Machine learning methods for biological hypothesis generation, facilitating new discoveries at lower costs

Adelaide Woods Chambers Woicik

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2024

Reading Committee:
Sheng Wang, Chair
Bill Noble
Linda G. Shapiro

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Machine learning methods for biological hypothesis generation, facilitating new discoveries at lower costs

Adelaide Woods Chambers Woicik

Chair of the Supervisory Committee:

Sheng Wang
Computer Science and Engineering

Machine learning methods for biological data have become increasingly popular in recent years, acknowledging the transformative applications, complex patterns, and latent variation underlying biological systems. Importantly, many biological measurements are very expensive to produce experimentally. This poses challenges for biological discovery, limiting the number of experiments that can practically be conducted, and for data-hungry machine learning methods, which may require massive datasets that are not publicly available. One approach to these challenges is computational simulation with generative machine learning models, leveraging available high-quality data from heterogeneous sources to synthesize additional datapoints for subsequent analyses, which can help propose novel and prioritize existing biological hypotheses that can subsequently be tested in an experimental lab. In this thesis, I present three methods for high-quality *in silico* data generation across three biological domains: genomic time series extrapolation with Sagittarius, high-resolution dense chromatin contact map generation with Capricorn, and approximately-automatically-curated gene network generation using augmented network integration with Gemini. These diverse applications focus on high-cost experimental data, highlighting the immense value of computational datapoint simulation, and heterogeneous biological measurements, requiring methods that account for the diverse inputs and leverage all sources of information to improve the generation process. Finally, I connect each model back to its practical applications in biology, ranging from assisting biological experts in their current work to novel hypothesis generation.

Acknowledgements

I'd like to thank the many people who helped make the work in this thesis possible. First and foremost, I'd like to thank my advisor, Professor Sheng Wang, for his guidance during the PhD. I'd also like to thank my other committee members Professors William Noble, Linda Shapiro, Simon Du, and Daniela Witten for their helpful feedback and support.

I'd also like to thank the members of the Wang lab for the useful collaborations and discussions throughout my tenure at the University of Washington. In particular, I'd like to thank Hanwen Xu, Tangqi Fang, Yifeng Liu, Mingxin Zhang, Zixuan Liu, Tong Chen, and Guang Yang. Furthermore, I'd like to acknowledge the members of the working computational Hi-C modeling group, including Anupama Jha, Xiao Wang, Borislav Hristov, Gang Li, and Shengqi Hang, for teaching me so much about underlying cellular biology and for all of their advice in the past two years. Finally, I'd like to thank Doctors Jay Pal and Shin Lin for their support in the past year, and for teaching me about various opportunities for generative machine learning in clinical settings, and particularly in cardiology.

DEDICATION

To my family: my husband, Matt Woicik; my dad, Craig Chambers; and my sister, Caitlin McWilliams.

And for my mom, Sylvia Chambers.

All my love, always.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Biological and medical discoveries have the potential to bring significant impact on peoples' lives. Penicillin, the first antibiotic, has been credited with saving hundreds of millions of lives since its initial discovery in 1928 by Drs. Fleming, Florey, Chain, and Heatley [1]. In 1952, cells taken without consent from Mrs. Henrietta Lacks's cervical cancer biopsy were found to grow and divide continuously in a laboratory. The HeLa cell line was the first "immortal" human cell line, and has since contributed to over $100{,}000$ scientific papers [2, 3], including the development of multiple anti-cancer drugs [4, 5, *inter alia*] and a method to distinguish cancerous and non-cancerous cell populations that has remained the standard-of-care for over 65 years [6]. Reverse transcriptase, discovered in 1970 by Drs. Baltimore and Temin, has enabled our understanding and treatment of diseases including human immunodeficiency virus (HIV) [7] and acquired immunodeficiency syndrome (AIDS) [8].

However, these discoveries are very expensive. For instance, a study found that pharmaceutical companies spent an average of $6.16 billion per drug approved by the US Food and Drug Administration (FDA) from 2001-2020 [9]. Other significant discoveries were at least partially serendipitous, like the discovery of penicillin [1]. Here, I focus on two main challenges for biomedical discovery. First, the space of possible experiments to run is massive. Second, each experiment is costly. In different settings, either the first or second challenge may pose a more or less impediment to biomedical research. However, when considered together, these challenges often mean that a straightforward, exhaustive approach to biological discovery, where all possible experiments are conducted, is infeasible.

A standard computational analysis pipeline uses curated, high-quality data from expensive biological experiments for downstream machine learning. There are many exciting opportunities in this canonical

$$\text{biological data generation} \xrightarrow{\text{data ingestion}} \text{computational analysis}$$

approach. These workflows have enabled valuable discoveries, including the derivation of the *BRCA1* cancer oncogene [10] and the identification of three-dimensional functional genomic structures like chromatin loops [11]. In recent years, machine learning has undergone a transformation, trending towards ever larger models trained on ever more expansive datasets, including the recent advent of foundation models [12–14, *inter alia*]. This line of work has also extended to computational biology, including massive models for deoxyribonucleic acid (DNA) sequence [15–17], protein sequence [18–20], and medical images [21, 22].

Importantly, these models also create opportunities for machine learning to help inform biological experiments and alleviate the challenges of the number and cost of experiments needed for biomedical discoveries. Generative models [12, 14, 23, 24, *inter alia*] have become more powerful, learning underlying patterns of a data distribution to simulate novel, realistic samples. Such models can then follow the earlier paradigm, training on the small, expensive, and high-quality biological datasets, but be used to cheaply approximate many more unobserved measurements during inference. These *in silico* measurements therefore reduce both sources of experimental cost: many experiments can be run (inferred) with a single call to the model, thereby making the massive experimental space much more searchable, and each experimental run has negligible cost after model training. Finally, these *in silico* results can be reincorporated back into the biological experiment pipeline by informing future hypothesis generation with the new paradigm

$$\text{simulated measurements} \xrightarrow{\text{hypothesis generation}} \text{biological experiment design.}$$

For example, if a model suggests that a drug might benefit a patient population that it had not previously been applied to, future biomedical experiments could prioritize this combination.

This thesis focuses on the loop between biological experiments and computational analyses shown in Figure 1.1, and particularly on methods to improve biological hypothesis generation in high-cost settings where computational methods may have the most impact. I describe novel methods for synthetic datapoint generation across three diverse data types, each with different primary expense sources.

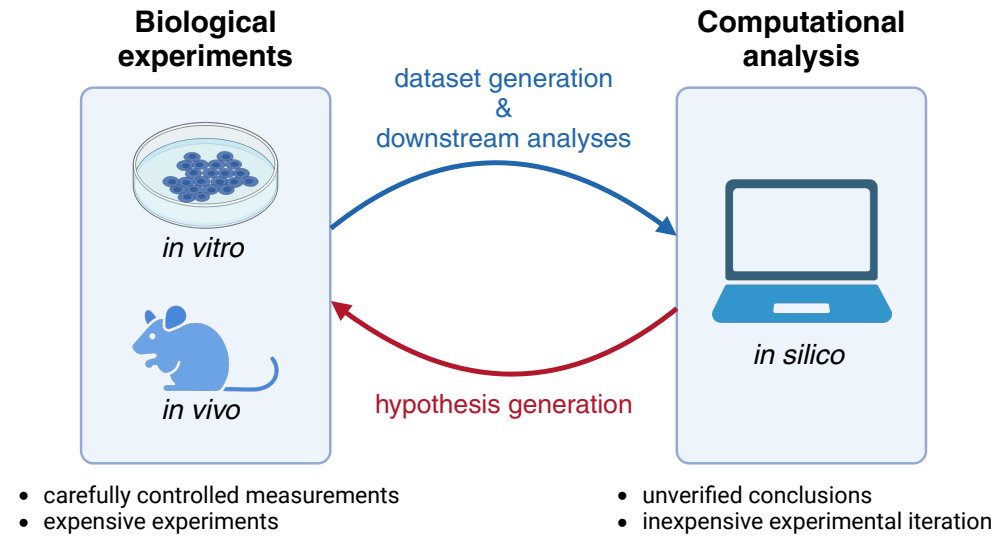


Figure 1.1: **The information loop between biological experiments and computational analyses.** Biological experiments, encompassing both *in vitro* (cell line) and *in vivo* (organism) experiments, produce high-quality measurements in carefully controlled study conditions, but are very expensive. Machine learning analysis, including *in silico* (computational) measurement simulation, is comparatively very inexpensive but may produce some spurious results. Experimental datasets are ingested by computational pipelines for various downstream analysis (blue arrow); results from computational analyses can also inform good directions for future experiments by aiding novel biological hypothesis generation. Created with `BioRender.com`.

1.1 Biological datasets: challenges and opportunities

Biological and medical analyses present great possibilities for scientific advancements. Unlike many other fields of machine learning, such as large language models (LLMs) that generate satirical reports [25] or stylized corgi paintings [14], human knowledge and ability are often not the gold standard. The underlying complexities of biology require much over-simplification and abstraction for humans to reason about, and there remain many open problems. Excitingly, this means that computational methods can be used to generate new biological hypotheses, which are then subsequently tested in experimental settings.

I specifically focus on computational generation of *in silico* measurements from lower-cost experimental measurements. These simulated datapoints can then be used to boost downstream analyses for the hypothesis generation step in Figure 1.1. Importantly, these computational methods should be highly sample-efficient to account for the scarcity of experimental data and the high costs of generating additional data. Therefore, effectively leveraging the heterogeneous data available for the task at hand is critical. Furthermore, variations between distinct datapoints may appear small to untrained eyes; this means that biologically-motivated

tasks must be included for the *in silico* datapoint evaluation task, rather than adhering entirely to traditional machine learning metrics like test loss or mean squared error.

1.2 *In silico* generation of high-cost biological data

In this work, I present machine learning methods to mitigate the high costs of *in vitro* or *in vivo* experimental studies, leveraging low-cost data and *in silico* augmentation to enable subsequent biological analyses that were not successful given only the low-cost data. I focus on three main sources of experimental expense:

1. studies that require many measurements over many years, stretching experimental costs;
2. studies for which a single, high-quality measurement is very expensive; and
3. studies that require expert curation, imposing high data annotation costs.

To investigate these three different types of costs, I investigate three distinct biological settings. First, I leverage gene expression time series data to study temporal and combinatorial extrapolation in high-cost, repeated measurement settings. Second, I investigate chromatin structure contact matrix resolution enhancement to generate higher-quality, higher-cost measurements from lower-quality, lower-cost experimental measurements. Third, I study network integration from minimally-curated gene network repositories to highlight the potential for automatic approximate curation with machine learning methods.

1.3 Roadmap

This thesis is structured as follows. In Chapter 2, I provide some context on the biological data described in this thesis for more computational readers, as well as an overview of two key machine learning model formulations used in this thesis. Then, in Chapter 3 I focus on costs stemming from many repeated measurements over long study durations, introducing the Sagittarius model [26] for gene time series extrapolation over both time ranges and studied conditions. In Chapter 4 I move on to costs that arise from a single, high-quality measurement, showing that lower-cost measurements of the genome's three-dimensional structure can be computationally enhanced with Capricorn [27] to capture small-scale structural features. In Chapter

5 I wrap up with costs due to expert curation, and introduce the Gemini method [28] for automatic approximate curation of gene network repositories. Finally, Chapter 6 discusses common themes across the three areas, and suggests additional directions for future work.

Chapter 2

Background

In this chapter, I provide a high-level overview of the biological data and machine learning methods that I discuss in this thesis. In Section 2.1 I provide brief explanations of how the biological data are collected; however, I focus principally on the data inputs from a computational perspective, and how they differ from other, non-biological data types. Section 2.1.1 covers genomic time series (the relevant data type for Chapter 3); Section 2.1.2 covers high-throughput chromosome conformation capture (Hi-C) contact maps (the relevant data type for Chapter 4); and Section 2.1.3 covers gene networks (the relevant data type for Chapter 5). In Section 2.2, I next present two types of machine learning architectures that are used in this thesis. I aim to provide intuition for these models, particularly for a non-computational or non-machine-learning audience. Section 2.2.1 provides an overview of the transformer model (relevant for Chapter 3) and Section 2.2.2 explains the key components of a diffusion model (relevant for Chapter 4). Further mathematical definitions and architecture details are provided in later chapters.

2.1 Biological data types

2.1.1 Genomic time series

Genomic time series data measure information about the genome at multiple points in time. In this work, I will mainly discuss gene expression, though also briefly introduce somatic mutations as the genomic profiles of interest. Subsequent sections focus on the biological details of each measurement type; however, both

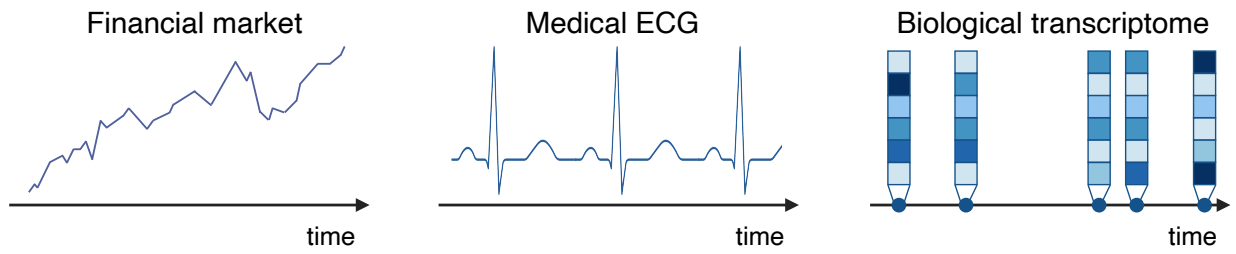


Figure 2.1: **Comparison of different kinds of time series sometimes modeled computationally.** Unlike time series seen in applications like financial forecasting (*left*) or medical time series like electrocardiograms (ECG, *middle*), biological time series like temporal gene expression measurements (*right*) often contain relatively few, high-dimensional measurements. Created with `BioRender.com`.

experimental assays produce a vector over the M genes included in the experimental study.

It is important to acknowledge several key distinctions between genomic time series and other types of time series sometimes modeled computationally (Figure 2.1). First, unlike more commonly seen financial or medical time series, such as stock prices over time or a medical electrocardiogram, measurements cannot reasonably be approximated as continuous over time. Relative to the scale of time an experiment covers, actual measurements are quite rare. Second, these experiments are carefully controlled for surrounding environmental conditions, so additional study costs scale with the study duration. Third, the measurements themselves are also often high-dimensional, such as a vector representation of gene expression measurements. Fourth, because the measured cells or organisms are often destroyed as part of the data acquisition process [29], additional samples and timepoints can be cost-prohibitive to procure.

Furthermore, irregular and unaligned temporal sampling is more common in biological time series datasets. *Irregular* means that the time gap between adjacent measurements varies over the course of the time series (unlike daily measurements in financial market time series or pseudo-continuous readings from electrocardiograms, for instance). The time series are also much sparser than time series in many other domains with potentially long time gaps between successive measurements, further compounding the irregular sampling challenge. *Unaligned* means that, given multiple time series in a dataset, the timepoints that do contain measurements may differ. This is often in contrast to other applications, where stock prices are measured at the same time across firms. Both the irregularity and lack of alignment present time series modeling challenges for biological measurements that are distinct from other domains.

Gene expression

Here, I briefly outline gene expression measurements. In particular, I focus on ribonucleic acid sequencing (RNA-seq) [29]. At a high level, the same deoxyribonucleic acid (DNA) is shared by all cells in a multicellular organism; however, different organs and cell types are able to carry out unique roles in the body by using the same DNA differently. This can involve *differential gene transcription*, where some genes are more activated than others in specific cells, tissues, or environments.

To quantitatively measure gene activity, RNA-seq acts after the DNA is transcribed to RNA, and measures the quantity of RNA found in a cell. In particular, RNA-seq first extracts the messenger RNA (mRNA) molecules found in a studied population of cells; next, the fragments of mRNA are converted to complementary DNA (cDNA) through reverse-transcription; then, the cDNA fragments are sequenced and finally mapped back to the genome, indicating which gene was "on" at the time of the study. The number of reads that map to a given gene provides a quantitative measure of that gene's relative activity in the sample. As of 2017, the ENCODE [30] data portal recommended a minimum of 30 million successfully aligned reads per gene expression measurement replicate[1]. To put these numbers in context, humans have approximately $20{,}000$ protein-coding genes covering almost 60 million genomic base pairs [31].

As an input to a computational pipeline, a gene expression experiment measurement can be represented as a non-negative integer-valued vector $\mathbf{x} \in \mathbb{N}^M$, where M is the number of genes considered in the experiment and $\mathbf{x}[m]$ is the number of reads aligned to the mth gene in the experiment.

Somatic mutations

I now provide an overview for somatic mutation profiles. Organisms naturally have mutations in their DNA due to mistakes during chromosome replication. Some of these mutations, called *germline mutations*, occur during reproduction and are inherited by the offspring, forming part of the basic evolutionary process. Others, called *somatic mutations*, are acquired over the course of an organism's life. In the time series context, I focus on somatic mutations, since these change over time. Such mutations are of particular interest in the study of cancer, which is often driven by specific types of somatic mutations [32].

Mutations can be measured by directly sequencing the DNA available in sampled cells and mapping the

[1]Guideline for small RNAs `https://www.encodeproject.org/data-standards/rna-seq/small-rnas/` and long RNAs `https://www.encodeproject.org/data-standards/rna-seq/long-rnas/`.

reads back to a reference genome. Differences between the studied genome and the reference genome are then considered mutations (variants) [33]. While not all mutations are pathological, mutation profiling for a subset of genes forms part of the standard-of-care for several diseases including cancer [34].

Computationally, a somatic mutation profile can be represented as a binary vector $\mathbf{x} \in \{0,1\}^M$ for M tested genes, where $\mathbf{x}[m] = 1$ if and only if the mth profiled gene contains an acquired mutation.

2.1.2 Hi-C contact maps

In Section 2.1.1 I discussed differential gene transcription, where genes are activated differently in different cells despite sharing the same DNA sequence. One mechanism for this differential transcription is the genome's three-dimensional (3D) structure, which I turn to now. High-throughput chromosome conformation capture (Hi-C) is an experimental protocol that identifies pairs of genomic loci that have come into contact with each other; the frequency of such contacts is then measured over a population of cells. In this section, I provide a brief overview of chromosome conformation capture protocols, which include Hi-C (Section 2.1.2). Then, I focus on the outputs of Hi-C experiments, which are later inputs to the computational pipeline.

Chromosome conformation capture

Several types of chromosome conformation capture (3C) protocols exist to measure the 3D structure of the genome. The key idea behind these protocols is that, at low DNA concentrations, nearby strands of DNA are much more likely to be involved in a biological interaction than to be co-located by chance, a concept known as *proximal ligation* [35]. The 3C family of approaches have similar initial experimental steps:

1. cross-link the chromatin with a fixation step, most commonly using formaldehyde, to bind DNA fragments that are interacting with each other;

2. use a restriction enzyme to digest the chromatin; ligate the cross-linked DNA in experimental conditions that favor proximal ligation; and

3. reverse cross-link to recover the paired DNA fragments.

Method	Year	Mapping	DNA matching	Chromatin digester	Genome wide?
3C	2002	one-versus-one	PCR	restriction enzyme	
4C	2006	one-versus-all	microarray, NGS	restriction enzyme	
5C	2006	many-versus-many	PCR	restriction enzyme	
ChIA-PET	2009	all-versus-all	NGS	restriction enzyme	✓
Hi-C	2009	all-to-all	NGS	restriction enzyme	✓
Micro-C	2020	all-to-all	NGS	micrococcal nuclease	✓

Table 2.1: **Comparison of 3C-derived methods.** Brief comparison of 3C [37], chromosome conformation capture-on-chip (4C) [38], chromosome conformation capture carbon copy (5C) [39], chromatin interaction analysis by paired-end tag sequencing (ChIA-PET) [40], high-throughput chromosome conformation capture (Hi-C) [41], and micro-C [42]. We list the method publication year, type of locus-to-locus mapping measured by the assay, method for matching paired DNA back to the genome, method for chromatin digestion after fixation, and whether the assay scales to the full genome. We describe ChIA-PET as an all-versus-all assay, although each individual experiment must select a specific protein to study.

After this common approach to identify pairs of interacting DNA sequences, various 3C-based methods use next generation sequencing (NGS) or polymerase chain reaction (PCR) approaches to match the short sequences back to their corresponding genomic loci [35, 36]. We provide a brief summary of some 3C-derived protocols in Table 2.1. In this thesis, we specifically use Hi-C data; however, the proposed methods extend to other genome-wide, all-to-all chromatin structure contact maps.

The 3C-derived methods output a *contact map*, represented as a matrix $\mathbf{X}$ of pairwise interaction frequencies between genomic loci [37], such as the example shown in Figure 2.2. In the case of Hi-C or other all-to-all genomic mapping techniques in Table 2.1, the contact map counts how many times two genomic loci were found to interact experimentally. During the matrix processing, the genome is binned into groups of loci, such that $\mathbf{X}[i, j]$ indicates the number of experimental reads, post any additional quality control, where the 3C experiment sequenced a paired DNA fragment with one DNA fragment matching a locus in the ith bin of the genome and another DNA fragment matching a locus in the jth bin of the genome.

Contact maps

Two key experimental and post-processing parameters impact the contact map matrix generated by Hi-C studies: *read depth* and *resolution*.

First, the Hi-C experimental settings define the read depth. This defines the number of pairs that are sequenced and mapped back to specific sequences of DNA. Consequently, the read depth determines the maximum possible number of contacts in the resulting contact matrix, though some reads are excluded due to uncertain genomic matches or other data quality issues. As of February 2017, the 4D Nucleome (4DN)

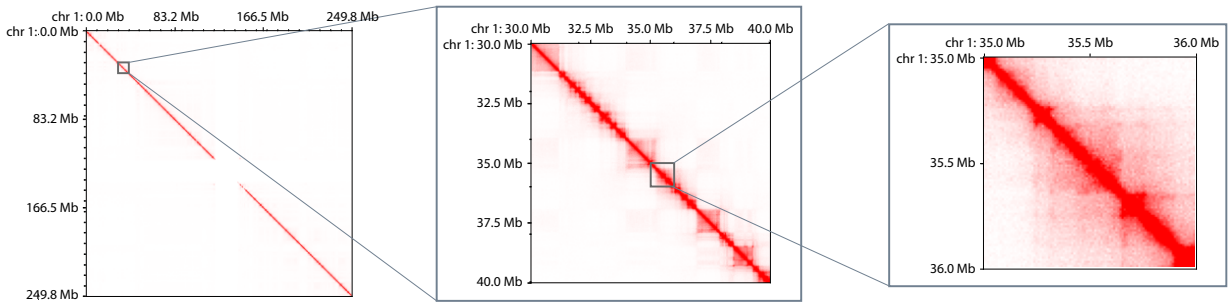


Figure 2.2: **Example Hi-C contact map showing *cis* interactions at different resolutions.** *Left*, the interaction frequency matrix for chromosome one from the GM12878 cell line [11], shown at 250 kilobase (kb) resolution. *Middle*, the same interaction frequency data from small gray square on the left plot, covering loci in the 30 megabase (Mb) to 40 Mb range of chromosome one, is shown at 20 kb resolution. *Right*, the same interaction frequency from the small gray square on the middle plot, covering loci from 35 Mb to 36 Mb of chromosome one, is shown at 10 kb resolution.

project [43] has maintained a standard that all experimental *in situ* Hi-C [11] replicates contain a minimum of 500 million paired reads[2]. Unlike gene expression (Section 2.1.1), Hi-C contact *matrices* are often much sparser than the gene expression *vectors* due to the pairwise nature of the measured interaction.

Second, the analysis determines the resolution. In theory, the paired reads could give us near-base-pair level resolution for genomic contacts, though this is sensitive to the choice of restriction enzyme [44]. However, given the length of the genome (approximately 3 billion base pairs in human) and the experimental read depths used in practice ($\geq$ 500 million), the resulting matrix would be extremely sparse and not informative. The analysis consequently determines the number of genomic base pairs Δ to group or bin together and model as a single locus. If any base pairs in that Δ-long region come into contact with base pairs in another Δ-long region, those two genomic regions are considered to come into contact with each other. Therefore, each entry in the analyzed contact matrix covers an $O(\Delta^2)$ set of possible base-pair-base-pair interactions.

Importantly, there is also an inherent tradeoff between contact map resolution and sparsity: higher analysis resolutions enable more precise attribution of interacting loci, but require exponentially more experimental reads to maintain the same measurement density in the contact matrix, and are therefore much more expensive.

From a computational perspective, contact map inputs can be represented as a symmetric, non-negative

[2]Standard defined at `https://drive.google.com/file/d/1-NEldtpuDuYXcbWngETltNPCRv0RZlIK/view`.

integer-valued matrix $\mathbf{X} \in \mathbb{N}^{L/\Delta \times L/\Delta}$ for a genome containing L base pairs and contact map resolution Δ. In this representation, $\mathbf{X}[i, j]$ is the number of experimental reads mapped back to DNA sequences contained in genomic loci i and j, and $\sum_{i=1}^{L/\Delta} \sum_{j=i}^{L/\Delta} \mathbf{X}[i, j] = C_q$, where C_q is the total experimental read count that passed quality control filtering.

2.1.3 Gene-gene networks

Finally, I also consider gene network repositories, made up of gene-gene networks. Fundamentally, we can form graphs that represent many different kinds genomic evidence. I define a *gene-gene network* as a graph where every vertex in the graph represents a gene, and edges tell us something about the nature of the relationship between its two anchor genes. The edges may be weighted or unweighted, depending on the goals of the analysis and the sources of experimental evidence. For example, we could take the results of several gene expression measurements [29] (Section 2.1.1) and form a weighted edge based on how frequently two protein-coding genes were jointly activated (co-expression, Figure 2.3). Similarly, we could take a Hi-C contact map [41] (Section 2.1.2) and form an unweighted edge between two genes if and only if their corresponding genomic loci were found to come into contact with one another (co-localization).

I further restrict the dataset of different gene-gene networks that are jointly considered (also referred to as the gene network repository of interest) to share the same vertex set. In practice, this is a reasonable assumption: many gene network repositories focus broadly on the same setting, such as protein-coding genes in a species of interest, which fixes the vertex set [45, 46]. In theory, network sets that do not fit this definition can also be coerced to share the vertex set, either by excluding rare vertices that occur in only a few networks, introducing non-connected components for vertices that are missing from only a few networks, or by forming supernodes that are shared by all networks and operating over those.

