Deep Learning for Transcriptomics and Proteomics

Ayse Berceste Dincer

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2022

Reading Committee:

William Stafford Noble, Chair

Walter Ruzzo

Sewoong Oh

Program Authorized to Offer Degree:

Computer Science and Engineering

University of Washington

Abstract

Deep Learning for Transcriptomics and Proteomics

Ayse Berceste Dincer

Chair of the Supervisory Committee:

William Stafford Noble
Department of Genome Sciences

Improvements in sequencing technologies increased the availability of omics data, such as transcriptomics and proteomics, providing information about various molecular mechanisms from complementary angles. These measurements can be key to gaining a better understanding of phenotype-genotype associations. Machine learning has great potential to capture the relevant signals from these datasets; however, the inherently complex nature of the measurements, where the signals of biological interest are entangled with technical and other biological factors, makes it difficult to apply these methods directly.

Our goal in this thesis is to address the fundamental challenges associated with transcriptomics and proteomics data hindering the application of machine learning models. Specifically, we tackle (1) high dimensionality, i.e., higher number of features than samples, (2) batch effects and confounders, i.e., signals introduced by technical or biological artefacts, and (3) experimental noise and bias, i.e., inaccuracies in measurements. To solve these problems, we develop three novel deep learning approaches: DeepProfile, AD-AE, and Pepper.

DeepProfile is an ensemble of unsupervised neural network models trained to learn lower dimensional embeddings, effectively reducing the dimensionality and complexity of gene expression profiles. By integrating expression profiles from different sources and adopting an interpretable framework, we generate embeddings to investigate cancer mechanisms. AD-AE disentangles the confounding sources of biological or

technical variance and the biological signals of interest. Our model consists of an unsupervised neural network to learn lower dimensional embeddings and an adversarial predictor to eliminate confounders. The resulting deconfounded representations improve accuracy of downstream prediction models and can be successfully transferred across domains. Pepper focuses on proteomics measurements and aims to reduce the effects of sequence-induced bias for the accurate quantification of proteins. We incorporate our biological hypothesis into the loss functions of our neural network approach to predict and correct for sequence-induced bias. This results in reduction in quantification bias as well as an increase in the correlation between gene and protein expression.

We demonstrate that each of these deep learning models can generate more informative and interpretable versions of our datasets. The resulting representations or the denoised measurements facilitate the application of machine learning techniques for the investigation of phenotypic variation and cellular mechanisms, which we hope will lead to a better understanding of underlying biology.

Acknowledgements

I want to thank my advisor, Bill, for helping me throughout the most difficult times of my Ph.D. and being a great advisor and mentor. I also would like to thank all my committee members for agreeing to be on my committee and for their help, suggestions, and support. I want to thank all the members of the Noble Lab and AIMS Lab for the useful discussions and feedback they provided. I thank all my co-authors, collaborators and mentors for their help and support. I am also grateful for the amazing CSE advising team. I was lucky to have an amazing family and friends who have been very supportive throughout my Ph.D. journey, I am thankful to have all of you in my life.

DEDICATION

To my family

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of transcriptomics and proteomics

The entire hereditary material of an organism is called its genome, the set of instructions that allow the organism to function. Within the cells of an organism lies DNA (deoxyribonucleic acid), a sequence written in a language of 4 nucleotides (A, T, C, G) with a length of 3 billion nucleotides in humans. The DNA is the blueprint of all operations carried out in the lifetime of a cell and organism. This happens through two key processes: *transcription* and *translation*, where the DNA sequence is used to produce proteins, the basic units of action in a cell. Proteins take part in essentially all cellular functions, including cell division, cell structure, and defense.

Specifically, certain regions in the DNA sequence, called *genes*, are expressed through transcription, where the DNA sequence is used as a template to generate RNA (ribonucleic acid). Messenger RNAs (mRNAs), one type of RNA molecule, are intermediate molecules made up of 4 possible nucleotides (A, U, C, G) that code for proteins. Each codon, i.e., each mRNA subsequence of length 3, codes for an amino acid, the basic unit of a protein. There are 20 different types of amino acids in nature and each protein consists of a sequence of these amino acids which go through folding operations to end up in a complex 3D structure and take its role(s) in the cell.

This overall flow of information from DNA to protein is called *the central dogma*, where the DNA is transcribed into RNA which is then translated into proteins. While the central dogma is an oversimplification

and the flow of information can be in multiple directions, this fundamental understanding of how the genetic material determines phenotype has led researchers to investigate the genome itself to understand various phenotypes, including diseases [1]. The Human Genome Project (HGP) was one of the major efforts towards a better understanding of the genome, resulting in successful sequencing of the entire human genome [2]. After the HGP, many studies and technological improvements followed in an attempt to increase our understanding of the genome. Another big step was the introduction of next generation sequencing technologies (NGS) that allowed higher throughput sequencing with reduced cost [3]. These high-throughput technologies increased the availability of sequencing data and enabled the analysis of the genome at various levels.

One term used to represent the high-throughput study of all these biological molecules is *omics* [4]. There are many different types of omics measurements, e.g., genomics, epigenomics, transcriptomics, proteomics, metabolomics, each one focusing on investigating the genetic material at a different stage. Genomics focuses on the DNA sequence itself and identifying genetic variants along the DNA while epigenomics deals with the overall structure of the DNA and modifications to the DNA, such as methylation and histone modifications [4]. Transcriptomics studies mRNA molecules to understand the expression patterns of gene transcripts [4]. The standard output of a transcriptomics experiment is a set of gene expression profiles, a matrix of abundances of transcripts for a set of samples. Going one step further, proteomics studies the proteins themselves and their interactions. A set of protein expression profiles, i.e., matrix of abundances of proteins for a set of samples, can be obtained from some types of proteomics experiments.

Although each of these technologies provides essential information for understanding the genome, in this thesis, we aimed to study the units of heredity and action: *genes* and *proteins*. Examining the expression and abundance of genes and proteins across the genome, across different individuals, tissues, cells, or different time points can reveal unique insights into phenotypic variation. Since proteins take part in many cellular functions, studying proteins, as well as the genes which code for proteins, is essential. Proteomics provides us with qualitative and quantitative information about the proteins; however, it is difficult to measure and sequence proteins, which makes it challenging to generate proteomics data. On the other hand, transcriptomics data is high-throughput and relatively easy to obtain; however, due to complications in the transcription and translation processes, the expression of transcripts is only a proxy for the abundance of

actual proteins. Although these two types of experiments measure highly relevant molecules, mRNAs and proteins, the correlation between gene and protein expression is not very strong due to both technical limitations and biological factors [5]. Considering these discrepancies, we view gene and protein expression as complementary omics measurements, providing us with different levels of information on cellular function. We focus on transcriptomics and proteomics measurements with the goal of developing techniques to extract biologically relevant signals from the data to enable a better understanding of cellular function, gene and protein networks, and disease mechanisms.

1.2 Challenges associated with learning from omics data

While gene and protein profiles have enormous potential, it is not straightforward to use these measurements to explain phenotypic diversity. These are complex datasets, with experimental noise and technical/biological confounders involved, containing unrelated signals entangled with signals of biological interest. One common way of disentangling these signals and learning meaningful patterns from them is using machine learning techniques. The ability of machine learning (ML) models to learn from data, capture generalizable patterns, and reduce the complexity of the original measurements makes them essential tools to deal with these complex datasets. Gene expression data is used with ML models to predict different disease phenotypes [6; 7; 8; 9; 10] and detect biomarkers [11; 12; 13; 14; 15]. Similarly, ML algorithms are used to relate proteomics measurements to phenotype and identify biomarkers [16; 17; 18; 19] as well as to classify patients [20; 21].

Even though these studies are promising, there are still fundamental challenges associated with these transcriptomics and proteomics measurements that make it hard to apply machine learning models out-of-the-box and gain meaningful insights. The prevalent problems we focus on in this thesis are (1) high dimensionality, (2) confounders and batch effects, and (3) experimental noise and biases. Each chapter in this thesis attempts to address one of these problems.

The most common challenge associated with learning from transcriptomics or proteomics data is high dimensionality, i.e., we have a higher number of features than samples. Even though the amount of available data is increasing rapidly with the advance of NGS technologies, training data can still be limited, especially when focusing on a certain disease. Expression profiles record the abundance of each transcript (or pro-

tein), providing us with as many as 50,000 features. This high dimensionality is problematic because it is difficult to learn generalizable associations between thousands of features using a limited set of samples. Furthermore, proteins and genes are highly correlated entities that form regulatory networks. These complex relations between the features also constitute a challenge to applying machine learning techniques to differentiate the relevant biomarkers from the correlated genes/proteins.

The second challenge we address is confounders and batch effects. Due to the complexity of the transcription and translation processes, gene and protein expression measurements are highly confounded with technical and biological factors. One common confounding signal is batch effects, discrepancies between measurements caused by experiments being carried out in multiple batches or in different experimental conditions. Batch effects introduce variation across samples in a dataset and can shadow true biological variation, making it hard to differentiate relevant signals from experimental artefacts. Furthermore, datasets from different studies are commonly combined to increase sample size which leads to even more significant batch effects and dataset biases. Besides batch effects, these omics measurements might also contain biological confounders, e.g., sex, age, medications used, entangled with signals of interest.

The third major problem is experimental noise and biases in measurements. Since both transcriptomics and proteomics try to measure abundances of small molecules with constantly changing characteristics, experimental noise is very common. Such limitations in measurement precision and lack of consistency across experiments lead to skewed or noisy measurements. Not accounting for these experimental biases might result in failure or lack of generalizability of machine learning models.

1.3 Developing deep learning techniques for representation learning and denoising of transcriptomics and proteomics measurements

In this thesis, we develop and apply machine learning techniques to address these limitations associated with transcriptomics or proteomics measurements. We aim to learn meaningful representations, disentangle relevant and confounding signals, and denoise measurements to extract key patterns from the data. By developing novel machine learning techniques that directly target each of these limitations, we translate these invaluable measurements into denoised, deconfounded, easier-to-interpret versions that are more suitable

for downstream analysis. We also demonstrate how the generated datasets allow prediction of relevant phenotypes successfully and detecting signals that provide new insights into biology.

To address these three challenges, we rely on deep neural networks. Deep neural networks are a subclass of machine learning models consisting of layers of hidden nodes, trained to learn the association between provided input and output features to minimize the defined loss function. We choose deep neural networks to address the problems associated with transcriptomics or proteomics data for two main reasons: (1) ability to learn non-linear patterns and (2) ability to customize architecture and loss function. While there are many machine learning algorithms that rely on linear representation of the input features, the interactions among genes and proteins are quite complex and are difficult to represent with simple models. Modeling non-linear functions has the potential to reveal underlying biological mechanisms. The ability to modify the architecture and customize the loss function gives us the opportunity to implement our custom metrics, integrate different networks, and incorporate biological hypotheses into the models.

Accordingly, we introduce three methods in this thesis: DeepProfile, AD-AE (Adversarial Deconfounding Autoencoder), and Pepper. All these methods are deep neural network models developed to address one of the fundamental problems listed above. By learning biologically relevant representations of data, eliminating confounders, or reducing bias in measurements, our goal is to provide researchers with tools to generate more informative versions of their data.

The first method, DeepProfile, focuses on transcriptomics data and tackles high dimensionality. DeepProfile is an unsupervised neural network, i.e., a network trained without the supervision from labels, that encodes high-dimensional gene expression data into an informative lower dimensional space. By reducing the dimensionality of the data, i.e., the number of features, DeepProfile aims to reveal biological signals hidden within high dimensional structure. In this approach, we make use of publicly available expression profiles with no labels, e.g., survival status or cancer subtype, and integrate thousands of samples into an informative representation. We then transfer the learned representation to new datasets and predict cancer patient survival. We also integrate explainability techniques into the model to understand how each gene contributes to the representations we learn. Having these interpretable latent spaces from different cancer types allows us to identify cancer-common and cancer-specific signatures as well as their associations with survival and mutation.

Similar to DeepProfile, AD-AE focuses on transcriptomics data and addresses confounding factors and batch effects. Reducing the dimensionality of a gene expression matrix can reveal insights; however, the experimental and biological artefacts present in the data can make it hard to generate informative lower-dimensional representations. AD-AE aims to learn representations free of these artefacts. It tackles this challenge by incorporating two networks to encode biologically relevant transcriptomics signals and eliminate confounders at the same time. AD-AE can successfully remove unwanted signals from expression datasets with different confounders, including batch effects as well as biological confounders. The deconfounded representation of AD-AE also increases the accuracy of disease phenotype prediction tasks as well as enabling transfer across different datasets.

While DeepProfile and AD-AE methods focus on gene expression, Pepper is designed to work with protein expression. Pepper targets biases that occur in a proteomics experiment and lead to skewed measurements affecting the accuracy of downstream analysis tasks. Our goal is to develop a general preprocessor for proteomics data to reduce bias that arises from the biochemical behavior of distinct peptide sequences. Since we do not have reference measurements to infer the biases, we incorporate our biological hypothesis on how protein quantification should work into the loss function of our neural network model. By adopting a convolutional neural network architecture and training from sequences directly, we build a generalizable model that can adjust abundances in any proteomics experiment. These corrected abundances can reduce the measurement error as well as detecting the biochemical features associated with experimental bias. Furthermore, Pepper improves the correlation between gene and protein expression measurements, highlighting the role of reducing technical noise in improving measurement precision.

1.4 Outline of dissertation

This thesis focuses on developing neural network approaches to address the fundamental challenges associated with transcriptomics and proteomics: high dimensionality, batch effects and confounders, experimental noise and biases. We achieve this by introducing three novel approaches: DeepProfile, AD-AE, and Pepper. We dedicate one chapter to each of these three studies. Chapter 2 introduces the DeepProfile ensemble learning algorithm and demonstrates its ability to improve cancer patient survival prediction accuracy. We then illustrate how we use the learned expression embeddings to identify cancer-common/specific signals and

their association with survival patterns. Chapter 3 describes the AD-AE neural network model and training algorithm followed by the discussion of its ability to increase phenotype prediction accuracy by learning confounder-free lower dimensional embeddings. Chapter 4 introduces Pepper, which integrates our biological hypothesis into the model and demonstrates Pepper's ability to minimize quantification bias as well as highlighting the biological factors contributing to bias. Chapter 5 concludes the thesis with a discussion of the contributions and future work.

Chapter 2

Deep profiling of a compendium of expression data from 18 human cancers

The work presented in this chapter is adapted from the following manuscript, which is currently under revision:

Ayse B. Dincer, Joseph D. Janizek, Safiye Celik, Mikael Pittet, Kamila Naxerova, Su-In Lee. A deep profile of gene expression across 18 human cancers.

Previous version of the manuscript is available in *bioRxiv* [22]:

Ayse B. Dincer, Safiye Celik, Naozumi Hiranuma, and Su-In Lee. DeepProfile: Deep learning of cancer molecular profiles for precision medicine. *bioRxiv*: 278739; doi: https://doi.org/10.1101/278739.

2.1 Background

Gene expression profiles are the reflections of a complex network of underlying cellular and molecular processes. These profiles are obtained using transcriptomics sequencing technologies, which aim to detect whether transcription occurs for each gene and record the relative abundances of the transcripts [23]. The abundances of the genes are commonly used as a proxy for protein abundances, due to both the efficiency of mRNA sequencing and the difficulty of measuring proteins themselves, as discussed in Chapter 1. A range

of different transcriptomics technologies exist to extract the mRNA molecules and identify and quantify them. One of the most common methods used today is next generation sequencing-based RNA-Seq which is preceded by microarray technology that relies on a set of predefined sequences [23]. In both experiments, mRNA is transcribed and extracted, and the complementary cDNA sequences for the mRNAs are recorded. As explained in Lowe *et al.* [23], quantification and detection are done through fluorescent labeling in microarray experiments. The technology relies on a sorted microarray containing the complementary cDNAs for all the target sequences. The extracted cDNA sequences are then captured and sorted in this microarray where the fluorescent intensities allow detection of expression along with relative abundances.

While microarray technology allows transcriptomics sequencing, it is limited to detecting a set of predefined sequences only, which might result in low coverage. In contrast, RNA-Seq uses high-throughput sequencing, allowing to sequence any transcript with the help of a reference genome. Similar to microarray, RNA-Seq extracts the mRNA and records complementary cDNA sequences. Then these cDNAs are sequenced using short read sequencing and aligned to a reference genome [23]. The count of reads aligned to the reference are used to detect the existence or abundance of transcription.

The generated transcriptomic profiles are used in a wide range of applications. One of the prevalent uses is 'disease diagnosis and profiling' [24]. Gene expression profiles from cohorts of patients are also used for diagnostic and therapeutic biomarker detection [25]. These abundances can further reflect the change in mechanisms in response environmental factors or external stimuli, such as drugs [26]. Many other applications can be listed including, but not limited to gene function annotations and study of non-coding RNAs [23], gene regulation, gene-gene interaction networks, and cell type differentiation.

With the advance of NGS technologies, the number of available transcriptomics datasets also increases. Major databases like Gene Expression Omnibus [27] or Expression Atlas [28] contain data from thousands of studies. There are also different consortia with efforts towards gaining a better understanding of the genome such as GTEx consortium [29] with 7,382 RNA-Seq samples from 948 healthy tissue donors or TCGA [30] with over 20,000 samples from 33 different cancer types profiled. These databases and consortia provide researchers with immense opportunities to investigate cellular mechanisms at the gene level and relate them to phenotypic variation.

While most of the recent efforts is towards generating more RNA-Seq data, there is still an existing

accumulation of microarray datasets. In this chapter, we aimed to make use of underutilized microarray profiles to study cancer mechanisms and gain a better understanding of the commonality as well as the heterogeneity of cancer. Public repositories provided us with many studies to gather samples from, however, our sample size is still not as high as our entire set of features, transcriptomes. This high dimensional nature of the data makes it difficult to use machine learning techniques to make phenotype predictions since it is highly challenging to learn robust associations between genes and disease phenotypes with a restricted set of biological samples. Furthermore, genes are highly correlated entities that regulate each other in complex networks. Decoding these interactions is essential for identifying biomarkers and understanding various roles genes take in disease.

We address this crucial problem of high dimensionality by learning lower-dimensional representations, i.e., latent spaces for gene expression profiles. Latent space learning is a key step to extract meaningful biological information from expression profiles and reduce the dimensionality of the data for downstream tasks, such as prediction of phenotypes. It projects high-dimensional input variables, i.e., genes, into a latent space consisting of a smaller set of latent nodes such that information present in the original space is largely preserved. Learned latent nodes represent sources of genome-wide expression variation across samples, for example large-scale transcriptional programs that define intrinsic disease subtypes or reflect extrinsic stimuli such as hypoxia or treatment pressure.

One key limitation of commonly used latent space learning approaches for expression data, such as principal component analysis (PCA), is that they can only extract latent nodes that have linear relationships with gene expression levels, while biological interactions can be more complex. The artificial intelligence field has achieved notable success in unsupervised learning by using deep neural networks that can capture highly complex relationships between variables. It has been shown that the latent nodes extracted by deep unsupervised learning approaches from image data represent high-level features that are intuitively important, for example: skin color, age, and gender from face images [31], lighting and room geometry from scene images [32], and rotation and size of an object from 3D images [33]. These informative and complex image features cannot be captured by models limited to learning linear feature interactions [34]. Similar to images, expression profiles can be decomposed into high-level features encoding the complex relations between genes. Thus, applying these advanced deep learning techniques to expression data can lead to

improved identification of molecular processes underlying expression profiles.

So far, three challenges have impeded the successful application of deep latent space learning approaches to cancer expression data. First, deep learning has a high risk of overfitting when not provided with large sample numbers. Second, the non-deterministic nature of the learning process impairs the robustness of the learned latent spaces. Each run of the neural networks, even using the same architecture, results in different models with different parameters which makes it difficult to capture consistent signals. Third, a neural network is a 'black box' by nature: since it is not clear how the model uses gene inputs to generate a latent node, biological interpretation of nodes is problematic. Here, we present DeepProfile, a framework that learns statistically robust and interpretable latent spaces from gene expression data. To address the non-deterministic nature of the deep learning process and capture robust latent spaces, we develop an ensemble approach to integrate the results of hundreds of deep unsupervised models generated from different random starting points and latent space sizes. Furthermore, we enable biological interpretation of each latent node by linking it to a set of genes and the pathways enriched in the gene set by adopting explainable artificial intelligence techniques.

We apply DeepProfile to a compendium of gene expression datasets from 18 human cancers available through a publicly available expression data repository. Our pan-cancer framework consists of four components (Figure 2.1): (1) Data Collector, which obtains all available expression datasets to train our models on, (2) Deep Learner, which adopts the novel ensemble approach of DeepProfile to learn cancer-specific latent spaces, (3) Interpreter, which biologically characterizes each latent node by mapping it to genes and pathways, and (4) Pan-Cancer Analyzer, which investigates all the cancer latent spaces to identify biologically relevant signals. We demonstrate that DeepProfile can learn robust cancer-specific latent spaces and biologically interpret each latent node. We then use these interpretable nodes to investigate the genes that played an important role in all cancer types and the genes specifically important for one cancer type as well as comparing the nodes to normal tissue expression embeddings. We further associate DeepProfile nodes with patient and tumor characteristics and identify survival or mutation associated gene sets.

Previous studies conducted pan-cancer analyses with various approaches ranging from learning co-expression networks or differential expression [35; 36; 37; 38; 39; 40; 41; 42] to adopting deep unsupervised learning approaches [43; 44]. Here, we built on these approaches by addressing the problems associated with

deep learning such as lack of robustness and interpretability. By employing a novel and interpretable deep learning ensemble approach on publicly available expression profiles, we seek to capture unique gene-gene relationships and detect robust and consistently captured signals related to transcriptomic heterogeneity. We leverage a variety of data sources such as GEO [27], TCGA [30], and GTEx [29] and integrate different data modalities such as clinical and mutational features. This rich resource of robust cancer-specific deep embeddings and biological characterization of the latent nodes enables us to examine cancer transcriptomic signals from a new angle and investigate their associations to various cancer phenotypes.

Using DeepProfile framework, we examined the genes and pathways identified as the major components of the expression matrices across all 18 cancer types and detected that the universally important genes are master switches of inflammatory state, activating immune response and antigen presentation within the tumor microenvironment. We further observed that the cancer specific genes and pathways were majorly determining the molecular disease subtypes or grades of differentiation within a tissue category. Associating DeepProfile cancer-specific nodes with patient prognosis and mutation frequencies allowed us to detect a transcript-level association between cell cycle with mutation frequency, and highlighted MHC class II antigen presentation and mismatch repair pathways to be important determinants of survival across multiple cancer types.

2.2 Results

2.2.1 DeepProfile learns robust latent spaces for 18 cancer types

Because highly expressive models such as deep neural networks tend to overfit when the sample size is small, we obtained all expression datasets from the most common microarray platforms available at the time for 18 human cancers through the Gene Expression Omnibus (GEO) [27] (Figure 2.1, Table 2.1, see 2.4.1), which amounts to 50,211 samples from 1,098 datasets. DeepProfile takes the expression data matrix as input and maps the data into the embedding space represented by a set of latent nodes using a novel ensemble approach for the variational autoencoder (VAE) [45]. The VAE is a special type of deep neural network that compresses high-dimensional data, i.e., tens of thousands of genes, into a low-dimensional embedding with minimal loss of information. More specifically, two neural networks – (i) the encoder that

models the relationship between input nodes and latent nodes in the embedding space, and (ii) the decoder that models the relationship between the latent nodes and the reconstructed input nodes– are trained such that the reconstructed input data are close to the input gene expression data (see 2.4.2).

VAE is a unique model that can discover non-linear relations among genes to reflect the true nature of gene interactions. However, it is not straightforward to apply it to expression data. Inherently, neural networks suffer from variability of learned models across different random initializations due to its intrinsic nature of non-convexity. This means that a conventional learning algorithm for VAE can result in a model that is different in every trial, which hinders the inference of robust biological signals. To improve the robustness, we present an ensemble of variational autoencoders, a new way to combine the learned models from different random runs and latent dimension sizes (Figure A.1, see 2.4.2). This approach integrates signals from hundreds of different embedding spaces into one information-rich space.

After learning these cancer-specific embedding spaces, we biologically characterize each latent node by mapping it to genes and pathways. The interpreter component of DeepProfile (Figure 2.1) achieves this by linking each latent node to genes using a principled *feature attribution* method [46] to quantify how much each latent node's value is attributed to input feature nodes (Figure 2.1 and A.2). In particular, for each latent node, DeepProfile produces a list of gene attribution scores, which indicate the relevance of each gene to that node and uses the top-listed genes for pathway enrichment tests which provide pathway-level attribution scores (see 2.4.2). We then performed a pan-cancer analysis based on the embeddings and gene relevance scores of each latent node (Figure 2.1). The trained DeepProfile model can also be applied to new cancer gene expression dataset to reduce its dimensionality (Figure A.2), as demonstrated in the next section.

2.2.2 DeepProfile's encoding is a powerful dimensionality reduction approach

We begin with evaluating DeepProfile's ability to preserve biologically relevant signals when encoding high-dimensional gene expression data by comparing with existing dimensionality reduction methods. Our evaluation strategy is to use low-dimensional embeddings to predict patient survival, which is a common analysis task of cancer gene expression data and is a clinically important problem. We used RNA-seq data from The Cancer Genome Atlas (TCGA) [30], which were not used for training of DeepProfile to evaluate the performance on independent data (Figure A.2, Table 2.2, see 2.4.3).

Using the trained DeepProfile models, we encoded the TCGA expression data into embeddings for TCGA samples. We then use these embeddings as features to predict 5-year patient survival (see 2.4.3). Then we compare the survival prediction accuracy of DeepProfile against multiple linear (random projections, PCA, ICA) and deep learning (standard autoencoder and denoising autoencoder) methods across all cancer types. DeepProfile significantly outperformed the alternative methods, including one of the most commonly used linear models, PCA, as well as other deep learning models, in 82% of the test cases (41 tests cases out of 50, proportions z-test P = 1.9×10^{-19}) (Figures 2.2A and A.3A). DeepProfile's superior performance indicates that it is capable of extracting more biologically meaningful embeddings relevant to prognosis than alternative approaches. Our result also highlights that DeepProfile can be successfully applied to RNA-Seq expression profiles despite being trained on microarray expressions. This is further supported by the high correlation between DeepProfile embeddings generated from microarray and RNA-seq expressions (Figure 2.2B).

Given that the dimension of the latent embedding affects what the VAE captures (e.g. 10- and 100-dimensional latent spaces will capture different levels of variation among cancer samples and thus, may detect a distinct set of biological signals), we investigated whether ensembling VAE embeddings with various dimensionalities can enhance the power of dimensionality reduction. Indeed, we found that DeepProfile outperformed all these different dimensional VAE embeddings in terms of 5-year survival prediction accuracy for 75% of the test cases (42 tests cases out of 56, proportions z-test p-value: 7.8×10^{-6}) (Figure 2.2C and A.3B).

These findings indicate that the ensemble of VAE embeddings preserves and integrates signals coming from a diverse set of VAE models and captures richer information, which should eventually help to better understand the underlying biology.

2.2.3 DeepProfile can learn biologically interpretable latent nodes enriched for a wide set of pathways

It is desirable for latent nodes to be biologically interpretable. DeepProfile provides gene relevance scores for each latent node, which enables a standard enrichment test to assess the statistical significance of the overlap using the Fisher's exact test between the top-scoring genes and predefined gene sets, such as KEGG

[47], BioCarta [48] and Reactome [49] databases. We compared the average number of pathways captured by nodes of DeepProfile and other dimensionality reduction methods (see 2.4.3). DeepProfile nodes were significantly enriched for a larger number of pathways compared to alternative methods (88 tests cases out of 90, proportions z-test P = 6.3×10^{-208}) (Figure 2.3A).

Further, when we focused on the pathways known to be dis-regulated in cancer, using the oncogenic signature gene sets, DeepProfile again outperformed the other methods in terms of number of gene sets captured (88 tests cases out of 90, proportions z-test p-value: 6.3×10^{-208}) (Figure 2.3A). This means that DeepProfile not only captures more pathways but also identifies the pathways relevant to cancer.

If a latent node is not associated with any known pathway, it would be hard to biologically characterize that node, and thus decreases the biological interpretability. We show that DeepProfile has the lowest number of such nodes (Figure 2.3B and A.4, see 2.4.3). Further, we show that, for varying p-value threshold, a higher percentage of DeepProfile nodes were biologically annotated based on pathways compared to other methods (Figure 2.3C and A.5, see 2.4.3). These results demonstrate that the unique deep learning ensemble approach adopted by DeepProfile contributes to the improved biological interpretability of the latent nodes.

2.2.4 Universally important genes modulate inflammatory pathways

Using the robustly identified embeddings from 18 cancers and the gene-level and pathway-level interpretation of each latent node, we performed various pan-cancer analyses to interrogate the transcription programs in 18 human cancers and their implications to prognosis.

We began by investigating genes with universally large gene relevance scores to DeepProfile nodes across all cancer types (see 2.4.4). These genes represent dominant gene expression programs that consistently explain significant portions of the transcriptional variance across many different cancers. We found that genes with high average gene relevance scores were primarily involved in immune response regulation and antigen presentation (35 genes out of 100 were immune related, p-value: 9.4×10^{-6}) (Figure 2.4A-C). Given that solid tumors (which constitute the majority of our data) can be infiltrated by immune cells to varying degrees, we hypothesized that genes with top relevance scores reflect the gene expression signatures of various admixing immune cell types. To test this hypothesis, we calculated the significance of the overlap between 108 immune cell signatures (T cells, B cells, neutrophils, and macrophages) [50] and genes with

top DeepProfile attribution scores using Fisher's exact test (see 2.4.4). Surprisingly, we did not find significant overlap for any of the signatures, suggesting that the top genes identified by DeepProfile were not simply transcripts that reflect the relative abundance of immune cells across tumors.

Next, we hypothesized that DeepProfile prioritized genes whose expression was associated with recurrent transcriptional phenotypes in tumor-infiltrating immune cells, such as signatures associated with activation or suppression of immune cell activity. To illustrate this concept, consider the gene with the highest average attribution: the alpha subunit of the interleukin 10 receptor (IL10RA). IL10RA scored among the top 1% of genes in 78% of cancer types (14 out of 18 cancers, top 10% in all 18 cancer types), indicating that DeepProfile consistently ascribed high explanatory power to this gene, regardless of tissue context (Figure 2.4A). IL10RA has been described as a 'master switch' regulating the balance between pro- and anti- tumor inflammation [51]. Transcript levels of IL10RA do not just reflect the presence or absence of IL10RA expressing immune cells, they are additionally predictive of the several thousand genes regulated by IL10RA [52], explaining the large role that this gene plays in DeepProfile's embeddings.

Next, to test the hypothesis that universally high-scoring DeepProfile genes were enriched for transcripts that, like IL10RA, modulate the transcriptional phenotype of immune cells, we quantified the enrichment of cell surface receptors among genes with top attribution scores. We reasoned that cell surface receptors are enriched for proteins that relay extra-cellular signals and thus have the potential to regulate immune cells' transcriptional phenotypes. We collected gene sets containing cell surface proteins and receptors from the Cell Surface Protein Atlas (CSPA) [53], the UniProt database [54], and the Gene Ontology (GO) [55]. We found highly significant overlap between these gene sets and genes with top average DeepProfile attribution scores across all cancers (Figure 2.4D). Importantly, PCA did not recover these cell surface proteins and receptors (Figure 2.4D), indicating that DeepProfile's ability to identify non-linear relationships is essential in capturing signals playing key roles in cancer.

In addition to IL10RA, DeepProfile top attributions contained many lesser known but potentially important genes that consistently played a large role in the embeddings of most cancer types. These included CD53, an immune-cell specific tetraspanin [56], EVI2B, a gene that controls the differentiation status of granulocytes [57] and TYROBP, an adaptor protein that in association with various receptors mediates immune cell activation [58](Figure 2.4A). As indicated above, none of these genes are likely to reflect the

presence of one particular immune cell type in the tumor microenvironment, as they are broadly expressed by many different cells, but instead may be involved in modulating the transcriptional phenotypes of tumor-resident immune cells.

2.2.5 Universally important pathways include cell cycle, immune system, and oxidative phosphorylation

Next, we investigated pathway-level information captured by DeepProfile by studying the relationship between the embeddings and signaling pathways in the KEGG, BioCarta and Reactome databases (see 2.4.4). We considered a pathway to be significantly enriched in a given cancer type if it overlapped with an FDR-corrected p-value below 0.05 with any DeepProfile node. We then extracted pathways that were significantly captured in the largest number of cancer types, grouped the universally important pathways by functional category, and sorted the categories by the average number of cancer types. As expected, cell cycle-related gene sets were near-universally important, confirming that differences in proliferative index (also called growth fraction) are a major source of variance across cancer transcriptomes (Figure 2.4E). Two cancer types stood out for a much less pronounced contribution of cell cycle-related gene sets: AML, whose embeddings mainly captured pathways related to the adaptive immune response and thyroid cancer, where the most important pathways were related to mitochondrial function. The two most common types of thyroid cancer (papillary and follicular) are exceptionally slow-growing neoplasms, potentially explaining the relative lack of contribution of cell cycle-related pathways. In AML, growth rates are more difficult to assess [59], but it is possible that the opposite is the case: namely that most patients experience uniformly high growth rates due to the aggressiveness of the disease and its lack of spatial restraint. In both cases, a lack of variation in proliferative fractions across patients would explain why DeepProfile does not detect the cell cycle as an important contributor of variance to these cancers' transcriptomes.

Immune-related pathways, as discussed in detail above, were the third-most frequently captured category (Figure 2.4E) followed by gene sets related to oxidative phosphorylation (OXPHOS), indicating that the distribution of individual tumors on the metabolic continuum between glycolysis and aerobic respiration explains global differences in their gene expression profiles [60]. One other category to emerge as relevant across a large number of cancers were genes related to RNA metabolism and ribosome function. Enrichment

p-values were particularly significant in this category.

2.2.6 DeepProfile can quantify the extent to which each latent node is cancerous or normal tissue specific

We hypothesized that these gene sets were not necessarily identified by DeepProfile because they captured variance related to the presence of different disease subtypes with a tissue of origin, but because they contained genes that are constitutively expressed in a highly correlated manner (in any tissue). To test this hypothesis, we generated DeepProfile embeddings for normal tissue gene expression profiles from the GTEx database [29] (see Table 2.1 for number of GTEx samples and cancer type mappings). We fitted predictor models to differentiate the normal from cancer embeddings which provided a score for each DeepProfile node denoting how successfully it can separate cancer from normal tissue (see 2.4.4). Using DeepProfile pathway-level node attributions, we mapped these node-level scores to pathways to define a cancer-relevance score for each pathway. A high cancer-relevance score indicates that the pathway is specifically important for cancer; it does not show as strong variance in expression in normal tissue as in cancer (Figure 2.4E). We found that in comparison with cell cycle pathways, the cancer-specificity score of the ribosomal gene sets was indeed lower (average cancer-specificity score of 82.39 for cell cycle compared to 63.19 for ribosomal pathways (p-value: 1.6×10^{-17}, Welch's T-test)), indicating that these genes also capture significant variance across normal tissue gene expression profiles. Nonetheless, we note that the degree of biosynthetic activity (as reflected by the expression of ribosomal proteins) has recently been shown to be associated with differentiation state in colorectal cancer [61], raising the intriguing possibility that DeepProfile's capture of ribosomal genes reflects variance in differentiation states across tumor samples within a given tumor type. This may explain why some relatively narrowly defined (and therefore more homogeneous) cancer types such as AML did not show significant enrichment of ribosome-related pathways. We further note that the two near-universally important pathways with the highest cancer-relevance scores were related to protein folding (prefoldin) and focal adhesions (Figure 2.4E). The latter result is consistent with DeepProfile capturing variation in epithelial-to-mesenchymal transition states that exist within a tumor [62].

2.2.7 Cancer-specific genes and pathways define molecular disease subtypes

After studying genes and pathways that were considered universally relevant by DeepProfile, we aimed to identify genes that play an important role in specific cancer types only. We calculated a per-gene cancer specificity score, defined as the difference between the gene percentile score for one cancer type and the highest gene percentile score across all other cancer types (see 2.4.4). High specificity scores indicate that a gene captures a large amount of variance in one particular cancer type but plays a more subordinate role in others.

We found that genes with high specificity scores generally defined dominant subtypes or grades of differentiation within a tissue category (Figures 2.5A). For example, the top breast-specific transcripts were prolactin-induced protein (PIP), a gene predominantly expressed in well-differentiated estrogen receptor-positive tumors [63], FOXC1, a gene that is specifically expressed in basal-like breast cancer [64], and GFRA1, which is specific to the luminal A subtype [65]. To formally test the hypothesis that DeepProfile captured genes that are differentially expressed among breast cancer subtypes, we calculated the overlap between breast cancer-specific genes and PAM50, a gene set that effectively distinguishes between basal-like, normal-like, luminal A, luminal B and HER2-enriched subtypes [66] (see 2.4.5). We obtained highly significant results ($P = 3.8 \times 10^{-3}$). Importantly, a linear model (PCA) was not able to identify subtype-specific genes effectively (P = 1.0, for PAM50 gene set enrichment), indicating that DeepProfile's ability to capture non-linear relationships is essential for learning of biologically meaningful patterns. Similarly, AML-specific genes comprised transcripts that had previously been associated with AML subtypes (e.g. HOXA7, TRH, MYL4, ANK1) [67; 68] and showed significant overlap with list of genes identifying AML subtypes [69] ($P = 4.2 \times 10^{-5}$) while PCA genes failed again (P = 1.0).

In the brain, DeepProfile identified genes that distinguish oligodendrogliomas from astrocytomas (e.g. CNP, [70]) or vary across glioblastoma subtypes (e.g. BCAN, [71]). Thyroid cancer-specific top genes included thyroid peroxidase (TPO) and thyroid stimulating hormone receptor (TSHR). These genes may indicate the presence of well-differentiated thyroid cancers (which to some degree retain the expression profiles of their normal tissue of origin) versus highly undifferentiated cancers which have largely lost the expression of tissue-specific transcripts, in the thyroid cancer data set. To support this hypothesis, we compared DeepProfile thyroid cancer-specific genes with the list of genes shown to be associated with

thyroid cancer subtypes [72]. We observed that the two lists significantly overlapped ($P = 4.4 \times 10^{-10}$) while the same analysis for the thyroid cancer-specific genes discovered by PCA showed no significance ($P = 1.0$). These examples demonstrate how DeepProfile successfully detects genes that differentiate cancer subtypes while a linear model fails capturing these important patterns.

Next, we extracted functional gene sets that DeepProfile recognized as cancer-specific (Figure 2.5B) (see 2.4.4). This approach is potentially more informative than a gene-level view, as it can go beyond the identification of subtype 'marker genes' to reveal coherent pathways that most prominently distinguish cancers from one tissue of origin. Thus, the analysis provides concrete information about the molecular mechanisms driving expression heterogeneity within cancer types. Indeed, DeepProfile assigned highly characteristic molecular processes to each cancer type. For example, top AML-specific pathways were related to porphyrin metabolism and heme biosynthesis. It has been known for more than half a century that leukemic cells show increased heme biosynthesis [73], but little is known about the mechanistic relevance of the porphyrin production pathway in leukemogenesis. Importantly, it was recently shown that MYC-overexpressing leukemic progenitors require porphyrin biosynthesis for self-renewal [74], indicating that this pathway plays a role in driving or facilitating leukemogenesis in a subset of these cancers. Notably, DeepProfile identified this pathway as relevant to AML in an unsupervised manner. As we had previously done in our analysis of genes and pathways that were universally important across cancers, we also calculated 'cancer-relevance' scores (via comparison of matched normal tissue embeddings from GTEx) to determine to what degree a pathway's importance was specific to malignancy. The AML-specific pathway with the highest cancer-relevance score was MHC class II antigen presentation, represented by HLA-DMA, HLA-DRB1, HLA-DMB, HLA-DPA1 and HLA-DPB1 genes. Downregulation of HLA-DPA1, HLA-DPB and HLA-DRB1 in AML has recently been reported at the time of relapse after allogeneic bone marrow transplant and has been interpreted as evidence of graft pressure on leukemic cells [75]. However, the prominent identification of the MHC class II antigen presentation pathway by DeepProfile indicates that heterogeneity in MHC class II protein expression may be a more general disease feature distinguishing subsets of AML.

