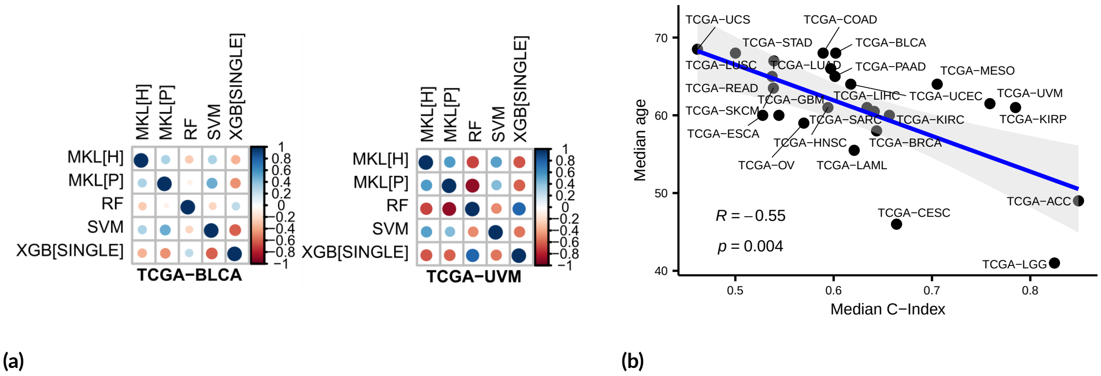


Figure 5.4: Correlation analyses of single-cohort results. **(a)** Spearman correlations between predictions of the different methods for test patients from the cohorts TCGA-BLCA (left) and TCGA-UVM (right). Larger circles correspond to a greater correlation, blue indicates a positive correlation and red indicates a negative correlation. **(b)** Spearman correlation (R) between median C-Indices of single-cohort XGBoost predictions and median ages for 25 different TCGA cohorts. The blue line shows the linear regression fit to the data and the gray area indicates the 95% confidence interval. This figure was published in [170].

best performance with a median C-Index above 0.8—comprised relatively young patients with median ages of 49 and 41 years, respectively, while for TCGA-KIRP and TCGA-UVM, which had higher median ages of 61 and 61.5 years, respectively, the median survival prediction performance dropped below a C-Index of 0.8 and for TCGA-UCS—a cohort with a particularly high median age of 68.5 years—prediction performance was low at a median C-Index of ∼0.5, not only with the single-cohort XGBoost method, but with all other evaluated survival prediction methods as well. Considering all cohorts jointly, we observed that for cancer types that are more prevalent in younger patients, survival prediction performance tended to be better than for cancers of older patients. In fact, the median C-Indices of the single-cohort XGBoost predictions in different cancer cohorts were negatively correlated (Spearman $R = -0.55, p = 0.004$, Figure 5.4b) with the median age of the respective cohort. This suggests the presence of age-specific gene expression signatures in the cancer cohorts under study that are resolvable more easily in younger patients by machine learning methods.

5.3.2 Important Features from Single-Cohort Survival Prediction Vary across Cancer Types

In addition to the quality of single-cohort XGBoost survival predictions, we were also interested in the features on which these predictions were based. To this end, we analyzed the feature importance of the gene expression features used by the single-cohort method to predict survival in the different TCGA cancer cohorts. XGBoost implements built-in feature importance metrics such as ‘gain’, ‘weight’, or ‘cover’, which are computed during model training and measure the relative importance of each feature (cf. `https://xgboost.readth` edocs.io/en/latest/python/python_api.html). Among these metrics, we chose 'gain' as the feature importance metric that was best suited for identifying genes relevant to survival. While 'weight' counts how often a feature is used as a split variable by the XGBoost model, it does not take into account where in a tree a feature is used and thus disregards how much the feature affects the prediction, since splits close to the root of the tree will generally have a higher impact on the prediction than splits more distant from the root. The metric 'cover', on the other hand, only considers the number of samples affected by a split in which a feature is involved, but does not consider how often the feature is used or how it affects the prediction. In contrast, the 'gain' metric measures the average improvement a feature adds to the prediction and thus better reflects the relative importance of a feature in the XGBoost model.

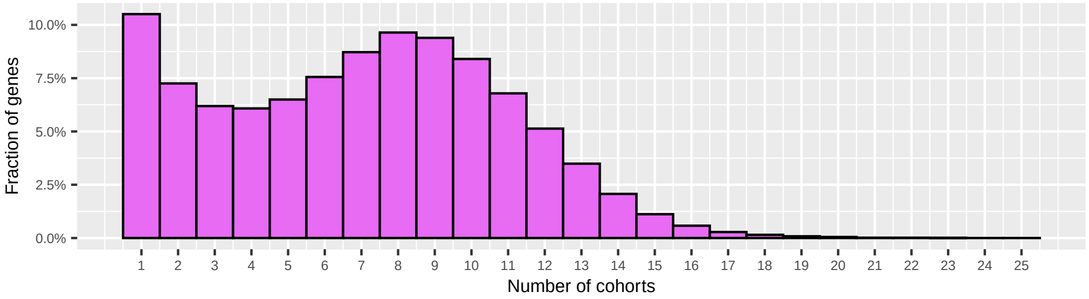


Figure 5.5: Fractions of genes shared over different cohorts for predicting survival in the single-cohort XGBoost approach. The histogram depicts the fractions of gene features that are shared for single-cohort survival prediction over different numbers of training cohorts (x-axis: number of TCGA cohorts a gene feature is shared over; y-axis: fraction of all 46,642 genes used in at least one single-cohort model). This figure was published in[170].

For each of the 25 TCGA cohorts under study, we extracted the 'gain' feature importance scores for all genes in each of the 100 model replications of the single-cohort XGBoost method for further analysis. Across all 25 cohorts and all 100 model replications per cohort, there were a total of 46,642 different genes (77% of all genes available in TCGA) that were used for cancer survival prediction in at least one of the 2,500 single-cohort XGBoost models. However, most genes were only used for prediction in a small number of cohorts (< 10) and only a very small number of genes were among the important features in a larger proportion of the studied cohorts (> 15 cohorts, Figure 5.5). This heterogeneity between cohorts and model replications in terms of important features is likely a reflection of cancer type differences and tissue specificity of some features, but also of inter-patient heterogeneity.

5.3.3 Pan-Cancer Training Improves Survival Prediction

To identify gene features with more general importance to cancer survival prediction and overcome the feature heterogeneity between cancer types, we also trained the XGBoost method on a combined dataset comprising all 25 studied TCGA cohorts (pan-cancer XGBoost approach, cf. Section 5.2.2) instead of training XGBoost models on each cohort separately.

When comparing the prediction performances for each of the 25 cohorts between the single-cohort and pan-cancer XGBoost approaches by means of C-Index over 100 replications of model training, we observed that for 15 (BLCA, COAD, HNSC, KIRC, KIRP, LIHC, LUAD, LUSC, MESO, PAAD, READ, SARC, STAD, UCEC, and UCS) out of the 25 cancer cohorts under study, pan-cancer training significantly improved over single-cohort training ($p \leq 0.05$ in Wilcoxon's unpaired rank-sum test comparing mean C-Indices, Figure 5.6). For nine additional cohorts (ACC, BRCA, CESC, ESCA, GBM, LGG, OV, SKCM, and UVM), the C-Indices obtained with single-cohort and pan-cancer training were comparable ($p > 0.05$), and only in acute myeloid leukemia (LAML), which is a cancer of the

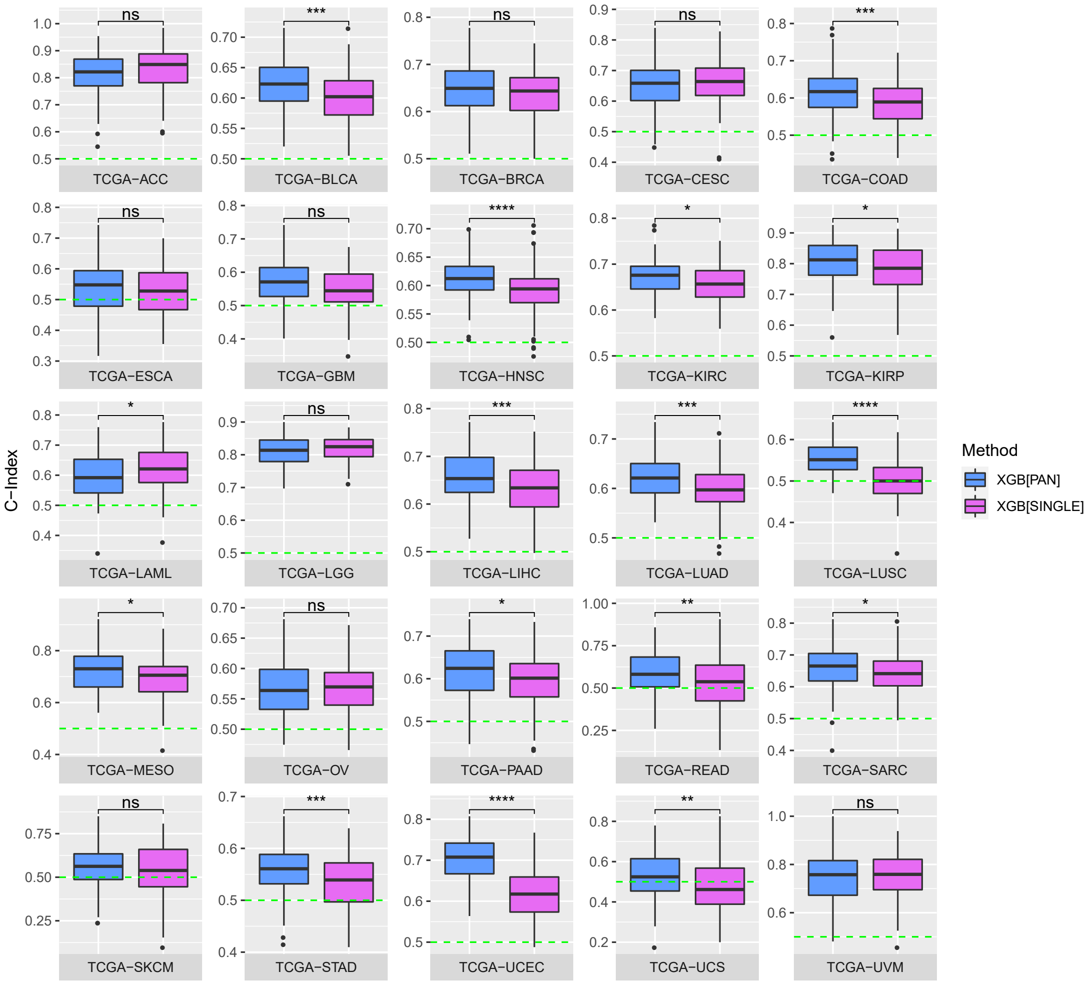


Figure 5.6: Pan-cancer prediction performance. This figure compares the prediction performances of the single-cohort XGBoost method (XGB[SINGLE]) and the pan-cancer XGBoost method (XGB[PAN]) on 25 different TCGA cancer cohorts, depicted as C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $* : p \leq 0.05$, $** : p \leq 0.01$, $*** : p \leq 0.001$, $**** : p \leq 0.0001$. This figure was published in [170].

blood and the bone marrow and the only studied cancer type that is not a solid tumor cancer, the pan-cancer XGBoost approach performed significantly worse ($p \leq 0.05$) than the single-cohort approach.

5.3.4 Important Features from Pan-Cancer Survival Prediction Generalize over Cancer Types

We compared the gene features used for predicting survival in the pan-cancer XGBoost approach with the features used by the single-cohort approach and found that the vast majority (98.6%) of genes used for pan-cancer survival prediction in at least one of the 100 model replications were also among the important features of single-cohort training in at least one cohort and replication (Figure 5.7a). Furthermore, the total number of genes used as features for cancer survival prediction in at least one of the 100 replications was reduced from 46,642 in the single-cohort XGBoost approach to 12,082 in pan-cancer training—a reduction of 74%—and the feature composition changed from 40.4% protein-coding genes, 25.0% lncRNAs, and 15.8% processed pseudogenes in single-cohort training (Figure 5.8a) to 56.5% protein-coding genes, 20.7% lncRNAs, and 11.9% processed pseudogenes in pan-cancer training (Figure 5.8b). This shift towards a larger fraction of protein-coding genes and a smaller fraction of lncRNAs and processed pseudogenes among the important features in pan-cancer survival prediction might be driven by tissue specificity of lncRNAs and patient-specific mRNA retrotransposition.

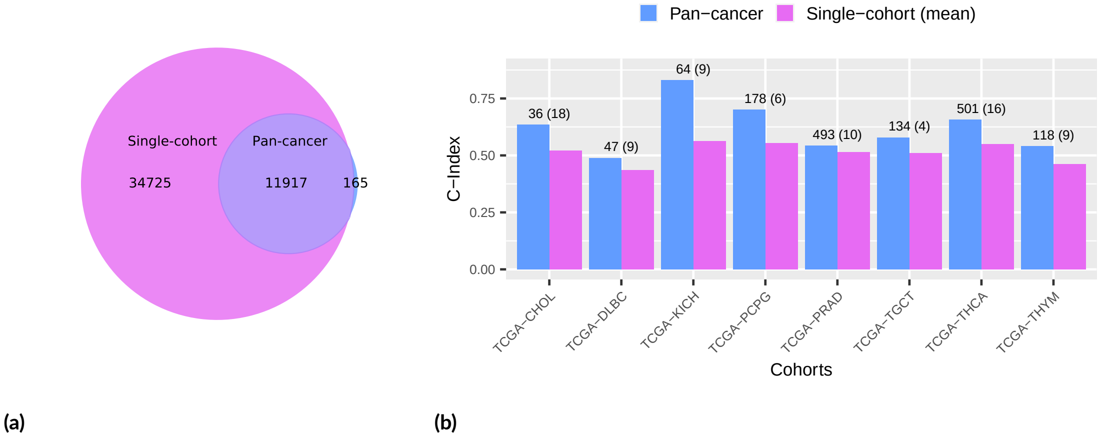
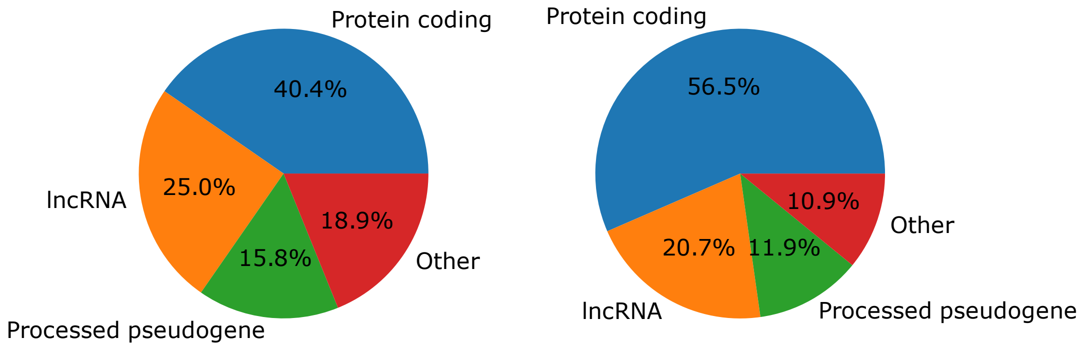


Figure 5.7: Single-cohort vs. pan-cancer survival prediction. **(a)** Venn diagram comparing features used for prediction in the single-cohort XGBoost method (pink) with those selected in the pan-cancer XGBoost method (blue). **(b)** Prediction performances (C-Indices) of single-cohort XGBoost (pink) and pan-cancer XGBoost (blue) for eight new cancer cohorts (not used in model training). For the single-cohort method, the mean C-Index over all 25 models trained on different TCGA cohorts is shown. This figure was published in[170].



(a) Single-cohort

(b) Pan-cancer

Figure 5.8: Single-cohort and pan-cancer gene types. This figure shows the types of important features identified in single-cohort and pan-cancer training. RNA types were obtained using the MyGene Python package (version 3.1, http://mygene.info)[186,187]. **(a)** Percentages of different types of RNAs identified as important features in the single-cohort XGBoost approach. **(b)** Percentages of different types of RNAs identified as important features in the pan-cancer XGBoost approach. This figure was published in[170].

The gene features selected in the pan-cancer XGBoost approach were not specific to a particular type of cancer, but tended to generalize over multiple cancer types. This advantage can be used to extrapolate survival prediction to yet unseen cancer types that were not represented in the training data. To test this claim, we additionally trained a single-cohort model on each of the 25 TCGA cohorts under study, using all available patients without holding out any test data, and a pan-cancer model, again using all available patients from all 25 cohorts, but this time combining all 25 cohorts to one pan-cancer dataset. Then, to evaluate the transferability of the trained models to new cancer types, we tested each of the trained models on eight additional TCGA cancer cohorts (CHOL, DLBC, KICH, PCPG, PRAD, TGCT, THCA, and THYM), which had previously been excluded due to small numbers of uncensored patients (cf. Section 5.2.1). For the 25 single-cohort models—each trained on the data of one TCGA cancer cohort—we summarized the survival prediction performance on each new cancer cohort as the mean of C-Indices, while testing the pan-cancer model only resulted in one C-Index per new cancer cohort and no aggregation of results was necessary. For all of the eight new cohorts, the C-Index of the prediction computed by the pan-cancer XGBoost model was better than the mean C-Index of predictions made by single-cohort XGBoost models and for seven of the eight cohorts, the pan-cancer model yielded a C-Index above 0.5 even though none of the eight cancer cohorts was represented in model training. This supports the hypothesis that genes identified in the pan-cancer XGBoost approach are more predictive of patient survival in previously unseen cancer types than genes identified by the single-cohort approach.

5.3.5 Important Features from Pan-Cancer Survival Prediction Are Biologically Plausible

To explore the biological plausibility of the genes identified as important features by the pan-cancer XGBoost method and to gain more insights into the underlying biology of cancer survival, we analyzed the distribution of gene weights, which were computed as the sum of feature importance scores per gene across all 100 replications of pan-cancer training. Figure 5.9 shows the top 100 genes with the highest weights. Ensembl gene identifiers were converted to HUGO gene symbols using the MyGene[186,187] Python package (version 3.1; http://mygene.info) if a gene symbol was available. In cases where the Ensembl gene identifier could not be mapped to a HUGO symbol, Ensembl identifiers were used as gene names. Noticeably, a few genes, and especially *IGF2BP3* (insulin-like growth factor 2 mRNA binding protein 3), have much higher gene weights than all other genes identified as important features by the pan-cancer XGBoost approach, indicating a particularly high prognostic potential for cancer survival. Indeed, *IGF2BP3* is overexpressed in many tumor types and has been associated with tumor progression, metastasis, and poor prognosis in multiple cancers[122], including colon cancer[117], oral squamous cell carcinoma[114], and melanoma[155].

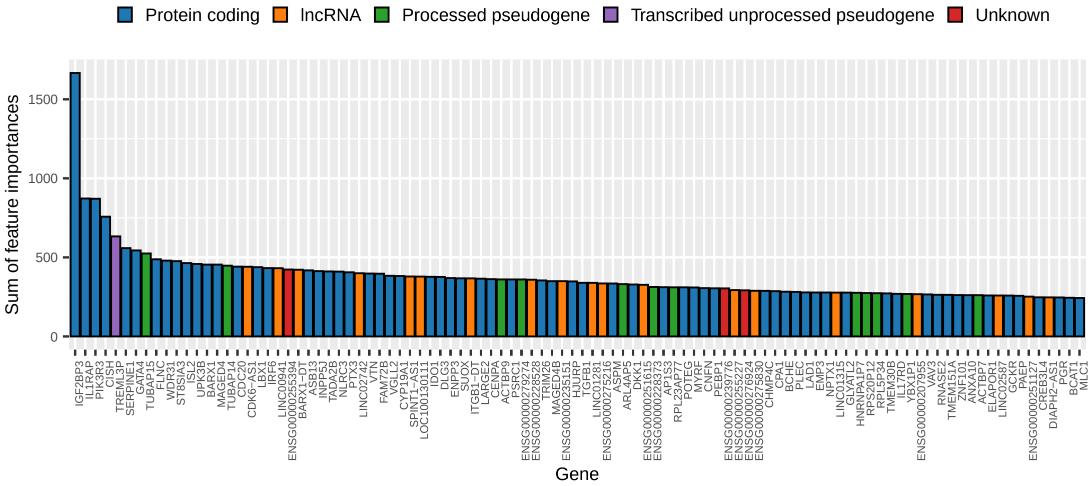
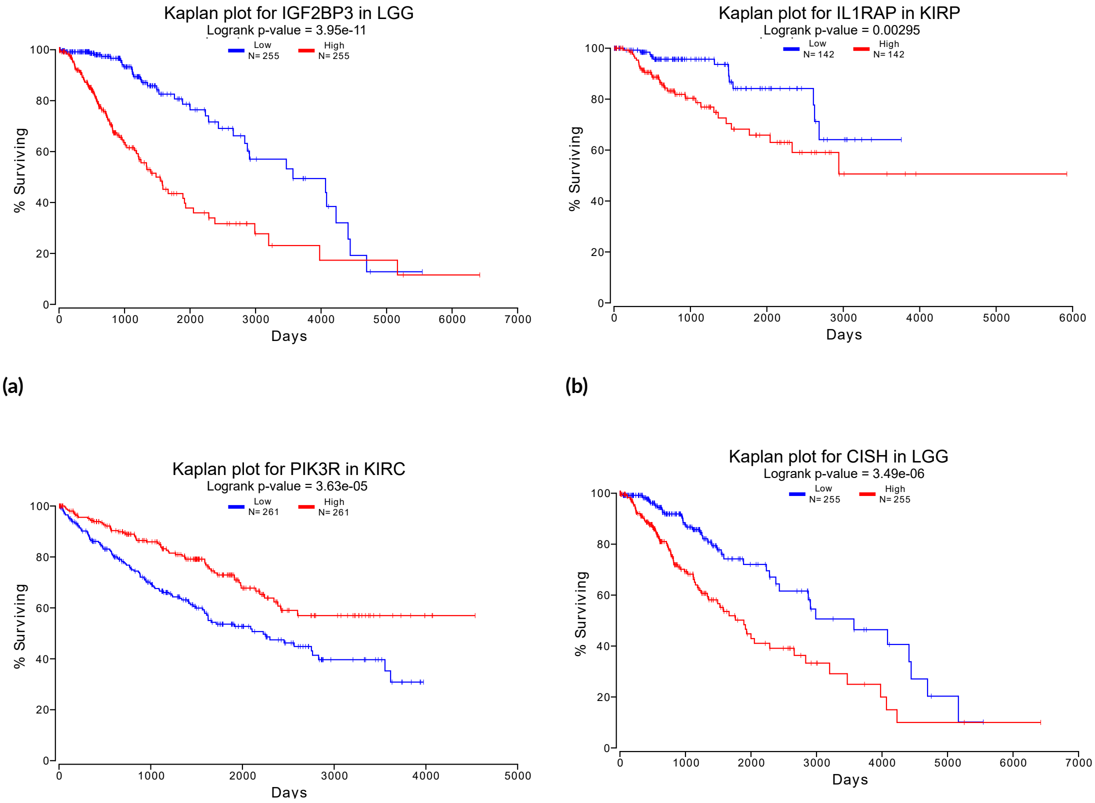


Figure 5.9: Pan-cancer feature importance. This figure shows the weight distribution for the 100 genes with the highest feature importance (sums of feature importance scores over 100 model replications) for pan-cancer XGBoost training (gene identifiers that did not map to a Hugo symbol are named with their Ensembl identifiers). The different colors indicate gene types (blue: protein coding, orange: lncRNA, green: processed pseudogenes, purple: transcribed unprocessed pseudogene, red: gene type unknown). These gene types were obtained using the MyGene Python package (version 3.1, http://mygene.info)[186,187]. This figure was published in[170].

To further analyze the prognostic potential of the genes attributed with the highest feature importance by the pan-cancer XGBoost approach, we queried OncoLnc[6]—an online tool providing Cox regression analyses and Kaplan-Meier survival plots on TCGA gene expression data of different cancer types—with the top four protein-coding genes with the highest gene weights (*IGF2BP3*, *IL1RAP*, *PIK3R3*, and *CISH*). Figure 5.10 shows one Kaplan-Meier plot for each of these four genes, where for each gene the cancer type with the lowest FDR-corrected p-value in the OncoLnc Cox regression was selected for display. *IGF2BP3* is



(a)

(b)

(c)

(d)

Figure 5.10: Kaplan-Meier plots for the four most important gene features from pan-cancer XGBoost. For each gene, we selected the cancer type with the lowest FDR-corrected p-value in Cox regression, respectively. As a cutoff for gene expression, the 50th percentile was selected. Cox regression data and Kaplan-Meier plots were retrieved from OncoLnc[6]. **(a)** Survival of brain lower grade glioma (LGG) patients, split by expression of *IGF2BP3*. **(b)** Survival of kidney renal papillary cell carcinoma (KIRP) patients, split by expression of *IL1RAP*. **(c)** Survival of kidney renal clear cell carcinoma (KIRC) patients, split by expression of *PIK3R3*. **(d)** Survival of brain lower grade glioma (LGG) patients, split by expression of *CISH*. This figure was published in[170].

most predictive for survival in brain lower grade glioma (LGG, FDR = 3.59×10^{9} in Cox regression), while *IL1RAP*, *PIK3R3*, and *CISH* have the highest predictive potential in kidney renal papillary cell carcinoma (KIRP, FDR = 1.65×10^{4} in Cox regression), kidney renal clear cell carcinoma (KIRC, FDR = 4.16×10^{3} in Cox regression), and LGG (FDR = 1.51×10^{5} in Cox regression), respectively. Furthermore, *IGF2BP3*, *IL1RAP*, *PIK3R3*, and *CISH* have significant prognostic value (FDR < 0.05 in Cox regression) for four (KIRP, KIRC, LUAD, and PAAD), two (LGG and PAAD), two (LGG and HNSC), and four (LUAD, LIHC, KIRP, and KIRC) additional TCGA cohorts, respectively. Kaplan-Meier plots for these gene-cohort pairs are shown in Supplementary Figure A.3. For generating the Kaplan-Meier plots for each gene-cohort pair, the 50th percentile was selected as a cutoff in OncoLnc, such that patients belonging to the respective cohort were split into two groups of equal size based on the gene expression value of the corresponding gene. We selected the 50th percentile as a cutoff to ensure that all patients from the respective cohort were included in the analysis and the two groups (low expression and high expression) the patients were split into had approximately equal size. OncoLnc uses the logrank test to assess if there is a significant difference in survival times between the low-expression and high-expression groups. According to this test, there is a significant survival difference ($p \leq 0.05$) between the low-expression and high-expression groups for all gene-cohort pairs with a significant FDR-corrected p-value in OncoLnc's Cox regression except for the gene *CISH* in the liver hepatocellular carcinoma (LIHC). This indicates that the top four genes with the highest feature importance in the pan-cancer XGBoost survival prediction method are indeed predictive for cancer survival in different cancer types.

To further evaluate whether the genes attributed with high feature importance values in the pan-cancer XGBoost approach generalize over cancer types and are prognostic for survival across multiple types of cancer, we additionally computed the entropy of the top 100 genes with the highest gene weights from the pan-cancer XGBoost approach with respect to the gene weights derived from the single-cohort approach and compared the resulting entropy distribution with the entropy distribution obtained by computing the entropies of the top 100 genes with the highest sum of gene weights across all 25 cohorts from the single-cohort XGBoost approach (cf. Section 5.2.5). In our case, where feature importance in 25 different cancer cohorts is analyzed, the entropy falls into a range between 0 and ~ 4.64 and measures to what extent each gene's prognostic value generalizes across cancer types. That is, a high entropy score indicates that the respective gene has similar gene weights in many or all of the 25 analyzed cancer types, while genes that are only predictive for survival in one or a few cohorts will have a low entropy. The comparison between the entropy distributions of the 100 most important pan-cancer survival prediction genes and the 100 most important single-cohort survival prediction genes shows that the pan-cancer prognostic genes have significantly higher entropy than the single-cohort genes ($p = 1.145 \times 10^{14}$ in a one-sided Wilcoxon unpaired rank-sum test, Figure 5.11), implying that the pan-cancer XGBoost survival prediction approach indeed generalizes better over different cancer types than the single-cohort XGBoost approach.

5.3.6 Network Propagation Identifies a Pan-Cancer Survival Network

As described in the previous Sections 5.3.4 and 5.3.5, the pan-cancer XGBoost approach uses substantially (74%) fewer genes for predicting cancer survival than the single-cohort XGBoost

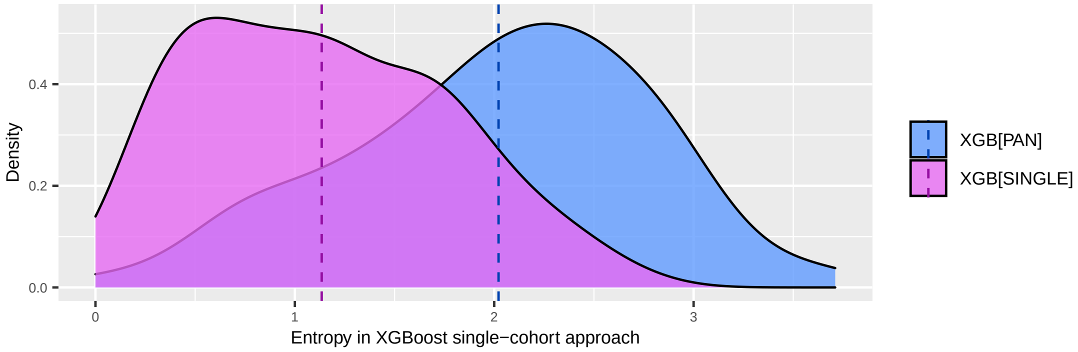


Figure 5.11: Single-cohort and pan-cancer feature entropy. The entropy distributions between the top 100 genes with the highest feature importance (feature importance is measured as sums of feature importance scores over 100 model replications) from the single-cohort approach and the pan-cancer approach are compared (mean entropies are indicated as dashed lines). The entropy measure (x-axis) is based on the genes used in the single-cohort approach (cf. Section 5.2.5). The density of the entropy distribution is displayed on the y-axis. This figure was published in[170].

approach and the genes with the highest feature importance are biologically highly plausible. However, there is still a total of 12,082 genes that are used for survival prediction in at least one of the 100 replications of pan-cancer training. This large number of features is not biologically focused and exacerbates the inference of mechanistic information. Furthermore, many of the genes are only identified as important features in a small number of replications, implying that their selection as survival features is highly dependent on the training set composition and they might not generally have high relevance for cancer prognosis. In fact, the distribution of pan-cancer gene weights—computed for each gene as the sum of feature importance scores over all model replications—resembles a "long-tail" distribution (Figure 5.12), a distribution that is also often visible with cancer-associated SNPs[7]. Nonetheless, genes with relatively lower gene weights might still be prognostic for cancer survival in a subset of patients and simply omitting genes that fall below some weight threshold would be rather arbitrary and could potentially lead to a loss of relevant information.

It has been suggested that domain-specific prior knowledge, such as that from biological networks, can improve the performance of machine learning methods and help to understand the underlying biological mechanisms[26]. One way to incorporate such prior knowledge is network propagation. Network propagation is a popular technique that leverages the prior knowledge from a network, such as a PPI network, to amplify biological signal and can help to gain insights into underlying biological mechanisms[43] (cf. Section 3.4). The technique has for example been used for the identification of genes that are associated with specific diseases[109,103,145] and recently, we have applied network propagation to time-resolved gene expression profiles of *Leishmania major* infected bone marrow-derived macrophages from mice with different responses to the infection (disease susceptible or resistant) and identified network modules of interacting proteins in the PPI network that aggregated infection

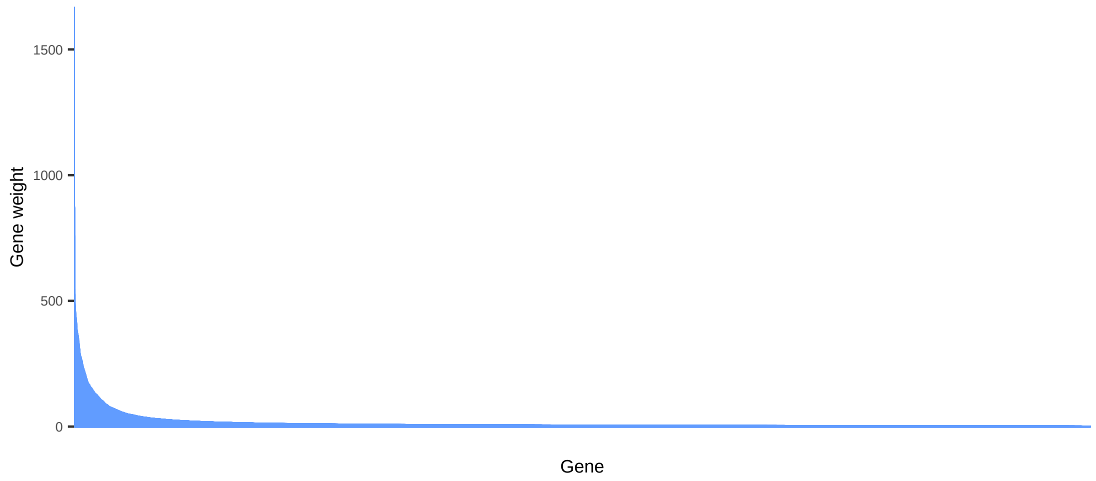


Figure 5.12: Distribution of pan-cancer gene weights. The pan-cancer gene weights resemble a "long-tail" distribution. The x-axis displays the 12,082 genes identified as important features in the 100 model replications of the pan-cancer XGBoost method and the y-axis shows the corresponding gene weights, computed as sums of feature importance scores across the 100 replications. A version of this figure was published in [170].

response signals for the susceptible and resistant mouse strains [20].

In this work, we applied the NetCore [13] network propagation method to the gene weights extracted from the 100 replications of the pan-cancer XGBoost survival prediction approach (cf. Section 5.2.6). To this end, the high-confidence CPDB PPI network [82,93] was initialized with the pan-cancer gene weights and the weights were then propagated over the network in a RWR until a steady state distribution was reached [13]. Based on this re-weighted PPI network, NetCore then identified network modules—connected subgraphs of the network that connect genes with initially high gene weights with genes that gained significantly high weights during network propagation. Figure 5.13 displays the largest network module identified by NetCore based on the pan-cancer gene weights. In total, NetCore could identify 13 different modules, each containing between 2 and 79 genes. Taken together, these modules compose a pan-cancer survival network comprising a total of 103 different genes, of which 76 are seed genes with high initial gene weights before network propagation and 27 genes were inferred during network propagation. All 103 genes, including their initial and propagated weights, are listed in Supplementary Table B.2.

The identified pan-cancer survival network is indeed informative for survival in different cancer types. For the 25 TCGA cancer types under study, an average of 41.48 genes of the 103 survival network genes were among the important features of the respective cohort in the single-cohort XGBoost survival prediction approach. For instance, in the single-cohort training of lung squamous cell carcinoma (TCGA-LUSC), 59 of the 103 survival network genes were among the important features, followed by head and neck squamous cell carcinoma

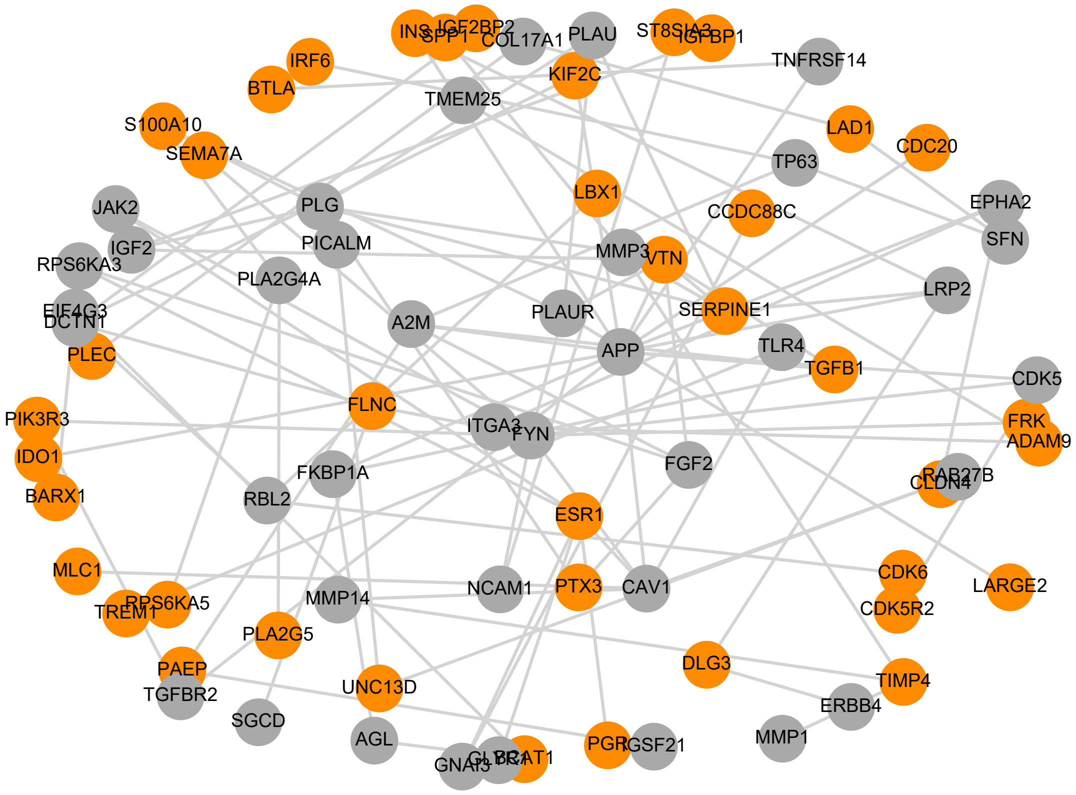


Figure 5.13: Largest network module. Network modules were identified by NetCore[13] network propagation and module identification based on pan-cancer gene weights, which were computed from XGBoost feature importance scores over 100 model replications. Orange nodes correspond to seed genes, while genes that were inferred during network propagation are colored in gray. This figure was published in[170].