Notably, the relative cost of each network in the gene network repository differs: in Figure 2.3, the network derived from gene co-expression would be much less expensive than a high-quality co-localization network. Consequently, we might expect a repository with many networks to contain more co-expression networks than co-localization networks, even though the genome's 3D structure is largely upstream of gene transcription in biology's central dogma, DNA $\rightarrow$ RNA $\rightarrow$ protein. Expert curation can help correct for some of these sources of imbalance, for instance by constructing a single co-expression-based network and

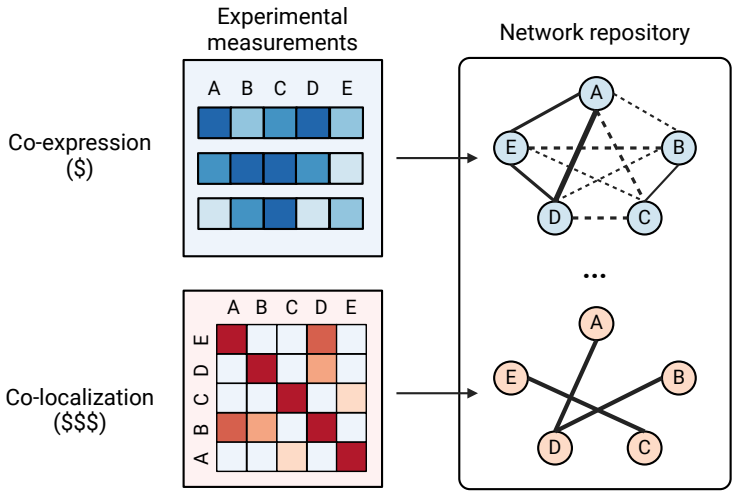


Figure 2.3: **Illustration of a gene network repository with associated experimental data.** *Top*, a less expensive gene expression dataset can produce a co-expression gene network. *Bottom*, a more expensive contact matrix dataset can produce a co-localization gene network. Figure created with `BioRender.com`.

single co-localization-based network that each account for the different cell types, genetic mutations, and other experimental conditions contained in the repository. However, this is a challenging, time-consuming, and knowledge-intensive curation task. Therefore, manual curation is very expensive in addition to the cost of the experiments underlying each network.

Computationally, we can represent the gene-gene network repository as $\boldsymbol{\mathcal{G}} = \{G_1, G_2, \ldots, G_M\}$ for M total networks in the dataset, where $\forall i \in \{1, \ldots, M\}, G_i \in \mathbb{R}_{\geq 0}^{N \times N}$ for N studied genes. The individual networks G_i may therefore be directed or undirected and weighted or unweighted, but may be represented with their adjacency matrix over the same set of graph vertices (genes).

2.2 Machine learning models

2.2.1 Transformers

Transformers are a class of machine learning model that has totally revolutionized deep learning practice. First introduced by Vaswani et al. [47] for natural language processing (NLP), the key idea behind transformers is that many distinct and possibly dispersed parts of an input can help inform our understanding of the input as a whole. This is in contrast to convolutional neural networks (CNNs) that were previously the

state-of-the-art, operating under the assumption that patterns form in localized regions, and larger patterns are themselves made up of localized smaller patterns. Since their original introduction, transformers have become much more widespread, including applications in images [48, *inter alia*] and biological sequences [15, 16, *inter alia*].

The essential operation in a transformer is *attention*, where learned representations for input components (usually called tokens) are informed by all other input components. Importantly, this attention mechanism is able to take the entire input into account when computing its representation rather than smaller chunks of the input, thereby expanding the model's field of view. At the same time, the attention mechanism is *permutation invariant*. That is, the ordering of the input components does not change the learned representation of the input as a whole. The permutation invariance encoded in the transformer's attention mechanism explicitly accounts for this behavior. In practice however, models often relax this constraint by including a *positional encoding* along with the input token in the attention mechanism, which enables the mechanism to develop some prior belief of token relevance based on proximity (e.g., related words in a sentence are more likely to be closer together). Therefore, the model can leverage information across the input sentence to best inform its understanding, while also learning the significance of word proximity that is encoded in the positional embeddings. Many of these attention mechanisms are then stacked on top of each other, forming the transformer.

2.2.2 Diffusion models

Diffusion models [49, 50], sometimes also called diffusion probability models, are a class of generative machine learning models based on principles from non-equilibrium statistical physics. Most frequently used for images [14, 51, 52, *inter alia*], a diffusion model learns how to simulate a realistic image output from input noise, which steers the generation process.

A diffusion model contains two main components: a *forward process* and a *reverse process*. During model training, the forward process steadily corrupts the image, adding noise to the input image in a carefully controlled way. This can take place in the "pixel space", meaning that the dimensions of the image do not change, and the result would still look like a noisy image, or in the "latent space", meaning that the result is an abstract, lower-dimensional representation of the higher-dimensional image, but would not be

visualized as an image itself. Noise is repeatedly added for a predetermined number of steps, until the result is effectively random. Then, the reverse process takes this fully corrupted image and gradually identifies structure, iteratively denoising the image to reconstruct the original input. After model training is complete, the model can then take random inputs and use the reverse process to generate a realistic image.

Diffusion models have become very popular in computer vision in particular as they produce photorealistic images [14, 51, 52], avoiding both the blurriness [23] and training instability [24] characteristic of some other generative architectures. These models can also be *steered* to produce tailored inputs rather than a fully random sample from the input training distribution. For instance, diffusion models can be accompanied by text inputs such that the user-defined prompt "photograph of an orange cat sitting on a bookshelf" will produce an image matching the given caption [14]. This can be obtained by providing an additional input to the reverse process that guides the denoising steps, such as a transformer-based representation of the input prompt (see Section 2.2.1).

2.3 Discussion

This concludes the necessary context for the biological data and machine learning methods presented in this thesis. The experimental data are incredibly valuable and informative; however, I have also tried to demonstrate how these biological data can be costly and challenging to obtain in their own right. Individual measurements in genomic time series are often not unreasonably costly, but the number of measurements and the duration of the study can introduce significant expenses. In contrast, it can be extremely expensive to conduct a single Hi-C experiment that produces a high-quality, dense contact matrix at high resolution. Finally, curation can present an additional expense after the initial experimental costs in settings like gene network repositories. In the following chapters, I will focus on extending computational methods like the transformers and diffusion models that I have described here to address and mitigate these costs.

Chapter 3

Sagittarius: Biological time series extrapolation

I first focus on biological data that are expensive to generate experimentally due to the need for many measurements across many experimental conditions. This chapter describes work originally presented in Woicik et al. [26][1].

Here, we consider *heterogeneous biological time series*, which require many repeated measurements and different study settings. In particular, I will focus on gene expression time series, where each individual measurement revels some information about the genetic activity in the profiled organism or cell line in question. Such time series datasets can incur significant costs. First, the measured cells or organisms are often destroyed as part of the data acquisition process using standard methods [29]; therefore, these time series are often not truly longitudinal, but instead control for the experimental and environmental conditions for repeated samples from the same cohort of interest. Second, acquiring *in vivo* samples requires caring for study organisms in carefully controlled environments, potentially for many years. This introduces significant costs for *in vivo* time series studies. Third, studies may include combinations of variables, such as pharmaceutical screens testing the impact of different drugs in different disease models [53]. The number of combinations grows exponentially with the number of studied drugs and disease models, so is wildly expensive to the point of infeasible in practice to fully screen all combinations.

[1] As part of this work, I contributed to conceptualizing the setting, method and experiments; I also ran the experiments, developed computational tools for Sagittarius, wrote the manuscript, and designed the figures.

To address these costs, I proposed Sagittarius [26], a transformer-based model to extrapolate gene expression profiles to unmeasured timepoints and experimental combinations[2]. One significant challenge for measurement extrapolation is that the experimental time series are *unaligned*: they measure different timepoints, have gene expression variation across study conditions, and the time series might proceed at different speeds. The key idea behind Sagittarius is its construction of a shared reference space for many heterogeneous time series measurements, which warps time and gene expression so that it can be compared across time series conditions. I named Sagittarius after Sagittarius A*, the supermassive black hole at the center of the Milky Way, which recently helped confirm predictions from Einstein's theory of relativity. This presents a most intriguing lens in which to study heterogeneous time series: space and time can be measured and compared from a single reference frame, although they may differ when viewed from different reference frames [54]. Sagittarius solves the challenge of time series alignment by mapping the sparse, unaligned measurements from heterogeneous time series to a shared, regularly-spaced reference space that can be used to accurately simulate genomic profiles at temporal extrema and under-measured or unmeasured experimental conditions.

In this chapter, I first cover the pressing needs and challenges of time series extrapolation in more depth. Next, I cover the Sagittarius model. Finally, I present results on several applications. I show Sagittarius's promising performance when extrapolating mammalian developmental gene expression [55], simulating drug-induced expression at unmeasured dose and treatment times [53], and augmenting datasets to accurately predict drug sensitivity. We also used Sagittarius to extrapolate mutation profiles for early-stage cancer patients [56], which enabled us to discover a gene set connected to the Hedgehog signaling pathway that may be related to tumorigenesis in sarcoma patients, including *PTCH1*, *ARID2*, and *MYCBP2*. By augmenting experimental temporal datasets with crucial but difficult-to-measure extrapolated datapoints, Sagittarius enables deeper insights into the temporal dynamics of heterogeneous transcriptomic processes and can be broadly applied to biological time series extrapolation.

[2]The code used in this chapter is available at `https://github.com/addiewc/Sagittarius`.

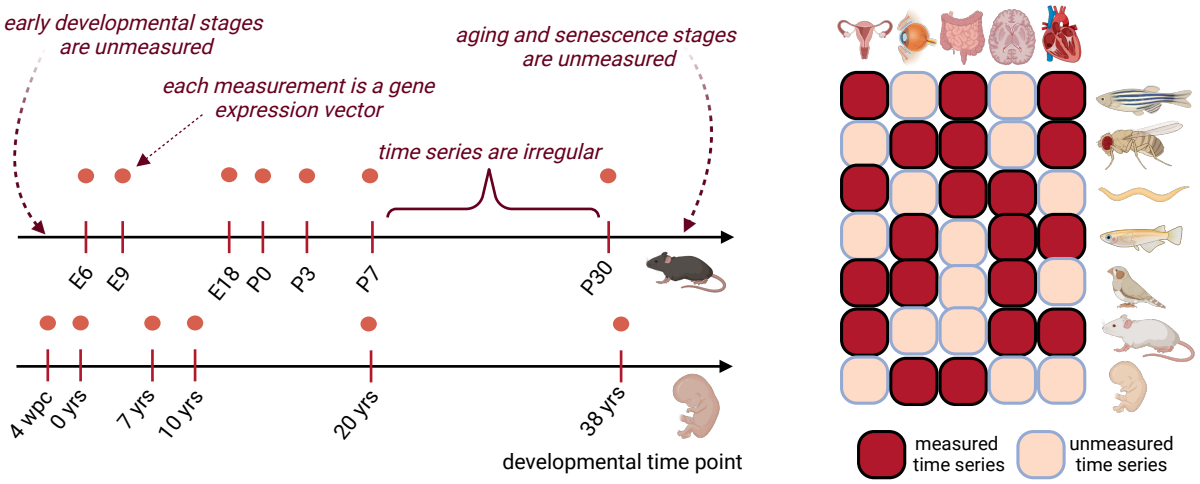
3.1 Biological time series extrapolation

The temporal dynamics of the transcriptome are key to the study of developmental biology, tumor biology [57, 58], immunobiology [59, 60], and pharmacogenomics [61, 62]. As bulk- and single-cell RNA-sequencing technologies have become cheaper [60, 63–65], more transcriptomic datasets include gene expression measurements at multiple timepoints [53, 55, 66–70]. Still, it often remains challenging to measure transcriptomic profiles at very early or very late stages of a biological process. For instance, senescent and extremely diseased tissue can be challenging to measure, but are of extreme interest for aging and therapeutics. Furthermore, accounting for the combinatorial effects of different experimental conditions grows the measurement space exponentially, and is not practical to measure experimentally.

Two key challenges that we seek to address are *temporal extrapolation* and *combinatorial extrapolation*. In temporal extrapolation, the timepoints of interest are outside the range of times containing experimental measurements. Accurate extrapolation on a single time series is challenging due to non-stationary features and temporal out-of-domain adaptation [71]. In combinatorial extrapolation, while each experimental variable of interest is measured at least once, a combination of variables of interest might not be observed in the dataset. Accurate extrapolation in this setting can be challenging due to non-linear interactions between different experimental conditions, but has significant value for areas like pharmacogenomics, where it is infeasible to test every drug on every cell line [53]. Both extrapolation challenges can be addressed by incorporating multiple time series into the prediction task. For instance, one possible solution for the temporal extrapolation problem is to combine sparse time series measurements from heterogeneous sequences. For example, mouse [55] and roundworm [72] transcriptomic time series measurements, combined with developmental human measurements, can help simulate early-stage embryonic transcriptomic profiles for human [73].

There are two major challenges in effectively utilizing other time series: unaligned measured timepoints and batch effects between experimental conditions (Figure 3.1). In other words, timepoints with measured gene expression differ between species, both over chronological time (e.g., 3 months of age) and biological age (e.g., senescence). Existing methods are unable to simultaneously consider the full sequence of measured timepoints [23, 74] or take into account the temporal batch effects between time series [75–78].

To address these limitations, I proposed Sagittarius [26], a model that maps heterogeneous gene ex-



(a) Temporal extrapolation.

(b) Combinatorial extrapolation.

Figure 3.1: **Challenges for temporal extrapolation and categorical extrapolation in time series datasets.** **a**, Gene expression vector measurements, depicted by circles, are not evenly spaced over time for either the mouse (top) or human (bottom) time series measurements, and the same timepoints are not measured across time series. Furthermore, the same developmental timepoints chronologically for mouse and human correspond to very different stages of organism development. "E" indicates embryonic day; "P" indicates postnatal day; "wpc" indicates weeks-post-conception; and "yrs" indicates years. **b**, Some combinations of categorical environmental variables are unmeasured in a time series dataset. In an example with time series for different species and organ combinations, only a subset of species, organ pairs are measured in the dataset. In categorical extrapolation, we aim to generate time series data for an otherwise unmeasured species and organ combination. However, each species and organ individually must be measured at least once in the dataset (each row (column) is measured in at least one time series). Figure created with `BioRender.com`.

pression time series to a shared reference space based on inferred biological age rather than the observed age (timepoint), enabling multiple sparsely measured time series to jointly inform extrapolation. Sagittarius leverages a transformer-based architecture with multi-head attention [47] to map the heterogeneous measurements from the irregular, unaligned, and sparse time series to a latent reference space shared by all time series, using high-frequency sinusoidal embeddings of the timestamp [78, 79] and experimental condition labels of each time series to define the mapping. After alignment in the shared reference space, Sagittarius can accurately predict new genomic profiles at extrapolated timepoints, as well as predict gene expression for unmeasured combinations of experimental conditions.

We evaluated Sagittarius in three diverse settings in developmental biology, pharmacogenomics, and cancer genomics. On the Evo-devo development dataset [55], Sagittarius accurately extrapolated gene ex-

pression profiles with a 0.983 Pearson correlation, enabling an in-depth analysis of mouse organ differentiation. To evaluate Sagittarius's robustness to extremely sparse measurements, we next applied it to the LINCS pharmacogenomics dataset [53] and found that Sagittarius was able to predict drug repurposing opportunities across drugs and cell lines. Finally, we applied Sagittarius to The Cancer Genome Atlas (TCGA) dataset [56], where Sagittarius was able to accurately extrapolate mutation profiles for patients with a long survival time. Our findings implicated a gene set related to the Hedgehog signaling pathway and *GLI* oncogene that can potentially drive tumorigenesis in sarcoma patients.

3.2 Methods

3.2.1 Overview of Sagittarius

Given a heterogeneous, unaligned, sparse, and irregular genomic time series dataset, Sagittarius is able to extrapolate gene expression profiles for unmeasured timepoints. Fundamentally, we hypothesize that the input time series are related by a common latent trajectory, such as a general developmental trajectory shared by humans and model organisms. The key idea behind Sagittarius is to learn a *shared reference space* that models this trajectory. All observed measurements are then modeled as transformations from some point along this trajectory, where the specific point can be determined by the inferred biological age. We use a transformer-based architecture [78] to map each input measurement to and from the reference space, where the learnable mapping is informed by the experimental conditions associated with the time series. This parameterization addresses both temporal extrapolation and batch effects between experimental conditions. During inference, Sagittarius can extrapolate gene expression profiles for a timepoint and condition of interest.

Illustrative example

Consider the following illustrative example: imagine that we have a dataset of gene expression measurements over time from human, orangutan, gorilla, and chimpanzee. We have aging and senescence measurements for the African great apes, but none for human; our aim is to leverage measurements from the other great apes to simulate senescent human gene expression profiles to better understand human aging.

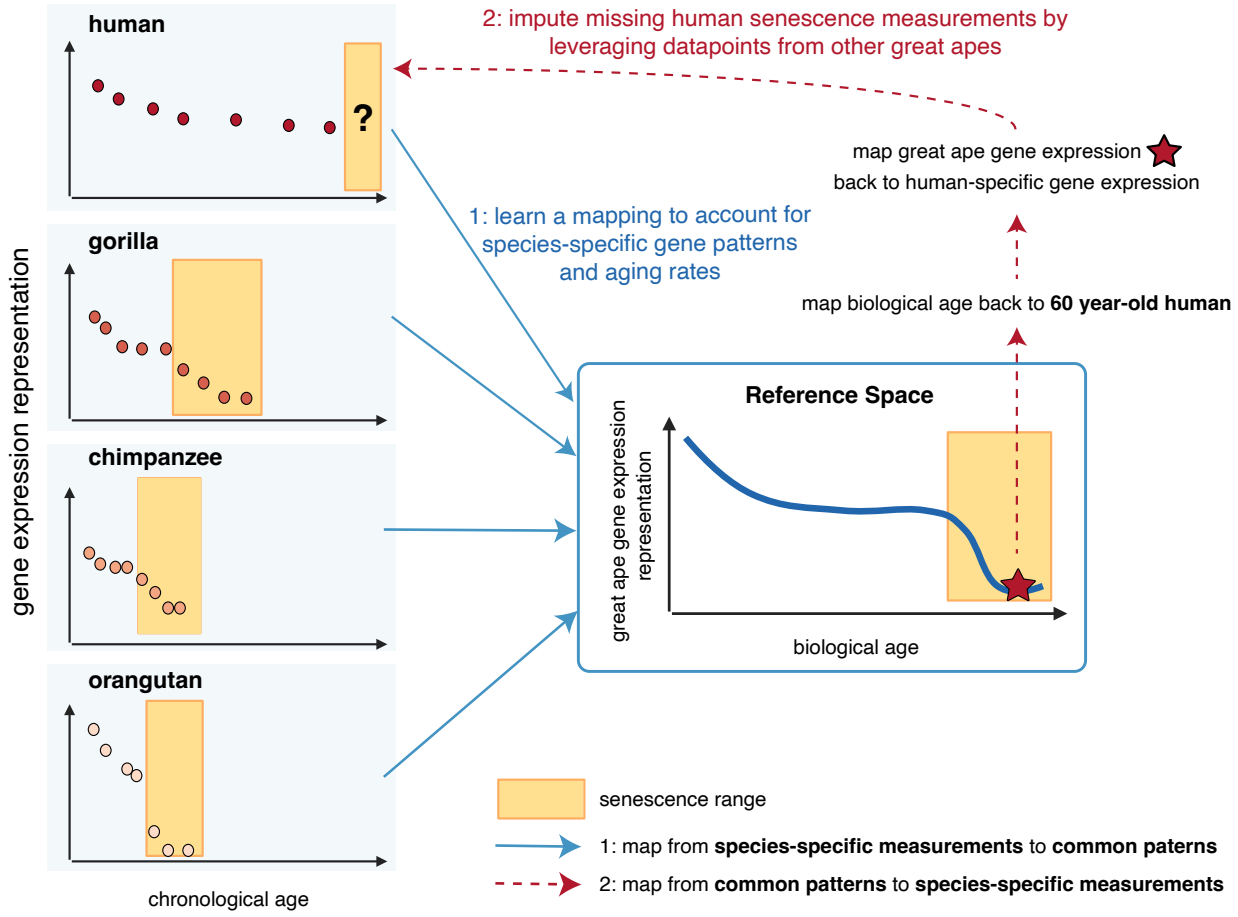


Figure 3.2: **Illustrative example of the Sagittarius reference space.** Given a dataset of great ape gene expression measurements over time, we can learn invertible, species-specific mappings to account for species-specific gene expression patterns and relationship between chronological and biological age. To obtain otherwise unavailable gene expression measurements for human, we can leverage the reference space's representation of great ape senescence constructed from other species' aging datapoints. Finally, we can simulate extrapolated human-specific senescence gene expression measurements using the inverse of the human-to-great-ape gene expression and aging transformations. Figure created with `BioRender.com`.

Conceptually, Sagittarius approaches this problem by learning a latent trajectory of great-ape development and aging (Figure 3.2). This involves a two-stage approach to account for both the natural variation of gene expression across species and the variation in chronological versus biological age between species. To address the former challenge, Sagittarius learns a mapping $f_s(\cdot)$ from a gene expression measurement for species s to a general gene expression measurement representation that is corrected for species-specific biases. After passing all measurements through their respective $f_s(\cdot)$, they are therefore comparable in terms

of gene measurements. To address the latter challenge, Sagittarius also learns a temporal mapping $g_s(t)$ from the chronological timepoint t associated with a measurement from species s to a chronological age, which are then comparable across all species. These two stages map all measurements in the dataset to a comparable hyperplane, which we term the *reference space*.

To extrapolate new measurements for human senescence, we can proceed with the reverse mappings, informed by the reference representation of senescent great ape gene expression measurements. In particular, we can use $g_{\text{human}}(t)$ to find the appropriate biological age for the sample to simulate, and $f^{(-1)}_{\text{human}}$ to reintroduce the human-specific gene expression patterns.

3.2.2 Problem setting

We are given a set of N heterogeneous but related input gene expression time series $\left\{\mathbf{x_i} \in \mathbb{R}^{T_i \times M}\right\}_{i=1}^{N}$, each associated with $C \geq 1$ categorical experimental labels $\left\{\mathbf{y_i} \in \{1, \ldots, C_c\}_{c=1}^{C}\right\}_{i=1}^{N}$ and $B \geq 1$ continuous variables $\left\{\mathbf{t_i} \in \mathbb{R}^{T_i \times B}\right\}_{i=1}^{N}$. The number of measured timepoints T_i can vary for each measured time series. For brevity of notation, we use $\mathbf{x_i}[r]$ to indicate the M-dimensional gene expression measurement at the $r \in \{1, \ldots, T_i\}$th earliest experiment in the measured sequence.

During inference time, we are given a timepoint of interest $\mathbf{t}_\star \in \mathbb{R}^B$ and some environmental conditions of interest $\mathbf{y}_\star$, where $\forall c \exists i \in \{1, \ldots, N\}$ s.t. $\mathbf{y}_{\star,c} = \mathbf{y}_{i,c}$; that is, we must have seen the condition for the cth component of the environment during training. However, we do not require that the complete, multivariate environmental condition $\mathbf{y}_\star$ appears in the training set. The aim is then to simulate the unmeasured gene expression profile(s) $\hat{\mathbf{x}_\star}(\mathbf{y}_\star, \mathbf{t}_\star)$ corresponding to the condition(s) and timepoint(s) of interest.

3.2.3 Sagittarius model architecture

The Sagittarius model is divided into encoder and decoder modules around the shared reference space (Figure 3.3).

Sagittarius encoder

The encoder first embeds each individual gene expression measurement, disentangling the environmental conditions from the embedded representation. Specifically, we learn the encoder $q_\xi(\cdot)$ of a conditional

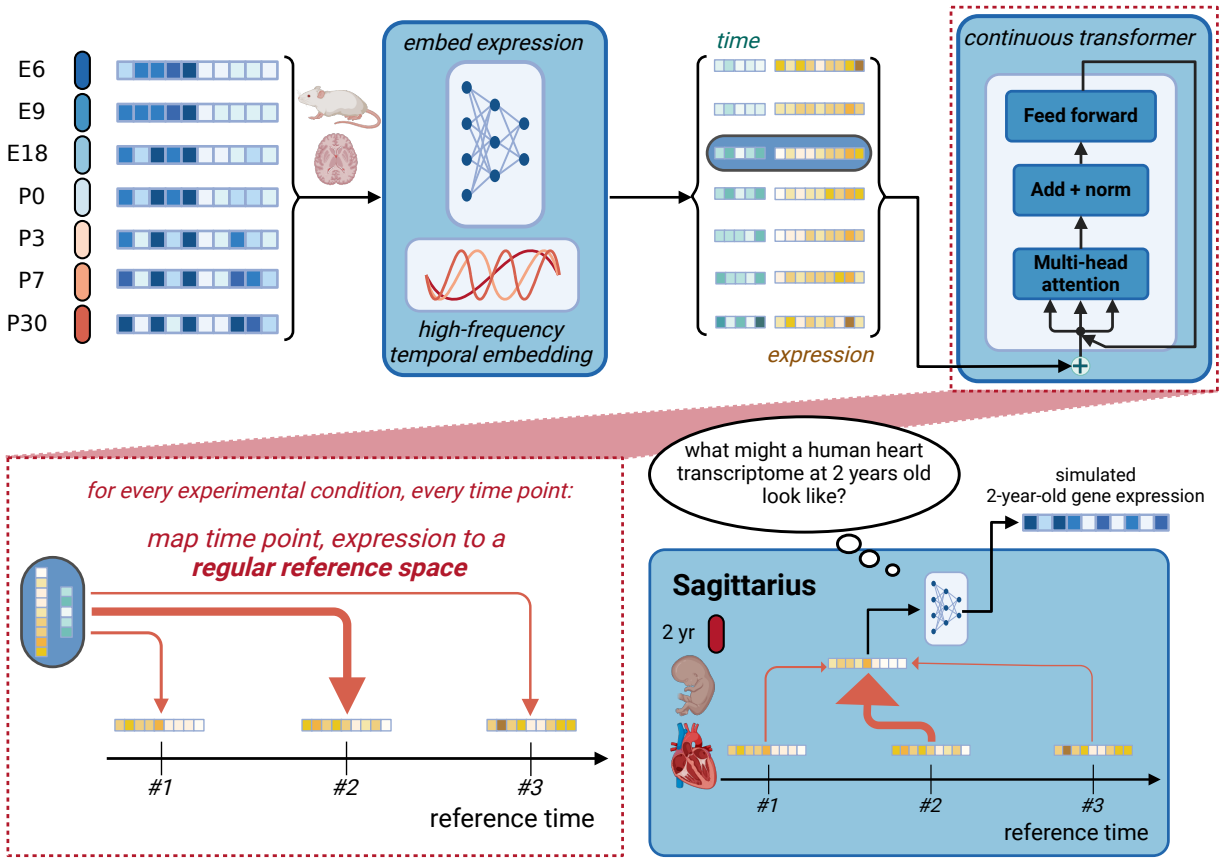


Figure 3.3: **Sagittarius model overview.** For each time series, Sagittarius computes a conditional high-frequency sinusoidal embedding of the measured timepoints and a conditional embedding of the gene expression measurements at each timepoint based on the species and organ. It then uses a continuous, multi-head attention transformer to map the embedded timepoints and expression vectors to the reference space. The continuous transformer takes each pair of species- and organ-conditioned time and expression embeddings and learns a mapping to the regular reference space, translating from measured age to a shared biological age. Users can request extrapolated expression vectors from Sagittarius, such as the expression profile of a human 2-year-old heart that may not be measured in the original dataset. Sagittarius maps the request from the regular reference space back to the data space to predict the unmeasured profile. "E" indicated embryonic days and "P" indicates postnatal day. Created with `BioRender.com`.

variational autoencoder (cVAE) [23] such that

$$\mu_i[r], \sigma_i[r] = q_\xi(\mathbf{x_i}[r], \mathbf{y_i}) \in \mathbb{R}^{2\times d}$$
$$\mathbf{z_i}[r] \sim \mathcal{N}(\mu_i[r], \sigma_i[r]) \in \mathbb{R}^d, \tag{3.1}$$

where $d \ll M$ is the latent dimension and $\mathcal{N}(\mu, \sigma)$ indicates a Gaussian distribution with mean μ and standard deviation σ.