In brain cancer, lipid transport scored as the most important pathway, with a high cancer-relevance score. Cholesterol is an essential component of myelin, and the brain contains approximately 20% of the

body's total cholesterol [76]. Astrocytes normally produce the majority of the brain's cholesterol, since it cannot be transported across the blood-brain-barrier. In glioblastoma, the brain's normal lipid metabolism is profoundly altered. Glioblastoma cells downregulate cholesterol biosynthesis and depend on exogenous cholesterol uptake for survival [77], making the identification of this pathway by DeepProfile a notable result. The highest cancer-relevance score was achieved by the Sprouty (SPRY) pathway, represented mainly by SPRY1 and SPRY4. These two genes are negative regulators of FGFR signaling, a pathway that has plays an important role in glioblastoma progression and is currently being targeted in clinical trials [78]. These and other examples, such as the identification of an important role for the peroxisome in liver cancer [79], illustrate DeepProfile's ability to extract cancer-specific and biologically meaningful expression patterns from large unstructured data depositories like the Gene Expression Omnibus.

The knowledge of expression subtypes and the pathways defining them is important from a basic science perspective, but from a translational point of view, pathways connected to clinical variables are of particular interest. We therefore set out to develop a rigorous methodology for connecting DeepProfile embeddings to relevant patient and tumor-level characteristics.

2.2.8 Detection of survival- and mutation burden-associated pathways via DeepProfile

The contribution of a pathway to DeepProfile nodes reflects to what degree it captures variance in the primary gene expression data, but it does not reveal whether the pathway relates to variables of clinical interest. We developed a general methodology for connecting pathways to clinical characteristics via DeepProfile nodes (Figure 2.6A, see 2.4.5). We tested the approach by extracting pathways that are relevant to two important patient-level and tumor-level features: survival and tumor mutation burden (TMB). Specifically, we associated each DeepProfile node with survival/TMB and generated p-values denoting the significance of association of each node with the phenotypes. Then, using the pathway-level attributions for DeepProfile nodes, we mapped the node-level phenotype associations to pathway-level associations, and obtained survival and TMB association p-values for each pathway. The same approach can readily be adapted to other variables of interest, for example tumor stage, or grade, or treatment response. The advantages of using DeepProfile nodes (instead of genes or pathways directly) are two-fold: first, DeepProfile embeddings encode the largest sources of variation among cancer samples; thus, the association search space is reduced

to biologically meaningful variables. Second, since each DeepProfile node is a non-linear combination of genes, it has the unique ability to capture complex interactions between genes and phenotypes of interest.

To test the effectiveness of this approach, we first investigated gene sets that DeepProfile recognized to be significantly related to arguably the most important patient-level trait – survival. As in our previous analyses, we first focused on pathways that were associated with survival across all cancer types (Figure 2.6B). Remarkably, in this pan-cancer analysis, the unifying theme of most survival-related pathways was adaptive immunity. High-scoring gene sets included adaptive immune system, MHC class I antigen presentation, antigen processing cross-presentation, B cell receptor signaling, the proteasome pathway and activation of NF-κB (all significantly detected in 5 cancer types). Three pathways stood out for scoring in more than 5 cancer types. These included DNA mismatch repair (6 cancers) – a process that can give rise to large numbers of neoantigens when impaired [80]- and MHC class II antigen presentation, which was the highest-scoring pathway overall (significantly detected in 7 cancer types; the section below will explore these two pathways in more detail).

To provide a contrast and comparison for these results, we next studied pathways with a significant connection to a tumor-level characteristic, namely tumor mutation burden (Figure 2.6C). Interestingly, in contrast to survival- and TMB-relevant pathways were most consistently linked to the cell cycle and included DNA replication, mitotic M-M/G1 phases, mitotic prometaphase, chromosome maintenance and others. The top-scoring TMB-linked pathway was mitotic G2-G2/M phases, significantly detected in 11 out 18 cancers. These results establish a link between a tumor's proliferative activity and its mutation burden. Since DNA replication is a powerful mutagen, this connection is highly plausible and carries interesting implications given the strong interest in TMB as a predictor of immunotherapy response [81].

Analogously to previous analyses, we also studied pathways that were survival- and TMB-related in a cancer-specific manner where we investigated the pathways with highest survival and TMB scores for each cancer type. Again, we found that DeepProfile identified distinct sets of pathways as being relevant to the two different traits. For example, survival-related pathways in brain cancer were dominated by interferon type I and II signaling and MHC class I-mediated immunity, while TMB-related pathways prominently featured cell-cell and cell-matrix interactions themes (Figure 2.6D). In sarcoma, survival-related pathways almost exclusively concerned DNA repair processes (mismatch repair, nucleotide excision repair) and repli-

some function, while TMB gene sets were strongly related to glucose metabolism.

2.2.9 DNA mismatch repair and antigen presentation via MHC class II are common survival-related pathways

We decided to investigate the striking pan-cancer association between survival and DNA mismatch repair and MHC class II antigen presentation in more detail. DeepProfile detects robust correlations between pathways and survival; however, it does not provide a direction for these associations. Therefore, to define this direction, we fitted univariate Cox regression models on the genes in the pathways being investigated; this returned a survival z-score for each gene and cancer type pair (see 2.4.6). A negative z-score means that lower expression leads to better chance of survival whereas a positive z-score means that higher expression leads to a better chance of survival).

Examining the z-scores of DNA mismatch repair genes across all cancers, we found that indeed many of them were strongly correlated with survival (Figure 2.7A), validating the findings of DeepProfile at the primary gene expression level. The direction of the association tended to be negative, in particular for the 6 cancers that scored with statistical significance in the DeepProfile-based analysis (Figure 2.6B). A negative correlation means that lower expression of DNA mismatch repair proteins associates with improved survival. We confirmed this finding further via Kaplan-Meier analyses that yielded consistent results (Figure 2.7B and A.8A) (see 2.4.6). The prognostic relevance of DNA mismatch repair gene expression across a large number of cancers is particularly notable given DeepProfile's identification of the adaptive immune response as a central survival-related pathway hub. Reduced expression of mismatch repair proteins can increase mutability and microsatellite instability [82]. Therefore, increased abundance of neoantigens in tumors with reduced mismatch repair protein abundance may make these tumors more visible to the immune system and thus contribute to the improved survival of patients with low expression of DNA mismatch repair proteins (Figure 2.7C).

Next, we investigated the MHC class II antigen presentation pathway in more detail. We focused on HLA-D genes, because they were top-scoring both in terms of both attribution scores and survival z-scores across all 18 cancer types among all genes included in the MHC class II antigen presentation pathway. In contrast to the DNA mismatch repair z-scores, which showed a negative correlation between expression

and survival across most cancer types, the association for HLA-D expression was bifurcated (Figure 2.7D). Pancreas, kidney, AML and brain showed a strong negative association between HLA-D gene expression and change of survival, while the correlation was positive for most other cancers, most prominently melanoma and uterine cancer. Again, we confirmed these findings via Kaplan-Meier analyses (Figure 2.7E and A.8B).

These results suggested that expression of HLA-D genes in the tumor and/or its environment is beneficial in some cancer types (melanoma, uterine cancer, breast cancer) and detrimental in others (brain cancer, kidney cancer). Since most cancers do not express MHC class II genes (with the exception of AML, where HLA-D expression is associated with an inflamed phenotype and therapy relapse [75]), we wondered which cell type in the tumor microenvironment might be the primary source of the HLA-D transcripts, and by extension, associated with differential survival. Tumor-resident immune cell types that express MHC class II genes include macrophages, dendritic cells and B cells.

To gauge the relative abundance of these cells in the tumor microenvironment, we measured the average percentile score of the signature genes for each cell type, where the most highly expressed gene had a score of 100. We found that of the three cell types, macrophage-specific genes were by far the most abundant signature across all studied cancers, in line with the notion that these cells can be highly prevalent across cancer types [83; 84; 85] (Figure 2.7F). Also, we found that in all cancers, the macrophage signature showed the best correlation with HLA-D expression, further supporting the notion that macrophages are the largest contributors to HLA-D transcript abundance in bulk tumor samples (Figure 2.7G). Considering that macrophages can have divergent functions, ranging from pro-tumor to anti-tumoral [83; 84; 85], we wondered whether the phenotypes of tumor-associated, HLA-D-expressing macrophages might explain the observed bifurcation in the correlation between HLA-D expression and survival. To this end we examined gene transcripts that may reflect macrophage function. Specifically we assessed expression of CD40, CXCL9, CXCL10, CXCL11, SLAMF1, and TNIP3, which are associated with anti-tumor activity, and of CFP, HRH1, NPL, PDCD1LG2, and CFP, which are typically indicative of immunosuppression and tumor promotion [86]. While these genes are not necessarily uniquely expressed by macrophages, the abundance of macrophages (Figure 2.7F) makes them plausible main sources of these transcripts. Examining the relative abundance of the gene transcripts mentioned above revealed that most tumor types expressed both signatures at similar levels (Figure 2.7H). The only large gap, with a large preponderance of immunosuppressive

transcripts, was observed in brain cancer and AML – the two cancer types with the most significant negative association between HLA-D expression and survival ($P = 3.4 \times 10^{-2}$ and $P = 1.6 \times 10^{-1}$, Welch's T test for brain cancer and AML, respectively). We repeated the same test with an extended list of pro- and anti-inflammatory macrophage signatures [87] and again observed a stronger immunosuppressive macrophage abundance in brain cancer ($P = 5.0 \times 10^{-2}$, Welch's T test) (Figure A.8C). The presence of macrophages that are polarized towards an immunosuppressive phenotype might therefore contribute to the strong negative correlation between HLA-D expression and survival in brain cancers and AML. In most other cancer types, HLA-D expression is correlated with improved outcomes.

2.3 Discussion

Gene expression profiles have immense potential to reveal insights into cellular and molecular processes. However, it is not straightforward to extract these signals due to the *high dimensionality* of the data: we have more features, i.e., transcripts, than samples which makes it hard to learn generalizable patterns. Moreover, genes act as parts of complex regulatory networks which complicates deciphering the underlying mechanisms. In this chapter, we addressed this high dimensionality problem associated with transcriptomics measurements by efficiently applying unsupervised deep neural networks to learn biologically-relevant latent spaces.

Our framework, DeepProfile, is an interpretable deep learning ensemble model that projects thousands of genes to a latent space to learn informative high-level transcriptomic features that represent complex, non-linear relationships between genes. We applied DeepProfile to the transcriptomic measurements from 18 human cancers and carried out a pan-cancer analysis to identify cellular and molecular processes reflected by the cancer gene expression profiles. By collecting publicly available cancer microarray datasets from the GEO database and ensembling hundreds of VAE models, we generated robust deep embeddings. Furthermore, we provided explanations for the latent nodes by attributing them to genes and pathways.

To our knowledge, we are the first transcriptomic pan-cancer study adopting an interpretable deep learning approach. Learning deep embeddings for 18 cancer types allowed us to compare and contrast cancer types in terms of the transcriptomic variation across patients. We also incorporated different data modalities and data sources in our pan-cancer analysis, i.e., normal tissue expressions from GTEx and survival and

mutation profiles from TCGA, and investigated cancer latent spaces from complementary angles.

DeepProfile can learn informative and interpretable cancer-expression expression embeddings. We showed that DeepProfile's ensemble approach can improve the downstream prediction performance by integrating signals from hundreds of individual models. Besides successfully encoding cancer related signals, our nodes were enriched for a wider set of biological pathways compared to alternative approaches, emphasizing its biological interpretability.

Learning these interpretable cancer latent spaces enabled us to examine the mechanisms shared across cancers. We identified immune response regulation and antigen presentation genes to be among the highest sources of transcriptomic variation among cancer samples across all cancer types. More interestingly, the universally important genes are significantly overlapping with cell surface and cytokine receptors; thus, playing a key role in regulating inflammation and immune response. Universally important pathways across cancers include cell cycle, which might reflect variance in proliferative index across cancer samples, and oxidative phosphorylation, which is associated with the metabolic spectrum of cells. The normal tissue analysis also allowed us to differentiate tissue-specific signals from cancer-tissue specific ones, such as focal adhesion, which is strongly important only in cancer tissues as it is associated with epithelial-to-mesenchymal transition states.

Besides highlighting interesting universal cancer mechanisms, we detected that the genes and pathways that are specifically important for particular cancer types define dominant subtypes or grades of differentiation within a tissue category. Again, the cancer character scores we calculated for each pathway and cancer type pointed to processes explicitly important for cancerous tissue.

When correlated with cancer phenotypes, genes and pathways that account for the variance in cancer transcriptomes become of particular interest. We investigated the associations of DeepProfile nodes with patient survival and tumor mutational burden which revealed mitotic activity as an important indicator of mutation burden for majority of the cancer types we analyzed. Furthermore, adaptive immunity related pathways were significantly associated with survival across multiple cancer types and specifically MHC-II antigen presentation and DNA mismatch repair pathways were captured by the highest number of cancer types. Analysis at the original expression-level highlighted that low expression of mismatch repair is associated with better prognosis. The deficiency of DNA mismatch repair can increase the mutation rate which

leads to stronger anti-tumor immune response and therefore, better chance of survival. On the other hand, the direction of association with MHC-II antigen presentation pathway was divergent across cancer types which can be explained by anti-inflammatory or immunosuppressive roles macrophages play in different cell types.

We offer DeepProfile as a general deep learning framework that can be applied to new expression datasets; the trained models can be used to encode new set of cancer expression profiles. DeepProfile ensemble approach can also be applied to any suitable expression data and looking forward, we seek to explore the feasibility of the model for different diseases and biological domains.

Another possible next step is to extend the pipeline to incorporate publicly available RNA-Seq expression profiles which will increase the sample size therefore, the statistical power of the model. Increasing the sample size can also make it possible to train specific models for cancer subtypes (e.g., glioma and glioblastoma for brain cancer) which would allow us to robustly compare the transcriptomic landscapes of subtypes of a cancer. Furthermore, adapting our framework for single cell expression profiles can be a very interesting future direction. In this chapter, we focused on the transcriptomic variation at the patient level; however, single cell data can facilitate encoding the variation across different cell types and examine the specific roles different cell types can play in the tumor environment.

DeepProfile is a pan-cancer resource providing cancer expression latent spaces and various characterizations of latent nodes including cancer-type specificity and cancer-tissue characteristic. We also developed a general methodology for connecting genes and pathways to sample- or tumor-level characteristics via DeepProfile nodes, as we exemplified with survival and TMB analysis, with the goal of allowing researchers to adopt this methodology for any cancer expression and phenotype. Looking forward, we want to extend this analysis to other phenotypes ranging from cancer grade or metastatic properties to in-vivo chemotherapy response.

Cancer Type	# Samples	# GEO Series	# Genes	# PCs
Acute Myeloid Leukemia	6,534	92	11,579	1,000
Bladder Cancer	371	15	13,237	250
Brain Cancer	4,282	108	7,656	1,000
Breast Cancer	11,963	194	7,592	1,000
Cervical Cancer	443	16	13,237	250
Colorectal Cancers	5,616	114	10,030	1,000
Head and Neck Cancers	643	26	11,020	500
Kidney Cancer	2,293	48	12,730	1,000
Liver Cancer	1,937	60	13,236	1,000
Lung Cancer	4,869	96	10,551	1,000
Ovarian Cancer	2,714	64	10,342	1,000
Pancreas Cancer	602	33	4,610	500
Prostate Cancer	1,195	47	11,646	1,000
Sarcoma	2,330	68	9,916	1,000
Melanoma	1,240	45	12,339	1,000
Stomach Cancer	1,742	33	12,641	1,000
Thyroid Cancer	776	18	13,237	500
Uterine Cancer	661	21	13,237	500

Table 2.1: **DeepProfile datasets.** The number of samples, datasets, and genes are reported for each GEO dataset.

2.4 Methods

2.4.1 Downloading and preprocessing of gene expression profiles

We downloaded publicly available gene expression datasets generated by either of the two microarray platforms: Affymetrix GeneChip Human Genome U133 Plus 2.0 (Affy HG-U133 Plus 2.0) and Affymetrix GeneChip Human Genome U133A 2.0 (Affy HG-U133A 2.0). These datasets were available from the National Center for Biotechnology Information (NCBI) Gene Expression Omnibus (GEO) database [27] for 18 cancer types. The number samples and genes for each cancer type is available in Table 2.1.

While GEO searching filters results according to supplied keywords, the returned results may still include gene expression samples from healthy tissues or patients with cancer types other than the queried cancer type. To eliminate these irrelevant samples, we removed the samples that do not contain the search keywords in their titles, characteristics, or descriptions. We also manually curated the data for further cleaning without unnecessarily eliminating relevant samples. With these steps, we aimed to minimize the number of incorrectly included or excluded samples. We also excluded cell line expression samples and used only

Cancer Type	TCGA Mapping	# Samples with RNA-Seq and Survival	# Samples with RNA-Seq and Microarray	# Samples with RNA-Seq and Mutation	GTEx Normal Tissue Mapping	# GTEx Samples
AML	LAML	149	-	170	Whole Blood	407
Bladder	BLCA	405	-	129	Bladder	10
Brain	GBMLGG	669	150	434	Brain	1,670
Breast	BRCA	1,079	529	979	Breast	289
Cervical	CESC	291	-	193	Cervix	10
Colorectal	COADREAD	598	220	220	Colon	506
Head and Neck	HNSC	518	-	279	-	-
Kidney	KIPAN	883	87	642	Kidney	44
Liver	LIHC	365	-	195	Liver	175
Lung	LUSC, LUAD	996	183	408	Lung	426
Melanoma	SKCM	459	-	339	Skin	859
Ovarian	OV	303	298	185	Ovary	132
Pancreas	PAAD	177	-	150	Pancreas	247
Prostate	PRAD	497	-	332	Prostate	151
Sarcoma	SARC	259	-	245	Muscle	563
Stomach	STAD	388	-	272	Stomach	261
Thyroid	THCA	500	-	403	Thyroid	445
Uterine	UCEC	543	54	247	Uterus	110

Table 2.2: **DeepProfile cancer type mappings.** The cancer type mapping and sample counts are reported for TCGA and GTEx datasets.

patient samples because the same cell line's low expression variance across datasets might prevent deep neural networks from learning a reliable model. Despite our automated and manual curation to eliminate samples from cell lines, other cancer types, and healthy tissue, it is still possible that some outlier samples are included in our GEO data collection.

To integrate data from various platforms, we converted platform-specific probe IDs to gene symbols using the GEO conversion lists for each platform. For each cancer, we took the genes present in all data series we have available. A study might have different sample batches submitted on different dates, we corrected for these potential batch effects within each study using ComBat [88], where different batches correspond to data subsets submitted at different dates. We log transformed the expression measurements, standardized (i.e., zero-mean and unit variance) each gene in each dataset to ensure that different input features (i.e., gene expression levels) are on the same scale, and applied mean imputation to impute missing gene-level measurements. We also excluded duplicate samples with the same GEO IDs. We concatenated all datasets and applied batch effect correction, once again using ComBat, considering each study to be a separate batch in order to minimize the effect of potential study-specific confounders.

2.4.2 DeepProfile pipeline, architecture and training

Training variational autoencoder models

An autoencoder is a neural network that consists of an encoder and a decoder network with an information bottleneck layer with D hidden nodes (i.e., $D \ll M$) in the middle [89]. It generates an embedding Z such that the information present in the original space is preserved in this lower dimensional space as well. Specifically, the encoder network, defined as $f_\phi : X \to Z$ maps from the input space $X \in \mathbb{R}^M$ to latent embedding $Z \in \mathbb{R}^D$. Similarly, the decoder network, defined as $g_\psi : Z \to X$, maps the embedding Z back to input space. We optimize over both networks to minimize the squared 2-norm distance between our input X and the reconstructed input as follows:

$$min_{\phi,\psi} \| x - g_\psi(f_\phi(x)) \|_2^2 \tag{2.1}$$

A *variational autoencoder* (VAE) is an extension of a standard autoencoder that takes as input an NxM

matrix X, where N denotes the number of samples, M denotes the number of features and X_{ij} denotes the feature j of sample i. It also consists of encoder and decoder networks but adopts a regularized training such that the model is robust to overfitting [45]. To perform regularization, VAE learns a distribution of the latent space rather than learning the encoding directly and samples from the learned distribution to generate an embedding. VAE trains the model to bring the distribution of the latent space as close to a standard Gaussian distribution (i.e., $N(0, 1)$) as possible, which regularizes the learned distribution.

We define the encoder network as $f_\theta : X \rightarrow \mu_x, \sigma_x$, which maps from the input space $X \in \mathbb{R}^M$ to latent space distribution mean $\mu_x \in \mathbb{R}^D$ and distribution standard deviation $\sigma_x \in \mathbb{R}^D$. We then sample from the distribution to define embedding $Z \in \mathbb{R}^D$: $Z \sim N(\mu_x, \sigma_x)$. The decoder is defined the same way as it is in a standard autoencoder. To regularize the distribution over the latent space, VAE adds a regularization term to the model's loss function, i.e., Kullback-Leibler divergence between the learned distribution and a normal distribution [90]. The network is trained to be optimized as follows:

$$min_{\phi,\psi} \| x - g_\psi(f_\phi(x)) \|_2^2 + KL[(\mu_x, \sigma_x), N(0, 1)] \tag{2.2}$$

where $KL[(\mu_x, \sigma_x), N(0, 1)]$ denotes the Kullback-Leibler divergence between the distributions. This regularization component forces the encoder and decoder networks to learn a generalizable, smooth latent space that embeds similar samples close to each other.

We trained VAE models using the cancer-specific gene expressions as inputs; the encoder and decoder networks both include 2 hidden layers, and the two networks mirror each other in structure. All layers use rectified linear unit activation except the last layers of both networks, where we applied linear activation with no dropout and batch normalization on all encoder layers. The number of hidden nodes in the intermediate layers varies according to the latent space size: we have 250 and 100 hidden nodes in the first two layers, respectively, for the latent space sizes 25, 50, 75, and 100; 250 and 50 hidden nodes for the latent space size 10; and 100 and 25 hidden nodes for the latent space size 5. The minibatch size is 50, and we trained the models using the Adam optimizer with a learning rate of 0.0005. We initialized each VAE model with a different random set of weights using Glorot uniform weight initialization. We tuned the hyperparameters of VAE models, the number of hidden layers, dropout rate, batch size, and the number of epochs with 5-fold cross validation, using validation reconstruction error as the metric. The same VAE model architecture is

used for all cancer types. We built the models using Keras with Tensorflow backend.

Before training different VAE models for a cancer type, we extracted the principal components [91] of the expression matrix; we trained the VAEs using these components as inputs, a commonly used approach for training deep neural networks to prevent overfitting [92]. We selected the number of principal components based on the number of samples (see Table 2.1 for the number of components for each cancer type).

Learning DeepProfile nodes

DeepProfile combines all embeddings generated by VAE models to learn a single, robust latent space that can preserve both high- and low-level features. We trained a total of $|D| * |R|$ models, where D is a set of possible latent space sizes for individual VAE models and R is a set of random seeds used to initialize model weights. We trained a VAE model for each latent space size $d \in D$ and for each random seed $r \in R$ for the initial weights, which we denote as $\text{VAE}_{d,r}$. For our experiments, we used D = 5, 10, 25, 50, 75, 100 and R = 0, . . . , 99, which corresponds to 100 random models for each of the 6 latent space sizes, for a total of 600 VAE models. Each VAE model takes the expression matrix $X \in \mathbb{R}^M$ as input and outputs an embedding $Z \in \mathbb{R}^D$.

Across all $|D| * |R|$ models, we have $|D| * |R|$ embeddings and $\sum_{d \in D} d * |R|$ nodes in total (600 embeddings and 26,500 latent nodes for our setting). To group similar data encodings, we applied k-means clustering to cluster all nodes from all models. k-means assigns each of $\sum_{d \in D} d * |R|$ nodes to one of the L clusters, where L is the number of DeepProfile latent nodes. Note that we disregard the information about which node came from which model: we simply applied clustering to all nodes by treating them as independent and identically distributed (i.i.d.). As a result, different nodes of the same VAE model might be in different clusters as well as in the same cluster. Also, one cluster may include nodes from different models with the same latent space size (i.e., different runs), or it can also include nodes from models with different latent space size. After k-means groups the nodes that are similar across runs and dimensions, we created one ensemble node per cluster by averaging the values of all nodes in that cluster to obtain a final embedding, $Z \in \mathbb{R}^L$ (Figures A.1 and A.2A).

To select the latent embedding size for DeepProfile, we applied G-means clustering, an extension of k-means clustering to determine the optimal number of clusters [93]. For each cancer type, we fitted G-

means clustering before training the k-means models to select the optimal k value. We averaged the optimal number of clusters across 18 cancers to set $L = 150$ as the final latent embedding size after rounding down the exact average, which was 157. We selected the same latent size for each cancer type to enable direct comparison between cancer-specific embeddings.

Our DeepProfile framework can encode user cancer expression samples. When user expression samples are passed to the DeepProfile model, we first apply the same preprocessing procedure we applied to our training samples after eliminating the genes not available for the training samples. We pass the preprocessed expression matrices to our trained VAE models to generate embeddings. We then use the learned ensemble assignments to cluster VAE nodes and take the average value in each cluster to define the final DeepProfile embedding for user samples (Figure A.2B). Users can select the number of latent dimensions, in which case ensemble label assignments will be calculated again to define the new ensemble nodes.

Gene- and pathway-level attributions of DeepProfile nodes

To calculate gene-level attributions of DeepProfile nodes, which denote how much each gene contributes to the learned latent nodes, we used Integrated Gradients, a gradient-based feature attribution method for neural networks [46] (Figure A.2C). When applied to a neural network model, Integrated Gradients learns the sample-level importance values of each input feature for each output node. To determine the global importance of each gene for a node, we calculated the absolute valued average of attribution scores across all training samples for each cancer type. Since DeepProfile is an ensemble of VAE models, where each DeepProfile node combines multiple VAE nodes, feature attributions for each DeepProfile node are calculated by averaging the attributions of the VAE nodes defining that ensemble node.

To calculate pathway-level attributions, we used gene-level attributions and carried out pathway enrichment tests using a total of 1,077 functional pathways from Reactome [49], BioCarta [48], and KEGG [47] from the C2 collection of the version 6.2 of MSigDB [94; 95]. For enrichment tests, we used Fisher's Exact Test's (FET) [96]. From the gene list for each pathway, we removed the genes that are not present in our input expression matrix and passed the top G genes with the highest importance values for a DeepProfile node to FET, where G is the average pathway length across all 1,077 functional pathways from Reactome, BioCarta, and KEGG. We applied Benjamini-Hochberg FDR correction [97] across all nodes.

2.4.3 Comparing DeepProfile to alternative dimensionality reduction methods

Training alternative approaches

We compared DeepProfile to alternative dimensionality reduction algorithms, including the commonly used linear methods as well as other deep learning approaches. We trained these algorithms using the same preprocessed gene expression levels that we used as input to DeepProfile VAE models. For all methods except PCA, which is a deterministic algorithm, we repeated the model training 10 times with different random seeds and generated 10 different embeddings.

Gaussian random projection maps the original input to a lower dimensional space, where each component is randomly drawn from a normal distribution.

Principal Component Analysis (PCA) [91] is a linear dimensionality reduction method that generates orthogonal components to encode variation in the original input space. We used the top 150 principal components when comparing it to the DeepProfile embedding, which has 150 nodes.

Independent Component Analysis (ICA) [98] is also a linear dimensionality reduction method that learns independent components from the original space.

Autoencoder (AE) [89] is a deep unsupervised neural network consisting of an encoder and decoder network trained to learn a latent space that can reconstruct the original space as successfully as possible. For autoencoder trainings, we used the same top principal components of the preprocessed gene expression levels as we did for training DeepProfile to enable a fair comparison between models. We tuned the hyperparameters of AE models, the number of layers, number of hidden nodes, dropout rate, and batch size using 5-fold cross validation with reconstruction error as the metric. In the final AE model, we have 1 hidden layer each in encoder and decoder networks with 750 hidden nodes, 0.1 dropout rate, and batch size of 100. The model was trained with the Adam optimizer using a learning rate of 0.0005.

Denoising Autoencoder (DAE) [99] is a regularized autoencoder model that adds noise to the input data in order to generate more robust embeddings. We applied the same procedure to denoising autoencoder models as autoencoders. The final tuned model has 1 hidden layer each in encoder and decoder networks, 750 hidden nodes and 0.1 dropout rate. We optimized the model using the Adam optimizer with a learning rate of 0.0005 and batch size of 100.

Creating TCGA RNA-Seq embeddings

We downloaded TCGA RSEM normalized log2 transformed RNA-Seq expression matrices for all cancer types from Broad Institute (data version 2016_01_28) (https://gdac.broadinstitute.org/) generated by TCGA Research Network (https://www.cancer.gov/tcga/). The mapping of DeepProfile and TCGA cancer types as well as the number of samples are listed in Table 2.2. We preprocessed the TCGA expressions with the same pipeline we used for preprocessing GEO expression datasets: we selected the genes available only in the training data, zero imputed the genes missing in the TCGA dataset, and standardized each gene to zero-mean univariance. We encoded the TCGA samples using the PCA model trained on the training data. To generate the DeepProfile embeddings, we loaded all trained VAE models, encoded TCGA PCA transformed input features with each of the models, and used the pre-learned ensemble labels to cluster the nodes of our VAE embeddings and define a 150-dimensional DeepProfile embedding for TCGA RNA-Seq samples. We repeated this procedure for each cancer type. Similarly, for all the alternative dimensionality reduction approaches, we used the trained models to encode TCGA RNA-Seq samples.

Predicting TCGA cancer patient survival

We downloaded TCGA merged clinical files for all cancer types from Broad Institute (data version 2016_01_28) generated by the TCGA Research Network. For the prediction task, we binarized the survival labels by 5-year survival and excluded samples with survival information that was censored within 5 years. Since the distribution of survival labels can be quite unbalanced, we randomly subsampled from the class with the highest number of samples to ensure that both classes have equal number of samples. We repeated this random sampling process 50 times and trained the prediction model for each sub-sampling, which is a logistic regression model with l1 regularization. The model was trained with nested-cross validation; 20-fold cross validation was used to split the data to training and test set and for each training fold, and the regularization coefficient was selected using 5-fold cross validation. The survival prediction models were trained separately for each cancer type.

For deterministic models, e.g., PCA, we trained one prediction model that outputs 50 accuracy scores for each 50 sub-samplings. For other models that were trained 10 times, e.g., ICA, we calculated accuracy score for each model for each sub-sampling and averaged the accuracies across 10 models to output an average

accuracy for each model. We repeated the same procedure for comparing DeepProfile's prediction accuracy with different dimensional VAE embeddings. To calculate the statistical significance of outperformance for each pair of methods, we used the Wilcoxon sign-ranked test.

Comparison of DeepProfile microarray and RNA-Seq embeddings

To demonstrate that DeepProfile can learn informative latent spaces from both microarray and RNA-Seq test data, we used the TCGA cancer samples for which we have both RNA-Seq and microarray expression available. We downloaded TCGA log2 LOWESS normalized microarray expression matrices for all the available cancer types from Broad Institute (data version 2016_01_28) generated by the TCGA Research Network. We selected the genes present in both microarray and RNA-Seq datasets to enable a fair comparison and preprocessed microarray expression profiles following the same preprocessing steps applied to GEO samples. We then measured the Pearson correlation coefficient between the gene expression matrices generated with the two technologies obtaining a correlation coefficient for each TCGA sample. We then created DeepProfile embeddings for TCGA microarray profiles and measured the Pearson correlation between DeepProfile RNA-Seq and microarray embeddings for each TCGA sample.

Comparing DeepProfile pathway coverage to alternative dimensionality reduction methods

When comparing DeepProfile to other dimension reduction methods in terms of pathway coverage, which we used as a metric for evaluating the biological relevance of the learned latent space, we followed the same procedure as we used for DeepProfile. We applied Fisher's Exact Test (FET) and obtained a p-value for each node-pathway pair, denoting the significance of enrichment.

To carry pathway enrichment tests, we first obtained the gene-level attributions for each dimensionality reduction method. For PCA, ICA, and RP, we obtained the absolute valued component matrices, which denotes the contribution of each gene to each learned component. For AE and DAE, we used Integrated Gradients [46] to obtain gene-level attributions for the embedding nodes, following the same procedure we applied to VAE models. Since we trained each model 10 times with different random initializations, we repeated the FET for each of the 10 models and averaged the pathway enrichment results over 10 runs.

We compared DeepProfile's pathway coverage to other dimension reduction methods using 3 different

metrics: First, we compared the average pathway coverages. The enrichment tests we carried provided us with an enrichment p-value for each node-pathway pair. After FDR correction, we marked the node-pathway pairs with a p-value < 0.05 as significant and calculated the total number of significant enrichments for each node. We defined the number of pathways significantly captured by each node as the pathway coverage of that node. Then, we averaged these node-level pathway coverages across all nodes to calculate the average final coverage of an embedding. This metric let us define an average pathway coverage score per model and per cancer type.

Second, we compared the distributions of node-level pathway coverages across models. Again, using the same pathway-level attribution p-values, we counted the number of pathways significantly captured by each node of each embedding.

Third, we compared the percent of nodes annotated by at least one pathway. For various significance threshold values that range from a p-value of 10^{-1} to 10^{-10}, we counted the number of pathways with a p-value below the threshold for each node. This again returned a pathway coverage value for each node of the embedding. We then calculated the percent of nodes with a pathway coverage above one, which is effectively the percent of nodes annotated by at least one pathway with a p-value below the threshold.

Comparing DeepProfile pathway coverage to VAE models

When comparing pathway enrichment of DeepProfile to VAE models, we used the gene-level attributions for each different dimensional VAE model and applied FET to obtain a p-value for each node of each 600 different models. DeepProfile is an ensemble model that combines 600 VAE models to define an ensemble embedding, and our aim was to show that the DeepProfile model can preserve the pathways captured by the individual VAE models. Accordingly, we used two different metrics to compare the pathway coverages of VAE models to DeepProfile:

First, we compared DeepProfile pathway coverages to the average pathway coverages of all 600 different VAE models. For each pathway, we calculated the percent of VAE models that captured this pathway significantly (i.e., with at least one node of the embedding with an FDR corrected p-value < 0.05). We then compared the pathways captured by the threshold percent of the VAE models, where the threshold ranges from 50 to 90, to DeepProfile to investigate whether the pathways captured by VAE models could also be

captured by DeepProfile.

Second, we compared DeepProfile model to VAE models with different dimension sizes. For each different dimensional VAE model, e.g., a 5-dimensional VAE model, we marked a pathway to be captured if the majority of the VAE models (at least 51 of 100 models) significantly captured the pathway. We repeated the same procedure for each of the 6 different dimensional VAE models to mark the pathways captured by different dimensional VAE models. We then compared the pathways captured by a threshold number of different dimensional models, where the threshold ranges from 1 to 6, to the DeepProfile model in order to investigate whether the pathways captured by these different VAE models could be detected by DeepProfile as well.

2.4.4 Pan-cancer gene and pathway analysis

Detecting universally important genes

To detect the highest-scoring genes across all cancer types, we used the gene-level attributions of DeepProfile nodes. We calculated the average attribution score across all nodes to define an overall importance score for each gene, and we converted these scores to percentile scores, where the highest and lowest scored genes takes value of 100 and 0, respectively. Once we separately obtained these percentile scores for each gene for each cancer type, we calculated the average percentile score across 18 cancer types as the universal percentile score of a gene. We could then sort the genes by their universal percentile scores to detect the top universally important ones. We generated a network of the top 100 universally important genes using STRING [100] with a medium confidence level of 0.4, using all interaction sources and eliminating disconnected nodes. We visualized the network using Cytoscape [101].

To detect the pathways enriched for the top universally important genes, we carried out FET on the top 100 universally important genes using Reactome, BioCarta, KEGG, and GO Biological Process [55] gene sets. We applied FDR correction across all pathways.

To detect whether DeepProfile's universally important genes are enriched for various immune cell type signatures, we collected gene signatures of T cells, B cells, neutrophils, and macrophages [50] and obtained a total of 108 genes available in our training expression dataset. We again carried out FET for the top 100 universally important genes using these markers.

To determine whether DeepProfile's universally important genes are enriched for cell surface and cytokine receptors, we first collected gene sets from the Cell Surface Protein Atlas (CSPA) [53], UniProt database [54], and Gene Ontology (GO). From CSPA, we downloaded the list of human surfaceome proteins and their annotations, selected the proteins with a high confidence CSPA category and protein probability of 1.0, and obtained a list of 555 human surface proteins. From the UniProt database, we downloaded human cell surface receptors using the keyword 'cell surface receptor', selected the reviewed proteins, and obtained a list of 1,307 genes. Similarly, we downloaded human cytokine receptors using the keyword 'cytokine receptor', selected the reviewed proteins, and obtained a list of 773 genes. From GO, we used the gene set 'Immune response regulating cell surface receptor signaling pathway' which has 346 genes. Note that for each gene set, we used the genes available only in DeepProfile's training expression matrix and reported the intersecting gene counts. We again carried out FET for the top 100 universally important genes using these 4 distinct gene lists to calculate enrichment scores.

To compare the top universally important genes detected by DeepProfile to the universally important genes for PCA, we carried out the same analysis applied on DeepProfile to PCA. Using the attribution scores of each gene for the PCA model, we calculated the average attribution score across all top 150 principal components, converted the scores to percentile scores, and calculated the average across 18 cancers to define a universal importance score for each gene for PCA. Similarly, we carried out FET for the top 100 universally important PCA genes using the same gene sets and the 4 receptor gene lists.

Detecting universally important pathways

To detect the highest-scoring pathways across all cancer types, we used the pathway-level attributions we have for each DeepProfile latent node, which contains the p-value of enrichment for each node-pathway pair, and selected the maximum -log10(p-value) across all nodes for each pathway to get an enrichment score for each cancer type-pathway pair. We marked a pathway as being significantly captured by a cancer type if the FDR corrected p-value is below 0.05. To determine the universally important pathways, we counted the number of cancer types that significantly captured each pathway. We also recorded the average -log10(p-value) of enrichment for each pathway by taking the mean across all cancer types that significantly captured the pathway. After obtaining the number of cancers and average enrichment score for each pathway, we

sorted the pathways first by the number of cancers and then by enrichment scores to get the list of universally important pathways.

Calculating cancer character scores for pathways

To calculate a cancer character score for each pathway, we carried out a normal tissue analysis. First, from the GTEx portal (https://gtexportal.org/home/datasets), we downloaded RNA-Seq expression (gene TPMs, accession number phs000424.v7.p2) [29] and selected the tissues corresponding to the 18 cancer types we have (see Table 2.2 for the mapping of DeepProfile and GTEx tissue types and the number of samples). We preprocessed and encoded GTEx expression profiles following the same pipeline used for TCGA RNA-Seq expression and generated normal tissue embeddings using DeepProfile.

To detect how successfully each node can differentiate cancer vs. normal tissue, we trained logistic regression classifiers by passing the cancer and normal tissue DeepProfile embeddings as input and predicted cancer vs. normal tissue labels. We repeated the training 500 times with different random samplings and recorded the mean of absolute value of classifier weights from all models. We defined the cancer character score of each node as the absolute valued classifier weight, denoting the importance of each DeepProfile node in differentiating cancer from normal tissue, where a high cancer character score indicates that the node is quite important for differentiating the tissue type.

We then mapped these node-level cancer character scores to pathways to determine the cancer-tissue specificity of each pathway for each cancer type. For each pathway, we calculated the weighted average of cancer character scores using the -log10(p-value) of enrichment scores for that pathway as the weights and obtained an average cancer character score for each pathway-cancer type pair. Note that if a pathway is not enriched for any of the nodes, we assigned it a cancer character score of 0. To define a universal cancer character score for each pathway, we calculated the average across the cancer character scores of 18 cancers, excluding the cancers with a score of 0.

Detecting cancer-specific genes and pathways

To identify the genes that are high scoring specifically for a certain cancer type, we used the importance scores we calculated for each gene-cancer type pair. For each gene, we calculated the difference between

the percentile score of a cancer and the maximum percentile score across all other 17 cancers. These cancer-specific difference scores allowed us to detect the top cancer-specific genes for each cancer.

To calculate the enrichment score for the PAM50 genes [66], we used FET for the top N highest scoring genes for breast cancer where N ranges from 1 to 1000. We then applied Bonferroni correction over all thresholds to report the final p-value of enrichment.

To detect cancer-specific pathways, we used the -log10(p-value) of enrichment scores we calculated for each pathway/cancer type pair and calculated the difference between the enrichment score for one cancer type and the maximum enrichment score across all other 17 cancers. We also assigned a cancer character score to each of these cancer-specific pathways using the absolute-valued classifier weights from the normal tissue analysis after converting them to percentile scores.