(TCGA-HNSC) and ovarian serous cystadenocarcinoma (TCGA-OV) with 54 genes each (Figure 5.14). However, the gene weights of the 103 survival network genes, which were computed from their pan-cancer XGBoost feature importance scores, were highly variable between cohorts. For instance, while the sum of gene weights over the 103 genes was relatively high in some cohorts, including TCGA-LUAD, TCGA-KIRC, TCGA-LUSC, and TCGA-HNSC, the gene weight of the same genes was much lower in other cohorts like TCGA-SKCM, TCGA-READ, and TCGA-UVM (Figure 5.14). The low gene weights in the latter cohorts might partly be attributed to the comparatively small sizes of these cohorts (<200 patients, cf. Supplementary Table B.1), possibly leading to a proportionally smaller contribution of these cohorts to pan-cancer XGBoost training and the associated feature importance scores.

Approximately a quarter of the genes in the pan-cancer survival network (27 of the 103 genes) are annotated cancer genes, which have been manually curated in NCG (version 6.0)[144]. Interestingly, 16 of these 27 genes were inferred by network propagation (Supplementary Table

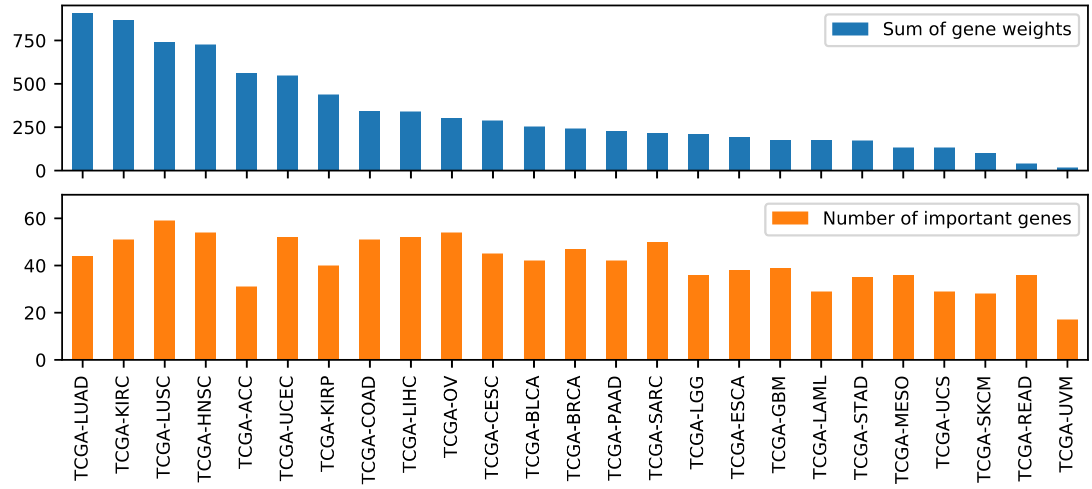


Figure 5.14: Feature importance of the 103 pan-cancer survival network genes in single-cohort training. Gene weights were computed based on feature importance scores from 100 replications of single-cohort XGBoost training. Top: Sum of gene weights of the survival network genes per cohort. Bottom: Number of genes (of the 103 survival network genes) per cohort that are among the important features in single-cohort training (gene weight $>$ 0). A version of this figure was published in[170].

B.2), meaning that they had relatively little or no feature importance in the XGBoost pan-cancer survival prediction, but received significantly high weights during network propagation due to their high connectivity with highly important gene features in the PPI network.

5.3.7 The Pan-Cancer Survival Network Is Strongly Associated with the Tumor Microenvironment

To further characterize the genes contained in the pan-cancer survival network, we performed an ORA using the QIAGEN IPA software[104] on a set of canonical pathways defined by IPA (Supplementary Table B.3) and additionally retrieved upstream regulators of the survival network genes (Supplementary Table B.4). The 103 pan-cancer survival network genes are most significantly enriched for the tumor microenvironment (TME) pathway ($p = 4.57 \times 10^{-10}$; overlapping genes: *FGF2*, *IDO1*, *IGF2*, *JAK2*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *PLAU*, *SPP1*, *TGFB1*; Supplementary Table B.3). The TME is implicated in tumor initiation, growth, invasion, metastasis, and response to therapies[92,140]. It comprises non-malignant host cells, blood vessels, nerves, lymph nodes, and lymphoid organs, as well as intercellular components and metabolites and forms in close vicinity of the tumor. It strongly interacts with the cancer cells, assisting the development of Hallmark capabilities (cf. Section 2.1) and supporting the cancer cells' survival and migration. The TME can be subdivided into several specialized microenvironments with distinct functions, such as the hypoxic, the acid, and the innervated niches, and the immune, metabolism, and mechanical microenvironments.

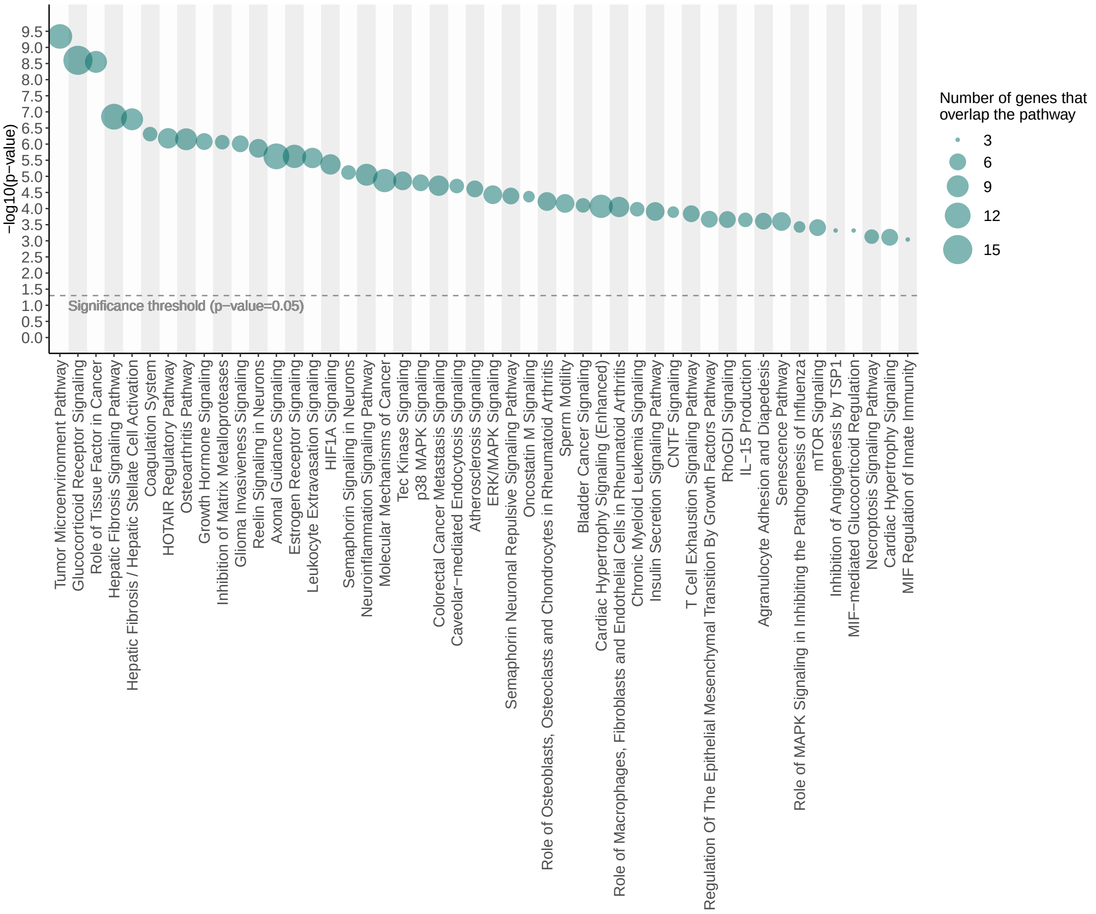


Figure 5.15: Over-represented pathways identified with QIAGEN Ingenuity Pathway Analysis (IPA). Pathways over-represented with $p < 0.001$ in the 103 survival network genes are displayed on the x-axis and the y-axis shows over-representation p-values (negative log-scale). The bubble sizes represent the numbers of survival network genes that overlap with the respective pathway and the dashed line indicates a significance threshold of 0.05. All shown pathways are still significantly over-represented ($p < 0.05$) after controlling the FDR according to Benjamini-Hochberg (see Supplementary Table B.3, Section 3.5.2). Inspired by https://digitalinsights.qiagen.com/wp-content/uploads/2016/12/1-for-akhil.png, accessed on May 9th, 2023.

Besides the TME as a whole, some of these specialized microenvironments, such as hypoxia-inducible factor 1A (HIF1A) signaling ($p = 4.27 \times 10^{-6}$; overlapping genes: *FGF2*, *IGF2*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *SERPINE1*, *TGFB1*), are also enriched by the pan-cancer survival network. Hypoxia is a known property of cancer and promotes angiogenesis through the upregulation of vascular endothelial growth factor (VEGF)[92,48]. It has been linked to cancer progression, therapeutic resistance, and poor prognosis[92], as well as metastasis[143].

Furthermore, there is significant enrichment for immune-related pathways in the pan-cancer survival network. For instance, glucocorticoid receptor (GR) signaling has been found to regulate CD8+ T cell differentiation, where increased GR signaling is associated with dysfunctional CD8+ tumor-infiltrating lymphocytes (TILs)[3], and is significantly enriched by the 103 survival network genes ($p = 2.51 \times 10^{-9}$; overlapping genes: *A2M*, *CAV1*, *ESR1*, *JAK2*, *MMP1*, *MMP3*, *PGR*, *PIK3R3*, *PLA2G4A*, *PLA2G5*, *PLAU*, *RPS6KA5*, *SERPINE1*, *TGFB1*, *TGFBR2*). While high levels of TILs are linked to improved patient survival in colorectal cancer[87] and some breast cancer subtypes[49], dysfunctional CD8+ TILs contribute to immunosuppression in the TME[3], potentially interfering with the positive effect of functional TILs on survival. Another significantly enriched pathway related to the immune system is the *T Cell Exhaustion Signaling Pathway* ($p = 1.44 \times 10^{-4}$; overlapping genes: *BTLA*, *JAK2*, *PIK3R3*, *TGFB1*, *TGFBR2*, *TNFRSF14*). T cell exhaustion is a phenomenon observed in chronic viral infection and cancer in response to chronic antigen stimulation[19,91] and is characterized by increased expression of inhibitory receptors, decreased production of effector cytokines, and reduced cytotoxicity[91]. Most T cells in the TME are exhausted and lose the competence to eliminate cancer, thus allowing the cancer to evade immune response[91].

Another pathway associated with the TME and significantly enriched for genes from the pan-cancer survival network is inhibition of matrix metalloproteases (MMPs) ($p = 1.28 \times 10^{-1}$, overlapping genes: *A2M*, *MMP1*, *MMP14*, *MMP3*, *TIMP4*). MMPs are a proteinase family that mediates molecular communication between tumor and stroma and can modulate the TME[96]. MMPs regulate signaling pathways that control cell growth, inflammation, and angiogenesis, play an important role in extracellular matrix turnover and cancer cell migration, and are thus tightly linked to tumorigenesis.

In addition to TME- and immune-related pathways, the pan-cancer survival network is also enriched by other signaling pathways that have been linked to cancer survival, such as the mechanistic target of rapamycin (mTOR) signaling pathway ($p = 2.86 \times 10^{-2}$, overlapping genes: *EIF4G3*, *FKBP1A*, *INS*, *PIK3R3*, *RPS6KA3*, *RPS6KA5*), which regulates essential cell processes like protein synthesis and autophagy, and, if deregulated, promotes cancer progression[148,173]. Notably, mTOR signaling is activated by PI3K/AKT in response to insulin (INS)[192] and the *INS* and *PI3KR3* genes were among the top 100 genes with the highest gene weights in the pan-cancer prediction approach. Another pathway closely linked to cancer and enriched by the pan-cancer survival network is ERK/MAPK signaling ($p = 3.47 \times 10^{-2}$, overlapping genes: *ESR1*, *FYN*, *ITGA3*, *PIK3R3*, *PLA2G4A*, *PLA2G5*, *RPS6KA5*). ERK/MAPK signaling is involved in the regulation of cell proliferation, differentiation, apoptosis, and stress responses[65] and is targeted by many cancer drugs[161].

Thorsson et al.[172] have divided TCGA cancer patients into six distinct immune subtypes based on immune expression signatures. The six immune subtypes are wound healing, IFN-γ dominant, inflammatory, lymphocyte depleted, immunologically quiet, and TGF-β dominant and are characterized by differences in somatic aberrations, tumor microenvironments, and patient prognosis. Since we found a strong association between the pan-cancer survival network and TME- and immune-related molecular pathways, we asked to what extent the gene expression signals of the 103 survival network genes reflect the six immune subtypes identified by Thorsson et al.[172]. For 7,475 of the 8,024 TCGA patients used for survival prediction, one of the six immune subtypes could be assigned according to Thorsson et al.[172]. PCA of these patients with respect to the 103 pan-cancer survival network genes showed a partial discrimination between immune subtypes (Figure 5.16).

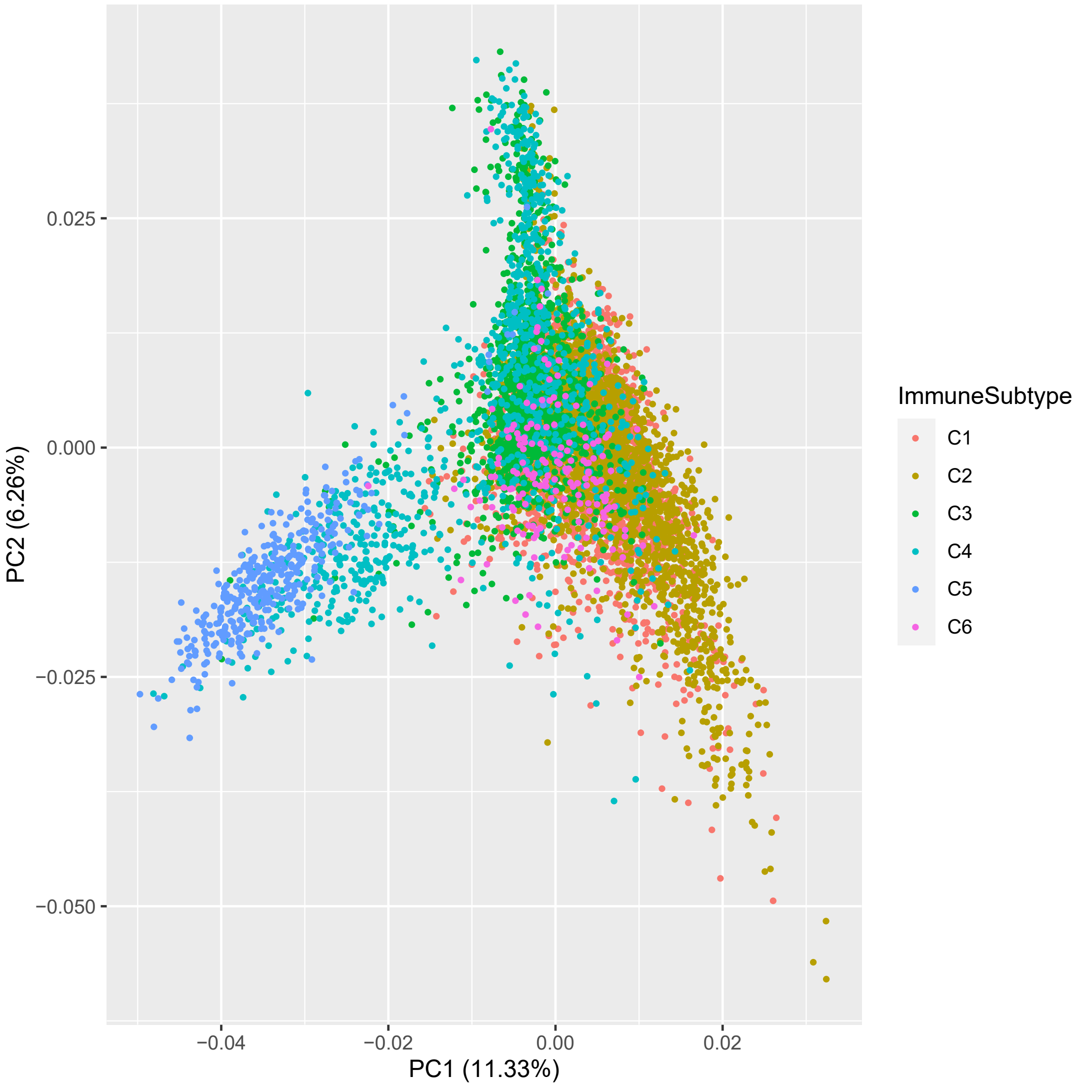


Figure 5.16: Association of the pan-cancer survival network with immune subtypes. PCA of the patients that can be assigned to an immune subtype according to Thorsson et al., 2018[172]. The PCA is based on the 103 pan-cancer survival network genes and patients are colored by their assigned immune subtype. The PCA was generated with the R library ggplot2[182]. This figure was published in[170].

In particular, patients belonging to immune subtype C5 (immunologically quiet), which predominantly comprised brain lower-grade gliomas (LGG), were separated from patients associated with other immune subtypes in the first two principal components of the PCA. This indicates that the pan-cancer survival network indeed contains gene expression signal that is informative for the patients' immune subtypes.

In addition to the ORA of the pan-cancer survival network and the analysis of immune subtypes, we also explored potential upstream regulators of the survival network genes that are frequently mutated in cancer and are annotated as cancer drivers or candidate cancer drivers in the Network of Cancer Genes (NCG)[144]. To this end, we used QIAGEN IPA[104] again to perform enrichment analysis on the annotation sets of "upstream regulators" and identified 47 significantly enriched upstream regulators ($p < 1 \times 10^{-5}$, Supplementary Table B.4). The top ten most significantly enriched upstream regulators are *JUN* (20 target genes in the pan-cancer survival network), *TNF* (34 target genes), *IL1B* (25 target genes), *TP53* (33 target genes), *IL1A* (13 target genes), *FGF2* (15 target genes), *MAP3K1* (7 target genes), *EGFR* (15 target genes), *STAT3* (16 target genes), and *HRAS* (16 target genes). Notably, several of these upstream regulators are associated with the TME. For instance, *TNF* (tumor necrosis factor alpha) is a multifunctional cytokine that regulates the tumor microenvironment and is involved in apoptosis, angiogenesis, inflammation, and immunity[177,66]. Another upstream regulator, *STAT3* (signal transducer and activator of transcription 3), is hyperactivated in cancer and normal cells in the tumor ecosystem and is a key regulator of the anti-tumor immune response[194]. It is involved in the inhibition of essential immune activation regulators and the production of immunosuppressive factors and is thus a promising target for immunotherapy. *TP53* is the most frequently mutated gene in human cancers and encodes for the p53 tumor suppressor protein, which is associated with the control of cell cycle progression, DNA repair, apoptosis, and cell survival and thus acts as a suppressor of tumorigenesis[8]. Additionally, p53 promotes an anti-tumor microenvironment, in part through secreted factors that modulate macrophage function[120]. Mutations of p53 can have non-cell autonomous effects, modulating the TME and impairing its tumor-suppressing function, thus allowing for cancer development[8].

Taken together, our results emphasize the strong association of the pan-cancer survival network with the TME, which is closely linked to cancer immune response and has an integral role in cancer progression and metastasis.

5.3.8 Gene Expression Is the Most Informative Data Modality

Cancer survival prediction based on gene expression data of TCGA cancer patients from multiple cancer types yielded good results (cf. Sections 5.3.1 and 5.3.3). However, we were also interested in whether the integration of additional molecular data modalities would further improve survival prediction performance. To this end, we applied the pan-cancer XGBoost approach (cf. Section 5.2.2) to mutation, copy number variation, and protein expression data (cf. Section 5.2.1) alone and in combination with TPM-normalized RNA-seq gene expression data. As described in Section 5.2.1, for this part of the work we used TPM-normalized gene expression data from the more recent GDC release v32.0 instead of the FPKM-normalized gene expression data from releases v22.0 and v24.0 that was used in the first part of the work described above. However, despite the different processing and normalization strategies applied to the two types of gene expression data, pan-cancer survival prediction with XGBoost yielded similar performances on both gene expression types (Supplementary Figure A.4).

Integration of Mutation Data

The genomic landscapes and mutation patterns of tumors vary between tissues and cell types, but also between tumors originating from the same tissue and cell type[25]. In fact, the vast majority of mutations occurring in tumors of a particular tissue type are only found in less than 5–10% of patients[25] and even when the same gene is mutated in different patients, the specific mutations occurring in that gene often differ between tumors[178]. Because of this large diversity and low prevalence of most mutations across tumors, it is not feasible to use the raw mutation data with locus-specific mutation information directly as input for our survival prediction method. The machine learning algorithm would likely not be able to extract much meaningful information from a very large number of low-prevalence mutations, some of which do not even affect protein function. To address this problem, we implemented different strategies, first selecting only high-impact mutations and then summarizing mutations at the gene or pathway level, or extracting other potentially affected genes through network propagation. More specifically, we first filtered simple nucleotide variations for their impact and for each TCGA patient only kept mutations annotated with "high" impact in the respective MAF file. The different "impact" categories are defined by Ensembl's variant effect predictor (VEP)[126] and reflect the impact of a mutation on the encoded protein. "High" impact means that the variant is assumed to have a disruptive impact like protein truncation or loss of function on the protein or may trigger nonsense-mediated decay, while "moderate" impact means the variant is not disruptive and might only change protein effectiveness and "low" impact mutations do likely not change protein behavior at all (cf. `https://docs.gdc.cancer.gov/Data/File_Formats/MAF_Format/`). Then, we binarized the high-impact mutations at the gene level, where for each patient, a gene was considered mutated if it contained at least one high-impact mutation. As an alternative to binarizing mutations at the gene level, we also computed pathway-level mutations or applied network propagation to mutations to identify genes that were not mutated themselves, but were likely affected by other mutated genes. For the computation of pathway-level mutations, high-impact mutations were also first mapped to genes, but instead of computing a binary mutation value for each gene, we counted the high-impact mutations per gene and for each CPDB[82,93] pathway computed the sum over all genes belonging to that pathway. Then, we normalized each pathway-level mutation value with the pathway size by dividing

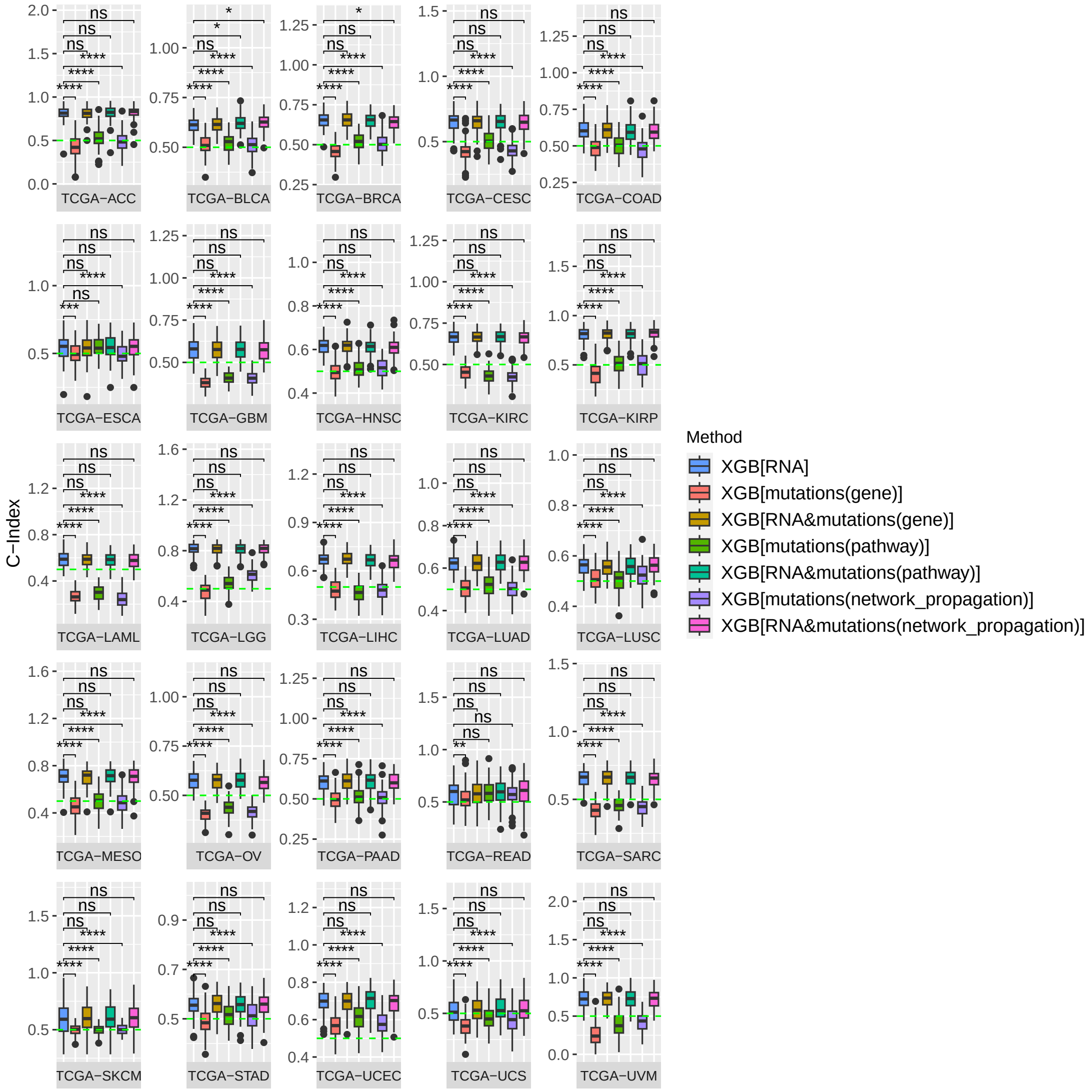


Figure 5.17: Pan-cancer survival prediction performance on mutation data. Comparison of the prediction performance of the pan-cancer XGBoost method on 25 different TCGA cancer cohorts, trained on gene expression data (XGB[RNA]), gene-level mutation data (XGB[mutations(gene)]), gene expression and gene-level mutation data (XGB[RNA&mutations-(gene)], pathway-level mutation data (XGB[mutations(pathway)]), gene expression and pathway-level mutation data (XGB[RNA&mutations(pathway)]), mutations processed with network propagation (XGB[mutations(network_propagation)]), and gene expression data in combination with mutations processed with network propagation (XGB[RNA&mutations(network_propagation)]). Performance is depicted by C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05, * : p \leq 0.05, ** : p \leq 0.01, *** : p \leq 0.001, **** : p \leq 0.0001$.

it by the number of genes belonging to the pathway. For the application of network propagation to mutations, on the other hand, the binarized gene-level mutations were mapped to the CPDB[82,93] (release 35) high-confidence (confidence > 0.9) PPI network and network propagation according to the NetCore[13] RWR (cf. Section 3.4.1; restart probability 0.8) was performed on the network. The re-weighted gene values computed during network propagation were then used as input for the pan-cancer XGBoost approach.

Figure 5.17 shows the performance (measured as C-Index) of pan-cancer XGBoost survival prediction in different settings that used either gene expression data (XGB[RNA]) or the processed mutation data (gene-level mutations: XGB[mutations(gene)], pathway-level mutations: XGB[mutations(pathway)], or mutations processed with network propagation: XGB[mutations(network_propagation)]) alone or in combination with gene expression data (XGB[RNA&mutations(gene)], XGB[RNA&mutations(pathway)], and XGB[RNA&mutations(network_propagation)]) as input. Analogously to pan-cancer survival prediction on gene expression data only (cf. Section 5.3.3), we repeated model training 100 times on different train-test splits and in each replication selected the 500 most predictive features per data modality as described in Section 5.2.2. In almost all (23) of the 25 TCGA cancer cohorts, survival prediction performance was significantly worse for all three types of processed mutation data when compared to survival prediction based on TPM-normalized RNA-seq gene expression data. When the processed mutation data modalities were combined with gene expression data to predict pan-cancer survival and compared to survival prediction on gene expression data alone, no significant performance differences were observed in most TCGA cohorts. Only in TCGA-BLCA, pathway-level mutations in combination with gene expression and mutations processed by network propagation in combination with gene expression yielded significantly ($p < 0.05$) better performance than gene expression alone, indicating that for most cancer types, mutations did not contain additional survival information that is complementary to the information contained in gene expression data and would improve survival prediction.

Integration of Copy Number Variation Data

Copy number variations (CNVs) are a type of SV[42]. They refer to genomic regions that vary in copy number either through amplification or deletion of DNA and can drive adaptive evolution and progression of genetic diseases like cancer[108]. By incorporating copy number variation data from TCGA into pan-cancer survival prediction with XGBoost, we sought to investigate whether and to what extent copy number variations provide information on patient survival and can complement gene expression data in survival prediction. To this end, we conducted pan-cancer survival prediction on copy number variation data only (XGB[CNV]), and copy number variation data in combination with gene expression data (XGB[RNA&CNV]) and compared these two settings with survival prediction based on gene expression data alone (XGB[RNA]). We repeated model training 100 times per setting on different train-test splits and in each replication selected the top 500 most predictive genes per data modality as features (cf. Section 5.2.2).

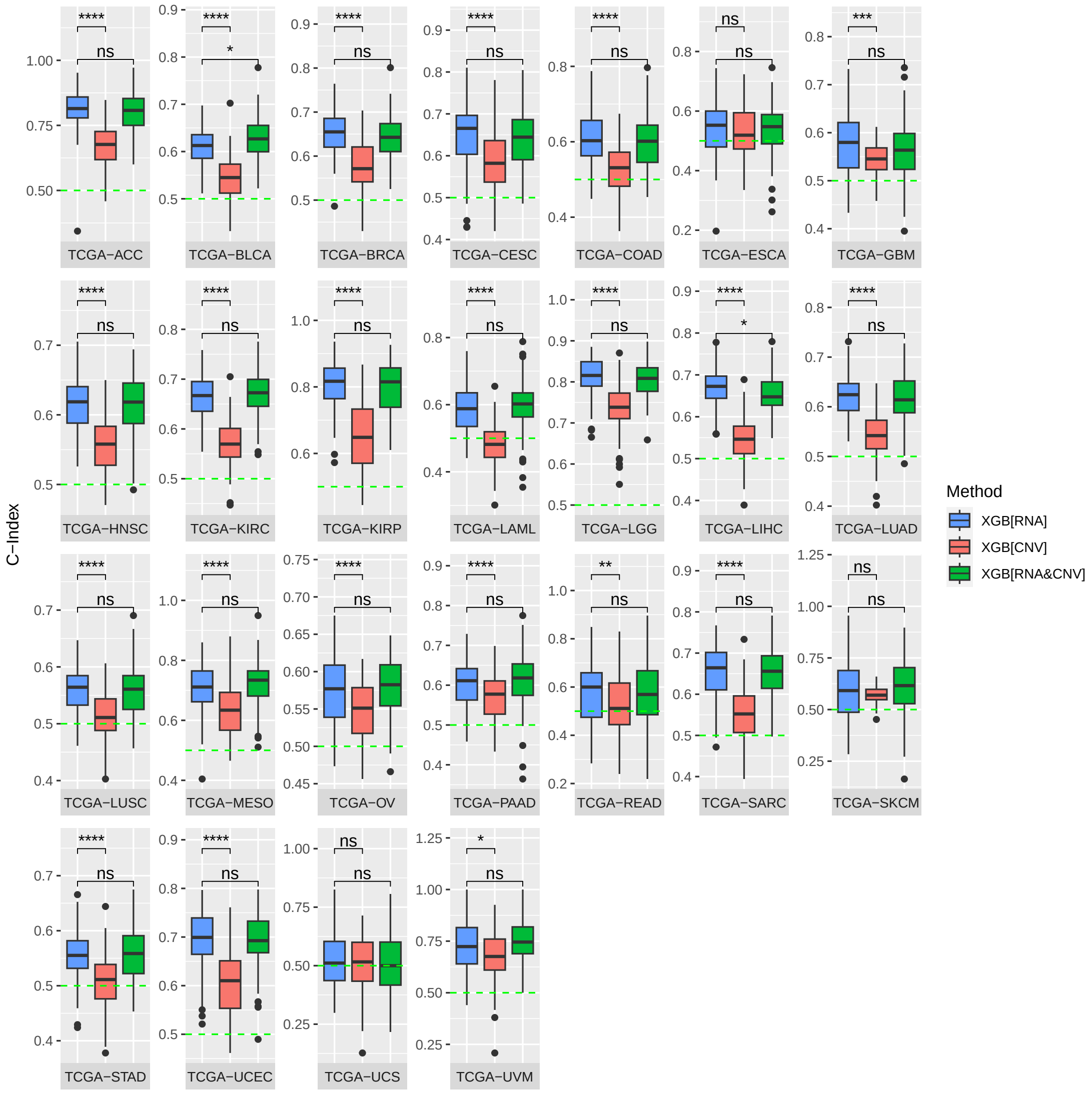


Figure 5.18: Pan-cancer survival prediction performance on copy number variation (CNV) data. Comparison of the prediction performance of the pan-cancer XGBoost method on 25 different TCGA cancer cohorts, trained on gene expression data (XGB[RNA]), copy number variation data (XGB[CNV]), and gene expression data in combination with copy number variation data (XGB[RNA&CNV]). Performance is depicted by C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $*: p \leq 0.05$, $**: p \leq 0.01$, $***: p \leq 0.001$, $****: p \leq 0.0001$.

Figure 5.18 shows the survival prediction performance of the pan-cancer XGBoost survival prediction model in all three settings. Although copy number variations seemed to be predictive for survival at least to some extent (C-Index > 0.5) in many of the investigated TCGA cohorts, they were significantly less predictive for patient survival than gene expression in the majority of cancer types (22 of 25 TCGA cohorts). When copy number variations were combined with gene expression data, this conferred a significant improvement over survival prediction on gene expression data alone in only one cohort (TCGA-BLCA), while in 23 cohorts, the difference in C-Index was insignificant and one cohort (TCGA-LIHC) even showed a drop in performance when both gene expression and copy number variation data were incorporated into survival prediction. These results suggest that copy number variation data does contain information on cancer patient survival, but this information does not appear to be complementary to the information contained in gene expression data and could thus not significantly improve prediction performance in most cancer types.

Integration of Protein Expression Data

With 6,256 patients and 487 proteins, protein expression measured by RPPA is the sparsest of the data modalities on which we evaluated our pan-cancer XGBoost survival prediction approach. For a fair comparison between survival prediction on the more comprehensive TPM-normalized gene expression data with 8,045 patients and survival prediction on protein expression data, we thus evaluated our pan-cancer survival prediction approach on different patient (sub-)sets from one of the two data modalities alone or from the combination of both modalities. More precisely, we evaluated pan-cancer survival prediction on gene expression data and all patients with measured gene expression (XGB[RNA_all_patients]), on gene expression data and only patients for which protein expression data was also available (XGB[RNA_patient_subset]), on protein expression data and all patients for whom protein expression measurements were available (XGB[protein_patient_subset]), and on the combination of gene expression and protein expression data and either only the patients with both gene expression and protein expression data available (XGB[RNA&protein_patient_subset]) or all patients with measured gene expression data (XGB[RNA&protein_all_patients]), where for patients with only gene expression data available protein expression was treated as missing data.