The encoder also uses the measurement timepoints and environmental variables to impute a comparable reference timepoint (for example, the biological age) of the sample. We first use a high-frequency sine wave that maps each timepoint to the range $[0, 1]$, mitigating potential out-of-domain challenges with increasingly large scalar inputs during temporal extrapolation, and helps Sagittarius learn high-frequency patterns in the data, which neural networks have been shown to learn more slowly than low-frequency patterns [78, 79]. We then learn the full sample timepoint embedding by incorporating the environmental variables, following

$$\begin{aligned}
\theta_{\mathbf{i}} &= \bigoplus_{h=1}^{H} \bigoplus_{r=1}^{T_i} \sin(\omega_h \mathbf{t_i}[r] + \alpha_h) \in \mathbb{R}^{H \times T_i \times \sum_{b=1}^{B} d_{temp}^{(b)}} \\
\zeta_{\mathbf{i}} &= \bigoplus_{c=1}^{C} f_{\nu}^{(c)}(\mathbf{y_i}^{(c)}) \in \mathbb{R}^{\sum_{c=1}^{C} d_{ytr}^{(c)}} \\
\mathbf{k_i}^{(enc)} &= \bigoplus_{h=1}^{H} \bigoplus_{r=1}^{T_i} \mathbf{W}^{(enc)}(\theta_{\mathbf{i}}[h, r] \oplus \zeta_{\mathbf{i}}) \in \mathbb{R}^{H \times T_i \times (\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)})},
\end{aligned} \tag{3.2}$$

where $\bigoplus$ indicates vector concatenation, H is the number of attention heads in the model, $d_{temp}^{(b)}$ is the dimension of the sine wave temporal embedding for the bth continuous variable, $f_{\nu}^{(c)}(\cdot)$ is a learnable embedding of dimension $d_{ytr}^{(c)}$ for the cth environmental variable, and $\mathbf{W}^{(enc)}$ is a learnable weight matrix of dimension $(\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)})^2$.

Given the model hyperparameter S such that $S + 1 < T_i \; \forall i$, defining the number of "reference timepoints" to parameterize the shared latent trajectory in the reference space, and a temporal basis range of interest $\left(\gamma_b^{(0)}, \gamma_b^{(1)}\right)$ for each of the $b \in \{1, \ldots, B\}$ continuous variables, Sagittarius defines the fixed temporal reference grid

$$\begin{aligned}
\mathbf{t_{ref}} &= \bigoplus_{b=1}^{B} \bigoplus_{s=0}^{S} \gamma_b^{(0)} + s\frac{(\gamma_b^{(1)} - \gamma_b^{(0)})}{S} \in \mathbb{R}^{B \times (S+1)} \\
\phi_{\mathbf{ref}} &= \bigoplus_{h=1}^{H} \bigoplus_{s=1}^{S+1} \sin(\omega_h \mathbf{t_{ref}}[s] + \alpha_h) \in \mathbb{R}^{H \times (S+1) \times \sum_{b=1}^{B} d_{temp}^{(b)}} \\
\mathbf{q}^{(enc)} &= \bigoplus_{h=1}^{H} \bigoplus_{s=1}^{S+1} \mathbf{U}^{(enc)} \phi_{\mathbf{ref}}[h, s],
\end{aligned} \tag{3.3}$$

where $\mathbf{t_{ref}}$ is a model hyperparameter and $\mathbf{U}^{(enc)}$ is a learnable weight matrix of dimension $(\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)}) \times \sum_{b=1}^{B} d_{temp}^{(b)}$.

Finally, we relate the learned sample timepoint embedding to the learned reference timepoint embeddings using a continuous transformer [47, 78], where the keys and queries of the transformer are the temporal embeddings and the values are the associated gene expression measurement embeddings. Specifically, we use a transformer encoder to learn reference space gene expression embeddings as

$$\mathbf{z_{ref}} = \bigoplus_{s=1}^{S+1} \left(\sum_{h=1}^{H} \sum_{r=1}^{T_i} \text{softmax}\Big(\frac{\mathbf{q}^{(enc)}[h,s](\mathbf{k_i}^{(enc)}[h,r])^\top}{\sqrt{\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)}}} \Big) \mathbf{z_i}[r] \right) \in \mathbb{R}^{(S+1)\times d}. \tag{3.4}$$

The encoder therefore yields $S+1$ reference pairs $\mathbf{t_{ref}}[s], \mathbf{z_{ref}}[s]$ that parameterize our understanding of the shared patterns in the time series dataset.

Sagittarius decoder

Sagittarius's decoder is structured analogously. Given a series of timepoints $\mathbf{t}_\star$ and environmental conditions $\mathbf{y}_\star$ of interest, Sagittarius's decoder uses the learned reference representation to simulate the time series $\hat{\mathbf{x}_\star}$. As a first step, the decoder component of the continuous transformer [47, 78] to map reference timepoint and gene expression embedding pairs back to values contextualized by $\mathbf{y}_\star$. In particular, we take

$$\begin{aligned} \theta_{\mathbf{ref}} &= \bigoplus_{h=1}^{H} \bigoplus_{s=1}^{S+1} \sin(\omega_h \mathbf{t_{ref}}[s] + \alpha_h) \in \mathbb{R}^{H\times(S+1)\times\sum_{b=1}^{B} d_{temp}^{(b)}} \\ \mathbf{k}^{(dec)} &= \bigoplus_{h=1}^{H} \bigoplus_{s=1}^{S+1} \mathbf{W}^{(dec)} \theta_{\mathbf{ref}}[h,s], \end{aligned} \tag{3.5}$$

where $\mathbf{W}^{(dec)}$ is a learnable weight matrix of dimension $(\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)}) \times \sum_{b=1}^{B} d_{temp}^{(b)}$, and ω_h and α_h are shared with the encoder. We also compute the temporal embeddings for the timepoint of interest

as

$$\phi_\star = \bigoplus_{h=1}^{H} \bigoplus_{r=1}^{T_\star} \sin(\omega_h \mathbf{t}_\star[r] + \alpha_h) \in \mathbb{R}^{H \times T_\star \times \sum_{b=1}^{B} d_{temp}^{(b)}}$$
$$\xi_\star = \bigoplus_{c=1}^{C} f_\upsilon^{(c)}(\mathbf{y}_\star{}^{(c)}) \in \mathbb{R}^{\sum_{c=1}^{C} d_{ytr}^{(c)}}$$
$$\mathbf{q}_\star{}^{(enc)} = \bigoplus_{h=1}^{H} \bigoplus_{r=1}^{T_\star} \mathbf{U}^{(dec)}(\phi_\star[h,r] \oplus \xi_\star) \in \mathbb{R}^{H \times T_\star \times (\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)})}, \tag{3.6}$$

for a learnable embedding $f_\upsilon(\cdot)$ and learnable weights $\mathbf{U}^{(dec)} \in \mathbb{R}^{(\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)})^2}$. We then leverage the continuous transformer framework to produce gene expression embeddings aligned at the timepoints of interest as

$$\mathbf{z}_\star = \bigoplus_{r=1}^{T_\star} \left(\sum_{h=1}^{H} \sum_{s=1}^{S+1} \text{softmax}\Big(\frac{\mathbf{q}_\star{}^{(dec)}[h,r](\mathbf{k}^{(dec)}[h,s])^\top}{\sqrt{\sum_{b=1}^{B} d_{temp}^{(b)} + \sum_{c=1}^{C} d_{ytr}^{(c)}}} \Big) \mathbf{z}_{\mathbf{ref}}[s] \right) \in \mathbb{R}^{T_\star \times d}. \tag{3.7}$$

Finally, we use the decoder structure of a cVAE [23] to simulate the full gene expression vectors as

$$\hat{\mathbf{x}_\star} = \bigoplus_{r=1}^{T_\star} p_\theta(\mathbf{z}_\star[r], \mathbf{y}_\star) \in \mathbb{R}^{T_\star \times M}. \tag{3.8}$$

3.2.4 Sagittarius objective

Given a gene expression time series data distribution $\mathcal{D}$, we consider source time series i and target time series j and use the evidence lower bound on the log-likelihood of our data to write the loss

$$\mathcal{L}\Big(\xi, \nu, \mathbf{W}^{(enc)}, \mathbf{U}^{(enc)}, \upsilon, \mathbf{W}^{(dec)}, \mathbf{U}^{(dec)}, \theta \Big| \mathcal{D}\Big) = \tag{3.9}$$
$$\mathbb{E}_{(\mathbf{x_j}, \mathbf{y_j}, \mathbf{t_j}) \sim \mathcal{D}} \Big[\mathbb{E}_{(\mathbf{x_i}, \mathbf{y_i}, \mathbf{t_i}) \sim \mathcal{D}} \Big[\mathbb{E}_{q_\xi(\mathbf{z_i}|\mathbf{x_i}, \mathbf{y_i})} \big[\log p_\theta(\mathbf{x_j}|\mathbf{z_i}, \mathbf{y_i}, \mathbf{t_i}, \mathbf{y_j}, \mathbf{t_j}) - \beta D_{KL}\big(q_\xi(\mathbf{z_i}|\mathbf{x_i}, \mathbf{y_i}) \parallel p(\mathbf{z})\big) \big] \Big] \Big],$$

where $p(\mathbf{z})$ is the standard normal d-dimensional distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to regularize the variational space and $p_\theta(\cdot)$ is written with dependence on the source sequence environmental and temporal variables as well as the target environment and timepoints to reflect the mapping to and from the reference space.

We train Sagittarius end-to-end with the Adam optimizer [80], using a batch size of 8 time series for the

Evo-devo [55] experiments, 16 time series for the LINCS [53] experiments, and 2 time series for the TCGA [56] experiments.

Importantly, Sagittarius's conceptual reference space requires that the common latent trajectory for all time series in the dataset is truly shared. To learn this, we include a series of objectives during training, controlling the relationship between source time series i and target time series j. Fundamentally, we combine a *reconstruction* objective with a *generation* objective. For the reconstruction objective, we take $\forall i \in \{1, \ldots, N\}, j = i$, teaching the model to encode and decode each time series in the training dataset. However, we also include the generation objective, which varies slightly by experimental setting.

Evo-devo generation objective

We randomly select four batches of 12 time series from the training dataset to treat as the source sequence, producing $N_{gen} = 48$ generation examples. One batch is then selected for each of the following tasks:

- *Temporal generation:* Mask three observations from each time series in the batch; treat those three observations as the generation target;
- *Cross-organ generation:* For each time series in the batch, randomly select a second training time series from the same species but a different organ; treat this second time series as the generation target;
- *Cross-species generation:* For each time series in the batch, randomly select a second training time series from the same organ but a different species; treat this second time series as the generation target;
- *Random generation:* For each time series in the batch, randomly select another training time series to use as the generation target.

LINCS generation objective

We consider the following three generation approaches, resulting in $N_{gen} = 80$ generative examples:

- *Cross-cell-line generation:* Randomly select 32 distinct drugs from the training data. For each drug, randomly select two time series for treatment combinations using that drug; treat one time series as the source and the other as the target;

- *Cross-drug generation:* Randomly select 32 distinct cell lines from the training data. For each cell line, randomly select two time series for treatment combinations using that cell line; treat one time series as the source and the other as the target;

- *Random generation:* Randomly select 16 pairs of treatment combination time series from the training data. For each pair, treat one time series as the source and the other as the target.

TCGA generation objective

We randomly select two batches of 12 time series from the training data to treat as the source sequence, resulting in $N_{gen} = 24$ generative examples. One batch is then selected for each of the following tasks:

- *Temporal generation:* Mask three observations from each time series in the batch; treat those three observations as the generation target;

- *Random generation:* For each time series in the batch, randomly select a second training time series (second cancer type) to use as the generation target.

3.2.5 Inference

As Sagittarius's encoder and decoder map around a conceptual reference space that is shared by all of the environmental conditions in the dataset, during inference for a possibly unseen combination of environmental variables $\mathbf{y}_\star$ we can choose any time series in the training dataset as the source time series $(\mathbf{x_i}, \mathbf{y_i}, \mathbf{t_i})$. For simplicity, we select the time series that we judge to be "closest" to the target time series (i.e., sharing the most individual environmental variables, and breaking ties randomly).

3.2.6 Dataset preprocessing

Evo-devo

The Evo-devo dataset [55] contains gene expression vectors for 7 species and 7 organs measured at multiple pre- and post-natal timepoints. We associated each measurement with the environmental variables $\mathbf{y}_i = [\text{species}_i, \text{organ}_i]$. The sole continuous variable in the dataset is organism age, so $B = 1$ in the problem setting formulation. We restricted the dataset to orthologous genes in all 7 species, identified with the python

`pybiomart` package [81] and the provided Ensembl [82] gene IDs, resulting in 5037 genes. The measured timepoints for each species were given as strings with different units by species. As a preprocessing step, we computed each timepoint's rank within the ordered timepoints for that species to use as the timepoint label. We then randomly selected the rabbit heart time series and used the Augmented Dickey-Fuller (ADF) test, which tests for stationary, and only retained genes with p-value > 0.05, resulting in $M = 4533$ retained genes.

LINCS

We used the LINCS L1000 Platform level 3 pharmacogenomic dataset [53]. We restricted the data experiments to dose measurements in micromolars (uM) that did not exceed 20 uM. We then further restricted the dataset to only include drug and cell line treatment combinations that had more than 15 measurements in the restricted dataset, resulting in 2687 retained combinations that each had between 16 and 78 measurements. We interpreted these as time series in two continuous variables ($B = 2$), with $\mathbf{y_i} = [\text{drug}_i, \text{cell\_line}_i]$ and $\mathbf{t_i} = [\text{dose}_i, \text{treatment\_time}_i]$. Each gene measurement measures $M = 978$ genes.

TCGA

We used the TCGA Firehose legacy dataset [56], independently considering the somatic mutation and RNA-seq datasets. Both datasets measure $20{,}501$ genes: we restricted the dataset to the $M = 1000$ most-frequently-mutated genes for the mutation experiments and the $M = 1000$ most-highly-variable genes for the gene expression experiments, jointly considering all cancer types. We removed all patients with missing event times, and excluded mutation patients with no mutations in the 1000 remaining genes. Finally, we excluded cancer types with fewer than 12 remaining patients. We finally constructed a time series for each cancer type $\mathbf{y_i}$, where the rth patient in the time series had event time $\mathbf{t_i}[r]$ ($B = 1$) and mutation or gene expression profile $\mathbf{x_i}[r]$.

Although our TCGA time series formulation enables extrapolation to cancer patients with good prognosis, *censored patients* have an event time that indicates the study's final contact with the patient rather than a time of death, thereby representing a lower-bound on survival time. Consequently, censored patients may have an event time that differs dramatically from their actual survival time, confounding the temporal

ordering of patients. For the TCGA gene expression experiments, we excluded all censored patients. For the mutation experiments, we leveraged recent techniques from Learning with Noisy Labels [83, 84] to identify censored patients who likely had a death event shortly after the reported event time while excluding patients who were more likely to survive long after losing contact with the study.

We constructed a neural network $f^{(y)}$ for each cancer type $\mathbf{y}$ to predict $\hat{\mathbf{t}[r]} = f^{(y)}(\mathbf{x}[r])$. Given the rth patient's binary censoring label $c[r]$, where $c[r] = 0$ indicates that the rth patient had a censored death event, we defined the per-patient loss as

$$\mathcal{L}_{individual}(\mathbf{x}[r], \mathbf{t}[r], c[r]) = \mathbb{1}\big[c[r] = 1\big]\big|\mathbf{t}[r] - f^{(y)}(\mathbf{x}[r])\big|_1 + \mathbb{1}\big[c[r] = 0\big] \max\big(\mathbf{t}[r] - f^{(y)}(\mathbf{x}[r]), 0\big) \tag{3.10}$$

where $\mathbb{1}[\cdot]$ is the indicator function, thereby not penalizing the model for overestimating the survival time of a censored patient. We trained $f^{(y)}$ with the sum of all individual patients' losses and a regularization term. We used the stochastic gradient descent (SGD) optimizer with learning rate 0.1, a single hidden layer with 32 neurons and regularization penalty weight of 0.3 for the L2 norm of the weights. We trained the model for 2500 epochs and selected the best model from the epoch with maximal concordance index among the observed patients. We then computed the absolute error $\big|f^{(y)}(\mathbf{x}[r]) - \mathbf{t}[r]\big|_1$ for each patient and fit two beta distributions β_{obs} and β_{cens} to the absolute errors for the observed and censored patient error observations respectively using the `scipy` [85] python package. Finally, for each censored patient, if the absolute error associated with their event time was more likely to be generated by β_{obs} than β_{cens}, or if the absolute error was smaller than the error associated with one or more observed patients, we retained that patient; otherwise, we excluded the patient from further analysis.

3.2.7 Evaluation metrics

We consider four main metrics to evaluate over N_{test} predicted time series of gene expression measurements $\hat{\mathbf{x}_\mathbf{j}} \in \mathbb{R}^{T_j \times M}$ given the true experimental measurements $\mathbf{x_j} \in \mathbb{R}^{T_j \times M}$ for test measurements at times indexed $r = 1, \ldots, T_j$. We define model root mean squared error (RMSE) as the average RMSE of each

measurement in the time series, or

$$RMSE = \frac{1}{N_{test}} \frac{1}{\sum_{j=1}^{N_{test}} T_j} |\mathbf{x_j} - \hat{\mathbf{x}}_\mathbf{j}|_2. \tag{3.11}$$

We also define a model's correlation comparing genes (within the same timepoint) as the average of each time series' correlations

$$\rho^{(genes)} = \frac{1}{N_{test}} \sum_{j=1}^{N_{test}} \frac{1}{T_j} \sum_{r=1}^{T_j} \rho(\mathbf{x_j}[r], \hat{\mathbf{x}}_\mathbf{j}[r]), \tag{3.12}$$

where $\rho(\cdot, \cdot)$ can be either the Pearson or Spearman correlation coefficient computation. Analogously, we define a model's correlation comparing timepoints (within the same gene) as

$$\rho^{(times)} = \frac{1}{N_{test}} \sum_{j=1}^{N_{test}} \frac{1}{M} \sum_{m=1}^{M} \rho\big(\bigoplus_{r=1}^{T_j} \mathbf{x_j}[r, m], \bigoplus_{r=1}^{T_j} \hat{\mathbf{x}}_\mathbf{j}[r, m]\big). \tag{3.13}$$

Finally, we evaluate the mutation extrapolation experiments using the area under the receiving operator characteristic (AUROC). For this evaluation, we restricted the evaluation genes to those with Augmented Dickey-Fuller (ADF) test p-value > 0.05 in at least δ cancer types, denoted $\gamma(\delta)$, empirically setting $\delta_{THCA} = 2$ and $\delta_{SARC} = 4$ to have a sufficiently large test set size. Limiting our analysis to test sets with at least one mutation in the remaining gene set resulting in 9 usable test splits for thyroid carcinoma (THCA) and 61 usable test splits for sarcoma (SARC) respectively. We then define the per-cancer-type AUROC as

$$\begin{aligned} AUROC_j = & \frac{1}{|\{r : c_j[r] = 1 \forall j \in \{1, \ldots, T_j\}\}|} \\ & \sum_{r=1}^{T_j} \mathbb{1}[c_j[r] = 1] \mathbb{1}[\mathbf{x_j}[r, \gamma(\delta)] \neq \mathbf{0}] AUROC(\mathbf{x_j}[r, \gamma(\delta)], \hat{\mathbf{x}}_\mathbf{j}[r, \gamma(\delta)]) \end{aligned} \tag{3.14}$$

for cancer type j with T_j test patients, where $\mathbf{x}[r, \gamma(\delta)]$ indicates the somatic mutation profile across the genes in gene set $\gamma(\delta)$ for the rth test patient, $\mathbb{1}[\cdot]$ is the indicator function (thereby excluding any censored patients or patients with no mutations in the gene set $\gamma(\delta)$), and $AUROC(\cdot, \cdot)$ is the standard AUROC computation between two vectors.

To assess whether Sagittarius statistically outperforms the comparison approaches, we used the one-sided paired t-test between Sagittarius's performance metrics and the best comparison approach's perfor-

mance per time series. For $\rho^{(genes)}$ and $\rho^{(times)}$, we first computed the Fisher z-transformation [86] of the correlation values, defined as

$$z(r) = \frac{1}{2} \ln \left(\frac{1+r}{1-r} \right) \tag{3.15}$$

for some correlation coefficient r to normalize the t-test inputs.

3.2.8 Comparison approaches

We consider both classical and deep learning comparison approaches (more information in Appendix Section A.1.2). In particular, we benchmark temporal extrapolation performance with a single continuous variable using standard *mean* and *linear* baselines, as well as a bidirectional neural ordinary differential equation (ODE) model [76], a bidirectional recurrent neural network (RNN) [75], a multi-time attention network (mTAN) [78], a conditional variational autoencoder (cVAE) [23], and a compositional perturbation autoencoder (CPA) [74]. As cVAE is the only model that can be applied to multiple continuous variables out-of-the-box, we also use this model for comparison in experiments with $B > 1$. All models are implemented in pytorch [87] and trained with the Adam optimizer [80]. We additionally use to python `torchdiffeq` package [76, 77] to implement the neural ODE models. We also use the public Github repositories for the official versions of CPA[3] and mTAN[4].

3.2.9 Evo-devo developmental transcriptomic analysis

To study the dynamics of the transcriptome at extrapolated time ranges, we used Sagittarius to simulate 10 random profiles $\hat{\mathbf{x}}(\mathbf{y}, \mathbf{t})$, where $\mathbf{t} = [t_0, t_0 + \Delta, t_0 + 2\Delta, \ldots, t_1 - \Delta, t_1]$ for some user-specified parameters t_0, Δ, t_1. We further smoothed these trajectories by computing the moving average with a window size of 10 (averaging over the 5 measurements before and after t).

Evo-devo transcriptomic velocity analysis

We computed the UMAP [88] embeddings of the smoothed trajectory, and optionally computed the gene expression velocity in the UMAP space as the vector between the predicted measurements at time $t + \Delta$ and

[3]Github: `https://github.com/facebookresearch/CPA`.
[4]Github: `https://github.com/reml-lab/mTAN`.

time t in the embedded space for all trajectories and all t. We then smoothed the velocity vectors by taking the average of the velocity at time t and time $t - \Delta$. We then took the normalized average (mean) over the smoothed velocities associated with each of the 10 simulated trajectories for a given organ. To decrease clutter in the resulting plots, we took the resulting velocity at integer timepoints. We then projected all gene expression measurements to a grid in the UMAP space, and defined the velocity at each point as the weighted average of the 100 velocity vectors nearest to that grid point using the `NearestNeighbors` module from `sklearn.neighbors` [89]. Finally, we discarded the 5% of velocities with smallest magnitude to simplify the resulting visualizations.

Evo-devo organ development genes analysis

To identify genes that had a very similar expression at early developmental stages but differing expression levels at later developmental stages, suggesting an organ-specific role, we considered the first and last 25 timepoints from each of the smoothed organ trajectories corresponding to early development and later development, resulting in a total of 250 timepoints each for the early and later development time ranges across all of the samples. We used the ANOVA test with Bonferroni multiple hypothesis testing correction to compare the gene expression values for each gene across organs in the early development time ranges and again to compare expression of each gene across organs in the later development time ranges.

Evo-devo aging gene evaluation analysis

To evaluate whether Sagittarius could accurately predict gene expression patterns during mouse aging, we computed the Pearson correlation over time for each of the genes in the extrapolated aging mouse time series. As a comparison benchmark, we also computed the Pearson correlation over time for the mouse organ time series measurements in the Evo-devo dataset [55], which end at postnatal day 63 (P63). Finally, we used the heart and aorta, kidney, and liver tissue data from the single-cell *Tabula Muris Senis* droplet dataset [67], which were the three tissues that aligned with the Evo-devo organs. We computed the average expression at each timepoint for each cell type in the tissue data, and then took the Pearson correlation of the average cell type expression over time. We compare the correlation over time $\rho^{(times)}$ for Sagittarius's extrapolated data and the measured Evo-devo data to the distribution of cell type correlation in the *Tabula*

Muris Senis dataset.

3.2.10 LINCS drug dosage similarity network analysis

After training Sagittarius on the complete LINCS dataset we randomly selected 78 distinct doses from the dataset, which ranged from $8.33\text{e}{-5}$ to 19.9998, and selected a treatment time of 6 hours. For each drug and cell line treatment in the dataset, we used Sagittarius to predict the drug-induced expression profile for the treatment at each of the 78 doses with a 6-hour treatment time, using the actual treatment measurements in the dataset as the source sequence. To remove the strong cell-type-specific signal in the profiles, we subtracted Sagittarius's predicted basal cell line expression from the drug-induced expression vectors.

We took the average over all 78 doses of the differential expression vectors to produce a single 978-dimensional vector for each of the 2687 treatment combinations. We then computed a normalized similarity matrix Σ as

$$\sigma_{i,j} = 1 - |\mathbf{x_i}^{\Delta} - \mathbf{x_j}^{\Delta}| \qquad\qquad \Sigma_{i,j} = \min_{i',j'} \big(\max_{i'',j''} \big(\frac{\sigma_{i,j} - \sigma_{i',j'}}{\sigma_{i'',j''} - \sigma_{i',j'}} \big) \big) \tag{3.16}$$

where $\mathbf{x_n}^{\Delta}$ indicates the differential expression of treatment combination n. We then constructed an average differential expression k-nearest-neighbors (KNN) network G_{KNN}, beginning from a fully connected graph with edge weights Σ, by first removing all edges with $\sigma_{i,j} < 0.95$, then removing the degree(i)-50 edges with lowest weight for each node i, removing all nodes with degree less than 30, and finally reducing the remaining graph to its largest-connected subgraph. We ran Louvain community detection [90] from the Python community package [91] (`python-louvain`) to identify communities in G_{KNN}. To have a reasonable population of labeled samples in each community and simplify the analysis, we combined neighboring communities until 4 remained, and then calculated the community IC_{50} by averaging the individual treatment IC_{50} over all treatments in the community that were also measured in the Genomics of Drug Sensitivity in Cancer (GDSC) [92] dataset. We visualized G_{KNN} using the edge-weight spring embedded layout in Cytoscape [93], with minimum, maximum, and default edge weights of 0, 1, and 0.5 respectively. We ran 200 average iterations for each node. The spring strength parameter was set to 15, spring rest length to 45, the disconnected spring strength to 0.05, and the disconnected spring rest length to 2000. We did not add any spring strength to avoid collisions, and used 2 layout passes. Finally, we randomized the graph before

computing the layout.