2.4.5 Pan-cancer survival and mutation analysis

With the goal of associating each pathway with patient survival, we used the TCGA RNA-Seq DeepProfile embeddings we learned for 18 cancers along with the survival status. We separately fitted univariate Cox regression models [102] to each DeepProfile node, recorded p-values of model coefficients, and applied FDR correction over all nodes for each pathway. We repeated the model training for each cancer type.

To detect the pathways determining patient survival, we mapped these node-level survival scores to pathways. For each pathway, we calculated the weighted average of -log10(survival p-values) across all nodes using the -log10(enrichment p-values) as the weights. When none of the nodes could significantly capture a pathway, we assigned a survival score of 0 to that pathway. We also masked the pathway enrichment p-values with a z-score below 0.25 to prevent the involvement of lowly ranked pathways in the average score calculation. Since the survival analysis using enrichment scores from FET did not provide a rich enrichment for survival, we repeated the FET using a broader set of top genes (4 * average pathway size) in order to carry out an informative survival analysis. This pipeline let us define a survival p-value for each pathway-cancer type pair. We then marked a pathway to be relevant to survival if both the FDR corrected enrichment p-value and the survival p-value were below 0.05.

To detect universally survival-related pathways, for each pathway, we counted the number of cancer types with significant survival scores. We also calculated the average enrichment and survival -log10(p-

values) across all the cancer types significantly associated with survival. Sorting the pathways by the number of cancer types and then by the average survival score provided us with the top universally survival-associated pathways. We visualized the top 20 universally survival-associated pathways using Cytoscape EnrichmentMap tool [103], where the connections between pathways were determined by the Jaccard similarity of gene memberships of pathways.

To carry out mutation analysis, we downloaded TCGA mutation profiles for all cancer types. We selected the samples for which we have both expression measurements and tumor mutational burden (TMB) data available and calculated the total number of mutations for each cancer sample by summing the number of mutations for each gene. If k different mutations occurred for one gene, we increased the total mutation count by k.

To assign a mutation association score to each DeepProfile node, we calculated the Pearson correlation between each node of DeepProfile embedding and the log of total mutation count after eliminating outlier mutation scores beyond the 95% confidence level (z-score > 1.96). We repeated these experiments for each of the 18 cancer types and obtained node-level TMB correlation p-values. To map the node-level p-values to pathways, we repeated the same procedure we followed for the survival analysis.

2.4.6 Downstream survival analysis

For pathways detected to be relevant to patient prognosis, we carried out a downstream survival analysis independent from DeepProfile model. We first fitted univariate Cox regression models to each of the 23 genes included in the KEGG mismatch repair pathway and 91 genes included in the Reactome MHC class II antigen presentation pathway using TCGA RNA-Seq profiles. We trained the models separately for each cancer type and recorded z-scores to get the direction of association.

To investigate the association of average expression of the selected pathways with survival, we first calculated the average expression of genes from the KEGG mismatch repair pathway and the average expression of HLA-D genes (HLA-DMA, HLA-DMB, HLA-DOA, HLA-DOB, HLA-DPA1, HLA-DPB1, HLA-DQA1, HLA-DQA2, HLA-DRB1, HLA-DRB5) from the Reactome MHC class II antigen presentation pathway across all TCGA samples with survival record. We then created Kaplan-Meier plots [104] for each pathway using the calculated average expression values. When generating the plots, we separated

the patients into two groups based on their average expressions: one group with expression above (mean + standard deviation) and one group with expression below -(mean + standard deviation).

To detect the immune cell type responsible from expression of HLA-D genes, we first calculated the average expression of each gene across all TCGA samples for each cancer. Note that we carried out the mean operation prior to preprocessing of the expression matrices. We sorted the genes by their average expression and converted the rankings to percentile scores. To order the importance of different immune cell types, we calculated the average gene percentile score of the gene signatures for each immune cell type, which we defined as XCR1 and CLEC9A for dendritic cells; MS4A1, CD79A, and PAX5 for B cells; and CD163, CD68, CSF1R for macrophages. We then recorded the Pearson correlation between average expression of the cell type signatures listed above and the average expression of HLA-D genes.

For the macrophage analysis, we downloaded the gene signatures for pro-inflammatory and immunosuppressive macrophages [86] and used the list of unique genes for each macrophage group (CD40, CXCL9, CXCL10, CXCL11, SLAMF1, and TNIP3 for pro-inflammatory macrophages; CFP, HRH1, NPL, PDCD1LG2, and RENBP for immunosuppressive macrophages) to again measure gene ranking percentile scores. We carried out the same analysis as we carried for immune cell types. We also repeated this analysis using the extensive list of pro-inflammatory or immunosuppressive macrophage signatures [87].

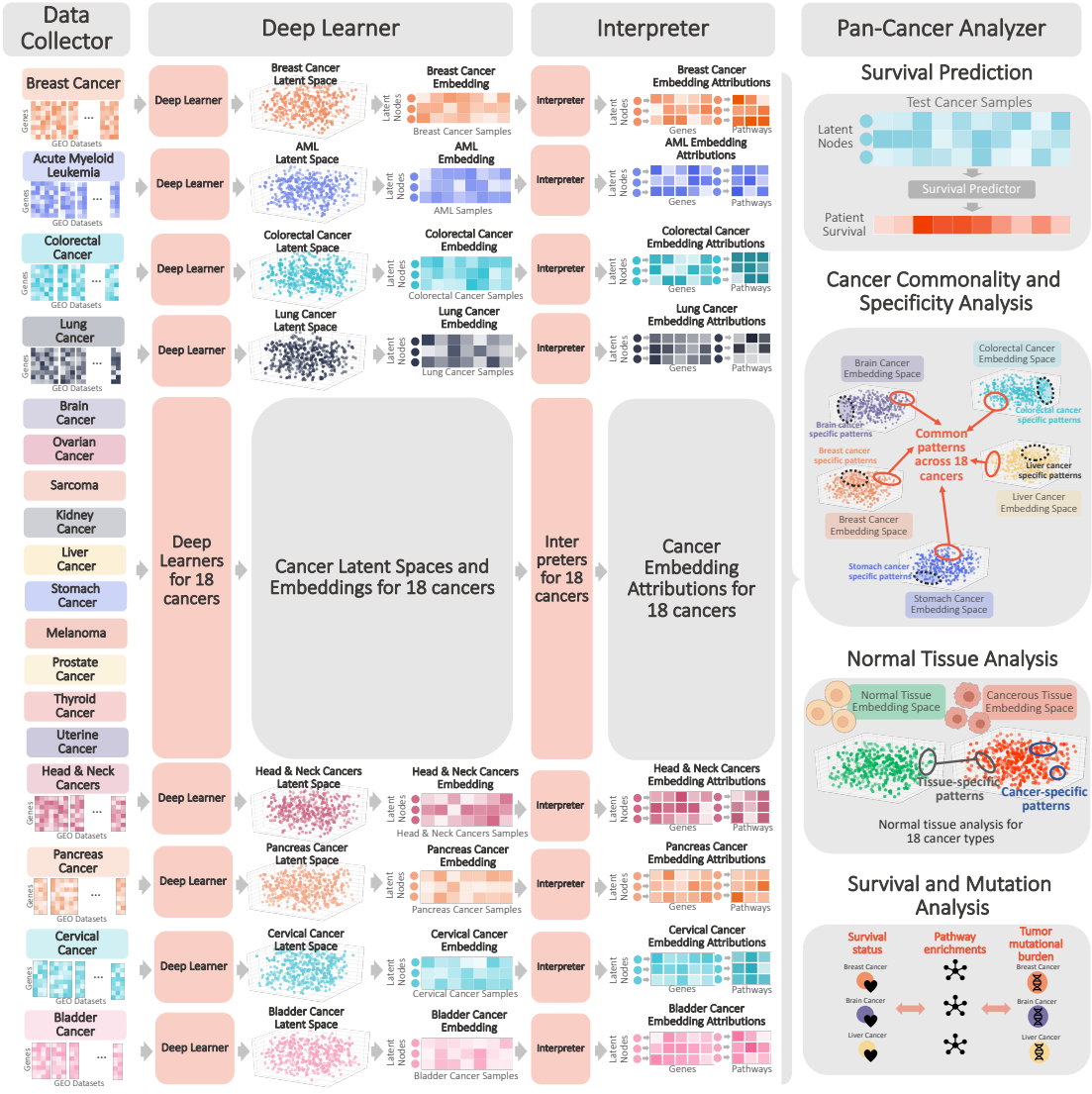

Figure 2.1: **DeepProfile pan-cancer framework** (see caption on next page).

Figure 2.1: **Data Collector.** For 18 different cancer types shown in the figure, we downloaded all gene expression datasets from the common microarray platforms available at the time from the NCBI Gene Expression Omnibus (GEO). We preprocessed and concatenated all downloaded datasets to define cancer-specific expression matrices containing the expression measurements for each gene and cancer sample pair. In total, we have 50,211 samples from 1,098 GEO datasets.
Deep Learner. We pass the expression matrices to Deep Learner models to learn cancer-specific latent spaces. Deep Learner is an ensemble of variational autoencoders (VAEs) that encodes the high-dimensional expression signals to a biologically informative lower-dimensional space called latent space. We then map the training samples to the learned latent spaces and define cancer sample embeddings where each DeepProfile latent node corresponds to one dimension of the latent space that encodes a certain source of variation across cancer samples.
Interpreter. We pass the learned embeddings to Interpreter models to extract gene-level and pathway-level attributions for each latent node. Gene-level attributions denote how much each gene contributes to a latent node. Similarly, pathway-level attributions denote the pathways significantly associated with the most important genes of each latent node.
Pan-Cancer Analyzer. Using the cancer-specific embeddings and attributions; we carry a detailed pan-cancer analysis including (1) evaluating how successful DeepProfile embeddings are at preserving important biological signals by predicting the survival status of cancer patients, (2) analyzing the latent spaces of 18 cancers to discover cancer-common and specific patterns, (3) differentiating cancer-specific patterns from tissue-specifying ones by contrasting cancer embeddings to normal tissue embeddings and (4) investigating survival and mutation related signals by integrating DeepProfile embeddings with survival and tumor mutational burden profiles.
(See Figures A.1 and A.2)

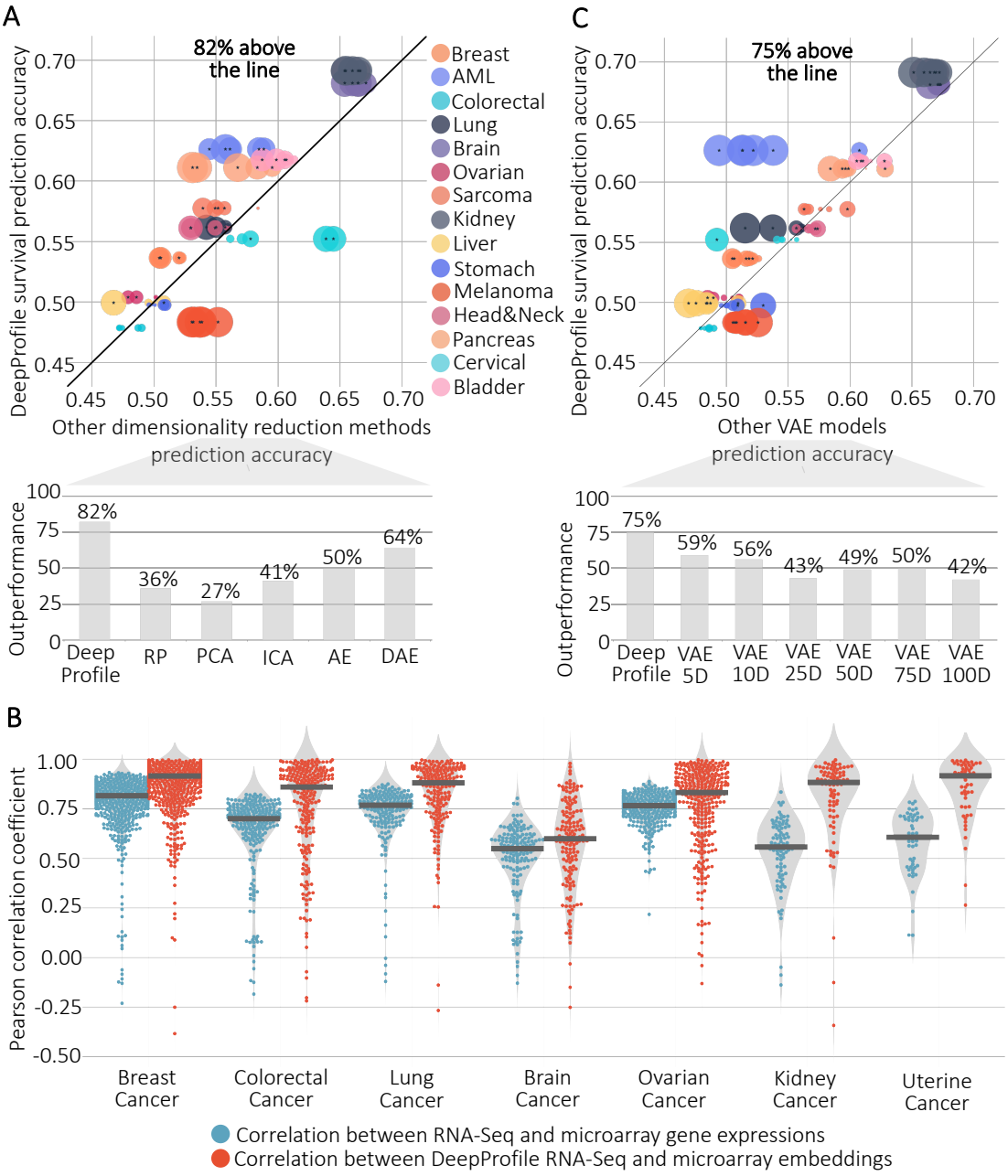

Figure 2.2: **DeepProfile survival prediction comparisons** (see caption on next page).

Figure 2.2: **(A) (Top Plot)** For 15 different cancer types shown in the figure, we compare the survival prediction accuracy of DeepProfile TCGA RNA-Seq cancer expression embeddings to embeddings generated by other dimensionality reduction methods. For 3 cancer types prediction could not be carried due to insufficient sample size. The models are trained to predict 5-year survival. Each dot denotes a method and a cancer type pair, where the dots are colored by cancer type. The size of the dots is determined by the negative p-value of the significance of outperformance measured by Wilcoxon sign-ranked test. All nodes with p-value < 0.05 are marked with a star. **(Bottom Plot)** The scatter plot is generated for each of the five alternative dimension reduction methods and the percent of cases where each method outperforms others is summarized in the bar plot.
(B) Distribution of correlation coefficients between TCGA RNA-Seq and microarray expression profiles and correlation coefficients between TCGA RNA-Seq and microarray DeepProfile embeddings. Each dot represents the correlation coefficient for one cancer sample and the distributions are plotted for 7 cancer types. The median correlation coefficient for each sub distribution is shown with the dark colored bars.
(C) The same plots in A are created for comparing DeepProfile to all different dimensional VAE models we used to define DeepProfile ensemble model. (See Figure A.3)

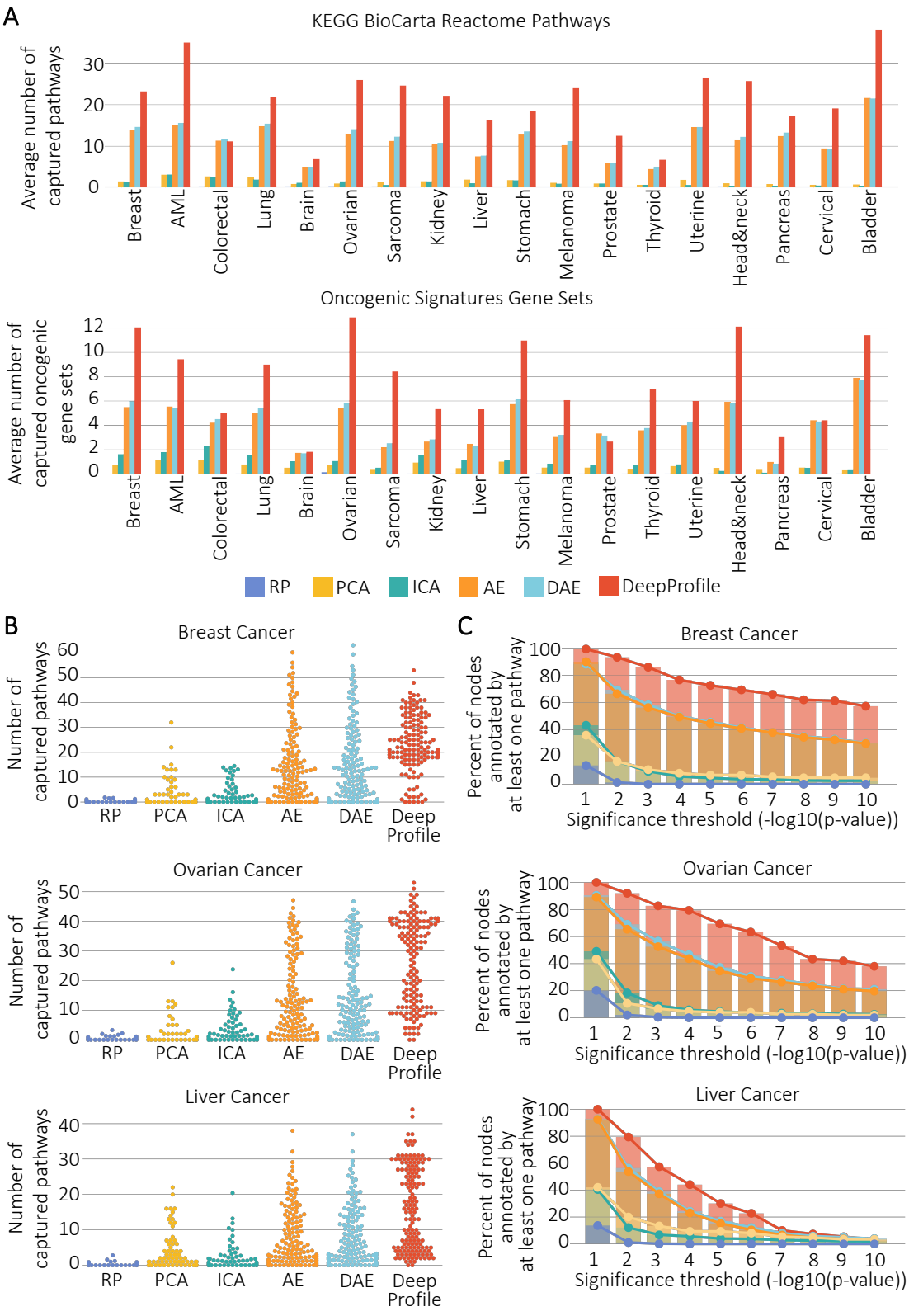

Figure 2.3: **Comparison of pathway enrichment of DeepProfile and other dimensionality reduction methods** (see caption on next page).

Figure 2.3: **(A)** The average number of pathways significantly captured (FDR corrected p-value < 0.05) by nodes of latent embeddings of DeepProfile and other dimensionality reduction methods are shown for KEGG, BioCarta, Reactome pathways **(Top plot)** and Oncogenic Signatures gene sets **(Bottom plot)**.
(B) Distribution plots of number of KEGG, BioCarta, Reactome pathways significantly captured by each latent node.
(C) Comparison of the percent of nodes annotated by at least one pathway above the significance threshold for a range of threshold values.
(See Figures A.4, A.5, A.6, , A.7)

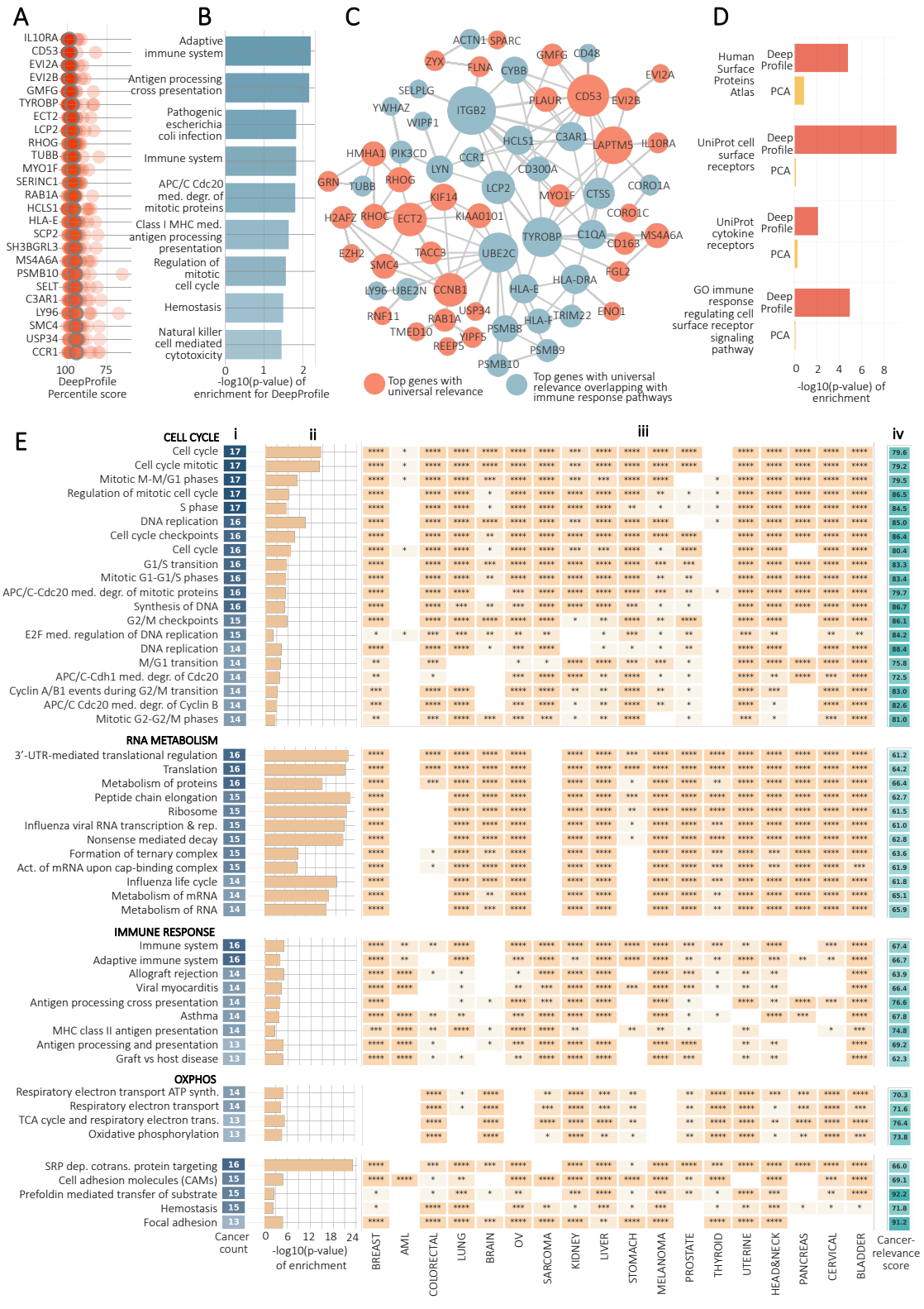

Figure 2.4: **DeepProfile cancer commonality analysis** (see caption on next page).

Figure 2.4: **(A)** List of top highest-scoring genes across 18 cancer types for DeepProfile. The percentile scores of the top scoring genes are shown for all cancers and the average percentile scores across 18 cancers are highlighted. The plot is zoomed in for clear comparison.
(B) The top enriched pathways (KEGG, BioCarta, Reactome) for the top 100 universally important DeepProfile genes and the corresponding FDR-corrected p-values.
(C) Network of top 100 genes with universal importance. The network is generated with STRING and disconnected nodes are excluded. The size of a node is determined by hubness, i.e., the number of edges. Genes that are included in immune response related pathways are colored blue.
(D) The enrichment p-values for cell surface and cytokine receptors for DeepProfile and PCA top 100 universally important genes.
(E) List of top pathways that are universally important sorted based on the number of cancer types significantly capturing the pathway. **(i)** Number of cancer types (out of 18) significantly capturing each pathway. **(ii)** –log10(p-value of enrichment) averaged over all cancers significantly capturing the pathway. **(iii)** Heatmap denoting the significance of enrichment p-values for top pathways and all cancer types. The star annotations correspond to the significance of enrichment (* = p-value < 0.05, ** = p-value < 0.01, *** = p-value < 0.001, **** = p-value < 0.0001). **(iv)** Cancer character scores of pathways. The cancer character score denotes the relevance of each pathway to normal or cancerous tissue where a higher score indicates that the pathway is specifically important for cancerous tissues. The pathways are grouped manually in terms of their functional relations. The order of the groups is determined by the average cancer character score of each pathway group.

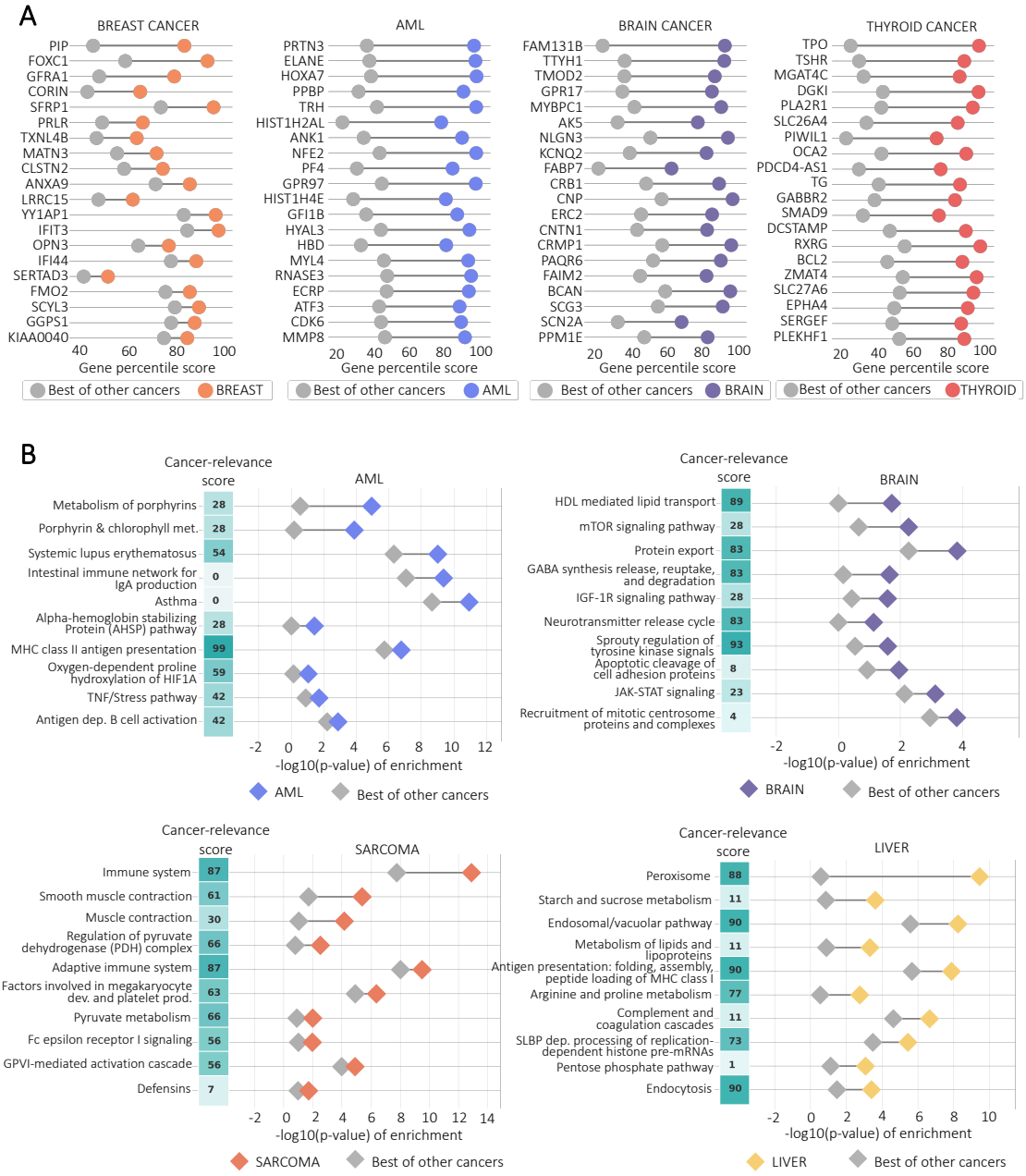

Figure 2.5: **DeepProfile cancer specificity analysis** (see caption on next page).

Figure 2.5: **(A)** The plots of cancer-specific genes shown for 4 cancer types. The difference between the percentile score for the specific cancer type and the highest percentile score among all the other 17 cancer types for the top 20 genes with the highest difference score are shown for each cancer type separately. The colored dots show the percentile score of one gene for the specific cancer type and the gray dots show the highest percentile score the same gene has among all the other cancer types. The genes are sorted based on the difference values.
(B) The plots of cancer-specific pathways along with cancer character scores for 4 cancer types. Pathways are sorted based on the difference between the -log10(p-value) for the specific cancer type and the highest -log10(p-value) among all the other 17 cancer types. Each dot pair represents the -log10(p-value) corresponding to one pathway for the specific cancer type and the highest -log10(p-value) among all the other cancer types. The vector of cancer character scores shows the cancer character percentile score of the node that is capturing the shown pathway.

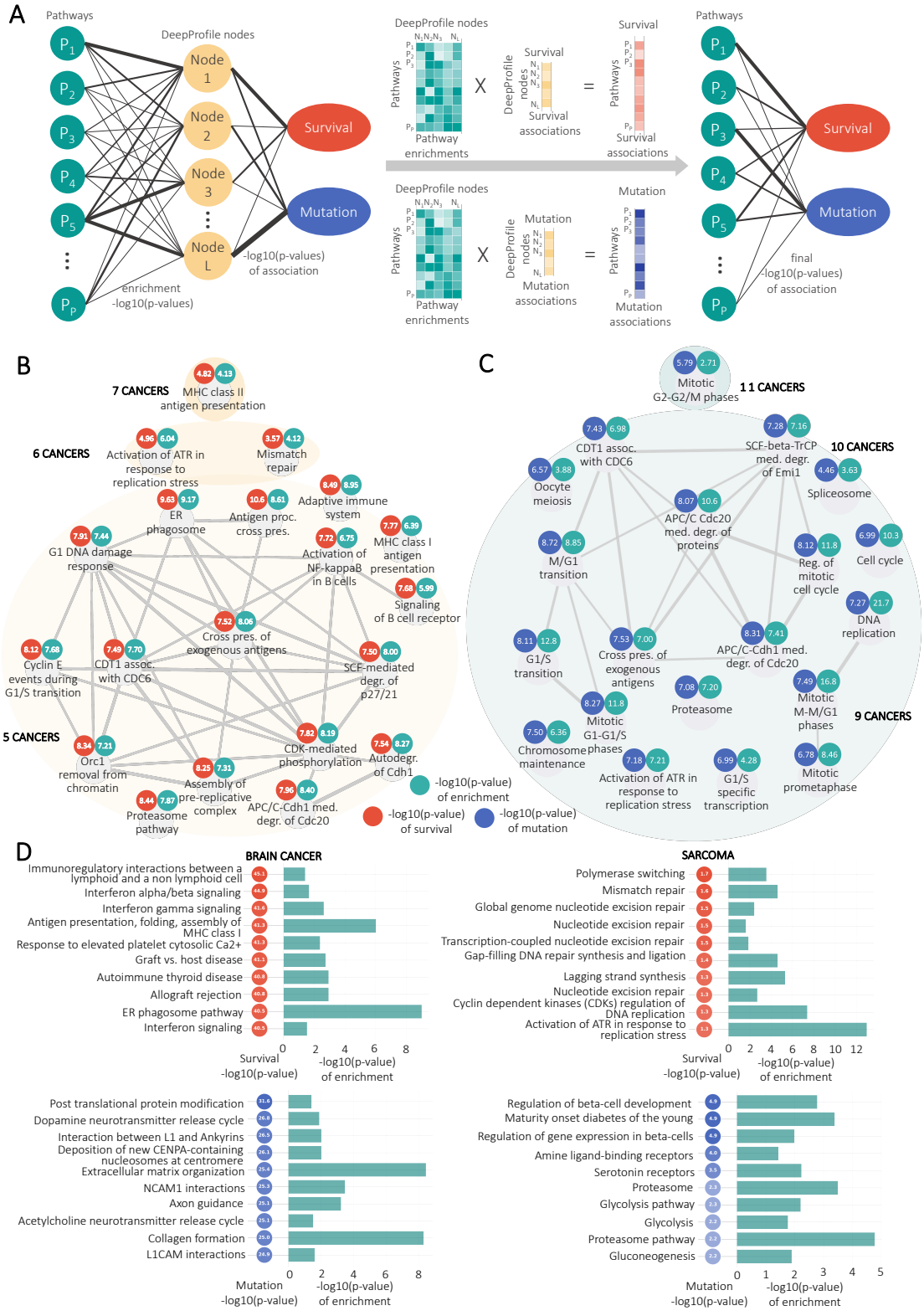

Figure 2.6: **DeepProfile survival and mutation analysis** (see caption on next page).

Figure 2.6: **(A)** The schematic of DeepProfile survival and mutation analysis at pathway level. Each pathway is connected to each DeepProfile node with a certain enrichment score (-log10(p-value)) as we extracted pathway-level explanations for DeepProfile nodes before. We then fit univariate Cox survival regression models to each DeepProfile node and obtain a p-value denoting the significance of association of a node with survival. We also measure the Pearson correlation between each DeepProfile node and tumor mutational burden (TMB) and obtain a p-value denoting the significance of association of a node with TMB. In order to calculate the overall pathway-level survival and mutation association scores, we take the inner product of enrichment and node-level association matrices and normalize the matrix. This way, we obtain the final –log10(p-values) of survival and mutation association for each pathway. We repeated this process for each cancer type which allows us to carry out cancer common and specific survival and mutation analyses.
(B) The network of top survival-related pathways. For each pathway group, we show the number of cancers for which the pathway is significantly enriched and significantly associated with survival (p-value < 0.05). We further show the –log10(p-value) of enrichment and –log10(p-value) of survival association averaged across all cancers detecting the pathway to be relevant to survival. The connections between pathways are determined based on gene membership Jaccard similarities.
(C) Same as (B) for the top TMB-related pathways.
(D) Plots of top survival and mutation associated pathways for brain cancer (left) and sarcoma (right). The upper plot shows the top 10 pathways with highest survival scores for the shown cancers along with the survival and enrichment –log10(p-values) and the lower plot shows the top 10 pathways with highest mutation scores for the shown cancers.

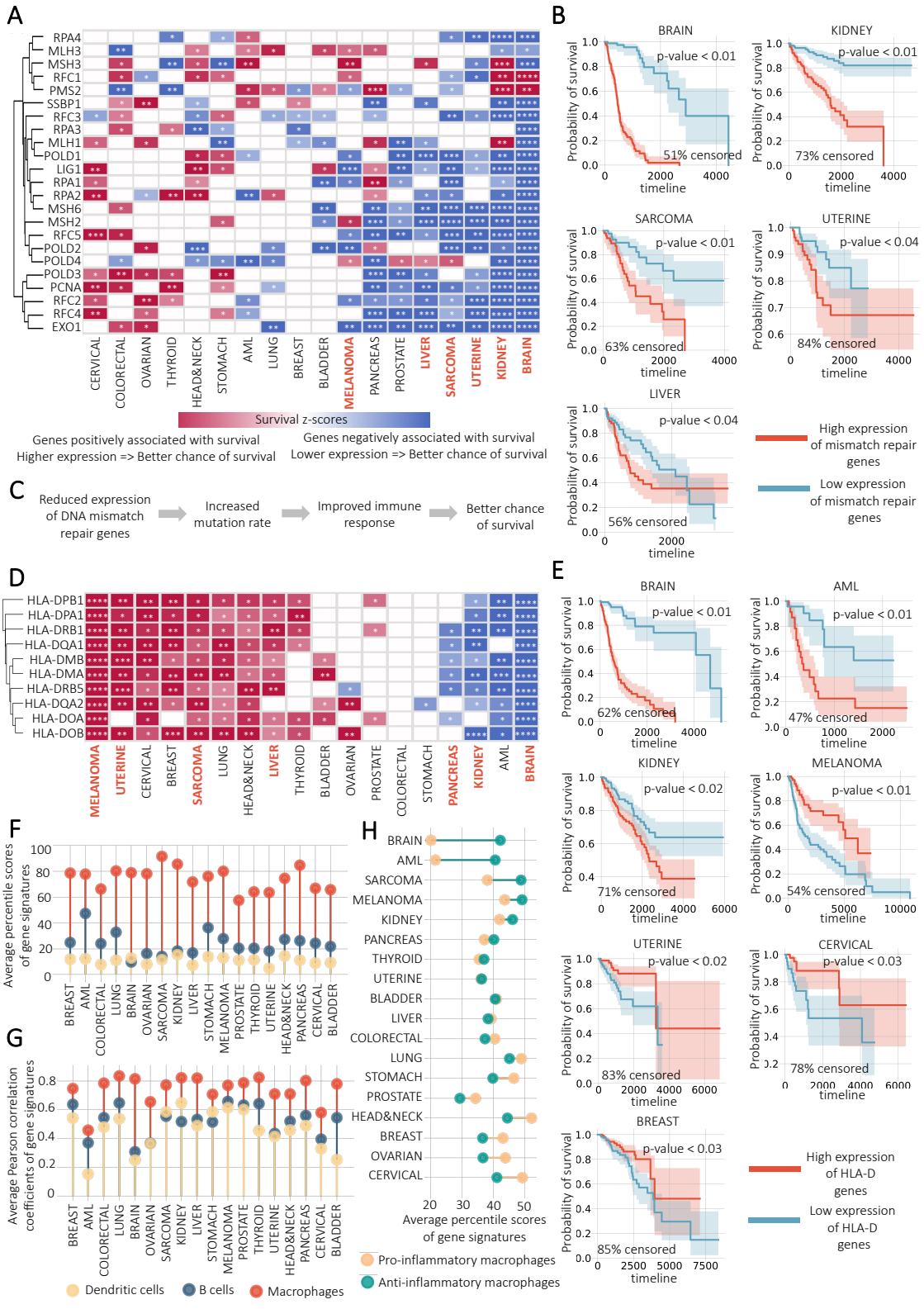

Figure 2.7: **Downstream survival analysis** (see caption on next page).

Figure 2.7: **(A-C) Mismatch repair pathway survival analysis**
(A) Heatmap of survival z-scores of all genes included in KEGG mismatch repair pathway (* = magnitude of z-score > 1, ** = magnitude of z-score > 2, *** = magnitude of z-score > 3, **** = magnitude of z-score > 4). 6 cancer types detected by DeepProfile are highlighted.
(B) Kaplan-Meier plots of average expression of mismatch repair pathway. The samples with an expression above (mean + one standard deviation) are marked as highly expressed and below -(mean + one standard deviation) are marked as lowly expressed. The log rank test p-values and the percent of censored samples are reported for each cancer. 5 cancer types with a log rank test p-value below 0.05 are shown.
(C) Schematic of mismatch repair mechanism.
(D-H) MHC class II pathway survival analysis
(D) Heatmap of survival z-scores of all HLA-D genes included in Reactome MHC class II antigen presentation pathway. 7 cancer types detected by DeepProfile are highlighted.
(E) Kaplan-Meier plots of average expression of HLA-D genes for cancer types with a log rank test p-value below 0.05.
(F) Comparison of average percentile scores of gene dendritic cells, B cells, and macrophages shown for 18 cancers.
(G) Comparison of average Pearson correlation between the expression of HLA-D genes and cell type signatures for the three cell types shown for 18 cancers.
(H) Comparison of average percentile scores of pro- and anti-inflammatory macrophages shown for 18 cancers.
(See Figure A.8)

Chapter 3

Adversarial deconfounding autoencoder for learning robust gene expression embeddings

The work presented in this chapter is adapted from [105], previously published by *Bioinformatics*:
Ayse B. Dincer, Joseph D. Janizek, and Su-In Lee. Adversarial Deconfounding AutoEncoder for learning robust gene expression embeddings. *Bioinformatics*, 36(Supplement_2):i573–i582, 2020.
https://doi.org/10.1093/bioinformatics/btaa796.