Figure 5.19 shows the performance (measured as C-Index) of our pan-cancer XGBoost model in the 25 different TCGA cohorts, comparing the different patient sets and data settings trained for 100 replications each. When comparing survival prediction based on gene expression data and protein expression data on the same set of patients (XGB[RNA_patient_subset] vs. XGB[protein_patient_subset]), gene expression yielded significantly better results than protein expression in 10 of 24 TCGA cohorts (there was no protein expression data available for TCGA-LAML), survival prediction based on protein expression data outperformed survival prediction based on gene expression data in 5 cohorts and both modalities yielded similar performances (no significant differences) in 9 cohorts. However, only for one cohort (TCGA-COAD), the combination of gene expression and protein expression yielded significantly better survival prediction results than gene expression data alone, indicating that gene expression and protein expression data likely contain similar information on patient survival, with gene expression being slightly more informative than protein expression. When com-

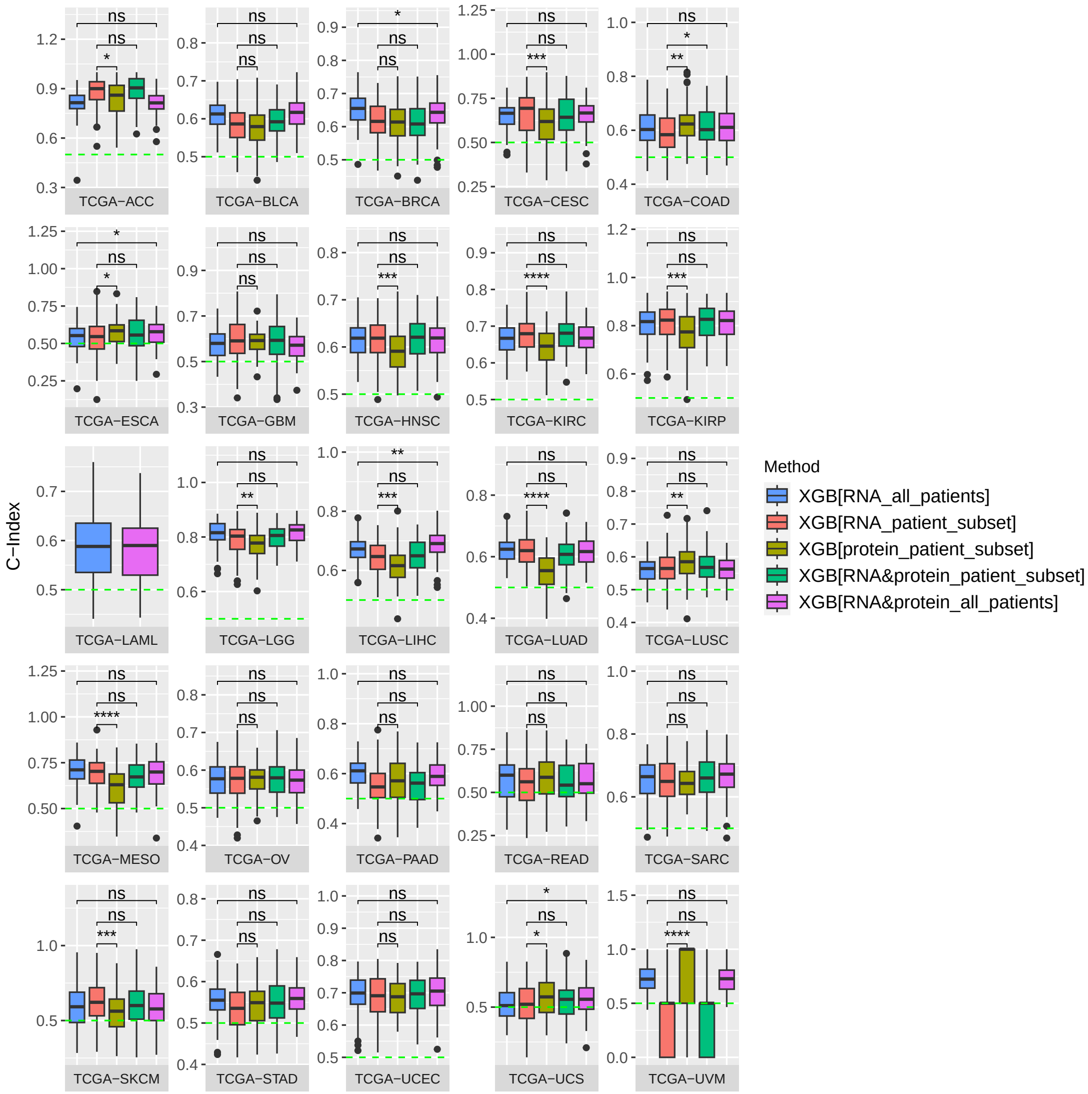


Figure 5.19: Pan-cancer survival prediction performance on protein expression data. Comparison of pan-cancer XGBoost trained on gene expression data and all patients with measured gene expression (XGB[RNA_all_patients]), gene expression and only patients with both gene and protein expression data available (XGB[RNA_patient_subset]), protein expression and all patients with protein expression data available (XGB[protein_patient_subset]), gene expression and protein expression and only patients with both gene expression and protein expression data available (XGB[RNA&protein_patient_subset]), and gene expression and protein expression and all patients with measured gene expression data (XGB[RNA&protein_all_patients]; unavailable protein expression is treated as missing data). Performance is measured by C-Index over training 100 replications. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $* : p \leq 0.05$, $** : p \leq 0.01$, $*** : p \leq 0.001$, $**** : p \leq 0.0001$.

paring survival prediction performance including all patients with measured gene expression rather than just those with both gene and protein expression available, the picture is similar:

For most (21 of 24) cohorts, adding protein expression for patients wherever this data type is available does not improve prediction performance over using gene expression alone, with TCGA-ESCA, TCGA-LIHC, and TCGA-UCS being the only three cohorts for which the combination of protein expression and gene expression improves survival prediction significantly.

5.3.9 Tumor Status Impacts Survival Prediction

The tumor status of a cancer patient refers to "the condition or state of the tumor at a particular time" (NCI Thesaurus version 23.05e; code C96643) and can assume the values "with tumor", "tumor free", or "unknown tumor status". For the TCGA data, information on tumor status is recorded for most cancer types in the clinical supplement data of the GDC database (the according files were retrieved from GDC data release v31.0). We hypothesized that patients who were tumor-free at a follow-up at some point after their initial cancer diagnosis should have a better prognosis than patients with tumor at the time of follow-up. To verify this assumption, we first split all patients with known tumor status from the 25 analyzed TCGA cohorts into two groups of tumor-free patients and patients with tumor, respectively. Then, based on this partition, we generated a Kaplan-Meier plot (Figure 5.20), which for each of the two groups visualizes the respective group's survival rate (i.e., the fraction of patients that are still alive after the respective time) over time. As expected, tumor-free patients have significantly better survival times compared to patients with tumor (logrank $p < 0.005$).

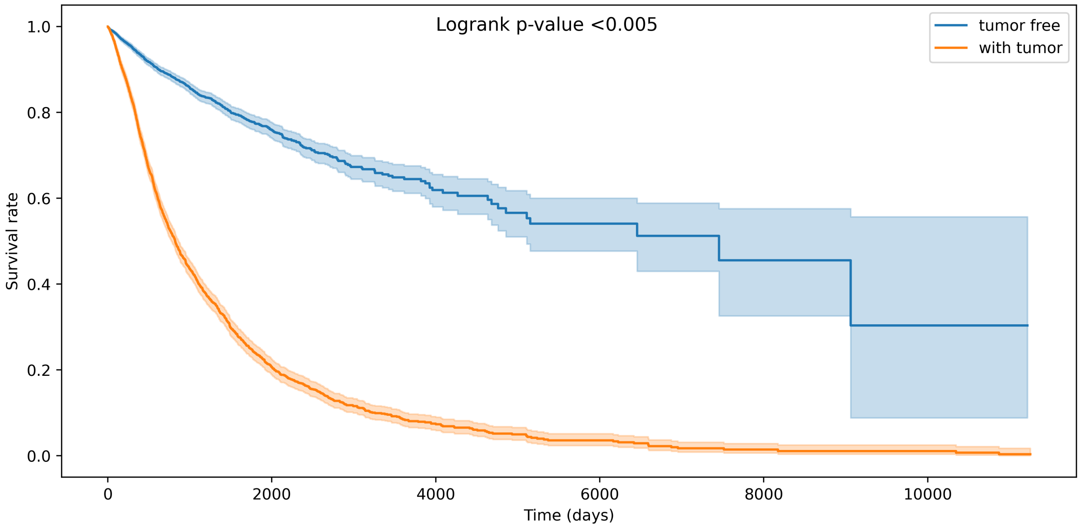


Figure 5.20: Kaplan-Meier plot by tumor status. The Kaplan-Meier plot shows the survival rate (fraction of patients still alive at a given time point) over time of tumor-free TCGA patients according to their tumor status compared with TCGA patients with tumor. Only TCGA patients with known tumor status "tumor free" or "with tumor" were included.

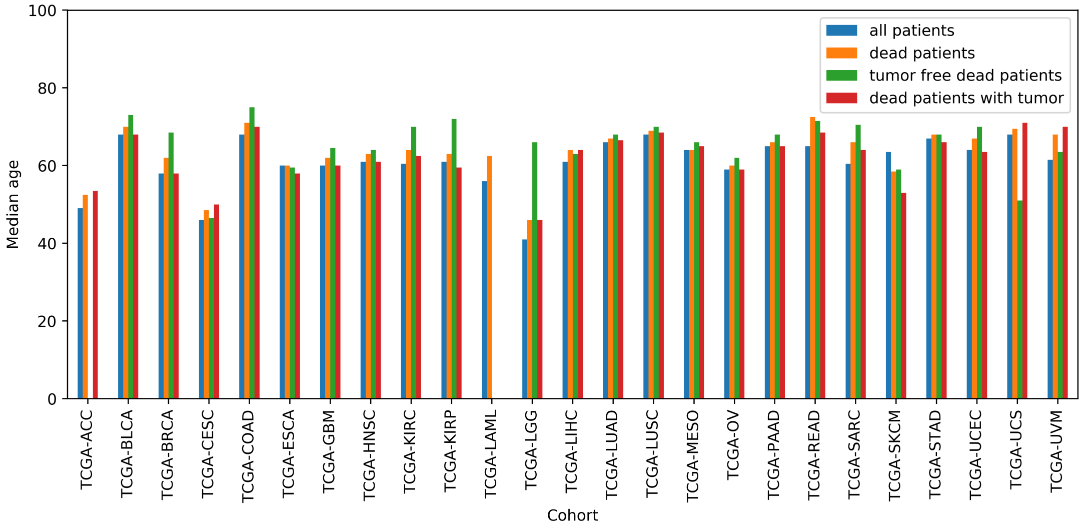


Figure 5.21: Median ages of TCGA patients according to cancer type and tumor status. For each of the 25 evaluated TCGA cohorts, the median age of all patients belonging to that cohort is compared with the median age of all dead patients, all dead tumor-free patients, and all dead patients with tumor. For the latter two categories, only patients with known tumor status ("tumor free" or "with tumor") were considered. If no patients with the respective tumor status were available for a cohort, no median age was computed.

When stratifying the TCGA patients by censoring status and tumor status and comparing the median ages of the different groups, it becomes apparent that the group of tumor-free patients that have died during the study period has a larger median age than dead patients with tumor in 19 of 23 TCGA cohorts with recorded tumor status (Figure 5.21). Furthermore, in 16 of these cohorts, the median age of the dead tumor-free patients is also larger than the median age of all patients of the respective cohort and all patients of the cohort that have died during the study period (Figure 5.21). A possible explanation for the overall older age of dead tumor-free patients compared to other patient groups could be that at least part of the tumor-free patients might not have died directly from their cancer, but rather from old age or other age-related co-morbidities. This would be a case of competing risks, where other types of events (e.g., death due to age-related causes) may forestall the event of interest (e.g., cancer-related death)[184]. Since the risk of dying from age-related causes rather than cancer can be expected to be especially high in older, tumor-free patients, we hypothesized that this could be a confounding factor for cancer survival prediction and by not considering the death of tumor-free patients as a death from cancer, we would be able to at least partly remove this confounding factor and might be able to improve our survival prediction model's performance.

To test this hypothesis with our gene expression-based pan-cancer XGBoost approach, we inverted the censoring status of all dead patients with recorded "tumor free" status, thus

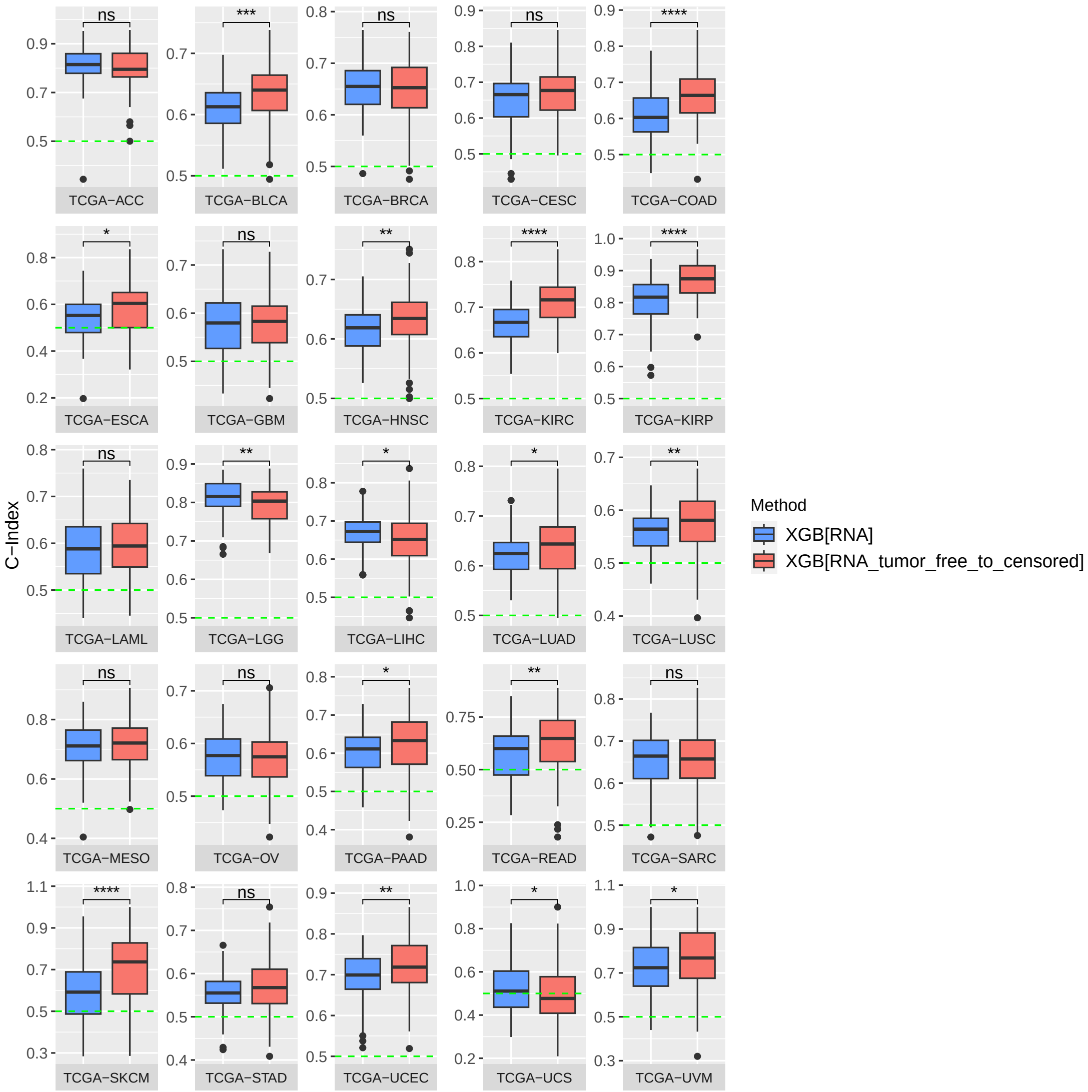


Figure 5.22: Pan-cancer survival prediction performance under consideration of the tumor status. Comparison of the prediction performance of the pan-cancer XGBoost method trained on gene expression data of 25 different TCGA cancer cohorts, either not taking the patients' tumor status into consideration (XGB[RNA]) or taking the tumor status into consideration and regarding the survival times of dead tumor-free patients as censored (XGB[RNA_tumor_free_to_censored]). Performance is depicted by C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $* : p \leq 0.05$, $** : p \leq 0.01$, $*** : p \leq 0.001$, $**** : p \leq 0.0001$.

considering these patients as censored when training and evaluating the XGBoost survival prediction method. Figure 5.22 shows the survival prediction performance (measured as C-Index) of the pan-cancer XGBoost approach on the 25 analyzed TCGA cohorts evaluated in the setting where dead tumor-free patients are considered as censored (XGB[RNA_tumor_free_to_censored]) compared to the original setting where these patients were considered as uncensored (XGB[RNA]). In agreement with our hypothesis, regarding dead tumor-free patients as censored instead of uncensored improves the survival prediction performance of our pan-cancer XGBoost model significantly ($p < 0.05$) in 13 (BLCA, COAD, ESCA, HNSC, KIRC, KIRP, LUAD, LUSC, PAAD, READ, SKCM, UCEC, and UVM) of the 25 TCGA cohorts. Interestingly however, it has the opposite effect in three other cohorts (LGG, LIHC, and UCS), where keeping the original censoring status for dead tumor-free patients leads to a significantly better survival prediction performance.

5.4 Discussion

Cancer is a leading cause of premature death worldwide[57]. To inform treatment decisions and ultimately reduce cancer mortality, it is critical to be able to quantify a patient's risk and estimate prognosis. Therefore, cancer survival prediction is an important computational task. Our goal was to develop a cancer survival prediction method with high biological plausibility. We achieved this by combining a gradient tree boosting approach for survival prediction with network propagation on a comprehensive high-confidence PPI network for the identification of a biologically plausible pan-cancer survival network.

More precisely, we applied XGBoost[31] tree ensemble learning to gene expression data from each of 25 cancer cohorts from TCGA and showed competitive performance of this single-cohort XGBoost approach with established survival prediction methods. To address the problem of low sample numbers in the single-cohort training approach, where a separate XGBoost survival prediction model is trained for each cohort, and to enable the identification of cross-cohort survival features, we then implemented pan-cancer training, where all 25 TCGA cancer cohorts were used jointly to train an XGBoost model, and showed improved performance of the pan-cancer approach over the single-cohort approach. We believe that a large part of this improvement is likely due to the larger sample numbers in pan-cancer training compared to single-cohort training. This assessment is consistent with findings from other prediction tasks like drug sensitivity prediction[116] and is supported by survival prediction results from pan-cancer XGBoost training—including feature selection and hyperparameter tuning—on randomly sampled patient subsets of different sizes (Supplementary Figure A.2). In these additional experiments, survival prediction performance deteriorated for many of the evaluated cancer types when the sample size of the training data was reduced. Interestingly, we also observed that pan-cancer features generalized well across different cancer types, whereas features from the single-cohort approach tended to be more cancer-type-specific. There were also notable differences in the types of features used in single-cohort and pan-cancer survival prediction. While 40.4% of the features used in the single-cohort approach were protein-coding genes, this proportion increased to 56.5% in the pan-cancer approach, and correspondingly, the proportion of other—possibly more tissue-specific—feature types such as lncRNAs and processed pseudogenes decreased in pan-cancer training compared to single-cohort training.

To provide biological plausibility for our pan-cancer survival prediction approach, we then applied the NetCore[13] network propagation method to the feature importance scores extracted from the pan-cancer XGBoost models of 100 training replications and identified a pan-cancer survival network comprising 103 genes. This survival network is strongly associated with the TME, as we showed by ORA and correlation with patient immune status. The TME has been found to play important roles in tumor initiation, growth, invasion, metastasis, as well as response to therapies[92,140], making the association we found between the TME and cancer survival highly plausible. Furthermore, our findings highlighted the particular importance of the hypoxic and immune-related aspects of the TME for cancer survival and identified a moderately negative correlation ($R = -0.55$) between survival prediction performance and age, suggesting that the aging TME may be more difficult to interpret by machine learning approaches than younger and presumably more intact states of the TME and supporting the notion that the aging TME could influence cancer progression and survival[54]. Survival prediction can potentially benefit from these findings by taking age-specific effects into account, for example by considering age when splitting the patients into training and test data.

In addition to applying our pan-cancer survival prediction method to gene expression data, we also evaluated the method on additional omics data modalities, including mutation, copy number variation, and protein expression data. DNA methylation was not evaluated as a data modality because the DNA methylation data available through the GDC data portal was not consistent across patients with respect to the measurement platform, thus making it difficult to compare patients. More specifically, the DNA methylation data available for the analyzed cancer patients was measured by either only one or both of two different generations of Illumina DNA methylation arrays (Human Methylation 27 and HumanMethylation 450), which are not directly comparable. Even when considering only the intersection of methylation sites measured by both arrays, substantial batch effects were observable between the DNA methylation beta values obtained from Human Methylation 27 and HumanMethylation 450 arrays, respectively (exemplarily shown for TCGA-COAD in Supplementary Figure A.5), making it difficult to compare patients with methylation beta values measured by Illumina Human Methylation 27 to patients with beta values measured by Illumina Human Methylation 450. Of all evaluated omics data modalities, gene expression showed to be the most informative, which is consistent with previous findings from different biomedical prediction tasks, including the results of Costello et al.[41] on drug sensitivity prediction and the results of Vale-Silva and Rohr[176] on pan-cancer survival prediction, but interestingly in contrast to the findings of Cheerla and Gevaert[29] on the same task. After gene expression, protein expression was the second most predictive datatype for cancer survival, outperforming survival prediction based on gene expression data for 5 of the 25 evaluated cancer cohorts.

Notably, protein expression data was the sparsest of the evaluated data modalities, both in terms of patient numbers and the number of measured features (i.e., proteins). Therefore, the relatively good performance of survival prediction based on protein expression data is noteworthy and it can be speculated that the predictive power of protein expression data for cancer survival prediction might be underestimated from our results. Hence, the survival prediction performance of the pan-cancer XGBoost model trained on protein expression data might be further improved with a more comprehensive protein expression dataset providing measurements for more patients and more proteins. Recalling the central dogma of molecular biology, genetic information can be transcribed from DNA to RNA and translated from RNA to protein[47,46]. Accordingly, proteins are usually the functional molecules that link genotype to phenotype[90] and we can speculate that protein expression should therefore be the data modality that is most immediately related to the phenotype of cancer survival, followed by gene expression and the genomic data modalities that measure mutations and copy number variations. Indeed, our results are largely consistent with these considerations, in that protein expression and gene expression yield better survival prediction performances than mutation and copy number variation data and the superior performance of gene expression over protein expression data for most cancer types might be explained by the greater sparsity of protein expression data compared to gene expression data, for which more patients and substantially more features are measured than for protein expression data.

Some TCGA patients with reported death during the respective study period have a recorded "tumor free" status, meaning they did not have any remaining tumor at some point after their initial diagnosis. For these patients, we speculated that they might have died from other—potentially age-related—causes rather than their cancer and hypothesized that by regarding these patients as censored rather than uncensored during survival prediction, we might be able to improve the prediction performance of our pan-cancer XGBoost model on gene expression data as the most predictive data modality. Indeed, we could show that considering the tumor status as described above significantly improved survival prediction performance for 13 of the 25 evaluated cohorts, supporting this hypothesis. Surprisingly, however, for three other cohorts, regarding dead tumor-free patients as censored significantly deteriorated performance instead. A possible explanation for this result could be that at least some of the tumor-free patients, especially in these three cohorts, were tumor-free at some point, but the cancer might have recurred later on, ultimately leading to death. For these patients, considering their survival time as censored might have led to losing important information on their survival, negatively impacting the survival prediction performance.

In summary, we have introduced a cancer survival prediction approach based on XGBoost tree ensemble learning and have shown that single-cohort training on gene expression data demonstrates highly competitive performance with established survival prediction methods, pan-cancer training significantly improves survival prediction performance compared to single-cohort training, and gene expression is the most informative data modality for cancer survival, closely followed by the more sparse protein expression data. Additionally, we found that taking patient tumor status into account can further improve survival prediction performance, suggesting that cancer and other—possibly age-related—causes of death may act as competing risks. Furthermore, we combined the gene expression-based pan-cancer survival prediction approach with network propagation to gain biological plausibility for the survival prediction step and identified a pan-cancer survival network, which highlighted the importance of the aging tumor microenvironment for cancer survival.

5.4.1 Limitations

Our survival prediction method is based on cancer patient data made available by the TCGA consortium. Although TCGA includes a relatively large number of patients from a wide variety of cancer types, all TCGA data has been processed according to uniform protocols. As a result, the TCGA data can be assumed to be at least somewhat consistent. When data from a new, non-TCGA domain that was processed according to different protocols should be used instead for survival prediction with the method trained on TCGA domain data, different marginal distributions of the source domain used for training and the target domain that should be used for survival prediction can be a problem. As described in Section 3.2.3, a possible solution to this type of problem is transfer learning. However, while transfer learning approaches such as pre-training the model on the source domain and then transferring it to the target domain, where it is fine-tuned, have been successfully applied and are widely used with neural networks[188], transfer learning is not as easily and widely applicable to other machine learning frameworks like XGBoost. Although with TransBoost[166] (cf. Section 4.3), there is a transfer learning approach building on XGBoost, it cannot be applied to already trained XGBoost models. Instead, TransBoost modifies the XGBoost implementation to allow transfer learning by co-learning two parallel models with shared tree structures but different node weights on the source domain instances and a combination of target domain instances and weighted source domain instances, respectively. This way, the method at the same time adjusts for different distributions of the source and target domains and trains a prediction model applicable to the target domain. However, TransBoost is implemented for classification problems only and is not readily applicable for regression or survival prediction problems.

6

Transfer Learning in Cancer Survival Prediction

In this chapter, different transfer learning scenarios for improving cancer survival prediction are explored and evaluated.

6.1 Motivation

Knowledge learned on a task with abundant data can be used to improve the performance of machine learning models on related tasks with less data through transfer learning (cf. Section 3.2.3). This can be especially useful in biomedical prediction tasks like cancer survival prediction, where large quantities of training data are usually not available. Transfer learning can be applied to cancer survival prediction in one of two ways: Either a machine learning model can be trained on a large dataset for an auxiliary prediction task that is in some way related to cancer survival, or the model can be directly trained for cancer survival prediction on a medium-sized cancer survival dataset such as the The Cancer Genome Atlas (TCGA) pan-cancer dataset and then transferred to another independent small cancer survival dataset obtained from a different clinical study or database. However, while transfer learning can potentially improve the performance of cancer survival prediction models, it is not compatible with every machine learning framework and prediction task. For instance, while with TransBoost[166] (cf. Section 4.3) there has been an attempt to combine XGBoost and transfer learning, the method can only be applied to classification tasks. To the best of our knowledge, there is no method implementing transfer learning for XGBoost that can be applied to survival prediction. Thus, despite XGBoost showing compelling results in cancer survival prediction, it cannot easily be combined with transfer learning to transfer knowledge from medium-sized to independent small cancer survival datasets or to leverage knowledge from other prediction tasks for cancer survival prediction. Therefore, in addition to pan-cancer survival prediction with XGBoost, we also investigated the application of neural networks, a machine learning methodology that can be conveniently combined with transfer learning, for transfer learning in cancer survival prediction. To this end, we developed two application scenarios of transfer learning in cancer survival prediction, which are described in this chapter: In the first scenario, we pre-trained a survival prediction neural network on the TCGA pan-cancer gene expression dataset and then fine-tuned it on smaller cancer survival datasets from different independent studies, and in the second scenario, we pre-trained neural networks on the tasks of tissue type classification and age prediction from gene expression data and transferred the trained networks to cancer survival prediction on the TCGA pan-cancer dataset. While the prediction of tissue type and age is not specific to cancer and indeed we pre-trained the tissue type and age prediction models on gene expression data from deceased donors with various non-cancer-related causes of death, both tasks are to some extent related to cancer survival prediction in that there are tissue- and cell-type-specific differences in tumorigenesis and in the organization of oncogenic signaling pathways[151], and aging is correlated with cancer incidence[50,130] and shares some of its hallmarks with cancer[121]. Therefore we think that tissue type and age prediction are promising candidate tasks for transfer learning.

6.2 Methods

In this section, the methodology used to pre-train survival prediction as well as tissue type and age prediction models is introduced and the investigated transfer learning strategies for cancer survival prediction are explained.

6.2.1 Data and Preprocessing

All transfer learning experiments described in this chapter are based on RNA-seq gene expression data.

For the first scenario, where a neural network was pre-trained and fine-tuned on gene expression data for the task of cancer survival prediction, TPM-normalized RNA-seq gene expression data and the corresponding clinical data were obtained from the GDC data portal and downloaded with the TCGAbiolinks R package[40,128,159]. More specifically, for pre-training, the TCGA dataset (GDC data release v32.0) was used, which comprised 60,616 gene expression values and 8,045 patients from 25 cancer types after excluding patients with incomplete data or inconsistent survival information. For fine-tuning the pre-trained model, datasets from three different cancer studies (CPTAC-3, CDDP_EAGLE-1, and CGCI-BLGSP were evaluated, which were all downloaded from GDC release v36.0 and comprised TPM-normalized RNA-seq gene expression data and corresponding clinical data. The Clinical Proteomic Tumor Analysis Consortium (CPTAC) is an effort by the National Cancer Institute (NCI) to explore the molecular basis of cancer through large-scale proteome and genome analysis. In the framework of CPTAC, the CPTAC-3 study investigated molecular and clinical data of endometrial, lung, kidney, brain, head and neck, and pancreatic cancers and provides both gene expression and survival data for primary cancers of 763 patients. Unlike TCGA and CPTAC-3, which are both pan-cancer studies, the other two studies, CDDP_EAGLE-1 and CGCI-BLGSP, each only investigated a specific cancer type. More specifically, the CDDP Integrative Analysis of Lung Adenocarcinoma (CDDP_EAGLE-1) project provides data for bronchus and lung adenomas and adenocarcinomas for 50 cases, 44 of which could be used in our transfer learning experiments because both primary-tumor gene expression and survival data were available. The goal of the Cancer Genome Characterization Initiative (CGCI) on the other hand, is to catalog genomic alterations in rare adult and pediatric cancers. In the framework of the CGCI, the CGCI Burkitt Lymphoma Genome Sequencing Project (CGCI-BLGSP), which we used in our transfer learning experiments, provides primary cancer gene expression and survival data for 29 patients with mature B-cell lymphoma, which is a type of non-Hodgkin lymphoma that is most prevalent in children and young adults. For pre-training and transfer learning, the 60,616 genes common to all cancer datasets were selected and gene expression data was log-transformed by $\log_2(\text{TPM}+1)$ to reduce the impact of extremely large values and data skewness[195].

For the second transfer learning scenario, where a neural network was pre-trained for tissue type or age prediction or both and then transferred to the cancer survival prediction task, TPM-normalized RNA-seq gene expression data from the GTEx project[118] was used for pre-training. The GTEx data was retrieved from the GTEx Portal on August 17 and 18, 2021 and included gene expression data for 56,156 genes and 17,382 samples from 30 tissue types and 948 deceased donors with tissue type information and donor age information in the form of 10-year age brackets. For transfer learning on cancer survival prediction, we used TPM-normalized gene expression and clinical data of 25 cancer types from TCGA, provided through the GDC data portal (GDC data release v32.0) and downloaded with the TCGAbiolinks R package[40,128,159]. The TCGA survival data comprised 60,616 gene expression values and 8,045 patients from 25 cancer types after excluding patients with incomplete data or inconsistent survival information. For pre-training and transfer learning, we only used the 55,617 genes common to the GTEx and TCGA datasets and log-transformed all gene expression values by $\log_2(\text{TPM}+1)$.

For both transfer learning scenarios, we used the same 25 cancer types from TCGA as previously used for survival prediction with XGBoost (cf. Section 5.2.1), excluding TCGA cancer cohorts with less than 20 uncensored patients. Analogously to what we described in Section 5.2.1, in the case of multiple tumor samples from the same patient, the sample with the lexicographically highest sample ID was selected in all investigated cancer datasets.

6.2.2 Transferring Survival Information from TCGA to Smaller Cancer Datasets

The goal of the first transfer learning scenario was to transfer knowledge within the same task of cancer survival prediction, but from one dataset to other independent datasets generated by different studies. To this end, we pre-trained a fully connected feed-forward neural network with a Cox regression output layer for cancer survival prediction on a TCGA pan-cancer dataset comprising 25 cancer types and then transferred the knowledge learned by this network to survival prediction on independent, substantially smaller cancer datasets. The survival prediction neural network had a single-neuron output layer with linear activation and without bias and was trained using negative partial log-likelihood (cf. Section 3.3.2, Equation 3.29) as a loss function and the C-Index (cf. Section 3.3.3) as a performance metric. For pre-training, the TCGA data was first split into 80% training and 20% test data, while keeping similar distributions of censored and uncensored patients and similar cancer type distributions in both the training and the test data. The training data was then further split into 80% training and 20% validation data under consideration of the censoring status and cancer type distributions, with the training data being used for model training and the validation data being used for hyperparameter optimization and early stopping. Then, the gene expression data was scaled between 0 and 1 based on the training data using scikit-learn's[135] MinMaxScaler, and the validation and test data were scaled accordingly. To find the combination of hyperparameters yielding the best model performance, we used the Optuna framework[5] with TPE, which is a Bayesian optimization method. The optimized hyperparameters included network architecture, such as the number and sizes of hidden layers, as well as other hyperparameters such as batch size, learning rate, regularization parameters, and dropout (the selected hyperparameters and tested parameter ranges are displayed in Supplementary Table B.5). We used the adaptive moment estimation (Adam) optimizer[99] for training and selected the micro-average C-Index on the TCGA validation data as the optimization objective. The micro-average C-Index was computed as the weighted average over C-Indices of the 25 cancer types contained in the validation data, weighted by the number of patients from each cancer type. More formally, it is defined as

$$CI_{\text{avg}} = \frac{\sum_{i=1}^{T} n_i c_i}{\sum_{i=1}^{T} n_i}, \tag{6.1}$$

where T is the number of evaluated cancer types, n_i is the number of patients from the cancer type with index i, and c_i is the C-Index computed from the patients of the same cancer type[98].

For transfer learning, we then extracted all weights and biases from the neural network trained with the best hyperparameter combination (i.e., the combination of hyperparameters that yielded the best micro-average C-Index on the validation data during hyperparameter tuning) for survival prediction on the TCGA data and fine-tuned them on different independent smaller cancer datasets (*fine-tuning* model type), which comprised data from the CPTAC-3, the CDDP_EAGLE-1 or the CGCI-BLGSP study. Before fine-tuning, however, we first scaled the gene expression data from all datasets using the MinMaxScaler that we had fitted on the TCGA training data before pre-training to ensure comparability between datasets. Because the CPTAC-3 study is a pan-cancer study reporting gene expression and survival data for seven different cancer types (glioma, bronchus and lung adenomas and adenocarcinomas, kidney adenomas and adenocarcinomas, uterus adenomas and adenocarcinomas, pancreas ductal and lobular neoplasms, lung and bronchus squamous cell neoplasms, and squamous cell neoplasms of other and ill-defined sites), we evaluated separate fine-tuning on each of the cancer types as well as fine-tuning on the pan-cancer CPTAC-3 dataset as a whole. For the other two studies (CDDP_EAGLE-1 and CGCI-BLGSP), we only conducted one fine-tuning experiment each since these two studies only report gene expression and survival data of a single cancer type (bronchus and lung adenomas and adenocarcinomas and mature B-cell lymphomas, respectively). In each fine-tuning experiment, we conducted 5-fold cross-validation on the respective small cancer dataset using the scikit-learn library[135] to obtain a reliable assessment of prediction performance for this dataset. In k-fold cross-validation, a set of observations if first randomly divided into k groups, called folds, of approximately equal size[89]. Then the model is trained on the last $k-1$ folds, while the first fold is held back as test data, which is then used to evaluate the performance of the trained model[89]. This procedure of training the model on $k-1$ folds while withholding the remaining fold is repeated k times for the different folds, until each of the k folds has been used once to evaluate model performance[89]. This results in k different evaluations of prediction performance, which in combination gives a more reliable estimation of model performance than evaluating the model only on one set of observations[89]. In each iteration of the 5-fold cross-validation on one of the small cancer datasets, we first split the four training folds further into 80% training and 20% validation data. Using these training and validation data, we then fine-tuned the weights and biases extracted from the pre-trained neural network using the Adam optimizer[99] with a learning rate of 0.1 times the learning rate used for pre-training on TCGA data. During fine-tuning, we applied early stopping with patience 5 on the validation data to prevent overfitting of the model on the training data. Finally, we used the respective test fold to evaluate the fine-tuned cancer survival prediction model. To this end, we computed the C-Index on the test fold if the respective test data contained only a single cancer type and the micro-average C-Index on the test fold in the case of the CPTAC-3 pan-cancer dataset.

To be able to assess the effect of transfer learning on survival prediction performance, we additionally trained survival prediction models 'from scratch' as a negative control for each of the evaluated datasets (*scratch* model type). To this end, we conducted another 5-fold cross-validation on each of the target cancer datasets. All steps of these control experiments were analogous to the fine-tuning experiments, except that the weights and biases of the trained neural networks were randomly initialized instead of initializing them with the pre-trained weights and biases and the same learning rate as used for pre-training on TCGA data was used to train the model (instead of using the reduced fine-tuning learning rate). All other model hyperparameters were also selected to be the same as for pre-training to ensure comparability between fine-tuned models and models trained from scratch.

Furthermore, to assess how well the pre-trained model was already adapted to predicting survival on the independent small cancer datasets, we also directly applied the pre-trained model to the target datasets without further fine-tuning (*pre-training* model type). To this end, we performed a 5-fold cross-validation, where in each iteration, we only evaluated the pre-trained model on the respective test split, such that all three model types (*fine-tuning*, *scratch*, and *pre-training*) were evaluated on the same patients and results were therefore directly comparable.

6.2.3 Transferring Knowledge from Tissue Type Classification and Age Prediction to Cancer Survival Prediction

In contrast to the first transfer learning scenario, where we investigated the transferability of knowledge learned for cancer survival prediction from a larger cancer survival dataset to smaller cancer survival datasets, the second transfer learning scenario explored the transferability of knowledge not only between different datasets, but also between different prediction tasks. More specifically, the transferability of knowledge learned from tissue type classification and age prediction to cancer survival prediction was investigated. To this end, three neural networks were first trained on GTEx gene expression data to predict tissue type, age, and both tissue type and age. For pre-training on all three prediction tasks, only genes measured in both the GTEx and the TCGA datasets were used to enable the transfer of the model from one dataset to the other.

In the first step of pre-training, the GTEx gene expression data was randomly split into 80% training and 20% test data, and the training data was further split into 80% training and 20% validation data. The splitting procedure took into account the donor of each sample so that all samples from the same donor were assigned to the same data split (training, validation, or test) to avoid data leakage and potentially inflated model performance. In addition, it accounted for the distribution of the target variable(s) such that according to the task at hand, either the tissue type distribution, the age distribution, or both distributions were similar across training, validation, and test splits. In the second step, gene expression was scaled between 0 and 1 by fitting a MinMaxScaler implemented by the scikit-learn library[135] to the training data and then applying it to training, validation, and test data. Next, class imbalances in the training data were addressed: Tissue types and 10-year age brackets are highly imbalanced in the GTEx dataset, where some tissue types and age brackets are vastly over-represented compared to others (Supplementary Figure A.6). This can bias the model towards predicting the majority classes over the minority classes, while still showing relatively good accuracy, and may lead to poor prediction performance, especially for under-represented classes. There are different strategies to counter this problem, including undersampling, where for over-represented classes only a subset of samples is selected, such that the new class size resembles the size of the smallest class, oversampling, where minority classes are supplemented with multiple copies of samples from the respective class to match the size of the largest class, or weighting strategies, where classes or samples are weighted according to class sizes. Class or sample weighting can compensate for class imbalances by assigning higher weights to minority classes or samples from minority classes than to majority classes when computing the loss. For pre-training our neural networks, we decided to use sample weighting because it doesn't have the disadvantages of undersampling, where some samples from the majority classes are disregarded, potentially leading to loss of information, or oversampling, where the size of the dataset is artificially increased, slowing down model training without adding further information. For the tissue type classification task, we used scikit-learn's[135] *compute_sample_weight* function with the *class_weight* parameter set to '*balanced*' to compute sample weights for all GTEx samples, while for the age prediction task, we directly implemented class weighting into the loss function. For all pre-training tasks of predicting tissue type, age, and both tissue type and age, we used the Adam optimizer[99] and tuned model hyperparameters on the training and validation data using the Optuna framework[5] with TPE for hyperparameter sampling. Among the optimized hyperparameters were the number of hidden layers of the respective neural network, layer sizes, activation function, regularization, and dropout parameters, but also training-related parameters like learning rate, batch size, and optimizer-specific parameters like weight decay and beta parameters of the Adam optimizer (a full list of optimized hyperparameters and their optimal values for each of the pre-training tasks can be found in Supplementary Tables B.6, B.7, and B.8). Additionally, we used early stopping with patience 5 on the validation data in all pre-training settings to avoid overfitting. The three different pre-training settings are explained in more detail in the following subsections.