3.2.11 LINCS drug sensitivity analysis

To evaluate Sagittarius's ability to predict drug sensitivity, we designed the following two experimental datasets. To simulate the *Sagittarius-augmented* dataset, we again randomly selected 78 doses in the LINCS dataset [53] and fixed the treatment time as 6 hours. We then used Sagittarius's learned weights to compute the transformer encoder's average key representation over the doses for a given drug and cell line combination $\frac{1}{78}\sum_{r=1}^{78}\mathbf{k_i}^{(enc)}(\mathbf{y}_\star, \mathbf{t}_\star[r])$, corresponding to the average treatment efficacy relative to the reference space. We do this for all drug and cell line treatment combinations in an external drug sensitivity dataset [92, 94] provided that the drug and cell line each appear at least once in the processed LINCS data. We fix the treatment time while varying dose in order to best capture the impact of dose on the treatment response, as half-maximal inhibitory concentration (IC_{50}) is based on drug dose-response curves [92, 94, 95]. We compare these data from Sagittarius to a LINCS-based dataset that uses the average drug-induced gene expression profiles that appear in LINCS for treatment combinations that also appear in the external dataset. For GDSC [92] sensitivity data, this results in 271 Sagittarius-GDSC datapoints and 151 LINCS-GDSC datapoints; for Cancer Therapeutic Response Portal (CTRP) [94] sensitivity data, this results in 2929 Sagittarius-CTRP datapoints and 625 LINCS-CTRP datapoints.

To evaluate the quality of the Sagittarius and LINCS-based IC_{50} prediction models, we conducted a 3-fold cross-cross validation where the test set made up $\frac{2}{3}$ of the data in each fold, and used 10% of the available training data as the validation set. We determined the test fold using the LINCS-based dataset's treatment combinations; all other combinations were used for training. Notably, the Sagittarius-augmented dataset therefore contains additional samples, which were simulated with combinatorial extrapolation.

We computed the average Spearman correlation between a downstream model's IC_{50} predictions and the dataset labels (either from GDSC or CTRP) for each test cell line. To compare overall test performance, we restricted our analysis to cell lines where at least one of the models had significant correlation (Spearman rank-order p-value < 0.05). We used the Spearman correlation between all predicted and measured validation data to quantify validation set performance, which we used for model hyperparameter selection. We considered both Support Vector Regression (SVR) and MLP-based regressors; for SVR regression con-

figurations, we considered linear, polynomial, and radial-basis-function (RBF) kernels; for MLP regression configurations, we considered a regularization weight $\alpha \in [1e-4, 1e-2, 1, 10]$. All other hyperparameters maintained the defaults in `sklearn` [89]. For the model trained on the LINCS-based dataset, the best-performing configuration on the validation data used an SVR regressor with an RBF kernel on the GDSC dataset and polynomial kernel on the CTRP dataset. For the model trained on the Sagittarius-augmented dataset, the MLP model with $\alpha = 10$ was selected for the GDSC dataset and $\alpha = 0.01$ was selected for the CTRP dataset.

3.2.12 LINCS cancer gene essentiality analysis

We used the DEMETER [96] and CERES [97] versions of the DepMap dataset, which quantify gene essentiality for a cell line y_{cell} and gene g via short hairpin RNAs and CRISPR-Cas9 essentiality screens respectively. For each gene and cell line combination in DepMap, we searched for a drug in the LINCS dataset that listed the given gene as its target, hypothesizing that the drug's inhibitory effect on a cell line is related to the cell line's dependency on the target gene [98]. As in the drug sensitivity analyses, we constructed a dataset from Sagittarius using the transformer's average key representation from 78 randomly selected doses and a treatment combination in the given cell line with a drug targeting the gene of interest for each DepMap essentiality pair. This resulted in 4216 and 1666 datapoints from Sagittarius for the DEMETER and CERES versions respectively. We analogously constructed LINCS-DepMap dataset used the average drug-induced expression across doses for DepMap pairs that matched available LINCS experiments, resulting in 765 and 353 datapoints for the two versions.

To evaluate the quality of the Sagittarius- and LINCS-based datasets, we computed the Spearman correlation between the gene essentiality scores measured in the DepMap dataset and those predicted by a regressor trained on each of the datasets. In particular, we trained a 2-layer MLP regressor with 200- and 100-hidden nodes respectively, ReLU activation functions, MSE loss, and the Adam optimizer [80] with a learning rate of $1\text{e}{-3}$. We used 5-fold cross-validation, where 20% of the LINCS-DepMap dataset was used as the test set, and we aligned the Sagittarius-augmented dataset's test set to match the LINCS-DepMap test set. We further held out 10% of the resulting training set for each of the 5 splits to use as a validation set for early stopping during model training.

3.2.13 TCGA-based early-stage cancer patient mutation profile analysis

To simulate the early-stage sarcoma patient mutation profiles, we trained Sagittarius on all available TCGA mutation data and then predicted mutation probability profiles at 27 survival timepoints, ranging from 203-283 months. Specifically, we selected the longest 27 survival times that appeared somewhere in the original TCGA mutation profile dataset, with

$$\begin{aligned} t \in \{&203.12, 204.01, 260.70, 208.23, 209.43, 210.51, 210.81, 211.01, 211.73, 212.09, 216.59, \\ &216.75, 225.43, 229.04, 230.72, 232.00, 232.62, 233.44, 234.10, 238.11, 244.32, \\ &244.91, 255.49, 263.07, 275.66, 281.08, 282.69\} \end{aligned} \tag{3.17}$$

months (Figure A.14). We then averaged the mutation profile predictions of the 27 timepoints and identified the 10 genes the model predicted as most likely to be mutated.

3.3 Results

3.3.1 Extrapolating gene expression to unmeasured timepoints

To assess the merit of our approach, we evaluated whether Sagittarius can extrapolate profiles for gene expression time series from multiple experimental conditions. We used the Evo-devo time series data [55], which contains bulk RNA-seq data from 7 species and 7 organs, where each time series ranges between 9 and 23 distinct measured timepoints that are not biologically aligned across species. Importantly, the developmental ranges measured by each species differ: primates include senescence measurements, while rhesus macaque and chicken do not contain early embryonic data. Therefore, the Evo-devo dataset can be used to assess whether Sagittarius can handle unaligned timepoints and differing biological ages measured across species.

To initially validate our model, we hid the last four measured timepoints from each species' organ time series to use as a test set. After training on the remaining Evo-devo data, we predicted the gene expression vectors for each species and organ combination at the four hidden timepoints and compared them to the held-out expression vectors. To benchmark Sagittarius's performance, we also evaluated the deep learning

methods Conditional Variational Autoencoder (cVAE) [23], Compositional Perturbation Autoencoder (CPA) [74], Multi-Time Attention Network27 (mTAN) [78], Neural ODE [76, 77], and Recurrent Neural Network (RNN) [75], as well as classical mean and linear methods. Overall, Sagittarius achieved the best average performance between the extrapolated and measured gene expression profiles in terms of Pearson correlation comparing genes within a measurement ($\rho^{(genes)} = 0.983$), Pearson correlation comparing timepoints for a single gene ($\rho^{(times)} = 0.458$), and root mean squared error (RMSE $= 0.087$), compared to 0.926, 0.142, and 0.163 respectively for the best-performing comparison approach, and this improvement was robust to many hyperparameter settings (Section A.1.3; Figures A.2, A.3, and A.4).

We further stratified our extrapolation results by species and organ. As shown in Figure 3.4, we found that our model achieved the best performance on all species and organs, with best absolute performance on the mouse testis time series. This demonstrates the benefit of the shared reference space, as mouse's final training timepoint is postnatal day 0 (P0) but the model is able to learn from later development in other species to inform extrapolation for mouse. In contrast, the two worst-performing species were human and chicken, which we believe to reflect larger distributional shifts in the data. All methods struggled on human test data, which are at much later developmental stages than the training dataset. We therefore conducted an analogous Evo-devo experiment, this time extrapolating to the earliest four timepoints as test data. We found that Sagittarius was still the best-performing method, and had stronger performance for extrapolation to early-stage human development (Figures 3.5, A.5), supporting this hypothesis. We believe that the relatively poor chicken performance also stems from a distributional shift, as chicken is the only non-mammal in the dataset. Therefore, the chicken time series are less evolutionarily related to the other time series in the dataset, and may be harder to represent as a transformation of a shared, otherwise mammalian developmental trajectory [55]. After better understanding Sagittarius's strengths and weaknesses, we then studied whether Sagittarius could simulate samples for unmeasured timepoints to gain new insights into tissue differentiation and aging.

3.3.2 Transcriptomic dynamics reveal organ-specific aging genes

To further examine the Sagittarius's extrapolated expression profiles, we next predicted developmental trajectories for each mouse organ, beginning at embryonic day 5.5 (E5.5) and continuing to P63 (using time-

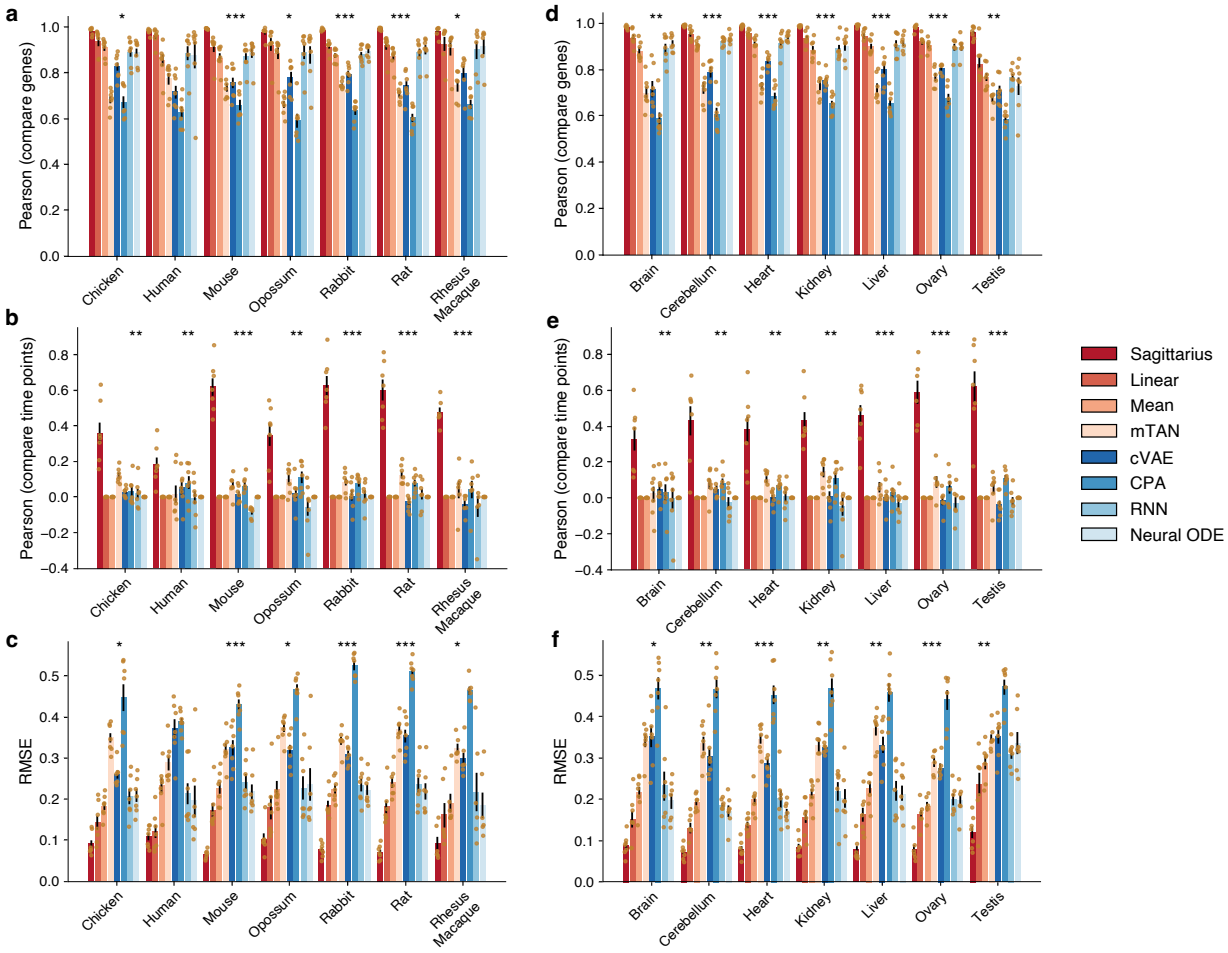


Figure 3.4: **Gene expression prediction for extrapolated timepoints in later-stage development. a-f**, Bar plots comparing the performance of Sagittarius and existing approaches when extrapolating to the four latest timepoints in the Evo-devo dataset. Test sequences are subdivided by species (**a-c**) and by organ (**d-f**). For Pearson correlation, comparing genes (**a,d**) or comparing timepoints (**b,e**), higher correlations indicate better performance; for RMSE (**c,f**), lower error indicates better performance. Data are presented as mean values +/- standard error. When stratified by species (**a-c**), n=6 organ time series for Rhesus Macaque and n=7 organ time series for all other species. When stratified by organ (**d-f**), n=6 species time series for ovary and n=7 species time series for all other organs. The * indicates that Sagittarius outperforms the next-best-performing model in the metric, with significance levels of p-value $< 5\text{e}{-2}$ for *, p-value $< 5\text{e}{-3}$ for **, and p-value $< 5\text{e}{-4}$ for ***. We use a one-sided Fisher z-transformed test for Pearson correlation comparing genes and comparing timepoints, and a one-sided t-test for RMSE.

point labels ranging from -5 to 13 with granularity 0.1, resulting in 180 predicted measurements per organ; Section 3.2.9). By extrapolating to early timepoints, we expect to observe a hypothetical trajectory that includes organogenesis, which takes place between E6.5 and E8.5 in mouse development [99, 100]. Specif-

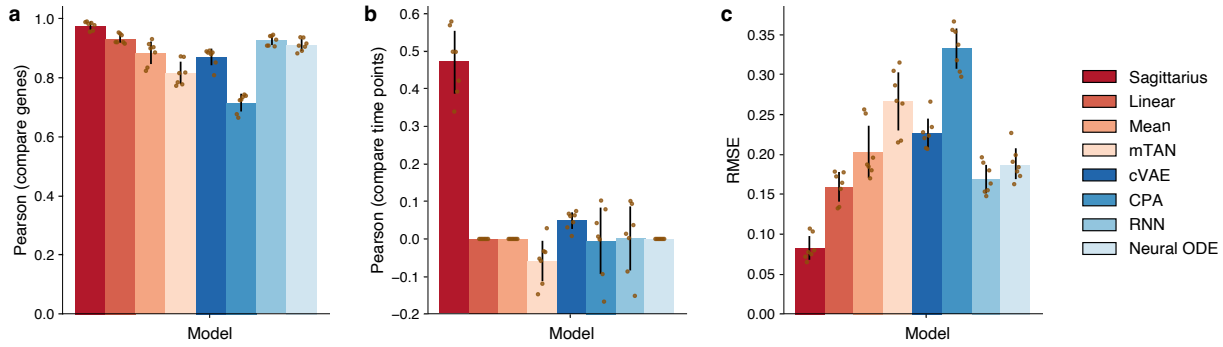


Figure 3.5: **Human gene expression extrapolation performance for Evo-devo extrapolation to early timepoints. a-c,** Bar plot comparing Sagittarius and existing approaches in terms of Pearson correlation comparing genes (**a**), Pearson correlation comparing timepoints (**b**), and RMSE (**c**) of the predicted human expression profile and measured human expression profile of each organ when extrapolating to the first four measured sequence timepoints in the Evo-devo dataset. For Pearson correlation, comparing genes or comparing timepoints (**a,b**), higher values indicate better performance; for RMSE (**c**), lower values indicate better performance. Data are presented as mean values $\pm$ standard error, with $n = 7$ organ time series.

ically, we expect that the earliest extrapolated timepoints result in very similar expression profiles across the different queried organs, which would not have differentiated at this stage. In subsequent days, we would then expect the organs to diverge according to germ layer, before finally separating by organ [55, 99–102]. We visualized the uniform manifold approximation and projection (UMAP) [88] embedding of the simulated organ time series (Figure 3.6a,b), as well as the top principal components [103] (Figure A.7). Our findings largely aligned with the understanding of mouse organogenesis. Namely, the developmental stage dominated the gene extrapolation measurements at the earliest timepoints, with multiple organs grouped in the same location of the UMAP space. At later timepoints, we found that the predicted expression values for brain and cerebellum were more closely grouped together, as were expression measurements for the heart, ovary, and testis, consistent with the ectoderm, mesoderm, and endoderm tissue germ layer classifications [55].

Given the increasing tissue-specific signal in Sagittarius's simulated gene expression vectors at later timepoints, we then investigated which genes most contributed to the differentiation of organ trajectories during development (Section 3.2.9). Excluding the heart and cerebellum, which we found to be the most developmentally distinct for many genes, we found that mouse *Xrn2* expression levels were comparable across organs at early extrapolated timepoints but differed significantly at later timepoints (ANOVA p-value

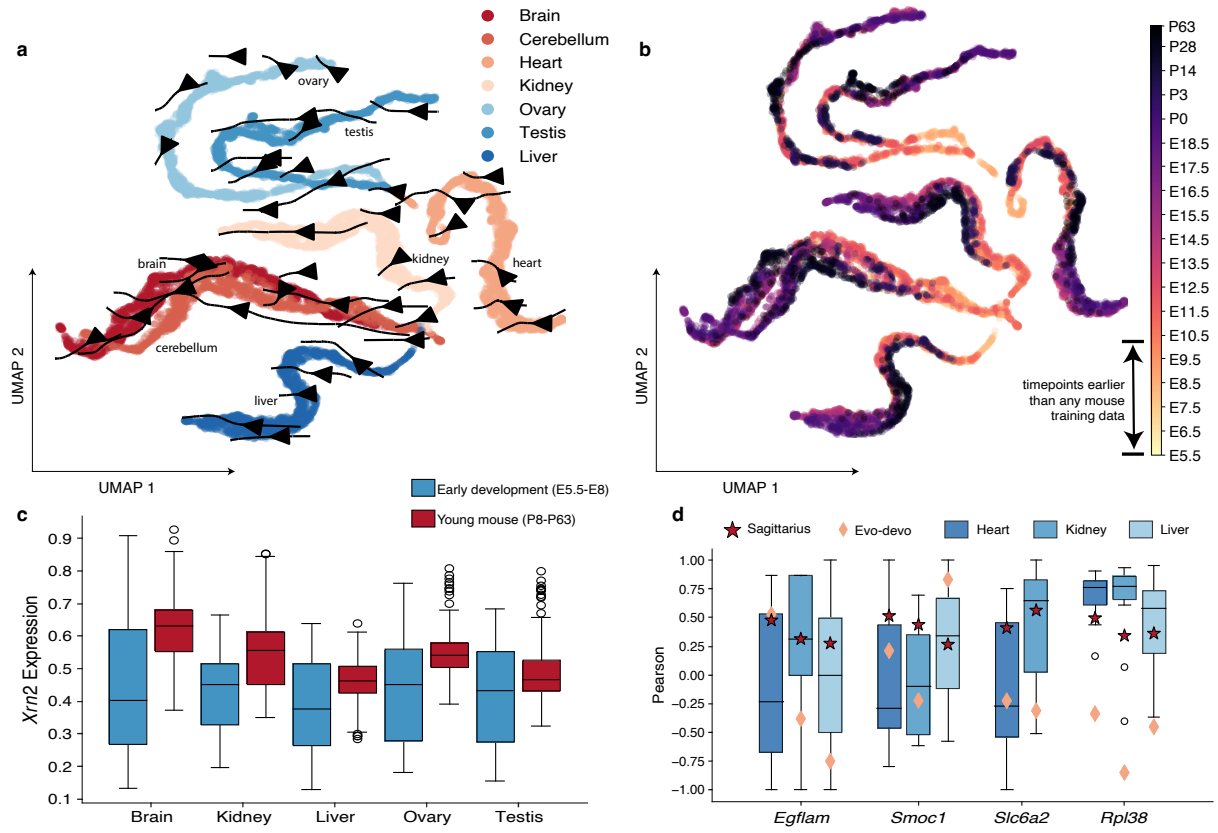


Figure 3.6: **Mouse transcriptomic velocity across organs.** **a,b**, UMAP plots showing simulated mouse gene expression from embryonic day 5.5 (E5.5) to postnatal day 63 (P63) for 7 organs, colored by organ (**a**) and time (**b**). The arrows in (**a**) indicate the transcriptomic velocity of each organ. **c**, Box plot comparing the simulated expression of *Xrn2* at early development (E5.5-E8) to young mouse (P8-P63) across five organs, with $n = 250$ simulated measurements per organ. *Xrn2* expression is not statistically different between the brain, kidney, liver, ovary, and testis organs at the early development (ANOVA p-value < 0.83), but differs between organs at the young mouse time range, particularly with lower expression levels in the liver relative to other organs (ANOVA p-value $< 9.37\text{e}{-78}$). **d**, Box plot examining the consistency of gene expression temporal patterns between simulated data and scRNA-seq data for *Egflam*, *Smoc1*, *Slc6a2*, and *Rpl38* in different tissues over time. Boxes indicate the distribution of cell type correlations for each tissue in *Tabula Muris Senis*, with $n = 7, 5, 2$ cell types in heart, kidney, and liver respectively for *Egflam*; $n = 7, 6, 3$ respectively for *Smoc1*; $n = 5, 7$ in heart and kidney respectively for *Slc6a2*; and $n = 9, 19, 8$ cell types in heart, kidney, and liver for *Rpl38*. Better predictions are closer to the distribution of *Tabula Muris Senis* cell type correlations for each tissue. The star shows the Pearson correlation from Sagittarius's simulated correlation for aging mouse tissues (140 timepoints beginning at P14), and the diamond shows the correlation with respect to time of the younger mouse organs measurements in the Evo-devo dataset. In both **c,d**, the box bounds show the interquartile range (IQR) from quartiles 1 to 3 (Q1-Q3), with the centerline indicating the median and the whiskers extending 1.5 IQR from the box.

> 0.05 and p-value $< 1e{-}98$ respectively), with lower expression levels in the liver than other organs at late timepoints (Figure 3.6c). Existing work has found that mouse *Xrn2* and its roundworm orthologue *xrn-2* play important biological roles during development [104–111], and *XRN2* overexpression in human has also been tied to poor liver cancer prognosis [112].

We next sought to further examine Sagittarius's organ-specific extrapolation potential in mouse. We predicted a transcriptomic profile trajectory beginning at P14 and continuing into senescence, using time-point labels ranging from 11 to 25 with granularity 0.1, resulting in 140 predicted measurements per organ (Section 3.2.9). The latest mouse measurement in the Evo-devo dataset is taken at P63, so we used the *Tabula Muris Senis* single cell RNA-seq dataset [67], which spans from a 1-month-old mouse to a 30-month-old mouse, to validate our results. We compared the Pearson correlation of the gene expression over time between the extrapolated profiles and the *Tabula Muris Senis* data for each tissue, and for mouse genes including *Egflam*, *Smoc1*, *Slc6a2*, and especially *Rpl38*, which previous work has suggested could regulate developmental processes in a tissue-specific way [113], found that Sagittarius's extrapolated aging trajectory aligned with the *Tabula Muris Senis* tissue measurements better than younger mouse trajectory taken directly from Evo-devo (Figure 3.6d). We attribute this to the shared reference space, which can identify aging and senescence patterns from other species like human and rhesus macaque to inform transcriptomic extrapolation for mouse aging. After applying Sagittarius to the Evo-devo dataset, we next considered whether the model could successfully extrapolate unmeasured experimental combinations in settings with multiple continuous variables.

3.3.3 Sagittarius simulates unmeasured drug perturbations

We next evaluated Sagittarius on extremely sparse multivariate data with multiple continuous temporal variables, thereby exponentially increasing the space of possible experimental settings. We applied Sagittarius to the larger, high-dimensional LINCS L1000 pharmacogenomics dataset [53]. In the LINCS dataset, compounds are experimentally applied to cell lines at specific doses and for a given treatment time before measuring the drug-induced expression profiles, although only 1.77% of possible drug and cell line combinations are screened (Figure 3.7a). Sagittarius models each treatment experiment in two continuous dimensions: dose and treatment time. Specifically, the model extends to $B = 2$ continuous axes in this

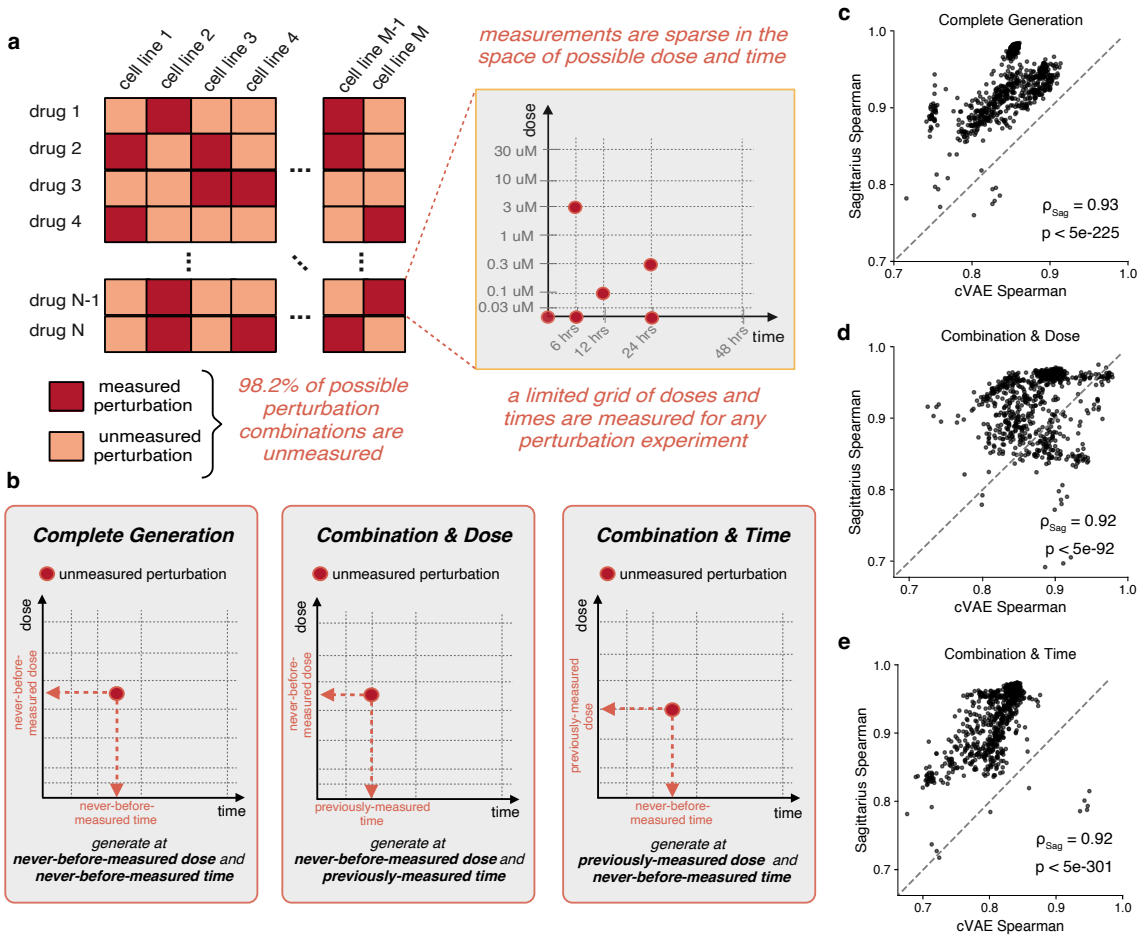


Figure 3.7: **Drug-induced gene expression extrapolation at unmeasured experimental combinations, doses, and times.** **a**, The LINCS pharmacogenomic dataset contains gene expression measurements from a set of experiments where a cancer cell line is treated with a therapeutic compound. The set of measured cell lines and compounds is sparse, with fewer than 1.77% of possible drug-and-cell-line pairs measured. The measured experiments are also only measured at select dose and treatment times, and the entire dataset includes a limited number of dose and treatment times. **b**, Illustration of the three extrapolation tasks we evaluate for the LINCS dataset: complete generation, where we predict an unmeasured cell line and compound experiment at both a dose and time that are unmeasured by any experiment in the dataset; combination & dose, where we predict an unmeasured cell line and compound experiment at a time that has been measured in the dataset but a dose that is unmeasured by all experiments; and combination & time, where we predict an unmeasured cell line and compound experiment at a dose that has been measured in the dataset but a time that is unmeasured by all experiments. **c-e**, Scatter plots comparing the average Spearman correlation of simulated test combinations from Sagittarius and the existing cVAE model for each test drug on the complete generation (**c**), combination & dosage (**d**), and combination & time (**e**) extrapolation tasks. Sagittarius's average correlation across test datapoints is reported inline along with the one-sided Fisher z-transformed test p-values testing whether Sagittarius outperforms cVAE in each settings. Figure created with `BioRender.com`.

setting.