3.1 Background

As discussed extensively in previous chapters, gene expression profiles provide a snapshot of cellular activity, which allows researchers to examine the associations among expression, disease, and environmental factors. As we have shown in Chapter 2, unsupervised deep learning has enormous potential to extract important biological signals from the vast amount of expression profiles, which was also explored by recent studies [106; 107]. Two features of unsupervised learning make it well suited to gene expression analysis. (i) *The ability to train informative models without supervision*, critical because it is challenging to obtain a high number of expression samples with coherent labels. Although many new expression profiles are released daily, the portion of the datasets with labels of interest is often too small. Moreover, different studies may collect information on different traits and even measure the same traits using different metrics [108].

(ii) *The ability to extract patterns from the data without imposed directions or restrictions*. Without focusing on a specific phenotype prediction, these models enable us to learn patterns unconstrained by the limited phenotype labels we have. This aspect can be key to unlocking biological mechanisms yet unknown to the scientific community.

Using unsupervised models to learn biologically meaningful representations would make it possible to map new samples to the learned space and adapt our model to any downstream task. We exemplified this in the previous chapter where we developed the DeepProfile framework to extract biologically relevant embeddings, which allowed us to examine cancer mechanisms and their association with patient survival. While DeepProfile approach addressed the fundamental limitations of unsupervised deep neural networks preventing them from being applied to high-dimensional biological data, there are still complexities within the data itself that makes it challenging to learn meaningful representations. Expression measurements often contain out-of-interest sources of variation, in addition to the signal we seek. When training an unsupervised model, we want the model to capture the true signal and learn latent dimensions corresponding to biological variables of interest. However, especially when collected from a large cohort or multiple cohorts, expression profiles have, in addition to the true signal, variations in expression measures across samples as a result of (1) technical artefacts that are not relevant to biology, such as batch effects, (2) out-of-interest biological variables, such as sex, age, medications, and (3) random noise. (See Fig. 3.1.) We call these biological or non-biological artefacts that systematically affect expression values *confounders*. Unfortunately, in many datasets, confounder-based variations often mask true signals, which hinders learning biologically meaningful representations.

As **a motivating example**, Fig. 3.2a shows how confounder signals might dominate true signals in gene expression data. We consider the KMPlot breast cancer expression dataset [109], which combines multiple microarray studies from The Gene Expression Omnibus (GEO) [27]. We take the two GEO datasets with the highest number of samples and plot the first two principal components (PCs) [91] to examine the strongest sources of variation. Figure 3.2a shows that the two datasets are clearly separated, exemplifying how confounder-based variations affect expression measurements.

We then apply an autoencoder [89] to this dataset, i.e., an unsupervised neural network that can learn a latent space that maps M genes to D nodes ($M \gg D$) such that the biological signals present in the

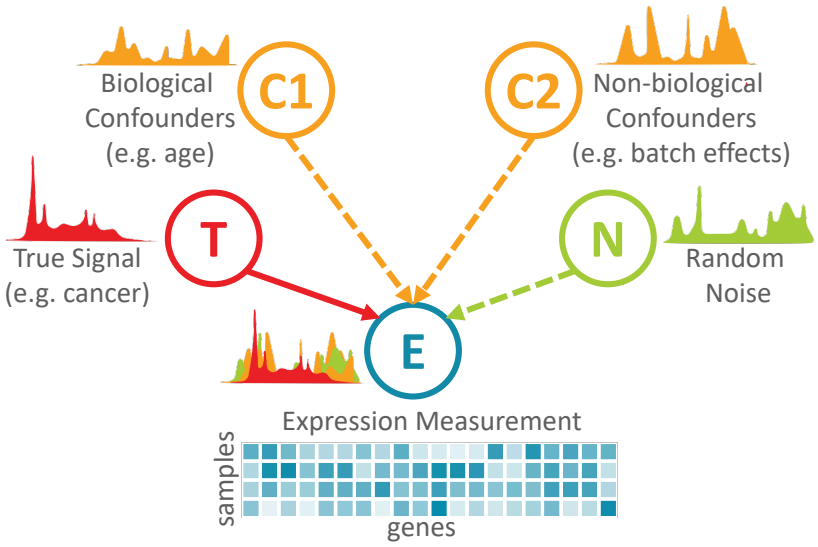

Figure 3.1: **A simplified graphical model of measured expression** shown as a mix of true signal, confounders of biological and non-biological origin, and random noise. Note that this model shows neither possible connections between a true signal and confounders nor connections among confounders.

original expression space can be preserved in D-dimensional space. The autoencoder tries to capture the strongest sources of variation to reconstruct the original input successfully. In our example, unfortunately, it is encoding variation introduced by confounders rather than interesting signals. Figure 3.2b depicts the PC plot of the autoencoder embedding. It shows that the dataset difference is encoded as the strongest source of variation. When we measure the Pearson's correlation coefficient between each node value and the binary dataset label, we observe that 78% of the embedding nodes are significantly correlated with the dataset label (p-value < 0.01). This means that most latent nodes are contaminated, making it difficult to disentangle biological signals from confounding ones.

Confounders also prevent our learning a robust, *transferable* model to generate generalizable embeddings that capture biological signals conserved across different domains. For instance, if we learn a model from one expression dataset that detects a disease signal, we want this signal to be valid for similar datasets. To simulate this problem, we use a separate set of samples from a different GEO study from the KMPlot data. We train the autoencoder using only the first two datasets, and we then encode the "external" samples from the third GEO study using the trained model. The PC plot in Figure 3.2c highlights the distinct separation between the external dataset and the two training datasets. This simple example shows how confounder effects can prevent us from learning transferable latent models.

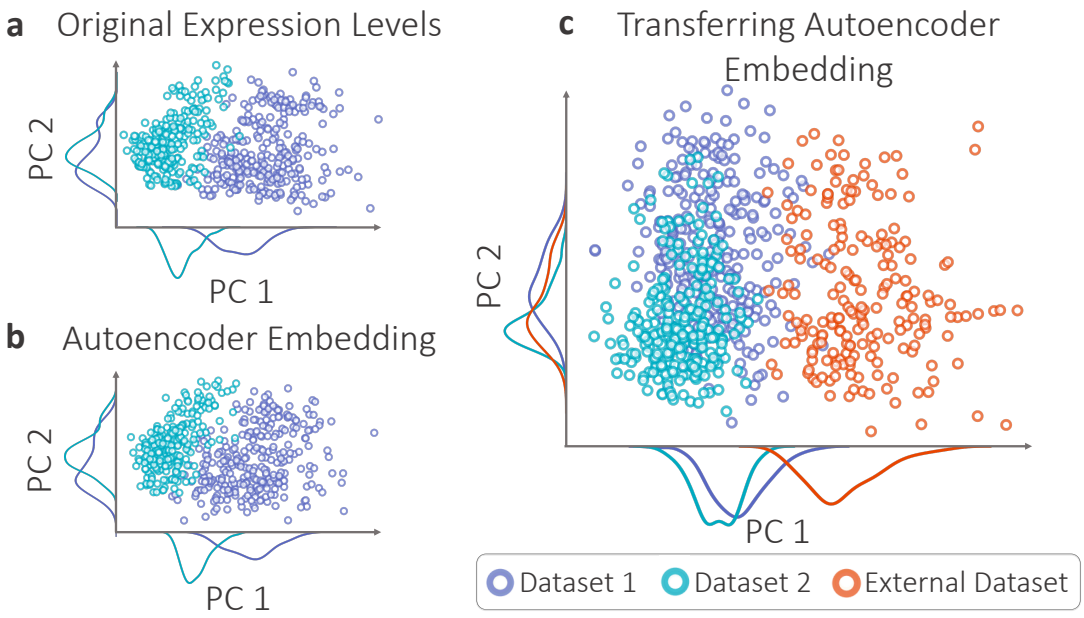

Figure 3.2: **An example of confounder effects.** The plot of top two principal components (PCs) colored by dataset labels generated for (a) the expression matrix, and (b) autoencoder embedding of the expression. (c) PC plot of the embeddings for training and external samples generated by the autoencoder trained from only the two datasets and transferred to the third external dataset.

In this chapter, we address the entanglement of confounders and true biological signals to show the power of deep unsupervised models to unlock biological mechanisms. While Chapter 2 focused on *high dimensionality* of expression profiles, now, we aim to tackle another fundamental problem preventing us from deciphering these biological datasets: *batch effects and confounders*. Our goal is to generate biologically informative expression embeddings that are both robust to confounders and generalizable. To achieve this goal, we propose a deep learning approach to learning deconfounded expression embeddings, which we call AD-AE (Adversarial Deconfounding AutoEncoder).

AD-AE consists of two neural networks trained simultaneously: (i) an autoencoder network optimized to generate an embedding that can reconstruct the data as successfully as possible, and (ii) an adversary network optimized to predict the confounder from the generated embedding. These two networks compete against each other to learn the optimal embedding that encodes important signals without encoding the variation introduced by the selected confounder variable. To demonstrate the performance of AD-AE, we used two expression datasets – breast cancer microarray and brain cancer RNA-Seq – with a variety of confounder variables, such as dataset label and age. We showed that AD-AE can generate unsupervised embeddings

that preserve biological information while remaining invariant to selected confounder variables. We also conducted transfer experiments to demonstrate that AD-AE embeddings are generalizable across domains. Our code and data are made available at https://gitlab.cs.washington.edu/abdincer/ad-ae.

3.1.1 Related work

In high-throughput data, we often experience systematic variations in measurements caused by technical artefacts unrelated to biological variables, called *batch effects*. Many techniques have been developed to eliminate batch effects and *correct* high-throughput measurement matrices. AD-AE differs from batch correction approaches in two ways. First, we do not focus only on *batch effects*; rather we aim to build a model generalizable to any biological or non-biological confounder. Second, we do not concentrate on correcting the data, i.e., trying to eliminate confounder-sourced variations from the expression and outputting a *corrected* version of the expression matrix. Instead, our major objective is learning a confounder-free representation. We seek to reduce the dimension of an expression matrix in order to learn meaningful biological patterns that do not include confounders.

While keeping these differences in mind, we can compare our approach to batch correction techniques to highlight the advantages of our adversarial confounder-removal framework. In their review, Lazar *et al.* [110], categorize batch correction techniques into two groups. (i) Location-scale methods, which match the distribution of different batches by adjusting the mean and standard deviation of the genes. Examples include mean-centering [111], gene-standardization [112], ratio-based correction [113], distance-weighted discrimination [114], and probably the most popular of these techniques, the Empirical Bayes method (i.e., ComBat) [88]. (ii) Matrix factorization techniques, which factorize the expression matrix to identify factors associated with batch effects and then reconstruct the data to eliminate batch-affected components. Examples include surrogate variable analysis [115] and various extensions of it [116; 117].

One limitation that applies to previously listed methods is that they model batch effects *linearly*. AD-AE, on the other hand, can eliminate *nonlinear* confounder effects as well. Several recent studies accounted for non-linear batch effects and tried modeling them with neural networks. These studies used either (i) maximum mean discrepancy [118] to match the distributions of two batches present in the data, such as

119 and 120, or (ii) an adversarial approach for batch removal, such as training an autoencoder with two separate decoder networks that correspond to two different batches along with an adversarial discriminator to differentiate the batches [121] or generative adversarial networks trained to match distributions of samples from different batches [122] or to align different manifolds [123]. These methods all handle nonlinear batch effects. However, their application domain is limited since they can correct only for binary batch labels. AD-AE is a general model that can be used with any categorical or continuous valued confounder. To our knowledge, only one study [124] used an adversarial model to remove categorical batch effects, extending the approaches limited to binary labels. Our approach is significantly different since we focus on removing confounders from the latent space to learn deconfounded embeddings instead of trying to deconfound the reconstructed expression. Another unique aspect of our approach is that we concentrate on learning generalizable embeddings for which we carry out transfer experiments for various expression domains and offer these domain transfer experiments as a new way of measuring the robustness of expression embeddings.

Our work takes its inspiration from research in *fair machine learning*, where the goal is to prevent models from unintentionally encoding information about sensitive variables, such as sex, race, or age. Multiple studies aimed to generate fair representations that try to learn as much as possible from the data without learning the membership of a sample to sensitive categories [125; 126]. Two studies with high relevance to our approach are Ganin *et al.* [127] and Louppe *et al.* 128, which use adversarial training to eliminate confounders. Ganin *et al.* applied this idea to an autoencoder network to predict a class label of interest while avoiding encoding the confounder variable. Louppe *et al.* also used an adversarial training approach by fitting an adversary model on the outcome of a classifier network to deconfound the predictor model. One advantage of Louppe's model over the others is that it can work with any confounder variable, including continuous valued confounders. Janizek *et al.* 129 applied this approach to predict pneumonia from chest radiographs, showing that the model performs successfully without being confounded by selected variables.

Inspired by this work, we adopt a similar adversarial training approach for expression data, which is highly prone to confounders. Unlike prior work, AD-AE fits an adversary model on the *embedding space* to generate robust, confounder-free embeddings.

3.2 Methods

3.2.1 Standard autoencoder

We used a standard autoencoder as the baseline for our experiments, which takes as input an expression vector x of M genes. The autoencoder consists of (i) an encoder network, defined as $f_\phi : X \mapsto Z$, which maps from the input space $X \in \mathbb{R}^M$ to latent embedding $Z \in \mathbb{R}^D$, and (ii) a decoder network, $g_\psi : Z \mapsto X$, that maps the embedding Z back to the input space. Our encoder/decoder networks are fully or densely connected neural networks with rectified linear unit (ReLU) activation between layers; thus, Z is in effect the network's information bottleneck. We then optimize over encoder and decoder networks as follows:

$$min_{\phi,\psi}\mathbb{E}\left\|x - g_\psi(f_\phi(x))\right\|_2^2, \tag{3.1}$$

where ϕ and ψ are the parameters of our encoder and decoder neural networks, respectively. The expectation is taken over the training data, and the loss is the squared 2-norm distance between the input x and the reconstructed input.

3.2.2 Our approach: Adversarial Deconfounding Autoencoder (AD-AE)

We propose the adversarial deconfounding autoencoder to generate biologically informative gene expression embeddings robust to confounders (Figure 2.3). AD-AE consists of two networks. The first is an autoencoder model *l* that is optimized to generate an embedding that can reconstruct the original input. The second is an adversary model *h* that takes the embedding generated by the autoencoder as input and tries to predict the confounder *C*. We note that *C* is not limited to being a single confounder and could be a vector of them.

Our goal is to learn an embedding *Z* that encodes as much information as possible while not encoding any confounding signal. To achieve this, we train models *l* and *h* simultaneously. Model *l* tries to reconstruct the data while also *preventing the adversary from accurately predicting the confounder*. At the same time, adversarial predictor *h* tries to update its weights to accurately predict the confounder from the generated embedding. As shown by Louppe *et al.* [128], assuming the existence of an optimal model and sufficient statistical power, models *l* and *h* will converge and reach an equilibrium after a certain number of epochs,

where *l* will generate an embedding *Z* that is optimally successful at reconstruction and *h* will only randomly predict a confounder variable from this embedding. In other words, the autoencoder will converge to generating an embedding that contains no information about the confounder, and the adversary will converge to a random prediction performance.

We train our model in three steps:

Step 1: The autoencoder model *l* is defined per Section 2.1. We pretrain the autoencoder to optimize equation 3.1 and generate an embedding *Z*.

Step 2: We define the adversary model $h_\upsilon : Z \mapsto C$, mapping the embedding *Z* to confounder *C*. We again use fully connected multilayer perceptron networks with ReLU activation for the adversary, which is optimized with the following objective:

$$min_\upsilon \mathbb{E}[L((h_\upsilon(x), c)]. \tag{3.2}$$

Here, we define a general loss function L that can be any differentiable function appropriate for the confounder variable (e.g., mean squared error for continuous confounders, cross-entropy for categorical confounders). We pretrain our adversary model accordingly to predict the confounder as successfully as possible.

Step 3: After separately pretraining both networks, we begin joint adversarial training by optimizing over the two networks. When optimizing the joint model, we first freeze the weights of the adversary model and train the autoencoder model for one epoch on a randomly selected minibatch of the data using stochastic gradient descent to optimize the following objective:

$$min_{\phi,\psi,\upsilon} \mathbb{E}\left[\left\|x - g_\psi(f_\phi(x))\right\|_2^2 - \lambda L(h_\upsilon(x), c)\right]. \tag{3.3}$$

This corresponds to updating the weights of the autoencoder to *minimize* equation 3.1 while *maximizing* 3.2 (minimizing the negative of the objective). We then freeze the autoencoder model and train the adversary

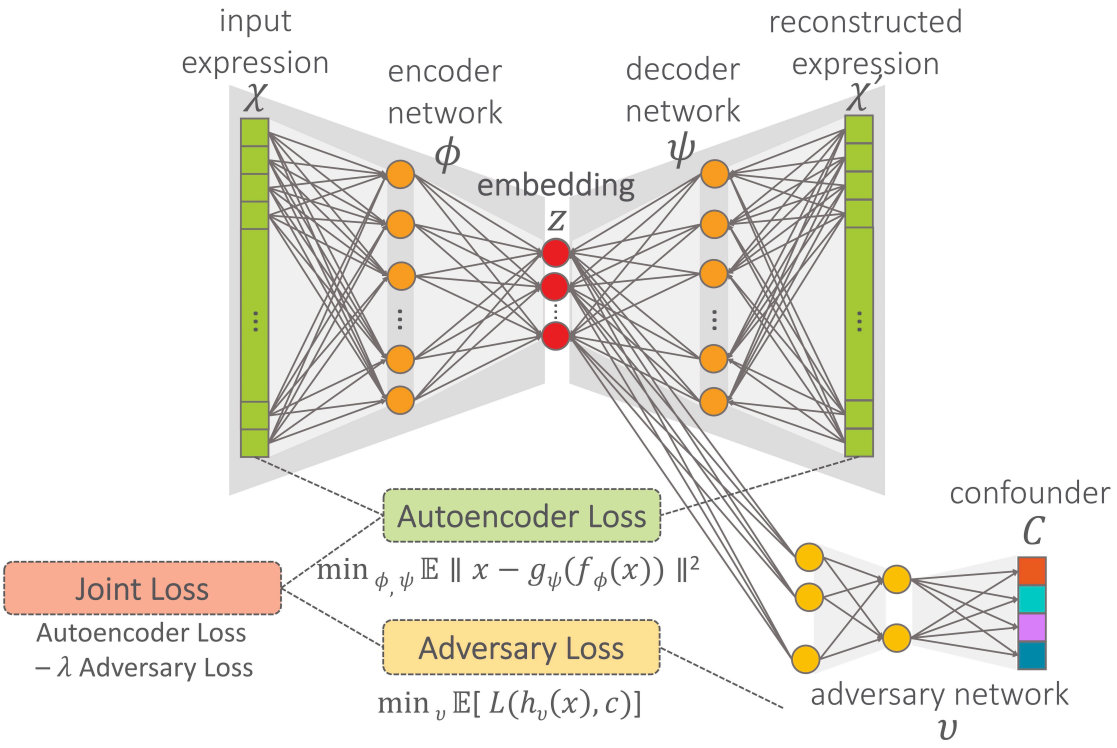

Figure 3.3: **Adversarial deconfounding autoencoder (AD-AE) architecture.** The model consists of an autoencoder and an adversary network. We jointly optimize the two models to minimize the joint loss, defined as the combination of reconstruction error and adversary loss.

for an entire epoch to *minimize* equation 3.2. We continue this alternating training process until both models are optimized. If no model is simultaneously optimal at reconstructing the input expression without encoding confounding signals, the λ variable determines the ratio of weight the model gives to reconstruction or deconfounding. Increasing the λ value would learn a more deconfounded embedding while sacrificing reconstruction success; decreasing it would improve reconstruction at the expense of potential confounder involvement. For our experiments, we set $\lambda = 1$ since we believe this value maintains a reasonable balance between reconstruction and deconfounding.

3.2.3 Datasets and use cases

We demonstrate the broad applicability of our model by employing it on two different expression datasets and experimenting with three different cases of confounders. Our method aims to both remove confounders from the embedding and encode as much biological signal as possible. Accordingly, we evaluate our model using two metrics: (i) *how successfully the embedding can predict the confounder*, where we expect a prediction performance close to random, and (ii) *the quality of prediction of biologically relevant variables*, where a better model is expected to lead to more accurate predictions.

Our first dataset was KMPlot [109], which offers a collection of breast cancer expression datasets from GEO microarray studies [27]. We selected five GEO datasets with the highest number of samples from KMPlot, yielding a total of 1,139 samples and 13,018 genes (GEO accession numbers: GSE2034, GSE3494, GSE12276, GSE11121, and GSE7390). The confounder variable, the dataset label that was a categorical variable, indicated which of the five datasets each subset came from. For this dataset, we chose estrogen receptor (ER) and cancer grade as the biological variables of interest, since both are informative cancer traits. ER is a binary label that denotes the existence of estrogen receptors in cancer cells, an important phenotype for determining treatment [130]. Similarly, cancer grade can take values 1, 2, or 3 for invasive breast cancer, an indicator of the differentiation and growth speed of a tumor [131].

Our second dataset was brain cancer (glioma) RNA-Seq expression profiles obtained from TCGA, which contained lower grade glioma (LGG) and glioblastoma multiforme (GBM) samples [132; 133; 134]. We had a total of 672 samples and 20,502 genes. For this dataset, we used two different confounder variables as two separate use cases: sex as a binary confounder, and age as a continuous-valued one. For the biological trait, we used cancer subtype label, a binary variable indicating whether a patient had LGG or GBM, the latter a particularly aggressive subtype of glioma.

We preprocessed both datasets by applying standard gene expression preprocessing steps: mapping probe ids to gene names, log transforming the values, and making each gene zero-mean univariate. We also applied k-means++ clustering [135] on the expression data before training autoencoder models to reduce the number of features and decrease model complexity (e.g., 13,082,068 trainable parameters for the all genes model compared to 1,052,050 trainable parameters for the 1,000 cluster centers model for KMPlot expression). We observed improvement in autoencoder performance when we applied clustering first and

passed cluster centers to the model (e.g., KMPlot expression validation reconstruction error of 0.624 for the all genes model compared to 0.522 for the 1,000 cluster centers model). All alternative approaches are trained on the same k-means++ clustered expression measurements passed to AD-AE model to ensure fair comparison. We further investigated the effect of the number of clusters on the AD-AE embedding and showed that AD-AE can learn biologically-informative embeddings independent of the number of clusters we train the model on (Chapter B.1 and Figure B.1).

3.2.4 Deep learning architecture

For each dataset, we applied 5-fold cross-validation to select the hyperparameters of autoencoder models. When training the model, we left out 20% of the samples for validation and determined the optimal number of epochs based on validation loss. We used the same autoencoder architecture for the AD-AE as well. The optimal number of latent nodes might differ based on the dataset and the specific tasks the embeddings will be employed on; we tried to select a reasonable latent embedding size with respect to the number of samples and features we had such that we reduce the dimension of the input features by 10%. To demonstrate that our model is invariant to the embedding size, we experimented with various sizes ranging from 10 to 150 and observed that independent of the number of latent nodes, AD-AE can learn deconfounded biologically meaningful embeddings (Chapter B.2 and Figure B.2). In terms of how to determine the number of latent nodes for new datasets and analyses, we refer to the review by Way *et al.* [136], which investigated the effect of the number of latent dimensions using multiple metrics on a variety of dimensionality reduction techniques.

For the breast cancer data, we extracted 1,000 k-means cluster centers since the number of samples was slightly above 1,000. The latent space size was set to 100. Our selected model had one hidden layer in both encoder and decoder networks, with 500 hidden nodes and a dropout rate of 0.1. The minibatch size was 128, and we trained with Adam optimizer [137] using a learning rate of 0.0005. ReLU activation was applied to all layers of the encoder and decoder except the last layer, where we applied linear activation. For the adversarial model, we used a fully connected neural network that had 2 hidden layers with 100 hidden nodes in each layer, and we used ReLU activation. The last layer had 5 hidden nodes corresponding to the number of confounder classes and softmax activation. The adversarial model was trained with categorical

cross entropy loss.

The architecture selected for brain cancer expression was very similar, with 500 k-means cluster centers, 50 latent nodes, one hidden layer with 500 nodes in both networks with no dropout, and ReLU activation at all layers except the last layers of the networks; the remaining parameters were the same as those for the breast cancer network. The adversarial model was also the same except for 50 hidden nodes in each layer. For the sex confounder, the last layer had 1 hidden node with sigmoid activation, trained with binary cross entropy loss; for the age confounder, the last layer used linear activation, trained with mean squared loss.

We implemented AD-AE using Keras with Tensorflow background.

3.2.5 Alternative approaches to deconfounding

When evaluating our model, the most straightforward competitor was a standard autoencoder, which allowed us to directly observe the effects of confounder removal. We also compared against other commonly used approaches to confounder removal. For all these different techniques, we first applied the correction method and then trained an autoencoder model to generate an embedding from the corrected data. We could not compare against nonlinear batch effect correction techniques (Chapter 3.3) since they were applicable only on binary confounder variables.

Batch mean-centering [111]: subtracts the average expression of all samples from the same confounder class (e.g., batch) from the expression measurements.

Gene standardization [112]: transforms each gene measurement to have zero mean and one standard deviation within a confounder class.

Empirical Bayes method (ComBat) [88]: matches distributions of different batches by mean and deviation adjustment. To estimate the mean and standard deviation for each confounder class, the model adopts a parametric or a non-parametric approach to gather information about confounder effects from groups of genes with similar expression patterns.

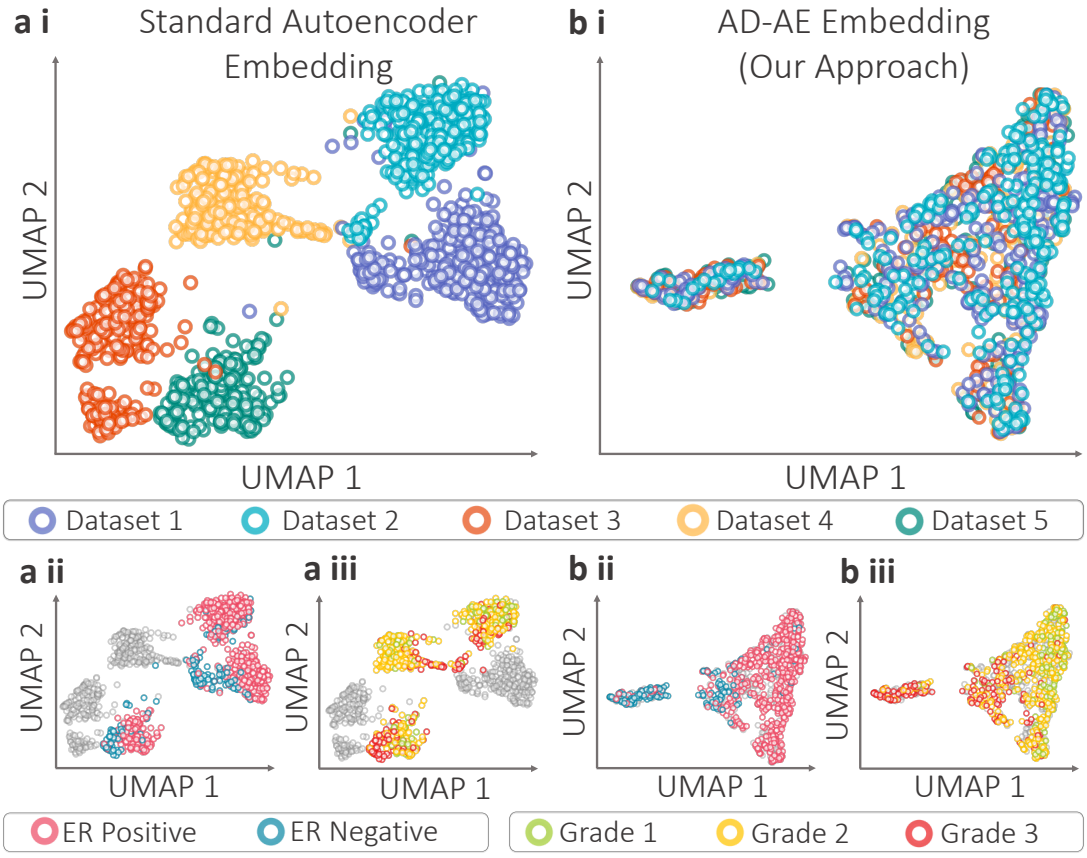

Figure 3.4: **UMAP plots of AD-AE embeddings.** UMAP plots of embeddings generated by (a) standard autoencoder, and (b) Adversarial Deconfounding Autoencoder (AD-AE). Subplots are colored by (i) dataset, **(ii)** ER status, and (iii) cancer grade. The gray dots denote samples with missing labels.

3.3 Results

3.3.1 Adversarial Deconfounding Autoencoder learns biologically meaningful deconfounded embeddings

Our first experiment aimed to demonstrate that AD-AE could successfully encode the biological signals we wanted while not detecting the selected confounders. We used the KMPlot breast cancer expression dataset and trained standard autoencoder and AD-AE to create embeddings, and generated UMAP plots [138] to visualize the embeddings (Figure 3.4). Observe that the standard autoencoder embedding clearly separates datasets, indicating that the learned embedding was highly confounded (Figure 3.4a i). On the other hand, the UMAP plot for AD-AE embedding shows that data points from different datasets are fused (Figure

3.4b i). This is expected since we trained our model until both networks converged, which means that we obtained a random prediction performance on the validation set for the adversarial network.

More interestingly, we colored the UMAP plots by biological variables of interest: ER status and cancer grade. Observe that for the autoencoder embedding, the samples are not differentiated by phenotype labels (Figure 3.4a ii&iii). This shows that when we learn an embedding with a standard autoencoder model, confounders might dominate the embedding, preventing it from learning clear biological patterns. On the other hand, the UMAP plot of the AD-AE embedding clearly distinguishes samples by ER label as well as cancer grade (Figure 3.4b ii & iii), showing the effects of deconfounding. We further investigate these results in Section 3.3.3 by fitting prediction models on the embeddings to quantitatively evaluate the models.

3.3.2 AD-AE can learn embeddings generalizable to different domains

AD-AE generates embeddings that are robust to confounders and generalizable to different domains. The most common applications for this model are learning an embedding from a dataset and transferring it to a separate dataset. To simulate this problem with breast cancer samples, we left one dataset out for testing and trained the standard autoencoder on the remaining four datasets. We then generated two embeddings for the internal and external datasets: (i) one for samples from the four datasets used for training, and (ii) another for the left out samples from the fifth dataset. Note that we trained the model using samples in the four datasets only, and we then used the already trained model to encode the fifth dataset samples. We then repeated the same training and encoding procedure for AD-AE to compare the generalizability of both models.

In Figure 3.5, the circle and diamond markers denote the UMAP representation of the embedding generated for training and left-out dataset samples, respectively. In Figure 3.5a i, we colored all samples by their ER labels. First of all, we draw attention to the external set data points that are clustered entirely separately from the training samples. This shows that the standard embedding does not precisely generalize to left-out samples. More importantly, we do not see a general direction of separation for the ER labels that is valid for both the training and left-out samples; (ER+ samples are clustered on the right for training samples and mainly on the left for external samples). This clustering indicates that the manifold learned for the training samples does not transfer to the external dataset. We observed the same scenario when we colored the same

plots by cancer grade (Figure 3.5a ii).

To examine whether the adversarial deconfounding autoencoder can better generalize to a separate dataset, we created UMAP plots (as in Figure 3.5a) for the AD-AE embedding (Figure 3.5b). We emphasize that it is not possible to distinguish training from external samples because the circle and diamond markers overlap one another. But the critical point is the separation of samples by ER label (Figure 3.5b i). Observe that ER- samples from the training set are concentrated on the upper left of the plot, while ER+ samples dominate the right. The same direction of separation applies to the samples from the external dataset. This plot concisely demonstrates that when we remove confounders from the embedding, we can learn generalizable biological patterns otherwise overshadowed by confounder effects. We applied the same analysis using the cancer grade labels and again observed the same pattern (Figure 3.5b ii).

3.3.3 AD-AE can successfully predict biological phenotypes

To show that AD-AE preserves the true biological signals present in the expression data, we predicted cancer phenotypes from the learned embeddings. In Chapters 3.3.1 and 3.3.2, we visualized our embeddings to demonstrate how our approach removes confounder effects and learns meaningful biological representations. Nonetheless, we wanted to offer a quantitative analysis as well to thoroughly compare our model to a standard baseline and to alternative deconfounding approaches. After generating embeddings with AD-AE and competitor models, we fit prediction models to the embeddings to predict biological phenotypes of interest. We also applied the prediction test on different domains to examine how well the learned embeddings generalized to external test sets and measure the generalization gap for each model as a metric of robustness.

In Figure 3.6a, we show the ER prediction performance of our model compared to all other baselines. To predict ER status, we used an elastic net classifier, tuning the regularization and l1 ratio parameters with 5-fold cross validation. We recorded the area under precision-recall curves (PR-AUC) since the labels were unbalanced. We separately selected the optimal model for each embedding generated by AD-AE and each competitor. To measure each method's consistency, we repeated the embedding generation process 10 times with *10 independent random trainings of the models*, and we ran prediction tasks for each of the 10 embeddings for each model. We used linear models for the prediction for two reasons. First, the sample size was small due to the missingness of phenotype labels for some samples and the splitting of samples across

domains, which made it difficult to fit complex models. Second, reducing the expression matrix dimension size let us reduce complexity and fit simpler models to capture patterns.

We trained AD-AE and the competitors using only four datasets, leaving the fifth dataset out. To train the linear prediction model, we left out 20% of the samples from the four datasets for testing, trained the model using the rest of the samples, then predicted on the left-out *internal* samples to measure PR-AUC. To measure prediction performance of the external dataset, we used the exact same training samples obtained from the four datasets and then predicted for the *external* dataset samples. In Figure 3.6a i, observe that for the internal dataset, our model barely outperforms other baselines and the uncorrected model. This is expected: when the domain is the same, we might not see the advantage of confounder removal. However, Figure 3.6a ii shows that when predicting for the left-out dataset, AD-AE clearly outperforms all other models. This result shows that AD-AE much more successfully generalizes to other domains.

We repeated the same experiments, this time to predict cancer grade, for which we fit an elastic net regressor tuned with 5-fold cross validation, measuring the mean squared error. Figure 3.6b shows that for the internal prediction, our model is not as successful as other models; however, it outperforms all baselines in terms of external test set performance. This result indicates that a modest decrease in internal test set performance could significantly improve our model's external test set performance. We further investigated the effect of the embedding size on the internal and external test set prediction performances and showed that AD-AE can successfully predict biological phenotypes of interest for a wide range of embedding sizes (Chapter B.2 and Figure B.3).

Moreover, we showed that the generalization gap of AD-AE is much smaller than the baselines we compare against (Figure 3.6). We calculated the generalization gap as the distance between internal and external test set prediction scores. A high generalization gap means that model performance declines sharply when transferred to another domain; a small generalization gap indicates a model can transfer across domains with minimal performance decline. Therefore, AD-AE successfully learns manifolds that are valid across different domains, as we demonstrated for both ER and cancer grade predictions.

3.3.4 AD-AE embeddings can be successfully transferred across domains

We next extend our experiments to the TCGA brain cancer dataset to further evaluate AD-AE. We trained our model and the baselines with the same procedure we applied to the breast cancer dataset and again fitted prediction models. We first trained an elastic net classifier to predict cancer subtype (LGG vs GBM) from the embeddings. We trained the predictor model using only female samples and predicted for male samples. We then repeated this transfer process, this time training from male samples and predicting on females. Note that the autoencoder was trained from all samples (male and female), and prediction models were trained from one class of samples (e.g., males) and transferred to another class (e.g., females). This experiment was intended to evaluate how accurate an embedding would be at predicting biological variables of interest when the confounder domain is changed. Figure 3.7 shows that AD-AE easily outperforms the standard baseline and all competitors for both transfer directions.

We find this result extremely promising since we offer confounder domain transfer prediction as a metric for evaluating the robustness of an expression embedding. Researchers want to generate informative embeddings that encode biological signals without being confounded by out-of-interest variables (e.g., sex). We note that the confounder variable is data and domain dependent, and sex can be a crucial biological variable of interest for certain diseases or datasets. In this experiment, we wanted to learn about cancer subtypes and severity independent of a patient's sex. We succeed at this task of accurately predicting complex phenotypes regardless of the distribution of the confounder variable. We also highlight that our model can solve the problem of class imbalance that commonly occurs in domain shift [139]. Figure 3.7c shows that the distribution of cancer subtypes differs for male and female domains. This might lead to discrepancies when transferring from one domain to another; however, AD-AE embeddings could be successfully transferred independent of the distribution of labels, a highly desirable property of a robust expression embedding. We also subsampled from the subtype classes to carry out the transfer experiments on the simulated balanced dataset and demonstrated that AD-AE could successfully transfer across domains in both cases of balanced and imbalanced class distributions (Chapter B.3 and Figure B.4).

We repeated the transfer experiments using age as the continuous-valued confounder variable. Other models were not applicable for continuous valued confounders; thus, we could compare only to the standard baseline. For the prediction transfer experiments, we again fit an elastic net classifier to predict cancer sub-

type and separated the samples into two groups: samples with age within one standard deviation (i.e., center of the distribution), and samples with age beyond one standard deviation (i.e., edges of the distribution). Figure 3.8c shows the age distribution of the brain cancer dataset, highlighting the samples in the center and on the edges. We trained the predictor on the center samples and predicted for samples on the edge, and vice versa (Figure 3.8a & b). Our model substantially outperforms the standard baseline in both transfer directions. Especially, when we trained on samples within one standard deviation and predicted for remaining samples, we can see a huge increase in performance compared to the standard baseline. This case simulates a substantial age distribution shift. It is promising to see that disentangling confounders from expression embeddings can be the key to capturing signals generalizable over different domains, such as different age distributions.

3.4 Discussion

Gene expression datasets contain valuable information central to unlocking biological mechanisms and understanding the biology of complex diseases. Unsupervised learning aims to encode information present in vast amounts of unlabeled samples to an informative latent space, helping researches discover signals without biasing the learning process. Hindering the learning of meaningful representations is the fact that gene expression measurements often contain unwanted sources of variation, such as experimental artefacts and out-of-interest biological variables. Variations introduced by these confounders can overshadow the true expression signal, preventing the model from learning accurate patterns. Particularly when we combine multiple expression datasets to increase statistical power, we can learn an embedding that encodes dataset differences rather than biological signals shared across multiple datasets.

In this chapter, we addressed such *batch effects and confounders* observed in gene expression measurements and introduced the Adversarial Deconfounding Autoencoder (AD-AE) to generate expression embeddings robust to confounders. AD-AE trains two neural networks simultaneously, an autoencoder to generate an embedding that reconstructs the original data successfully and an adversary model that predicts the selected confounders from the generated embedding. We jointly optimized the two models; the autoencoder tries to learn an embedding free from the confounder variable, while the adversary tries to predict the confounder accurately. On convergence, the encoder learns a latent space where the confounder cannot be

predicted even using the optimally trained adversary network.

We evaluated our model based on (1) deconfounding of the learned latent space, (2) preservation of biological signals, and (3) prediction of biological variables of interest when the embedding is transferred from one confounder domain to another. We experimented with two datasets, KMPlot breast cancer expression, where we used dataset labels as the confounder variable, and TCGA brain cancer RNA-Seq expression, where we used both sex and age as separate confounders. For these different use cases, we showed that AD-AE generates deconfounded embeddings that successfully predict biological phenotypes of interest. Importantly, we showed the advantage of our model over standard autoencoder and alternative deconfounding approaches on transfer experiments, where our model generalized much better to different domains.

A potential limitation of our approach is that we extend an unregularized autoencoder model by incorporating an adversarial component. We can improve our model by adopting a regularized autoencoder such as denoising autoencoder [99], or variational autoencoder [45]. In this way, we could prevent model overfitting and make our approach more applicable to datasets with smaller sample sizes. Another limitation is that although our model can train an adversary model to predict a vector of confounders, we have not yet conducted experiments to correct for multiple confounders simultaneously. We could extend our model by incorporating multiple adversarial networks to account for various confounders.

AD-AE is an adversarial approach for generating confounder-free embeddings for gene expression that can be easily adapted for any confounder variable. In this chapter, we tested our model on cancer expression datasets since cancer expression samples are available in large numbers. However, we would like to extend testing to other expression datasets as well, including samples from different diseases and normal tissues. We also see as future work experimenting on single cell RNA-Seq data to learn informative embeddings combining multiple datasets.

Furthermore, investigating the deconfounded latent spaces and reconstructed expression matrices learned by AD-AE, as we have done in Chapter 2, using feature attribution methods such as 'expected gradients' [140; 141] would allow us to detect the biological differences between the confounded and deconfounded spaces and carry enrichment tests to understand the relevance to biological pathways.