Pre-Training for Tissue Type Classification

The first evaluated pre-training task was tissue type classification on GTEx gene expression data. In this pre-training task, the goal was to predict which of 30 tissue types a sample was from. To this end, a fully connected neural network with 30 output units and softmax activation in the output layer was trained using categorical cross-entropy as a loss function. The remaining architecture, such as the number and sizes of hidden layers, and other model hyperparameters were optimized using the Optuna hyperparameter tuning framework[5], as described in the previous section. To assess the performance of each hyperparameter combination, the respective models were evaluated on the validation data with the multi-class version of Matthew's Correlation Coefficient (MCC), which is based on the confusion matrix C and is defined as

$$MCC = \frac{cs - \sum_{k=1}^{K} p_k t_k}{\sqrt{\left(s^2 - \sum_{k=1}^{K} p_k^2\right)\left(s^2 - \sum_{k=1}^{K} t_k^2\right)}}, \qquad (6.2)$$

where K is the number of classes, $c = \sum_{k=1}^{K} C_{kk}$ is the number of correctly predicted samples, $s = \sum_{i=1}^{K} \sum_{j=1}^{K} C_{ij}$ is the total number of samples, $p_k = \sum_{i=1}^{K} C_{ki}$ is the column total of the confusion matrix, which indicates the number of times each class k was predicted, and $t_k = \sum_{i=1}^{K} C_{ik}$ is the row total of the confusion matrix, indicating the number of times each class k truly occurred[64]. Compared to using accuracy as a performance metric, Matthew's Correlation Coefficient has the advantage that it can be applied to unbalanced prediction problems without the majority class or classes dominating its value as would happen with accuracy[64,33].

Pre-Training for Age Prediction

The second pre-training task we evaluated was age prediction, where we trained a neural network to predict sample donor ages by classifying samples into 10-year age brackets based on gene expression. However, predicting age brackets is not a typical classification task, where the different classes are independent from each other. For instance, when the true age of an individual is between 60 and 69, predicting the age bracket 50-59 would be closer to the true age and thus better than predicting for example the age bracket 20-29. Typical classification models that use loss functions such as the categorical cross-entropy cannot take this type of dependency between classes into account. However, the age prediction task can instead be formulated as an ordinal regression problem, where the distance between classes is considered in the loss function. For this formulation, the age classes must be encoded in a way that reflects the similarity between classes, and the loss function must penalize predictions that are further away from the true age class more than predictions that are closer to the true age class. To this end, the ordinal regression problem is converted into multiple binary classification problems, which is inspired by Cheng et al.[32]. Given K classes $Y = 1, ..., K$, an ordinal relation between the classes with $1 < 2 < ... < K$ is assumed and each class k is encoded as a $(K-1)$-dimensional vector $o^{(k)} = (o_1^{(k)}, o_2^{(k)}, ..., o_{K-1}^{(k)})$ with

$$o_i^{(k)} = \begin{cases} 1, & \text{if } i < k \\ 0, & \text{else.} \end{cases}$$

That is, considering for instance six age brackets with the age bracket 20-29 corresponding to class 1, age bracket 30-39 corresponding to class 2, and so on, class 1 will be encoded as $(0, 0, 0, 0, 0)$, class 2 as $(1, 0, 0, 0, 0)$, and class 3 as $(1, 1, 0, 0, 0)$, while class 6 will be encoded as $(1, 1, 1, 1, 1)$.

According to this formulation, the neural network trained for predicting age had $K-1$ nodes in the output layer and each of these output nodes used the sigmoid function as activation. Thus, the output of the kth output node can be interpreted as the probability that the age of a given individual is higher than the age bracket represented by class k. The network was then trained using a loss function that is based on binary cross-entropy. More precisely, the loss of each sample was computed as the mean over the $K-1$ output nodes, for each of which a weighted version of the binary cross-entropy was calculated. Using just the standard binary cross-entropy without introducing any weights would bias the neural network towards overpredicting classes in the middle range, while underpredicting classes with small or high numbers, even for class-balanced training data. The reason for this phenomenon lies in the encoding of the ordinal classes: The target value of the first position of the output encoding is 0 only for class 1 and 1 for all other classes, while the target value of the last position only becomes 1 for class K and 0 for all other classes, introducing a form of class imbalance with the same effects on model training. Hence, the model might learn to always predict 1 for the first position, while always predicting 0 for the last position. On the other hand, the numbers of zeros and ones in the target output become more and more balanced when moving towards classes in the middle range, making it easier for the model to learn the correct classes. A second source of bias comes from inherent class imbalances in the training data, which can influence the model towards overpredicting majority classes while underpredicting minority classes. To mitigate both sources of bias caused by the two different forms of class imbalance, we computed two weights for zeros and ones, respectively, for each output node based on the numbers of zeros and ones at this position in the target output encodings of the training data. Based on these weights, a weighted binary cross-entropy for each sample i and each output node k can be computed as

$$\text{crossentropy}_{ik} = -\left(w_k^{(1)} y_{ik} \log(\hat{y}_{ik}) + w_k^{(0)} (1-y_{ik}) \log(1-\hat{y}_{ik})\right), \tag{6.3}$$

where y_{ik} is the target value at the kth position of the output encoding of sample i, $\hat{y}_{ik}$ is the prediction of the kth output node for sample i, and $w^{(0)}$ and $w^{(1)}$ are weight vectors of length $K-1$ containing the weights that should be given to zeros and ones, respectively, in each of the $K-1$ output nodes. These weight vectors are computed as $w^{(0)} = \frac{1}{\max(n_0,1)}$ and $w^{(1)} = \frac{1}{\max(n_1,1)}$, where $n_0 \in \mathbb{N}^{K-1}$ and $n_1 \in \mathbb{N}^{K-1}$ are the numbers of zeros and ones respectively at each of the $K-1$ output encoding positions in the target y. For example, if the input data contained ten samples and four different classes with two samples from class 1, five samples from class 2, and three samples from class 4, then there would be 8 ones (from the samples of classes 2 and 4) and 2 zeros (from the samples of class 1) in the first position of the output encoding, 3 ones (from the samples of class 4) and 7 zeros (from the samples of classes 1 and 2) in the second position, and 3 ones and 7 zeros in the third position. Accordingly, the weight vectors would then become $w^{(0)} = (\frac{1}{2}, \frac{1}{7}, \frac{1}{7})$ and $w^{(1)} = (\frac{1}{8}, \frac{1}{3}, \frac{1}{3})$.

The complete loss function for the ordinal regression task can then be computed as the mean of weighted binary cross-entropies across the $K-1$ output encoding positions, averaged over the N input samples:

$$l_{\text{ordinal}} = -\frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{K-1}\sum_{k=1}^{K-1} w_k^{(1)} y_{ik}\log(\hat{y}_{ik}) + w_k^{(0)}(1-y_{ik})\log(1-\hat{y}_{ik})\right) \tag{6.4}$$

$$= -\frac{1}{N(K-1)}\sum_{i=1}^{N}\sum_{k=1}^{K-1} w_k^{(1)} y_{ik}\log(\hat{y}_{ik}) + w_k^{(0)}(1-y_{ik})\log(1-\hat{y}_{ik}) \tag{6.5}$$

The class prediction for a sample i can be derived from the predicted output encoding as the first position in the encoding that has a predicted value smaller than 0.5. For instance, the predicted output encoding $(0.9, 0.8, 0.6, 0.2, 0.1)$ would correspond with the predicted class 4.

The hyperparameters of the age prediction neural network, including the number and sizes of hidden layers, learning rate, and other model hyperparameters, were optimized using the Optuna hyperparameter tuning framework[5]. To this end, the weighted Cohen's κ with quadratic weights was used as a performance metric to evaluate the different hyperparameter combinations, and the hyperparameter combination that yielded the best weighted Cohen's κ score on the validation data was selected for pre-training the final age prediction model. Cohen's κ was first proposed by Jacob Cohen in 1960[38] to quantify the level of agreement between two judges or annotators on a classification problem with independent classes. Since a certain amount of the agreement between annotators is expected by chance, Cohen proposed to include this proportion of expected chance agreement into his κ score. More precisely, Cohen's κ is defined as

$$\kappa = \frac{p_o - p_c}{1 - p_c}, \tag{6.6}$$

where p_o is the proportion of units where the annotators agree and p_c is the proportion of units for which agreement is expected by chance. That is, the numerator $p_o - p_c$ reflects the proportion of cases in which the annotators agree beyond chance and the denominator $1 - p_c$ represents the proportion of cases for which disagreement between the annotators would be expected by chance. Hence, κ is the proportion of agreement between the annotators after removing chance agreement and positive values of κ indicate more agreement between annotators than expected by chance, while negative values of κ indicate less than chance agreement.

In ordinal regression, classes are not independent from each other, but have an ordinal relationship, and misclassification in a more distant class is worse than misclassification in a more proximate class. However, Cohen's κ as described above treats all disagreements between annotators equally. In 1968, Jacob Cohen proposed a generalization to his κ metric, the weighted Cohen's κ_w, which incorporates weights to account for different degrees of agreement or disagreement between the K classes[39]. The weighted Cohen's κ_w with agreement weights is defined as

$$\kappa_w = \frac{\sum_{i,j} w_{ij} p_{o_{ij}} - \sum_{i,j} w_{ij} p_{c_{ij}}}{w_{max} - \sum_{i,j} w_{ij} p_{c_{ij}}}, \tag{6.7}$$

where w_{ij} is the agreement weight between class i and class j, w_{max} is the maximum agreement score for complete agreement ($i = j$), $p_{o_{ij}}$ is the proportion of joint annotations observed in the cell ij of the $K \times K$ contingency table, and $p_{c_{ij}}$ is the proportion of joint annotations expected by chance in the same cell. To compute the agreement weights w_{ij} between class i and class j, different weighting schemes can be applied. For example, linear weights, which are inversely proportional to the distance between classes, can be computed as[189]

$$w_{ij} = 1 - \frac{|i-j|}{K-1} \tag{6.8}$$

with K being the total number of classes. Quadratic weights, which are quadratically decreasing for classes that are further away, can be computed with the following formula[189]:

$$w_{ij} = 1 - \frac{(i-j)^2}{(K-1)^2} \tag{6.9}$$

For the age prediction pre-training task, we used quadratic weights for computing the weighted Cohen's κ_w.

Pre-Training for Tissue Type Classification and Age Prediction

In the third pre-training setting, we trained a neural network to predict tissue type and age simultaneously from the gene expression data of GTEx samples. To this end, we trained a multitask model with shared hidden layers, followed by task-specific layers for each task. Similar to the pre-training on tissue type classification or age prediction alone, the model architecture (including the number of shared and task-specific layers) and other hyperparameters were optimized using Optuna[5] with multivariate TPE (see Supplementary Table B.8 for the selected hyperparameters). In addition to the prediction performance in tissue type classification and age prediction, we also optimized the size of the last shared layer (latent size) to be as small as possible without compromising the performance of the other two prediction tasks to prevent the model from simply selecting one large and very general shared layer and then shifting the learning of the actual prediction tasks to task-specific layers that would not be transferred to the survival prediction model. During hyperparameter tuning, we used Matthew's Correlation Coefficient to evaluate the tissue classification performance of the model, while the weighted Cohen's κ_w with quadratic weights was used for assessing the age prediction performance. The hyperparameter combination used for the final pre-training was then selected by manual inspection of the Pareto front of the optimization objectives (Figure 6.2), i.e. the set of all Pareto efficient solutions where none of the optimization objectives can be further improved without deteriorating another[37].

Transfering the Pre-Trained Model to Survival Prediction

Once training the neural network in one of the three described pre-training settings was complete, the knowledge learned by that model in the form of weights and biases could be transferred to the cancer survival prediction task. To this end, the first n hidden layers of the tissue type classification or the age prediction neural network, or the first n shared layers of the multitask neural network trained for both tissue type and age prediction (with $1 \leq n \leq$ total number of hidden/task-specific layers) were transferred to a new neural network with the same model architecture (i.e., the same number and sizes of hidden layers) as the respective pre-trained model. While in this way, the first n hidden layers of each new neural network were initialized with the learned weights of the respective pre-trained neural network, the remaining hidden layers were randomly initialized. Additionally, to adapt each new neural network for survival prediction, the output layer was replaced by a Cox regression output layer with linear activation and no bias term.

Each new neural network was then further trained for cancer survival prediction on the TCGA pan-cancer dataset in one of two different transfer learning settings: In the first setting, only the randomly initialized last layer(s) of the neural network were trained while freezing the n pre-trained layers (*transfer* setting), meaning that weights and biases of the pre-trained layers were not updated during training. In the second setting, on the other hand, both the randomly initialized last layer(s) and the pre-trained layers were further trained for the survival prediction task (*fine-tuning* setting). In addition to these two transfer learning settings, we evaluated another setting (*scratch* setting), where corresponding to each pre-trained model, another randomly initialized survival prediction neural network with the same model architecture and hyperparameters as the pre-trained model (except for the task-specific output layer) was trained from scratch and without transferring any pre-trained weights and biases. This neural network trained from scratch served as a negative control to assess the effect of transfer learning on survival prediction performance. In all three settings, the respective neural network was trained for survival prediction using the Adam algorithm[99] as the optimizer, the negative partial log-likelihood (cf. Section 3.3.2, Equation 3.29) as a loss function, and the C-Index (cf. Section 3.3.3) as a performance metric. In the *transfer* setting, the weights and biases of the transferred pre-trained layers were frozen and only the randomly initialized layers were trained using the same learning rate as used in the respective pre-trained model. In the *fine-tuning* setting, in contrast, two different learning rates were used for training the model, where the transferred pre-trained layers were fine-tuned with a learning rate of 0.1 times the learning rate of the pre-trained model, and the randomly initialized non-transferred layers were trained with the same learning rate as the respective pre-trained model. Lastly, in the *scratch* setting, all layers of the neural network were trained using the learning rate of the respective pre-trained model. In all three settings, all other model hyperparameters except for the learning rate were kept the same as for the respective pre-trained model to allow for a fair comparison between the settings.

To obtain a reliable assessment of the survival prediction performance in the different pre-training and transfer learning settings, we applied a stratified 5-fold cross-validation scheme using the scikit-learn library[135]. In this scheme, the TCGA data was first split into five parts (folds) of equal size and equal distributions of cancer types. Then, in each iteration of the cross-validation, one of the folds was used as test data for model evaluation, while the remaining four folds were further split into 80% training and 20% validation data used for model training and early stopping (with patience 5), respectively. For model training and evaluation in each iteration, the TCGA training, validation, and test data were scaled using the MinMaxScaler fitted to the respective GTEx pre-training data. Next, transfer learning for survival prediction was performed on the training data according to one of the transfer learning settings described above with early stopping based on the validation data. Finally, after transfer learning was completed, the trained model was evaluated on the test data.

6.2.4 Implementation

All transfer learning experiments were implemented in Python (release 3.10) and based on the Keras[36] and TensorFlow[1] machine learning libraries. For hyperparameter tuning, the Optuna framework[5] was used. Pre-training and transfer learning of all models and all evaluated settings was conducted on a Tesla V100-PCIE-32GB GPU using the NVIDIA CUDA platform (version 11.6).

6.3 Results

This section describes the results obtained in the two scenarios exploring the potential of transfer learning for cancer survival prediction. The first scenario investigated the transferability of knowledge learned for cancer survival prediction on the TCGA dataset to smaller, independent cancer datasets from different studies. In the second scenario, transfer learning was used to extract knowledge from tissue type classification and age prediction and explore the potential benefits of this knowledge for cancer survival prediction.

6.3.1 Survival Information Can Be Partially Transferred between Cancer Studies

As described in Section 6.2.2, we first explored transfer learning from a survival prediction model pre-trained on 25 cancer cohorts from TCGA to survival prediction on smaller, independent cancer datasets, including CPTAC-3, CDDP_EAGLE-1, and CGCI-BLGSP. To this end, we pre-trained a feed-forward neural network on 5,148 patients (training data) from

25 TCGA cohorts and during training applied early stopping with patience 5 based on the validation performance to avoid overfitting. On the TCGA test data (1,609 patients), the pre-trained model achieved a micro-average C-Index of 0.6595, which was calculated by computing the C-Indices for every cancer cohort in the test data separately and then computing the mean of these C-Indices weighted by the number of test patients from each cohort. On a per-cohort level, the survival prediction performance of the pre-trained model appears to be largely on par with the average performance of the XGBoost pan-cancer model described in Chapter 5 over 100 training replications with different training and test splits (Figure 6.1). However, because we only evaluated the pre-trained model on a single training and test split of the TCGA data, its performance on the test data may provide a less accurate and less robust estimate of model quality than the averaged results of the XGBoost models.

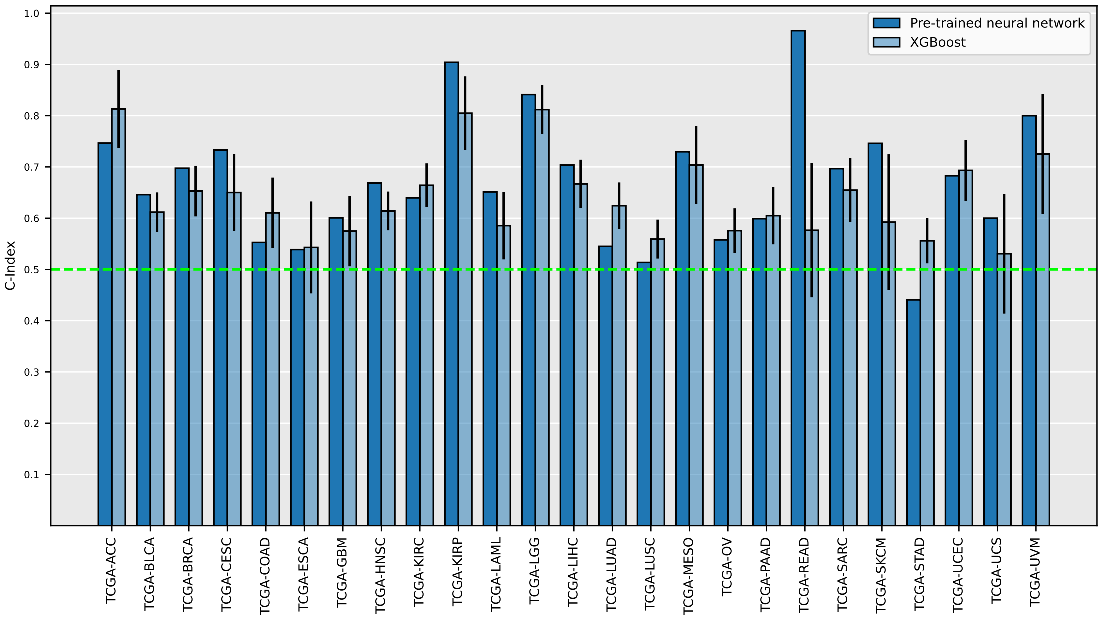


Figure 6.1: Performance of pre-trained survival prediction model. This figure shows the test performance (measured as C-Index) of the pre-trained survival prediction neural network (darker blue) on the 25 TCGA cancer cohorts. The performance for each cohort is compared with the average test survival prediction performance of the pan-cancer XGBoost method (lighter blue) described in Chapter 5, which was trained and evaluated on 100 different train-test splits. Error bars represent the standard deviation of C-Indices over the 100 replications of XGBoost.

After pre-training the survival prediction model on TCGA data, the weights and biases of the pre-trained neural network were transferred to predict survival on smaller, independent cancer datasets, including CPTAC-3 as a pan-cancer dataset and all cancer types contained in CPTAC-3 separately, the lung adenocarcinoma dataset from the CDDP_EAGLE-1 study and the CGCI-BLGSP mature B-cell lymphoma dataset. For each dataset, we compared the 5-fold cross-validated survival prediction performances from *pre-training*, where the pre-trained model was directly applied to the respective small cancer dataset without further fine-tuning of weights and biases, *fine-tuning*, where the transferred weights and biases of the pre-trained model were further fine-tuned on the small cancer dataset with a small learning rate (0.1 times the original learning rate), and training from *scratch*, where a randomly initialized neural network with the same model architecture and the same hyperparameters as the pre-trained model was trained without prior transfer of weights and biases.

For seven out of ten evaluated datasets (CPTAC-3 pan-cancer, CPTAC-3 Bronchus and lung – Adenomas and Adenocarcinomas, CPTAC-3 Bronchus and lung – Squamous Cell Neoplasms, CPTAC-3 Kidney – Adenomas and Adenocarcinomas, CPTAC-3 Other and ill-defined sites – Squamous Cell Neoplasms, CDDP_EAGLE-1 Bronchus and lung – Adenomas and Adenocarcinomas, and CGCI-BLGSP Mature B-Cell Lymphomas), applying the pre-trained survival prediction model directly to the respective dataset without further fine-tuning of weights and biases yielded the best average performance in terms of C-index (or micro-average C-index in case of the CPTAC-3 pan-cancer dataset) of all three evaluated model types (Table 6.1). For the remaining three datasets (CPTAC-3 Brain – Gliomas, CPTAC-3 Pancreas – Ductal and Lobular Neoplasms, and CPTAC-3 Uterus, NOS – Adenomas and Adenocarcinomas), the models trained from scratch without any transfer learning showed the best average performance in 5-fold cross-validation. Interestingly, fine-tuning the transferred weights and biases on the small datasets did not yield the best average survival prediction performance for any of the evaluated datasets. However, for six of the seven datasets where pre-training without any further fine-tuning yielded the best performance of all model types, the models with fine-tuning still outperformed the models that were trained from scratch, further confirming that for these datasets transfer learning provides an advantage over training from scratch.

6.3.2 Transfer of Knowledge from Auxiliary Tasks Can Improve Survival Prediction Performance

In addition to the first transfer learning scenario, where we explored the effect of transferring knowledge from one cancer survival dataset to different independent, smaller cancer survival datasets, we also investigated a second scenario for transfer learning (described in Section 6.2.3). In this second scenario, we pre-trained neural networks for different auxiliary tasks, including tissue type classification, age prediction, and joint tissue type and age prediction, on gene expression data from the GTEx project and tried to transfer the knowledge learned from these tasks to cancer survival prediction on a pan-cancer dataset from TCGA, which comprised gene expression and survival data from 25 different cancer types.

The hyperparameters of each of the pre-trained neural networks were optimized using the Optuna framework[5] with TPE. For the tissue type classification and age prediction models, the hyperparameters were optimized according to only one metric (Matthew's correlation in the case of tissue type classification and weighted Cohen's κ_w in the case of age predic-

Table 6.1: Transfer learning results on independent small cancer datasets. *Study* contains the source cancer study of each dataset, *Cancer type* describes the cancer type(s) contained in the dataset ('Pan-cancer (micro-averaged)' in case of the CPTAC-3 pan-cancer dataset), and *# Patients (uncensored)* contains the number of total and uncensored patients in each dataset. The three columns *Scratch*, *Pre-training*, and *Fine-tuning* contain the survival prediction performance of each dataset measured as mean C-Index (mean micro-average C-Index over the different cancer types for the CPTAC-3 pan-cancer dataset) over test folds of the 5-fold cross-validation ± standard error of the mean (SEM). The best average survival prediction performance for each dataset is marked in **bold**.

Study	Cancer type	# Patients (uncensored)	Scratch	Pre-training	Fine-tuning
CPTAC-3	Pan-cancer (micro-averaged)	763 (214)	0.5867 ± 0.0141	**0.6219 ± 0.0295**	0.6202 ± 0.0144
	Brain – Gliomas	71 (52)	**0.5869 ± 0.0549**	0.5848 ± 0.0350	0.5556 ± 0.0568
	Bronchus and lung – Adenomas and Adenocarcinomas	157 (26)	0.5999 ± 0.0654	**0.6917 ± 0.0223**	0.6437 ± 0.0731
	Bronchus and lung – Squamous Cell Neoplasms	84 (15)	0.5365 ± 0.1056	**0.6315 ± 0.0833**	0.5691 ± 0.1522
	Kidney – Adenomas and Adenocarcinomas	174 (27)	0.6026 ± 0.0534	**0.7442 ± 0.0410**	0.7380 ± 0.0185
	Pancreas – Ductal and Lobular Neoplasms	93 (67)	**0.6029 ± 0.0388**	0.5260 ± 0.0357	0.5638 ± 0.0259
	Uterus, NOS – Adenomas and Adenocarcinomas	96 (11)	**0.8188 ± 0.0698**	0.5885 ± 0.0766	0.6192 ± 0.0850
	Other and ill-defined sites – Squamous Cell Neoplasms	88 (16)	0.3870 ± 0.0415	**0.5670 ± 0.0532**	0.5572 ± 0.0277
CDDP_EAGLE-1	Bronchus and lung – Adenomas and Adenocarcinomas	44 (28)	0.5744 ± 0.0527	**0.6997 ± 0.0337**	0.6865 ± 0.0710
CGCI-BLGSP	Mature B-Cell Lymphomas	29 (29)	0.5667 ± 0.0596	**0.6200 ± 0.1052**	0.5533 ± 0.0629

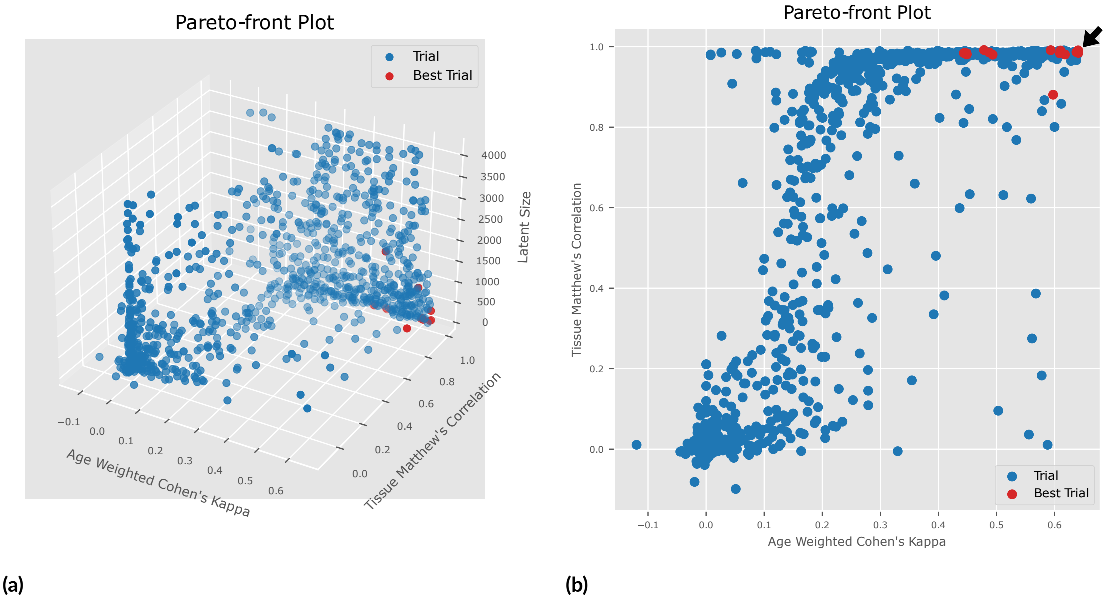


Figure 6.2: Results of hyperparameter optimization for joint tissue type and age prediction. Dots represent trials of the Optuna[5] hyperparameter optimization procedure and mark the achieved prediction performance on the GTEx validation data with respect to the optimized performance metrics. Each trial is associated with a distinct configuration of model hyperparameters. For the tissue type and age prediction multi-task model, three metrics, including the weighted Cohen's κ_w for age prediction, Matthew's correlation for tissue type classification, and the size of the last layer shared between both tasks (latent size), were jointly optimized. Pareto-efficient solutions, which are solutions where none of the optimization objectives can be further improved without compromising another[37], are marked in red. **(a)** Plot of the pareto front including all three optimized metrics. **(b)** Plot of the two Pareto-front dimensions representing tissue type and age prediction performance. The black arrow marks the trial whose hyperparameters were selected to train the tissue type and age prediction model that was then used for transfer learning for cancer survival prediction.

tion), resulting in a single hyperparameter combination that produced the best results on the validation data. In contrast, for the tissue type and age prediction multi-task model, hyperparameters needed to be optimized according to both metrics simultaneously. In addition to the two metrics for tissue type classification and age prediction, we also optimized the hyperparameters of the multi-task model to minimize the size of the last layer shared between tasks to prevent the model from just sharing one large, very general layer between tasks, while shifting most of the task-specific knowledge to the remaining, task-specific layers, whose weights and biases would not be used in transfer learning. When optimizing hyperparameters simultaneously, it is not always possible to find a solution (i.e., a set of hyperparameters) that is optimal according to all metrics (optimization objectives) at the same time. Instead, a solution is selected that lies on the Pareto front of the optimization problem, which is an image of the set of all Pareto optimal solutions of the optimization problem[37]. A solution is said to be Pareto optimal if there is no other solution that improves one of the optimization objectives without compromising another. For our tissue type and age prediction multi-task model, we selected a hyperparameter combination from the set of Pareto optimal solutions by manual inspection of the Pareto front (Figure 6.2) and pre-trained the tissue type and age prediction model used for transfer learning based on this hyperparameter combination.

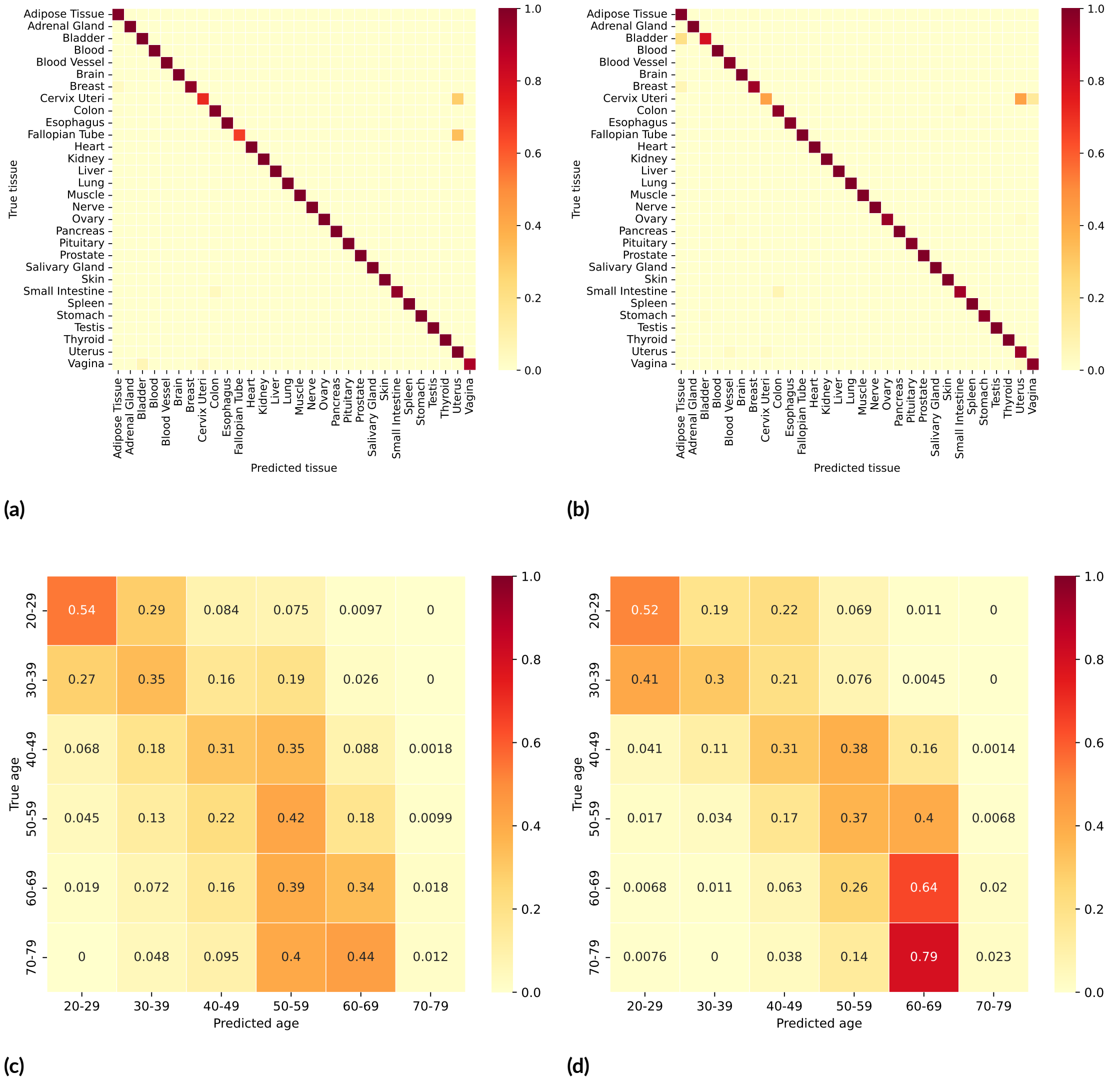


Figure 6.3: Tissue type and age prediction performance of pre-trained models. Shown are the row-normalized confusion matrices for **(a)** tissue type prediction of the model trained for tissue type classification **(c)** tissue type prediction of the model trained for tissue type and age prediction **(b)** age prediction of the model trained for age prediction, and **(d)** age prediction of the model trained for tissue type and age prediction. The rows of the confusion matrices represent ground truth tissue types and age ranges and columns represent predicted tissue types and age ranges, respectively.

Figure 6.3 visualizes the prediction performances of all pre-trained neural networks on GTEx test data that was withheld during training. Prediction performances are presented in the form of row-normalized confusion matrices of true (rows) versus predicted (columns) classes. Subfigure 6.3a compares the true and predicted tissue types of test samples predicted with the model pre-trained for tissue type prediction only, while subfigure 6.3b shows true and predicted tissue types of the model pre-trained for tissue type and age prediction simultaneously. Both models yield almost perfect classification accuracy (0.9956 for the tissue type classification model and 0.9883 for the tissue type and age prediction model) and Matthew's correlation scores of 0.9953 and 0.9874, respectively, on the test data. Donor age prediction from gene expression data, however, seems to be a more challenging task. Both models trained for this task show suboptimal prediction accuracy of 0.3707 for the model trained on age prediction only and 0.4492 for the multi-task model trained on both age prediction and tissue type prediction simultaneously. However, the 10-year age brackets predicted in this task are not independent from each other, but have an ordinal relationship. Therefore, considering only prediction accuracy to assess model performance may be somewhat misleading because it fails to reflect the ordinal relationship between classes and treats a false prediction that deviates from the true age of the tissue donor by only one age bracket in the same way as a false prediction that deviates from the true class by multiple age brackets, which is arguably a larger error. To circumvent this problem and account for the relationship between age brackets, we also evaluated the test performance based on the weighted Cohen's κ_w with quadratic weights, which has values in the range $[-1, 1]$, with 1 indicating a perfect prediction. This way, prediction errors were weighted according to the distance between the true class and the predicted class, providing a more complete picture of model quality than accuracy alone. The model trained for age prediction alone yielded a weighted Cohen's κ_w score of 0.5225 on the GTEx test data, while the model trained on both age and tissue type prediction had a weighted Cohen's κ_w score of 0.6706. Looking at the respective confusion matrices of the two models (Figure 6.3 c and d), it is noticeable that while many predictions do not lie on the diagonal of correctly predicted age brackets, there is a distinctive accumulation of predictions close to the diagonal, meaning that for many samples, the predicted age was not exactly correct, but at least close to the true age bracket.

Using the three pre-trained models for tissue type classification, age prediction, and joint tissue type and age prediction, we explored two different modes of transfer learning for each of the models and additionally compared the results achieved in these two modes with survival prediction models that had the same model architecture as the respective pre-trained model, but were trained on the TCGA data from scratch and without any knowledge transfer (*scratch* mode). In the first transfer learning mode (*transfer*), weights and biases of the first *n* layers of the network were initialized with the weights and biases of the respective pre-trained model. The remaining, randomly initialized, layers were then trained for survival prediction, while the transferred layers were frozen and not trained further. In the second transfer learning mode (*fine-tune*), on the other hand, the first *n* layers of the neural network were also initialized with the weights and biases of the respective pre-trained model, but instead of training only the remaining, randomly initialized layers, all layers were trained further for survival prediction, using a lower learning rate for the transferred layers than for the randomly initialized layers to prevent the neural network from 'forgetting' the pre-learned knowledge

Table 6.2: Transfer learning results of models pre-trained on tissue type and age prediction. All models were trained for cancer survival prediction on TCGA pan-cancer data and had the same model architecture as the pre-trained model of the respective *Pre-training task*. In the "Transfer" and "Fine-tune" modes (*Mode*), weights and biases of the first n neural network layers (*Transferred layers*) were initialized with values transferred from models pre-trained on GTEx data for either tissue type classification, age prediction, or joint tissue type and age prediction. In "Transfer" mode, transferred layers were frozen and only the remaining, randomly initialized layers were trained and in "Fine-tune" mode, transferred and randomly initialized layers were trained with different learning rates. In "Scratch" mode, all layers were randomly initialized (no transfer learning) and trained for survival prediction. The last column (*Micro-average C-Index (± SEM)*) displays the average survival prediction performance (measured as micro-average C-Index across cancer cohorts) ± standard error of the mean (SEM) of 5-fold cross-validation on the TCGA data. The best average survival prediction performance for each pre-training task is marked in **bold** and the overall best performance is additionally marked with †.