To validate Sagittarius's ability to extrapolate to new perturbation experiments, we then designed three extrapolation tasks: complete generation, combination & dose, and combination & time (Figure 3.7b). For each task, we randomly selected five drug and cell line treatment combinations to mask from the training data, requiring that both the drug and cell line of each combination appeared at least once in the remaining, visible training set. In every setting, the resulting time series are treated as test data. In complete generation, we additionally select three non-zero doses and one non-zero treatment time at random from the train data; in combination & dose we randomly select three non-zero doses; in combination & time we randomly select one non-zero treatment time. All measurements in the training data taken at one of these selected doses or treatment times is then hidden from the training set and added to the test set. This resulted in 2144 training sequences and 269 validation sequences for each of the three tasks. There were 7651, $27{,}242$, and $10{,}417$ total training measurements and 924, 3326, and 1202 validation measurements for the complete generation, combination & dose, and combination & time tasks respectively. Finally, the three tasks had 7441, 7377, and 7395 test sequences with $15{,}068$, $14{,}905$, and $14{,}966$ test measurements respectively.

For each task, we trained Sagittarius on a subset of the LINCS dataset, withholding the remaining measurements as test data. We compared Sagittarius's performance to a cVAE [23], the only comparison approach that could handle multiple continuous variables off-the-shelf. Evaluating model predictions based on held-out test data, we found that Sagittarius had an average Spearman correlation $\rho^{(genes)}$ of 0.93, 0.92, and 0.88 for the three tasks respectively, compared to 0.85, 0.88, and 0.81 for the cVAE (Figure 3.7c-e; one-sided Fisher z-transformed test p-value $< 5\text{e}{-225}$, $5\text{e}{-92}$, and $5\text{e}{-301}$ respectively). This indicates that aligning all perturbations experiments to the shared reference space enables Sagittarius to accurately extrapolate drug-induced gene expression vectors for unmeasured drug treatment experiments at doses and times that are not contained in the training data, enabling an easy, unbiased search approach to drug sensitivity markers. This can greatly increase our understanding of the molecular basis of cancer and of drug response.

3.3.4 A drug sensitivity similarity network for drug repurposing

Gene expression has been widely used to identify the drug-induced and diseased-induced gene expression signatures in drug repurposing studies [114–116], partly due to the scale at which analyses can be efficiently performed and validated. As Sagittarius can accurately predict expression for any perturbation combinations, we applied Sagittarius to drug repurposing by constructing a similarity network of extrapolated perturbation experiments (Section 3.2.10). We found that communities within the network demonstrated a pattern with respect to treatment sensitivity, with average half-maximal inhibitory concentration (IC_{50}) doses of 1.68, 1.83, 1.90, and 2.40 uM in the Genomics of Drug Sensitivity in Cancer (GDSC) dataset [92] (Figure 3.8a). To further investigate the potential for drug-repurposing opportunities, we conducted a case study on a 8-experiment subgraph from the sensitive community, shown in the inset of Figure 3.8a.

The subnetwork includes six perturbations for the breast carcinoma cell line MCF7 and non-small cell lung cancer cell line A549, all of which are measured in the GDSC dataset. A549 is sensitive to treatment with Vorinostat, Gefitinib, and Selumetinib (IC_{50} of 0.49, 0.67, 0.83 uM respectively), and MCF7 is sensitive to treatment with Palbociclib, MK-2206, and Gefitinib (IC_{50} of 0.40, 0.89, and 1.03 uM respectively). The existence of edges between different drugs for the same cell line and the edge between two cell lines for the same drug connect to drug-repurposing work based on cell line gene expression signatures [117] and drug mechanisms of action [118].

Importantly, by comparing extrapolated differential expression signatures to those of successful treatments, Sagittarius enables drug repurposing recommendations where neither the drug nor cell line needs to occur in a known successful therapy. The 8-perturbation subnetwork also includes the prostate adenocarcinoma cell line PC3 treated with Pictilisib and the colorectal adenocarcinoma cell line HT29 treated with Nintedanib, although these drugs and cell lines do not appear elsewhere in the subnetwork. The GDSC dataset does not screen either of these treatment combinations, but previous work has found that Pictilisib inhibited PC3 proliferation ($IC_{50} = 0.28$ uM) [119] and Nintedanib had significant antitumor efficacy in HT29 xenograft models [120, 121] and cell lines ($IC_{50} = 1.40$ uM) [121]. This implies that Sagittarius can extrapolate perturbation experiments to identify candidate drug repurposing targets across cell lines, cancer types, and therapeutic compounds, creating new opportunities for inexpensive and unbiased drug screening as an initial step in the precision medicine pipeline.

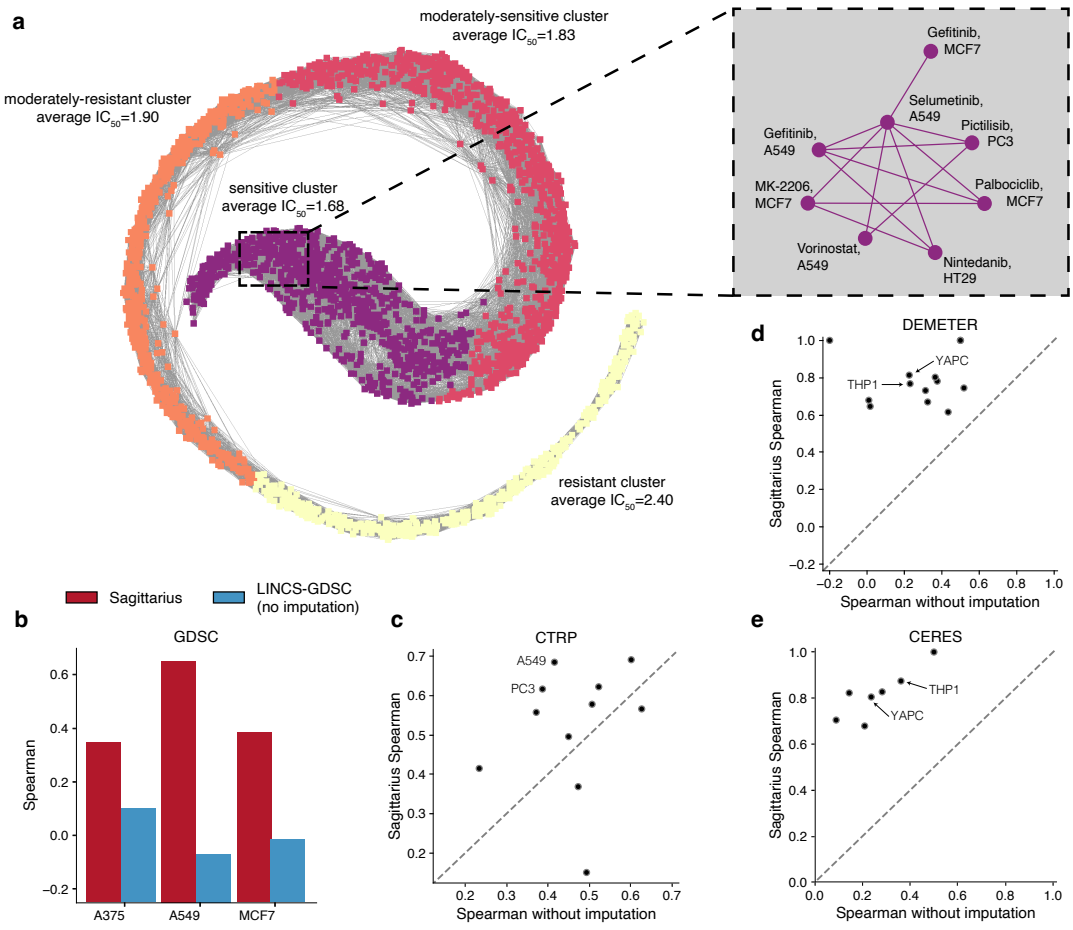


Figure 3.8: **Drug and cell line treatment efficacy extrapolation analysis. a**, k-nearest neighbor network where each node represents a drug and cell line combination, with edges between the most similar drug-induced expression effect. The four communities in the graph are shown in different colors and labeled according to the average GDSC-measured IC_{50} dose of that community. The inset shows a connected 8-node subgraph from the sensitive community. **b,c**, Bar plot (**b**) and scatter plot (**c**) of Spearman correlation between predicted and GDSC-measured (**b**) or CTRP-measured (**c**) IC_{50} doses per cell line, comparing a neural network trained with imputed data from Sagittarius and a neural network trained with only the GDSC or CTRP treatment combinations that are also measured in LINCS. Each cell line has one correlation coefficient, meaning $n = 1$ cell line sensitivity ranking for the bar plot in **b**. Points above the $y = x$ line are cell lines for which Sagittarius's imputed dataset improved the downstream prediction accuracy (**c**). **d,e**, Scatter plot of Spearman correlation between predicted and DepMap-measured cancer gene essentiality scores for each cancer line, with the DEMETER (**d**) and CERES (**e**) DepMap versions. A neural network trained with imputed data from Sagittarius is compared to a model trained on the LINCS treatment combinations that correspond to cell line and gene pairs in the DEMETER or CERES datasets. The $y = x$ line is also shown.

3.3.5 Perturbation augmentation improves drug response prediction

Given its drug repurposing potential, we systematically evaluated Sagittarius on two large-scale cell line drug response prediction datasets, GDSC [92] and the Cancer Therapeutic Response Portal (CTRP) dataset [94]. Drug-induced expression profiles have been useful for drug response prediction [122], but are expensive to measure compared to basal cell line expression, making Sagittarius's extrapolated data especially valuable. We used a downstream neural network to predict the IC_{50} label for treatment perturbations, and compared the performance of a model trained on extrapolated data from Sagittarius to a model trained on the available measured perturbations in LINCS (Section 3.2.11). When trained on data from Sagittarius, the downstream model had an average Spearman correlation of 0.46 and 0.52 per test cell line, comparing the predicted drug sensitivities to the measured sensitivities in GDSC and CTRP respectively (Figure 3.8b,c). In comparison, the model trained on the available experimentally-measured data had an average correlation of 0.004 in GDSC and 0.42 in CTRP. We attribute the poor GDSC performance to the little overlap in treatments screened by the two datasets and consequently to the extremely small size of the training dataset. Sagittarius, in contrast, was able to extrapolate missing profiles and improve downstream performance, particularly for cell lines and drugs that were among the most frequently measured in the LINCS dataset (Figures A.8 and A.9). This shows that Sagittarius can take advantage of the many perturbation experiments to inform better predictions for each drug and cell line, even when applied to unmeasured or sparsely measured combinations.

3.3.6 Improved cancer-essential gene prediction using Sagittarius

In addition to drug response, we analyzed Sagittarius's ability to predict cancer gene essentiality. We used the Cancer Dependency Map (DepMap), considering both the DEMETER [96] and the CERES [97] versions. We used a downstream neural network to predict gene essentiality in a given cell line, where we featurized each cell line and gene pair using a drug-induced expression vector from the given cell line and a LINCS drug with the target gene of interest (Section 3.2.12). We trained one version of the downstream model on extrapolated data from Sagittarius and another on the available experimentally-measured drug-induced profiles in LINCS. Comparing the predicted gene essentiality scores with the labels in DepMap, the model trained on data from Sagittarius obtained an average test Spearman correlation of 0.789 and 0.816 per

cell line for DEMETER and CERES respectively, relative to 0.278 and 0.261 for the model trained solely on the *in vitro* LINCS data (Figure 3.8d,e). Again, we found that the Sagittarius-backed model particularly improved predictions for well-measured LINCS cell lines including the THP1 leukemia and YAPC pancreatic cell lines (Figure A.10). We attribute the strong performance across many different cancer types and drug targets to the shared reference space, where dose- and treatment-time response can be compared across cancer cell lines and compounds.

3.3.7 Simulating mutation profiles for early-stage cancer patients

Having extrapolated gene expression time series in one- and two-continuous dimensions, we then sought to apply Sagittarius to cancer survival time data. We focus on extrapolating somatic mutation profiles as strong signals of disease, but also validate our results on patient gene expression profiles (Figure A.15). It remains very challenging to measure genomic profiles from patients with nascent tumors, as they are rarely diagnosed at this stage [123], and yet these initial mutations can be the most informative as to the cancer's mechanisms and potential early-intervention therapies before other passenger mutations accumulate [32]. We therefore designed a time-series formulation for patient data from 24 cancer types in The Cancer Genome Atlas (TCGA) dataset [56], where extrapolation to later timepoints corresponds to the mutation profiles of patients with longer survival times (Figure 3.9a)[5]. In particular, we formulated our experiment by treating the $k \in [1, T_i)$ patients from the cancer type of interest i with earliest event times as the training data; the remaining $T_i - k$ patients were used as the test set. All patients from other cancer types are also included in the training set. To account for censored event times, we leverage recent machine learning techniques [83, 84] to exclude censored patients whose event time likely differs from their survival time. After filtering all cancers types following the steps in Section 3.2.6 (Figure A.11), our mutation dataset contained 2297 cancer patients. The sarcoma (SARC) cancer type time series contains 115 patients, 31 of whom had a censored death event (Figure 3.9b).

Focusing on the SARC and thyroid carcinoma (THCA) cancer types as case studies, we designated the longest-surviving patients as test data and used Sagittarius to extrapolate mutation profiles with the same survival times. Varying the number of test patients $T_i - k$ to examine the models' performance in

[5]The results shown here are in part based upon data generated by the TCGA Research Network: `https://www.cancer.gov/tcga`. I further thank the specimen donors for their contributions towards advancing science and medicine in oncology.

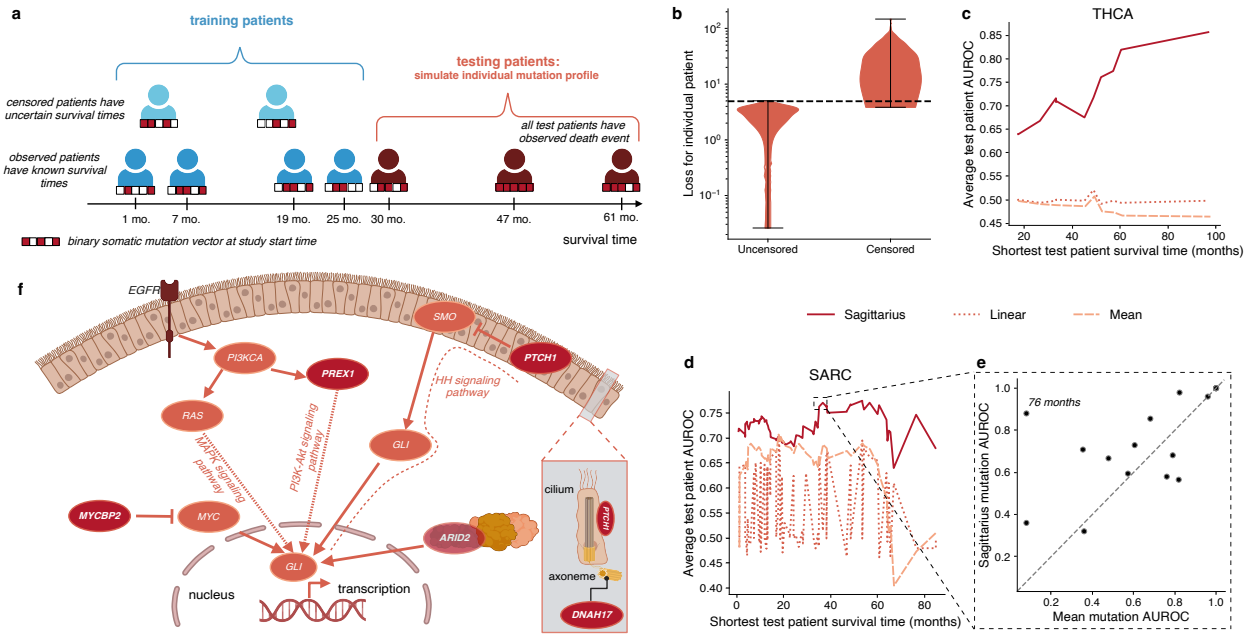


Figure 3.9: **Early cancer patient mutation profile extrapolation.** **a**, Illustration of the training and testing splits for a given cancer type in the TCGA extrapolation task, where training patients have the shortest survival times and test patients have longer survival times for that cancer type. **b**, Violin plot of the survival time regression model's absolute error per patient for the SARC cancer type, divided according to the patient's censoring label. We remove all patients with a loss above the dashed line from the dataset, and train Sagittarius only on the patients below the dashed line. **c,d**, Plot of the average predicted mutation profile AUROC for each of the THCA (**c**) and SARC (**d**) cancer type test splits, ordered according to the shortest survival time in that test split. **e**, Scatter plot comparing the per-patient predicted mutation profile AUROC from Sagittarius and the mean comparison approach for the SARC test split including patients with an observed death event more than 37 months after diagnosis. Points above the $y = x$ line indicate that Sagittarius had a better predicted mutation profile than the comparison approach. **f**, Illustration of the ties between the *GLI* oncogene in the Hedgehog (HH) signaling pathway and the *PTCH1*, *PREX1*, *MYCBP2*, *ARID2*, and *DNAH17* genes that Sagittarius predicted as among the most likely mutations in early-stage sarcoma patients. Figure created with `BioRender.com`.

different settings led to 9 distinct THCA test splits and 61 SARC test splits. We found that Sagittarius had an average test set AUROC of 0.72 and 0.73 between the extrapolated and measured mutation profiles for THCA and SARC respectively (Figure 3.9c,d), representing 45% and 11% improvements respectively over classical methods, and also improved over other deep learning methods (Figure A.13). As a further case study, we focused on a sarcoma patient with a 76-month post-diagnosis survival time, where Sagittarius had particularly improved the test AUROC (Figure 3.9e). The patient had a mutation in *LRP1B*, but the mean method assigned 0 probability to this mutation, reflecting the observed distribution of sarcoma patients

with worse prognosis. Meanwhile, Sagittarius predicts *LRP1B* as the fourth most likely mutation, perhaps learning that *LRP1B* mutations are associated with good therapeutic response in many cancer types [124]. Sagittarius also assigns higher likelihood to common sarcoma mutations like *ADGRV1* [125], suggesting that the model can jointly leverage patterns within the SARC training data and patterns from other cancer types to improve extrapolation.

3.3.8 Hedgehog signaling pathway in simulated early-stage sarcoma

Having confirmed our ability to extrapolate mutation profiles for sarcoma patients with longer survival times, we retrained Sagittarius on all cancer type time series and then extrapolated gene mutation profiles for 27 early-stage sarcoma patients (Section 3.2.13, Figure A.14), resulting in the most-likely mutated gene set *DNAH17*, *PREX1*, *EGFLAM*, *FAM47B*, *DSEL*, *ARID2*, *TRPM1*, *NLGN1*, *PTCH1*, and *MYCBP2*. We found that many of these genes are related to the Hedgehog (HH) signaling pathway and improper activation of the *GLI* oncogene (Figure 3.9f), which has been connected to improved survival outcomes in sarcoma patients [126, 127]. For instance, *PTCH1* is a tumor suppressor gene in the HH pathway connected to some sarcomas [128–130]; *MYCBP2* encodes a protein that has been shown to interact with *GLI* [131] via *MYC* upregulation [132, 133]; the protein encoded by *ARID2* directly interacts with *GLI1* [134, 135]; *DNAH17* encodes a protein that affects the HH pathway through its role in the primary cilia [136, 137]; *PREX1* encodes a protein whose pathway has been associated with *GLI* code regulation [138] and cross-talk with the HH pathway in melanoma [130, 139]. In addition to these connections to the *GLI* oncogene, *EGFLAM* has been shown to induce activation of *PREX1* [140], and *NLGN1* encodes a protein that was found to be significantly enriched with the HH pathway in colorectal carcinoma [141] (Section A.1.4). Furthermore, previous gene expression analyses found that the HH signaling pathway was enriched for differentially expressed genes in multiple sarcomas [142, 143]. Sagittarius's extrapolated mutation profiles therefore point to the HH signaling pathway and particularly the hyperactivation of the *GLI* oncogene as potentially significant sources of tumorigenesis in sarcomas.

3.4 Discussion

In this chapter, I have presented Sagittarius [26], which enables extrapolation of genomic profiles from sparse, heterogeneous time series without requiring aligned timepoints or batch correction between different experimental conditions. Although Sagittarius can extrapolate to unseen timepoints, the model still struggles with large domain shifts between developmental stages in training and test, as we identify in the Evo-devo human extrapolation task. Similarly, Sagittarius is unable to extrapolate to precise timepoints. The learned mapping to and from the shared reference space enables comparison between diverse and unaligned time series, but also warps the queried and measured timepoints to align with biological age and thereby alters the timepoints outside the range of the dataset in potentially unforeseen ways. In the future, Sagittarius could be integrated with models to predict the chronological and biological age associated with a new sample. Furthermore, Sagittarius could benefit from work in model calibration [144] to output a model confidence score along with a predicted profile.

Sagittarius is inspired by decades of work in modeling cell dynamics. One key difference between Sagittarius and pseudotime cell fate models like Monocle [101], Palantir [145], and Slingshot [146] is that these models reconstruct cell lineage within the bounds of one or more originator states and one or more terminal states, while Sagittarius is able to extrapolate beyond the range of timepoints in the data. Pseudodynamics [147] later extended cell fate modeling to time-resolved steps, enabling extrapolation, but operates in a low-dimensional cell state space. Most recently, PRESCIENT [148] enabled single-cell transcriptomic trajectory modeling, but suffers worse performance in bulk expression modeling (Figure A.6). Sagittarius strives to build upon these works by leveraging time series from related yet heterogeneous biological time series for accurate high-dimensional profile extrapolation.

Notably, Sagittarius is able to simulate measurements that would otherwise be extremely expensive to obtain. Datasets that including aging and senescence measurements are uncommon due to study costs, and those that do exist are parts of massive consortia efforts [67]. Furthermore, these efforts only exist for limited model organisms, and are impractical to scale to understudied experimental conditions. Similarly, publicly available pharmacogenomic screens do not include all possible combinations of drug and cell line pairs. This problem is only further exacerbated when considering drug synergy experiments [149], where drugs are applied in combination. However, Sagittarius is able to simulate informative drug-induced gene expres-

sion measurements for untested combinations and learns useful drug sensitivity relationships for treatments without an IC_{50} label in our external datasets. Finally, obtaining early-stage cancer patient somatic mutation profiles is extremely valuable for understanding the sources of tumorigenesis, but would require a massive operation in longitudinally tracking a large population with minimally invasive technology for many years [123]. Inevitably, such a study would be extremely expensive, and the sample size would need to be large enough to account for covariates and a relatively low cancer incidence rate overall. By generating these measurements *in silico*, Sagittarius is able to contribute to our understanding and interpretation of biological time series at low costs by effectively leveraging data that are already available.

Chapter 4

Capricorn: Hi-C contact map resolution enhancement

I now turn to a setting where high-quality experimental data are themselves very expensive to procure. This chapter contains work previously published in Fang et al. [27][1].

In cases where high-quality experiments are very expensive, even single experiments can be prohibitively costly for many data-generating labs. Specifically, I focus on experimental measurements of the three-dimensional (3D) genome architecture, each of which require many internal samples to cover the size of the whole genome (approximately 3 billion base pairs in human) with identifiable structural features and specific attribution of interactions to genomic loci. These two needs are often in conflict with each other: *High-resolution* high-throughput chromosome conformation capture (Hi-C) [41] contact matrices reveal the detailed 3D architecture of the genome (providing specific attribution), but are very expensive to generate at *high-coverage* (providing dense interaction maps that work well with existing chromatin structure analysis tools). To assist with these data, computational methods can learn to transform patterns from lower-quality, less expensive data to simulate the high-quality, prohibitively expensive measurements.

In this chapter, I present Capricorn[2], a deep learning method for Hi-C [41] resolution enhancement that incorporates small-scale chromatin features and leverages a diffusion probability model backbone to gener-

[1] As part of this work, I contributed to conceptualizing the setting and experiments; I also wrote the manuscript and designed the figures.

[2] The code used in this chapter is available at `https://github.com/CHNFTQ/Capricorn`.

ate a high-coverage, high-resolution contact matrix from a low-coverage, high-resolution input. We show that Capricorn outperforms the state of the art in a cross-cell-line setting, improving on existing methods by 17% in mean squared error and 26% in F_1 score for chromatin loop identification from the generated high-coverage data. We also demonstrate that Capricorn performs well in the cross-chromosome setting and cross-chromosome, cross-cell-line setting, improving the downstream loop F_1 score by 14% relative to existing methods. Finally, we use DNA sequence to validate discovered loops and find that the fraction of CTCF-supported loops from Capricorn is similar to those identified from the high-coverage data. Capricorn is a powerful Hi-C resolution enhancement method that enables scientists to find chromatin features that cannot be identified in the low-coverage contact matrix.

4.1 Hi-C resolution enhancement

Chromosomes encode genetic and epigenetic cellular programs, leveraging a complex 3D genome architecture in eukaryotic cells that is critical for many biological processes, including modulating gene regulatory relationships, RNA splicing sites, and DNA repair mechanisms [150, 151]. These architectures within a cell can be measured with assays such as Hi-C [41], genome architecture mapping (GAM) [152], split-pool recognition of interactions by tag extension (SPRITE) [153], and HiChIP [154]. *High-resolution* Hi-C datasets employ smaller bin sizes, thereby more precisely characterizing genomic substructures, but consequently require more experimental reads in order to produce a suitably dense matrix. Importantly, doubling the resolution requires quadrupling the experimental read counts due to the pairwise nature of interactions [155].