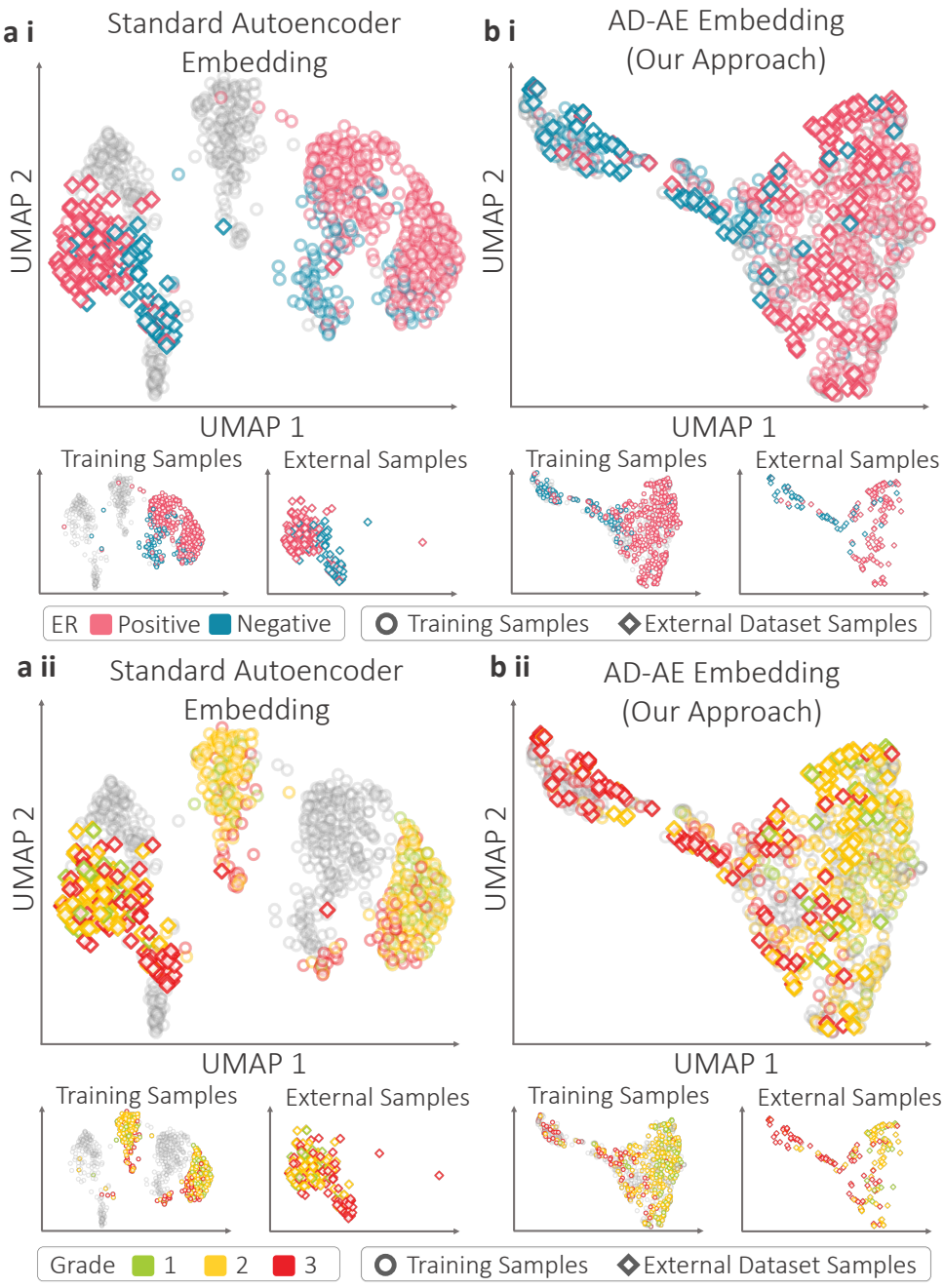

Figure 3.5: **UMAP plots of AD-AE external dataset embeddings** UMAP plots of embeddings generated by (a) standard autoencoder, and (b) Adversarial Deconfounding Autoencoder. Plots are colored by (i) Estrogen Receptor (ER) labels, and (ii) cancer grade labels. The circle and diamond markers denote training and external dataset samples, respectively. The gray dots denote samples with missing labels. For clarity, the subplots for the training and external samples are provided below the joined plots.

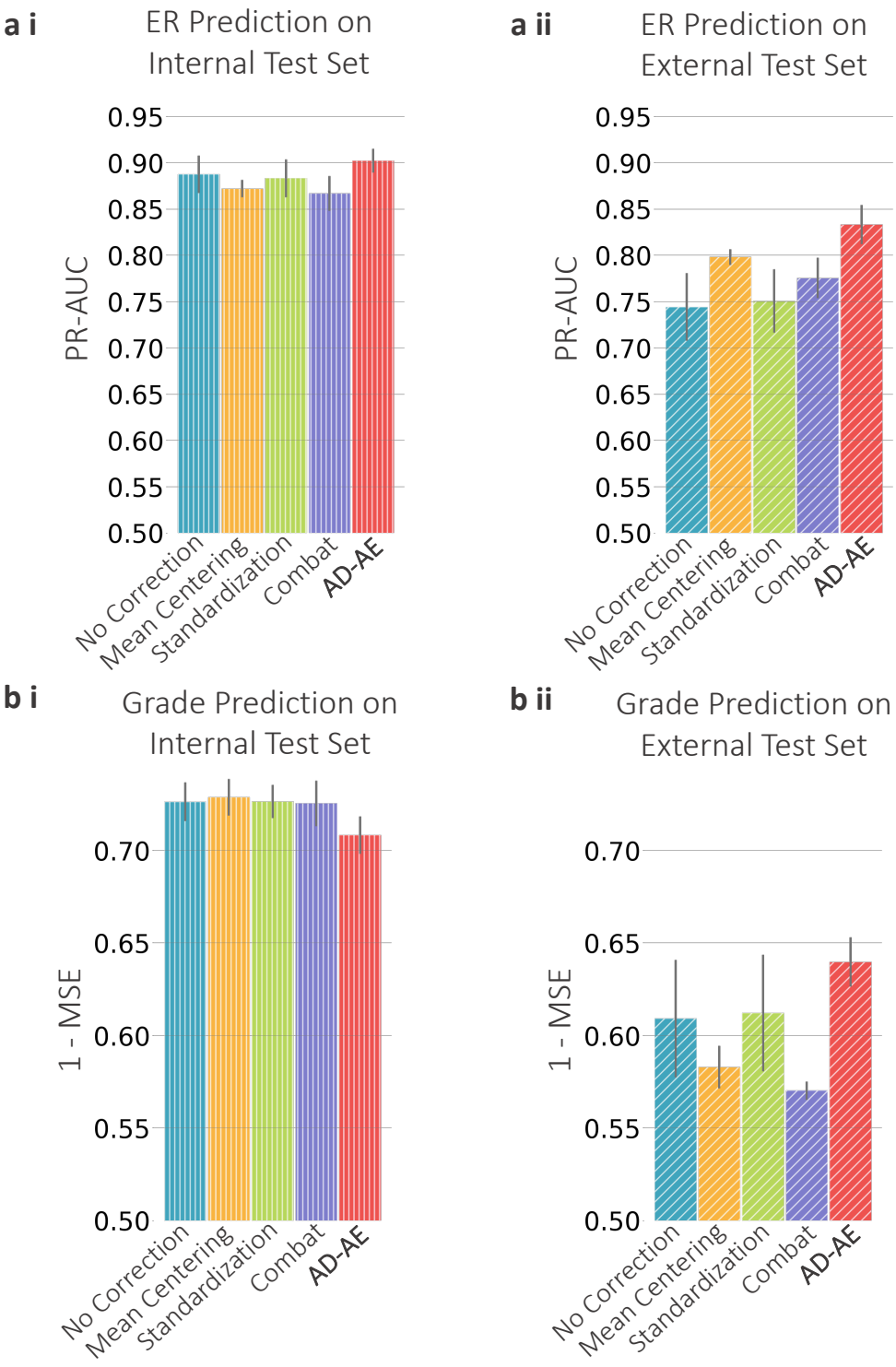

Figure 3.6: **Phenotype prediction plots.** (a) Estrogen Receptor (ER) prediction plots for (i) internal test set, and (ii) external test set. (b) Cancer grade prediction plots.

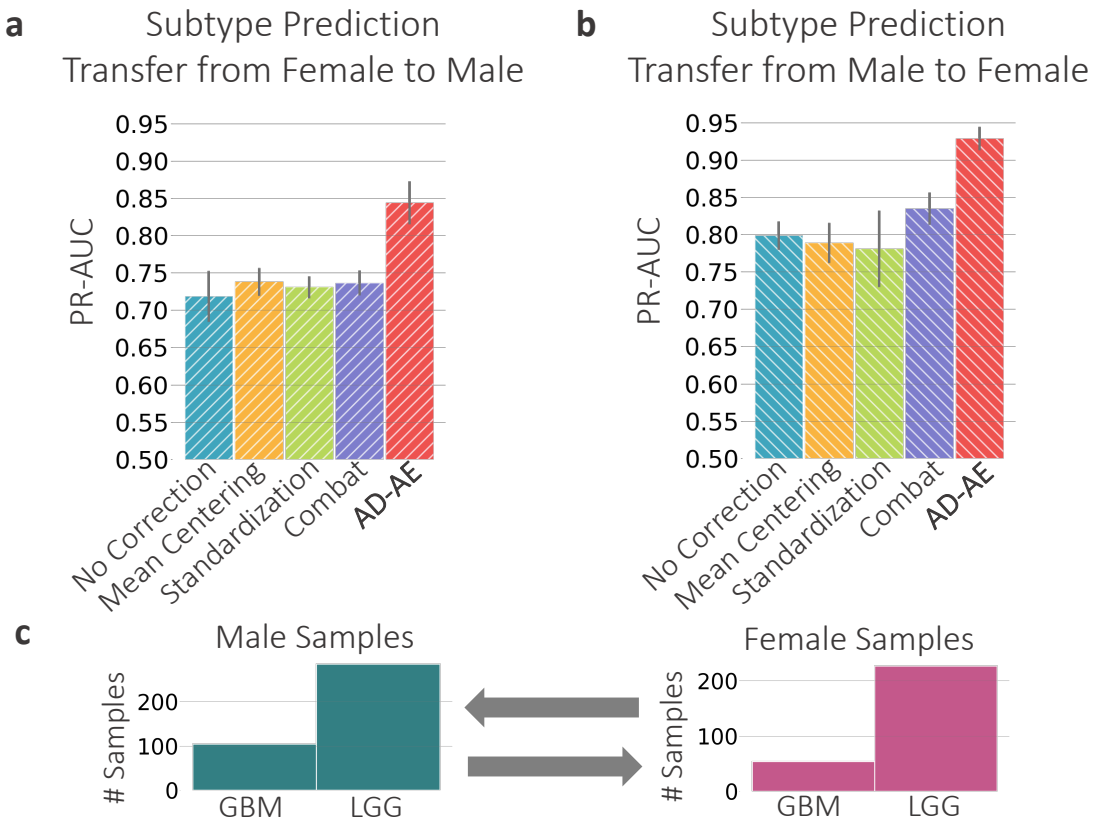

Figure 3.7: **AD-AE: Transferring AD-AE embeddings.** Glioma subtype prediction plots for (a) model trained on female samples transferred to male samples and (b) model trained on male samples transferred to female samples. (c) Subtype label distributions for male and female samples.

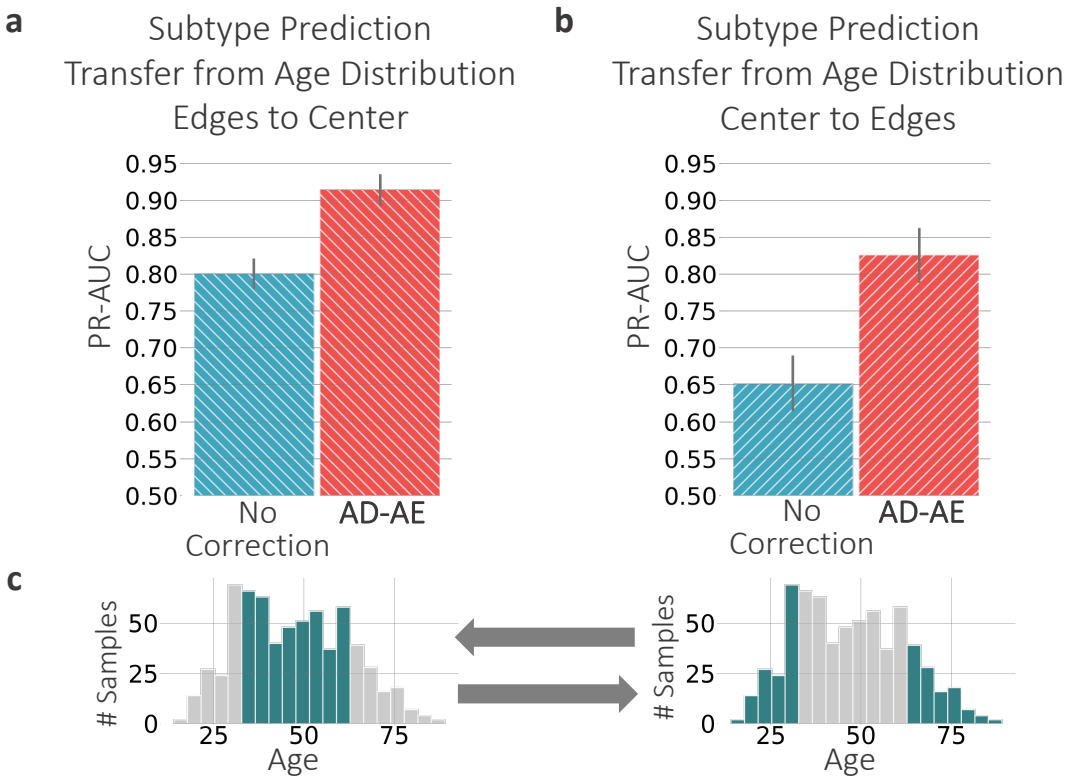

Figure 3.8: **AD-AE: Transferring AD-AE embeddings across different age groups.** Glioma subtype prediction plots for (a) model trained on samples beyond one standard deviation of the age distribution (i.e., edges of the distribution) transferred to samples within one standard deviation (i.e., center of the distribution), and (b) vice versa. (c) Age distributions of all samples. Comparison to other approaches was not possible due to inapplicability of these methods on continuous-valued confounders.

Chapter 4

Reducing peptide sequence bias in quantitative mass spectrometry data with machine learning

The work presented in this chapter is adapted from the manuscript [142] accepted for publication by *Journal of Proteome Research*, available in *bioRxiv*:

Ayse B. Dincer, Yang Lu, Devin Schweppe, Sewoong Oh, William Stafford Noble. Reducing peptide sequence bias in quantitative mass spectrometry data with machine learning. *bioRxiv* 2022.04.11.487945; doi: https://doi.org/10.1101/2022.04.11.487945.

4.1 Background

In Chapters 2 and 3, we focused on gene expression profiles and developed deep learning approaches to address the fundamental issues preventing us from capturing biological signals associated with these measurements. While gene expression data allows investigating cellular mechanisms and reveals key insights into disease, the abundances of transcripts, mRNAs, cannot precisely reflect the protein abundances. Understanding the protein expression patterns is important since proteins take part in essentially all cellular function ranging from DNA replication and protein production to controlling cell division and regulating

the metabolism [143]. As discussed extensively in Chapter 1, while related, gene and protein expressions are not highly correlated, highlighting that the protein abundances themselves might reveal biological signals that are not possible to detect by studying gene expression profiles alone. Thus, studying and understanding the proteome, the entire set of proteins functioning in a certain location in the cell at a specific time, is crucial for understanding the cellular mechanisms.

Mass-spectrometry-based (MS) methods make it possible to identify and quantify thousands of peptides in a complex biological mixture. While there are techniques to directly identify and quantify proteins, due to many challenges associated with analysis at the protein level, bottom-up workflows are prevalent [144]. In bottom-up analysis, proteins are first digested into peptides, which are much smaller molecules that can be more efficiently detected [145]. After sample preparation, fragmentation, protein excision, and digestion of the proteins into peptides, the peptides are separated (e.g., using liquid chromatography column) to be passed to the mass spectrometer [146]. A mass spectrometer, the key instrument in a proteomics experiment, ionizes these peptides (e.g., using electrospray ionization) and separates them by mass using electric or magnetic field. The output of the experiment is a mass spectrum (MS1 spectrum) showing the intensity for a range of mass-to-charge ratios [146]. The peptides are then fragmented for the tandem mass spectrometry (MS/MS) stage which outputs a second spectrum (MS2 spectrum) [145]. This tandem mass spectra are searched across peptide sequence databases such that peptide sequences can be identified and matched with proteins. In the end, we get a list of peptides and proteins that exists in the mixture.

A mass spectrometer can also be used for quantifying the peptides to obtain the absolute concentration of a protein or the relative abundances between pairs of samples [147]. Many different techniques exist for quantification of peptides, such as label-free quantification by consecutive MS runs, quantification with heavy isotope labeling (e.g., SILAC), or multiplexed proteomics using isobaric mass tags [147]. Especially sample-multiplexing approaches, such as tandem mass tag (TMT), allow quantifying over 10 samples within one run, enabling efficient and reliable quantification of thousands of peptides from large number of samples.

Such an improvement in protein quantification increased the number of large-scale proteomics datasets

such as Clinical Proteomic Tumor Analysis Consortium (CPTAC) with over 500 cancers profiled [148] or Cancer Cell Line Encyclopedia (CCLE) dataset consisting of 375 cancer cell lines with protein expression [149]. The growing number of datasets makes it possible to investigate the changes in protein abundance in response to environmental changes, disease, or different experimental conditions, which can lead to a better understanding of pathogenicity mechanisms and functional pathways [150]. Proteomics analysis further facilitates the detection of diagnostic markers and generation of candidates for vaccine production [150] as well as enabling the study of protein modules and networks [144]. Mass spectrometry is also the only method for studying post translational modifications (PTMs). Identification and characterization of protein modifications (e.g., phosphorylation, ubiquitylation, and methylation) are essential since they can play critical roles in signaling and cell regulation. Other applications of proteomics include analysis of protein structure, inference of proteotype-phenotype and proteotype-genotype associations [144].

While mass spectrometry experiments enable quantifying thousands of peptides in a complex biological mixture for the exploration of cellular mechanisms, it is quite challenging to accurately detect and quantify proteins. A mass spectrometry experiment involves many different stages, each with its unique shortcomings and biases. Thus, these measurements are prone to *experimental noise and biases* that impair quantification accuracy, yielding peptide quantities that are systematically under-/over-estimated or that fluctuate from run to run. All quantitative measurements from MS experiments, regardless of how the quantitative values are extracted from the data—using labeling strategies such as iTRAQ or TMT, peak areas from precursor scans, or spectral counts— exhibit biases. In general, "bias" in our context means that quantitative measurements from a mass spectrometry experiment are systematically skewed, either positively or negatively, relative to the true abundance of the measured molecular species. Some of these biases depend in part on properties of the peptide sequence, such as how susceptible the peptide is to enzymatic cleavage, how efficiently the peptide traverses the liquid chromatography column, how easily the peptide ionizes in electrospray, and how easily and uniformly the peptide fragments in the dissociation phase of the MS/MS. In this chapter, we address these peptide-specific biases that occur in proteomics experiments, with the goal of improving quantification accuracy.

Numerous efforts have been made to identify and quantify these sequence-specific effects [151; 152;

153; 154; 155; 156; 157]. For example, a protein's "proteotypic peptides"—peptides that are repeatedly and consistently identified for a given protein—can be identified using machine learning methods that take into account a wide range of physicochemical properties of amino acids, including charge, secondary structure propensity and hydrophobicity. Peptide hydrophobicity, in particular, strongly affects ionization in electrospray settings [158; 159].

In this chapter, we focus on biases that arise directly from the amino acid sequence itself. Many biases exist that our approach is not designed to address. This includes, for example, the effect of secondary or tertiary protein structure, which could inhibit cleavage by trypsin, as well as competitive effects due to chromatographic coelution of other peptides. We focus on sequence-induced biases because our machine learning framework depends on peptide-level labels, as we describe below.

Our goal is to train a machine learning model to quantify these peptide-specific properties, with the aim of adjusting the observed quantities to remove these effects. Our approach rests on two primary assumptions. First, we assume that each peptide is measured in its linear dynamic range and that the observed measurement q_{ik} of peptide i in run k can be decomposed into a peptide coefficient c_i and an adjusted peptide abundance α_{ik} such that $q_{ik} = c_i \alpha_{ik}$. Second, we assume that unique sibling peptides, i.e., peptides that occur exactly once in the protein database and that co-occur on a given protein sequence, should have equal abundances within a single MS/MS run. We use these assumptions to train a deep neural network, Pepper, that takes as input a peptide sequence p_i and charge state z_i and produces as output the corresponding peptide coefficient c_i, thereby revealing the adjusted peptide abundance α_{ik}.

We demonstrate that, by removing peptide-specific effects from the observed MS/MS quantities, the adjusted abundance values α_{ik} provide more accurate abundance measurements than the observed values. First, we provide empirical evidence for the consistency of peptide coefficients inferred from disjoint sets of runs, and we show that a Pepper model trained to predict these coefficients can generalize to new peptides in new runs. We demonstrate the robustness of the approach to the type of noise that one would expect to arise from the presence of proteoforms in the sample that are not represented in the reference proteome database that was used during the original processing of the data. We show that the learned coefficients exhibit significant correlation with several key peptide properties, in agreement with previous research [151; 152; 153; 154; 155; 157; 156]. We also show that the adjusted peptide abundances yield a small but highly

	Year	Model	Experiment type
Mallick *et al.* [151]	2006	Gaussian mixture model	DDA
Sanders *et al.* [152]	2007	Neural network	DDA
Webb-Robertson *et al.* [153]	2008	Support vector machine	DDA
Fusaro *et al.* [154]	2009	Random forest	Targeted MS
CONSeQuence [155]	2011	Ensemble model	Absolute quantification
PREGO [156]	2015	Neural network	Targeted MS
PPA [157]	2015	Neural network	DDA

Table 4.1: **Methods for predicting proteotypic peptides.**

significant improvement in the degree of correlation between MS/MS and mRNA-based gene expression measurements. We provide The Apache licensed Pepper source code for training models from MS/MS data, which can be used to de-bias any given matrix of peptide-level abundances: (`http://github.com/Noble-Lab/Pepper`).

4.1.1 Related work

The measurements produced by a mass spectrometry experiment invariably exhibit biases. In particular, among the many thousands of distinct tryptic peptides in a complex biological mixture, only a small fraction are typically observed [160], and some peptides are preferentially identified regardless of whether they occur on the most abundant proteins [151]. These commonly observed peptides are called *proteotypic peptides*, and automatically identifying them can be important for accurate protein detection and quantification [151]. In particular, better understanding of the biases underlying proteotypicity can lead to changes in experimental design, and knowing the potentially observable peptides for a protein beforehand might increase our confidence in the identification of missing proteins in a sample [152]. Accordingly, numerous methods have been developed to predict proteotypic peptides using machine learning.

The first such method used physicochemical properties of amino acids summarized at the peptide level and selected properties that can most successfully differentiate observed versus unobserved peptides using Kolmogorov-Smirnov and Kullback-Leibler distances [151]. Using these most discriminative features, Mallick *et al.* fitted a Gaussian mixture likelihood function to predict the probability of detection for each peptide, thereby achieving a test accuracy above 85%. The trained model showed robust performance across various datasets and organisms.

Thereafter, a series of machine learning methods were developed to improve the accuracy of proteotypic peptide prediction and to target the predictions to particular applications (Table 4.1). Sanders *et al.* [152] developed a neural network-based predictor, which can encode non-linear relations among the input features. Similarly, Webb-Robertson *et al.* [153] developed a non-linear support vector machine classifier to predict proteotypic peptides, after first applying the Fisher Criterion Score to select a subset of peptide features to train on. Unlike previous approaches, they jointly predict proteotypicity (i.e., detection probability) and the elution time of a peptide. Several methods focused on predicting which peptides will be most useful in a targeted MS setting. In one such study, Fusaro *et al.* used a random forest classifier for the prediction of high-responding peptides to be used as targets [154], and Searle et al.'s PREGO model [156] adopted a similar approach, using data independent acquisition (DIA) fragment intensities to define their training set. The Peptide Prediction with Abundance (PPA) method also uses a neural network and predicts not only the probability of detection for the peptide but also the corresponding protein abundance that would enable detecting that peptide [157]. Finally, CONSeQuence uses a consensus machine learning approach—an ensemble of support vector machine, random forest, genetic programming, and neural network models—to select peptides for absolute quantification experiments [155].

All of the studies cited above have focused on identifying proteotypic or high-responding peptides. In contrast, our goal is to quantify the peptide biases in quantitative MS experiments. Thus, rather than classifying the peptides as low-responding versus high-responding, we aim to learn peptide coefficients that can allow us to reduce bias in our measurements. To our knowledge, ours is the first machine learning approach to quantifying sequence-induced bias for mass spectrometry.

Another key point that differentiates our work from the previous studies is that instead of using the physicochemical properties of the amino acids, we take the peptide sequence itself as input. Using the physicochemical properties of peptides has many intrinsic limitations. Most importantly, all previous studies used the amino acid properties summarized at the peptide level (e.g., using the mean or sum) which results in losing sequence-specific information. In practice, small changes in the order of amino acids within a peptide sequence might significantly affect how the peptide behaves in a mass spectrometer. Therefore, our model can potentially explain biases that a model trained solely on summary-level peptide properties might fail to capture. Our sequence-based approach has the added advantage that we do not need to worry about

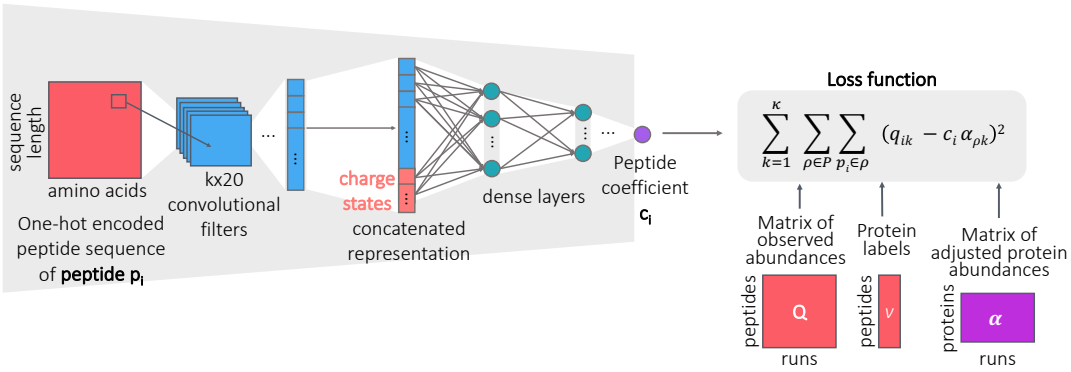

Figure 4.1: **Peptide coefficient predictor**. The neural network architecture for predicting peptide coefficients. The network takes as input the one-hot encoded peptide sequence and the charge state, and runs through convolutional layers followed by densely connected layers to output a peptide coefficient.

pre-processing large tables of peptide features in order to reduce redundancy, as was done in many previous studies [151; 152; 153; 156; 155].

4.2 Methods

4.2.1 A neural network model for predicting peptide coefficients

We designed a neural network architecture, Pepper, that aims to learn peptide coefficients from quantitative mass spectrometry measurements (Figure 4.1). The inputs to Pepper are a matrix of measured peptide abundances and corresponding databases of peptides and proteins. Say that we are given a matrix Q of peptide measurements, where q_{ik} is the observed abundance of peptide i in run k. We column normalize Q so that the sum of all abundances within a given run is constant across runs. We are also given a peptide database P and corresponding protein database $\mathcal{P}$. For the purposes of training our model, we preprocess P to eliminate all peptides that appear in more than one protein. In addition, to reduce issues related to unexpected proteoforms, we identify any peptide that contains a variable modification, such as phosphorylation or oxidation of methionine, and we eliminate both the modified and unmodified form of the peptide from P. Similarly, we eliminate all peptides that overlap one another, due to missed or non-enzymatic cleavages. Finally, we retain in $\mathcal{P}$ only proteins that contain at least two unique peptides. The column normalization was applied to the raw abundances without applying any other transformation and after the filtering of the peptides.

Pepper takes as input one-hot encoded peptide sequence (i.e., a binary representation in which each

		Total		Filtered				
Dataset	Runs	Proteins	Peptides	Proteins	Peptides	#Quants	% missing	Accession
Slevlek *et al.*	18	3,529	34,490	2,669	23,971	269,039	37.65	PXD001010
Thomas *et al.*	103	4,479	25,311	3,199	19,742	884,245	59.65	PXD010437
Guo *et al.*	120	3,171	22,554	2,122	14,472	1,736,638	0.00	PXD003539
CPTAC S16	95	36,350	164,395	4,129	19,151	193,069	89.39	CPTAC S016
CPTAC S19	30	53,143	93,704	5,727	13,583	141,645	65.24	CPTAC S019
CPTAC S37	100	40,089	172,589	3,670	18,392	202,465	88.99	CPTAC S037
CPTAC S47	226	43,984	189,784	3,587	22,601	1,286,418	74.81	CPTAC S047
CPTAC S48	109	67,395	462,909	4,887	42,015	1,773,366	61.28	CPTAC S048
CPTAC S49	330	72,659	585,621	5,768	58,909	6,314,106	67.52	CPTAC S049
CPTAC S51	35	51,632	291,317	3,810	30,184	603,070	42.91	CPTAC S051
CPTAC S54	191	68,240	489,102	5,166	45,184	3,018,826	65.02	CPTAC S054
CPTAC S58	217	75,951	639,233	5,519	51,255	3,556,689	68.02	CPTAC S058
CPTAC S61	248	68,424	618,020	5,232	52,820	3,389,987	74.12	CPTAC S061

Table 4.2: **Pepper datasets.** The table lists datasets used in the study. The numbers of peptides and proteins are given before ("Total") and after ("Filtered") eliminating shared peptides and peptides with missed cleavages, peptides that occur in modified and unmodified forms, and peptides with no siblings.

amino acid is represented using 20 bits, exactly one of which is set to 1), p_i, along with the charge state, and produces as output the corresponding coefficient c_i (i.e., $f(p_i)$). The network is then trained using a loss function that captures our assumption that the adjusted abundances of all sibling peptides should be equal to one another and, thus, should be equal to the adjusted abundance of the corresponding protein, ρ:

$$L(Q, \mathcal{P}, f(\cdot)) = \sum_{k=1}^{\kappa} \sum_{\rho \in \mathcal{P}} \sum_{p_i \in \rho} (q_{ik} - f(p_i)\alpha_{\rho,k})^2 \tag{4.1}$$

where $\alpha_{\rho,k}$ is the adjusted abundance of protein ρ in run k and κ is the number of runs in Q. The model is trained subject to the constraints $\forall i \; c_i > 0$ and $\forall_{\rho,k} \; \alpha_{\rho,k} > 0$. Note that the resulting coefficients can be used to adjust the measured abundances of all peptides, not just unique peptides, via $\alpha_{ik} = \frac{q_{ik}}{c_i}$.

We note that the elements of the α matrix are parameters of our model which we optimize along with the network weights while training the model. When calculating the loss function in Equation 4.1, we initialize the α matrix to the median observed peptide abundance per protein. To obtain the final adjusted protein abundance matrix $\hat{\alpha}$ for a test set, we optimize over the fixed set of Q matrix and predicted c_i values, and we use the resulting $\hat{\alpha}$ to calculate the final loss for the test set.

4.2.2 Data sets

To train and validate Pepper, we downloaded a collection of quantitative proteomics data from a variety of previous studies (Table 2). In most cases, we directly downloaded a matrix of peptide-level quantities. For the CPTAC datasets, we downloaded matrices of PSM-level quantities, for which we followed the steps decsribed below to map to the peptide level. These studies employed a variety of instrument types, acquisition strategies, and processing pipelines, but Pepper is designed to be agnostic to the specifics of the underlying quantitation strategy.

The primary dataset was obtained from Guo *et al.* [161]. The data was generated from NCI-60 cancer cell lines using the PCT-SWATH workflow. Two replicates were obtained for each cell line, resulting in a total of 120 runs. SWATH-MS acquisition was used in a Sciex TripleTOF 5600 mass spectrometer with 32 windows of isolation width of 25. Peptides were detected at a peptide-level FDR threshold of 1% with OpenSWATH, using a human cancer cell line spectral library containing 86,209 proteotypic peptide precursors.

The other datasets were selected to reflect a range of instrument types, experimental protocols, and quantification methods (Table 4.2). The Slevlek *et al.* and Thomas *et al.* datasets provided peptide-level matrices of intensities. The CPTAC datasets were processed to obtain peptide quantities from the PSM files by repeating the preprocessing pipeline implemented by the CPTAC consortium, which starts with selecting the highest observed intensity for each sample and peptide across all fractions. Each sample was analyzed in 24 or 25 fractions, depending on the dataset. For each peptide, we selected the PSM with the highest "TotalAb" (i.e., total intensity across all TMT channels) among all fractions. The result is a peptide-by-sample matrix of measured intensities.

For each dataset, we normalize the peptide measurements so that the sum of peptide abundances is equal across all runs. This normalization is carried out after filtering peptides, as described next.

4.2.3 Filtering peptides

For a given quantitative matrix, we construct a filtered set of unique peptides by reducing the number of rows (peptides) in the matrix in four steps. First, we identify all peptides that occur in more than one protein, and we eliminate these from the matrix. Second, we identify all peptides that occur in both modified

and unmodified forms, and we eliminate these from the matrix. Third, we eliminate all pairs of peptides that overlap one another due to missed cleavages. Fourth, among the remaining peptides, we identify and remove singletons, i.e., peptides with no siblings. The remaining peptides comprise the set of "filtered" peptides (Table 4.2). Note that if a peptide occurs in more than one charge state, these are treated as distinct peptides.

For input to the model, each peptide is encoded in a 1206-dimensional vector. The first six dimensions represent a one-hot encoding of the charge state (from +1 to +6). The remaining 1200 dimensions represent the peptide sequence, where each position is encoded with a 20-dimensional vector corresponding to different amino acids, up to a maximum length of 60.

4.2.4 Train and test set construction

To construct the train/test split, we split the data along two axes. First, we randomly segregate the runs in a ratio of 80%/20%. In this step, if a dataset contains replicate sets of runs, we make sure to keep the replicate runs within the same set. Second, we identify all proteins that contain at least one pair of sibling peptides, and we segregate the proteins into train and test sets in the same ratio.

The training set is comprised of all peptides that occur in training proteins, using measurements drawn from the training set runs. Conversely, the test set contains measurements of peptides within test proteins and measured in test runs. We also defined a validation set with the same process by splitting the training set into training and validation runs with a ratio of 90%/10%. We used this validation set for optimizing our deep learning model, such as hyperparameter tuning and early stopping.

4.2.5 Neural network architecture and training

The Pepper network takes as input the one-hot encoded peptide sequence and charge states and outputs a peptide coefficient. The neural network consists of a 2D convolutional layer containing $20 \times k$ filters, each of which effectively extracts k-mers from the sequence. The output is flattened and concatenated with the charge state, which is then passed to the dense layers. ReLu activaton is used in each layer, and dropout layers are included after every layer. Pepper is trained using the Adam optimizer with an initial learning rate of 0.001 with gradient normalization. Early stopping on the validation set is used with a patience of 100

epochs and a threshold of 0.01 improvement in the loss. The best model, as measured by the loss function (Equation 4.1) on the validation set, is recovered when the training is done.

When calculating the loss function to update the model, we also take into account the protein labels P and the peptide-level measurements Q; however, these are not inputs to the model and thus are not used once the model has been trained. Note that missing values or zeros in the Q matrix are excluded from the loss calculation. The calculated loss values are normalized by dividing them by the total number of peptides and runs, which enables direct comparison between training, validation, and test sets.

Along with the dense and convolutional layers, we also define the α matrix as one of the parameters of our model, which we update while optimizing our loss function. We initialize the α matrix as the median abundance per protein.

For hyperparameter tuning, we used a random search over a grid of hyperparameters. Specifically, we sampled from a grid of filter size (3, 4, 5, 6, 7), number of filters (5, 10, 20, 40), number of layers (1, 2, 3, 4, 5), number of hidden nodes (10, 20, 40, 80), dropout rate (0.0, 0.25, 0.5, 0.75), learning rate (5e-4, 1e-3), and batch size (500, 1000), selecting the hyperparameters that yield the lowest loss (Equation 4.1) on the validation set. The selected hyperparameters were 10 filters each with a size of 3, a total of 4 hidden layers with 40 nodes in each dense layer, trained with a learning rate of 0.001, batch size of 1000, and a dropout rate of 0.25.

We repeated the hyperparameter selection procedure for each dataset separately, extending the grid when necessary, and selected the optimal hyperparameters. We used the same model architecture for all the CPTAC TMT11 datasets. We implemented our model using Keras with a Tensorflow backend.

4.3 Results

4.3.1 Empirical investigation of peptide coefficients

Prior to training a machine learning model to predict peptide coefficients from sequence, we investigated whether we observe consistency of quantities between pairs of sibling peptides across multiple MS runs. For this and most of the remaining experiments, we used SWATH-MS data obtained from the NCI-60 cancer cell lines [161]. The data consists of a total of 14,472 peptides and 120 runs, where two replicate runs were

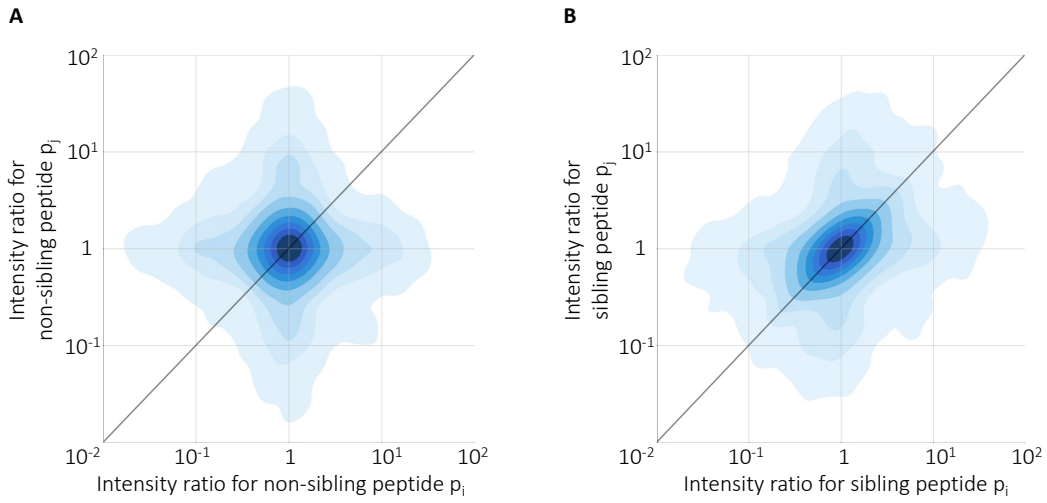

Figure 4.2: **Consistency of sibling peptide ratios across experiments**. The figure plots the log ratio of intensities of pairs of peptides across pairs of MS runs. Each point corresponds to a randomly selected pair of (A) non-sibling or (B) sibling peptides observed in a randomly selected pair of runs. Each panel contains 10,000 randomly selected pairs.

carried out for each of the 60 cancer cell lines. We made a list of all peptides identified in any one of the runs, and we eliminated shared peptides (i.e., peptides that appear in more than one protein) from the list. We then randomly selected a pair of unique peptides (p_i, p_j) and a random pair of runs (r_k, r_ℓ). If both peptides were observed in both runs, then we recorded the corresponding observed quantities $(q_{ik}, q_{i\ell}, q_{jk}, q_{j\ell})$. We assume that each of these quantities can be decomposed into a peptide coefficient and an adjusted abundance; e.g., $q_{ik} = c_i \alpha_{ik}$. Furthermore, we hypothesize that the adjusted abundances for a sibling peptide pair should be approximately equal to one another; i.e., if p_i and p_j are siblings, then $\alpha_{ik} \approx \alpha_{jk}$ and $\alpha_{i\ell} \approx \alpha_{j\ell}$. It follows, therefore, that the ratio of the abundances for the two sibling peptides between the two runs should be approximately equal:

$$\frac{q_{ik}}{q_{i\ell}} = \frac{c_i \alpha_{ik}}{c_i \alpha_{i\ell}} = \frac{\alpha_{ik}}{\alpha_{i\ell}} \approx \frac{\alpha_{jk}}{\alpha_{j\ell}} = \frac{c_j \alpha_{jk}}{c_j \alpha_{j\ell}} = \frac{q_{jk}}{q_{j\ell}}$$

By contrast, we do not expect to observe any correspondence between ratios of pairs of non-sibling peptides.