Pre-training task	Mode	Transferred layers	Micro-average C-Index (± SEM)
Tissue type classification	Transfer	3	0.6028 ± 0.0057
		2	0.6170 ± 0.0042
		1	0.6295 ± 0.0071
	Fine-tune	3	**0.6481 ± 0.0089**†
		2	0.6451 ± 0.0022
		1	0.6362 ± 0.0082
	Scratch	0	0.6295 ± 0.0087
Age prediction	Transfer	2	0.5482 ± 0.0068
		1	0.6011 ± 0.0061
	Fine-tune	2	0.6430 ± 0.0056
		1	**0.6457 ± 0.0006**
	Scratch	0	0.6326 ± 0.0046
Tissue type & age prediction	Transfer	3	0.5467 ± 0.0058
		2	0.5812 ± 0.0075
		1	0.6100 ± 0.0086
	Fine-tune	3	0.6287 ± 0.0054
		2	0.6282 ± 0.0050
		1	**0.6322 ± 0.0048**
	Scratch	0	0.6013 ± 0.0058

again by changing the transferred layers to much based on the survival data. For both transfer learning modes (*transfer* and *fine-tune*), we evaluated models with different numbers of transferred layers n with n between 1 and the number of hidden layers (or shared layers in the case of the tissue type and age prediction multi-task model).

Overall, a survival prediction model with three transferred layers that were pre-trained on the tissue classification task and were further trained for survival prediction in *fine-tune* mode showed the best mean micro-average C-index in a 5-fold cross-validation on the TCGA data (Table 6.2). The micro-average C-index was computed by calculating the C-Indices for every cancer cohort based on the test data of the respective iteration of the cross-validation separately and then computing the mean of these C-Indices weighted by the number of test patients from each cohort. Notably, also for the other two evaluated pre-training tasks of age prediction and joint tissue type and age prediction, a survival prediction model trained in *fine-tune* mode showed the best prediction performance, respectively, in terms of mean micro-average C-index, indicating that cancer survival prediction can benefit from transfer learning on auxiliary tasks such as tissue type classification and age prediction. However, while for the models pre-trained on tissue type classification, the fine-tuned model with the maximal number of transferred layers (3) yielded the best survival prediction performance, the best-performing models pre-trained on one of the other two tasks (age prediction or joint tissue type and age prediction) only used one transferred layer each, while all consecutive layers were randomly initialized. Moreover, the improved performance of the best-performing fine-tuned models of all three pre-training tasks over the respective models trained from scratch was observable for many, but not all 25 cohorts of the TCGA pan-cancer dataset (Supplementary Figures A.7, A.8, and A.9). This emphasizes the heterogeneity of the different cancer types and their different relationships to the pre-training tasks, which can have more or less relevance for patient survival in a certain cancer type. For example, for bladder urothelial carcinoma (TCGA-BLCA), the neural networks pre-trained for tissue type classification and fine-tuned for survival prediction on TCGA pan-cancer data performed worse than the corresponding model that was trained from scratch (Supplementary Figure A.7), while the fine-tuned models outperformed the scratch model for the same cancer type when the pre-training task was age prediction or joint tissue type and age prediction (Supplementary Figures A.8 and A.9, respectively). We speculate that the lack of benefit from transfer learning from tissue type classification for this cancer type might be caused by the composition of the pre-training data, which contained only 21 bladder tissue samples ($\sim$ 0.12% of all samples, cf. Supplementary Figure A.6a) and might thus not represent the molecular characteristics of bladder cancer comprehensively enough. On the other hand, bladder cancer is a cancer of old age (median age at diagnosis 73 and median age at death 79 in the U.S.[129]), which might explain why survival prediction in this cancer type benefited from pre-training on age prediction and joint age and tissue type prediction.

Models that were initialized with transferred layers that were not further fine-tuned for cancer survival prediction (*transfer* mode) showed an overall worse performance (measured as micro-average C-Index averaged over 5-fold cross-validation) than the fine-tuned models (*fine-tune* mode), which was also consistently worse than the model trained from scratch when more than one layer was transferred.

Overall, these results suggest that transfer learning from auxiliary tasks such as tissue type classification or age prediction can be beneficial for the training of cancer survival prediction models. However, while the pre-training tasks explored in this work are to some extent related to cancer survival, this relationship is not particularly close for all cancer types, such that the weights and biases learned for the tasks of tissue type classification or age prediction appear to require further fine-tuning on cancer survival data for the model to unfold its full predictive capacity for most cancer types.

6.4 Discussion

Transfer learning is a concept used in machine learning to leverage knowledge learned from one task to improve the prediction performance of a machine learning model on a different but related task (cf. 3.2.3). Here, we explored the potential of transfer learning for cancer survival prediction by pre-training neural networks on different tasks and then transferring the learned knowledge to predict patient survival. In contrast to the first part of our work, where we applied the tree ensemble learning method XGBoost[31] to predict survival for pan-cancer patients from TCGA, we switched to neural networks in this part of the work because neural networks allow for easier knowledge transfer between models than XGBoost, for which further training on data not seen during initial training is not straightforward because it has a fixed tree structure and split variables once trained. Neural networks, on the other hand, have weights (and often biases) associated with each of their layers, which can be trained on one dataset, and then either all or a subset of the trained layers can be transferred to a new dataset and optionally be fine-tuned for a new task. In this work, we investigated two different transfer learning scenarios for cancer survival prediction.

In the first scenario, we pre-trained a neural network for cancer survival prediction on gene expression data from TCGA and transferred the learned knowledge to different independent, smaller single- and pan-cancer datasets. Since in this scenario, both the source and the target task were to predict cancer survival from gene expression data, we transferred all layers from the pre-trained neural network to the survival prediction models for the target datasets and found that for seven of the ten evaluated target cancer datasets, applying the pre-trained model directly to the target data without fine-tuning the weights and biases of the respective model further yielded a better prediction performance than fine-tuning the model on the target dataset or training a model from scratch on the target dataset only. For six out of the seven datasets in which the pre-trained model yielded the best performance, the fine-tuned model still performed better than the corresponding model trained from scratch, but worse than the pre-trained model without fine-tuning. We speculate that a reason for this outcome might be that the evaluated target datasets are relatively small with only 28 to 174 patients per cancer type (with 11–52 uncensored patients for the cancer types in which the pre-trained model showed the best performance) and further fine-tuning the pre-trained model on these small datasets can easily lead to overfitting on the training data, thus deteriorating the test performance. On the other hand, training from scratch without the involvement of any transfer learning showed the best performance for only three of the ten evaluated datasets, suggesting that pre-training survival prediction models on larger (pan-)cancer datasets can be beneficial for survival prediction on smaller datasets and that the learned knowledge can indeed be successfully transferred to other cancer studies in many cases. Nevertheless, we also note that due to the small size of the explored target datasets, the evaluated patients might not always be entirely representative of the respective cancer type and the observed results might not be very robust in some cases. Hence, we think that the results obtained in this transfer learning scenario (Table 6.1) should be interpreted with caution and are an indication rather than proof that survival prediction on small cancer datasets can benefit from transfer learning.

In the second transfer learning scenario, we investigated the effects of transfer learning when knowledge was transferred between different but related tasks. More specifically, in this scenario, we pre-trained three different neural networks on gene expression data from the GTEx project, which is not associated with cancer and comprises 17,382 samples and corresponding meta-data collected from multiple tissue types of 948 deceased donors, and transferred the learned knowledge to predict survival for pan-cancer patients from TCGA. The pre-training tasks of the three neural networks were the prediction of donor age, the classification of tissue types, or joint age prediction and tissue type classification, respectively. Both age prediction and tissue type classification are to some extent related to cancer survival prediction and are thus promising pre-training tasks. On the one hand, aging and cancer are tightly connected with each other and share some common biological mechanisms[17]. For instance, the incidence of many cancer types is positively correlated with age[50,130] and cancer survival is higher in younger patients than in older patients[141]. Additionally, cancer and aging share some key biological characteristics, including genomic instability, epigenetic alterations, chronic inflammation, and dysbiosis, which is characterized by the disruption of bacteria-host communication of the gut microbiome and can contribute to aging and aging-associated diseases like cancer[121]. On the other hand, there are tissue- and cell-type-specific differences in tumorigenesis and in the organization of oncogenic signaling pathways, such that the signaling output of oncogenic drivers may vary considerably between tissue types[151], suggesting tissue type classification as another suitable pre-training task. In addition to the two neural networks trained for age prediction and tissue type classification, respectively, we also trained a third, multi-task neural network that combined both pre-training tasks and thus learned both types of knowledge simultaneously, which we thought could potentially be even more beneficial for transfer learning for cancer survival prediction than learning either task alone. For all three pre-training tasks, transferring one or multiple layers of the pre-trained neural network to a new neural network and then further fine-tuning all layers of the new neural network for survival prediction on pan-cancer TCGA data resulted in improved performance over training a neural network with the same model architecture from scratch, while without further fine-tuning the weights and biases of the transferred layers, the model trained from scratch outperformed the model with transferred layers in most cases.

At first glance, these results seem to conflict with the findings from the first transfer learning scenario, where knowledge was transferred between different cancer survival datasets and pre-training without further fine-tuning on the target dataset produced the best results in most cases. However, we do not believe that this is actually a contradiction, because the two transfer learning scenarios are different from each other in many aspects. For instance, in the first scenario, where knowledge learned from survival prediction on one cancer dataset was transferred to other cancer datasets from different studies, the evaluated target datasets were relatively small, making fine-tuning on the target datasets more difficult because models trained with small sample sizes can easily overfit on the training data, while in the second transfer learning scenario, where knowledge from auxiliary tasks such as tissue type classification and age prediction was transferred to cancer survival prediction, the target dataset was substantially larger, allowing for successful fine-tuning. Furthermore, while in the first scenario, knowledge was transferred for the same task of predicting cancer survival, making further fine-tuning less relevant, in the second scenario, knowledge was transferred between different prediction tasks, making further fine-tuning on the target task of cancer survival prediction more important.

In summary, we have investigated the effects of transfer learning on cancer survival prediction and have found that cancer survival prediction models can benefit from knowledge transfer, both between domains with datasets from different cancer studies and the same task of predicting cancer survival, but also between different tasks such as tissue type and age prediction and cancer survival prediction. Depending on the similarity of the source and target tasks (i.e., same task vs. different tasks), but also on the size and other characteristics (e.g., the cancer type) of the target data, the effects of transfer learning were different. For example, we observed positive transfer with improved prediction performance between TCGA cancer survival prediction and cancer survival prediction on seven out of ten evaluated datasets (Table 6.1) in the first transfer learning scenario and between all three pre-trained models and TCGA pan-cancer survival prediction in terms of micro-average C-Index in the second transfer learning scenario (Table 6.2), but negative transfer with reduced prediction performance for the remaining three datasets of the first scenario (Table 6.1) and for some individual cancer types and pre-training tasks (e.g., bladder urothelial carcinoma with tissue type classification pre-training; Supplementary Figure A.7) in the second scenario.

6.4.1 Limitations

Although our transfer learning experiments showed some promising results, there are a few limitations to transfer learning for cancer survival prediction. In contrast to some popular pre-trained models such as ResNet50[80], VGG16[160], or Xception[35], which are commonly used in transfer learning for image classification and are typically pre-trained on extremely large datasets with more than a million samples, our pre-training datasets were much smaller, with sample sizes in the thousands (pre-training on TCGA for cancer survival prediction) or tens of thousands (pre-training on GTEx for tissue type and age prediction). Therefore, the potential of the investigated pre-training tasks for transfer learning on cancer survival prediction might not have been fully exploited and models pre-trained on larger datasets might yield even better performance on the pre-training tasks and thus provide more comprehensive knowledge that can be transferred. In addition to the size of the source dataset used for pre-training, the size of the target dataset can be another limiting factor for transfer learning. In the first transfer learning scenario, where knowledge was transferred between datasets from different studies, but on the same task of cancer survival prediction, the target datasets had very small sample sizes and the few test samples for each dataset might not have been entirely representative of the respective cancer type, such that the robustness of the obtained results might be somewhat limited. This problem of limited robustness of performance estimation due to small sample sizes can be especially pronounced in cancer survival prediction with Cox loss and C-Index as a performance metric, where prediction performance is judged by comparing predicted risks and true survival times between samples and for small sample sizes, outliers can substantially influence the performance estimate. Lastly, the relationship between the source and target domains can be a limiting factor for the success of transfer learning. In the first transfer learning scenario, where knowledge was transferred between different cancer datasets, but for the same task of cancer survival prediction, this relationship was naturally fairly close. However, in the second transfer learning scenario, knowledge was transferred between more distinct tasks. Although the two pre-training tasks of tissue type classification and age prediction are related to the task of cancer survival prediction in that there are tissue- and cell-type-specific differences in tumorigenesis and in the organization of oncogenic signaling pathways[151], and aging shares some of its hallmarks with cancer[121] and is correlated with cancer incidence[50,130], this relationship is not extremely close, limiting the amount of knowledge that can be effectively transferred between the tasks. Nevertheless, the observed improvement in prediction performance through both pre-training tasks suggests that their relationship to cancer survival was still close enough for transfer learning to be successful.

7

Outlook and Conclusion

In this chapter, the results of this dissertation are summarized. Furthermore, an outlook is given on how challenges of cancer survival prediction on currently available cancer datasets could be resolved and how cancer survival prediction could be further advanced in the future if more comprehensive data becomes available. Finally, a brief conclusion is given.

7.1 Summary of the Work

Cancer is one of the leading causes of death worldwide[55,57] and the second leading cause of death in Germany[163]. Reducing cancer mortality and improving patient survival are the primary goals of cancer therapy. However, therapy choices are usually influenced by patient prognosis, thus making cancer survival prediction an important computational task, which can help to estimate prognosis and quantify individual risk.

The first goal of this dissertation was to develop a machine learning method for cancer survival prediction based on molecular patient data and to derive the biological plausibility of this method. Furthermore, we tried to answer the question of whether cancer survival prediction can be improved by transferring knowledge from machine learning models that were trained on different datasets and tasks.

To achieve the first objective, we developed a survival prediction method that applied XG-Boost tree ensemble learning to gene expression data of patients from 25 different cancer types from TCGA. We investigated two versions of this survival prediction approach, a single-cohort version in which we trained separate survival prediction models for each cancer type, and a pan-cancer version in which we combined patients from all 25 cancer types into one pan-cancer training dataset to overcome the small sample sizes as an observable shortcoming of the single-cohort approach and to enable the identification of cross-cohort survival features. Indeed, pan-cancer training showed improved performance over single-cohort training, suggesting that biological mechanisms affecting survival can be shared across different cancer types and the machine learning model could benefit from the increased sample size of the pan-cancer dataset. In addition to gene expression, we also evaluated our pan-cancer survival prediction approach on other molecular data types, including mutation, copy number variation, and protein expression data. The results indicated that gene expression was the most informative data modality, consistent with previous findings on other biomedical prediction tasks[41,176], closely followed by protein expression as the modality that is most directly related to the phenotype according to the central dogma of molecular biology. The biological plausibility of the gene expression-based approach was investigated by applying network propagation on gene weights derived from the pan-cancer XGBoost survival prediction method. This way, we inferred a pan-cancer survival network and further analyzed it with respect to biological pathways and mechanisms, revealing a strong association with the tumor microenvironment, which has been linked to several molecular processes affecting cancer survival.

The second objective of this dissertation was to explore whether cancer survival prediction can be improved through transfer learning, where knowledge learned from training a machine learning model on a source domain is transferred to a prediction task on a target domain. For this objective, we decided to switch machine learning frameworks from XGBoost to neural networks. The reason for this change was that, to the best of our knowledge, there is no method implementing transfer learning for XGBoost that is suited for survival prediction. On the other hand, neural networks are well-suited for transfer learning due to their multi-layered architecture. To understand the effect of transfer learning on cancer survival prediction, we investigated two different transfer learning scenarios based on neural networks. In the first scenario, we pre-trained a neural network on pan-cancer gene expression data from TCGA and transferred the learned weights and biases to predict cancer survival on smaller, independent datasets. Applying the pre-trained neural network directly to the respective target dataset yielded the best performance for seven out of ten evaluated small cancer datasets. Conversely, training from scratch without any transfer learning performed better for only three of the small cancer datasets, confirming that pre-training can have a positive effect on cancer survival prediction. In the second transfer learning scenario, we took the transfer of knowledge a step further and investigated whether survival prediction performance can also benefit from knowledge learned in different but related prediction tasks. We pre-trained three different neural networks on gene expression data from the GTEx project, which contains samples from multiple tissues of deceased donors. More specifically, we trained one of the neural networks on the task of tissue type classification, one on age prediction, and one on the dual task of simultaneous tissue type classification and age prediction. Our results showed that knowledge from all three models could be successfully transferred to cancer survival prediction, with neural networks initialized with pre-trained weights and biases and further fine-tuned for cancer survival prediction consistently performing the best.

7.2 Outlook

Both aspects of cancer survival prediction investigated in this dissertation showed encouraging results. However, there are still some unresolved challenges regarding cancer survival prediction and room for further advancement.

First and foremost, machine learning approaches like the ones applied in this work to predict cancer survival rely on abundant data to learn from. However, currently available cancer survival datasets like the TCGA dataset have limited numbers of patients (ranging from less than 100 to ∼1,000 patients per cancer type and less than 10,000 patients in total). We suspect that this relatively small number of patients available for most cancer types is a limiting factor for model performance and that our proposed survival prediction approaches would benefit from more training data. This hypothesis is supported by our findings from survival prediction with XGBoost, where we observed that the increased number of training samples in the pan-cancer approach as compared to single-cohort training could improve survival prediction performance for most cancer types (Figure 5.6) and pan-cancer prediction performance generally deteriorated when model training was conducted on randomly sampled subsets of the data (Supplementary Figure A.2). Additionally, a larger number of training samples could mitigate the "curse of dimensionality"[15,111], a phenomenon commonly encountered in machine learning and often responsible for overfitting caused by a surplus of features over samples (e.g., <10,000 patients, but ∼60,000 gene expression values per patient in TCGA). Therefore, if either the TCGA dataset is expanded with additional patients or larger similar datasets become available in the future, it would be worthwhile to re-evaluate our proposed cancer survival prediction approaches incorporating this additional data.

Undoubtedly, cancer treatment has a major effect on patient survival. However, treatment information is incomplete for all 25 analyzed TCGA cancer cohorts and is available for only a small proportion of patients in many cohorts (Supplementary Figure A.1). Overall, treatment information (for either drug treatment, radiation treatment, or both) is available for only 48.6% of TCGA patients (3,911 of 8,045 patients with available survival and gene expression data for GDC data release v.32.0). This lack of comprehensive treatment data and the diversity of treatment regimens (e.g. in terms of radiation therapy administration, administered drugs or drug combinations, drug or radiation doses, and frequency and duration of drug or radiation therapy) makes it extremely difficult to include treatment information into any survival prediction model that is trained on this data. In fact, if treatment information was to be considered in the survival prediction model, one would have to discard more than half of the cancer patients due to lack of this information. However, less training data would likely result in degraded model performance, as discussed above and observed for example in Supplementary Figure A.2. In addition, the diversity of treatment regimens and lack of drug naming conventions (some drugs are named by their molecule names, while others are named by their commercial names, and additionally some of the drug names in the TCGA data contain misspellings[127]) would make it extremely difficult for any machine learning model to extract meaningful information from the treatment data, given the relatively small number of available patients, and would require elaborate manual feature engineering to standardize drug names and make treatments comparable across patients. Taking all of this into consideration, we believe that more complete and consistent treatment data is needed and has the potential to further improve cancer survival prediction. For future cancer studies, we suggest that standardized treatment information should be routinely recorded and made available in addition to clinical and molecular data. If such data becomes available to a greater extent in the future, it would also be worth exploring the incorporation of auxiliary knowledge on drug sensitivity, for example learned from abundant drug response datasets such as GDSC or CCLE, in addition to the treatment information. This could enable the machine learning method to better model the effects of different drugs on cancer survival, which would likely further improve prediction performance.

In addition, competing risks can pose a challenge to cancer survival prediction. Competing risks are death events that preclude the event of interest[10], which is death from cancer in our case. Cancer studies such as TCGA often record the overall survival of cancer patients, meaning the time from entry of the patient into the study until their death. However, especially for older patients, it is often questionable if the recorded death is due to the diagnosed cancer or due to a competing risk such as another co-morbidity or simply old age. Nevertheless, information on death causes is often not recorded. When a machine learning model is now trained for survival prediction with overall survival as the target variable, the training procedure cannot distinguish between causes of death and may, for example, penalize a low predicted risk for a patient who does not have an aggressive cancer but has died of another cause unrelated to the cancer in the same way as an incorrect prediction, where a patient with an aggressive cancer is predicted to have a low risk or a patient with a less aggressive cancer has a high predicted risk. This, in turn, makes it more difficult for the model to learn patterns that are truly related to cancer survival and can degrade the overall prediction performance of the trained model, as we also found when we compared the performance of our XGBoost pan-cancer survival prediction model including all recorded survival times with a model where we considered dead patients with recorded "tumor free" tumor status as censored instead of dead (Figure 5.22). In this analysis, we observed that regarding tumor-free patients as censored significantly improved prediction performance for 13 of the 25 TCGA cohorts. However, a "tumor free" tumor status does not guarantee that a patient has died from a competing risk rather than from cancer, since the cancer may have recurred at a later time. Thus, considering the survival times of all tumor-free patients as censored is not a satisfactory solution to the problem of competing risks and may also result in the loss of valuable survival information. Instead, if in the future the cause of death would be recorded in addition to the overall survival time for all or most deceased patients, we think that explicitly modeling these competing risks in the survival prediction model could reduce the confounding effect of competing risks and would be beneficial for the prediction performance.

All three challenges described above are related to different aspects of the training data that is used for cancer survival prediction. Therefore, they are not easily resolved by improving the machine learning methods used for survival prediction, but could be overcome through more comprehensive and complete data, which might become available in the future.

Lastly, as briefly mentioned in Chapter 6, the effect and success of transfer learning depends at least partly on the similarity between the source and target domains. Therefore, we suggest that in future works, in addition to tissue type classification and age prediction, other pre-training tasks and datasets that are potentially even more closely related to cancer survival prediction could be evaluated for transfer learning.

7.3 Conclusion

To summarize, this dissertation has investigated two different aspects of cancer survival prediction, with the first part focusing on biological plausibility with respect to the underlying molecular mechanisms, and the second part highlighting avenues for further improvement of cancer survival prediction through transfer learning. In the first part, we have introduced a machine learning approach for pan-cancer survival prediction that combines XGBoost tree ensemble learning with network propagation to derive a pan-cancer survival network. This pan-cancer survival network is significantly enriched for the TME, confirming the biological plausibility of our approach and highlighting the important role of the TME in cancer prognosis. In addition, we have investigated whether transfer learning can improve the performance of survival prediction neural networks and found that cancer survival prediction can indeed benefit from the transfer of knowledge, not only between datasets from different cancer studies, but also from pre-training on different auxiliary tasks such as tissue type or age prediction. However, we have also observed that the beneficial effect of transfer learning may depend on the size and characteristics of the target data, as well as the similarity between the source and target tasks.

A

Supplementary Figures

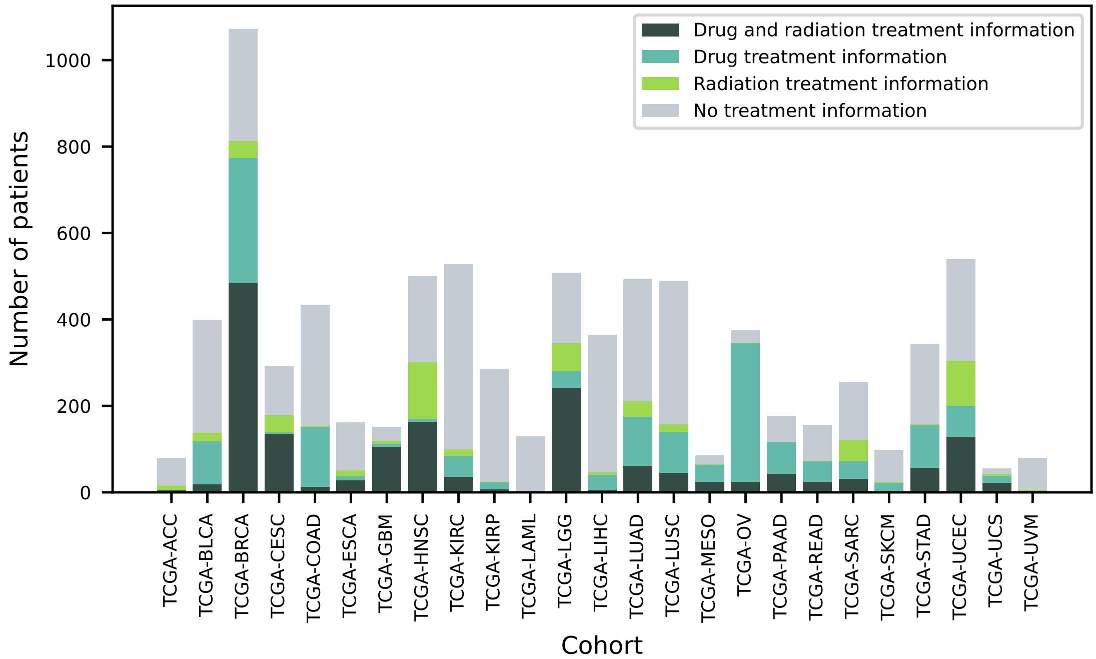


Figure A.1: TCGA patient treatment data. Number of TCGA patients per cancer cohort with available drug and radiation treatment information, only drug treatment information, only radiation treatment information, or no treatment information. Only patients with available survival and gene expression data for GDC data release v.32.0 were considered.

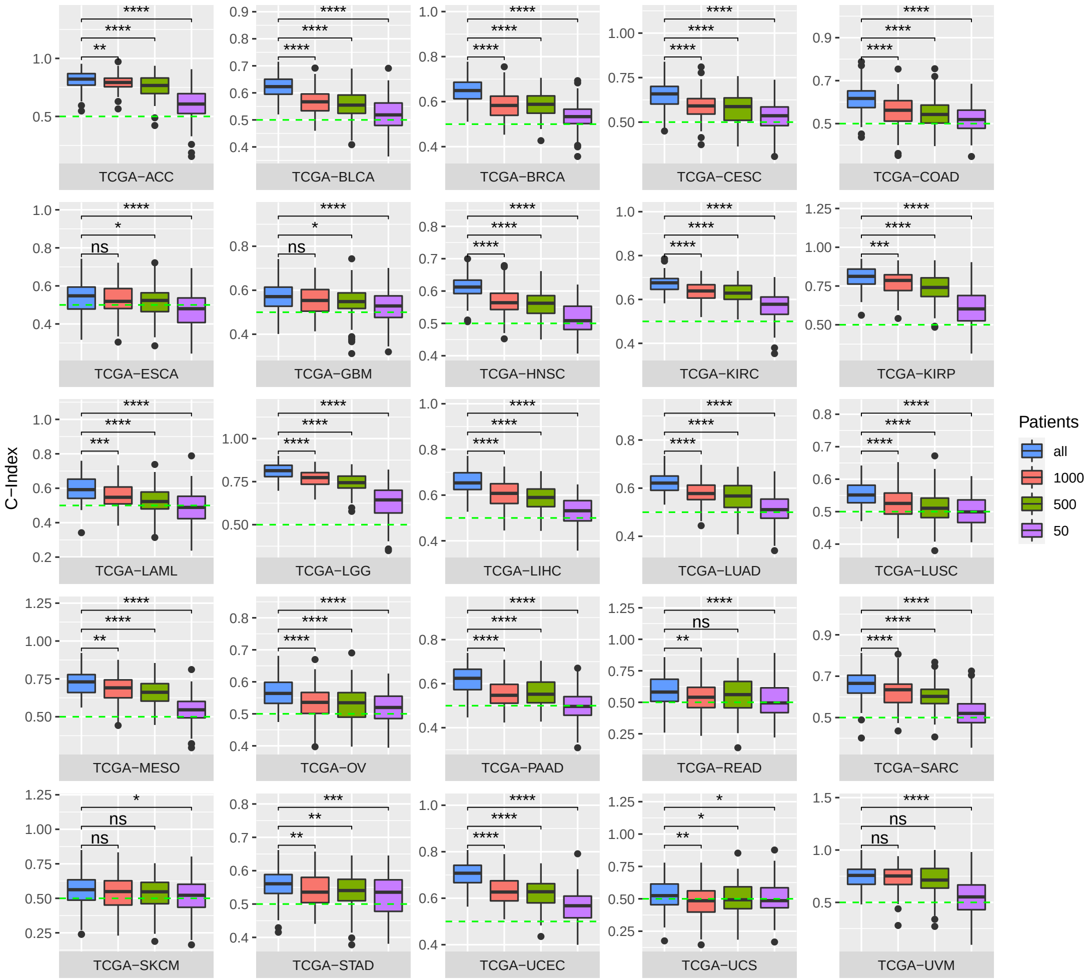
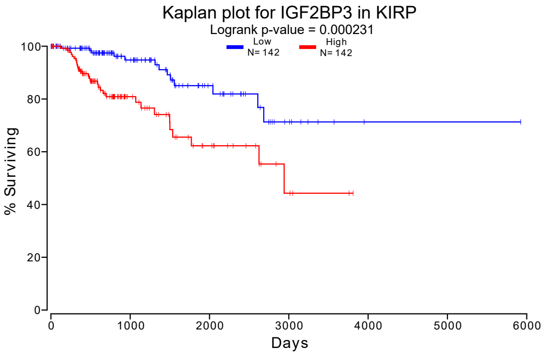
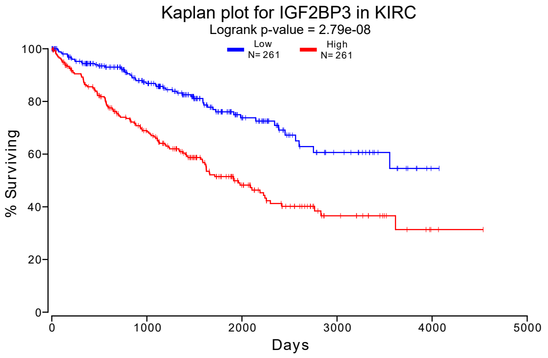


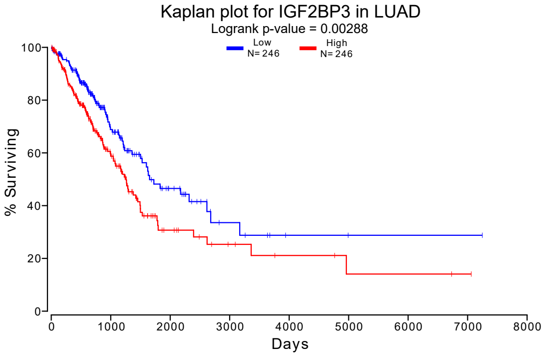
Figure A.2: Pan-cancer prediction performance on random patient subsets. The results obtained from 100 replications of pan-cancer model training on FPKM-normalized gene expression data are compared between training on all pan-cancer patients (N = 6,419) contained in the training data of the respective replication (blue) and training on patient subsets of different sizes. For training on the patient subsets, in each replication a pre-defined number of patients (either 1,000, 500, or 50) was randomly selected from the training data of the respective replication, where each subset contained approximately the same number of patients from each of the 25 TCGA cohorts. This random patient subsampling was performed before the feature selection step, such that feature selection as well as hyperparameter tuning and the training of the final survival prediction model were performed on this patient subset only. Model evaluation was then done on all patients belonging to the test data of the respective replication and no subsampling was performed. Performance is depicted by C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $* : p \leq 0.05$, $** : p \leq 0.01$, $*** : p \leq 0.001$, $**** : p \leq 0.0001$. This figure was published in [170].


Kaplan plot for IGF2BP3 in KIRP
Logrank p-value = 0.000231
Low N= 142
High N= 142
% Surviving
0
20
40
60
80
100
Days
0
1000
2000
3000
4000
5000
6000

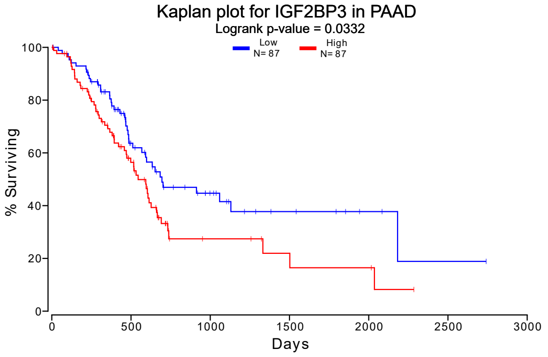
(a)


Kaplan plot for IGF2BP3 in KIRC
Logrank p-value = 2.79e-08
Low N= 261
High N= 261
% Surviving
0
20
40
60
80
100
Days
0
1000
2000
3000
4000
5000

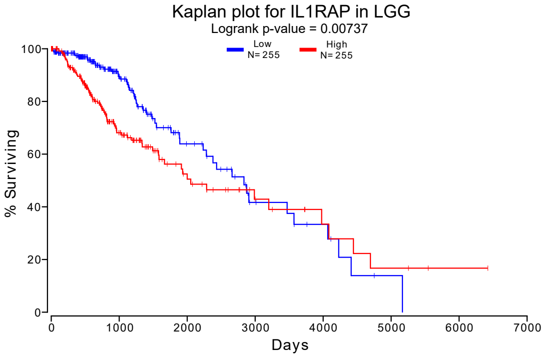
(b)


Kaplan plot for IGF2BP3 in LUAD
Logrank p-value = 0.00288
Low N= 246
High N= 246
% Surviving
0
20
40
60
80
100
Days
0
1000
2000
3000
4000
5000
6000
7000
8000

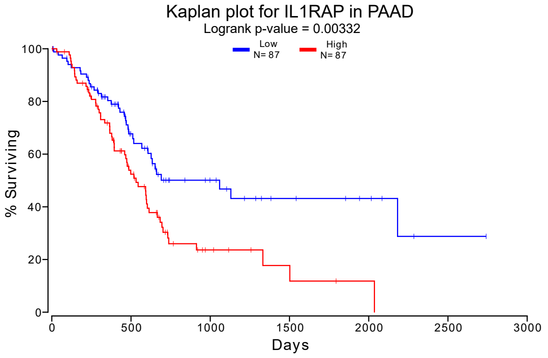
(c)


Kaplan plot for IGF2BP3 in PAAD
Logrank p-value = 0.0332
Low N= 87
High N= 87
% Surviving
0
20
40
60
80
100
Days
0
500
1000
1500
2000
2500
3000

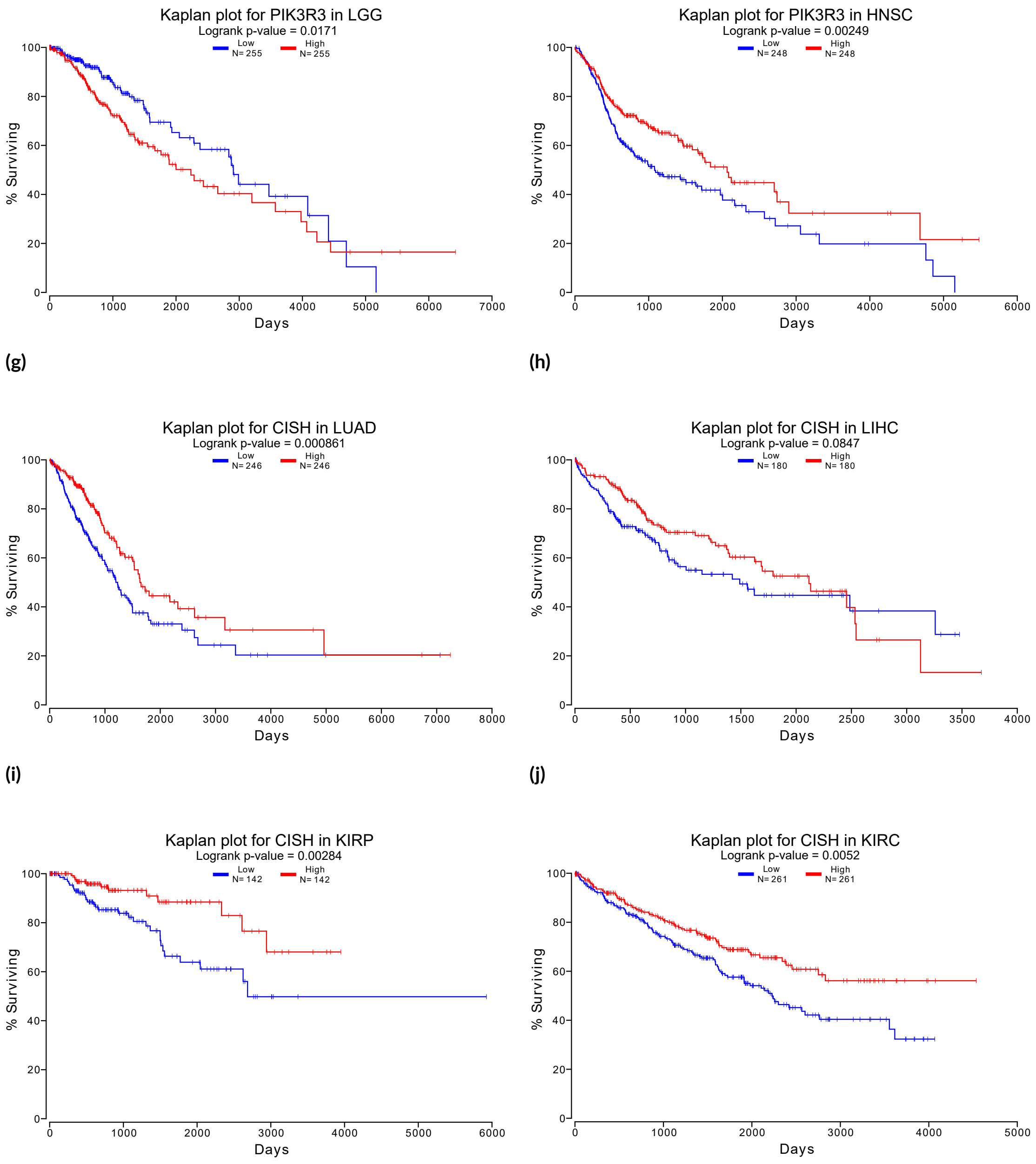
(d)


Kaplan plot for IL1RAP in LGG
Logrank p-value = 0.00737
Low N= 255
High N= 255
% Surviving
0
20
40
60
80
100
Days
0
1000
2000
3000
4000
5000
6000
7000

(e)


Kaplan plot for IL1RAP in PAAD
Logrank p-value = 0.00332
Low N= 87
High N= 87
% Surviving
0
20
40
60
80
100
Days
0
500
1000
1500
2000
2500
3000

(f)



(g)

(h)

(i)

(j)

(k)

(l)

Figure A.3: Additional Kaplan-Meier plots for *IGF2BP3*, *IL1RAP*, *PIK3R3*, and *CISH*. The Kaplan-Meier plots shown here were obtained from OncoLnc[6] and correspond to the four (KIRP, KIRC, LUAD, and PAAD), two (LGG and PAAD), two (LGG and HNSC), and four (LUAD, LIHC, KIRP, and KIRC) additional cohorts that were not shown in Figure 5.10, but also show significant survival performance ($FDR < 0.05$ in Cox regression) in the OncoLnc analyses for *IGF2BP3*, *IL1RAP*, *PIK3R3*, and *CISH*, respectively. For grouping the patients into two groups the 50th percentile of gene expression was selected as a cutoff in all cases. This figure was published in[170].