In order to obtain denser contact matrices, computational *resolution enhancement* methods take low-coverage, high-resolution Hi-C matrices with fewer measured contacts and generate the corresponding high-coverage, high-resolution matrices. Existing approaches adopt techniques from computer vision, such as convolutional neural networks (CNNs) [156–159], generative adversarial networks (GANs) [160–164], and Markov random fields (MRFs) [165] to produce detailed contact matrices. The resulting data can then be used to perform analyses of genome folding, including classification of large-scale chromatin features like A/B compartments [41, 166–169] and identification of small-scale chromatin features like topologically associating domains (TADs) [170–174] and chromatin loops [11, 175–178]. Although existing approaches

Method	Publication date	Pixel-wise loss	Adversarial loss	TV loss	Perceptual loss	Biology	Backbone
HiCPlus [159]	Feb. 2018	✓					CNN
hicGAN [164]	July 2019		✓				CNN (GAN)
HiCNN2 [158]	Oct. 2019	✓					CNN
HiCNN [157]	Nov. 2019	✓					CNN
HIFI [165]	Jan. 2020					✓	MRF
DeepHiC [163]	Feb. 2020	✓	✓	✓	✓		CNN (GAN)
SRHiC [156]	Apr. 2020	✓					CNN
HiCSR [160]	July 2020	✓	✓		✓		CNN (GAN)
VEHiCLE [162]	Apr. 2021	✓	✓			✓	CNN (GAN)
EnHiC [179]	July 2021	✓	✓		✓		CNN (GAN)
HiCARN-1 [161]	Apr. 2022	✓		✓	✓		CNN
HiCARN-2 [161]	Apr. 2022	✓	✓	✓	✓		CNN (GAN)
Capricorn [27]	June 2024 (ours)	✓				✓	Diffusion

Table 4.1: **A summary of existing Hi-C resolution enhancement methods.** We indicate whether each method adopts common computer vision loss terms, with "TV" for total variation and where pixel-wise losses include mean squared error (MSE) and L1 loss. Notably, only VEHiCLE [162], HIFI [165], and Capricorn incorporate biological features into the resolution enhancement model. Although VEHiCLE uses TAD identification in its loss and HIFI accounts for TAD boundaries during local smoothing, Capricorn is trained to enhance loop and TAD features along with the contact matrix. Capricorn is the only method that uses a diffusion model as the backbone. We limit the comparison to methods for Hi-C resolution enhancement for proximal contacts (interaction within 2 megabases) that do not require additional high-coverage input data, excluding distal-contact enhancement techniques like BoostHiC [180] and reference-based methods like RefHiC-SR [181].

aim to minimize mean-squared error (MSE) and perform well according to metrics developed for natural image analyses, these metrics may not be well suited to teach the model to capture biologically relevant chromatin features, and especially small-scale loops.

We hypothesize that resolution enhancement can produce contact matrices that better capture these higher-order chromatin structures if we design a loss function that explicitly models structures like loops and TADs during resolution enhancement (Table 4.1). Such an approach can both provide additional supervision for the resolution enhancement task and help teach the model to distinguish and enhance particularly interesting genomic contacts. We therefore propose Capricorn, which incorporates additional biological views of the contact matrix to emphasize important chromatin interactions and leverages powerful diffusion models from computer vision [52] for the model backbone (Figure 4.1). In particular, Capricorn learns a diffusion model that enhances a five-channel image, containing the primary Hi-C matrix as well as representations of loops, TADs, and distance-normalized counts computed from the original low-coverage matrix. Capricorn thereby learns meaningful structural contacts as well as the general high-coverage matrix structure.

We compare Capricorn to four existing Hi-C resolution enhancement approaches representing the state of the art, and we find that Capricorn outperforms the others in terms of both MSE and its ability to detect

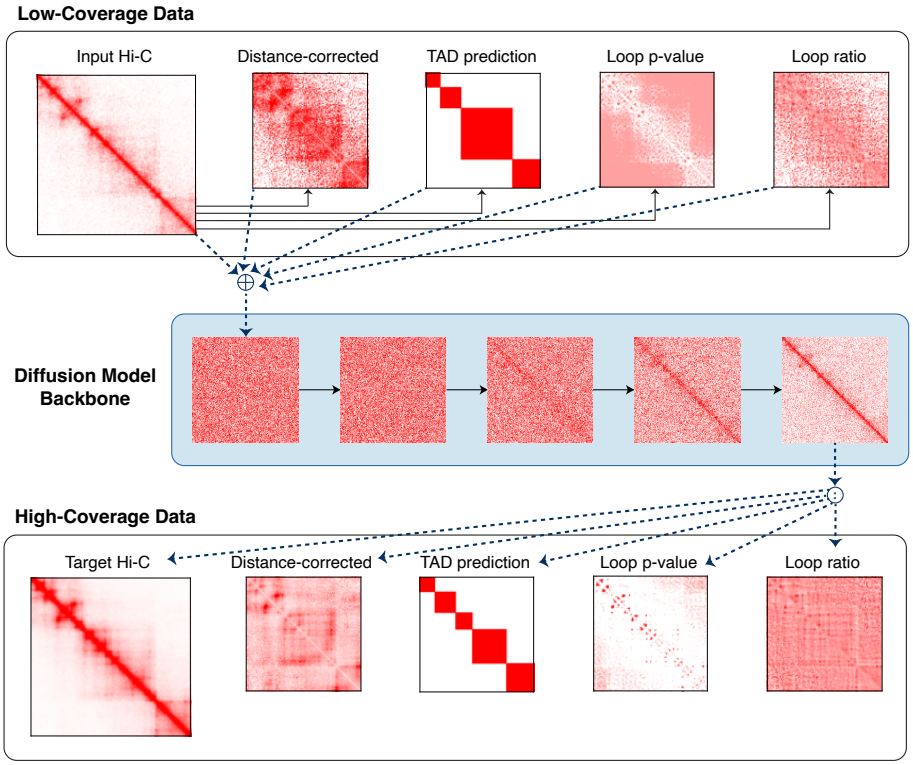


Figure 4.1: **Overview of Capricorn architecture.** Given the low- and high-coverage Hi-C contact matrices we compute small-scale chromatin features that explicitly teach Capricorn to recognize biologically meaningful contacts. We then leverage a diffusion model backbone to iteratively denoise a random contact matrix, conditioned on the low-coverage data.

loops from the predicted high-coverage contact matrix. We tested the models' generalizability across both cell line and chromosome and found that Capricorn's enhanced matrices had a 17.3% lower MSE and 25.6% higher loop F_1-score on average when transferring learned patterns across cell lines. We further found that Capricorn's key idea of incorporating higher-order chromatin features as additional input views is a broadly applicable technique that improved the comparison approaches as well, though Capricorn's diffusion model backbone still provides the best performance. As a final validation, we use DNA sequence to evaluate the fraction of identified loops from the high coverage, low coverage, and Capricorn-generated contact matrices that are supported by flanking CTCF motifs, and find that Capricorn's loops have comparable CTCF support to the high-coverage-derived loops. In summary, Capricorn is a general-purpose approach for chromatin conformation capture contact matrix resolution enhancement, and in the future additional feature views can be easily incorporated into the framework for various downstream tasks of interest.

4.2 Methods

4.2.1 Problem setting

In the supervised resolution enhancement task, we are given a dataset with N pairs of low- and high-coverage contact matrices $\{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{N}$. For a fixed resolution Δ defining the size of each grouped (binned) genomic locus and chromosome length of L base pairs, the contact matrix shapes are identical, with $\mathbf{X}, \mathbf{Y} \in \mathbb{N}^{L/\Delta \times L/\Delta}$. However, the high-coverage matrix $\mathbf{Y}$ contains γ-fold more measured contacts in the matrix. The aim is then to obtain a model f_θ such that $\mathbf{Y}_i \approx f_\theta(\mathbf{X}_i)$. Furthermore, we want $f_\theta(\cdot)$ to generalize to both new cell lines and new chromosomes.

We process the input Hi-C matrices for efficient and structurally informative resolution enhancement. Most Hi-C interactions occur between nearby intrachromosomal loci [11]. As we focus on small-scale chromatin loops, we follow previous work in resolution enhancement [161, 164] and restrict the model to enhance intrachromosomal contacts within 2 megabases (Mb) of each other. For computational efficiency, we further tile paired contact matrices' near-diagonals into 40×40 non-overlapping matrices using a resolution $\Delta = 10$ kb such that each input covers a 400^2 kb region of contacts, consistent with many existing Hi-C resolution frameworks [156–161, 163, 164].

4.2.2 Chromatin structure resolution enhancement

The key idea behind Capricorn is that explicitly modeling small-scale chromatin features such as loops and TADs will improve the method's ability to identify and enhance meaningful contacts. We therefore train Capricorn to enhance the structural interpretation of the low-coverage contact matrix as well as the low-coverage contact matrix itself, thereby explicitly training the model to recognize important 3D chromatin structures from low-coverage data.

Toward this end, we compute additional views of the paired $(\mathbf{X}, \mathbf{Y})$ contact matrices using chromatin features derived from $\mathbf{X}$ and $\mathbf{Y}$.

Distance-corrected view

Capricorn includes a view that accounts for the fact that nearby loci are more likely to interact by chance than more distal loci, such as the strong number of contacts observed along the diagonal of an uncorrected Hi-C matrix. Therefore, one common preprocessing step for Hi-C analyses involves correcting for this proximity bias [182–184].

Consider the input, low-coverage Hi-C contact matrix $\mathbf{X} \in \mathbb{N}^{l \times l}$, where $l = \lfloor \frac{L}{\Delta} \rfloor$ indicates the number of binned genomic loci in the matrix. The distance-correction module $D : \mathbf{X} \rightarrow \mathbf{X}^{(oe)}$ is defined as follows. First, compute a distance-based expected matrix by taking the contacts for all pairs of loci (i, j) that are $|i - j|$ apart in the genome. In practice, this is computed by taking the average along each diagonal of the contact matrix as

$$\mathcal{E}(\mathbf{X})[i, j] = \frac{1}{l - |i - j|} \sum_{k=0}^{l-|i-j|} \mathbf{X}[k, k + |i - j|] \tag{4.1}$$

This can be repeated for all $i \in [1, \ldots, n]$ and $j \in [1, \ldots, n]$ to produce $\mathcal{E}(\mathbf{X}) \in \mathbb{R}^{l \times l}$. After observing that most distance-expected values occurred in the range $[0, 16]$, we further clamp and normalize a distance-corrected version of the input matrix as

$$\mathbf{X}^{(oe)}[i, j] = \begin{cases} 0 & \text{if } \mathcal{E}(\mathbf{X})[i, j] = 0 \\ \frac{\min\left(\frac{\mathbf{X}[i,j]}{\mathcal{E}(\mathbf{X})[i,j]}, 16\right)}{16} & \text{otherwise.} \end{cases} \tag{4.2}$$

We follow the same steps to compute $\mathcal{E}(\mathbf{Y})$ and $\mathbf{Y}^{(oe)}$ given the high-coverage contact matrix $\mathbf{Y}$.

Chromatin loop views

We include two additional views based on chromatin loop structure identified from the input contact matrices. We use the Hi-C Computational Unbiased Peak Search (HiCCUPS) loop calling tool [11]. HiCCUPS tests whether the measured contacts for possible (i, j) loop anchor points are significantly more enriched than the surrounding contacts. HiCCUPS computes both a *loop ratio* of measured contacts relative to the experimental background as well as a *loop p-value* indicating the significance of this enrichment.

Specifically, the method combines a 10×10 donut kernel centered at a specific locus with horizontal, vertical, and lower-left quadrant kernels. For a given locus (i, j), we use the distance-based expected matrix

as defined in (4.1) to compute

$$\alpha_p = \kappa_p * \mathbf{X} \qquad\qquad \beta_p = \kappa_p * \mathcal{E}(\mathbf{X}) \tag{4.3}$$

for each kernel $\kappa_p = \{\kappa_1, \ldots, \kappa_4\}$. Then we compute the *loop ratio* for the input matrix as

$$\mathbf{X}^{(loop-r)} = \min_{p \in \{1,\ldots,4\}} \frac{\beta_p \mathbf{X}}{\alpha_p \mathcal{E}(\mathbf{X})}. \tag{4.4}$$

We follow the HiCCUPS algorithm and fit a Poisson distribution to the observed measurements with λ-chunking, where we set $\lambda = 2^{1/3}$ as in the default HiCCUPS settings [11]. In the λ-chunking protocol, we define $\mu_{i,j} = \max_c \lambda^c$ subject to $\frac{\alpha_p}{\beta_p}\mathcal{E}(\mathbf{X})[i,j] < \lambda^c$, grouping pixels into relative intensity ranges of $\{[0,1), [1, 2^{1/3}), [2^{1/3}, 2^{2/3}), \ldots\}$. Finally, we use the result of λ-chunking to compute the *loop p-value* as

$$\mathbf{X}^{(loop-p)}[i,j] = \max_{p \in \{1,\ldots,4\}} \mathtt{BH}_{\mu_{i,j}}\big(f(\mathbf{X}[i,j] \geq t \big| \mu = \mu_{i,j})\big), \tag{4.5}$$

where $f(\cdot \geq t|\mu)$ represents the survival function of the Poisson distribution with expected value μ, and $\mathtt{BH}_{\mu_{i,j}}$ indicates the Benjamini-Hochberg false discovery rate correction [185, 186] over the p-values for all pixels with the same λ-chunked $\mu_{i,j}$. We compute $\mathbf{Y}^{(loop-r)}$ and $\mathbf{Y}^{(loop-p)}$ based on the ground-truth high-coverage matrix $\mathbf{Y}$ in an analogous manner.

TAD view

We compute the insulation score (IS) [170] for TAD detection, using sliding windows along the contact matrix diagonal to identify insulated regions with low scores and in-domain regions with high scores. As the average human TAD is approximately 1 Mb long [11, 187], this view provides chromatin context that extends beyond a single tiled 400^2 kb submatrix, thereby providing the resolution enhancement model information about the greater surrounding chromatin structure. We smooth the IS scores by averaging them over a 21-bin domain and assign the within-TAD label 1 to contiguous regions with monotonically decreasing ISs centered around some locus (i, j), where all ISs are still larger than the contact matrix average IS and the IS of (i, j) is at least 10% larger than the IS at the region boundary.

Multi-view resolution enhancement construction

Importantly, we compute each of the low-coverage chromatin features directly from the low-coverage experimental matrix $\mathbf{X}$, so there is no leakage between the high-coverage contact matrix and low-coverage inputs to Capricorn, which would prevent Capricorn's practical utility during inference. The original contact matrices and derived chromatin structures are then concatenated to form the full input and output

$$\begin{aligned}\tilde{\mathbf{X}}(\mathbf{X}) &= [\mathbf{X}, \mathbf{X}^{(oe)}, \mathbf{X}^{(tad)}, \mathbf{X}^{(loop-p)}, \mathbf{X}^{(loop-r)}] \\ \tilde{\mathbf{Y}}(\mathbf{Y}) &= [\mathbf{Y}, \mathbf{Y}^{(oe)}, \mathbf{Y}^{(tad)}, \mathbf{Y}^{(loop-p)}, \mathbf{Y}^{(loop-r)}],\end{aligned} \tag{4.6}$$

such that $\tilde{\mathbf{X}}(\mathbf{X}), \tilde{\mathbf{Y}}(\mathbf{Y}) \in \mathbb{R}^{5\times L/\Delta \times L/\Delta}$.

4.2.3 Multi-view weighting

Each of the computed biological views naturally has a different distribution of values and presents different challenges for correct prediction. To correct for these differences, we perform a two-stage iterative view-weighting process based on the low-coverage representation $\tilde{\mathbf{X}}(\mathbf{X})$.

Given the input, multi-view low-coverage contact matrix representation $\tilde{\mathbf{X}}(\mathbf{X}) \in \mathbb{R}^{5\times L/\Delta \times L/\Delta}$, we computed an initial view weight vector $\omega_0 \in \mathbb{R}+^5$ as

$$\omega_0[v] = \sqrt{\frac{1}{|\Xi|}\sum_{\xi\in\Xi}\frac{\texttt{var}(\tilde{\mathbf{X}}(\mathbf{X}_\xi)[0]}{\texttt{var}(\tilde{\mathbf{X}}(\mathbf{X}_\xi)[v]}} \tag{4.7}$$

for the vth view, where $\texttt{var}(\cdot)$ denotes variance, Ξ indicates the set of all chromosomes tested for the given cell line, and $\tilde{\mathbf{X}}(\mathbf{X}_\xi)$ indicates the views computed from the low-coverage cis-contact matrix $\mathbf{X}$ for chromosome ξ. The computed ω_0 for both GM12878 and K562 is reported in Table A.2.

We then further refine the weights to account for the difficulty of generating the high-resolution version of each view. Specifically, we compute a view-specific difficulty score using the validation set, as

$$d[v] = \texttt{mse}\left(f\left(\sqrt{\omega_0}\tilde{\mathbf{X}}(\mathbf{X}^{\mathbf{val}})\right)[v], (\omega_0\tilde{\mathbf{Y}}(\mathbf{Y}^{\mathbf{val}})[v])\right), \tag{4.8}$$

where $\texttt{mse}(\cdot,\cdot)$ indicates the mean squared error, $f(\cdot)$ is the diffusion model backbone, and $(\mathbf{X}^{\mathbf{val}}, \mathbf{Y}^{\mathbf{val}})$ are the paired low- and high-resolution contact matrices in the validation set. Finally, we compute $\omega \in \mathbb{R}+^5$ as

$$\omega[v] = \sqrt{\frac{\min_{u\in\{0,\ldots,4\}} d[u]}{d[v]}} \text{ for } v = 1,\ldots,4 \qquad\qquad \omega[0] = 1. \tag{4.9}$$

We explicitly set the weight of the 0th view, which contains the low-coverage input contact matrix without further alteration, to 1. This avoids altering the basic matrix view that we aim to enhance. The precise weight vector ω is given in Table A.2 for both GM12878-based training and K562-based training.

During all subsequent training runs, we more precisely compute the input as

$$\tilde{\mathbf{X}}(\mathbf{X}|\omega) = \left[\sqrt{\omega[0]}\mathbf{X}, \sqrt{\omega[1]}\mathbf{X}^{(oe)}, \sqrt{\omega[2]}\mathbf{X}^{(tad)}, \sqrt{\omega[3]}\mathbf{X}^{(loop-p)}, \sqrt{\omega[4]}\mathbf{X}^{(loop-r)}\right] \tag{4.10}$$

$$\tilde{\mathbf{Y}}(\mathbf{Y}|\omega) = \left[\sqrt{\omega[0]}\mathbf{Y}, \sqrt{\omega[1]}\mathbf{Y}^{(oe)}, \sqrt{\omega[2]}\mathbf{Y}^{(tad)}, \sqrt{\omega[3]}\mathbf{Y}^{(loop-p)}, \sqrt{\omega[4]}\mathbf{Y}^{(loop-r)}\right]. \tag{4.11}$$

For simplicity of notation, we refer to this as $\tilde{\mathbf{X}}(\mathbf{X})$ and $\tilde{\mathbf{Y}}(\mathbf{Y})$ in the manuscript for all model training after the initial weight tuning.

4.2.4 Review of diffusion probability models

We use a *diffusion probability model* backbone [49, 50] to carry out the resolution enhancement task given the low-coverage contact matrix and derived chromatin feature views. Diffusion probability models are easier to train than generative adversarial networks and recently have been shown to excel in image generation tasks [188–190].

To the best of our knowledge, Capricorn is the first approach for Hi-C resolution enhancement that leverages diffusion probability models, and we therefore provide a brief overview here.

Diffusion models leverage a latent Markov chain framework that iteratively denoises an input to produce diverse and realistic outputs [49, 50]. The models are split into a T-step *forward process* $q(\mathbf{y}_t|\mathbf{y}_{t-1})$ and T-step *reverse process* $p(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{x})$, where $\mathbf{y}_0 \in \mathbb{R}^{C\times W\times H}$ is the C-channel original, high-fidelity image, $\mathbf{y}_t \in \mathbb{R}^{C\times W\times H}$ is a latent variable, and $\mathbf{x}$ is an optional input term on which to condition the process.

Specifically, the forward process computes progressively noisier latent representations of the true image as

$$q(\mathbf{y}_t|\mathbf{y}_{t-1}) \sim \mathcal{N}(\sqrt{\alpha_t}\mathbf{y}_{t-1}, \beta_t\mathbf{I}) \tag{4.12}$$

for $t = 1, \ldots, T$, where β_t is defined by a noise schedule, such as a cosine schedule [52, 191], and $\alpha_t = 1 - \beta_t$. The reverse diffusion process, optionally conditioned on input $\mathbf{x}$, can be then parameterized by a learnable $\epsilon_\theta(\cdot)$ as

$$p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{x}, t) = \mathcal{N}\left(\frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{y}_t - \frac{\beta_t}{\sqrt{1 - \prod_{s=1}^{t}\alpha_s}\epsilon_\theta(\mathbf{y}_t, \mathbf{x}, t)}\Big), \beta_t\mathbf{I}\right), \tag{4.13}$$

where $\epsilon_\theta(\cdot)$ is trained to predict noise from the previous step's $\mathbf{y}_t$ prediction and the conditional input $\mathbf{x}$. We denote a sample from a step of the reverse process as $\hat{\mathbf{y}}_{t-1} \sim p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{x}, t)$, or $\hat{\mathbf{y}}_{t-1} \sim p_\theta$ for brevity of notation.

We specifically focus on conditional diffusion models trained with a MSE objective in the pixel space of an image, or the bin space of a contact matrix. We compute the loss

$$\mathcal{L}(\theta) = \mathbb{E}_{(\mathbf{x},\mathbf{y}_0)}\Big[\mathbb{E}_{t\in\{1,\ldots,T\}}\Big[w_t\mathbb{E}_{\mathbf{y}_t|\mathbf{y}_0}\big[\mathbb{E}_{\hat{\mathbf{y}}_{t-1}\sim p_\theta}[\|\hat{\mathbf{y}}_{t-1} - \mathbf{y}_{t-1}\|_2^2]\big]\Big]\Big], \tag{4.14}$$

where w_t is a diffusion loss hyperparameter impacting the the relative amount of sampled Gaussian noise [52]. The expectation over $t \in \{1, \ldots, T\}$ reflects how we can compute loss at a different number of steps of the reverse diffusion process, introducing t noise samples to the target $\mathbf{y}_0$ during the forward process before denoising.

Importantly, the diffusion model framework enables inference to proceed from an initial Gaussian noise sample, provided that the total noise $1 - \prod_{t=1}^{T}\alpha_t$ added by all steps in the Markov chain is sufficiently large. During inference, we can therefore proceed with the optional guiding input $\mathbf{x}$ using

$$\hat{\mathbf{y}}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \in \mathbb{R}^{C\times W\times H} \tag{4.15}$$

$$\hat{\mathbf{y}}_{t-1} \sim p_\theta(\hat{\mathbf{y}}_t, \mathbf{x}, t) \qquad \forall t = T, \ldots, 1. \tag{4.16}$$

This enables a prediction $\hat{\mathbf{y}}_0$ given only the source input $\mathbf{x}$.

4.2.5 Low-coverage guided diffusion

We leverage conditional diffusion probability models for the resolution enhancement task. The combined contact matrix and its derived chromatin feature views can be interpreted as a five-channel image, so we approach the problem as high-coverage image generation where low-coverage inputs used to guide the generation process. Specifically, Capricorn is built around a module that approximates $\epsilon_\theta(\tilde{\mathbf{Y}}_{t-1}|\tilde{\mathbf{Y}}_t, \tilde{\mathbf{X}}, t)$ as part of the reverse process given in (4.13) (Figure 4.1).

Capricorn's multi-channel framework encodes additional biological understanding of the contact matrix data into the diffusion model resolution enhancement task, teaching the model to identify significant contacts while enabling the standard MSE loss formulation. Specifically, we adopt the loss function in (4.14) to operate over all five biological views of the output, therefore treating $\tilde{\mathbf{X}}(\mathbf{X})$ and $\tilde{\mathbf{Y}}(\mathbf{Y})$ as five-channel source and target images respectively. Furthermore, during inference we are able to use the trained diffusion model backbone and low-coverage contact matrix views to generate a high-coverage contact matrix estimate $\hat{\tilde{\mathbf{Y}}} \sim f_\theta(\theta, \tilde{\mathbf{X}}(\mathbf{X}))$ using equations (4.15) and (4.16). After inference, we discard all views except for the enhanced Hi-C contact matrix (the 0th image channel) to produce $\hat{\mathbf{Y}} \in \mathbb{R}^{L/\Delta \times L/\Delta}$.

4.2.6 Performance measures

We evaluate the results using both image-based and biologically-motivated metrics, focusing on the generated high-coverage contact matrix. Specifically, given a model's predicted contact matrix $\hat{\mathbf{Y}}$, we compute the MSE as $\|\hat{\mathbf{Y}} - \mathbf{Y}\|_2^2$.

We also measure the structural accuracy of the generated high-coverage contact matrix, focusing on chromatin loops. We use the enhanced high-coverage matrix to call loops with the HiCCUPS algorithm [11], which uses four different kernels to compare measured contacts with the contacts in the local neighborhood and test for enrichment, producing both an enrichment ratio and p-value. We annotate all loci (i, j) with p-value < 0.1 and enrichment ratio > 1.75 for HiCCUPS's donut or lower-left-quadrant kernels or > 1.5 for the horizontal or vertical kernels as a loop, following the default tool parameters (see Section 4.2.2). As HiCCUPS is the algorithm that we also use for loop-based view computation, we additionally call loops

with the Mustache [175] and Chromosight [192] loop-detection tools during evaluation. Mustache identifies loops using computer vision approach that considers loop features at multiple resolutions, and Chromosight leverages a loop template to detect similar patterns in a given contact matrix. For all tools, we use the default parameters for loop detection.

For each loop calling tool, we compute a loop F_1 score $F_1 = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$ with a 5 pixel tolerance range, such that TP are the true positive loops called from the predicted data that appear within $[i-5 : i+5, j-5 : j+5]$ in the loops called from the ground-truth data, FP are the loops called from the predicted data that do not appear within the five-pixel range from the ground-truth data, and FN the loops that are called from the ground-truth data but do not occur in the five-pixel tolerance range for predicted data. We do not use true negatives in our evaluations, as this would include most of the genome and not be an informative metric.

4.2.7 Hi-C data preprocessing

We collected Hi-C data using the Gene Expression Omnibus (GEO) database [193] for the GM12878 Epstein-Barr-virus-infected human lymphoblastoid cell line and the K562 human chronic myelogenous leukemia lymphoblast cell lines from Rao et al. [11][3], restricting the contact matrix to read mapping quality ≥ 30 and processed to 10 kilobase (kb) resolution following previous work [161]. We adopt the contact matrix preprocessing techniques from HiCARN [161] and DeepHiC [163], including tiling the contact matrices into 40×40 submatrices, only retaining submatrices in the 2 megabase Mb region around the diagonal, clamping the high-coverage matrix to $[0, 255]$ and then normalizing to $[0, 1]$, and clamping the low-coverage matrix to $[0, 100]$ and then normalizing to $[0, 1]$.