To test our hypothesis, we repeated this sampling procedure many times and segregated the observed cross-run ratios into sibling and non-sibling pairs. The results of these analysis show evidence for consistency of peptide coefficients across runs (Figure 4.2): the Pearson correlation between intensity ratios

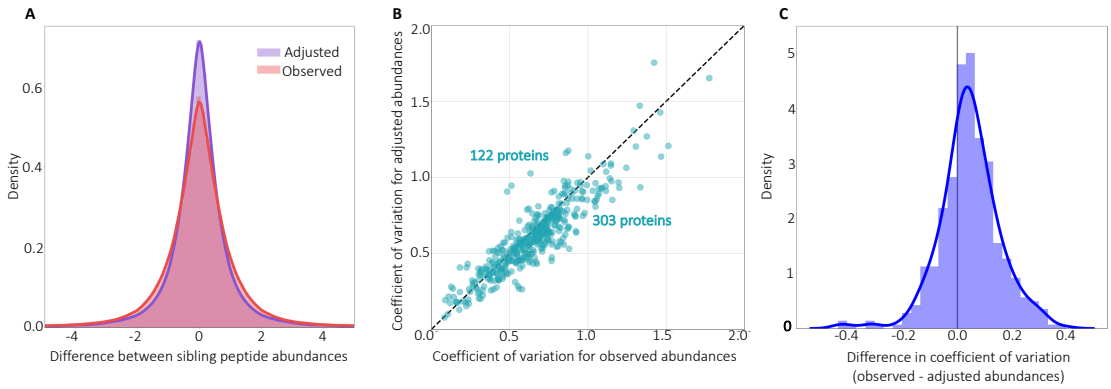

Figure 4.3: **Predicting peptide coefficients across proteins and runs.** (A) The figure plots a histogram of the difference between the observed and adjusted peptide quantities of all sibling peptide pairs, on a logarithmic axis, for the test peptides and runs. (B) The figure plots the coefficient of variation (CV) before and after adjustment, for the test proteins and runs. (C) The figure plots a histogram of the difference between the coefficient of variation for the observed and adjusted peptide quantities for the test proteins and runs in (B).

of peptides are 0.39 and 0.03 for sibling and non-sibling peptides, respectively. This high correlation between ratios of sibling peptides demonstrates that sibling peptide measurements can be used as anchors for quantifying the peptide biases.

4.3.2 The model successfully generalizes to new peptides in new runs

Directly computing ratios of sibling peptide abundances is not a suitable strategy for inferring sequence-induced bias because the empirical ratios potentially reflect additional bias and noise. Accordingly, we turned to our neural network model, which learns to predict the peptide coefficient directly from the peptide sequence and charge state. We hypothesized that, if a Pepper model is truly learning sequence-specific biases, then the model should be able to generalize to new runs and new peptides. Accordingly, we segregated our data into a collection of training and test runs, and we similarly segregated proteins into train and test sets. We then trained a model using only training proteins drawn from the training runs, and we used the trained model to adjust the quantities associated with test proteins in the test runs.

The results of this experiment show that the model successfully generalizes (Figure 4.3A). In particular, we find that the model reduces the loss—i.e., it succeeds in pushing the sibling peptide abundance differences closer to zero—in the test set. The kurtosis of the distribution of sibling peptide abundance differences

is also reduced from 6.01 to 5.18, highlighting the peak at 0 after adjusting with Pepper coefficients.

We repeated the training of the model 10 times with different random initializations and obtained an average loss reduction (i.e., Equation 4.1) of 28.71% in the test set. We also computed the coefficient of variation (CV) per protein (i.e., the standard deviation divided by the mean across all peptides in the protein), before and after adjusting with Pepper coefficients, and observed that CV decreases for the majority (71.3%) of the test proteins (Figure 4.3B-C), highlighting the ability of the model to reduce the within-protein variance in test set abundances. To further examine the proteins which lead to an increase in the CV, we recorded the CV change in proteins with >5 peptides and observed that an even higher percentage of proteins had a decreased CV (80.8%).

To further visualize this result, we selected the 10 proteins from the test set with the highest number of quantified peptides and plotted the abundance of all peptides occurring on each protein before and after adjusting with the coefficients (Figure 4.4). In each case, we observe that the adjusted peptide abundances fall closer to the mean, indicating that our coefficients can minimize the bias in sibling peptide measurements for an unseen protein. We also report the per-protein change in loss values, alongside the change in CV for each protein. For all but one of the proteins, an improvement in the loss function corresponds to a decrease in CV. The only exception is protein O75533, where the increasing CV corresponds to one of the lowest loss changes. We further examined the results for this protein and found that a peptide with charge +6 is responsible for the high standard deviation; after excluding this peptide, the CV decreased by 25.4%. This observation matches with our expectation that the Pepper predictions are less reliable for high charge peptides, because our training set has only a small number of examples for higher charge states. These results support our two central hypotheses—that measured peptide quantities can be decomposed into a sequence-specific bias term and that the adjusted peptide quantities for sibling peptides should be approximately equal—and suggest that Pepper is able to learn to predict the sequence-induced bias terms. Note that, a priori, we do not expect the model to be capable of reducing the loss to zero, even on the training set, because the observed data presumably contains many biases that are not predictable from the amino acid sequence alone.

We next tested whether this approach generalizes to other instrument types, acquisition strategies, and quantification schemes by training and testing the model using a variety of datasets (Table 4.3). In each case,

Dataset	Instrument	Acquisition	Quantification	Improvement
Slevlek *et al.*	5600 TripleTOF	DIA	SWATH-MS	33.00 ($\pm$4.31)
Thomas *et al.*	5600+ TripleTOF	DIA	SWATH-MS	27.90 ($\pm$4.89)
Guo et al.	5600 TripleTOF	DIA	SWATH-MS	28.71 ($\pm$4.10)
CPTAC S16	LTQ Orbitrap Velos	SIM	Label Free	22.05 ($\pm$2.56)
CPTAC S19	LTQ Orbitrap Velos	DDA	Label Free	9.55 ($\pm$5.05)
CPTAC S37	Q-Exactive Plus	SRM	Label Free	15.83 ($\pm$0.24)
CPTAC S47	Orbitrap Fusion Lumos	DDA	TMT11	20.49 ($\pm$1.68)
CPTAC S48	Orbitrap Fusion Lumos	DDA	TMT11	14.88 ($\pm$1.91)
CPTAC S49	Q-Exactive HF	DDA	TMT11	17.50 ($\pm$4.57)
CPTAC S51	Orbitrap Fusion Lumos	DDA	TMT11	24.84 ($\pm$1.76)
CPTAC S54	Orbitrap Fusion Lumos	DDA	TMT11	15.31 ($\pm$0.65)
CPTAC S58	Orbitrap Fusion Lumos	DDA	TMT11	20.39 ($\pm$0.97)
CPTAC S61	Orbitrap Fusion Lumos	DDA	TMT11	20.30 ($\pm$1.79)

Table 4.3: **Performance on various datasets.** The table lists a variety of datasets, reporting in the final column the mean percentage reduction in loss on the test set, relative to the baseline, along with the standard deviation.

we segregated the runs and proteins into train and tests sets in a ratio of 80% to 20%, and we used a fixed model architecture for training and testing. We observed that our model can successfully predict coefficients for datasets with different characteristics, in each case substantially reducing the test set loss relative to the baseline.

4.3.3 Pepper learns successfully in the presence of mislabeled proteoforms

One potential challenge faced by our model arises from the necessarily incomplete and inaccurate collection of proteoforms in our database. Even if we train Pepper using data that was processed using a reference proteome containing many known isoforms, the database cannot possibly account for the huge number of proteoforms that exist in a complex mixture, including unexpected isoforms, post-translational amino acid modifications, and truncation events. In practice, the incompleteness of our database will most often give rise to false positive labels in our training set, i.e., pairs of peptides that we believe to be siblings but which actually lie on different proteoforms (Figure 4.5A). To the extent that such false positives occur in real data, our model will be harder to train.

To investigate the robustness of our approach to proteoform noise, we artificially injected false positives into our training procedure and examined the behavior of the trained model. Specifically, we created a training set consisting of 14,472 peptides, and then we randomly permuted the protein labels for a fixed

percentage of the training peptides. Effectively, we are introducing falsely labeled proteoforms by modifying the protein labels of the peptides. For a noise level of 0, this corresponds to making no change in the dataset. A noise level of 100 corresponds to randomly shuffling the protein labels, i.e., randomly assigning each peptide to another protein that exists in the database. Similarly, for a noise level of 50, we keep the labels of 50% of the peptides as is and randomly shuffle the labels for the remaining 50%, resulting in half of the peptides being assigned to a protein other than the original protein they were mapped to. This procedure has the effect of creating false positive pairs, mimicking pairs of peptides that occur on a single proteoform in our database but occur on distinct proteoforms in the sample.

This experiment shows that the Pepper model is robust to such noise. We observe a smooth degradation of performance as the percentage of false positive pairs increases, with the improvement on the test set remaining above 10% even up to 70% false positives (Figure 4.5B). This result shows that even if our training set contained label noise, we are able to successfully learn to identify sequence-induced bias.

While we demonstrate the generalizability of our model in the presence of mislabeled sibling peptides, the test set performance is highly dependent on the number of noisy samples in the training set. Thus, we aimed to adapt our model to better handle label noise. Borrowing from a popular machine learning technique, Robust PCA [162], we extended our coefficient predictor to model the corrupted labels as parameters to be inferred. Specifically, we trained the same network using a modified loss function that is more robust to label noise:

$$L(Q, \mathcal{P}, f(\cdot)) = \sum_{k=1}^{\kappa} \sum_{\rho \in \mathcal{P}} \sum_{p_i \in \rho} (q_{ik} - f(p_i)\alpha_{\rho,k} - s_{ik})^2 + \lambda \sum_{i=1}^{n} \sqrt{\sum_{k=1}^{\kappa} {s_{ik}}^2} \tag{4.2}$$

where $s_{i,k}$ is the noise term associated with peptide i and run k. The model is trained subject to the same constraints as the loss function in Equation 4.1. The second term in Equation 4.2 corresponds to the regularizer for the S matrix. The lambda value determines the strength of regularization, which we tuned using a validation set for each different noise ratio value.

We hypothesized that, by properly accounting for the label noise that we know exists in our data, this method will improve our ability to accurately infer peptide coefficients. Accordingly, we trained the robust model and compared the percent improvement across all noise ratios (Figure 4.5B). We observe that the

Peptide feature	$\|r\|$
Hydration number	0.451
Relative preference value at N3 (ends of alpha helices)	0.442
Relative preference value at N2 (ends of alpha helices)	0.438
Percentage of exposed residues	0.437
Side chain oriental preference	0.437
Alpha-helix indices for beta-proteins	0.436
Average accessible surface area	0.436
Polar requirement	0.435
Hydrophilicity scale	0.434
Helix initiation parameter	0.431

Table 4.4: **Top physicochemical peptide features.** The table lists the peptide features yielding the highest correlation with the learned coefficients. All the features are obtained from the AAindex database [163]. For each feature, the table reports the absolute value of the Pearson correlation coefficient ($|r|$).

robust model outperforms the regular model, indicating that the robust model is useful for eliminating label noise. We expect this extension of the model to be especially useful for datasets with high ratios of proteoform noise. On the other hand, this extension to the model significantly increases the number of parameters (by roughly a factor of 5 on average), making the training procedure significantly slower.

4.3.4 The coefficients reflect physicochemical properties of the peptide sequences

Previous machine learning models that aim to characterize peptide-specific biases in the context of identifying, rather than quantifying, peptides from mass spectrometry data have shown that the predictions from the models correlate strongly with several key peptide features, including hydrophobicity and peptide length [151; 152; 153; 154; 155; 157; 156]. Accordingly, we segregated Pepper's coefficient predictions from our model according to these two features. In both cases, we observe a strong trend (Figure 4.6A-B), with coefficients taking smaller values for longer peptides or peptides with extreme values (high or low) of hydrophobicity, in agreement with previous work. These results indicate that the instruments are yielding under-estimates of the quantities of long or highly hydrophobic/hydrophylic peptides. By learning small coefficients for such peptides, our model aims to correct the associated biases.

We further calculated the correlation between the learned coefficients and 494 different physicochemical properties obtained from the AAIndex database [163]. Table 4.4 lists the highly correlated features, including polarity, hydration, and structural features. Polarity and hydrophobicity of a peptide determine its

behavior in the solvent [152]. Structural features are also highly relevant because the structure can affect tryptic digestion [151]. It is promising to see our model capturing these properties of the peptides affecting how they behave in the mass spectrometer and adjusting their abundances to provide more accurate quantification.

While the previous methods relied on amino acid features summarized at the peptide level, they highlighted that amino acid composition is a potentially important feature affecting tryptic digestion [151; 152; 155]. Accordingly, we wanted to investigate the effect of amino acid substitutions on the peptide bias. To do so, we randomly sampled real peptide sequences from the NCI-60 dataset and generated simulated peptide sequences by substituting each amino acid at every position with every other amino acid. We then used our trained model to predict coefficients for all pairs of sequences and calculated the differences between the predicted coefficients.

The resulting clustered heat map (Figure 4.6C) suggests that the learned clusters align with physicochemical features of the amino acids. We observe that the two prominent clusters consist of mostly polar versus non-polar amino acids. Polarity was among the most discriminative features for some previous approaches [151; 153; 155]. The polar cluster particularly consists of charged amino acids (D, E, H, K). The hydrophilicity and the charge of the residues affect fragmentation, ionization, and detection processes and thus are critical for determining the behavior of a peptide in mass spectrometer [151]. Similarly, the non-polar cluster contains a group of small amino acids (S, T, G, P, A) where the size of the side chain is known to be related to the flexibility of the amino acid [152]. The cluster map also highlights that the learned coefficients become larger, in general, when a hydrophobic amino acid is substituted with a polar one, indicating that more polar peptides are favored by the instrument. This finding agrees with studies that detected a negative correlation between hydrophobicity of the peptide and the probability of detection [156].

We also investigated the effect of the substitution position on the peptide coefficient. We grouped the scores by position with respect to the N- and C-terminus and plotted the distribution of the coefficients (Figure 4.6D–E). We find that amino acids at either end of the sequence are strongly related to sequence-induced bias. This might be because these residues play an important role in susceptibility of the peptide to enzymatic cleavage or absorption to solid phase extraction matrices [154]. Recapitulating the features that were shown to be important for determining the detectability of a peptide in the context of quantification

Model	Test set percent improvement
Pepper	28.72 ($\pm$4.10)
Amino acid (1-mer) count linear predictor	13.30 ($\pm$5.46)
2-mer count linear predictor	8.48 ($\pm$3.05)
3-mer count linear predictor	15.03 ($\pm$1.24)

Table 4.5: **Performance for baseline approaches.** The table lists baseline approaches, reporting in the final column the mean percentage reduction in loss on the test set, relative to the baseline, along with the standard deviation.

highlights the biological relevance of the learned coefficients.

4.3.5 Pepper outperforms a simple linear model

The observed correlation between our predicted coefficients and amino acid composition suggests that perhaps a simple linear regressor trained using compositional features might be sufficient to accurately model peptide bias. To test this hypothesis, we trained a linear regression model from amino acid counts vector (i.e., vector of length 20 containing the number of occurences of each amino acid) trained using the same loss function as our neural network (Equation 4.1). We further trained linear regression models trained using 2-mer or 3-mer counts (i.e., vectors of length 400 or 8000 containing the number of occurrences of each k-mer).

The comparison of the models (Table 4.5) shows that Pepper clearly outperforms the alternative approaches, highlighting that the neural network architecture, which allows for nonlinearities and for dependencies between input features, is essential for the accurate prediction of the coefficients.

4.3.6 Factoring out sequence-specific bias improves correlation with gene expression

One of the criteria for evaluating the success of our approach is whether the adjusted abundances can provide more accurate quantification. We hypothesized that improving the accuracy of protein quantification would lead to higher correlation with the mRNA measurements. As has been discussed extensively in the literature, we do not expect a very strong correlation between these two data modalities, due to effects such as post-translation modifications and variations in protein degradation rates [164; 165]. Nonetheless, we reasoned that a small proportion of the discordance between protein and mRNA expression might be explained by sequence-specific biases in the quantitative proteomics data. Accordingly, we used a paired set of RNA-

Seq and mass spectrometry measurements to calculate the correlation between the gene and protein-level abundances for each sample before and after adjustment using Pepper. Specifically, NCI-60 protein-level abundances were available for 60 cancer cell lines, and corresponding gene-level abundances were also available for 59 of these cell lines. We first preprocessed the gene-level abundances to select the proteins that are also available in the mass spectrometry dataset as well, and then we calculated the Pearson correlation per sample, i.e., the correlation between all gene abundances and all protein abundances in one cell line. As expected, we observe that Pepper increases the correlation between protein and mRNA-based measurements (Figure 4.7). Strikingly, the correlation improves in 59 out of 59 runs that we tested on ($p < 1.2 \times 10^{-11}$, signed-rank test) highlighting the ability of the learned coefficients to improve the accuracy of downstream analysis.

As a control, we further generated a random set of coefficients by sampling from the range of the learned coefficients. Adjusting the abundances with these random coefficients and recalculating the correlations with the mRNA measurements resulted in deterioration of correlation, highlighting the significance of our results (Figure 4.7). This analysis suggests that our coefficients can be used to reduce the biases associated with peptide measurements.

4.3.7 The model learns successfully from a few runs

Finally, we investigated the effect of the number of training runs on the model performance, and whether it is possible to reduce peptide bias using a few runs. Accordingly, we downsampled the training runs in the Guo et al. dataset and repeated the model training while recording the percent reduction on a fixed set of test runs. We observe the test set performance increasing as the number of training runs increase, as expected, indicating that training from a higher number of runs can be helpful in increasing the generalizability of the model (Figure 4.8). On the other hand, the Pepper model trained from only two runs can achieve a test set loss reduction of 18.5%, indicating that our coefficient predictor can learn effectively from a small number of runs.

We also investigated Pepper's ability to generalize across different mass spectrometry experiments. Accordingly, we trained our coefficient predictor on a combined set of all five CPTAC TMT11 datasets. We then applied the trained model to the held-out CPTAC TMT11 dataset (CPTAC S51) containing 35 runs.

This analysis showed a marked decrease in the model's performance in the cross-experiment setting. In particular, when generalizing to new runs within the held-out dataset, the model achieved a reduction in loss of 24.84% ($\pm$1.76). In contrast, when generalizing to the held-out experiment, the improvement was only 13.51% ($\pm$1.45). The limited ability of our model to generalize across experiments might be related to experiment-specific biases associated with each dataset, which restricts our ability to transfer the learned coefficients to unseen datasets, as also observed in previous studies [151; 152].

4.4 Discussion

Mass spectrometry experiments enable quantifying peptides in complex mixtures, facilitating the identification of diagnostic markers, selection of targets for vaccine production, and study of protein pathways in disease. However, these experiments have intrinsic limitations that cause some peptides to be detected much more often and more accurately compared to other peptides. In this chapter, we address the peptide bias associated with proteomics experiments hindering our ability to accurately quantify the proteome. Specifically, we focus on peptide sequence-induced biases, the properties of the peptide sequence, such as susceptibility to enzymatic cleavage, efficiency of ionization, and uniformity of fragmentation, affecting how it behaves in the mass spectrometer. We aim to quantify these peptide-specific biases, with the goal of adjusting the observed abundances to reduce bias. We developed a deep learning model, Pepper, that takes the peptide sequence and charge state as input and predicts a peptide coefficient to account for peptide-specific biases. Pepper was trained based on our assumption that the abundances of unique sibling peptides should be equal.

We demonstrated that the predicted peptide coefficients successfully reduce our pre-defined loss function for new peptides and runs, which corresponds to reducing the CV of peptide intensities associated with a given protein. This generalization performance was also replicated on multiple datasets generated with different MS/MS instruments using different acquisition and quantification techniques. We also detected significant correlation between various physicochemical features of peptides and the learned coefficients, highlighting that our model captures features, such as hydrophobicity and secondary structure, which were previously shown to affect how a peptide behaves in a mass spectrometer [151; 154; 155; 157; 156]. We demonstrated that our coefficients significantly improve the correlation between protein and mRNA expression, and we examined Pepper's ability to learn from datasets of varying sizes and to generalize across

MS/MS experiments.

One caveat of our approach is that Pepper learns a single peptide coefficient to be applied across all runs. However, each mass spectrometry run exhibits specific biases. With our current approach, our analysis showed that the model pretrained on a different dataset—even if much larger—and transferred to the target dataset does not outperform a model trained directly on the target dataset itself. Previous studies also highlighted the same drawback, where the ability to predict across datasets was quite poor [151; 152]. Some of these studies even found that the most discriminative features for predicting the detection probability of a peptide changed from experiment to experiment [151; 152; 154]. Although training Pepper separately to learn distinct coefficients per run might seem like a plausible extension, our training procedure requires learning a function to map a peptide sequence to a generalizable coefficient. Hence, it is not possible to train our model using a single run. Alternatively, eliminating the run bias along with the peptide bias might be possible by learning run-specific peptide coefficients, but such a training scheme would require labels other than those produced using sibling peptide relationships. Such labels might, for example, be drawn from metadata about the run, embedded in the mzML header. Improving our model to overcome the dependency between the quantitative biases and the experiment would enable jointly training from hundreds of datasets and making predictions for new experiments. If this approach is successful, our ultimate goal would be to offer our trained model as a general preprocessor for any quantitative mass spectrometry data to improve downstream analysis.

In addition, even within a single MS/MS experiment, Pepper is currently limited to capturing only sequence-related biases. However, many additional biases exist that our approach is not designed to address, such as protease cleavage rates and effects of chromatographic elution. Many properties of the protein sequence may be relevant, since burial or surface exposure of any given peptide depends on the 3D structure of the protein. Thus, one of future directions is to generalize our model to take into account a wider variety of features for each protein.

An important caveat to the analyses reported here is that we have largely restricted ourselves to studies that focus on unmodified, fully tryptic peptides. As with any machine learning system, the parameters of the model are fit to the characteristics of the input data and hence likely capture trends that are specific to such peptides. In the future, we plan to investigate quantitative proteomics studies that employ a wider range of

peptides, including partial cleavage products, cleavages from other enzymes, or that include various types of post-translational modifications.

The Pepper software enables researchers with quantitative peptide measurements to infer coefficients and improve protein expression analysis. Better detection and understanding of bias in mass spectrometry experiments can change how we carry out and interpret experiments, leading to a better understanding of the proteome.

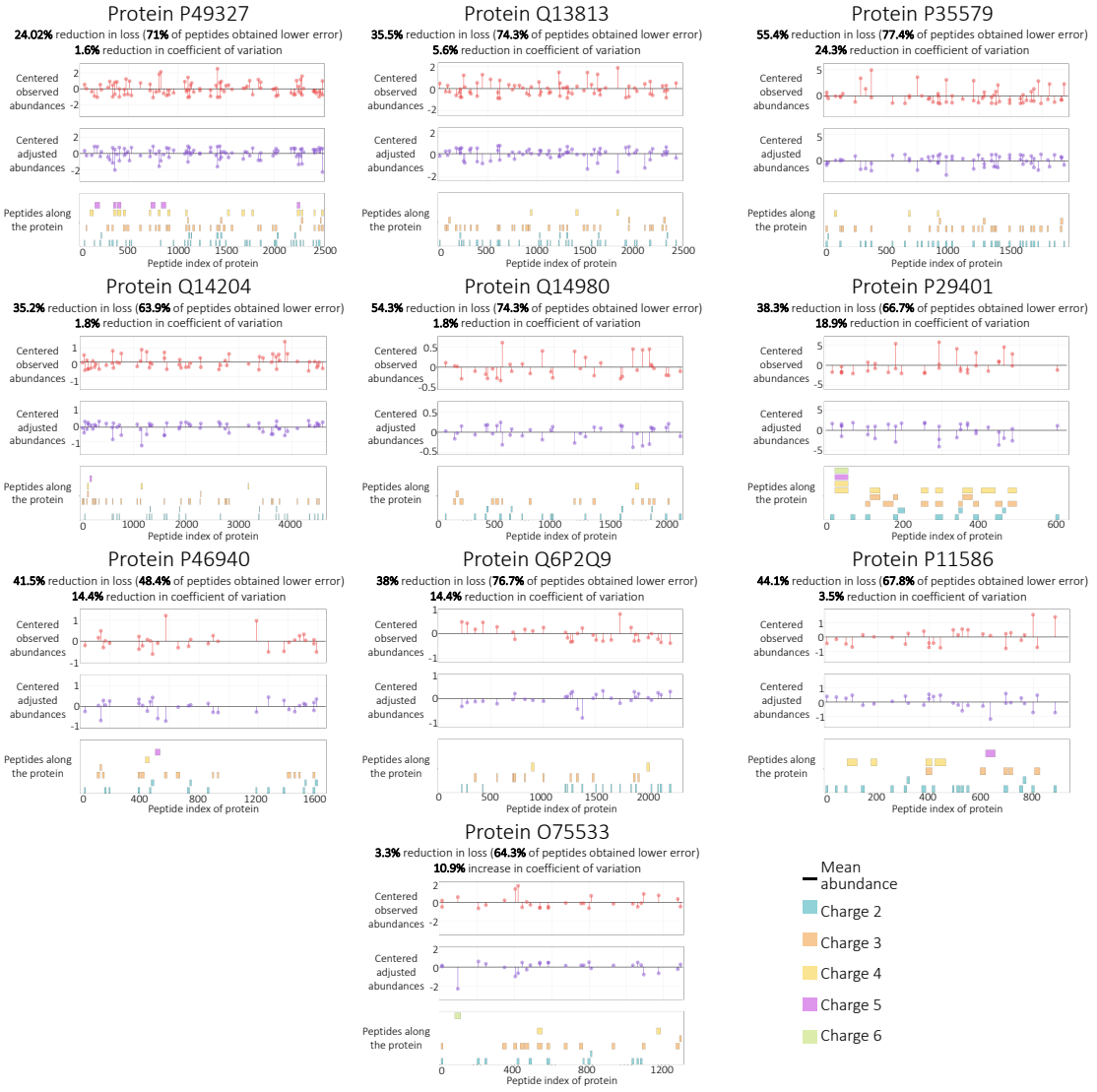

Figure 4.4: **Comparing observed and adjusted abundances.** The figure plots the abundance for all peptides occurring on 10 test set proteins with the highest number of peptides. The horizontal axis is amino acid position along the sequence, and the vertical axis shows the mean-centered peptide abundance for the original (top) and adjusted (bottom) abundances. The percent improvement is calculated with respect to the loss function (Equation 4.1). The percent reduction in coefficient of variation (CV) is also reported for each protein. Individual peptide sequences, segregated by charge state, are arrayed along the bottom of each figure.

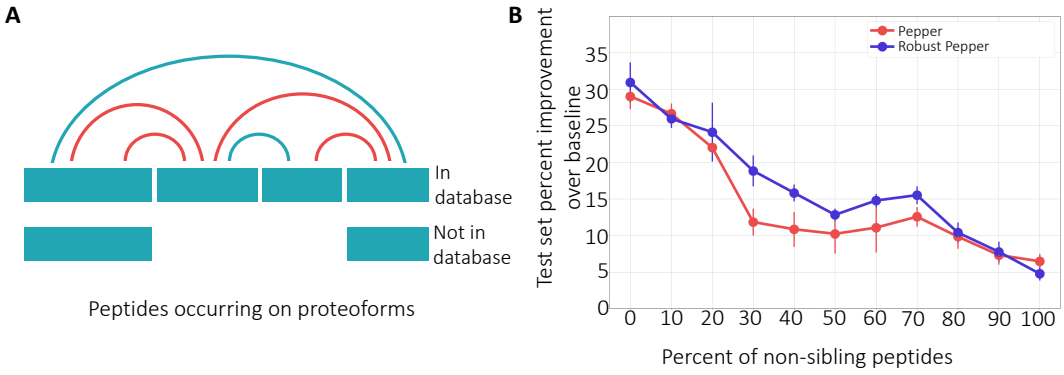

Figure 4.5: **The model's robustness to proteoform noise.** (A) A protein that has two isoforms, only one of which is in the database. The four peptides yield six pairs of apparent siblings. However, because of the presence of the unknown isoforms, some of the sibling relationships (marked in red) are invalid. (B) The figure plots the percent improvement over the baseline (when all coefficients are set to 1), over ten runs, of a fixed set of test peptides as the percentage of label noise increases. Error bars correspond to standard error.

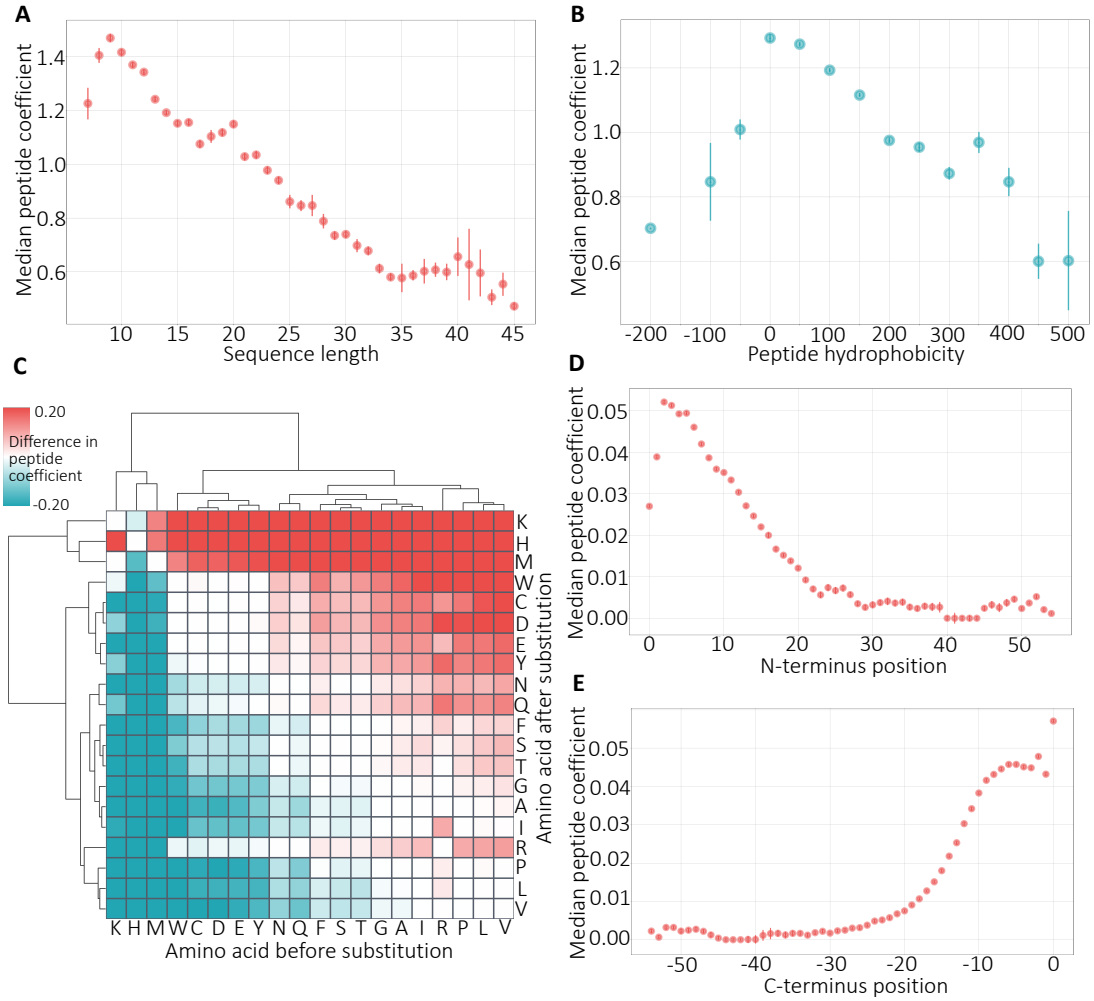

Figure 4.6: **Physicochemical properties of peptides.** (A) The figure plots the relationship between median peptide coefficient (y-axis) and sequence length. Bars represent standard error. (B) Same as panel A but for peptide hydrophobicity. (C) Cluster map of the change in the peptide coefficient per amino acid substitution. The values are the median over all instances of the given substitution in our simulation. (D) The figure plots the distribution of the change in the peptide coefficient plotted separately for each N-terminus position in the peptide sequence. (E) Same as panel D but for C-terminus positions.

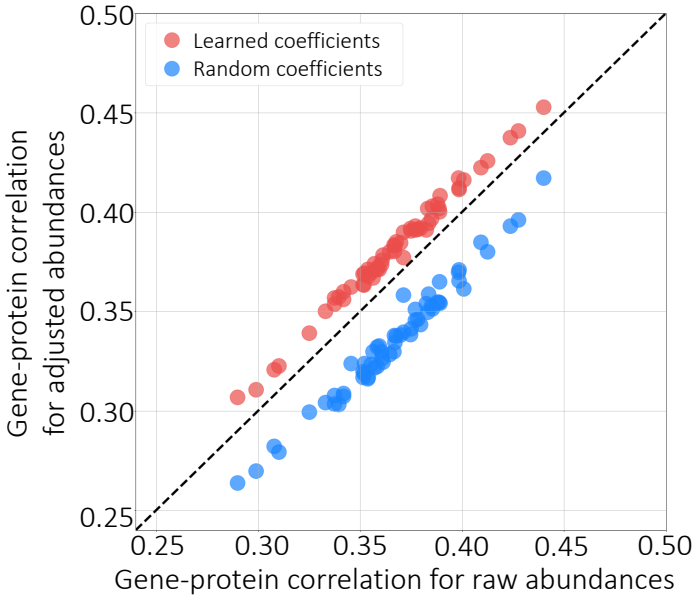

Figure 4.7: **Factoring out sequence-specific biases.** The figure plots the per-protein correlation between gene expression and protein expression, before (x-axis) and after (y-axis) adjusting the quantities using the deep neural network (red points) or using randomly selected coefficients (blue).

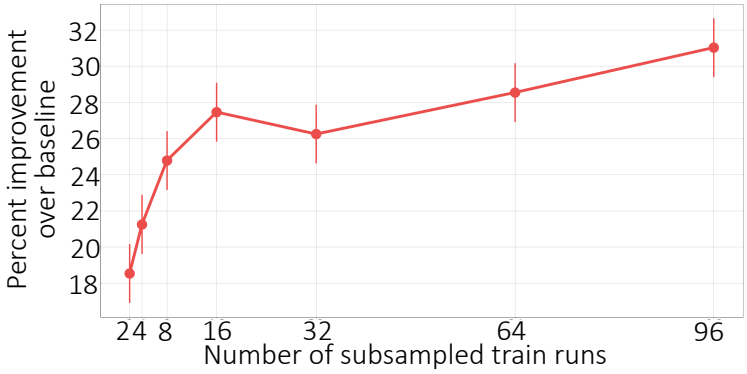

Figure 4.8: **Peptide coefficient predictor learning curve.** The figure plots the percent improvement over the baseline (when all coefficients are set to 1), over ten runs, of a fixed set of test peptides as the number of training runs increase, where 96 runs is the entire training set. Error bars correspond to standard error.

Chapter 5

Conclusion

Deciphering the association between the genotype and phenotype is the key to understanding cellular and molecular mechanisms. The advance of sequencing technologies increased the availability of omics data, providing information from different stages of the central dogma ranging from the DNA sequence all the way to the proteins. This surge in publicly accessible datasets brought forward machine learning as a potentially useful tool for unraveling the complex biology behind these measurements. Unfortunately, limiting the applicability of these models is the complex and heterogeneous nature of the data; using machine learning techniques directly becomes challenging due to experimental artefacts and confounders entangled with signals of interest. We hypothesized that developing computational techniques to eliminate the major problems associated with these measurements can allow us to obtain more informative representations to explore phenotype-genotype relations.

In this thesis, we focus on transcriptomics and proteomics as complementary measurements that facilitate the study of protein and gene expression mechanisms and networks. We develop three different deep learning models to address three major problems associated with transcriptomics and proteomics data: (1) high dimensionality, (2) batch effects and confounders, and (3) experimental noise and biases. These generalizable methods and frameworks allow us to overcome the challenges preventing the application of statistical methods and machine learning techniques to these datasets. Accordingly, we capture the non-linear associations between biological entities, improve downstream analysis tasks, and increase the availability, accuracy, and interpretability of transcriptome or proteome profiles.

In Chapter 2, we focused on the high dimensionality of gene expression profiles and developed DeepProfile approach: an ensemble framework based on unsupervised neural networks to learn robust latent spaces for transcriptomics datasets. We reduced the dimensionality of gene expression profiles and generated a set of interpretable latent nodes, containing key information about cellular processes. By applying this pipeline to 18 human cancer types, we showed that we could effectively predict cancer related phenotypes as well as highlighting mechanisms related to cancer type specificity and survival.

In Chapter 3, we introduced AD-AE, which is an adversarial deconfounding autoencoder that addresses the problem of batch effects and confounders. AD-AE integrates two neural networks: one unsupervised neural network for latent space learning and one adversarial network for preventing the encoding of confounding signals. We demonstrated that the gene expression embeddings learned by AD-AE can better predict patient phenotypes as well as eliminating technical and biological confounders. Our model further allowed transferring downstream prediction models across various datasets.

Chapter 4 focused on proteomics measurements with the goal of reducing peptide-level bias. We proposed a general preprocessor, Pepper, that predicts the associated bias when provided with a peptide sequence. Our approach is a convolutional neural network that incorporates our biological hypothesis about protein quantification into the model loss. The measurements corrected by Pepper reduced the bias as well as highlighting the physicochemical properties of these molecules that lead to sequence-induced bias and improving protein-gene correlation.

By developing these deep learning methods, our aim is to increase the usability and the interpretability of transcriptomics and proteomics measurements. The learned latent representations, denoised embeddings, and de-biased quantifications have led to improved downstream analysis. We also illustrated that the generated representations reveal new insights into biology, enable integration of large amounts of data to decipher disease mechanisms, or tell us more about the experimental procedures that lead to bias.

One of the goals of this thesis is to present methods that are generalizable and that can be applied to any transcriptomics/proteomics dataset. While we explored a few different domains and datasets, it is possible to extend these approaches to many other areas, such as neurodegenerative diseases or microorganism biology. Another key feature of the suggested methods is that by defining general yet customizable methods, we aim for these approaches to be easily adapted for specific tasks. For example, it is possible to use custom

regularizers or incorporate biological priors to DeepProfile, AD-AE, or Pepper and explore new domains.

The methods presented in this thesis focus on solving one fundamental issue at a time. However, it is possible to build upon these approaches to tackle multiple challenges in parallel. For instance, in Chapter 3, we eliminated the confounders observed in transcriptomics measurements while also reducing the dimensionality. Similarly, our models can be integrated to eliminate experimental noise and confounders concurrently or extended for differentiating sources of technical vs. biological variance. Thus, we see building pipelines that target multiple problems as an exciting avenue for future work.

Another interesting direction is investigating the application of these methods to multi-omics datasets. Each chapter of this thesis is dedicated to a single measurement type. On the other hand, combining these datasets, learning latent representations for both gene and protein expression profiles, or generating mappings between the embeddings to detect the differences between the biological structures are among the possible extensions to this thesis. While each different omics measurement captures the biological processes from a different angle, many of the limitations and problems are common to all of them. Thus, multi-omics applications and extensions to larger multi-modality datasets are promising next steps.

In this thesis, we developed deep learning techniques to overcome the limitations of transcriptomics and proteomics measurements preventing us from investigating the underlying biology. By adapting neural networks to omics data, we generated more informative and interpretable versions of our datasets and demonstrated their superior ability to detect cellular and molecular signals. Our models are generalizable to allow researchers to apply these techniques to any transcriptomics/proteomics dataset from different organisms, diseases, or environmental conditions. We believe that our approach will make these datasets more accessible and thus, will open the path to gaining a better understanding of disease biology, developing new treatments, and deciphering the genotype-phenotype relations.

Bibliography

[1] S. Franklin and T. M. Vondriska. Genomes, proteomes, and the central dogma. *Circulation: Cardiovascular Genetics*, 4(5):576–576, 2011.