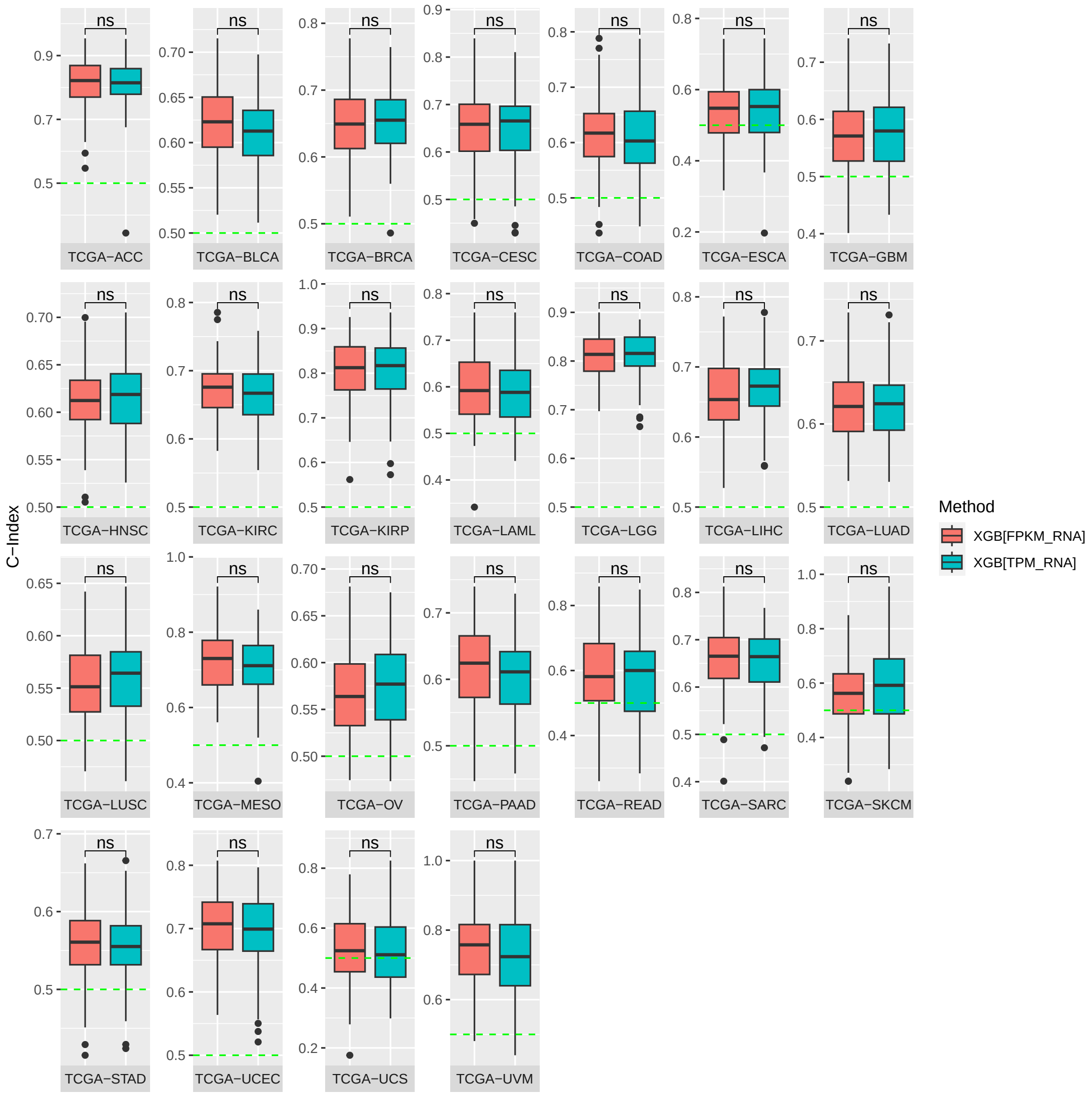


Figure A.4: FPKM vs. TPM survival prediction performance. Comparison of the prediction performance of the pan-cancer XGBoost method trained on FPKM-normalized gene expression data (XGB[FPKM_RNA]) and TPM-normalized gene expression data (XGB[TPM_RNA]). Performance is depicted by C-Index boxplots over 100 replications of model training. Mean C-Indices were compared with Wilcoxon's unpaired rank-sum test and significance levels are defined as ns : $p > 0.05$, $*: p \leq 0.05$, $** : p \leq 0.01$, $*** : p \leq 0.001$, $**** : p \leq 0.0001$.

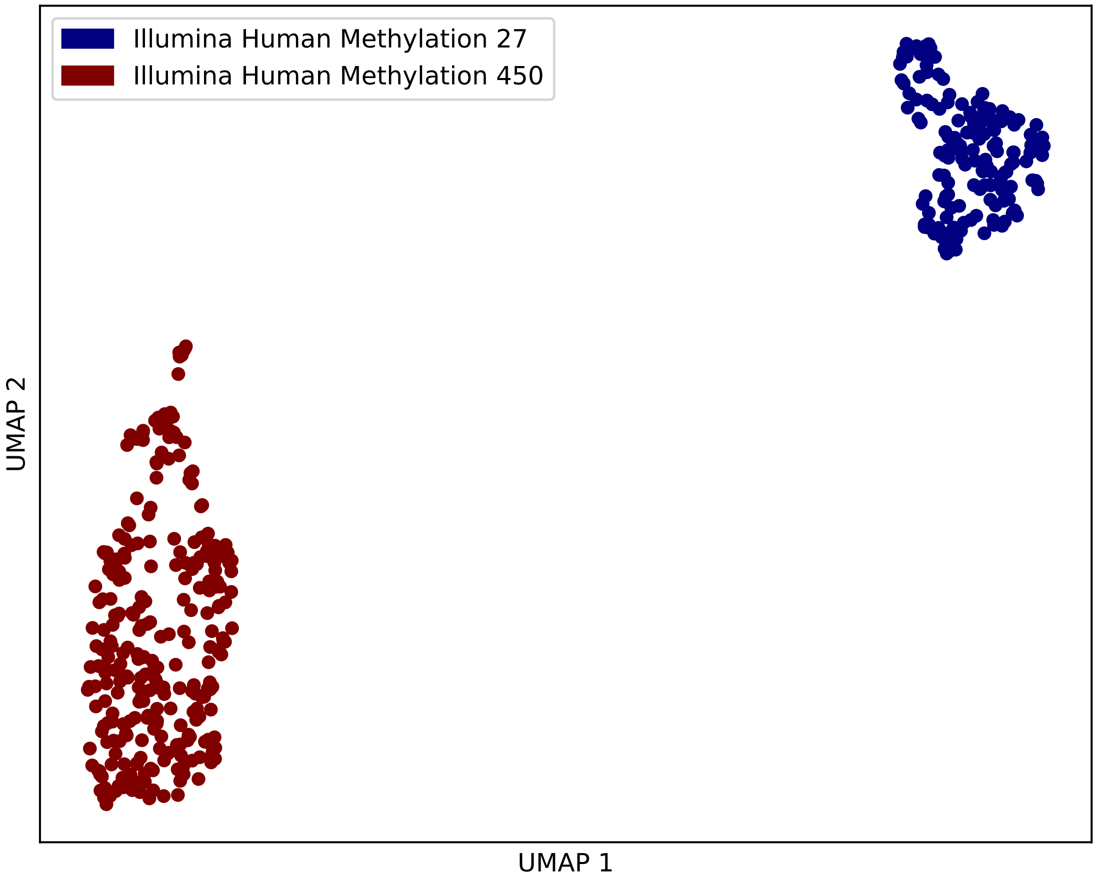
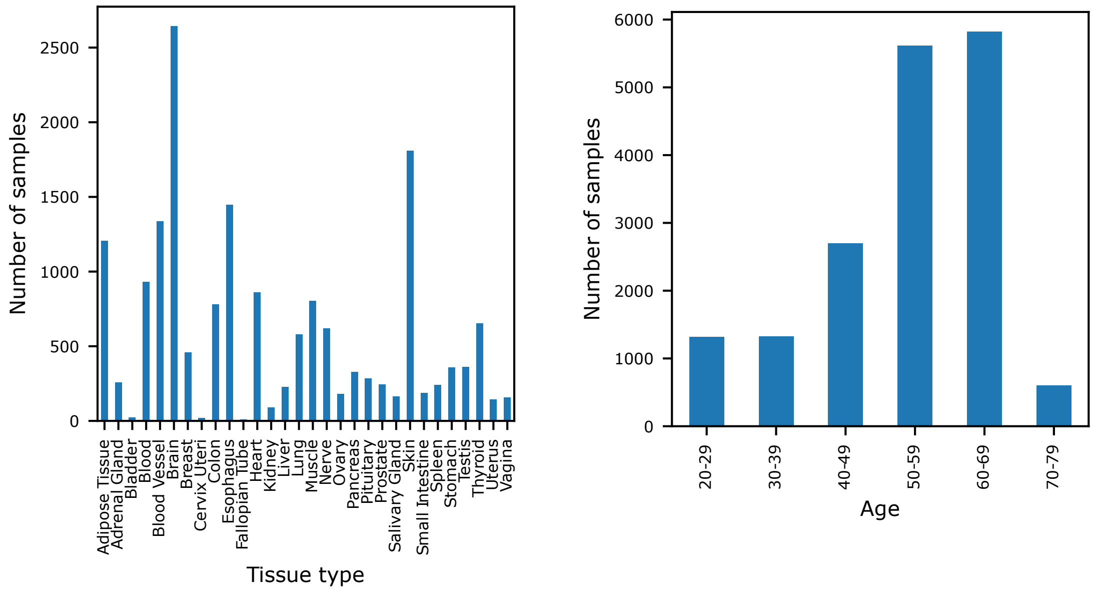


Figure A.5: Batch effects in methylation data. UMAP of methylation beta values for TCGA-COAD. Blue marks show samples measured by Human Methylation 27 array and red marks show samples measured by HumanMethylation 450 array. For generating the UMAP, only methylation sites measured by both arrays were considered and methylation sites with missing values were dropped. Methylation beta values were standardized to zero mean and unit variance, and principal component analysis (PCA) was performed to reduce the dimensionality to 50 dimensions before applying UMAP.



(a)

(b)

Figure A.6: GTEx tissue type and age distributions. **(a)** Distribution of tissue types in the GTEx dataset. **(b)** Distribution of 10-year age brackets in the GTEx dataset.

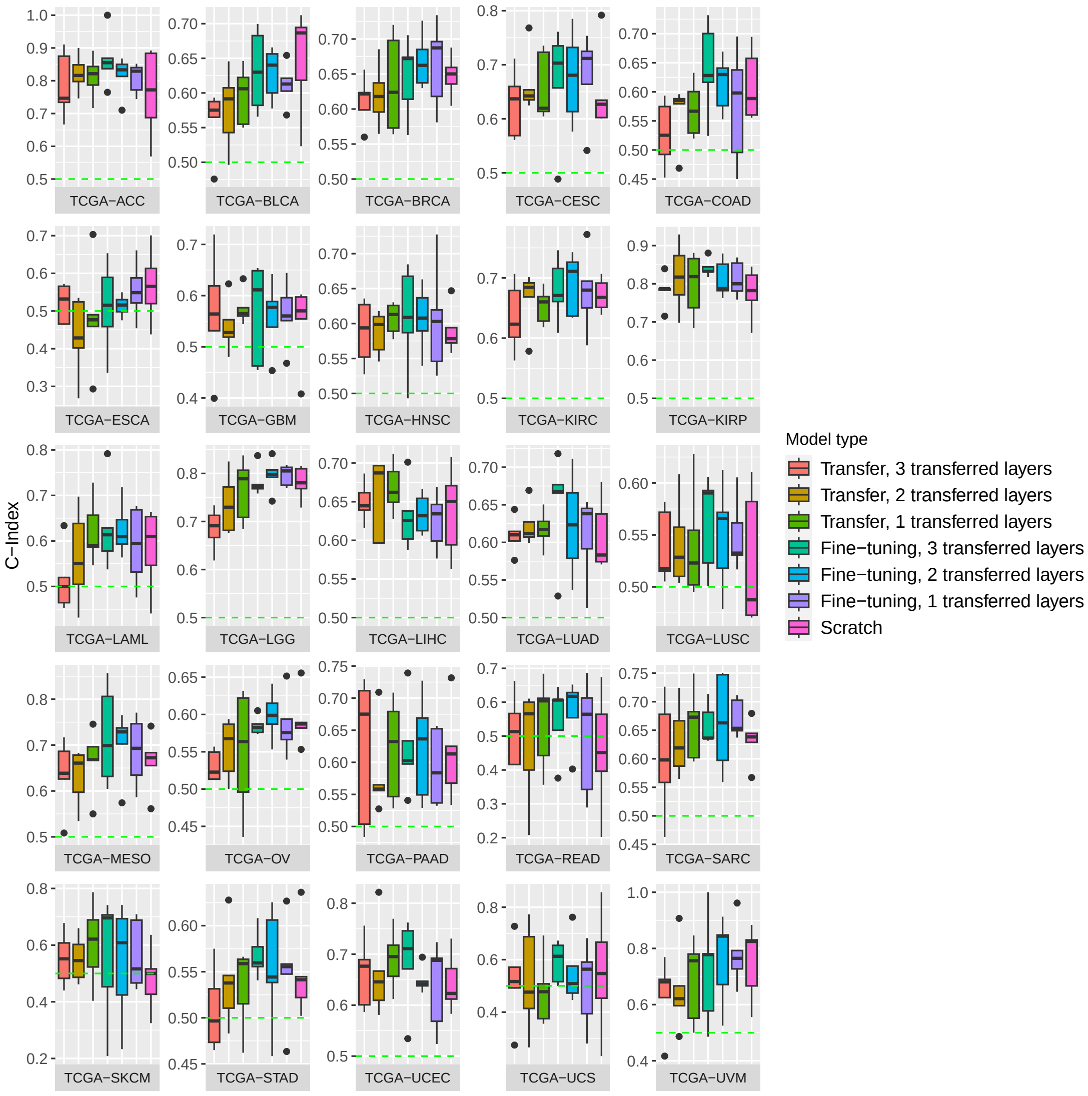


Figure A.7: Survival prediction performance of the neural network pre-trained for tissue type classification. Performance is depicted by C-Index boxplots for 5-fold cross-validation on TCGA.

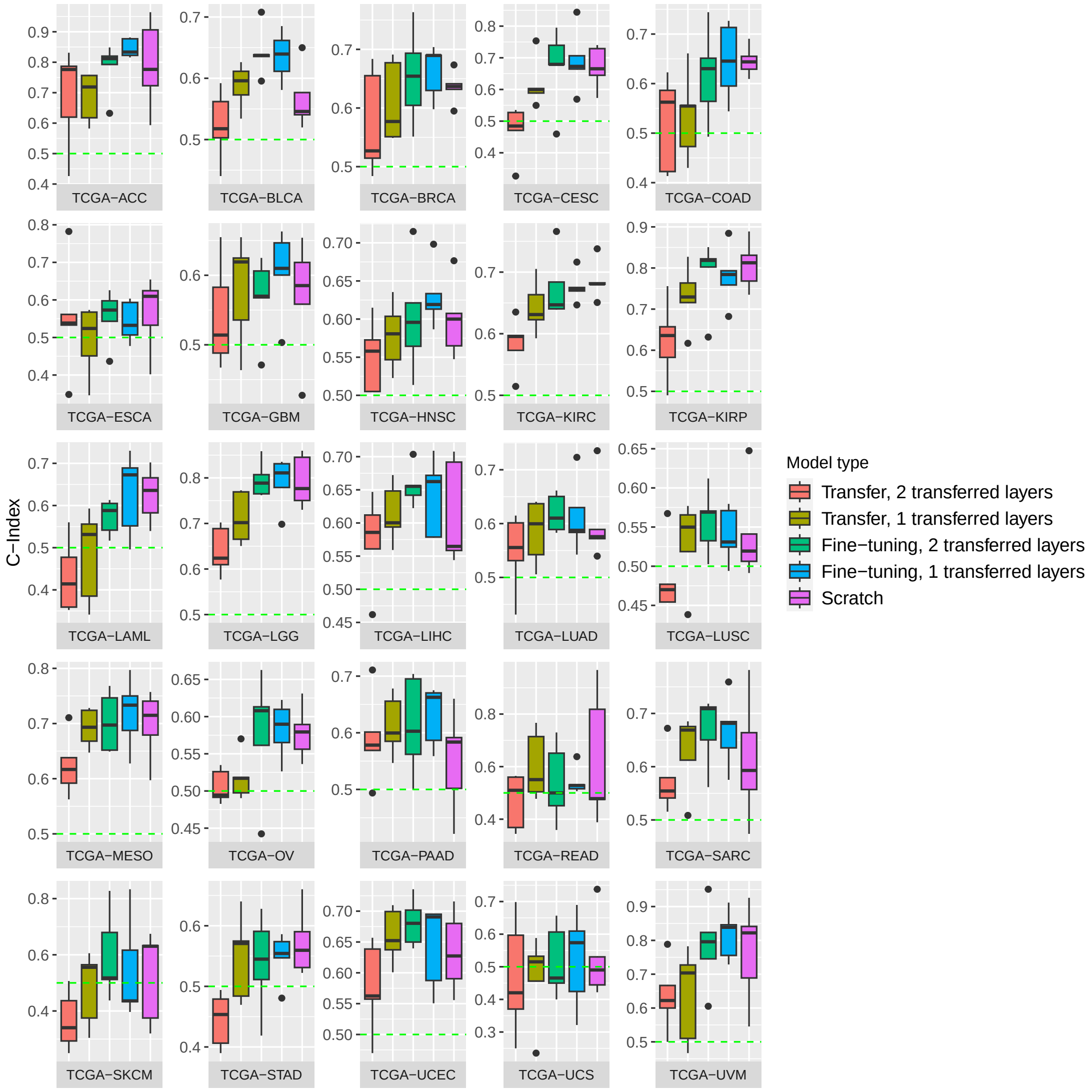


Figure A.8: Survival prediction performance of the neural network pre-trained for age prediction. Performance is depicted by C-Index boxplots for 5-fold cross-validation on TCGA.

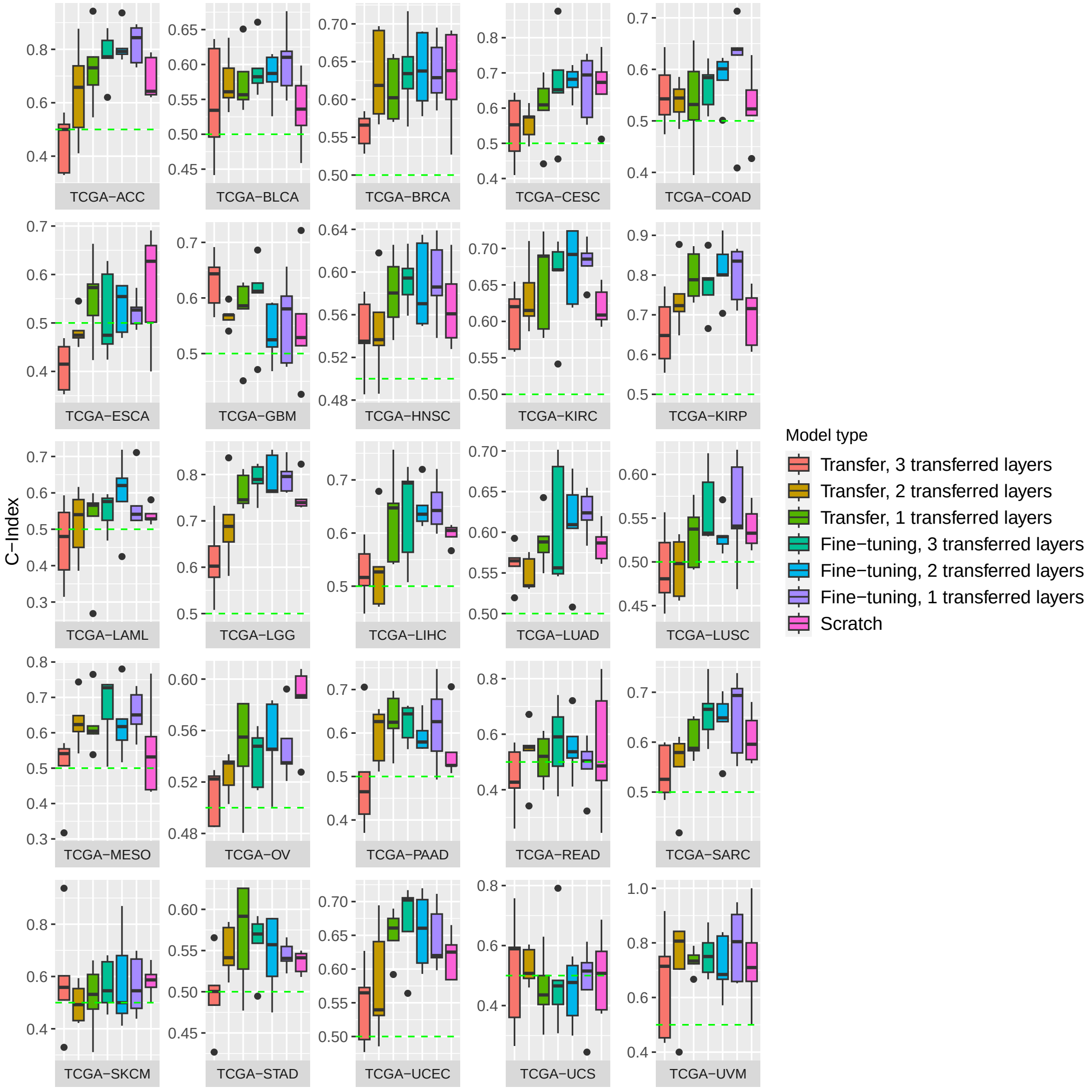


Figure A.9: Survival prediction performance of the neural network pre-trained for tissue type and age prediction. Performance is depicted by C-Index boxplots for 5-fold cross-validation on TCGA.

B

Supplementary Tables

Table B.1: Summary of TCGA cohorts used for pan-cancer survival prediction and survival network identification (from GDC data releases v22.0 and v24.0). *Number of Pan-Cancer Features* refers to the number of pan-cancer gene expression features that were also among the important features in single-cohort training for the respective cohort in XGBoost survival prediction. *IQR* indicates the interquartile range, which measures how much the data is spread. Information on the organ system of each cancer cohort was obtained from [133]. More detailed information on the different cancer types can be found at `https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga/studied-cancers`. This table was published in [170].

Cohort Abbreviation	Cohort Name	Organ System	Number of Pan-cancer Features	Number of Patients	Number of Uncensored Patients	Median Age (IQR)	Gender
TCGA-ACC	Adrenocortical carcinoma	Endocrine	2027	79	28	49.0 (24.50)	48 female, 31 male
TCGA-BLCA	Bladder urothelial carcinoma	Urologic	4814	401	176	68.0 (16.00)	104 female, 297 male

TCGA-BRCA	Breast invasive carcinoma	Gyneco-logic	4864	1068	149	58.0 (18.25)	1056 female, 12 male
TCGA-CESC	Cervical squamous cell carcinoma and endocervical adenocarcinoma	Gyneco-logic	4359	291	72	46.0 (18.00)	291 female
TCGA-COAD	Colon adenocarcinoma	Gastrointestinal	4719	433	95	68.0 (19.00)	200 female, 233 male
TCGA-ESCA	Esophageal carcinoma	Gastrointestinal	4048	160	63	60.0 (19.00)	23 female, 137 male
TCGA-GBM	Glioblastoma multiforme	Central nervous system	4351	151	122	60.0 (18.00)	54 female, 97 male
TCGA-HNSC	Head and neck squamous cell carcinoma	Head and neck	5157	498	217	61.0 (16.00)	132 female, 366 male
TCGA-KIRC	Kidney renal clear cell carcinoma	Urologic	4605	526	171	60.0 (17.00)	183 female, 343 male
TCGA-KIRP	Kidney renal papillary cell carcinoma	Urologic	3540	283	44	61.0 (17.00)	75 female, 208 male

TCGA-LAML	Acute myeloid leukemia	Hematologic and lymphatic malignancies	3827	130	78	55.5 (24.75)	60 female, 70 male
TCGA-LGG	Brain lower grade glioma	Central nervous system	3999	505	125	41.0 (20.00)	226 female, 279 male
TCGA-LIHC	Liver hepatocellular carcinoma	Gastrointestinal	4678	364	130	61.0 (17.25)	119 female, 245 male
TCGA-LUAD	Lung adenocarcinoma	Thoracic	4984	490	179	66.0 (13.00)	266 female, 224 male
TCGA-LUSC	Lung squamous cell carcinoma	Thoracic	5391	488	211	68.0 (11.00)	127 female, 361 male
TCGA-MESO	Mesothelioma	Thoracic	3731	84	72	64.0 (12.00)	15 female, 69 male
TCGA-OV	Ovarian serous cystadenocarcinoma	Gynecologic	4696	372	229	59.0 (17.00)	372 female
TCGA-PAAD	Pancreatic adenocarcinoma	Gastrointestinal	4489	176	92	65.0 (16.00)	80 female, 96 male
TCGA-READ	Rectum adenocarcinoma	Gastrointestinal	3137	156	26	65.0 (15.00)	68 female, 88 male

TCGA-SARC	Sarcoma	Soft tissue	4503	256	98	60.5 (17.25)	139 female, 117 male
TCGA-SKCM	Skin cutaneous melanoma	Melanocytic	3116	98	28	63.5 (18.75)	40 female, 58 male
TCGA-STAD	Stomach adenocarcinoma	Gastrointestinal	4864	344	143	67.0 (14.00)	123 female, 221 male
TCGA-UCEC	Uterine corpus endometrial carcinoma	Gynecologic	4723	537	90	64.0 (14.00)	537 female
TCGA-UCS	Uterine carcinosarcoma	Soft tissue	3436	54	33	68.5 (13.50)	54 female
TCGA-UVM	Uveal melanoma	Melanocytic	2062	80	23	61.5 (23.25)	35 female, 45 male
TCGA-CHOL	Cholangiocarcinoma	Gastrointestinal	N/A	36	18	66.5 (15.50)	20 female, 16 male
TCGA-DLBC	Lymphoid neoplasm diffuse large B-cell lymphoma	Hematologic and lymphatic malignancies	N/A	47	9	58.0 (21.00)	26 female, 21 male
TCGA-KICH	Kidney chromophobe	Urologic	N/A	64	9	50.0 (19.25)	26 female, 38 male

TCGA-PCPG	Pheochromocytoma and paraganglioma	Neuralcrest derived	N/A	178	6	46.0 (23.50)	101 female, 77 male
TCGA-PRAD	Prostate adenocarcinoma	Urologic	N/A	493	10	61.0 (10.00)	493 male
TCGA-TGCT	Testicular germ cell tumors	Urologic	N/A	134	4	31.0 (11.00)	134 male
TCGA-THCA	Thyroid carcinoma	Endocrine	N/A	501	16	46.0 (23.00)	366 female, 135 male
TCGA-THYM	Thymoma	Hematologic and lymphatic malignancies	N/A	118	9	59.5 (20.50)	56 female, 62 male

Table B.2: The 103 survival network genes. Each of the 103 module genes identified in the network propagation and module identification steps is annotated with its original feature importance weight derived from pan-cancer XGBoost training, the weight and corresponding p-value after network propagation, and the type of the gene (seed gene, other pan-cancer feature, or inferred during network propagation). † after a gene name indicates known cancer genes and ‡ indicates candidate cancer genes according to NCG 6.0[144]. This table was published in[170].

Gene	Original Feature Importance	Network Propagation Weight	Network Propagation P-value	Type of Gene
BCHE	282.92	227.86	0.01	Seed gene
TMEM30B	272.23	221.27	0.01	Seed gene
INS	235.63	206.38	0.01	Seed gene
TREM1	231.79	185.75	0.01	Seed gene
ADRA1D	210.45	168.53	0.01	Seed gene
SEMA7A	190.95	154.80	0.01	Seed gene
CDH10†	177.00	143.35	0.01	Seed gene
SPP1	164.45	136.74	0.01	Seed gene
APP	50.89	135.16	0.01	Pan-cancer feature
BTLA‡	158.03	127.95	0.01	Seed gene

SCG5	153.32	123.76	0.01	Seed gene
PLAU	112.53	95.14	0.01	Pan-cancer feature
NCAM1	0.00	77.62	0.01	Inferred gene
RBL2	0.00	72.48	0.01	Inferred gene
TEAD1	10.50	71.72	0.01	Pan-cancer feature
PLAUR	2.27	65.34	0.01	Pan-cancer feature
FYN‡	1.53	56.14	0.01	Pan-cancer feature
FBLN1	0.00	54.58	0.01	Inferred gene
A2M	0.00	52.55	0.01	Inferred gene
COLQ	0.00	45.69	0.01	Inferred gene
CDK5	4.38	43.72	0.01	Pan-cancer feature
SGCD	53.26	43.37	0.01	Pan-cancer feature
PLA2G4A	5.90	42.16	0.01	Pan-cancer feature
CAV1	0.00	40.55	0.01	Inferred gene
TP63†	0.00	40.54	0.01	Inferred gene
TMEM25	42.74	38.56	0.01	Pan-cancer feature
ITGA3	39.06	33.29	0.01	Pan-cancer feature
PLG	0.00	32.80	0.01	Inferred gene
SFN	0.00	32.00	0.01	Inferred gene
TLR4‡	0.00	31.18	0.01	Inferred gene
ATP8B2‡	0.00	29.59	0.01	Inferred gene
FKBP1A	15.99	29.09	0.01	Pan-cancer feature
EPHA2‡	7.48	28.90	0.01	Pan-cancer feature
DCTN1†	0.00	28.64	0.01	Inferred gene
MMP14	7.40	28.13	0.01	Pan-cancer feature
PCSK2	1.78	27.47	0.01	Pan-cancer feature
ERBB4†	0.00	26.77	0.01	Inferred gene
MMP3	4.19	26.65	0.01	Pan-cancer feature
TNFRSF14†	0.00	23.19	0.01	Inferred gene
FGF2‡	11.51	23.13	0.01	Pan-cancer feature
LUZP1	0.00	22.71	0.01	Inferred gene
CDH6	8.46	22.23	0.01	Pan-cancer feature
AGL	0.00	22.13	0.01	Inferred gene
PRMT6	0.00	21.52	0.01	Inferred gene
IGSF21	18.58	21.05	0.01	Pan-cancer feature
GLYR1‡	0.00	20.84	0.01	Inferred gene
MMP1	13.14	20.74	0.01	Pan-cancer feature
TGFBR2†	0.00	20.68	0.01	Inferred gene

JAK2†	0.00	20.39	0.01	Inferred gene
LRP2‡	8.62	19.98	0.01	Pan-cancer feature
PICALM†	0.00	19.34	0.01	Inferred gene
RAB27B	19.94	19.27	0.01	Pan-cancer feature
ADRA1A‡	0.00	18.64	0.01	Inferred gene
RPS6KA3‡	0.00	18.51	0.01	Inferred gene
EIF4G3	0.00	18.05	0.01	Inferred gene
DPYSL3	10.86	18.05	0.01	Pan-cancer feature
HSF2BP	0.00	17.93	0.01	Inferred gene
IGF2‡	0.00	17.24	0.01	Inferred gene
GNAI3	0.00	17.14	0.01	Inferred gene
COL17A1	2.45	16.91	0.01	Pan-cancer feature
SERPINE1	558.51	451.64	0.02	Seed gene
VTN	397.78	327.91	0.02	Seed gene
LARGE2	365.16	292.92	0.02	Seed gene
TGFB1	339.00	282.17	0.02	Seed gene
PAEP	256.85	205.82	0.02	Seed gene
CLDN4	240.49	192.73	0.02	Seed gene
IGFBP1	221.18	177.40	0.02	Seed gene
ADAM9	170.62	138.16	0.02	Seed gene
DPYSL5	161.96	129.96	0.02	Seed gene
FLNC	488.05	399.73	0.03	Seed gene
UNC13D	207.33	166.16	0.04	Seed gene
PTX3	405.79	326.90	0.05	Seed gene
PIK3R3	870.79	701.96	0.06	Seed gene
DKK1	328.54	262.89	0.06	Seed gene
MLC1	243.16	194.63	0.06	Seed gene
TIMP4	218.82	175.46	0.07	Seed gene
EYA4‡	203.69	162.99	0.10	Seed gene
S100A10	164.35	134.44	0.12	Seed gene
ST8SIA3	476.24	381.05	0.13	Seed gene
ARHGEF3	157.88	126.33	0.14	Seed gene
LAD1	278.45	222.81	0.15	Seed gene
PLEC‡	281.59	229.50	0.16	Seed gene
DLG3‡	376.41	303.90	0.17	Seed gene
HJURP	348.03	278.72	0.18	Seed gene
BARX1	454.28	363.58	0.19	Seed gene
CENPA	362.38	298.74	0.19	Seed gene

BCAT1	244.63	195.73	0.21	Seed gene
VGLL2	383.43	306.77	0.23	Seed gene
CCDC88C	167.97	134.40	0.29	Seed gene
IRF6‡	432.21	345.81	0.32	Seed gene
PLA2G5	209.22	167.38	0.35	Seed gene
FRK	155.23	124.26	0.40	Seed gene
KIF2C	158.16	128.35	0.41	Seed gene
CDK6†	173.38	142.07	0.46	Seed gene
LBX1	438.19	350.56	0.51	Seed gene
CDK5R2	238.29	190.64	0.52	Seed gene
ZNF557	193.75	155.01	0.55	Seed gene
CDC20	441.45	358.13	0.83	Seed gene
IDO1	377.05	301.65	0.86	Seed gene
RPS6KA5	152.61	123.09	0.87	Seed gene
PGR‡	246.63	198.55	0.99	Seed gene
IGF2BP2†	241.03	193.01	0.99	Seed gene
ESR1†	240.89	215.59	1.00	Seed gene

Table B.3: Over-represented pathways ($p < 0.001$) computed with QIAGEN Ingenuity Pathway Analysis (IPA)[104]. *Pathway*: annotated pathway name; $-log_{10}$*(p-value)*: $-\log_{10}$ of enrichment p-value computed with Fisher's exact test; $-log_{10}$*(q-value)*: $-\log_{10}$ of Benjamini-Hochberg (cf. Section 3.5.2) adjusted p-value; *Ratio*: number of genes in the survival network that map to the respective pathway divided by the overall number of genes in the pathway; *Molecules*: survival network genes that overlap with the pathway. This table was partly published in[170].