In order to simulate low-coverage data, we randomly downsampled the GM12878 and K562 cell line Hi-C matrices [11] to $\frac{1}{16}$ of their original read count. We treated these downsampled data as the low-coverage matrices and the original matrices as their high-coverage pairs. We evaluate model performance in a cross-cell-line setting where we train on either the GM12878 data or K562 data, and test the model on the other cell line. In both experiments, we withhold chromosomes 4, 5, 11, and 14 from the training cell line as our validation set. Chromosome 9 is also excluded from the K562 data due to extreme sparsity at 10 kb

[3] Accession code GSE63525.

resolution.

4.2.8 Existing model implementations

We use the publicly available python HiCARN [161] repository from Github[4] for both their model and contact matrix preprocessing implementation, as well as source code implementations for HiCNN [157] and HiCSR [160]. We retrained all models on our data. We reimplemented HiCCUPS [11] to accept `npz` input files, enabling loop calling for low-coverage and generated contact matrices that had a fixed resolution. We verified that our implementation was reasonable by comparing the number of called loops on the high-coverage, primary GM12878 Hi-C matrix used in our experiments [11]. In total, we call 10,179 loops compared to the default implementation identifying 7949 loops. We also find comparable CTCF support [194] for our called chromatin loops as the original data reported in Rao et al. [11] (see Section 4.3.4, Table 4.2). We used the available Python implementation of Mustache[5] [175] and the packaged version of Chromosight[6] [192].

We use the conditional diffusion probability model Imagen [52] as the resolution enhancement backbone model, updating the model to condition on low-coverage contact matrices rather than text. We choose Imagen rather than other image diffusion models [14, 51, 195] due to its efficient U-Net architecture, which is faster and more memory efficient than other diffusion generators. We accessed Imagen with its Github repository[7]. This enabled model training in approximately 28 hours and inference in approximately 45 minutes on an NVIDIA A4000 GPU in the cross-cell-line experiment.

4.2.9 CTCF loop validation

Given a set of loops called from a Hi-C matrix, we follow the CTCF loop validation protocol used by Rao et al. [11].

1. We obtained CTCF, SMC3, and RAD21 ChIP-seq experimental datasets that identify binding sites along the genome for the given transcription factor.

[4]Github: `https://github.com/OluwadareLab/HiCARN`.
[5]Github: `https://github.com/ay-lab/mustache`.
[6]Package: `https://pypi.org/project/chromosight/`.
[7]Github: `https://github.com/lucidrains/imagen-pytorch`.

2. We would expect CTCF-mediated loops to co-occur with CTCF, SMC3, and RAD21 ChIP-seq peaks. For each loop spanning from locus i to locus j, we identified the *peak-associated loops*, defined as the subset of called loops that also contain a ChIP-seq peak in the loci $[i, j]$ for all of the CTCF, SMC3, and RAD21 datasets. If i and j were fewer than 15 kb apart, we symmetrically expanded the peak search window around the anchor loci until it was 15 kb.

3. Although the presence of a CTCF ChIP-seq peak indicates an accessible CTCF binding site, it does not capture the *orientation*. We therefore leveraged DNA sequence to establish motif orientation at each of the peak-associated loops. We used FIMO [196] to search the human reference genome (build hg19), using a CTCF motif probability weight matrix[8] and a p-value threshold of $1e{-}4$. For each peak-associated loop, we investigated whether each 10 kb region around the anchor loci contained a CTCF motif. If a locus contained multiple nearby CTCF motifs, we assigned the motif with highest predicted likelihood from FIMO [196] as the loop anchor motif. Finally, if the two CTCF motifs assigned to the loop anchor points were in the convergent orientation, we marked the loop as "CTCF validated".

For this analysis, we used the ENCODE data portal [30] to access the ChIP-seq experimental datasets. For GM12878, we used four CTCF ChIP-seq datasets[9], two RAD21 ChIP-seq datasets[10], and one SMC3 ChIP-seq dataset[11]. For K562, we used five CTCF ChIP-seq datasets[12], one RAD21 ChIP-seq dataset[13], and one SMC3 ChIP-seq dataset[14].

4.3 Results

4.3.1 Capricorn accurately enhances contact matrices and loop features

We first sought to evaluate Capricorn in the cross-cell-line setting, where the model is trained on the simulated low-coverage data and measured high-coverage data on one cell line and tested on the simulated

[8] `https://jaspar.elixir.no/api/v1/matrix/MA0139.1.meme`.
[9] Data accession codes: ENCFF833FTF, ENCFF002DAJ, ENCFF473RXY, and ENCFF710VEH.
[10] Data accession codes: ENCFF002CPK and ENCFF753RGL.
[11] Data accession code: ENCFF686FLD.
[12] Data accession codes: ENCFF002DDJ, ENCFF738TKN, ENCFF002CEL, ENCFF002DBD, and ENCFF085HTY.
[13] Data accession code: ENCFF002CXU.
[14] Data accession code: ENCFF041YQC.

low-coverage data of another cell line. We compare Capricorn to four deep learning approaches for Hi-C resolution enhancement. HiCNN [157] trains a deep CNN architecture with pixel-wise MSE. HiCARN-1 and HiCARN-2 [161] both use a deep CNN architecture trained with pixel-wise MSE, total variation loss to encourage relatively smooth output images, and perceptual loss that encourages representative features computed by a separate neural network to be similar for the ground-truth target and machine-generated output; HiCARN-2 further incorporates an adversarial loss term with a GAN framework. HiCSR [160] includes a pixel-wise L1 loss, adversarial loss, and perceptual loss. Unlike the HiCARN models, HiCSR trains a separate network directly on the target high-coverage contact matrices, rather than directly adopting a model trained on natural images.

We find that Capricorn outperforms other methods both in terms of its ability to enhance chromatin loops from the low-coverage data and its accuracy in producing high-coverage data for both test cell lines in the cross-cell-line experimental setting. As shown in Figure 4.2a,b, Capricorn outperforms all other approaches in terms of its ability to recognize and enhance loop chromatin structures when transferred to both the GM12878 and K562 cell lines, a pattern which holds using the HiCCUPS [11], Mustache [175], or Chromosight [192] loop calling tools. Capricorn has an average loop F_1 score of 0.50 and 0.35 when tested on GM12878 and K562 respectively with HiCCUPS, 0.28 and 0.21 for GM12878 and K562 with Chromosight, and 0.42 and 0.34 for GM12878 and K562 with Mustache. The loop F_1 score from the Capricorn-enhanced matrix is significantly better than the next-best performing enhancement tool in five out of six settings (Bonferroni-corrected Wilcoxon signed-rank test p-values $< 1.5\text{e}{-6}$ and $< 3.0\text{e}{-6}$ for GM12878 and K562 respectively when called with HiCCUPS; $< 1.5\text{e}{-6}$ and $< 3.0\text{e}{-6}$ for GM12878 and K562 respectively when called with Chromosight; $< 2.2\text{e}{-2}$ and $< 2.1\text{e}{-1}$ for GM12878 and K562 respectively when called with Mustache). These results demonstrate that Capricorn is able to successfully identify and enhance meaningful biological contacts, such as those involved in loop formation.

We observe both that GM12878 enhancement outperforms K562 enhancement, and that some loop callers lead to better performance than others, though overall performance trends largely remain similar. The performance difference for all methods between GM12878 and K562 test data can be explained by the difference in pairwise contacts in the original dataset: GM12878 has approximately five times more measured contacts than K562 [11]. We also find that Chromosight in particular calls many more loops than

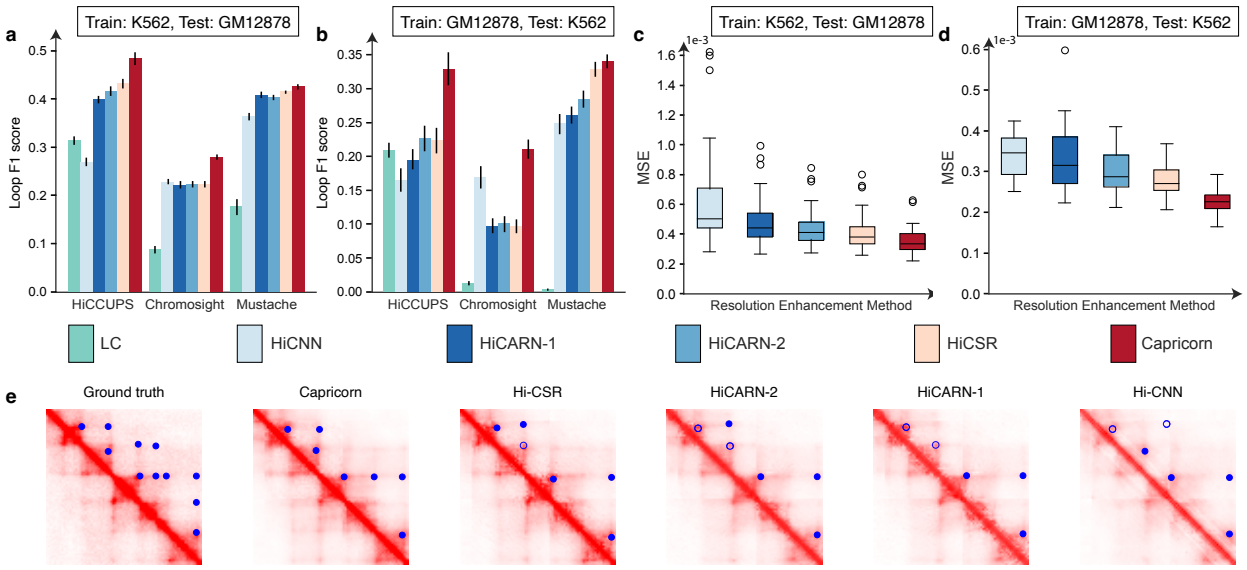


Figure 4.2: **Cross-cell-line resolution enhancement model performance. a,b,** Barplot comparison of F_1-score for loop detection from the generated high-coverage matrices using the HiCCUPS, Chromosight, and Mustache loop calling tools (**a,b**, higher is better), showing the average and standard error bars by chromosome **c,d**, Boxplot comparison of the generated matrix MSE (**c,d**, lower is better), showing the median, interquartile range (IQR), $1.5\times$IQR, and outliers by chromosome. We omit the low-coverage ("LC") baseline here as the MSE is so much larger than other methods. **e**, Ground-truth high-coverage submatrix covering genomic loci from 47.3 Mb to 48.1 Mb on GM12878 chromosome 17, compared to the generated high-coverage submatrix output by each method. Blue circles indicated called loops in the ground-truth or predicted high-coverage data. Circles for loops that are also called in the ground truth data are filled in; loops that are called from the generated data but not the ground-truth data are empty circles.

the other methods, predicting nearly twice the number of loops called by HiCCUPS on high-coverage data (Table A.3), and that this difference becomes even more pronounced for computationally enhanced data (Table A.4). This may be partially attributed to the loop template's sensitivity to slight differences between true experimental contact matrices and computationally generated contact matrices. However, our findings suggest that Capricorn's performance relative to other resolution enhancement approaches is largely robust to the choice of loop calling method. Moving forward, we therefore simplify the analysis and use HiCCUPS [11] as the default loop detection tool.

Capricorn further achieved a lower prediction MSE than any of the comparison approaches, indicating that the additional views can also boost the overall resolution enhancement results. Figure 4.2c,d, shows that Capricorn has an average MSE of $3.6\text{e}{-4}$ and $2.3\text{e}{-4}$ for the GM12878 and K562 test cell lines respectively, relative to $4.3\text{e}{-4}$ and $2.8\text{e}{-4}$ for HiCSR, the best-performing comparison approach, and to $2.4\text{e}{-3}$ and

1.1e−3 if directly using the input low-coverage data scaled by the experimental downsampling rate.

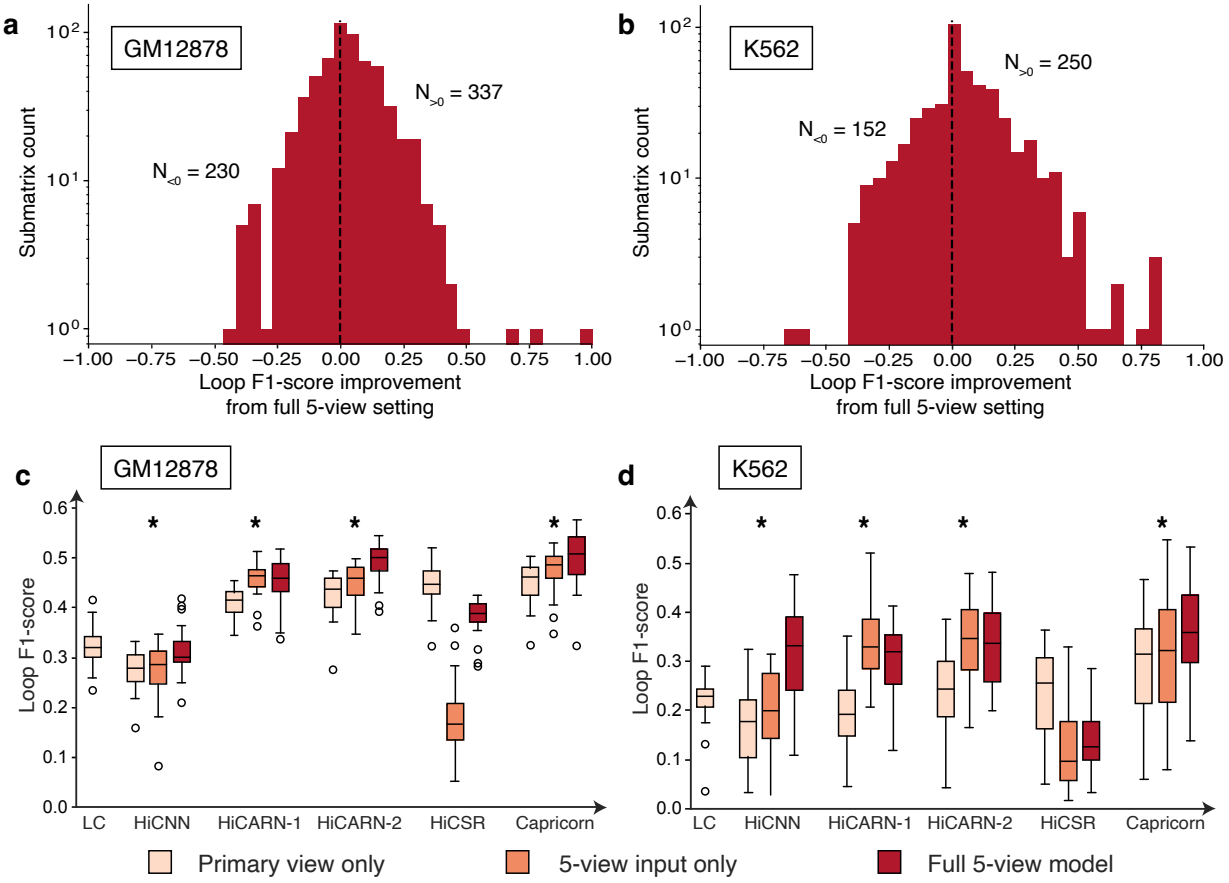


Figure 4.3: **Study of Capricorn's multi-view chromatin feature framework. a,b,** Histogram of the difference in loop F_1-score from the predicted high-coverage contact matrix in the GM12878 (**a**) and K562 (**b**) test datasets using the full five-view Capricorn framework and an alternate version of Capricorn that only includes the primary Hi-C matrix view as the input and output for the diffusion model backbone. The counts $N_{>0}$ and $N_{<0}$ indicate the number of submatrices for which the five-view setting respectively outperform or underperform the primary-only setting. **c,d**, Boxplot comparing the performance of resolution enhancement methods in the primary-view-only setting where enhancement is only performed with Hi-C matrices, the five-view-input setting where small-scale chromatin features are included as input to the resolution model, and the full-five-view setting where small-scale chromatin features are used both as input and output to train the model backbone. "LC" indicates the results taking the low-coverage matrix and scaling all contacts by the downsampling factor 16. The boxplots show the median, IQR, $1.5\times$ IQR, and outliers by chromosome; * indicates that the full five-view model significantly improves on the primary view model with Bonferroni-corrected one-sided paired t-test p-value $< 2e{-}3$.

4.3.2 Small-scale chromatin features are critical to model improvement and are model-agnostic

To better attribute Capricorn's strong performance, we next investigated the specific impact of including small-scale chromatin features as additional views to train the resolution enhancement model and enhance structurally meaningful contacts. We compared Capricorn's performance when trained to enhance chromatin features as well as the Hi-C matrix to its performance when performing resolution enhancement without any additional views. As shown in Figure 4.3a,b, we find that explicitly training the model to enhance meaningful biological contacts, as captured in the small-scale chromatin features, significantly improves our ability to identify these features from the enhanced contact matrices (Bonferroni-corrected Wilcoxon signed-rank test p-value $< 2.3\text{e}{-6}$, $< 4.3\text{e}{-7}$ comparing loop F_1-score over submatrices for test GM12878 and K562 respectively). We also test Capricorn's performance over a subset of the input views, considering settings that incorporate one source of biological evidence (either distance-normalized contacts, loops, or TADs) in addition to the input contact matrix, and settings containing all of Capricorn's biologically-augmented views except one. We find that the full version of Capricorn performs best out of all variants in terms of the loop F_1 score (Table A.3).

Because our multi-view idea is not architecture dependent, we then tested whether the benefits of including small-scale chromatin features as part of model training generalized to other resolution enhancement network architectures. To this end, we updated each of the comparison models to accept multi-channel input matrices and compared the results of three model formulations (Figure 4.3c,d).

1. Primary view: This setting uses the comparison models' default resolution enhancement pipelines with the low- and high-coverage Hi-C matrices as input and output. We also consider Capricorn's performance when trained without the additional biological views in this setting.

2. Five-view input only: This setting uses the additional chromatin feature views as input to the model, but is still trained to only predict the high-coverage view. As shown in Figure 4.3c,d, many methods perform better in this setting than the original primary view, indicating the utility of the additional biological feature inputs.

3. Full five-view model: This setting uses Capricorn's complete multi-view setting, including all five

chromatin feature views as input and training the model to enhance the small-scale chromatin features in addition to the Hi-C matrices. This setting yields the best downstream loop calling performance for Capricorn (one-sided paired t-test p-value < 0.02 relative to the five-view input-only setting) and three of the four comparison approaches. Although the multi-view biological inputs from the five-view input only setting generally improve performance relative to the primary view setting, the additional performance gains in this setting highlight the additional value of including small-scale chromatin features as model outputs that are explicitly included in the loss function.

Although we show our key multi-view idea to be generalizable to many model architectures and loss formulations, we still find that Capricorn's diffusion model backbone outperforms the convolutional models. Comparing the full five-view model results for Capricorn and HiCARN-2 [161], the best-performing comparison approach, Capricorn still performs slightly better than the other enhanced approaches, with a average loop F_1 scores of 0.50 and 0.35 for GM12878 and K562 respectively relative to HiCARN-2's 0.49 and 0.33 (Bonferroni-corrected one-sided paired t-test p-values $< 3e{-}3$ and $< 5e{-}5$, respectively).

Capricorn's multi-view framework is beneficial for loop calling in the generated high-coverage matrix for all models except HiCSR. We hypothesize that HiCSR performs poorly when including additional biological views due to its denoising autoencoder [160] for perceptual loss on the generated outputs, which distinguishes its architecture from HiCARN-2's convolutional GAN architecture. However, this approach may require further hyperparameter tuning such that the perceptual loss term is well balanced with components of the overall loss function, particularly in the full five-view model setting where the additional biological views are modeled by the denoising autoencoder.

This observation points to the broad applicability of Capricorn's key idea, and also suggests opportunities for including additional Hi-C-derived views based on expert domain knowledge for various downstream genome folding analyses.

4.3.3 Capricorn generalizes across chromosomes

To confirm that Capricorn's strong performance is not due to memorizing the training chromosomes with relatively small cell-line differences, we conducted a second experiment to rigorously test Capricorn in the cross-chromosome setting. Here, we withhold chromosomes 2, 6, 10, and 12 as a validation set and reserve

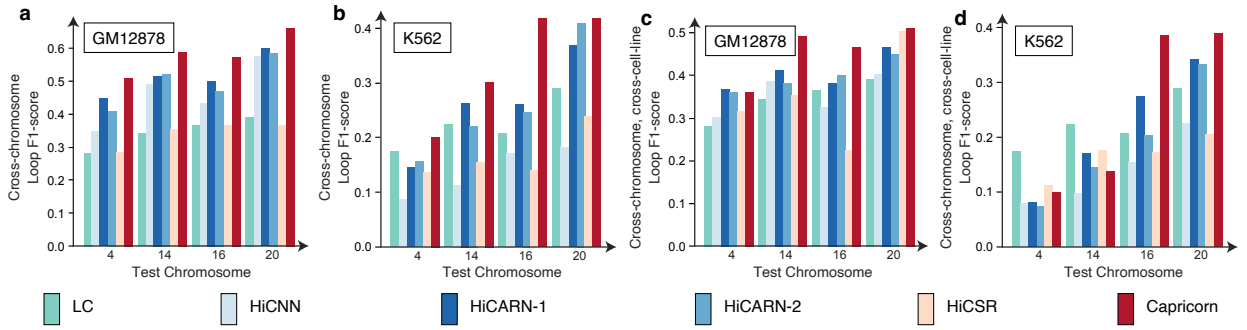


Figure 4.4: **Model performance comparison in cross-chromosome and cross-chromosome, cross-cell-type experimental settings. a-d**, Barplots showing loop F_1-score by test chromosome in a cross-chromosome experiment (**a,b**) and cross-chromosome, cross-cell-type experiment (**c,d**). "LC" indicates the results using only the low-coverage input data. Plots are labeled with the cell line of the test data.

chromosomes 4, 14, 16, and 20 as test data. We carried out two experiments to examine cross-chromosome and intra-cell-line generalization as well as cross-chromosome and cross-cell-line generalization.

We find that Capricorn is able to transfer its learned resolution enhancement patterns to never-before-seen genomic loci in the cross-chromosome intra-cell-line setting better than other methods (Wilcoxon signed-rank test p-value $< 7.9\text{e}{-3}$). Across the four test chromosomes, Capricorn's generated high-coverage contact matrix had an average loop F_1-score of 0.58 in the GM12878 experiment (Figure 4.4a) and 0.35 in the K562 experiment (Figure 4.4b), relative to the best-performing comparison approaches at 0.52 for HiCARN-1 and 0.26 for HiCARN-2. Similarly, Capricorn had the highest loop F_1 scores in the cross-chromosome, cross-cell-line experiments as shown in Figure 4.4c,d (Wilcoxon signed-rank test p-value $< 5.5\text{e}{-2}$), with an average loop F_1 score of 0.46 on GM12878 and 0.25 on K562 test data, relative to the best comparison approaches' respective loop F_1-scores of 0.40 and 0.21 from HiCARN-1. As in the cross-cell-line setting, all methods perform better on GM12878 test data because they contain many more measured contacts than the K562 experimental data. This result highlights Capricorn's ability to generalize the informative, small-scale chromatin patterns it learns across genomic loci as well as cell lines, reiterating the effectiveness of diffusion-based modeling of additional chromatin feature views.

4.3.4 Loops discovered with Capricorn are enriched for convergent CTCF motifs

We further validated the loops identified from Capricorn's generated high-coverage contact matrix with additional data not used in the resolution enhancement framework, including DNA sequence and ChIP-seq

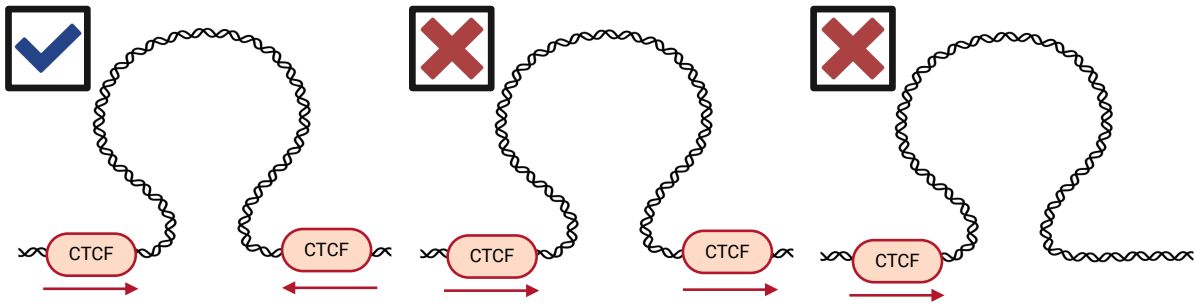


Figure 4.5: **Illustration of convergent CTCF motif validation.** Arrows indicate the directionality of the CTCF binding sites. *Left*, convergent CTCF motifs are supporting evidence for cohesin-mediated loop formation; *middle*, non-convergent CTCF motifs are not positive evidence for loops; *right*, CTCF loops on only one anchor point of the loop are not positive evidence for loops. Figure created with `BioRender.com`.

[197] experimental measurements. The CCCCTC-binding factor (CTCF) is a key protein in 3D structure determination for mammalian genomes [194], and many loops anchor at CTCF motifs [11]. In particular, cohesin-mediated loop formation is facilitated by pairs of flanking CTCF binding sites occurring in opposite orientations. Because our methods do not make use of the primary DNA sequence, we can use pairs of inward-facing CTCF motifs as additional experimental evidence in support of a candidate loop (Figure 4.5).

We followed the previously-described CTCF validation protocol [11]. First, to validate that our implementation is correct, we applied the validation protocol to the high-coverage GM12878 data. We find that a similar proportion of loops are associated with convergent CTCF loops (41% in our analysis versus 42% in the original analysis). The small discrepancy in the percentages is likely due to differences in the exact ChIP-seq experimental data used, our choice of FIMO [196] for motif search, and the preprocessing applied to both the low- and high-coverage contact matrices in our resolution enhancement setting. However, the results are still very similar, and give us confidence in our analysis steps.

Next, we turned to the main experiment, applying the CTCF validation procedure to the loops called for GM12878 and K562 across the high-coverage, low-coverage, and Capricorn-generated contact matrices. Importantly, while many chromatin loops are mediated by CTCF binding sites [11], non-CTCF-mediated loops also exist, so the lack of CTCF anchor motifs should not be interpreted as clear evidence of a false positive; however, we would expect the *ratio* of CTCF-mediated loops to be similar to the high-coverage data for well-enhanced contact matrices.

Cell line	Contact matrix	Total loops	CTCF-validated loops	Validation rate
GM12878	High-coverage	10,176	4182	41.1%
	Low-coverage	7029	1733	24.7%
	Capricorn	5798	2479	42.8%
K562	High-coverage	5142	1932	37.6%
	Low-coverage	4668	738	15.8%
	Capricorn	2223	817	36.8%

Table 4.2: **Analysis of CTCF-validated loops for high-coverage, low-coverage, and Capricorn-generated contact matrices.** We show the total number of called loops, total number of called loops that were also supported by ChIP-seq analyses and inward-facing CTCF motifs in the DNA sequence, and the percentage of called loops that were also validated in our CTCF analysis for GM12878 and K562.