[2] L. Hood and L. Rowen. The Human Genome Project: big science transforms biology and medicine. *Genome Medicine*, 5(9):79, 2013.

[3] B. E. Slatko, A. F. Gardner, and F. M. Ausubel. Overview of next-generation sequencing technologies. *Current Protocols in Molecular Biology*, 122(1):e59, 2018.

[4] Y. Hasin, M. Seldin, and A. Lusis. Multi-omics approaches to disease. *Genome Biology*, 18(83), 2017.

[5] P. S. Hegde, I. R. White, and C. Debouck. Interplay of transcriptomics and proteomics. *Current Opinion in Biotechnology*, 14(6):647–651, 2003.

[6] A. Moslemi, H. Mahjub, M. Saidijam, J. Poorolajal, and A. R. Soltanian. Bayesian Survival Analysis of High-Dimensional Microarray Data for Mantle Cell Lymphoma Patients. *Asian Pacific Journal of Cancer Prevention*, 17(1):95–100, 2016.

[7] E. Glaab, J.Bacardit, J. M. Garibaldi, and N. Krasnogor. Using Rule-Based Machine Learning for Candidate Disease Gene Prioritization and Sample Classification of Cancer Gene Expression Data. *PLOS ONE*, 7(7):1–18, 2012.

[8] T. Lee and H. Lee. Prediction of Alzheimer's disease using blood gene expression data. *Scientific Reports*, 10(1):3485, 2020.

[9] M. Gönen and A. A. Margolin. Drug susceptibility prediction against a panel of drugs using kernelized Bayesian multitask learning. *Bioinformatics*, 30(17):i556–i563, 2014.

[10] B. Kegerreis, M. D. Catalina, P. Bachali, N. S. Geraci, A. C. Labonte, et al. Machine learning approaches to predict lupus disease activity from gene expression data. *Scientific Reports*, 9(1):9617, 2019.

[11] A. A. Tabl, A. Alkhateeb, W. ElMaraghy, L. Rueda, and A. Ngom. A Machine Learning Approach for Identifying Gene Biomarkers Guiding the Treatment of Breast Cancer. *Frontiers in Genetics*, 10, 2019.

[12] D. G. P. van IJzendoorn, K. Szuhai, I. H. Briaire de Bruijn, M. Kostine, M. Kuijjer, and J. V. M. G. Bovée. Machine learning analysis of gene expression data reveals novel diagnostic and prognostic biomarkers and identifies therapeutic targets for soft tissue sarcomas. *PLOS Computational Biology*, 15(2):1–19, 2019.

[13] T. Jin, N. D. Nguyen, F. Talos, and D. Wang. ECMarker: interpretable machine learning model identifies gene expression biomarkers predicting clinical outcomes and reveals molecular mechanisms of human disease in early stages. *Bioinformatics*, 37(8):1115–1124, 2020.

[14] P. Mamoshina, M. Volosnikova, I. V. Ozerov, E. Putin, E. Skibina, et al. Machine Learning on Human Muscle Transcriptomic Data for Biomarker Discovery and Tissue-Specific Drug Target Identification. *Frontiers in Genetics*, 9, 2018.

[15] S. Perera, K. Hewage, C. Gunarathne, R. Navarathna, D. Herath, and R. G. Ragel. Detection of Novel Biomarker Genes of Alzheimer's Disease Using Gene Expression Data. In *2020 Moratuwa Engineering Research Conference (MERCon)*, pages 1–6, 2020.

[16] L. Niu, M. Thiele, P. E. Geyer, D. N. Rasmussen, H. E. Webel, et al. A paired liver biopsy and plasma proteomics study reveals circulating biomarkers for alcohol-related liver disease. *bioRxiv*, 2020.

[17] L. Higginbotham, L. Ping, E. B. Dammer, D. M. Duong, M. Zhou, et al. Integrated proteomics reveals brain-based cerebrospinal fluid biomarkers in asymptomatic and symptomatic Alzheimer's disease. *Science Advances*, 6(43):eaaz9360, 2020.

[18] W. S. Virreira, O. Karayel, M. T. Strauss, S. Padmanabhan, M. Surface, et al. Urinary proteome profiling for stratifying patients with familial Parkinson's disease. *EMBO Molecular Medicine*, 13(3):e13257, 2021.

[19] J. M. Bader, P. E. Geyer, J. B. Müller, M. T. Strauss, M. Koch, et al. Proteome profiling in cerebrospinal fluid reveals novel biomarkers of Alzheimer's disease. *Molecular Systems Biology*, 16(6):e9356, 2020.

[20] W. Guan, M. Zhou, C. Y. Hampton, B. B. Benigno, L. D. Walker, et al. Ovarian cancer detection from metabolomic liquid chromatography/mass spectrometry data by support vector machines. *BMC Bioinformatics*, 10(1):259, 2009.

[21] K. Bloemen, R. Van Den Heuvel, E. Govarts, J. Hooyberghs, V. Nelen, et al. A new approach to study exhaled proteins as potential biomarkers for asthma. *BMC Bioinformatics*, 41(3):346–356, 2010.

[22] A. B. Dincer, S. Celik, N. Hiranuma, and S. Lee. DeepProfile: Deep learning of cancer molecular profiles for precision medicine. *bioRxiv: 278739*, 2018.

[23] R. Lowe, N. Shirley, M. Bleackley, S. Dolan, and T. Shafee. Transcriptomics Technologies. *PLOS Computational Biology*, 13(5):1–23, 2017.

[24] F. Ozsolak and P. M. Milos. RNA sequencing: advances, challenges and opportunities. *Nature Reviews Genetics*, 12(2):87–98, 2011.

[25] X. Yang, L. Kui, M. Tang, D. Li, K. Wei, et al. High-Throughput Transcriptome Profiling in Drug and Biomarker Discovery. *Frontiers in Genetics*, 11, 2020.

[26] N. Goossens, S. Nakagawa, X. Sun, and Y. Hoshida. Cancer biomarker discovery and validation. *Translational Cancer Research*, 4(3):256–269, 2015.

[27] R. Edgar, M. Domrachev, and A. E. Lash. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210, 2002.

[28] I. Papatheodorou, N. A. Fonseca, M. Keays, Y. A. Tang, E. Barrera, et al. Expression Atlas: gene and

protein expression across multiple studies and organisms. *Nucleic Acids Research*, 46(D1):D246–D251, 2017.

[29] GTEx Consortium. The Genotype-Tissue Expression (GTEx) project. *Nature Genetics*, 45(6):580–585, 2013.

[30] Cancer Genome Atlas Research Network, J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. Shaw, B. A. Ozenberger, et al. The Cancer Genome Atlas Pan-Cancer analysis project. *Nature Genetics*, 45(10):1113–1120, 2013.

[31] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, et al. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*, 2017.

[32] I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville. PixelVAE: A Latent Variable Model for Natural Images. *arXiv preprint arXiv:1611.05013*, 2016.

[33] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, et al. Early Visual Concept Learning with Unsupervised Deep Learning. *arXiv preprint arXiv:1606.05579*, 2016.

[34] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[35] C. Cava, G. Bertoli, A. Colaprico, C. Olsen, G. Bontempi, and I. Castiglioni. Integration of multiple networks and pathways identifies cancer driver genes in pan-cancer analysis. *BMC Genomics*, 19(1):25, 2018.

[36] F. Chen, Y. Zhang, S. Varambally, and C. J. Creighton. Molecular Correlates of Metastasis by Systematic Pan-Cancer Analysis Across The Cancer Genome Atlas. *Molecular Cancer Research*, 17(2):476–487, 2019.

[37] K. A. Hoadley, C. Yau, D. M. Wolf, A. D. Cherniack, D. Tamborero, et al. Multiplatform analysis of 12 cancer types reveals molecular classification within and across tissues of origin. *Cell*, 158(4):929–944, 2014.

[38] Y. Li, K. Kang, J. M. Krahn, N. Croutwater, K. Lee, D. M. Umbach, and L. Li. A comprehensive genomic pan-cancer classification using The Cancer Genome Atlas gene expression data. *BMC Genomics*, 18(1):508, 2017.

[39] S. R. Rosario, M. D. Long, H. C. Affronti, A. M. Rowsam, K. H. Eng, and D. J. Smiraglia. Pan-cancer analysis of transcriptional metabolic dysregulation using The Cancer Genome Atlas. *Nature Communications*, 9(1):5330, 2018.

[40] Q. Wan, H. Dingerdissen, Y. Fan, N. Gulzar, Y. Pan, et al. BioXpress: an integrated RNA-seq-derived gene expression database for pan-cancer analysis. *Database*, 2015:bav019, 2015.

[41] G. P. Way, F. Sanchez-Vega, K. La, J. Armenia, W. K. Chatila, et al. Machine Learning Detects Pan-cancer Ras Pathway Activation in The Cancer Genome Atlas. *Cell Reports*, 23(1):172–180.e3, 2018.

[42] Q. Xu, J. Chen, S. Ni, C. Tan, M. Xu, et al. Pan-cancer transcriptome analysis reveals a gene expression signature for the identification of tumor tissue origin. *Modern Pathology*, 29(6):546–556, 2016.

[43] 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.

[44] G. P. Way and C. S. Greene. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. *Pacific Symposium on Biocomputing*, 23:80–91, 2018.

[45] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[46] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org, 2017.

[47] M. Kanehisa and S. Goto. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1):27–30, 2000.

[48] A. D. Rouillard, G. W. Gundersen, N. F. Fernandez, Z. Wang, C. D. Monteiro, M. G. McDermott, and A. Ma'ayan. The harmonizome: a collection of processed datasets gathered to serve and mine knowledge about genes and proteins. *Database*, 2016.

[49] B. Jassal, L. Matthews, G. Viteri, C. Gong, P. Lorente, et al. The reactome pathway knowledgebase. *Nucleic Acids Research*, 48(D1):D498–D503, 2019.

[50] G. Bindea, B. Mlecnik B, M. Tosolini, A. Kirilovsky, M. Waldner, et al. Spatiotemporal dynamics of intratumoral immune cells reveal the immune landscape in human cancer. *Immunity*, 39(4):782–795, 2013.

[51] M. Oft. IL-10: Master switch from tumor-promoting inflammation to antitumor immunity. *PLOS ONE*, 2(3):194–199, 2014.

[52] M. Jung, R. Sabat, J. Krätzschmar, H. Seidel, K. Wolk, et al. Expression profiling of IL-10-regulated genes in human monocytes and peripheral blood mononuclear cells from psoriatic patients during IL-10 therapy. *European Journal of Immunology*, 34(2):481–493, 2004.

[53] D. Bausch-Fluck, A. Hofmann, T. Bock, A. P. Frei, F. Cerciello, et al. A Mass Spectrometric-Derived Cell Surface Protein Atlas. *PLOS ONE*, 10(4):1–22, 2015.

[54] The UniProt Consortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Research*, 47(D1):D506–D515, 2018.

[55] Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32(suppl_1):D258–D261, 2004.

[56] V. E. Dunlock, A. B. Arp, E. Jansen, S. Charrin, S. J. van Deventer, et al. Dynamic regulation of CD45 by tetraspanin CD53. *bioRxiv: 854323*, 2019.

[57] P. Zjablovskaja, M. Kardosova, P. Danek, P. Angelisova, T. Benoukraf, et al. EVI2B is a C/EBP target gene required for granulocytic differentiation and functionality of hematopoietic progenitors. *Cell Death & Differentiation*, 24(4):705–716, 2017.

[58] L. L. Lanier, B. Corliss, J. Wu, and J. H. Phillips. Association of DAP12 with activating CD94/NKG2C NK cell receptors. *Immunity*, 8(6):693–701, 1998.

[59] P. P. Brons, C. Haanen, J. B. Boezeman, P. Muus, R. S. Holdrinet, et al. Proliferation patterns in acute myeloid leukemia: leukemic clonogenic growth and in vivo cell cycle kinetics. *Annual Hematology*, 66(5):225–33, 1993.

[60] C. Jose, N. Bellance, and R. Rossignol. Choosing between glycolysis and oxidative phosphorylation: a tumor's dilemma? . *Biochimica et biophysica acta*, 1807(6):552–61, 2011.

[61] C. Morral, J. Stanisavljevic, X. Hernando-Momblona, E. Mereu, A. Álvarez Varela, et al. Zonation of Ribosomal DNA Transcription Defines a Stem Cell Hierarchy in Colorectal Cancer. *Cell Stem Cell*, 26(6):845–861.e12, 2020.

[62] J. Roche. The Epithelial-to-Mesenchymal Transition in Cancer. *Cancers*, 10(2):52, 2018.

[63] J. W. Clark, R. P. C. Shiu, F. W. Orr, D. J. Cole, and P. H. Watson. The potential role for prolactin-inducible protein (PIP) as a marker of human breast cancer micrometastasis. *British Journal of Cancer*, 81:1002–1008, 1999.

[64] J. Li, P. S. Choi, C. L. Chaffer, K. Labella, J. H. Hwang, et al. An alternative splicing switch in FLNB promotes the mesenchymal cell state in human breast cancer. *ELife*, 7:e37184, 2018.

[65] S. Bhakta, L. M. Crocker, Y. Chen, M. Hazen, M. M. Schutten, et al. An Anti-GDNF Family Receptor Alpha 1 (GFRA1) Antibody-Drug Conjugate for the Treatment of Hormone Receptor-Positive Breast Cancer. *Molecular Cancer Therapeutics*, 17(3):638–649, 2018.

[66] J. S. Parker, M. Mullins, M. C. U. Cheang, S. Leung, D. Voduc, and P. S. Bernard. Supervised risk predictor of breast cancer based on intrinsic subtypes. *Journal of Clinical Oncology*, 27(8):1160–1167, 2009.

[67] R. A. Alharbi, R. Pettengell, H. S. Pandha, and R. Morgan. The role of HOX genes in normal hematopoiesis and acute leukemia. *Leukemia*, 27(5):1000–1008, 2013.

[68] P. J. M. Valk, R. G. W. Verhaak, M. A. Beijen, C. A. J. Erpelinck, S. B. Van Waalwijk Van Doorn-Khosrovani, et al. Prognostically Useful Gene-Expression Profiles in Acute Myeloid Leukemia. *New England Journal of Medicine*, 350(16):1617–1628, 2004.

[69] R. G. W. Verhaak, B. J. Wouters, A. J. Claudia, S. A. Erpelinck, H. B. Beverloo, et al. Prediction of molecular subtypes in acute myeloid leukemia based on gene expression profiling. *Haematologica*, 94(1):131–134, 2009.

[70] B. Popko, D. K. Pearl, D. M. Walker, T. C. Comas, K. D. Baerwald, et al. Molecular Markers that Identify Human Astrocytomas and Oligodendrogliomas. *Journal of Neuropathology Experimental Neurology*, 61(4):329–338, 2002.

[71] A. Wade, A. E. Robinson, J. R. Engler, C. Petritsch, C. D. James, and J. J. Phillips. Proteoglycans and their roles in brain cancer. *The FEBS Journal*, 280(10):2399–2417, 2013.

[72] S. K. Yoo, Y. Song, E. K. Lee, J. Hwang, H. H. Kim, et al. Integrative analysis of genomic and transcriptomic characteristics associated with progression of aggressive thyroid cancer. *Nature Communications*, 10(1):2764, 2019.

[73] T. R. Walters, F. H. Welland, T. J. Gribble, and H. C. Schwartz. Biosynthesis of heme in leukemic leukocytes. *Cancer*, 20(7):1117–1123, 1967.

[74] Y. Fukuda, Y. Wang, S. Lian, J. Lynch, S. Nagai, et al. Upregulated heme biosynthesis, an exploitable vulnerability in MYCN-driven leukemogenesis. *JCI Insight*, 2(15), 2017.

[75] M. J. Christopher, A. A. Petti, M. P. Rettig, C. A. Miller, E. Chendamarai, et al. Immune escape of relapsed AML cells after allogeneic transplantation. *New England Journal of Medicine*, 379(24):2330–2341, 2018.

[76] F. Ahmad, Q. Sun, D. Patel, and J. M. Stommel. Cholesterol Metabolism: A Potential Therapeutic Target in Glioblastoma. *Cancers (Basel)*, 11(2):146, 2019.

[77] G. R. Villa, J. J. Hulce, C. Zanca, J. Bi, S. Ikegami, et al. An LXR-Cholesterol Axis Creates a Metabolic Co-Dependency for Brain Cancers. *Cancer Cell*, 30(5):683–693, 2016.

[78] A. Jimenez-Pascual and F. A. Siebzehnrubl. Fibroblast Growth Factor Receptor Functions in Glioblastoma. *Cells*, 8(7):715, 2019.

[79] M. Cai, X. Sun, W. Wang, Z. Lian, P. Wu, et al. Disruption of peroxisome function leads to metabolic stress, mTOR inhibition, and lethality in liver cancer cells. *Cancer Letters*, 421:82–93, 2018.

[80] G. Germano, S. Lamba, G. Rospo, L. Barault, A. Magrì, et al. Inactivation of DNA repair triggers neoantigen generation and impairs tumour growth. *Nature*, 552(7683):116–120, 2017.

[81] T. A. Chan, M. Yarchoan, E. Jaffee, C. Swanton, and S. A. Quezadaand others. Development of tumor mutation burden as an immunotherapy biomarker: utility for the oncology clinic. *Annals of Oncology*, 30(1):44–56, 2019.

[82] M. Russo, G. Crisafulli, A. Sogari, N. M. Reilly, S. Arena, et al. Adaptive mutability of colorectal cancers in response to targeted therapies. *Science*, 366(6472):1473–1480, 2019.

[83] C. Engblom, C. Pfirschke, and M. J. Pittet. The role of myeloid cells in cancer therapies. *Nature Reviews Cancer*, 16(7):447–462, 2016.

[84] B. Qian and J. W. Pollard. Macrophage Diversity Enhances Tumor Progression and Metastasis. *Cell*, 141(1):39–51, 2010.

[85] B. Qian and J. W. Pollard. Microenvironmental regulation of tumor progression and metastasis. *Nature Medicine*, 19(11):1423–1437, 2013.

[86] T. Davoli, H. Uno, E. C. Wooten, and S. J. Elledge. Tumor aneuploidy correlates with markers of immune evasion and with reduced response to immunotherapy. *Science*, 355(6322):eaaf8399, 2017.

[87] F. O. Martinez, S. Gordon, M. Locati, and A. Mantovani. Transcriptional Profiling of the Human Monocyte-to-Macrophage Differentiation and Polarization: New Molecules and Patterns of Gene Expression. *The Journal of Immunology*, 177(10):7303–7311, 2006.

[88] W. E. Johnson, C. Li, and A. Rabinovic. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127, 2006.

[89] G. E. Hinton and R. R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[90] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

[91] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[92] L. Van Der Maaten, E. Postma, and J. Van den Herik. Dimensionality reduction: a comparative review. *Journal of Machine Learning Research*, 10:66–71, 2009.

[93] G. Hamerly and C. Elkan. Learning the k in k-means. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.

[94] A. Liberzon, A. Subramanian, R. Pinchback, H. Thorvaldsdóttir, P. Tamayo, and J. P. Mesirov. Molecular signatures database (MSigDB) 3.0. *Bioinformatics*, 27(12):1739–1740, 2011.

[95] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, et al. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

[96] Michel Raymond and Francois Rousset. An exact test for population differentiation. *Evolution*, 49(6):1280–1283, 1995.

[97] 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.

[98] P. Comon. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314, 1994. Higher Order Statistics.

[99] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, 2008.

[100] D. Szklarczyk, A. L. Gable, D. Lyon, A. Junge, S. Wyder, et al. STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Research*, 47(D1):D607–D613, 2019.

[101] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.

[102] D. R. Cox. Regression Models and Life-Tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[103] D. Merico, R. Isserlin, O. Stueker, A. Emili, and G. D. Bader. Enrichment Map: A Network-Based Method for Gene-Set Enrichment Visualization and Interpretation. *PLOS ONE*, 5(11):1–12, 2010.

[104] E. L. Kaplan and Paul Meier. Nonparametric Estimation from Incomplete Observations. *Journal of the American Statistical Association*, 53(282):457–481, 1958.

[105] A. B. Dincer, J. D. Janizek, and S. Lee. Adversarial deconfounding autoencoder for learning robust gene expression embeddings. *Bioinformatics*, 36(Supplement_2):i573–i582, 2020.

[106] J. Tan, J. H. Hammond, D. A. Hogan, and C. S. Greene. ADAGE-Based Integration of Publicly Available Pseudomonas aeruginosa Gene Expression Data with Denoising Autoencoders Illuminates Microbe-Host Interactions. *mSystems*, 1(1), 2016.

[107] J. Du, P. Jia, Y. Dai, C. Tao, Z. Zhao, and D. Zhi. Gene2vec: Distributed representation of genes based on co-expression. *BMC Genomics*, 20(Suppl 1):82, 2019.

[108] B. Haibe-Kains, N. El-Hachem, N. J. Birkbak, A. C. Jin, A. H. Beck, H. J. W. L. Aerts, and J. Quackenbush. Inconsistency in large pharmacogenomic studies. *Nature*, 504(7480):389–393, 2013.

[109] B. Győrffy, A. Lanczky, A. C. Eklund, C. Denkert, J. Budczies, Q. Li, and Z. Szallasi. An online survival analysis tool to rapidly assess the effect of 22,277 genes on breast cancer prognosis using microarray data of 1,809 patients. *Breast Cancer Research and Treatment*, 123(3):725–731, 2010.

[110] C. Lazar, S. Meganck, J. Taminau, D. Steenhoff, A. Coletta, et al. Batch effect removal methods for microarray gene expression data integration: a survey. *Briefings in Bioinformatics*, 14(4):469–490, 2012.

[111] A. H. Sims, G. J. Smethurst, Y. Hey, M. J. Okoniewski, S. D. Pepper, et al. The removal of multiplicative, systematic bias allows integration of breast cancer gene expression datasets–improving meta-analysis and prediction of prognosis. *BMC Medical Genomics*, 1(42), 2008.

[112] C. Li and W. H. Wong. Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. *Proceedings of the National Academy of Sciences*, 98(1):31–36, 2001.

[113] J. Luo, M. Schumacher, A. Scherer, D. Sanoudou, D. Megherbi, et al. A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data. *The Pharmacogenomics Journal*, 10(4):278–291, 2010.

[114] M. Benito, J. Parker, Q. Du, J. Wu, D. Xiang, C. M. Perou, and J. S. Marron. Adjustment of systematic microarray data biases. *Bioinformatics*, 20(1):105–114, 2004.

[115] J. T. Leek and J. D. Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3(9):e161, 2007.

[116] A. E. Teschendorff, J. Zhuang, and M. Widschwendter. Independent surrogate variable analysis to deconvolve confounding factors in large-scale microarray profiling studies. *Bioinformatics*, 27(11):1496–1505, 2011.

[117] H. S. Parker, J. T. Leek, A. V. Favorov, M. Considine, X. Xia, et al. Preserving biological heterogeneity with a permuted surrogate variable analysis for genomics batch correction. *Bioinformatics*, 30(19):2757–2763, 2014.

[118] K. M. Borgwardt, A. Gretton, M. J. Rasch, H. P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by Kernel Maximum Mean Discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

[119] U. Shaham, K. P. Stanton, J. Zhao, H. Li, K. Raddassi, R. Montgomery, and Y. Kluger. Removal of batch effects using distribution-matching residual networks. *Bioinformatics*, 33(16):2539–2546, 2017.

[120] A. Matthew, D. van Dijk, K. Srinivasan, W. S. Chen, H. Mohsen, et al. Exploring single-cell data with deep multitasking neural networks. *Nature Methods*, 16(11):1139–1145, 2019.

[121] U. Shaham. Batch Effect Removal via Batch-Free Encoding. *bioRxiv: 380816*, 2018.

[122] U. Upadhyay and A. Jain. Removal of Batch Effects using Generative Adversarial Networks. *arXiv preprint arXiv:1901.06654*, 2019.

[123] M. Amodio and S. Krishnaswamy. MAGAN: Aligning biological manifolds. *arXiv preprint arXiv:1803.00385*, 2018.

[124] J. B. Dayton. Adversarial Deep Neural Networks Effectively Remove Nonlinear Batch Effects from Gene- Expression Data. Master's thesis, Brigham Young University, 2019.

[125] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on Machine Learning*, pages 325–333, 2013.

[126] L. Christos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

[127] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, et al. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(59):1–35, 2016.

[128] G. Louppe, M. Kagan, and K. Cranmer. Learning to Pivot with Adversarial Networks. In *Proceedings of Advances in Neural Information Processing Systems 30*, pages 981–990. 2017.

[129] J. D. Janizek, G. Erion, A. J. DeGrave, and S. Lee. An Adversarial Approach for the Robust Classification of Pneumonia from Chest Radiographs. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, page 69–79, 2020.

[130] W. A. Knight, R. B. Livingston, E. J. Gregory, and W. L. McGuire. Estrogen Receptor as an Independent Prognostic Factor for Early Recurrence in Breast Cancer. *Cancer Research*, 37(12):4669–4671, 1977.

[131] E. A. Rakha, J. S. Reis-Filho, F. Baehner, D. J. Dabbs, T. Decker, et al. Breast cancer prognostic classification in the molecular era: the role of histological grade. *Breast Cancer Research*, 12(4):207, 2010.

[132] Cancer Genome Atlas (TCGA) Research Network. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216):1061–1068, 2008.

[133] C. W. Brennan, R. G. W. Verhaak, A. McKenna, B. Campos, H. Noushmehr, et al. The somatic genomic landscape of glioblastoma. *Cell*, 155(2):462–477, 2013.

[134] Cancer Genome Atlas (TCGA) Research Network. Comprehensive, Integrative Genomic Analysis of Diffuse Lower-Grade Gliomas. *New England Journal of Medicine*, 372(26):2481–2498, 2015.

[135] D. Arthur and S. Vassilvitskii. k-means++: The Advantages of Careful Seeding. Technical Report 2006-13, Stanford InfoLab, 2006.

[136] G. P. Way, M. Zietz, V. Rubinetti, D. S. Himmelstein, and C. S. Greene. Compressing gene expression data using multiple latent space dimensionalities learns complementary biological representations. *Genome Biology*, 21(1):109, 2020.

[137] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR*, 2015.

[138] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[139] T. M. H. Hsu, W. Y. Chen, C. A. Hou, Y. H. H. Tsai, Y. R. Yeh, and Y. C. F. Wang. Unsupervised Domain Adaptation With Imbalanced Cross-Domain Data. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 4121–4129, 2015.

[140] G. G. Erion, J. D. Janizek, P. Sturmfels, S. Lundberg, and S. Lee. Learning Explainable Models Using Attribution Priors. *arXiv preprint arXiv:1906.10670*, 2019.

[141] P. Sturmfels, S. Lundberg, and S. Lee. Visualizing the Impact of Feature Attribution Baselines. *Distill:10.23915/distill.00022*, 2020. https://distill.pub/2020/attribution-baselines.

[142] A. B. Dincer, Y. Lu, D. Schweppe, S. Oh, and W. S. Noble. Reducing peptide sequence bias in quantitative mass spectrometry data with machine learning. *bioRxiv: 2022.04.11.487945*, 2022.

[143] B. Alberts, A. Johnson, and J. Lewis. *Molecular Biology of the Cell*. Garland Science, New York, NY, 2002.

[144] R. Aebersold and M. Mann. Mass-spectrometric exploration of proteome structure and function. *Nature*, 537:347–355, 2016.

[145] R. Aebersold and M. Mann. Mass spectrometry-based proteomics. *Nature*, 422:198–207, 2003.

[146] H. Steen and M. Mann. The ABC's (and XYZ's) of peptide sequencing. *Nature Reviews Molecular Cell Biology*, 5:699–711, 2004.

[147] N. Pappireddi, L. Martin, and M. Wühr. A Review on Quantitative Multiplexed Proteomics. *ChemBioChem*, 20(10):1210–1224, 2019.

[148] F. Chen, D. S. Chandrashekar, S. Varambally, and C. J. Creighton. Pan-cancer molecular subtypes revealed by mass-spectrometry-based proteomic characterization of more than 500 human cancers. *Nature Communications*, 10(1):5679, 2019.

[149] D. P. Nusinow, J. Szpyt, M. Ghandi, C. M. Rose, E. R. McDonald, et al. Quantitative Proteomics of the Cancer Cell Line Encyclopedia. *Cell*, 180(2):387–402.e16, 2020.

[150] B. Aslam, M. Basit, M. A. Nisar, K. Mohsin, and M. H. Rasool. Proteomics: Technologies and Their Applications. *Journal of Chromatographic Science*, 55(2):182–196, 2017.

[151] P. Mallick, M. Schirle, S. S. Chen, M. R. Flory, H. Lee, et al. Computational prediction of proteotypic peptides for quantitative proteomics. *Nature Biotechnology*, 25:125–131, 2006.

[152] W. S. Sanders, S. M. Bridges, F. M. McCarthy, B. Nanduri, and S. C. Burgess. Prediction of peptides observable by mass spectrometry applied at the experimental set level. *BMC Bioinformatics*, 8(7):S23, 2007.

[153] B. J. Webb-Robertson, W. R. Cannon, C. S. Oehmen, A. R. Shah, V. Gurumoorthi, M. S. Lipton, and K. M. Waters. A support vector machine model for the prediction of proteotypic peptides for accurate mass and time proteomics. *Bioinformatics*, 24(13):1503–9, 2008.

[154] V. A. Fusaro, D. R. Mani, J. P. Mesirov, and S. A. Carr. Prediction of high-responding peptides for targeted protein assays by mass spectrometry. *Nature Biotechnology*, 27(2):190–198, 2009.

[155] C. E. Eyers, C. Lawless, D. C. Wedge, K. W. Lau, S. J. Gaskell, and S. J. Hubbard. CONSeQuence: Prediction of reference peptides for absolute quantitative proteomics using consensus machine learning approaches. *Molecular & Cellular Proteomics*, 10(11):M110.003384, 2011.

[156] B. C. Searle, J. D. Egertson, J. G. Bollinger, A. B. Stergachis, and M. J. MacCoss. Using Data Independent Acquisition (DIA) to Model High-responding Peptides for Targeted Proteomics Experiments. *Molecular & Cellular Proteomics*, 14(9):2331–2340, 2015.

[157] J. Muntel, S. A. Boswell, S. Tang, S. Ahmed, I. Wapinski, et al. Abundance-based classifier for the prediction of mass spectrometric peptide detectability upon enrichment. *Molecular & Cellular Proteomics*, 14(430–440), 2015.

[158] C. M. Shuford, D. L. Comins, J. L. Whitten, J. C. Burnett, and D. C. Muddiman. Improving limits of detection for B-type natriuretic peptide using PC-IDMS: An application of the ALiPHAT strategy. *Analyst*, 135(1):36–41, 2010.

[159] C. M. Shuford and D. C. Muddiman. Capitalizing on the hydrophobic bias of electrospray ionization through chemical modification in mass spectrometry-based proteomics. *Expert Reviews in Proteomics*, 8(3):317–323, 2011.

[160] B. Kuster, M. Schirle, P. Mallick, and R. H. Aebersold. Scoring proteomes with proteotypic peptide probes. *Nature Reviews Molecular Cell Biology*, 6:577–583, 2005.

[161] T. Guo, A. Luna, V. N. Rajapakse, C. C. Koh, Z. Wu, et al. Quantitative Proteome Landscape of the NCI-60 Cancer Cell Lines. *iScience*, 21:664–680, 2019.

[162] H. Xu, C. Caramanis, and S. Sanghavi. Robust PCA via Outlier Pursuit. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems-Volume 2*, pages 2496–2504, 2010.

[163] S. Kawashima and M. Kanehisa. AAindex: Amino Acid index database. *Nucleic Acids Research*, 28(1):374, 2000.

[164] D. Greenbaum, C. Colangelo, K. Williams, and M. Gerstein. Comparing protein abundance and mRNA expression levels on a genomic scale. *Genome Biology*, 4(9):117, 2003.

[165] Y. Liu, A. Beyer, and R. Aebersold. On the Dependency of Cellular Protein Levels on mRNA Abundance. *Cell*, 165(3):535–550, 2016.

Chapter A

Appendix A; Appendix to Deep profiling of a compendium of expression data from 18 human cancers

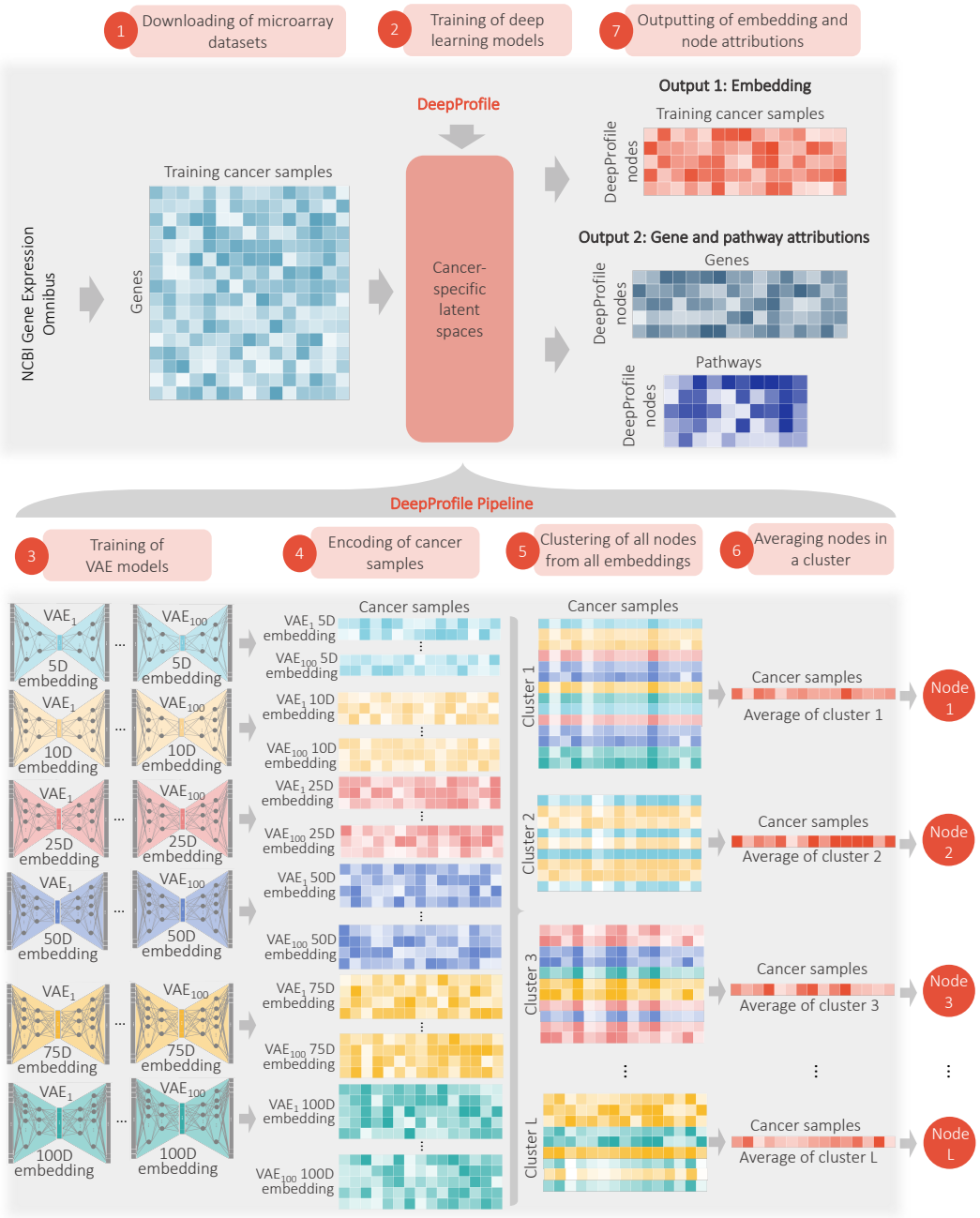

Figure A.1: DeepProfile deep learner pipeline (see caption on next page).

Figure A.1: **(1)** We downloaded all gene expression datasets from the most common microarray platforms available at the time from the NCBI Gene Expression Omnibus (GEO) database. We preprocessed each dataset separately, concatenated all the dataset samples and corrected for batch effects to define a matrix of expression samples containing the expression of all genes for all cancer samples. We pass the expression matrices to DeepProfile pipeline to learn DeepProfile embeddings. Note that the training of the DeepProfile pipeline is carried separately for each cancer type.
(2) DeepProfile is a deep learning model that uses all the downloaded cancer expression profiles to learn a cancer-specific latent space.
(3) Using the training cancer samples, we train hundreds of different Variational Autoencoder (VAE) models. We train each VAE model with a different latent dimension size 100 times with different random weight initializations.
(4) We encode the training cancer samples using each of these VAE models to generate an embedding for each VAE model. Since VAE models have varying number of latent dimension sizes, the generated embeddings also have varying number of latent nodes.
(5) We cluster all nodes of all VAE embeddings to group together nodes that have similar patterns for the training samples.
(6) To define the final node values, we average the node values in a cluster and combine them to define the final DeepProfile embedding and latent nodes. DeepProfile learns L nodes where each node is an ensemble of VAE nodes from different models.
(7) DeepProfile generates two outputs. Output 1 is an embedding of all the cancer samples where the number of DeepProfile nodes is much smaller than the original number of genes passed to the model. Output 2 is the gene-level and pathway-level attributions for each DeepProfile node. Gene-level attributions contain the measure of how much a gene is contributing to each node. Similarly, pathway-level attributions contain p-values from pathway enrichment tests applied to each node-pathway pair.

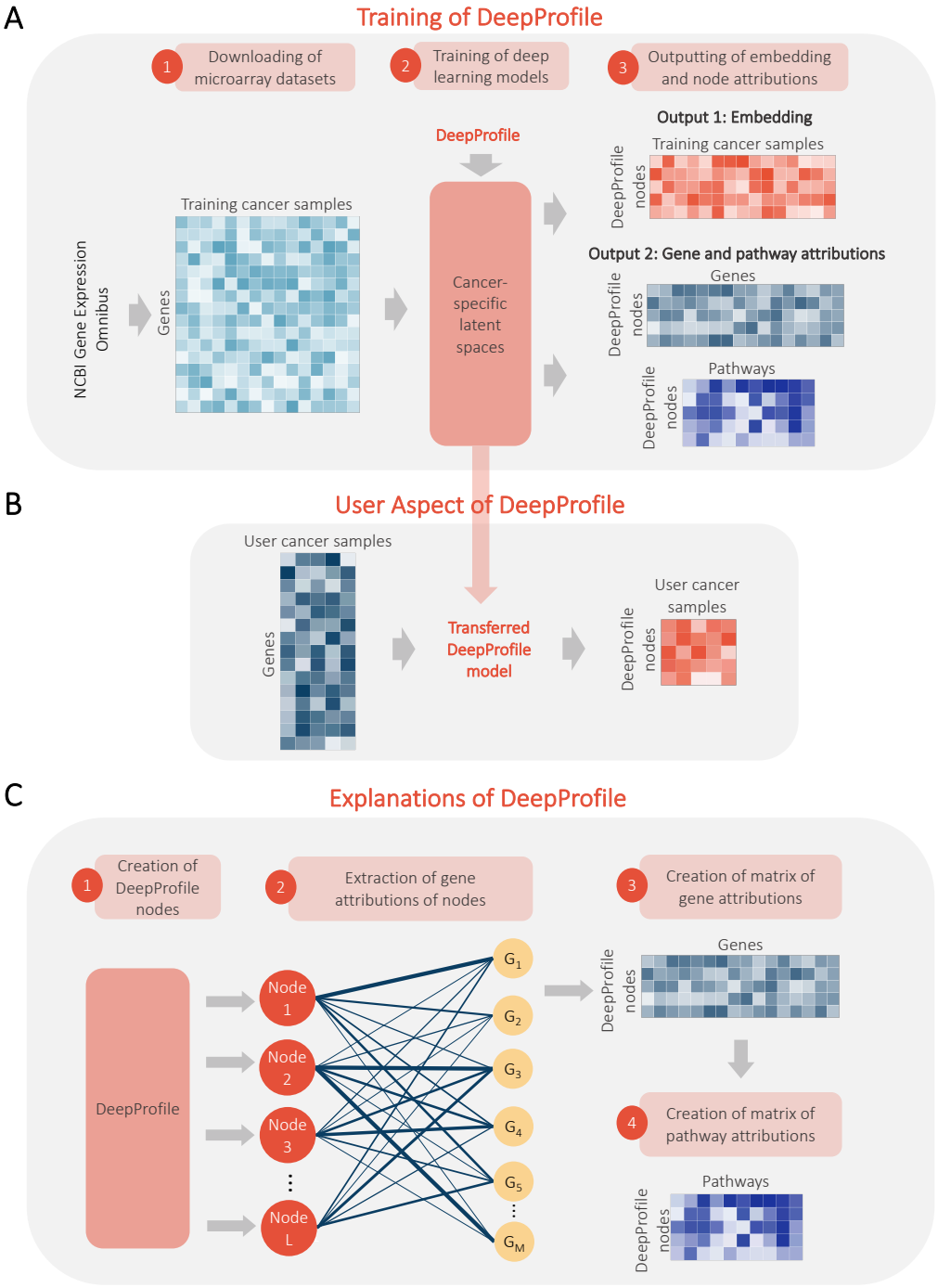

Figure A.2: DeepProfile training, user aspect, and explanations (see caption on next page).