Ingenuity canonical pathway	$-\log_{10}$ **(p-value)**	$-\log_{10}$ **(q-value)**	**Ratio**	**Molecules**
Tumor microenvironment pathway	9.34	6.48	6.25 × 10−2	*FGF2*, *IDO1*, *IGF2*, *JAK2*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *PLAU*, *SPP1*, *TGFB1*

Glucocorticoid receptor signaling	8.60	6.17	3.25 × 10−2	*A2M*, *CAV1*, *ESR1*, *JAK2*, *MMP1*, *MMP3*, *PGR*, *PIK3R3*, *PLA2G4A*, *PLA2G5*, *PLAU*, *RPS6KA5*, *SERPINE1*, *TGFB1*, *TGFBR2*
Role of tissue factor in cancer	8.55	6.17	7.76 × 10−2	*FRK*, *FYN*, *ITGA3*, *JAK2*, *MMP1*, *PIK3R3*, *PLAUR*, *RPS6KA3*, *RPS6KA5*
Hepatic fibrosis signaling pathway	6.85	4.61	3.17 × 10−2	*FGF2*, *GNAI3*, *INS*, *ITGA3*, *JAK2*, *MMP1*, *PIK3R3*, *SERPINE1*, *SPP1*, *TGFB1*, *TGFBR2*, *TLR4*
Hepatic fibrosis/Hepatic stellate cell activation	6.77	4.61	4.84 × 10−2	*A2M*, *COL17A1*, *FGF2*, *IGF2*, *MMP1*, *SERPINE1*, *TGFB1*, *TGFBR2*, *TLR4*
Coagulation system	6.31	4.23	1.43 × 10−1	*A2M*, *PLAU*, *PLAUR*, *PLG*, *SERPINE1*
HOTAIR regulatory pathway	6.18	4.20	5.00 × 10−2	*ESR1*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *SPP1*, *TGFB1*, *TLR4*
Osteoarthritis pathway	6.15	4.20	4.09 × 10−2	*DKK1*, *FGF2*, *ITGA3*, *MMP1*, *MMP3*, *SPP1*, *TGFB1*, *TGFBR2*, *TLR4*
Growth hormone signaling	6.08	4.20	8.45 × 10−2	*A2M*, *IGF2*, *JAK2*, *PIK3R3*, *RPS6KA3*, *RPS6KA5*

Inhibition of matrix metalloproteases	6.06	4.20	1.28 × 10−1	*A2M*, *MMP1*, *MMP14*, *MMP3*, *TIMP4*
Glioma invasiveness signaling	6.01	4.19	8.22 × 10−2	*PIK3R3*, *PLAU*, *PLAUR*, *PLG*, *TIMP4*, *VTN*
Reelin signaling in neurons	5.87	4.09	5.74 × 10−2	*APP*, *ARHGEF3*, *CDK5*, *FRK*, *FYN*, *ITGA3*, *PIK3R3*
Axonal Guidance signaling	5.62	3.91	2.43 × 10−2	*ADAM9*, *CDK5*, *DPYSL5*, *EPHA2*, *FYN*, *GNAI3*, *ITGA3*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *SEMA7A*
Estrogen receptor signaling	5.62	3.91	3.05 × 10−2	*CAV1*, *ESR1*, *GNAI3*, *IGF2*, *JAK2*, *MMP1*, *MMP14*, *MMP3*, *PGR*, *PIK3R3*
Leukocyte extravasation signaling	5.57	3.89	4.15 × 10−2	*CLDN4*, *GNAI3*, *ITGA3*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *TIMP4*
HIF1A signaling	5.37	3.72	3.90 × 10−2	*FGF2*, *IGF2*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *SERPINE1*, *TGFB1*
Semaphorin signaling in neurons	5.12	3.49	8.33 × 10−2	*CDK5*, *DPYSL3*, *DPYSL5*, *FYN*, *SEMA7A*
Neuroinflammation signaling pathway	5.05	3.45	3.00 × 10−2	*APP*, *JAK2*, *MMP3*, *PIK3R3*, *PLA2G4A*, *PLA2G5*, *TGFB1*, *TGFBR2*, *TLR4*

Molecular mechanisms of cancer	4.87	3.30	2.50 × 10−2	*ARHGEF3*, *CDK5*, *CDK6*, *FYN*, *GNAI3*, *ITGA3*, *JAK2*, *PIK3R3*, *TGFB1*, *TGFBR2*
Tec kinase signaling	4.86	3.30	4.05 × 10−2	*FRK*, *FYN*, *GNAI3*, *ITGA3*, *JAK2*, *PIK3R3*, *TLR4*
p38 MAPK signaling	4.80	3.26	5.08 × 10−2	*PLA2G4A*, *PLA2G5*, *RPS6KA3*, *RPS6KA5*, *TGFB1*, *TGFBR2*
Colorectal cancer metastasis signaling	4.71	3.20	3.16 × 10−2	*JAK2*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *TGFB1*, *TGFBR2*, *TLR4*
Caveolar-mediated endocytosis signaling	4.70	3.20	6.85 × 10−2	*CAV1*, *FLNC*, *FYN*, *INS*, *ITGA3*
Atherosclerosis signaling	4.61	3.13	4.72 × 10−2	*MMP1*, *MMP3*, *PLA2G4A*, *PLA2G5*, *TGFB1*, *TNFRSF14*
ERK/MAPK signaling	4.43	2.97	3.47 × 10−2	*ESR1*, *FYN*, *ITGA3*, *PIK3R3*, *PLA2G4A*, *PLA2G5*, *RPS6KA5*
Semaphorin neuronal repulsive signaling pathway	4.39	2.95	4.32 × 10−2	*CDK5*, *DPYSL3*, *DPYSL5*, *FYN*, *ITGA3*, *PIK3R3*
Oncostatin M signaling	4.37	2.94	9.30 × 10−2	*JAK2*, *MMP1*, *MMP3*, *PLAU*
Role of osteoblasts, osteoclasts and Chondrocytes in rheumatoid arthritis	4.22	2.81	3.21 × 10−2	*DKK1*, *MMP1*, *MMP14*, *MMP3*, *PIK3R3*, *SPP1*, *TGFB1*
Sperm motility	4.16	2.76	3.14 × 10−2	*EPHA2*, *ERBB4*, *FRK*, *FYN*, *JAK2*, *PLA2G4A*, *PLA2G5*

Bladder cancer signaling	4.10	2.72	5.15 × 10−2	*FGF2*, *MMP1*, *MMP14*, *MMP3*, *RPS6KA5*
Cardiac hypertrophy signaling (enhanced)	4.07	2.70	2.01 × 10−2	*ADRA1A*, *ADRA1D*, *FGF2*, *GNAI3*, *ITGA3*, *JAK2*, *PIK3R3*, *RPS6KA5*, *TGFB1*, *TGFBR2*
Role of macrophages, fibroblasts and endothelial cells in rheumatoid arthritis	4.05	2.70	2.55 × 10−2	*DKK1*, *FGF2*, *JAK2*, *MMP1*, *MMP3*, *PIK3R3*, *TGFB1*, *TLR4*
Chronic myeloid leukemia signaling	3.98	2.64	4.85 × 10−2	*CDK6*, *PIK3R3*, *RBL2*, *TGFB1*, *TGFBR2*
Insulin secretion signaling pathway	3.91	2.58	2.87 × 10−2	*EIF4G3*, *FYN*, *INS*, *JAK2*, *PCSK2*, *PIK3R3*, *RPS6KA5*
CNTF signaling	3.89	2.58	7.02 × 10−2	*JAK2*, *PIK3R3*, *RPS6KA3*, *RPS6KA5*
T cell exhaustion signaling pathway	3.84	2.54	3.43 × 10−2	*BTLA*, *JAK2*, *PIK3R3*, *TGFB1*, *TGFBR2*, *TNFRSF14*
Regulation of the epithelial mesenchymal transition by growth factors pathway	3.67	2.38	3.19 × 10−2	*FGF2*,*JAK2*,*MMP1*, *PIK3R3*, *TGFB1*, *TGFBR2*
RhoGDI signaling	3.66	2.38	3.17 × 10−2	*ARHGEF3*, *CDH10*, *CDH6*, *ESR1*, *GNAI3*, *ITGA3*
IL-15 production	3.65	2.38	4.13 × 10−2	*EPHA2*, *ERBB4*, *FRK*, *FYN*,*JAK2*

Agranulocyte adhesion and diapedesis	3.61	2.35	3.11 × 10−2	*CLDN4*, *GNAI3*, *ITGA3*, *MMP1*, *MMP14*, *MMP3*
Senescence pathway	3.60	2.35	2.55 × 10−2	*CDK6*, *PIK3R3*, *RBL2*, *RPS6KA5*, *SERPINE1*, *TGFB1*, *TGFBR2*
Role of MAPK signaling in inhibiting the pathogenesis of influenza	3.43	2.20	5.33 × 10−2	*PLA2G4A*, *PLA2G5*, *RPS6KA3*, *TLR4*
mTOR signaling	3.41	2.19	2.86 × 10−2	*EIF4G3*, *FKBP1A*, *INS*, *PIK3R3*, *RPS6KA3*, *RPS6KA5*
Inhibition of angiogenesis by TSP1	3.32	2.12	8.82 × 10−2	*FYN*, *TGFB1*, *TGFBR2*
MIF-mediated glucocorticoid regulation	3.32	2.12	8.82 × 10−2	*PLA2G4A*, *PLA2G5*, *TLR4*
Necroptosis signaling pathway	3.13	1.93	3.18 × 10−2	*FKBP1A*, *PLA2G4A*, *PLA2G5*, *RBL2*, *TLR4*
Cardiac hypertrophy signaling	3.11	1.92	2.50 × 10−2	*ADRA1A*, *ADRA1D*, *GNAI3*, *PIK3R3*, *TGFB1*, *TGFBR2*
MIF regulation of innate immunity	3.04	1.86	7.14 × 10−2	*PLA2G4A*, *PLA2G5*, *TLR4*

Table B.4: Top 15 cancer-relevant upstream regulators computed with QIAGEN Ingenuity Pathway Analysis (IPA)[104]. *Upstream regulator*: gene name of an annotated cancer driver gene or a potential cancer driver gene[144] as an upstream regulator of the pan-cancer survival network; *Molecule type*: molecule type of the upstream regulator; *P-value of overlap*: Fisher test p-value for over-representation of survival network genes in the target set of the upstream regulator; *Target molecules in dataset*: survival network genes downstream of the upstream regulator. This table was published in[170].

Upstream Regulator	**Molecule Type**	**P-value of Overlap**	**Target Molecules in Dataset**

JUN	Transcription regulator	1.20E-13	*A2M, APP, CAV1, CDC20, DKK1, FGF2, FLNC, IGFBP1, MMP1, MMP3, NCAM1, PGR, PLA2G4A, PLAU, PLAUR, PTX3, S100A10, SERPINE1, SPP1, TGFB1*
TNF	Cytokine	1.88E-12	*A2M, APP, CAV1, CLDN4, COLQ, DKK1, EPHA2, ESR1, FGF2, FYN, GNAI3, IDO1, IGF2, IGFBP1, INS, LAD1, MMP1, MMP14, MMP3, NCAM1, PLA2G4A, PLA2G5, PLAU, PLAUR, PTX3, S100A10, SERPINE1, SPP1, TGFB1, TGFBR2, TIMP4, TLR4, TP63, TREM1*
IL1B	Cytokine	7.12E-12	*A2M, APP, ESR1, FGF2, IDO1, IGFBP1, INS, ITGA3, MMP1, MMP14, MMP3, PCSK2, PGR, PLA2G4A, PLA2G5, PLAU, PTX3, S100A10, SERPINE1, SPP1, TGFB1, TGFBR2, TIMP4, TLR4, TREM1*
TP53	Transcription regulator	1.63E-11	*A2M, ADRA1A, APP, BCAT1, CAV1, CDC20, CDH10, CENPA, DKK1, EIF4G3, EPHA2, ESR1, EYA4, FGF2, FKBP1A, FYN, HJURP, IGF2, IGF2BP2, INS, MMP1, MMP3, PGR, PIK3R3, PLAU, PLAUR, RBL2, SERPINE1, SFN, SPP1, TGFB1, TGFBR2, TP63*
IL1A	Cytokine	1.37E-10	*APP, FGF2, MMP1, MMP14, MMP3, PLA2G4A, PLAU, PTX3, RBL2, S100A10, SERPINE1, SPP1, TGFB1*
FGF2	Growth factor	1.28E-09	*AGL, CAV1, DKK1, FGF2, IGF2, ITGA3, MMP1, MMP3, PCSK2, PLAU, PLAUR, S100A10, SERPINE1, SPP1, TGFB1*
MAP3K1	Kinase	5.30E-09	*MMP3, PGR, PLA2G4A, PLAU, PLAUR, SERPINE1, TGFB1*
EGFR	Kinase	1.03E-08	*APP, CAV1, CDK6, EPHA2, ERBB4, ESR1, IGF2, MMP1, MMP14, MMP3, PLA2G4A, PLAU, PLAUR, SEMA7A, SERPINE1*

STAT3	Transcription regulator	3.37E-08	*A2M, DKK1, ESR1, FGF2, IGFBP1, JAK2, LRP2, MMP1, MMP3, PGR, PLA2G4A, PLAU, PLAUR, SERPINE1, SPP1, TGFB1*
HRAS	Enzyme	8.45E-08	*A2M, APP, CAV1, EIF4G3, FGF2, IGF2, MMP1, MMP14, MMP3, PLA2G4A, PLAU, PLAUR, SERPINE1, SPP1, TGFB1, TP63*
CDH1	Other	1.32E-07	*ERBB4, MMP1, MMP14, MMP3, NCAM1, PLAUR, TGFB1*
AKT1	Kinase	1.36E-07	*ESR1, FGF2, IGF2, IGFBP1, MMP14, PGR, PLA2G4A, PLG, SERPINE1, SPP1, TP63*
PTEN	Phosphatase	1.38E-07	*CDC20, CDK6, ESR1, IGF2, LAD1, MMP14, MMP3, NCAM1, PLAU, PLEC, RBL2, SERPINE1, SPP1, TGFB1, TGFBR2, TNFRSF14*
FOXO1	Transcription regulator	2.36E-07	*A2M, CAV1, FYN, IGFBP1, INS, ITGA3, MMP1, MMP3, RBL2, RPS6KA3, SERPINE1, SFN, TGFB1*
SMARCA4	Transcription regulator	2.49E-07	*A2M, GNAI3, IRF6, ITGA3, MMP1, PAEP, PLAUR, PTX3, SCG5, SEMA7A, SERPINE1, SPP1, TNFRSF14, TREM1, UNC13D*
ERBB2	Kinase	2.91E-07	*CDC20, CDK6, CENPA, CLDN4, ERBB4, ESR1, IGF2, IRF6, MMP1, MMP14, MMP3, PLAU, PLAUR, RBL2, SCG5, SERPINE1, TP63, TREM1*
PRKCB	Kinase	3.05E-07	*APP, FGF2, INS, SERPINE1, TGFB1, TGFBR2*
ITGAV	Transmembrane receptor	3.43E-07	*MMP1, PLAU, SERPINE1, TGFB1, VTN*
CD36	Transmembrane receptor	3.77E-07	*FGF2, MMP1, MMP14, MMP3, PLAU, PLAUR, SERPINE1*
TP73	Transcription regulator	4.29E-07	*CDC20, EPHA2, FGF2, MMP14, NCAM1, PIK3R3, PLAU, SERPINE1, SFN, SPP1, TGFB1, TIMP4*

CDKN1A	Kinase	4.90E-07	*APP*, *CDC20*, *HJURP*, *KIF2C*, *MMP1*, *MMP3*, *RBL2*, *SERPINE1*, *TP63*, *TREM1*
FGFR1	Kinase	4.94E-07	*FGF2*, *MMP1*, *MMP14*, *MMP3*, *PLAU*, *PLAUR*, *SFN*
FOXO3	Transcription regulator	5.05E-07	*CAV1*, *CDC20*, *ESR1*, *FYN*, *IGFBP1*, *KIF2C*, *PLAU*, *RBL2*, *SERPINE1*, *TGFB1*, *TP63*
NRG1	Growth factor	5.07E-07	*CAV1*, *CDC20*, *EPHA2*, *FGF2*, *IGF2*, *PGR*, *PLAU*, *PLAUR*, *PLG*, *SERPINE1*
HGF	Growth factor	5.24E-07	*A2M*, *CAV1*, *CDC20*, *KIF2C*, *MMP1*, *MMP14*, *PLAU*, *PLAUR*, *SERPINE1*, *SPP1*, *TGFB1*, *TGFBR2*, *TP63*
ETV4	Transcription regulator	6.33E-07	*CAV1*, *MMP14*, *PLEC*, *SPP1*, *TGFBR2*
MYC	Transcription regulator	7.56E-07	*APP*, *BCAT1*, *CAV1*, *CDC20*, *CDK6*, *DKK1*, *EPHA2*, *GLYR1*, *INS*, *IRF6*, *ITGA3*, *NCAM1*, *PLAU*, *PLAUR*, *S100A10*, *SERPINE1*, *SPP1*, *TEAD1*, *TGFB1*, *TGFBR2*
RAF1	Kinase	7.95E-07	*ESR1*, *INS*, *LAD1*, *MMP1*, *MMP3*, *PLAU*, *PLAUR*, *PLEC*, *RPS6KA5*
ETS1	Transcription regulator	9.96E-07	*CAV1*, *CDK6*, *MMP1*, *MMP3*, *PGR*, *PLAU*, *SERPINE1*, *SPP1*, *TGFBR2*
NFKBIA	Transcription regulator	1.01E-06	*A2M*, *FGF2*, *IGF2*, *ITGA3*, *MMP1*, *MMP14*, *MMP3*, *PICALM*, *PLAU*, *PTX3*, *TGFB1*, *TLR4*
NR3C1	Ligand-dependent nuclear receptor	1.07E-06	*A2M*, *ADRA1D*, *APP*, *ARHGEF3*, *CAV1*, *IGFBP1*, *LAD1*, *MMP1*, *PIK3R3*, *PLA2G4A*, *SERPINE1*, *SPP1*, *TGFB1*, *TIMP4*, *VGLL2*
GLIS2	Transcription regulator	1.24E-06	*INS*, *MMP14*, *SERPINE1*, *TGFB1*
ESR1	Ligand-dependent nuclear receptor	1.65E-06	*BCAT1*, *CAV1*, *CDK5*, *CDK6*, *CENPA*, *CLDN4*, *ERBB4*, *ESR1*, *FBLN1*, *HSF2BP*, *IGF2*, *JAK2*, *MMP1*, *PGR*, *PLAU*, *PLAUR*, *PTX3*, *RBL2*, *SERPINE1*, *SPP1*, *TGFB1*

NAB2	Transcription regulator	1.94E-06	*FGF2, MMP3, PLAU, TGFB1*
TGFA	Growth factor	2.17E-06	*ESR1, PLA2G4A, PLA2G5, S100A10, SERPINE1, TGFB1*
TP63	Transcription regulator	2.37E-06	*CDK6, DKK1, EPHA2, ITGA3, MMP14, PIK3R3, PLAU, SERPINE1, SFN, TGFB1, TGFBR2, TP63*
DLC1	Other	3.28E-06	*CDK6, S100A10, SERPINE1*
AREG	Growth factor	3.29E-06	*CDC20, CENPA, HJURP, MMP1, PLAU, PTX3*
MAP2K1	Kinase	3.39E-06	*DKK1, FGF2, INS, MMP1, MMP14, MMP3, PLA2G4A, PLAUR*
PPARG	Ligand-dependent nuclear receptor	3.82E-06	*APP, CAV1, CDK6, IGFBP1, INS, MMP14, MMP3, SERPINE1, SPP1, TGFBR2, TIMP4, TLR4*
NOTCH1	Transcription regulator	4.45E-06	*DKK1, FGF2, MMP1, MMP3, SERPINE1, SPP1, TGFB1, TGFBR2, TP63*
ZEB1	Transcription regulator	4.54E-06	*MMP1, PLAU, RBL2, S100A10, SERPINE1, TP63*
TGFBR2	Kinase	4.57E-06	*FGF2, MMP14, MMP3, RBL2, SERPINE1, SPP1, TGFB1, TGFBR2*
TERT	Enzyme	4.91E-06	*CAV1, COLQ, FGF2, MMP1, MMP14, MMP3, SPP1*
ABL2	Kinase	5.23E-06	*MMP1, MMP14, MMP3*
ERG	Transcription regulator	5.66E-06	*FLNC, FYN, MMP1, MMP3, PLAU, PLAUR, SPP1, TGFBR2*
IKBKB	Kinase	7.93E-06	*FYN, MMP1, MMP3, PLA2G4A, PLAU, PTX3, TGFB1, TGFBR2, TP63*

Table B.5: Optimized hyperparameters of the model pre-trained for survival prediction. The model was trained on TCGA gene expression data and hyperparameters were optimized using Optuna[5] with Tree-structured Parzen Estimator (TPE), considering the *Hyperparameters* and *Considered values* displayed in this table.

Hyperparameter		Considered values	Selected value	Notes
Hidden layers	Number of layers	[1,4]	1	
	Layer sizes	[32,4096]	4071	Each layer $\leq$ than preceding layer
Activation		{ReLU,ELU,GELU}	GELU	
Learning rate		[1e-7,1e-4]	2.858e-07	
Batch size		[32,512]	32	
Dropout		[0,0.9]	0	
L_2 regularization	Use regularization	{True,False}	True	
	L_2 regularization factor	[1e-4,0.1]	0.0547	
Learning rate decay	Use learning rate decay	{True,False}	False	
	Decay rate	[0.9,1]	N/A	
Weight decay (adam optimizer)	Use weight decay	{True,False}	True	
	Weight decay rate	[1e-4,0.1]	1.15e-4	
β_1 (adam optimizer)		[0.7,0.9999]	0.7110	
β_2 (adam optimizer)		[0.9,0.9999]	0.9602	

Table B.6: Optimized hyperparameters of the model pre-trained for tissue type classification. The model was trained on GTEx gene expression data and hyperparameters were optimized using Optuna[5] with Tree-structured Parzen Estimator (TPE), considering the *Hyperparameters* and *Considered values* displayed in this table.

Hyperparameter		Considered values	Selected value	Notes
Hidden layers	Number of layers	[1,4]	3	
	Layer sizes	[32,4096]	3704,1853,1407	Each layer $\leq$ than preceding layer
Activation		{ReLU,ELU,GELU}	ReLU	
Learning rate		[1e-7,1e-4]	4.830e-05	
Batch size		[32,512]	42	
Dropout		[0,0.9]	0.1	
L_2 regularization	Use regularization	{True,False}	True	
	L_2 regularization factor	[1e-8,0.1]	4.013e-4	
Learning rate decay	Use learning rate decay	{True,False}	True	
	Decay rate	[0.9,1]	0.9088	
Weight decay (adam optimizer)	Use weight decay	{True,False}	False	
	Weight decay rate	[1e-8,0.1]	N/A	
β_1 (adam optimizer)		[0.7,0.9999]	0.8960	
β_2 (adam optimizer)		[0.9,0.9999]	0.9248	

Table B.7: Optimized hyperparameters of the model pre-trained for age prediction. The model was trained on GTEx gene expression data and hyperparameters were optimized using Optuna[5] with Tree-structured Parzen Estimator (TPE), considering the *Hyperparameters* and *Considered values* displayed in this table.

Hyperparameter		Considered values	Selected value	Notes
Hidden layers	Number of layers	[1,4]	2	
	Layer sizes	[32,4096]	2713,659	Each layer $\leq$ than preceding layer
Activation		{ReLU,ELU,GELU}	ReLU	
Learning rate		[1e-7,1e-4]	6.456e-05	
Batch size		[32,512]	32	
Dropout		[0,0.9]	0	
L_2 regularization	Use regularization	{True,False}	True	
	L_2 regularization factor	[1e-8,0.1]	9.562e-05	
Learning rate decay	Use learning rate decay	{True,False}	True	
	Decay rate	[0.9,1]	0.9377	
Weight decay (adam optimizer)	Use weight decay	{True,False}	False	
	Weight decay rate	[1e-8,0.1]	N/A	
β_1 (adam optimizer)		[0.7,0.9999]	0.9346	
β_2 (adam optimizer)		[0.9,0.9999]	0.9482	

Table B.8: Optimized hyperparameters of the model pre-trained for tissue type classification and age prediction. The model was trained on GTEx gene expression data and hyperparameters were optimized using Optuna[5] with Tree-structured Parzen Estimator (TPE), considering the *Hyperparameters* and *Considered values* displayed in this table.

Hyperparameter		Considered values	Selected value	Notes
Hidden layers	Number of shared layers	[1,4]	3	
	Sizes of shared layers	[128,4096]	2242,614,140	Each layer $\leq$ than preceding layer
	Number of task layers	[0,2]	1	
	Sizes of task layers	[32,4096]	67	Each layer $\leq$ than preceding layer
Activation		{ReLU,ELU,GELU}	ReLU	
Learning rate		[1e-7,1e-4]	6.456e-05	
Batch size		[32,512]	459	
Dropout		[0,0.9]	0.1	
L_2 regularization	Use regularization	{True,False}	True	
	L_2 regularization factor	[1e-8,0.1]	1.745e-05	
Learning rate decay	Use learning rate decay	{True,False}	False	
	Decay rate	[0.9,1]	N/A	
Weight decay (adam optimizer)	Use weight decay	{True,False}	True	
	Weight decay rate	[1e-8,0.1]	0.0951	
β_1 (adam optimizer)		[0.7,0.9999]	0.8380	
β_2 (adam optimizer)		[0.9,0.9999]	0.9038	

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`.

[2] T. A. A. Abdullah, M. S. M. Zahid, and W. Ali. A review of interpretable ML in healthcare: Taxonomy, applications, challenges, and future directions. *Symmetry*, 13 (12):2439, 2021. ISSN 2073-8994. DOI10.3390/sym13122439.

[3] N. Acharya, A. Madi, H. Zhang, M. Klapholz, G. Escobar, S. Dulberg, E. Christian, M. Ferreira, K. O. Dixon, G. Fell, K. Tooley, D. Mangani, J. Xia, M. Singer, M. Bosenberg, D. Neuberg, O. Rozenblatt-Rosen, A. Regev, V. K. Kuchroo, and A. C. Anderson. Endogenous glucocorticoid signaling regulates CD8+ T cell differentiation and development of dysfunction in the tumor microenvironment. *Immunity*, 53(3): 658–671.e6, 2020. ISSN 1074-7613. DOI10.1016/j.immuni.2020.08.005.

[4] C. C. Aggarwal et al., editors. *Neural networks and deep learning*. Springer, Cham, Switzerland, 2018. ISBN 978-3-319-94463-0. DOI10.1007/978-3-319-94463-0.

[5] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pages 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. DOI10.1145/3292500.3330701.

[6] J. Anaya. OncoLnc: Linking TCGA survival data to mRNAs, miRNAs, and lncRNAs. *PeerJ Computer Science*, 2:e67, 2016. DOI10.7287/peerj.preprints.1780v1.

[7] J. Armenia, S. A. Wankowicz, D. Liu, J. Gao, R. Kundra, E. Reznik, W. K. Chatila, D. Chakravarty, G. C. Han, I. Coleman, et al. The long tail of oncogenic drivers in prostate cancer. *Nature Genetics*, 50(5):645–651, 2018. DOI10.1038/s41588-018-0078-z.

[8] E. R. Asl, D. Rostamzadeh, P. H. Duijf, S. Mafi, B. Mansoori, S. Barati, W. C. Cho, and B. Mansoori. Mutant p53 in the formation and progression of the tumor microenvironment: Friend or foe. *Life Sciences*, 315:121361, 2023. ISSN 0024-3205. DOI10.1016/j.lfs.2022.121361.

[9] B. Aslam, M. Basit, M. A. Nisar, M. Khurshid, and M. H. Rasool. Proteomics: Technologies and Their Applications. *Journal of Chromatographic Science*, 55(2):182–196, 2017. ISSN 0021-9665. DOI10.1093/chromsci/bmw167.

[10] P. C. Austin, D. S. Lee, and J. P. Fine. Introduction to the analysis of survival data in the presence of competing risks. *Circulation*, 133(6):601–609, 2016. DOI10.1161/CIRCULATIONAHA.115.017719.

[11] I. Babajide Mustapha and F. Saeed. Bioactive molecule prediction using extreme gradient boosting. *Molecules*, 21(8), 2016. ISSN 1420-3049. DOI10.3390/molecules21080983.

[12] A.-L. Barabasi and Z. N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113, 2004. DOI10.1038/nrg1272.

[13] G. Barel and R. Herwig. NetCore: a network propagation approach using node coreness. *Nucleic Acids Research*, 48(17):e98–e98, 2020. ISSN 0305-1048. DOI10.1093/nar/gkaa639.

[14] S. Behjati and P. S. Tarpey. What is next generation sequencing? *Archives of Disease in Childhood - Education and Practice*, 98(6):236–238, 2013. ISSN 1743-0585. DOI10.1136/archdischild-2013-304340.

[15] R. E. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, 1961. ISBN 9781400874668. DOI10.1515/9781400874668.

[16] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995. ISSN 0035-9246. URL `http://www.jstor.org/stable/2346101`. Accessed 20 March 2023.

[17] L. Berben, G. Floris, H. Wildiers, and S. Hatse. Cancer and aging: two tightly interconnected biological processes. *Cancers*, 13(6):1400, 2021. DOI10.3390/cancers13061400.

[18] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer New York, NY, 1 edition, 2006. ISBN 978-0387-31073-2.

[19] C. U. Blank, W. N. Haining, W. Held, P. G. Hogan, A. Kallies, E. Lugli, R. C. Lynn, M. Philip, A. Rao, N. P. Restifo, et al. Defining 'T cell exhaustion'. *Nature Reviews Immunology*, 19(11):665–674, 2019. DOI10.1038/s41577-019-0221-9.

[20] C. Bouabid, S. Rabhi, K. Thedinga, G. Barel, H. Tnani, I. Rabhi, A. Benkahla, R. Herwig, and L. Guizani-Tabbane. Host M-CSF induced gene expression drives changes in susceptible and resistant mice-derived BMdMs upon Leishmania major infection. *Frontiers in Immunology*, 14, 2023. ISSN 1664-3224. DOI10.3389/fimmu.2023.1111072.

[21] E. I. Boyle, S. Weng, J. Gollub, H. Jin, D. Botstein, J. M. Cherry, and G. Sherlock. GO::TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. *Bioinformatics*, 20(18):3710–3715, 2004. ISSN 1367-4803. DOI10.1093/bioinformatics/bth456.

[22] P. Braun and A.-C. Gingras. History of protein–protein interactions: From egg-white to complex networks. *PROTEOMICS*, 12(10):1478–1498, 2012. DOI10.1002/pmic.201100563.

[23] L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001. DOI10.1023/A:1010933404324.

[24] A. J. Brookes. The essence of SNPs. *Gene*, 234(2):177–186, 1999. ISSN 0378-1119. DOI10.1016/S0378-1119(99)00219-X.

[25] R. A. Burrell, N. McGranahan, J. Bartek, and C. Swanton. The causes and consequences of genetic heterogeneity in cancer evolution. *Nature*, 501(7467):338–345, 2013. DOI10.1038/nature12625.

[26] D. M. Camacho, K. M. Collins, R. K. Powers, J. C. Costello, and J. J. Collins. Next-generation machine learning for biological networks. *Cell*, 173(7):1581–1592, 2018. ISSN 0092-8674. DOI10.1016/j.cell.2018.05.015.

[27] L. A. Cartwright, L. Dumenci, L. A. Siminoff, and R. K. Matsuyama. Cancer patients' understanding of prognostic information. *Journal of Cancer Education*, 29:311–317, 2014. DOI10.1007/s13187-013-0603-9.

[28] J. K. C. Chan. The wonderful colors of the hematoxylin–eosin stain in diagnostic surgical pathology. *International Journal of Surgical Pathology*, 22(1):12–32, 2014. DOI10.1177/1066896913517939. PMID: 24406626.

[29] A. Cheerla and O. Gevaert. Deep learning with multimodal representation for pancancer prognosis prediction. *Bioinformatics*, 35(14):i446–i454, 2019. ISSN 1367-4803. DOI10.1093/bioinformatics/btz342.

[30] S. W. Cheetham, G. J. Faulkner, and M. E. Dinger. Overcoming challenges and dogmas to understand the functions of pseudogenes. *Nature Reviews Genetics*, 21(3): 191–201, 2020. DOI10.1038/s41576-019-0196-1.

[31] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. DOI10.1145/2939672.2939785.

[32] J. Cheng, Z. Wang, and G. Pollastri. A neural network approach to ordinal regression. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1279–1284, 2008. DOI10.1109/IJCNN.2008.4633963.

[33] D. Chicco and G. Jurman. The advantages of the matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):1–13, 2020. DOI10.1186/s12864-019-6413-7.

[34] T. Ching, X. Zhu, and L. X. Garmire. Cox-nnet: An artificial neural network method for prognosis prediction of high-throughput omics data. *PLOS Computational Biology*, 14(4):1–18, 2018. DOI10.1371/journal.pcbi.1006076.

[35] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, 2017. DOI10.1109/CVPR.2017.195.

[36] F. Chollet et al. Keras, 2015. URL `https://keras.io`.

[37] G. Cocchi, M. Lapucci, and P. Mansueto. Pareto front approximation through a multi-objective augmented Lagrangian method. *EURO Journal on Computational Optimization*, 9:100008, 2021. ISSN 2192-4406. DOI10.1016/j.ejco.2021.100008.

[38] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. DOI10.1177/001316446002000104.

[39] J. Cohen. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70(4):213, 1968. DOI10.1037/h0026256.

[40] A. Colaprico, T. C. Silva, C. Olsen, L. Garofano, C. Cava, D. Garolini, T. S. Sabedot, T. M. Malta, S. M. Pagnotta, I. Castiglioni, M. Ceccarelli, G. Bontempi, and H. Noushmehr. TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data. *Nucleic Acids Research*, 44(8):e71–e71, 2015. ISSN 0305-1048. DOI10.1093/nar/gkv1507.

[41] J. C. Costello, L. M. Heiser, E. Georgii, M. Gönen, M. P. Menden, N. J. Wang, M. Bansal, M. Ammad-Ud-Din, P. Hintsanen, S. A. Khan, et al. A community effort to assess and improve drug sensitivity prediction algorithms. *Nature Biotechnology*, 32 (12):1202–1212, 2014. DOI10.1038/nbt.2877.

[42] M. Coutelier, M. Holtgrewe, M. Jäger, R. Flöttman, M. A. Mensah, M. Spielmann, P. Krawitz, D. Horn, D. Beule, and S. Mundlos. Combining callers improves the detection of copy number variants from whole-genome sequencing. *European Journal of Human Genetics*, 30(2):178–186, 2022. DOI10.1038/s41431-021-00983-x.

[43] L. Cowen, T. Ideker, B. J. Raphael, and R. Sharan. Network propagation: a universal amplifier of genetic associations. *Nature Reviews Genetics*, 18(9):551–562, 2017. DOI10.1038/nrg.2017.38.

[44] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972. DOI10.1111/j.2517-6161.1972.tb00899.x.

[45] D. R. Cox and D. Oakes. *Analysis of Survival Data*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1984. DOI10.1201/9781315137438.

[46] F. Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970. DOI10.1038/227561a0.

[47] F. H. Crick. On protein synthesis. In *Symp Soc Exp Biol*, volume 12, pages 138–163, 1958. PMID: 13580867.

[48] M. De Palma, D. Biziato, and T. V. Petrova. Microenvironmental regulation of tumour angiogenesis. *Nature Reviews Cancer*, 17(8):457–474, 2017. DOI10.1038/nrc.2017.51.

[49] C. Denkert, G. von Minckwitz, S. Darb-Esfahani, B. Lederer, B. I. Heppner, K. E. Weber, J. Budczies, J. Huober, F. Klauschen, J. Furlanetto, W. D. Schmitt, J.-U. Blohmer, T. Karn, B. M. Pfitzner, S. Kümmel, K. Engels, A. Schneeweiss, A. Hartmann, A. Noske, P. A. Fasching, C. Jackisch, M. van Mackelenbergh, P. Sinn, C. Schem, C. Hanusch, M. Untch, and S. Loibl. Tumour-infiltrating lymphocytes and prognosis in different subtypes of breast cancer: a pooled analysis of 3771 patients treated with neoadjuvant therapy. *The Lancet Oncology*, 19(1):40–50, 2018. ISSN 1470-2045. DOI10.1016/S1470-2045(17)30904-X.

[50] R. A. DePinho. The age of cancer. *Nature*, 408(6809):248–254, 2000. DOI10.1038/35041694.

[51] O. Dereli, C. Oğuz, and M. Gönen. Path2Surv: Pathway/gene set-based survival analysis using multiple kernel learning. *Bioinformatics*, 35(24):5137–5145, 2019. ISSN 1367-4803. DOI10.1093/bioinformatics/btz446.

[52] H. Dillekås, M. S. Rogers, and O. Straume. Are 90% of deaths from cancer caused by metastases? *Cancer Medicine*, 8(12):5574–5576, 2019. DOI10.1002/cam4.2474.

[53] Z. Dong and Y. Chen. Transcriptomics: advances and approaches. *Science China Life Sciences*, 56:960–967, 2013. DOI10.1007/s11427-013-4557-2.

[54] M. Fane and A. T. Weeraratna. How the ageing microenvironment influences tumour progression. *Nature Reviews Cancer*, 20(2):89–106, 2020. DOI10.1038/s41568-019-0222-9.

[55] J. Ferlay, M. Ervik, F. Lam, M. Colombet, L. Mery, M. Piñeros, A. Znaor, I. Soerjomataram, and F. Bray. Global Cancer Observatory: Cancer today, 2020. URL `https://gco.iarc.fr/today`. Accessed: 23 November 2023.

[56] M. Feurer and F. Hutter. Hyperparameter optimization. In F. Hutter, L. Kotthoff, and J. Vanschoren, editors, *Automated Machine Learning: Methods, Systems, Challenges*, pages 3–33. Springer International Publishing, Cham, 2019. ISBN 978-3-030-05318-5. DOI10.1007/978-3-030-05318-5_1.

[57] M. M. Fidler-Benaoudia and F. Bray. Transitions in human development and the global cancer burden. In C. P. Wild, E. Weiderpass, and B. W. Stewart, editors, *World Cancer Report: Cancer Research for Cancer Prevention*, chapter 1.3, pages 34–44. International Agency for Research on Cancer, Lyon, France, 2020. ISBN 978-92-832-0448-0. URL `http://publications.iarc.fr/586`. Licence: CC BY-NC-ND 3.0 IGO.

[58] J. Fürnkranz. Decision tree. In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning*, pages 263–267. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. DOI10.1007/978-0-387-30164-8_204.

[59] A. Garas, F. Schweitzer, and S. Havlin. A k-shell decomposition method for weighted networks. *New Journal of Physics*, 14(8):083030, 2012. DOI10.1088/1367-2630/14/8/083030.

[60] M. A. García-Campos, J. Espinal-Enríquez, and E. Hernández-Lemus. Pathway analysis: State of the art. *Frontiers in Physiology*, 6, 2015. ISSN 1664-042X. DOI10.3389/fphys.2015.00383.

[61] M. Gillespie, B. Jassal, R. Stephan, M. Milacic, K. Rothfels, A. Senff-Ribeiro, J. Griss, C. Sevilla, L. Matthews, C. Gong, et al. The reactome pathway knowledgebase 2022. *Nucleic Acids Research*, 50(D1):D687–D692, 2022. DOI10.1093/nar/gkab1028.

[62] J. Gillis, S. Ballouz, and P. Pavlidis. Bias tradeoffs in the creation and analysis of protein–protein interaction networks. *Journal of Proteomics*, 100:44–54, 2014. DOI10.1016/j.jprot.2014.01.020.

[63] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL `http://www.deeplearningbook.org`.