Figure A.2: **(A) The training of DeepProfile.**
(1) DeepProfile first collects publicly available expression dataset for each cancer type from NCBI GEO and defines an expression matrix, containing the expression measurements for all genes and samples.
(2) DeepProfile is a deep learning model that uses all the cancer samples to learn cancer-specific latent spaces.
(3) DeepProfile generates two outputs: embedding and gene-level and pathway-level attributions.
(B) The transferring of DeepProfile model. DeepProfile model trained from thousands of cancer samples can be used to generate embeddings for new expression profiles. When an expression matrix of user cancer samples is passed to DeepProfile, it uses the already trained model and maps the user samples to the learned cancer-specific latent space and outputs an embedding for the user samples.
(C) Explanations of DeepProfile nodes.
(1) DeepProfile pipeline shown in A generates L latent nodes.
(2) For each latent node we obtain gene attributions. Each DeepProfile node is connected to genes through a set of fully or densely connected multilayer perceptron layers. We use Interpreter model to simplify the multilayer connections and define the weights connecting each gene to each DeepProfile node.
(3) The graph of gene-level attributions of DeepProfile nodes in (2) is converted to a gene attribution matrix, containing the contribution of each DeepProfile node to each gene.
(4) From gene-level attributions, using the pathway memberships of genes, pathway-level attributions are obtained, containing the p-value indicating the significance of the overlap between gene sets and the most important genes of DeepProfile nodes.

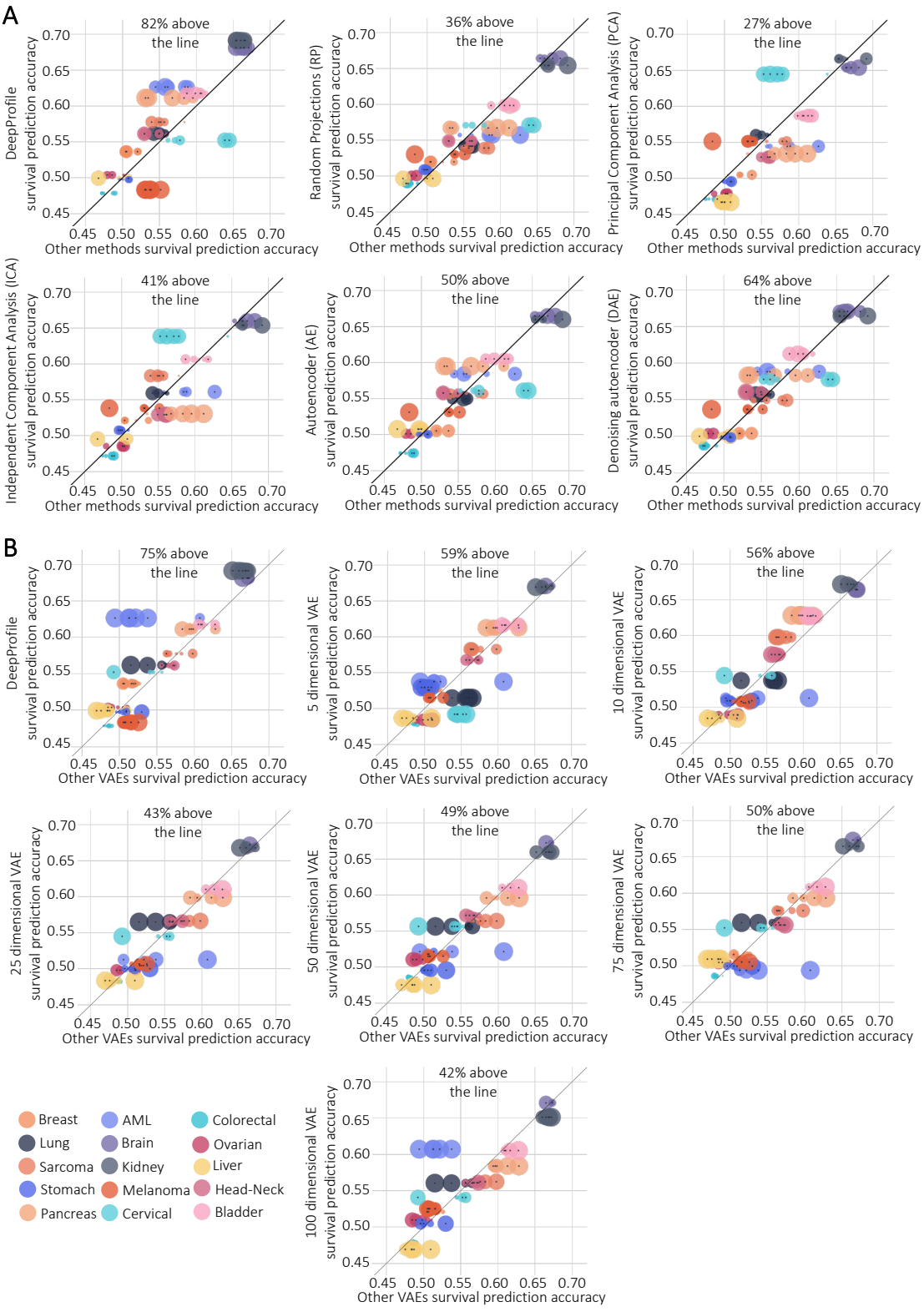

Figure A.3: Survival prediction accuracy comparisons (see caption on next page).

Figure A.3: **(A)** For 5 alternative dimension reduction methods and DeepProfile, we compare 5-year survival prediction accuracies of TCGA RNA-Seq cancer expression embeddings. The y-axis shows the accuracy of the specific method we are investigating, and the x-axis denotes the accuracy of the other methods. Each dot denotes a method and cancer type pair, where the dots are colored by cancer type. The size of the dots is determined by the negative p-value of the significance of outperformance measured by Wilcoxon sign-ranked test, where a larger sized node denotes a lower p-value. All nodes with p-value < 0.05 are marked with a star. The number of cases where the method outperforms the other methods is indicated on the plot as a percentage. A scatter plot is generated for each of the 6 methods.
(B) The same plots created for comparing different dimensional VAE models.

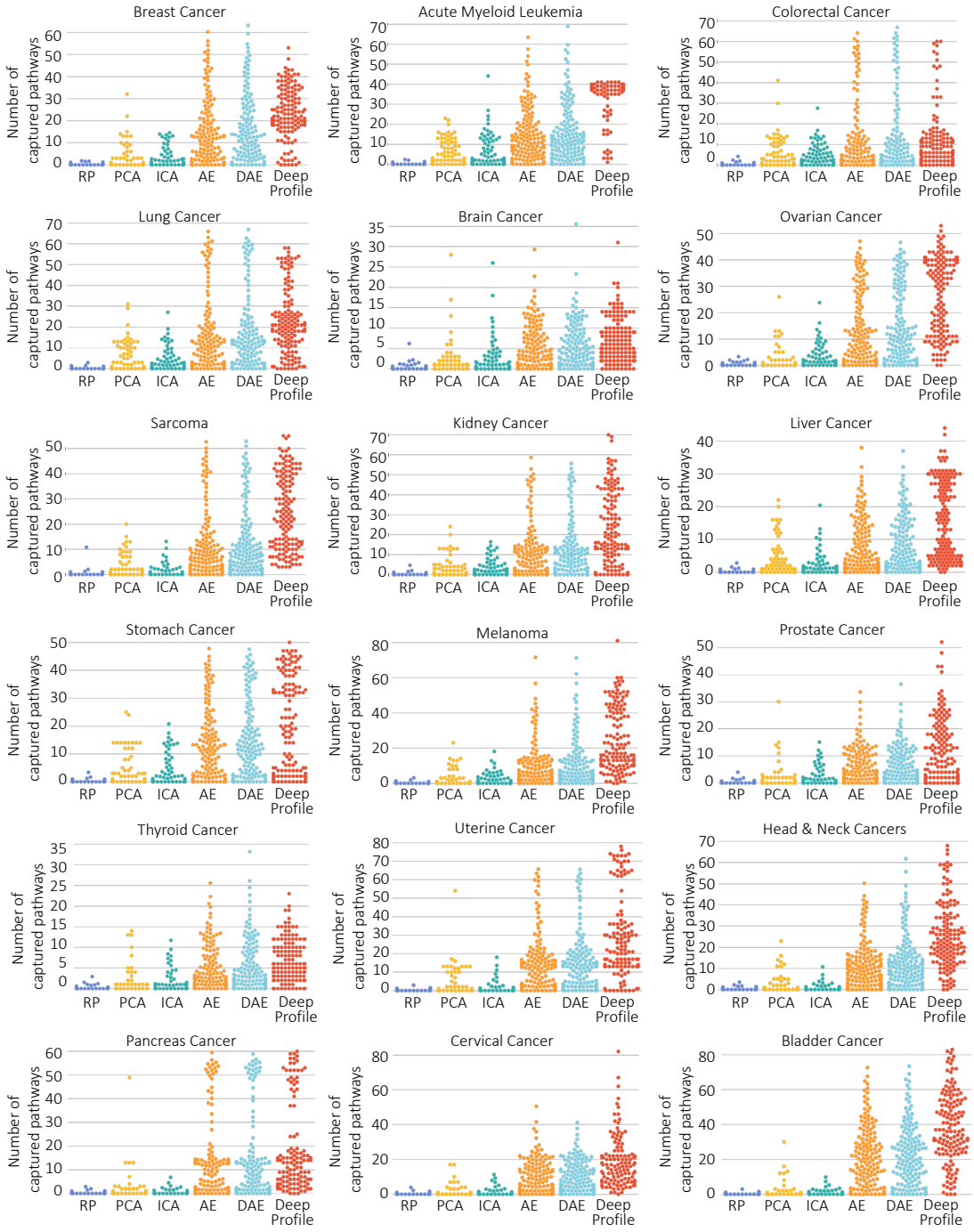

Figure A.4: Distribution plots of number of KEGG, BioCarta, Reactome pathways significantly captured (FDR corrected p-value < 0.05) by each latent node of embeddings generated by DeepProfile and other methods are shown.

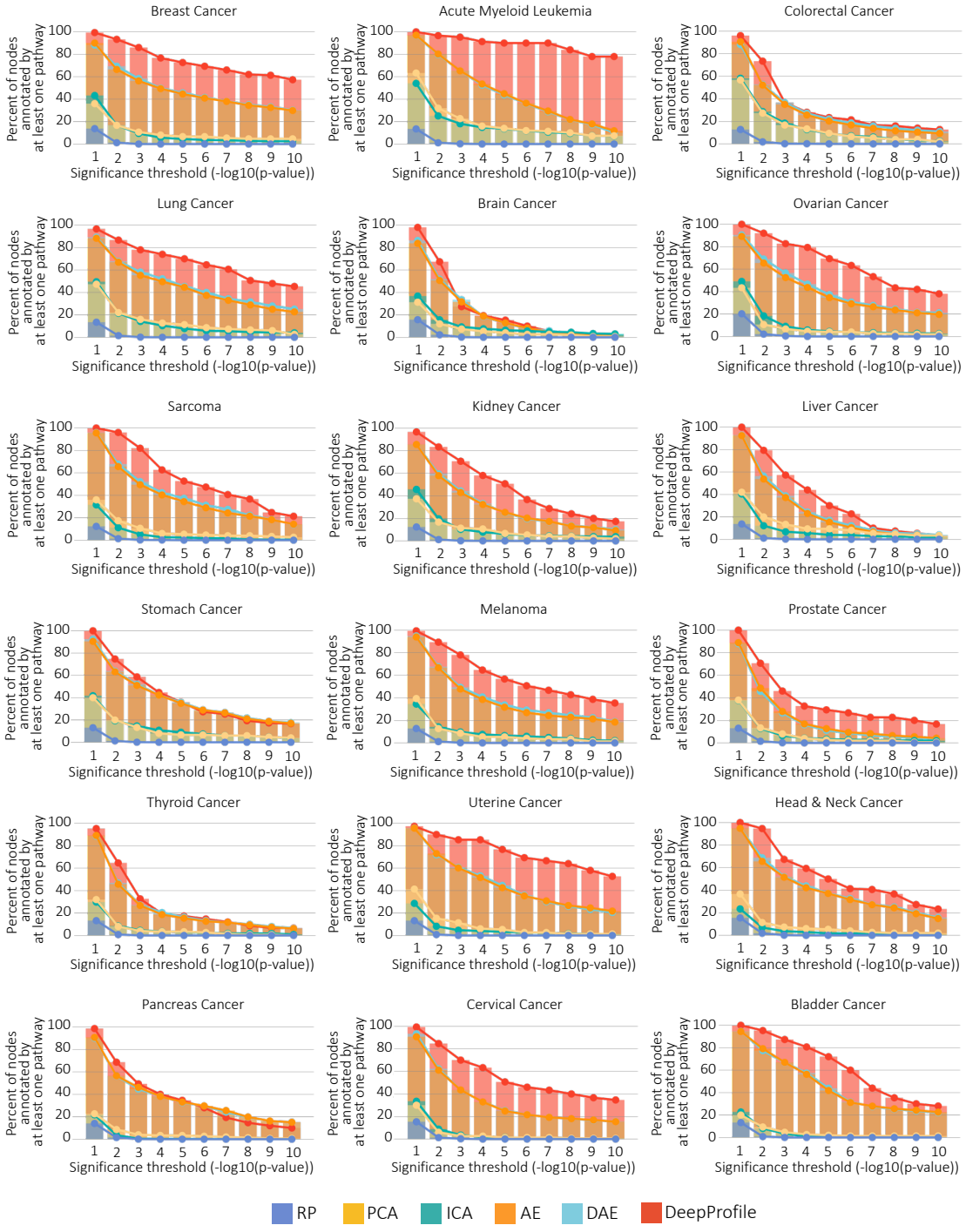

Figure A.5: Comparison of biologically annotated nodes. Comparison of the percent of nodes annotated by at least one pathway above the significance threshold. The percent of annotated nodes are shown for multiple significance thresholds for DeepProfile and alternative dimensionality reduction methods.

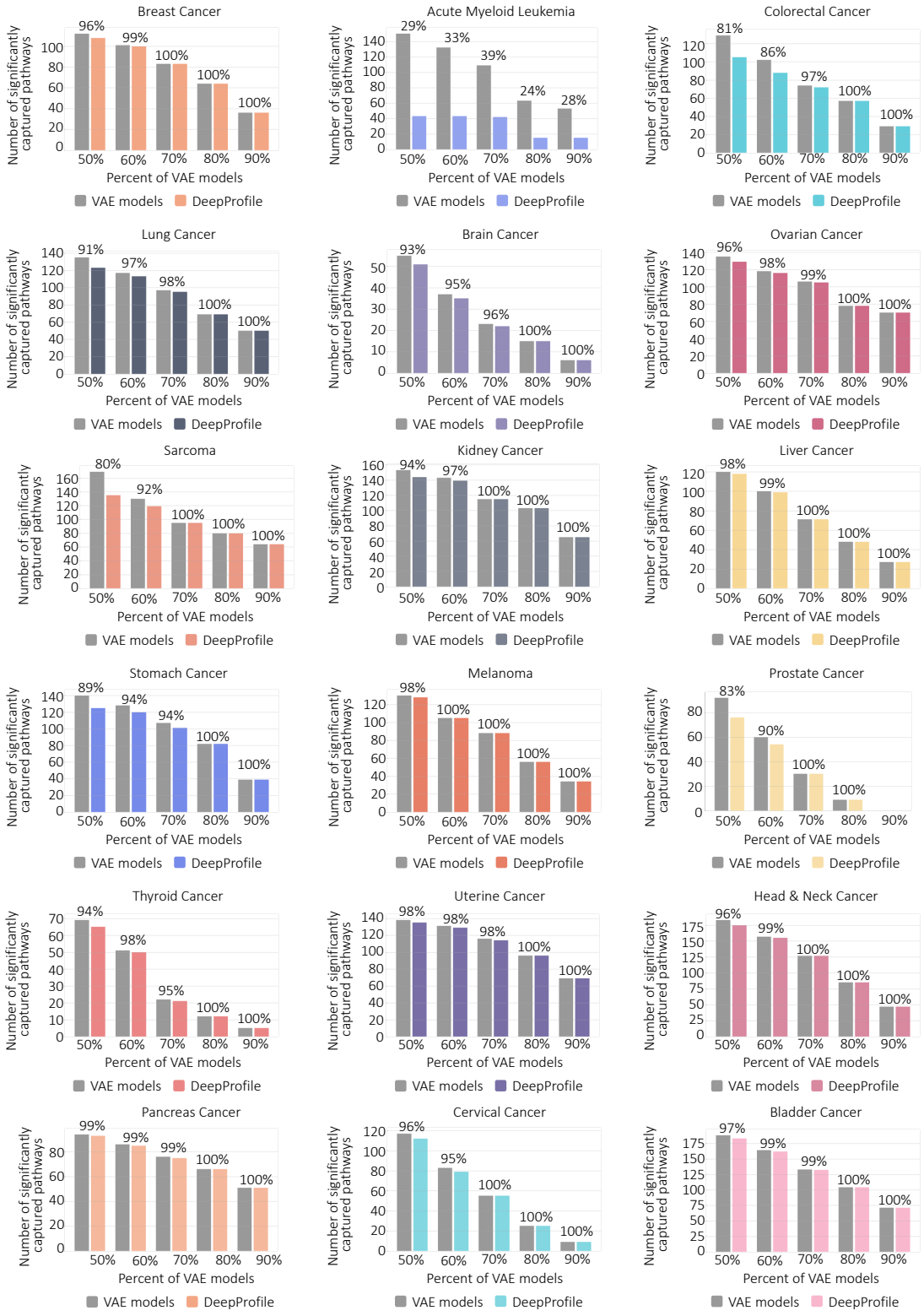

Figure A.6: Comparison of DeepProfile's pathway coverage with individual VAE models (see caption on next page).

Figure A.6: Plots of pathway coverage comparison of DeepProfile model and VAE models. For each pathway, we count the percent of all the VAE models that can significantly capture this pathway (FDR corrected p-value < 0.05). We also check if DeepProfile can significantly capture this pathway as well. For various percentage thresholds shown on the y-axis, we show the number of pathways captured by at least threshold percent of the VAE models (left bar) and the number of pathways captured by DeepProfile as well (right bar). The percentage values shown for each threshold denote the percent of pathways covered by DeepProfile. For 17 cancer types out of 18, DeepProfile can capture at least 80% of the pathways captured by at least 50% of the VAE models and 100% of the pathways captured by at least 80% of the VAE models.

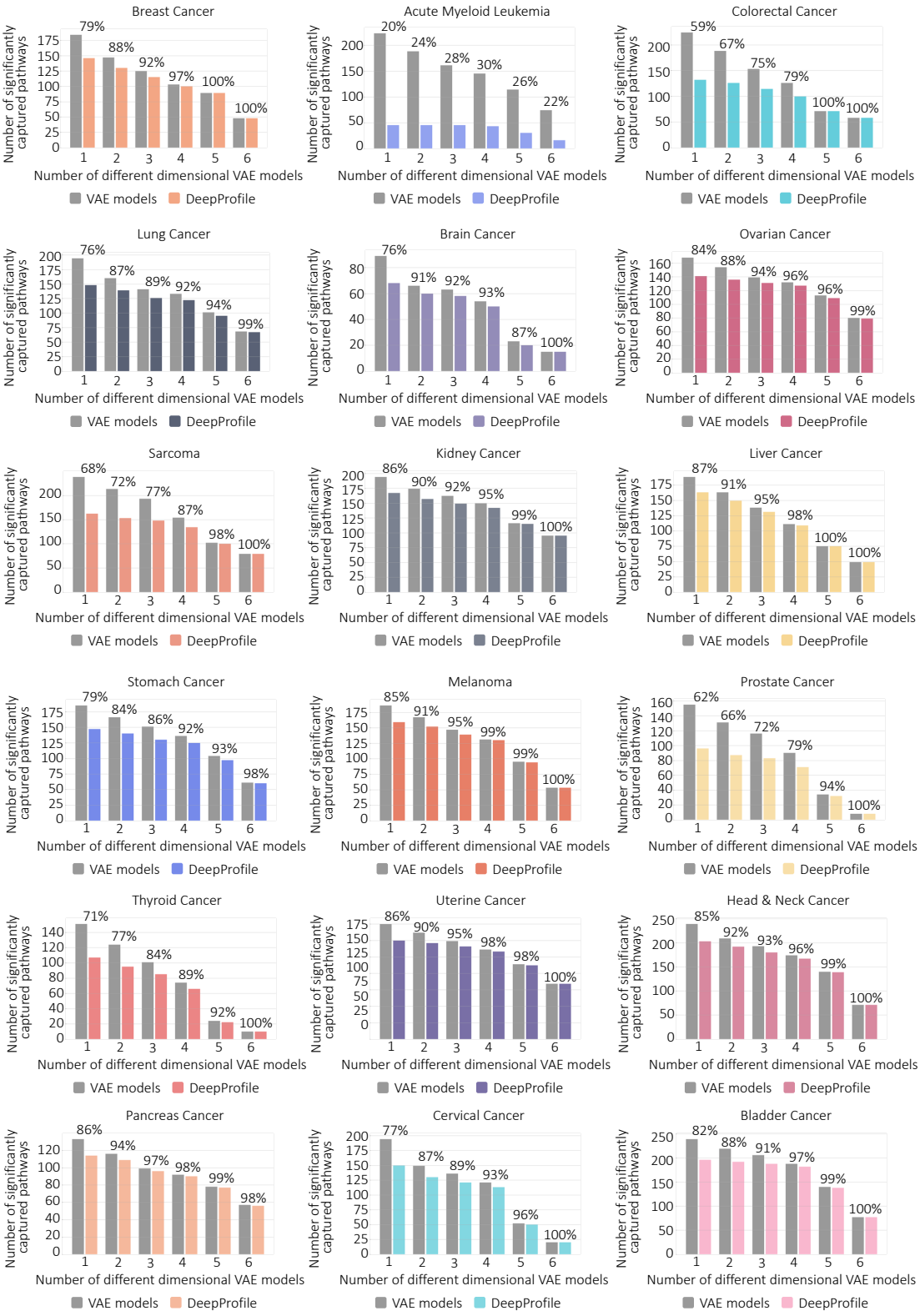

Figure A.7: Comparison of DeepProfile's pathway coverage with different dimensional VAE models (see caption on next page).

Figure A.7: Plots of pathway coverage comparison of DeepProfile model and different dimensional VAE models. For each pathway, we count the number of different dimensional models (out of 6 different dimension sizes) that can significantly capture this pathway (FDR corrected p-value < 0.05). We also check if DeepProfile can significantly capture this pathway as well. For various threshold counts shown on the y-axis, we show the number of pathways captured by at least threshold number of different dimensional VAE models (left bar) and the number of pathways captured by DeepProfile as well (right bar). The percentage values shown for each threshold denote the percent of pathways covered by DeepProfile. For 17 cancer types out of 18, DeepProfile can capture at least 98% of the pathways captured by all different dimensional VAE models.

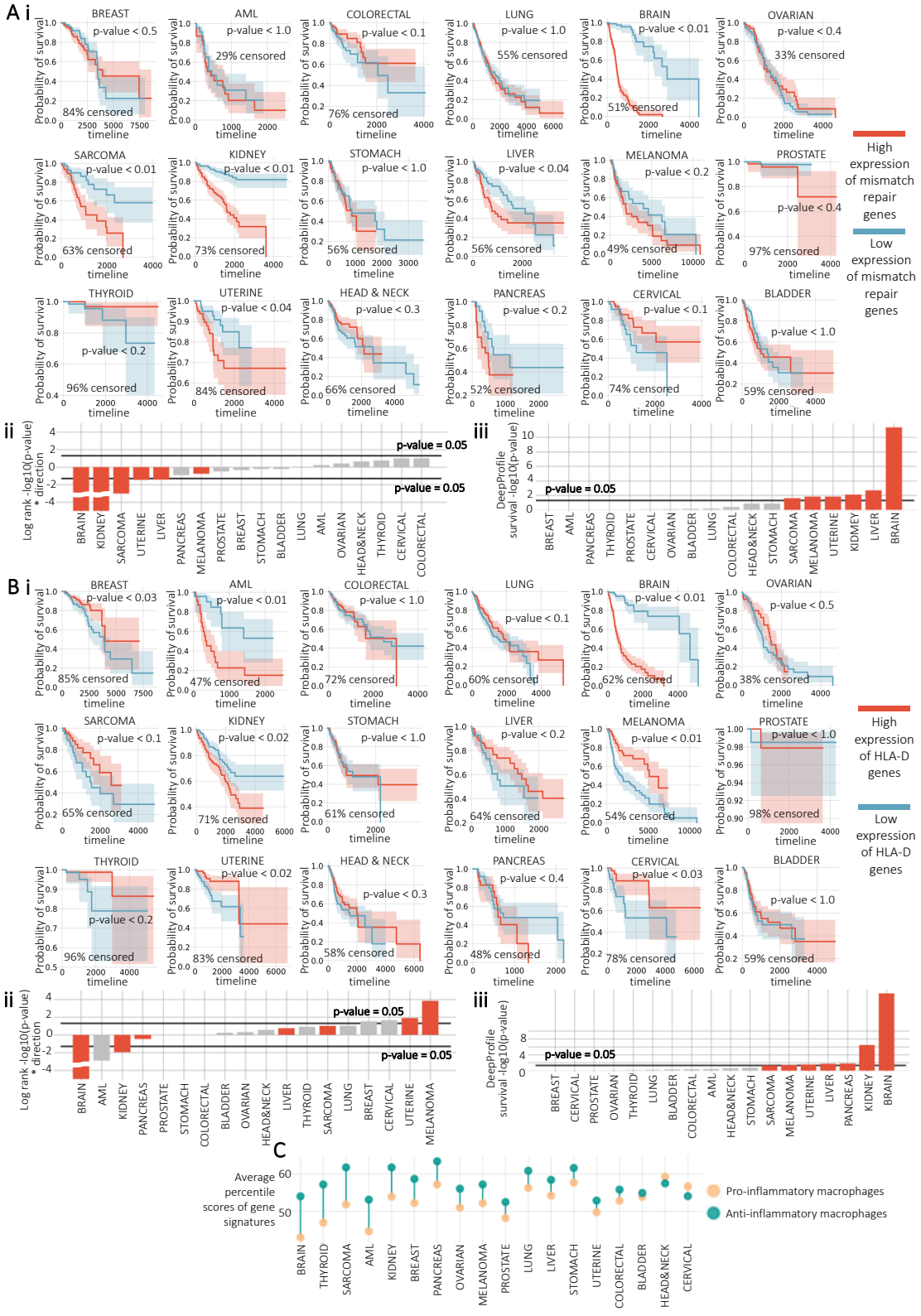

Figure A.8: Downstream survival analysis (see caption on next page).

Figure A.8: **(A)** Mismatch repair pathway survival analysis.
(i) Kaplan-Meier plots of average expression of KEGG mismatch repair pathway. The samples with an expression above (mean + one standard deviation) are marked as highly expressed and below -(mean + one standard deviation) are marked as lowly expressed. The log rank test p-values and the percent of censored samples are reported for each cancer.
(ii) Plot of log rank test p-values and the direction from Kaplan-Meier plots. The cancer types are sorted by the direction of association and the -log10(p-values).
(iii) Plot of DeepProfile survival p-values for KEGG mismatch repair pathway. The cancer types are sorted by the -log10(p-values).
(B) MHC class II pathway survival analysis
(i) Kaplan-Meier plots of average expression of HLA-D genes in Reactome MHC class II antigen presentation pathway.
(ii) Plot of log rank test p-values and the direction from Kaplan-Meier plots.
(iii) Plot of DeepProfile survival p-values for Reactome MHC class II antigen presentation pathway.

Chapter B

Appendix B; Appendix to Adversarial deconfounding autoencoder for learning robust gene expression embeddings

The work presented in this chapter is adapted from the supplementary material of [105], previously published by *Bioinformatics*.

B.1 Investigating the Effect of Number of Latent Nodes on AD-AE Performance

Adversarial Deconfounding Autoencoder aims to learn deconfounded and robust embedding; thus, we wanted to show that we can learn biologically meaningful embeddings independent of the latent embedding size. We experimented on the KMPlot expression [109] for a wide range of number of latent nodes: 10, 25, 50, 100, and 150. For each latent dimension size, we tuned the hyperparameters of AD-AE using 5-fold cross-validation.

To show that AD-AE latent spaces are deconfounded and biologically meaningful, we created UMAP plots [138] of AD-AE embeddings from all different dimension sizes and colored the plots by dataset label,

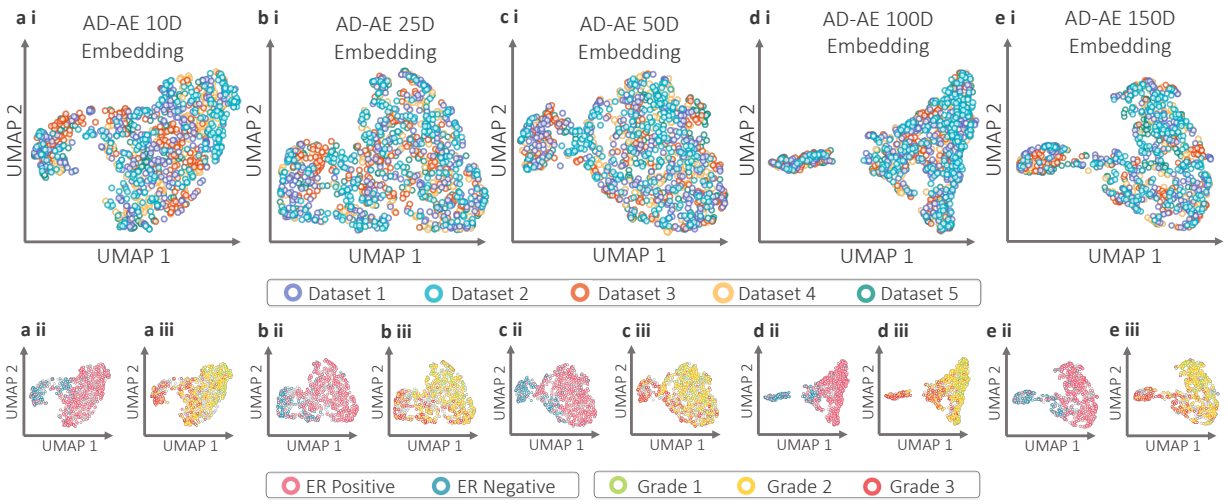

Figure B.1: UMAP plots of embeddings generated by AD-AE with embedding size (a) 10, (b) 25, (c) 50, (d) 100, and (e) 150. Subplots are colored by (i) dataset, **(ii)** ER status, and (iii) cancer grade. The gray dots denote samples with missing labels.

Estrogen Receptor (ER) status, and cancer grade (Figure B.1). First, we highlight that all the different dimensional embeddings are highly deconfounded where it is not possible to distinguish the samples by dataset label. Furthermore, the samples can be clearly separated by both by ER label and cancer grade. These results indicate that AD-AE can learn biologically informative embeddings independent of the number of latent nodes.

We also wanted to examine the effect of the number of latent nodes on the prediction performance. Thus, we predicted ER and cancer grade on the AD-AE embeddings with different sizes and reported the internal and external test set prediction performances. We trained AD-AE for each embedding size 10 times with different random initializations and averaged the prediction performances across 10 runs. We observed that the prediction performances across different runs are very similar to each other, indicating the ability of AD-AE to successfully predict biological variables of interest independent of the latent embedding size (Figure B.2). We also noticed that the embeddings with the smallest (10) and the largest size (150) obtained slightly worse performances compared to other sized embeddings, especially for the external test set (Figure B.2 a ii & b ii). This result indicates that the AD-AE performance might be negatively affected if the number of latent nodes is too small such that we cannot preserve important biological signals or too large such that we cannot compress the latent space that can extract informative manifolds.

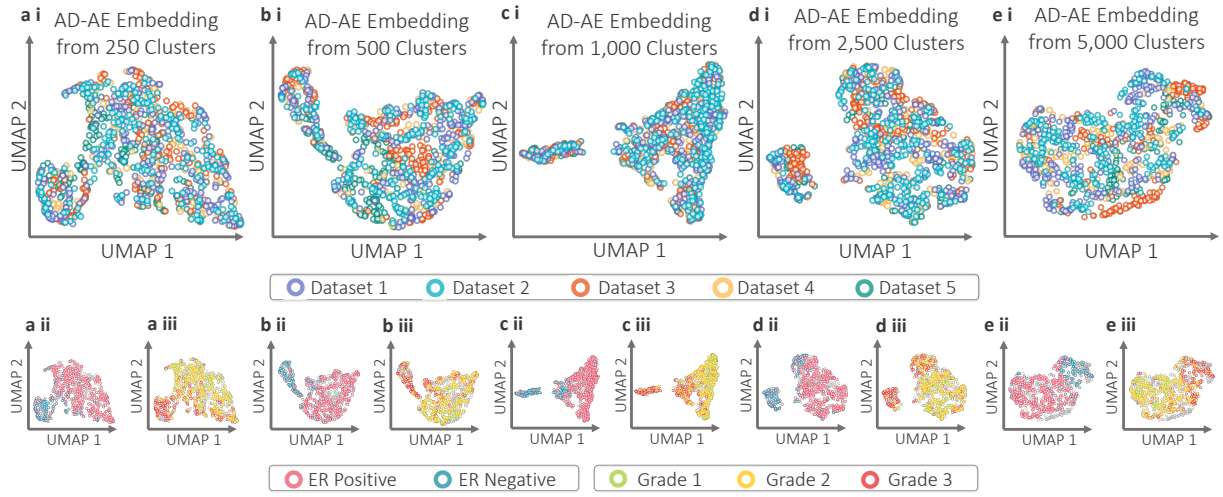

Figure B.2: (a) Estrogen Receptor (ER) prediction plots for (i) internal test set, and (ii) external test set. (b) Cancer grade prediction plots.

B.2 Investigating the Effect of Clustering The Expression Measurements on AD-AE Performance

We observed an improvement in the autoencoder models' performance when we clustered the expression measurements with k-means++ clustering [135] and passed the cluster centers as inputs to our autoencoder models. We wanted to examine further the effect of the number of clusters on the AD-AE performance and highlight the robustness of AD-AE. Accordingly, we clustered the expression matrix using a wide range of number of clusters: 250, 500, 1000, 2500, 5000. We used these cluster centers as inputs of AD-AE and tuned the hyperparameters for each model separately with 5-fold cross-validation. We then created the UMAP plots colored by dataset, ER, and cancer grade labels. Figure B.3 shows that for the wide range of the number of clusters we experimented on, AD-AE learned deconfounded embeddings while preserving the biological signals since we can successfully separate the samples by both ER and cancer grade labels. This analysis demonstrates that AD-AE can successfully learn informative embeddings independent of the number of clusters we select as a preprocessing step.

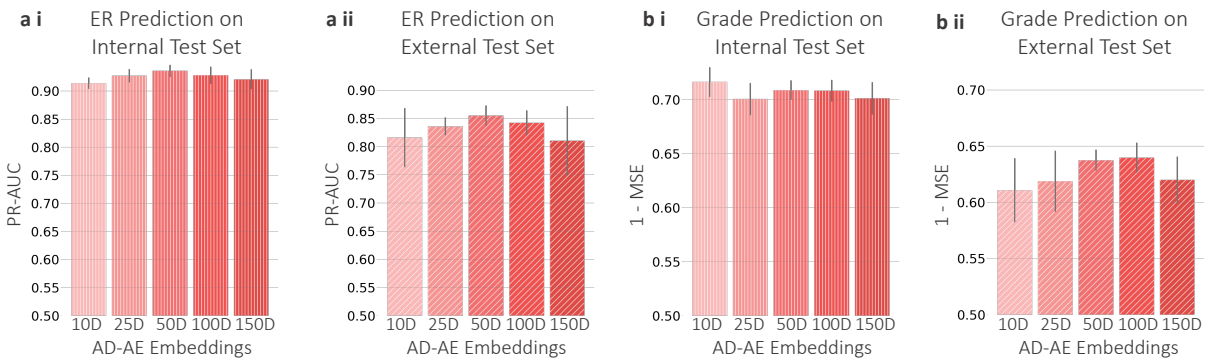

Figure B.3: UMAP plots of embeddings generated by AD-AE trained on (a) 250, (b) 500, (c) 1000, (d) 2500, (e) 5000 expression measurement cluster centers. Subplots are colored by (i) dataset, **(ii)** ER status, and (iii) cancer grade. The gray dots denote samples with missing labels.

B.3 Subsampling Experiment for Transferring Models Across Confounder Domains

AD-AE aims to learn deconfounded and generalizable embeddings that can successfully transfer across different confounder domains. In Chapter 3.3.3, we showed that AD-AE can successfully transfer across male and female sample domains even though the cancer subtype distributions were different for males and females (Figure 3.7). This analysis also demonstrated that AD-AE successfully handles class imbalance that commonly occurs in domain shift [139].

In other to investigate whether AD-AE can preserve its advantage when the subtype classes are balanced, we carried the same transfer experiment in Figure 3.7 for a balanced distribution on TCGA brain cancer expression [133; 132; 134]. To simulate a balanced distribution, we subsampled from the higher sample numbered class. More specifically, for female and male samples separately, we subsampled from LGG class to ensure that we have an equal number of LGG and GBM samples in both male and female samples. This provided us with 108 male and 108 female samples with 54 samples from each subtype class. We then carried out the transfer experiments where we train our predictor on female sample embeddings and then transfer to male sample embeddings and vice versa. We repeated the subsampling five times and averaged the prediction across five runs. We observed that, again, in both transfer directions, AD-AE outperforms all the other methods (Figure B.4). We also notice that since we are simulating a balanced distribution, the PR-AUCs between the male and female external prediction performances are negligible.

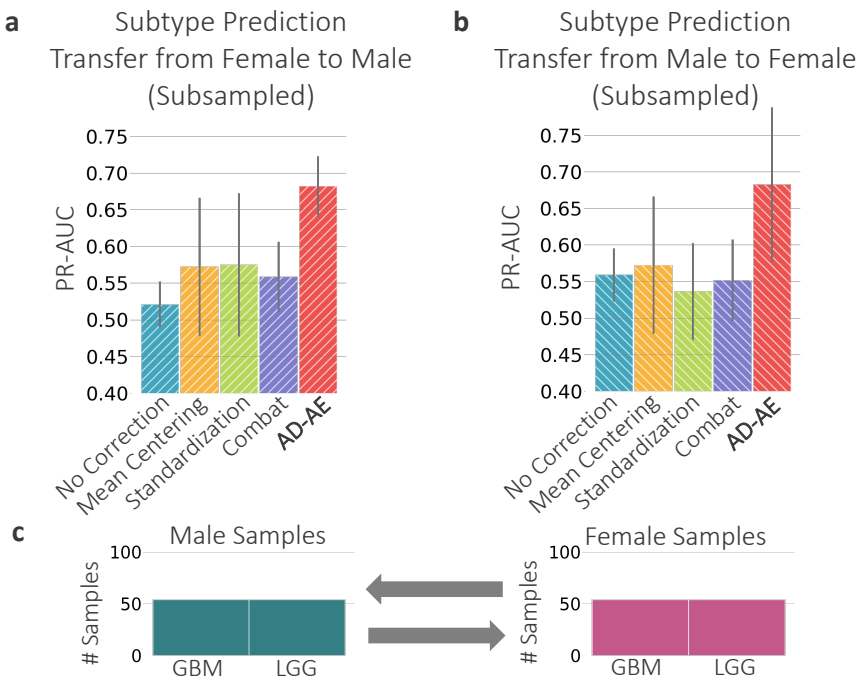

Figure B.4: Glioma subtype prediction plots for (a) model trained on female samples transferred to male samples and (b) model trained on male samples transferred to female samples. We subsampled from female and male classes to have an equal number of samples from the two subtypes. (c) Subsampled subtype label distributions for male and female samples.