[64] M. Grandini, E. Bagli, and G. Visani. Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*, 2020. DOI10.48550/arXiv.2008.05756.

[65] Y.-J. Guo, W.-W. Pan, S.-B. Liu, Z.-F. Shen, Y. Xu, and L.-L. Hu. ERK/MAPK signalling pathway and tumorigenesis. *Experimental and Therapeutic Medicine*, 19(3): 1997–2007, 2020. DOI10.3892/etm.2020.8454.

[66] B. Ham, M. C. Fernandez, Z. D'costa, and P. Brodt. The diverse roles of the TNF axis in cancer progression and metastasis. *Trends in Cancer Research*, 11(1):1–27, 2016. PMID: 27928197; PMCID: PMC5138060.

[67] D. Hanahan and R. A. Weinberg. The hallmarks of cancer. *Cell*, 100(1):57–70, 2000. DOI10.1016/S0092-8674(00)81683-9.

[68] D. Hanahan and R. A. Weinberg. Hallmarks of cancer: the next generation. *Cell*, 144 (5):646–674, 2011. DOI10.1016/j.cell.2011.02.013.

[69] J. Hao, Y. Kim, T. Mallavarapu, J. H. Oh, and M. Kang. Cox-PASNet: Pathway-based sparse deep neural network for survival analysis. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 381–386, 2018. DOI10.1109/BIBM.2018.8621345.

[70] W. K. Härdle, S. Klinke, and B. Rönz. Statistical tests. In *Introduction to Statistics: Using Interactive MM*Stat Elements*, pages 311–418. Springer International Publishing, Cham, 2015. ISBN 978-3-319-17704-5. DOI10.1007/978-3-319-17704-5_9.

[71] F. E. Harrell Jr., K. L. Lee, and D. B. Mark. Multivariable prognostic models: Issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine*, 15(4):361–387, 1996. DOI10.1002/(SICI)1097-0258(19960229)15:4<361::AID-SIM168>3.0.CO;2-4.

[72] J. Harrow, A. Nagy, A. Reymond, T. Alioto, L. Patthy, S. E. Antonarakis, and R. Guigó. Identifying protein-coding genes in genomic sequences. *Genome Biology*, 10:1–8, 2009. DOI10.1186/gb-2009-10-1-201.

[73] P. Hartmann. *Mathematik für Informatiker: Ein praxisbezogenes Lehrbuch*. Vieweg, 2002. ISBN 3-528-03181-6.

[74] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402(Suppl 6761):C47–C52, 1999. DOI10.1038/35011540.

[75] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009. ISBN 978-0-387-84858-7. DOI10.1007/978-0-387-84858-7.

[76] W. Haynes. Benjamini–Hochberg method. In W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, editors, *Encyclopedia of Systems Biology*, pages 78–78. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. DOI10.1007/978-1-4419-9863-7_1215.

[77] W. Haynes. Bonferroni correction. In W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, editors, *Encyclopedia of Systems Biology*, pages 154–154. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. DOI10.1007/978-1-4419-9863-7_1213.

[78] W. Haynes. Student's t-test. In W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, editors, *Encyclopedia of Systems Biology*, pages 2023–2025. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. DOI10.1007/978-1-4419-9863-7_1184.

[79] W. Haynes. Wilcoxon rank sum test. In W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, editors, *Encyclopedia of Systems Biology*, pages 2354–2355. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. DOI10.1007/978-1-4419-9863-7_118.

[80] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. DOI10.1109/CVPR.2016.90.

[81] M. G. V. Heiden, L. C. Cantley, and C. B. Thompson. Understanding the Warburg effect: The metabolic requirements of cell proliferation. *Science*, 324(5930):1029–1033, 2009. DOI10.1126/science.1160809.

[82] R. Herwig, C. Hardt, M. Lienhard, and A. Kamburov. Analyzing and interpreting genome data at the network level with ConsensusPathDB. *Nature Protocols*, 11(10): 1889–1907, 2016. DOI10.1038/nprot.2016.117.

[83] R. Higdon. Multiple hypothesis testing. In W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, editors, *Encyclopedia of Systems Biology*, pages 1468–1469. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. DOI10.1007/978-1-4419-9863-7_1211.

[84] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933. DOI10.1037/h0071325.

[85] L.-L. Hsiao, R. Jensen, T. Yoshida, K. Clark, J. Blumenstock, and S. Gullans. Correcting for signal saturation errors in the analysis of microarray data. *Biotechniques*, 32(2): 330–336, 2002. DOI10.2144/02322st06.

[86] IBM. ILOG CPLEX interactive optimizer, 2017. URL `https://www.ibm.com/products/ilog-cplex-optimization-studio`.

[87] G. E. Idos, J. Kwok, N. Bonthala, L. Kysh, S. B. Gruber, and C. Qu. The prognostic implications of tumor infiltrating lymphocytes in colorectal cancer: a systematic review and meta-analysis. *Scientific Reports*, 10(1):3360, 2020. DOI10.1038/s41598-020-60255-4.

[88] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, and M. S. Lauer. Random survival forests. *The Annals of Applied Statistics*, 2(3):841–860, 2008. DOI10.1214/08-AOAS169.

[89] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*. Springer New York, NY, 2 edition, 2021. ISBN 978-1-0716-1418-1. DOI10.1007/978-1-0716-1418-1.

[90] A. F. Jarnuczak, H. Najgebauer, M. Barzine, D. J. Kundu, F. Ghavidel, Y. Perez-Riverol, I. Papatheodorou, A. Brazma, and J. A. Vizcaíno. An integrated landscape of protein expression in human cancer. *Scientific Data*, 8(1):115, 2021. DOI10.1038/s41597-021-00890-2.

[91] Y. Jiang, Y. Li, and B. Zhu. T-cell exhaustion in the tumor microenvironment. *Cell Death & Disease*, 6(6):e1792–e1792, 2015. DOI10.1038/cddis.2015.162.

[92] M.-Z. Jin and W.-L. Jin. The updated landscape of tumor microenvironment and drug repurposing. *Signal Transduction and Targeted Therapy*, 5(1):166, 2020. DOI10.1038/s41392-020-00280-x.

[93] A. Kamburov and R. Herwig. ConsensusPathDB 2022: molecular interactions update as a resource for network biology. *Nucleic Acids Research*, 50(D1):D587–D595, 2021. ISSN 0305-1048. DOI10.1093/nar/gkab1128.

[94] M. Kanehisa and S. Goto. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1):27–30, 2000. ISSN 0305-1048. DOI10.1093/nar/28.1.27.

[95] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger. DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network. *BMC Medical Research Methodology*, 18(24):1–12, 2018. DOI10.1186/s12874-018-0482-10.

[96] K. Kessenbrock, V. Plaks, and Z. Werb. Matrix metalloproteinases: Regulators of the tumor microenvironment. *Cell*, 141(1):52–67, 2010. ISSN 0092-8674. DOI10.1016/j.cell.2010.03.015.

[97] F. M. Khan and V. B. Zubek. Support vector regression for censored data (SVRc): A novel tool for survival analysis. In *2008 Eighth IEEE International Conference on Data Mining*, pages 863–868, 2008. DOI10.1109/ICDM.2008.50.

[98] S. Kim, K. Kim, J. Choe, I. Lee, and J. Kang. Improved survival analysis by learning shared genomic information from pan-cancer data. *Bioinformatics*, 36(Supplement_1):i389–i398, 2020. ISSN 1367-4803. DOI10.1093/bioinformatics/btaa462.

[99] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. DOI10.48550/arXiv.1412.6980.

[100] J. P. Klein and M. L. Moeschberger. Basic quantities and models. In *Survival Analysis: Techniques for Censored and Truncated Data*, pages 21–61. Springer New York, New York, NY, 2003. ISBN 978-0-387-21645-4. DOI10.1007/0-387-21645-6_2.

[101] T. A. Knijnenburg, L. F. A. Wessels, M. J. T. Reinders, and I. Shmulevich. Fewer permutations, more accurate p-values. *Bioinformatics*, 25(12):i161–i168, 2009. ISSN 1367-4803. DOI10.1093/bioinformatics/btp211.

[102] G. C. K. W. Koh, P. Porras, B. Aranda, H. Hermjakob, and S. E. Orchard. Analyzing protein-–protein interaction networks. *Journal of Proteome Research*, 11(4):2014–2031, 2012. DOI10.1021/pr201211w. PMID: 22385417.

[103] S. Köhler, S. Bauer, D. Horn, and P. N. Robinson. Walking the interactome for prioritization of candidate disease genes. *The American Journal of Human Genetics*, 82 (4):949–958, 2008. DOI10.1016/j.ajhg.2008.02.013.

[104] A. Krämer, J. Green, J. Pollard, Jack, and S. Tugendreich. Causal analysis approaches in Ingenuity Pathway Analysis. *Bioinformatics*, 30(4):523–530, 2013. ISSN 1367-4803. DOI10.1093/bioinformatics/btt703.

[105] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, 2017. ISSN 0001-0782. DOI10.1145/3065386.

[106] M. Kuhn and K. Johnson. Over-fitting and model tuning. In *Applied Predictive Modeling*, pages 61–92. Springer New York, New York, NY, 2013. ISBN 978-1-4614-6849-3. DOI10.1007/978-1-4614-6849-3_4.

[107] H. Kvamme, Ø. Borgan, and I. Scheel. Time-to-event prediction with neural networks and Cox regression. *arXiv preprint arXiv:1907.00825*, 20(129):1–30, 2019. DOI10.48550/arXiv.1907.00825.

[108] S. Lauer and D. Gresham. An evolving view of copy number variants. *Current Genetics*, 65(6):1287–1295, 2019. DOI10.1007/s00294-019-00980-0.

[109] M. D. Leiserson, F. Vandin, H.-T. Wu, J. R. Dobson, J. V. Eldridge, J. L. Thomas, A. Papoutsaki, Y. Kim, B. Niu, M. McLellan, et al. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nature Genetics*, 47(2):106–114, 2015. DOI10.1038/ng.3168.

[110] J. Li and C. Liu. Coding or noncoding, the converging concepts of RNAs. *Frontiers in Genetics*, 10:496, 2019. DOI10.3389/fgene.2019.00496.

[111] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. *ACM Comput. Surv.*, 50(6), 2017. ISSN 0360-0300. DOI10.1145/3136625.

[112] Y. Li, J. Wang, J. Ye, and C. K. Reddy. A multi-task learning formulation for survival analysis. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1715–1724, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. DOI10.1145/2939672.2939857.

[113] A. Liberzon, C. Birger, H. Thorvaldsdóttir, M. Ghandi, J. Mesirov, and P. Tamayo. The molecular signatures database hallmark gene set collection. *Cell Systems*, 1(6): 417–425, 2015. ISSN 2405-4712. DOI10.1016/j.cels.2015.12.004.

[114] C.-Y. Lin, S.-T. Chen, Y.-M. Jeng, C.-C. Yeh, H.-Y. Chou, Y.-T. Deng, C.-C. Chang, and M. Y.-P. Kuo. Insulin-like growth factor II mRNA-binding protein 3 expression promotes tumor formation and invasion and predicts poor prognosis in oral squamous cell carcinoma. *Journal of Oral Pathology & Medicine*, 40(9):699–705, 2011. DOI10.1111/j.1600-0714.2011.01019.x.

[115] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021. ISSN 1099-4300. DOI10.3390/e23010018.

[116] J. P. Lloyd, M. B. Soellner, S. D. Merajver, and J. Z. Li. Impact of between-tissue differences on pan-cancer predictions of drug sensitivity. *PLOS Computational Biology*, 17(2):1–25, 2021. DOI10.1371/journal.pcbi.1008720.

[117] P. Lochhead, Y. Imamura, T. Morikawa, A. Kuchiba, M. Yamauchi, X. Liao, Z. R. Qian, R. Nishihara, K. Wu, J. A. Meyerhardt, C. S. Fuchs, and S. Ogino. Insulin-like growth factor 2 messenger RNA binding protein 3 (IGF2BP3) is a marker of unfavourable prognosis in colorectal cancer. *European Journal of Cancer*, 48(18):3405–3413, 2012. ISSN 0959-8049. DOI10.1016/j.ejca.2012.06.021.

[118] J. Lonsdale, J. Thomas, M. Salvatore, R. Phillips, E. Lo, S. Shad, R. Hasz, G. Walters, F. Garcia, N. Young, et al. The genotype-tissue expression (GTEx) project. *Nature Genetics*, 45(6):580–585, 2013. DOI10.1038/ng.2653.

[119] R. Lowe, N. Shirley, M. Bleackley, S. Dolan, and T. Shafee. Transcriptomics technologies. *PLOS Computational Biology*, 13(5):1–23, 2017. DOI10.1371/journal.pcbi.1005457.

[120] A. Lujambio, L. Akkari, J. Simon, D. Grace, D. F. Tschaharganeh, J. E. Bolden, Z. Zhao, V. Thapar, J. A. Joyce, V. Krizhanovsky, and S. W. Lowe. Non-cell-autonomous tumor suppression by p53. *Cell*, 153(2):449–460, 2013. ISSN 0092-8674. DOI10.1016/j.cell.2013.03.020.

[121] C. López-Otín, F. Pietrocola, D. Roiz-Valle, L. Galluzzi, and G. Kroemer. Meta-hallmarks of aging and cancer. *Cell Metabolism*, 35(1):12–35, 2023. ISSN 1550-4131. DOI10.1016/j.cmet.2022.11.001.

[122] C. Mancarella and K. Scotlandi. IGF2BP3 from physiology to cancer: Novel discoveries, unsolved issues, and future perspectives. *Frontiers in Cell and Developmental Biology*, 7, 2020. ISSN 2296-634X. DOI10.3389/fcell.2019.00363.

[123] M. Maqsood, F. Nazir, U. Khan, F. Aadil, H. Jamal, I. Mehmood, and O.-y. Song. Transfer learning assisted classification and detection of Alzheimer's disease stages using 3D MRI scans. *Sensors*, 19(11), 2019. ISSN 1424–8220. DOI10.3390/s19112645.

[124] A. B. Mariotto, A.-M. Noone, N. Howlader, H. Cho, G. E. Keel, J. Garshell, S. Woloshin, and L. M. Schwartz. Cancer survival: an overview of measures, uses, and interpretation. *Journal of the National Cancer Institute Monographs*, 2014(49): 145–186, 2014. DOI10.1093/jncimonographs/lgu024.

[125] M. Martens, A. Ammar, A. Riutta, A. Waagmeester, D. N. Slenter, K. Hanspers, R. A. Miller, D. Digles, E. N. Lopes, F. Ehrhart, et al. WikiPathways: connecting communities. *Nucleic Acids Research*, 49(D1):D613–D621, 2021. DOI10.1093/nar/gkaa1024.

[126] W. McLaren, L. Gil, S. E. Hunt, H. S. Riat, G. R. Ritchie, A. Thormann, P. Flicek, and F. Cunningham. The ensembl variant effect predictor. *Genome Biology*, 17(1): 1–14, 2016. DOI10.1186/s13059-016-0974-4.

[127] E. Moiso. Manual curation of TCGA treatment data and identification of potential markers of therapy response. *medRxiv*, pages 2021–04, 2021. DOI10.1101/2021.04.30.21251941.

[128] M. Mounir, M. Lucchetta, T. C. Silva, C. Olsen, G. Bontempi, X. Chen, H. Noushmehr, A. Colaprico, and E. Papaleo. New functionalities in the TCGAbiolinks package for the study and integration of cancer data from GDC and GTEx. *PLOS Computational Biology*, 15(3):1–18, 2019. DOI10.1371/journal.pcbi.1006701.

[129] National Cancer Institute. Cancer of the Urinary Bladder - Cancer Stat Facts, 2023. URL https://seer.cancer.gov/statfacts/html/urinb.html. Accessed: 22 December 2023.

[130] National Cancer Institute (NCI). Age and cancer risk, 2021. URL https://www.cancer.gov/about-cancer/causes-prevention/risk/age. Accessed: 29 November 2023.

[131] A. Ogunleye and Q.-G. Wang. XGBoost model for chronic kidney disease diagnosis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(6):2131–2140, 2020. DOI10.1109/TCBB.2019.2911071.

[132] R. Oughtred, J. Rust, C. Chang, B.-J. Breitkreutz, C. Stark, A. Willems, L. Boucher, G. Leung, N. Kolas, F. Zhang, S. Dolma, J. Coulombe-Huntington, A. Chatr-aryamontri, K. Dolinski, and M. Tyers. The BioGRID database: A comprehensive biomedical resource of curated protein, genetic, and chemical interactions. *Protein Science*, 30(1):187–200, 2021. DOI10.1002/pro.3978.

[133] T. Z. Parris. Pan-cancer analyses of human nuclear receptors reveal transcriptome diversity and prognostic value across cancer types. *Scientific Reports*, 10(1):1–12, 2020. DOI10.1038/s41598-020-58842-6.

[134] K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901. DOI10.1080/14786440109462720.

[135] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[136] H. Phan, O. Y. Chén, P. Koch, Z. Lu, I. McLoughlin, A. Mertins, and M. De Vos. Towards more accurate automatic sleep staging via deep transfer learning. *IEEE Transactions on Biomedical Engineering*, 68(6):1787–1798, 2021. DOI10.1109/TBME.2020.3020381.

[137] S. Prasad. Hypothesis testing. In *Elementary Statistical Methods*, pages 147–240. Springer Nature Singapore, Singapore, 2022. ISBN 978-981-19-0596-4. DOI10.1007/978-981-19-0596-4_4.

[138] P. Prasse, P. Iversen, M. Lienhard, K. Thedinga, C. Bauer, R. Herwig, and T. Scheffer. Matching anticancer compounds and tumor cell lines by neural networks with ranking loss. *NAR Genomics and Bioinformatics*, 4(1):lqab128, 2022. ISSN 2631-9268. DOI10.1093/nargab/lqab128.

[139] P. Prasse, P. Iversen, M. Lienhard, K. Thedinga, R. Herwig, and T. Scheffer. Pre-training on in vitro and fine-tuning on patient-derived data improves deep neural networks for anti-cancer drug-sensitivity prediction. *Cancers*, 14(16), 2022. ISSN 2072-6694. DOI10.3390/cancers14163950.

[140] D. F. Quail and J. A. Joyce. Microenvironmental regulation of tumor progression and metastasis. *Nature Medicine*, 19(11):1423–1437, 2013. DOI10.1038/nm.3394.

[141] M. Quaresma, M. P. Coleman, and B. Rachet. 40-year trends in an index of survival for all cancers combined and survival adjusted for age and sex for each cancer in England and Wales, 1971–2011: a population-based study. *The Lancet*, 385(9974):1206–1218, 2015. ISSN 0140-6736. DOI10.1016/S0140-6736(14)61396-9.

[142] K. Raman. Construction and analysis of protein–protein interaction networks. *Automated Experimentation*, 2:1–11, 2010. DOI10.1186/1759-4499-2-2.

[143] E. B. Rankin, J.-M. Nam, and A. J. Giaccia. Hypoxia: Signaling the metastatic cascade. *Trends in Cancer*, 2(6):295–304, 2016. ISSN 2405-8033. DOI10.1016/j.trecan.2016.05.006.

[144] D. Repana, J. Nulsen, L. Dressler, M. Bortolomeazzi, S. K. Venkata, A. Tourna, A. Yakovleva, T. Palmieri, and F. D. Ciccarelli. The Network of Cancer Genes (NCG): a comprehensive catalogue of known and candidate cancer genes from cancer sequencing screens. *Genome Biology*, 20:1–12, 2019. DOI10.1186/s13059-018-1612-0.

[145] M. Ruffalo, M. Koyutürk, and R. Sharan. Network-based integration of disparate omic data to identify ”silent players” in cancer. *PLOS Computational Biology*, 11(12):1–20, 2015. DOI10.1371/journal.pcbi.1004595.

[146] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007. ISSN 1367-4803. DOI10.1093/bioinformatics/btm344.

[147] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 44(1.2):206–226, 2000. DOI10.1147/rd.441.0206.

[148] R. A. Saxton and D. M. Sabatini. mTOR signaling in growth, metabolism, and disease. *Cell*, 168(6):960–976, 2017. DOI10.1016/j.cell.2017.02.004.

[149] C. F. Schaefer, K. Anthony, S. Krupa, J. Buchoff, M. Day, T. Hannay, and K. H. Buetow. PID: the Pathway Interaction Database. *Nucleic Acids Research*, 37(suppl_1):D674–D679, 2008. ISSN 0305-1048. DOI10.1093/nar/gkn653.

[150] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990. DOI10.1007/BF00116037.

[151] G. Schneider, M. Schmidt-Supprian, R. Rad, and D. Saur. Tissue-specific tumorigenesis: context matters. *Nature Reviews Cancer*, 17(4):239–253, 2017. DOI10.1038/nrc.2017.5.

[152] J. P. Shaffer. Multiple hypothesis testing. *Annual Review of Psychology*, 46(1):561–584, 1995. DOI10.1146/annurev.ps.46.020195.003021.

[153] D. Shamsutdinova, D. Stamate, A. Roberts, and D. Stahl. Combining Cox model and tree-based algorithms to boost performance and preserve interpretability for health outcomes. In I. Maglogiannis, L. Iliadis, J. Macintyre, and P. Cortez, editors, *Artificial Intelligence Applications and Innovations*, pages 170–181, Cham, 2022. Springer International Publishing. ISBN 978-3-031-08337-2. DOI10.1007/978-3-031-08337-2_15.

[154] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. DOI10.1002/j.1538-7305.1948.tb01338.x.

[155] Y.-S. Sheen, Y.-H. Liao, M.-H. Lin, C.-Y. Chu, B.-Y. Ho, M.-C. Hsieh, P.-C. Chen, S.-T. Cha, Y.-M. Jeng, C.-C. Chang, H.-C. Chiu, S.-H. Jee, M.-L. Kuo, and C.-Y. Chu. IMP-3 promotes migration and invasion of melanoma cells by modulating the expression of HMGA2 and predicts poor prognosis in melanoma. *Journal of Investigative Dermatology*, 135(4):1065–1073, 2015. ISSN 0022-202X. DOI10.1038/jid.2014.480.

[156] P. K. Shivaswamy, W. Chu, and M. Jansche. A support vector approach to censored targets. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 655–660, 2007. DOI10.1109/ICDM.2007.93.

[157] R. L. Siegel, K. D. Miller, and A. Jemal. Cancer statistics, 2018. *CA: A Cancer Journal for Clinicians*, 68(1):7–30, 2018. DOI10.3322/caac.21442.

[158] R. L. Siegel, K. D. Miller, N. S. Wagle, and A. Jemal. Cancer statistics, 2023. *CA: A Cancer Journal for Clinicians*, 73(1):17–48, 2023. DOI10.3322/caac.21763.

[159] T. C. Silva, A. Colaprico, C. Olsen, F. D'Angelo, G. Bontempi, M. Ceccarelli, and H. Noushmehr. TCGA workflow: Analyze cancer genomics and epigenomics data using Bioconductor packages. *F1000Research*, 5, 2016. DOI10.12688/f1000research.8923.2.

[160] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. DOI10.48550/arXiv.1409.1556.

[161] Y. Song, Z. Bi, Y. Liu, F. Qin, Y. Wei, and X. Wei. Targeting RAS–RAF–MEK–ERK signaling pathway in human cancer: Current status in clinical trials. *Genes & Diseases*, 10(1):76–88, 2023. ISSN 2352-3042. DOI10.1016/j.gendis.2022.05.006.

[162] L. Statello, C.-J. Guo, L.-L. Chen, and M. Huarte. Gene regulation by long non-coding RNAs and its biological functions. *Nature Reviews Molecular Cell Biology*, 22 (2):96–118, 2021. DOI10.1038/s41580-020-00315-9.

[163] Statistisches Bundesamt (Destatis). Genesis-Online, 2023. URL `https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Todesursachen/_inhalt.html`. Accessed: 23 November 2023, Data license by-2-0 (`www.govdata.de/dl-de/by-2-0`).

[164] K. B. Stibius and K. Sneppen. Modeling the two-hybrid detector: Experimental bias on protein interaction networks. *Biophysical Journal*, 93(7):2562–2566, 2007. ISSN 0006-3495. DOI10.1529/biophysj.106.098236.

[165] J. Sun, K. A. Kopciuk, and X. Lu. Polynomial spline estimation of partially linear single-index proportional hazards regression models. *Computational Statistics & Data Analysis*, 53(1):176–188, 2008. ISSN 0167-9473. DOI10.1016/j.csda.2008.07.003.

[166] Y. Sun, T. Lu, C. Wang, Y. Li, H. Fu, J. Dong, and Y. Xu. TransBoost: A boosting-tree kernel transfer learning algorithm for improving financial inclusion. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(11):12181–12190, 2022. DOI10.1609/aaai.v36i11.21478.

[167] D. Szklarczyk, A. L. Gable, K. C. Nastou, D. Lyon, R. Kirsch, S. Pyysalo, N. T. Doncheva, M. Legeay, T. Fang, P. Bork, L. J. Jensen, and C. von Mering. The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets. *Nucleic Acids Research*, 49 (D1):D605–D612, 2020. ISSN 0305-1048. DOI10.1093/nar/gkaa1074.

[168] A. Tan, G. R. Abecasis, and H. M. Kang. Unified representation of genetic variants. *Bioinformatics*, 31(13):2202–2204, 2015. ISSN 1367-4803. DOI10.1093/bioinformatics/btv112.

[169] K. Thedinga and R. Herwig. Gradient tree boosting and network propagation for the identification of pan-cancer survival networks. *STAR Protocols*, 3(2):101353, 2022. ISSN 2666-1667. DOI10.1016/j.xpro.2022.101353.

[170] K. Thedinga and R. Herwig. A gradient tree boosting and network propagation derived pan-cancer survival network of the tumor microenvironment. *iScience*, 25(1), 2022. ISSN 2589-0042. DOI10.1016/j.isci.2021.103617.

[171] S. Theußl, F. Schwendinger, and K. Hornik. ROI: An extensible R optimization infrastructure. *Journal of Statistical Software*, 94(15):1–64, 2020. DOI10.18637/jss.v094.i15.

[172] V. Thorsson, D. L. Gibbs, S. D. Brown, D. Wolf, D. S. Bortone, T.-H. O. Yang, E. Porta-Pardo, G. F. Gao, C. L. Plaisier, J. A. Eddy, et al. The immune landscape of cancer. *Immunity*, 48(4):812–830, 2018. DOI10.1016/j.immuni.2018.03.023.

[173] T. Tian, X. Li, and J. Zhang. mTOR signaling in cancer and mTOR inhibitors in solid tumor targeting therapy. *International Journal of Molecular Sciences*, 20(3):755, 2019. DOI10.3390/ijms20030755.

[174] L. Torlay, M. Perrone-Bertolotti, E. Thomas, and M. Baciu. Machine learning—XGBoost analysis of language networks to classify patients with epilepsy. *Brain Informatics*, 4(3):159–169, 2017. DOI10.1007/s40708-017-0065-7.

[175] M. Vailati-Riboni, V. Palombo, and J. J. Loor. What are omics sciences? In B. N. Ametaj, editor, *Periparturient Diseases of Dairy Cows: A Systems Biology Approach*, chapter 1, pages 1–7. Springer International Publishing, Cham, 2017. ISBN 978-3-319-43033-1. DOI10.1007/978-3-319-43033-1_1.

[176] L. A. Vale Silva and K. Rohr. Pan-cancer prognosis prediction using multimodal deep learning. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pages 568–571, 2020. DOI10.1109/ISBI45749.2020.9098665.

[177] R. van Horssen, T. L. M. ten Hagen, and A. M. M. Eggermont. TNF-α in cancer treatment: Molecular insights, antitumor effects, and clinical utility. *The Oncologist*, 11(4):397–408, 2006. ISSN 1083-7159. DOI10.1634/theoncologist.11-4-397.

[178] B. Vogelstein, N. Papadopoulos, V. E. Velculescu, S. Zhou, L. A. Diaz, and K. W. Kinzler. Cancer genome landscapes. *Science*, 339(6127):1546–1558, 2013. DOI10.1126/science.1235122.

[179] J. Wang and Y. Chen. *Introduction to Transfer Learning: Algorithms and Practice*. Springer Singapore, 1 edition, 2023. ISBN 978-981-19-7584-4. DOI10.1007/978-981-19-7584-4.

[180] W. Wang and W. Liu. Integration of gene interaction information into a reweighted random survival forest approach for accurate survival prediction and survival biomarker discovery. *Scientific Reports*, 8(1):13202, 2018. DOI10.1038/s41598-018-31497-0.

[181] Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009. DOI10.1038/nrg2484.

[182] H. W. Wickham, Hadley. *ggplot2: Elegant Graphics for Data*. Springer International Publishing, 2009. ISBN 978-0-387-98140-6, 978-0-387-98141-3. DOI10.1007/978-0-387-98141-3.

[183] C. Wieder, C. Frainay, N. Poupin, P. Rodríguez-Mier, F. Vinson, J. Cooke, R. P. Lai, J. G. Bundy, F. Jourdan, and T. Ebbels. Pathway analysis in metabolomics: Recommendations for the use of over-representation analysis. *PLOS Computational Biology*, 17(9):1–23, 2021. DOI10.1371/journal.pcbi.1009105.

[184] M. Wolbers, M. T. Koller, V. S. Stel, B. Schaer, K. J. Jager, K. Leffondré, and G. Heinze. Competing risks analyses: objectives and approaches. *European Heart Journal*, 35(42): 2936–2941, 2014. ISSN 0195-668X. DOI10.1093/eurheartj/ehu131.

[185] World Health Organization. *WHO report on cancer: setting priorities, investing wisely and providing care for all*, page 18. World Health Organization, Geneva, 2020. ISBN ISBN 978-92-4-000129-9. Licence: CC BY-NC-SA 3.0 IGO.

[186] C. Wu, I. MacLeod, and A. I. Su. BioGPS and MyGene.info: organizing online, gene-centric information. *Nucleic Acids Research*, 41(D1):D561–D565, 2012. ISSN 0305-1048. DOI10.1093/nar/gks1114.

[187] J. Xin, A. Mark, C. Afrasiabi, G. Tsueng, M. Juchler, N. Gopal, G. S. Stupp, T. E. Putman, B. J. Ainscough, O. L. Griffith, et al. High-performance web services for querying gene and variant annotation. *Genome Biology*, 17(1):1–7, 2016. DOI10.1186/s13059-016-0953-9.

[188] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan. Introduction. In *Transfer Learning*, pages 3–22. Cambridge University Press, 2020. DOI10.1017/9781139061773.003.

[189] A. E. Yilmaz and H. Demirhan. Weighted kappa measures for ordinal multi-class classification performance. *Applied Soft Computing*, 134:110020, 2023. ISSN 1568-4946. DOI10.1016/j.asoc.2023.110020.

[190] X. Zhang, J. Liang, Z. Du, Q. Xie, T. Li, and F. Tang. Comparison of nomogram with random survival forest for prediction of survival in patients with spindle cell carcinoma. *Journal of Cancer Research and Therapeutics*, 18(7):2006–2012, 2022. DOI10.4103/jcrt.jcrt_2375_21.

[191] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2021. DOI10.1109/JPROC.2020.3004555.

[192] R. Zoncu, A. Efeyan, and D. M. Sabatini. mTOR: from growth signal integration to cancer, diabetes and ageing. *Nature Reviews Molecular Cell Biology*, 12(1):21–35, 2011. DOI10.1038/nrm3025.

[193] B. Zoph, D. Yuret, J. May, and K. Knight. Transfer learning for low-resource neural machine translation, 2016.

[194] S. Zou, Q. Tong, B. Liu, W. Huang, Y. Tian, and X. Fu. Targeting STAT3 in cancer immunotherapy. *Molecular Cancer*, 19(1):1–19, 2020. DOI10.1186/s12943-020-01258-7.

[195] I. Zwiener, B. Frisch, and H. Binder. Transforming RNA-Seq data to improve the performance of prognostic gene signatures. *PLOS ONE*, 9(1):e85150, 2014. DOI10.1371/journal.pone.0085150.

Summary

Cancer is a leading cause of death worldwide and the second leading cause of death in Germany. The primary goal of cancer therapy is to reduce mortality and improve patient survival. However, the choice of therapy is heavily influenced by the patient's prognosis, highlighting the importance of cancer survival prediction as a means to quantify the patient's risk and estimate prognosis.

This dissertation presents a cancer survival prediction approach that uses XGBoost tree ensemble learning and is based on gene expression data of 25 different cancer types from The Cancer Genome Atlas (TCGA). We evaluate two versions of this approach, one trained on each cancer type separately and the other trained on pan-cancer data comprising all 25 cancer types, and find that the pan-cancer approach yields improved performance over the single-cancer approach. Furthermore, we evaluate the pan-cancer approach on additional molecular data types, including mutations, copy number variations, and protein expression data, and identify gene expression as the most informative data type. To assess the biological plausibility of the gene expression-based pan-cancer survival prediction approach, we apply network propagation to gene weights derived from the survival prediction model and infer a pan-cancer survival network comprising 103 genes. These 103 genes are most significantly enriched for the tumor microenvironment, which has been associated with cancer progression, metastasis, and response to therapy, validating the biological plausibility of our survival prediction approach.

Furthermore, we explore the potential of transfer learning for cancer survival prediction. To this end, we pre-train neural networks for cancer survival prediction, but also for related tasks such as tissue type and age prediction. We then transfer the learned knowledge to cancer survival prediction on independent datasets from TCGA, as well as substantially smaller cancer studies. We find that transfer learning can indeed improve cancer survival prediction, although the benefit of transfer learning may depend on the size and characteristics of the datasets used.

Zusammenfassung

Krebs ist eine der häufigsten Todesursachen weltweit und die zweithäufigste Todesursache in Deutschland. Das vorrangige Ziel von Krebstherapie ist es, die Sterblichkeit zu reduzieren und das Überleben von Patienten zu verbessern. Die Wahl der Therapie wird jedoch stark von der Prognose des Patienten beeinflusst, was die Bedeutung von Krebsüberlebensvorhersage als Mittel zur Quantifizierung des Patientenrisikos und zur Einschätzung der Prognose hervorhebt.

Diese Dissertation stellt einen Ansatz zur Vorhersage des Überlebens von Krebspatienten vor, der XGBoost Tree-Ensemble-Learning nutzt und auf Genexpressionsdaten von 25 verschiedenen Krebsarten aus The Cancer Genome Atlas (TCGA) basiert. Wir evaluieren zwei Versionen dieses Ansatzes, wobei in der einen Version für jede Krebsart separat und in der anderen auf Pan-Krebs-Daten von allen 25 Krebsarten trainiert wird, und stellen fest, dass das Pan-Krebs-Training zu besseren Ergebnissen führt als das Training für einzelne Krebstypen. Außerdem evaluieren wir den Pan-Krebs-Ansatz auf zusätzlichen molekularen Datentypen, einschließlich Mutationen, Copy Number Variations, und Proteinexpressionsdaten, und identifizieren Genexpression als den informativsten Datentypen. Um die biologische Plausibilität des auf Genexpression basierenden Pan-Krebs-Ansatzes zu untersuchen, wenden wir Network Propagation auf aus dem Vorhersagemodell abgeleitete Gengewichte an und leiten ein 103 Gene umfassendes Pan-Krebs-Überlebensnetzwerk ab. Diese 103 Gene sind angereichert für die Mikroumgebung des Tumors, die mit Krebsfortschritt, Metastasierung und dem Ansprechen auf Therapien assoziiert ist, was die biologische Plausibilität unserer Vorhersagemethode bestätigt.

Darüber hinaus untersuchen wir das Potenzial von Transferlernen für die Vorhersage von Krebsüberleben. Dazu trainieren wir zunächst Neuronale Netze für die Vorhersage von Krebsüberleben, aber auch für verwandte Aufgaben wie die Vorhersage von Gewebsarten und Alter. Dann übertragen wir das gelernte Wissen auf die Vorhersage von Krebsüberleben für unabhängige Datensätze von TCGA, aber auch aus wesentlich kleineren Krebsstudien. Wir stellen fest, dass Transferlernen tatsächlich die Vorhersage von Krebsüberleben verbessern kann, obgleich der Nutzen von Transferlernen von der Größe und den Eigenschaften der verwendeten Datensätze abhängen kann.

List of Publications

Bouabid, C., Rabhi, S., **Thedinga, K.**, Barel, G., Tnani, H., Rabhi, I., ... & Guizani-Tabbane, L. (2023). Host M-CSF induced gene expression drives changes in susceptible and resistant mice-derived BMdMs upon Leishmania major infection. *Frontiers in immunology*, 14, 1111072. DOI 10.3389/fimmu.2023.1111072.

Prasse, P., Iversen, P., Lienhard, M., **Thedinga, K.**, Herwig, R., & Scheffer, T. (2022). Pre-Training on in vitro and fine-tuning on patient-derived data improves deep neural networks for anti-cancer drug-sensitivity prediction. *Cancers*, 14(16), 3950. DOI 10.3390/cancers14163950

Thedinga, K., & Herwig, R. (2022). Gradient tree boosting and network propagation for the identification of pan-cancer survival networks. *STAR protocols*, 3(2), 101353. DOI 10.1016/j.xpro.2022.101353.

Thedinga, K., & Herwig, R. (2022). A gradient tree boosting and network propagation derived pan-cancer survival network of the tumor microenvironment. *Iscience*, 25(1). DOI 10.1016/j.isci.2021.103617

Prasse, P., Iversen, P., Lienhard, M., **Thedinga, K.**, Bauer, C., Herwig, R., & Scheffer, T. (2022). Matching anticancer compounds and tumor cell lines by neural networks with ranking loss. *NAR Genomics and Bioinformatics*, 4(1), lqab128. DOI 10.1093/nargab/lqab128

Selbstständigkeitserklärung

Name: Thedinga
Vorname: Anna Kristina

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Dissertation wurde in gleicher oder ähnlicher Form noch in keinem früheren Promotionsverfahren eingereicht.

Mit einer Prüfung meiner Arbeit durch ein Plagiatsprüfungsprogramm erkläre ich mich einverstanden.

____________________ ________________________________

Datum Unterschrift