The results (Table 4.2) show that the Capricorn-enhanced matrix identifies plausible loops. Notably, loops identified from Capricorn's generated high-coverage matrices have very similar rates of CTCF support to the loops called from the experimental high-coverage data for both GM12878 and K562, with 41.1% support for high-coverage-based loops compared to 42.8% support for Capricorn-based-loops in GM12878 and 37.6% support for high-coverage-based loops compared to 36.8% support for Capricorn-based loops in K562. By comparison, loops called from the low-coverage data exhibit much lower levels of CTCF support, suggesting that the loop caller's false discovery rate is not well controlled with such sparse contact matrices. This finding is especially important, because it indicates that loop calling results produced from low-coverage data without a resolution enhancement tool may be problematic. Hence, our results further support the need for resolution enhancement methods like Capricorn, which can produce denser contact matrices that are compatible with existing loop calling algorithms. (A second analysis of CTCF validation, following the procedure in Roayaei Ardakany et al. [175], is provided in Section A.2.2.)

4.4 Discussion

In this chapter I have presented Capricorn [27] as a tool for Hi-C resolution enhancement. Capricorn explicitly models the biology underlying experimental contact matrices by incorporating small-scale chromatin features into the model formulation and loss function. Furthermore, we find that this key insight is widely applicable, improving performance for three out of four comparison approaches. However, the small-scale chromatin feature views still perform best with Capricorn's conditional diffusion model backbone.

We demonstrate Capricorn's strong performance in cross-cell-line, cross-chromosome, and cross-chromosome-cross-cell-line settings with the GM12878 and K562 datasets. In all three measured settings, Capricorn is best able to generate high-coverage, high-resolution data containing accurate chromatin loops that can be called with standard tools [11, 175, 192]. This highlights Capricorn's generalizability as well as the value of including additional biological data views, differentiating Hi-C resolution enhancement from super-resolution in natural image applications. Finally, we leverage DNA sequence to further validate the loops identified from Capricorn's generated data and find CTCF support for Capricorn-based loops similar to the experimentally-generated high-coverage-based loops.

In the future, Capricorn's framework can be further broadened to include additional biological views designed specifically for downstream tasks of interest. In particular, new views that contain structural information covering a more than 400^2kb locus could help enhancement for TADs and A/B compartments. Such analyses could also test the relevance of various biological views on different tasks. We also imagine that future work will study the impact of even more model backbones on the multi-view resolution enhancement problem.

Future work can further study Capricorn's generalizability. Here, we have focused on two human cell lines measured with *in situ* Hi-C. Follow-up work can apply Capricorn to more available Hi-C cell line data and study the impact of training data size. This work could also apply Capricorn to a cross-species transfer learning setting, such as applying the model to mouse data after training only on human data. Another generalizability study could apply a model trained on *in situ* Hi-C data to other experimental contact matrix types, such as micro-C [198, 199].

In conclusion, Capricorn is a method that simulates prohibitively expensive, high-quality biological data from their less expensive, lower-quality counterparts. While experimental data are still the gold-standard in biology, this provides an important tool for labs with realistic computational constraints to simulate high-coverage studies. This simulation is a very important step to enable subsequent analyses that fail on low-coverage input data, as I have shown with chromatin loop calling. Therefore, further work on generating high-cost data from low-cost data remains an important research direction for the field.

Chapter 5

Gemini: Biological network integration

Finally, I turn to a different source of biological data cost: expert curation. This chapter contains work previously published in Woicik et al. [28][1].

The best trusted, highest quality biological datasets are usually carefully curated to capture an appropriate breadth and depth of examples, where "appropriate" is defined by domain experts performing the curation. While these resources generally represent the gold standard in biological data, there are still some important drawbacks to keep in mind. First, and perhaps most obviously, this curation process is expensive. Crowdsourced annotation, while feasible in some areas like computer vision with natural images, cannot be applied to most biological topics. (While I can distinguish a cat from a dog, I can't tell you which numerical vector is the profile of a red blood cell and which measures a neuron.) Compensating experts for their time and knowledge is therefore a major cost of curation. Second, repositories take time to curate, leading to temporal bias. This can prevent recent discoveries from appearing in the curated repository even when the high-quality results are publicly available. Lastly, the curators will naturally impose their beliefs in what constitutes high-quality and how to weigh different sources of evidence. While this can often be beneficial, it nevertheless can have unforeseen impacts on the resulting repository, particularly in the era of deep learning.

However, using lightly curated or even uncurated experimental data also comes with its own set of drawbacks. I focus on three main challenges: first, there are many possible experiments to use as data sources,

[1] As part of this work, I contributed to conceptualizing the setting and experiments; I also conducted experiments, wrote the manuscript, and designed the figures.

and the data are often extremely large; second, many experiments likely contain redundant information; third, the associated experimental metadata may be insufficient to identify what datapoints are repeated. In this chapter, I specifically focus on the application of uncurated experimental gene network analysis, where every node in a graph corresponds to a gene and edges indicate some sort of experimentally-derived gene-gene relationship, such as co-expression or physical interaction.

To best leverage uncurated gene network repositories, I proposed Gemini [28], a memory-efficient method to automatically identify and balance unique sources of information from a massive, heterogeneous, and uncurated gene network dataset[2]. More specifically, Gemini performs network integration and produces an embedding representation for each gene, which can then be used for many downstream tasks of interest. Gemini approximates the uniqueness of each input network and uses high-order pooling to represent and weight each network according to its redundancy.

We show that, for a human protein function prediction downstream task, Gemini leads to a more than 10% improvement in F_1 score, 15% improvement in micro area under the precision-recall curve (AUPRC), and 63% improvement in macro-AUPRC by integrating hundreds of experimental networks from BioGRID [46], and that Gemini's performance significantly improves when more networks are added to the input network collection, while Mashup [200, 201] and BIONIC [202] embeddings' performance degrades. Gemini enables memory-efficient and informative network integration for large gene networks, and can be used to massively integrate and analyze networks in other domains.

5.1 Biological network integration

Biological networks can provide insights into the underlying mechanisms of human diseases [203–213], cell differentiation [214, 215], and protein essentiality [216]. As high-throughput functional genomic screens have become more accessible, many large-scale biological networks have been produced, including protein-protein interaction (PPI) networks [217–220], genetic interaction networks [221], metabolic networks [222, 223], disease networks [224, 225], and patient similarity networks [212]. The experimentally-derived networks capture valuable information about the underlying biological system. However, each individual network is noisy and incomplete. Network integration methods seek to denoise the underlying

[2]The code used in this chapter is available at `https://github.com/MinxZ/Gemini`.

biological patterns and pool information from different types of biological networks, which can contain heterogeneous information about the underlying biology based on study conditions and experimental measurement techniques, and thereby improve understanding of gene biology [226]. Importantly, some kinds of networks may be less common than others, such as those derived from functional screens with more expensive equipment. These rare networks may still contain high-quality genetic patterns that are informative for many biological prediction tasks.

Gene networks, where each vertex represents a gene, are one common type of biological network. Such gene networks can be derived from many different experimental sources, including genetic interaction [221], co-expression [227], physical interaction [228], and co-localization [229], among many others. These different types of experimental data show different patterns, which can enhance our biological understanding of genes [216, 230]. In addition, gene networks are often very large, containing several thousand genes each. In this work, we focus on unsupervised network integration methods, which can be broadly applicable for many biological tasks without the need for retraining and which learn gene-gene relationships directly from the interactome, thereby avoiding a bias towards known, labeled genetic interactions [231]. Although unsupervised network integration is an active and prolific area of research, including matrix decomposition approaches [200, 201, 212], deep-learning approaches [202, 232–234], kernel-based approaches [235, 236], and regression-based approaches [237, 238], many existing network integration methods cannot scale to hundreds of genome-scale gene networks. For example, the BioGRID [46] human gene network collection contains 895 gene networks, and jointly storing all network diffusion states [239, 240] would require 2.77 TB of memory. Network integration methods must therefore be very memory-efficient in order to be applied to these data. The scalability challenge, while already a significant constraint for large network collections, is also likely to worsen, as improved high-throughput screening technology enables more measurements for more genes under more conditions. It is therefore critical that gene network integration methods be scalable to increasing numbers of networks and of genes.

Moreover, existing state-of-the-art unsupervised network integration methods for massive biological datasets, such as Mashup [200, 201, 241], assume that important biological signal for a given gene is evenly distributed across networks in the dataset. Although this may hold for smaller, expertly-curated network collections, this is unlikely to extend to hundreds of networks, and is not the case in large interactome

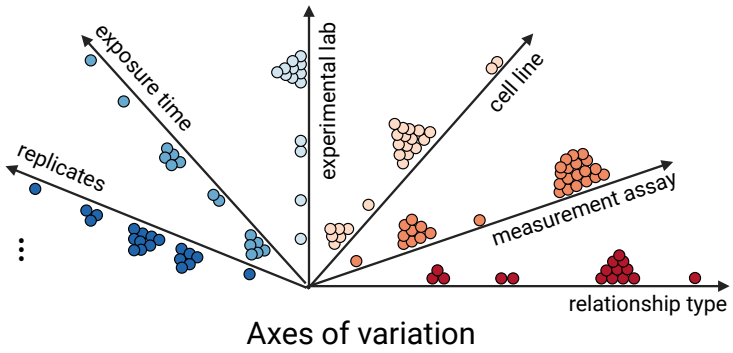


Figure 5.1: **Illustration of unevenly measured networks for different types of latent experimental variation.** Each axis considers different sources of possible variation. Circles represent networks contained in the repository. Along each axis of variation, the distribution of networks is highly non-uniform.

datasets that measure diverse genetic properties [242]. Intuitively, different types of networks, such as co-expression and co-localization [243], can encode different features of a gene. Furthermore, measurements taken from different experimental conditions, such as diseased versus healthy or *in vitro* versus *in vivo*, can also contain complementary information about the underlying genetic system. We can imagine many latent sources of variation in heterogeneous gene network repositories (Figure 5.1); however, these may not be well annotated, particularly for lightly curated datasets.

Importantly, networks derived from rare measurement assays or unusual cell line disease models may reflect conditions that are more expensive to study or culture, and therefore less well studied, but equally high quality as the more redundant experiments. Therefore these unique networks might be even more valuable than the repeated networks due to their added marginal information. This presents two diverging ways in which to consider uniqueness in a network repository: traditional network integration methods often view rare networks as low-quality outliers, less likely to be trustworthy and therefore to have minimal impact on the learned patterns; we view rare networks as sources of unique and valuable information, and therefore try to account for network redundancy during the learning process.

To jointly handle the massive scale of gene network repositories and the latent redundancy of many networks we present Gemini, an unsupervised network integration approach that has a memory-complexity that is constant to the number of networks. Our key idea is to represent the diffusion state of each network as a vector using fourth-order kurtosis pooling. We then weight each network by an approximation of its uniqueness using this vector, which serves as a more memory-efficient surrogate for the full diffusion

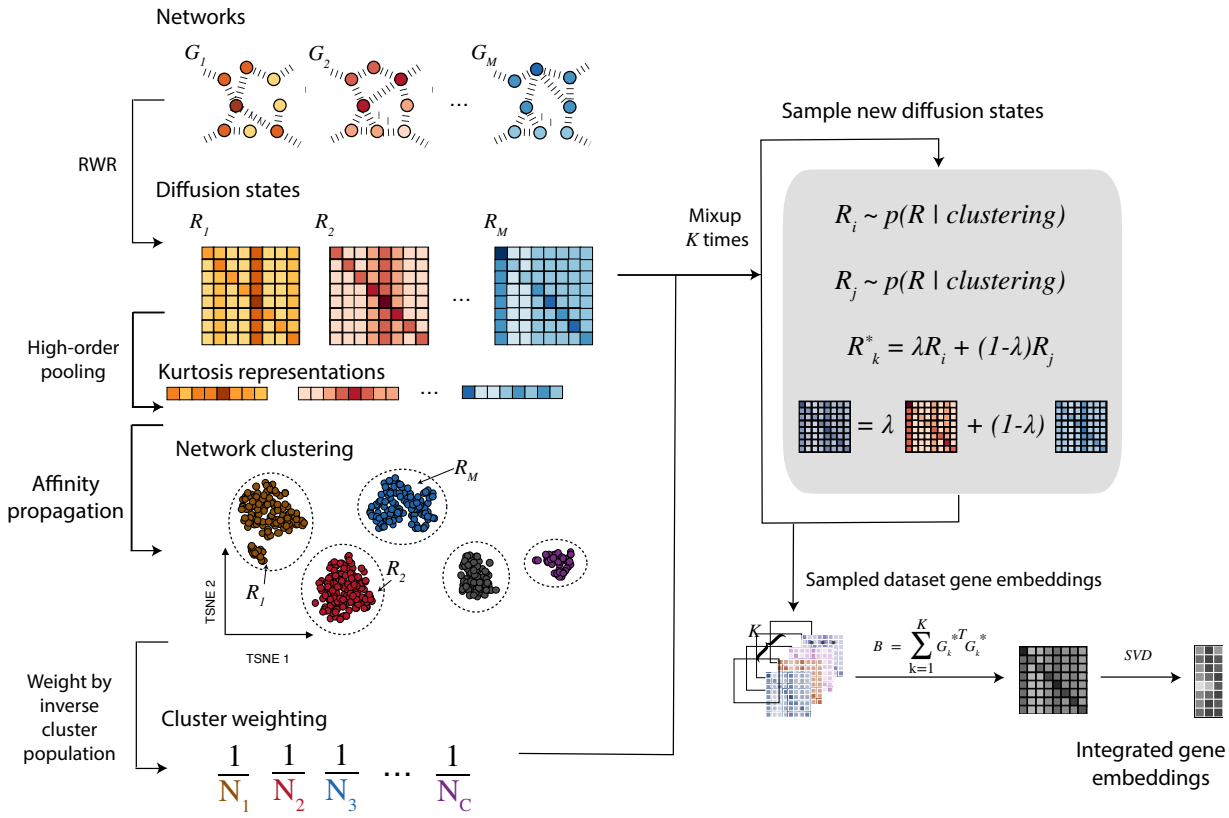


Figure 5.2: **Overview of Gemini algorithm.** Given a set of input networks, Gemini uses random walk with restart to compute the diffusion states. Gemini then uses fourth-order kurtosis pooling of the diffusion state matrix as the feature vectors to cluster all networks. Gemini assigns each network a weight inversely proportional to its cluster size. It then randomly samples pairs of networks according to their weights. These pairs of diffusion state matrices are then mixed-up to create a new simulated network collection that more evenly covers the possible diffusion states. We finally aggregate the synthetic dataset and perform an efficient singular value decomposition to produce embeddings for all vertices in the network collection.

state representation. We then simulate many new networks by sampling and mixing-up original networks according to the uniqueness-based weights (Figure 5.2). This simulation process enables users to either obtain more networks for a more robust integration or consider fewer representative networks for a shorter running time.

We evaluate Gemini on network-based protein function prediction, which has been extensively used to assess biological network integration methods [200, 201, 230, 232, 233, 235–238, 241, 244–249]. When integrating hundreds of human gene networks, we found that Gemini outperforms existing network integration methods by more than 10% in F_1 score, 15% in micro-AUPRC, and 63% in macro-AUPRC. Most importantly, we found that the quality of Gemini's embeddings improved by 15% with respect to down-

stream micro-AUPRC when additional human networks were added to the dataset, while Mashup's [201] performance degraded by 4%. Gemini can therefore balance the many different sources of biological information in the final embeddings while maintaining computational scalability, and can be broadly applied to networks both within and outside of biology for efficient integrative analysis.

5.2 Methods

5.2.1 Problem definition

We define our input dataset of M networks as $\mathcal{G}^{(M)} = \{G_1, G_2, \ldots, G_M\}$, where all G_i have the same n vertices and can be represented by their adjacency matrix $\mathbf{A_i}$. Each network G_i may or may not be weighted or directed. Gemini then outputs an embedding for each vertex, $\mathbf{Z} \in \mathbb{R}^{n \times d}$, where d is a user-defined embedding dimension, and the embedding $\vec{z_j}$ can be used as the feature vector for vertex j in downstream tasks.

5.2.2 Review of Mashup

Gemini's skeleton is based on the memory-efficient version of Mashup [201], the state-of-the-art network integration approach with memory complexity that is constant in the number of networks and quadratic in the number of nodes, which is critical when integrating hundreds of large networks. Fundamentally, Mashup contains three important steps:

1. Compute the diffusion state for each network using random walk with restart (RWR);
2. Integrate the diffusion state for each network;
3. Decompose the integrated states with singular value decomposition (SVD).

In the first step, the diffusion state is calculated from each network's transition matrix $\mathbf{T_i}$ using the RWR algorithm, iteratively computing the state for vertex j in network i as

$$\vec{r_{ij}}^{(t+1)} = (1-\alpha)\mathbf{T_i}^\top \vec{r_{ij}}^{(t)} + \alpha\vec{e_j} \qquad\qquad \vec{r_{ij}} = \vec{r_{ij}}^{(\infty)} \in \mathbb{R}^n, \tag{5.1}$$

where α is a user-specified restart probability and $\vec{e_j}$ is a standard basis vector, and $\vec{r_{ij}}[k]$ is the probability assigned to vertex k given starting state j in network i. For notational simplicity, we denote the RWR diffusion matrix for all vertices in network i as $\mathbf{R_i} \in \mathbb{R}^{n \times n}$.

In the second step, Mashup integrates the M diffusion state matrices with concatenation, with $P = [\mathbf{R_1}, \ldots, \mathbf{R_M}] \in \mathbb{R}^{Mn \times n}$. In the third and final step, Mashup computes the embedding $\vec{z_i} \in \mathbb{R}^d$ for the ith vertex as

$$\mathbf{U\Sigma V} = \texttt{SVD}_{\text{trunc}}\big([\log(\mathbf{R_1}), \ldots, \log(\mathbf{R_M})]\big) \tag{5.2}$$

$$\vec{z_i} = \sqrt{\sigma_i}\vec{v_i}, \tag{5.3}$$

where $\texttt{SVD}_{\text{trunc}}$ indicates the truncated SVD, and σ_i is the ith singular value and v_i is the ith right singular vector of the concatenated log-transformed RWR matrices. To reduce memory usage, Mashup uses the identity that the eigenvectors of $\mathbf{A}^\top\mathbf{A}$ are the same as the right singular vectors of $\mathbf{A}$. Therefore, rather than the $O(Mn^2)$ memory requirement for (5.2), we can instead compute

$$\begin{aligned}\mathbf{Q} &= \big[\log(\mathbf{R_1}), \ldots, \log(\mathbf{R_M})\big]^\top \big[\log(\mathbf{R_1}), \ldots, \log(\mathbf{R_M})\big] \\ &= \sum_{i=1}^{M} \log(\mathbf{R_i})^\top \log(\mathbf{R_i}),\end{aligned} \tag{5.4}$$

which can be summed one network at a time, thereby only requiring $O(n^2)$ memory. Finally, Mashup takes $\vec{z_i} = (\lambda_i)^{1/4}\vec{\nu_i}$, where $\vec{\nu_i}$ is the ith eigenvector of $\mathbf{Q}$ and λ_i is the corresponding eigenvalue, to efficiently compute the ith vertex embedding.

Although this enables fast and scalable integration of large networks, the concatenation of all diffusion state matrices in the second step introduces several important limitations. First, Mashup evenly weights information from all networks during the integration step. When applied to increasingly large number of networks without extensive preprocessing and curation, redundant networks may therefore dominate the downstream embeddings. Second, the networks contained in the input dataset are unlikely to cover the set of possible and reasonable inputs. Therefore, further pooling information from different networks could be useful for further improving downstream vertex embeddings. We aim to address these limitations by refining

the second step with Gemini. Gemini maintains the same first step and third step as Mashup.

5.2.3 Efficient network similarity calculation

Rather than equally combining all networks in the final vertex embedding computation, Gemini weights networks according to their uniqueness, while maintaining efficient memory usage for the large network collection. Mean squared error (MSE) is a widely-used similarity measurement metric. It can therefore be used to quantify the similarity of input networks G_p and G_q based on their diffusion states as

$$\delta_{\text{mse}}(\mathbf{R_p}, \mathbf{R_q}) = \|\mathbf{R_p} - \mathbf{R_q}\|_F^2. \tag{5.5}$$

However, in real-world gene network collections there may be too many large networks to run this computation efficiently. Instead, we use RWR to construct a probability distribution centered around the starting vertex, and then quantify the dispersion to each vertex according to its kurtosis, the fourth standardized moment. Compared to low-order pooling (e.g., mean and variance), kurtosis can better capture the tailedness of the diffusion distribution, which can help quantify both global- and local centrality of each vertex in the network, thereby compactly representing its structure. We hypothesize that high-order pooling yields a good approximation for $\delta_{\text{mse}}(\mathbf{R_p}, \mathbf{R_q})$ in our real-world datasets, and chose a fourth-order kurtosis pooling. Specifically, we approximate $\delta_{\text{mse}}(\mathbf{R_p}, \mathbf{R_q})$ with

$$\delta_{\text{kurt}}(\mathbf{R_p}, \mathbf{R_q}) = \|\texttt{kurt}(\mathbf{R_p}) - \texttt{kurt}(\mathbf{R_q})\|_2^2, \tag{5.6}$$

$$\texttt{kurt}(\mathbf{R_i})[j] = \frac{\frac{1}{n}\sum_{k=1}^{n}\left(\vec{r_{ik}}[j] - \left(\frac{1}{n}\sum_{l=1}^{n}\vec{r_{il}}[j]\right)\right)^4}{\left(\frac{1}{n}\sum_{k=1}^{n}\left(\vec{r_{ik}}[j] - \left(\frac{1}{n}\sum_{l=1}^{n}\vec{r_{il}}[j]\right)\right)^2\right)^2} \tag{5.7}$$

for $j \in \{1, \ldots, n\}$. Using the kurtosis pooling, we then compute network similarity with $\delta_{\text{kurt}}(\cdot)$ in an $O(n)$ space, rather than an $O(n^2)$ space, enabling more network representations to be loaded in memory simultaneously and thereby speeding computation.

5.2.4 Network weighting according to uniqueness

To identify groups of redundant networks, we then run the affinity propagation clustering algorithm [250] on the kurtosis representations. This does not require a prespecified number of clusters, but instead computes the number of clusters C based on the data. Therefore, if the dataset is already uniformly distributed (all networks are sufficiently dissimilar), they will not be clustered together; however, if the input dataset contains redundancies then similar networks, as defined by (5.6), will be more likely to cluster together. This step assigns each network to one of the C clusters computed by the algorithm as

$$\vec{c} \in \{1, \ldots, C\}^M = \texttt{affn\_prop}(\texttt{kurt}(R_1), \ldots, \texttt{kurt}(R_M)). \tag{5.8}$$

Finally, using N_c to denote the number of input networks assigned to cluster c, for all networks $i \in \{1, \ldots, M\}$ we use propensity weighting to define the sampling probability distribution

$$p(i|\vec{c}[i] = c) = \frac{1}{CN_c}. \tag{5.9}$$

5.2.5 Diffusion state sampling with mixup

The second limitation that we sought to address is that the input network collections may be unevenly distributed. To address this, Gemini assumes that the diffusion states jointly parameterize a latent understanding of gene similarity. Specifically, Gemini uses the the sampling distribution in (5.9) as an inverse probability weighting [251]. This leads to instances from different clusters to be sampled equally frequently in expectation. In order to construct a new dataset that can be tuned to the user's computational resources, we then construct new examples using mixup augmentation [252], simulating

$$G_i, G_j \sim p(G|\vec{c}) \tag{5.10}$$

$$\tilde{\mathbf{R}} = \lambda \mathbf{R_i} + (1 - \lambda)\mathbf{R_j} \tag{5.11}$$

for $\lambda \in [0, 1]$ sampled uniformly-at-random. We repeat this process K times, where K is a hyperparameter defined by the user based on their computational resources, to produce the simulated dataset $\tilde{\boldsymbol{\mathcal{R}}}^{(K)} = \{\tilde{\mathbf{R}}_\mathbf{1}, \ldots, \tilde{\mathbf{R}}_\mathbf{K}\}$, which can better represent possible diffusion state matrices for the input data than the

original M networks. Although larger values of K improve the robustness of $\tilde{\boldsymbol{\mathcal{R}}}$, smaller values of K enable users with limited computational resources to efficiently integrate input networks with a reduced network collection size.

Using the simulated dataset $\tilde{\boldsymbol{\mathcal{R}}}^{(K)}$, we then repeat memory-efficient embedding step in (5.4) from Mashup [201] to efficiently compute the vertex representations $\vec{z_j}$ for each vertex j. We linearly normalize $\vec{z_j}[d] \in [0, 1]$ with min-max normalization [253]. As Gemini's simulated, mixed-up network collection more evenly covers the types of evidence included in the input networks, the vertex embeddings also contain a more even sampling of evidence for all downstream tasks.

5.2.6 Application to protein function prediction

Biological network integration has been shown to work well in settings where guilt-by-association [254] applies, meaning that connected nodes in the network demonstrate similar phenotypes of interest [255]. One such setting is protein function prediction, which has been used as an important application to validate network integration approaches [200, 201, 232, 233, 235–238, 241, 244, 245, 248, 249]. We therefore evaluate Gemini's ability to efficiently generate informative embeddings from a heterogeneous set of input networks using a protein function prediction task, classifying each vertex in a network to a subset of G possible functions.

Given the vertex embeddings $\vec{z_i}$, we predict the protein's function $y_i \in [0, 1]^G$ as

$$\vec{y_i} = \sigma(f_\theta(\vec{z_i})) \in \mathbb{R}^G, \tag{5.12}$$

where $\sigma(\cdot)$ represents the sigmoid function. We learn $f_\theta(\cdot)$ on a subset of the n protein embeddings learned by the unsupervised network integration methods, and evaluate the embedding quality based on the remaining proteins.

5.2.7 Network data repositories

We separately consider integration tasks for mouse, human, and yeast datasets to demonstrate Gemini's applicability to different species with different dataset and network sizes.

BioGRID biological networks

We download the processed mouse, human, and yeast BioGRID [46] gene networks from Warde-Farley et al. [256]. Each organism collection contains more than 400 networks (Table A.8). Importantly, the BioGRID repository is *lightly curated*, using experimental vocabularies and text mining to identify networks that should be included, as well as incorporating additional data from model organism databases [257]. However, the inclusion criteria lead to many included networks and there is not subsequent network processing during curation.

STRING biological networks

We also use gene networks from the STRING dataset [45], where edges indicate an association between two genes based on co-expression data, experimental data, or curated datasets. We use the preprocessed networks from Mashup [201], where network edges are weighted in $[0, 1]$ according to the probability of edge presence. Each organism has 6 STRING networks (Table A.8). Although the STRING database only contains 6 networks for each species, they are carefully curated; in comparison, GeneMANIA's BioGRID dataset has many more networks for each species, but they are also more noisy.

GOA protein function annotations

We use the Gene Ontology Annotation (GOA) database [258, 259] for the downstream protein function prediction task. The GOA dataset comprises $G = (16{,}626,\ 21{,}655,\ 8{,}387)$ total protein function labels for mouse, human, and yeast respectively. To further investigate results, we can stratify GO terms according to the three GO sub-ontologies: biological process (BP), molecular function (MF), and cellular component (CC). The size of different sub-ontologies is given in Table A.9.

5.2.8 Comparison approaches

We compare Gemini to the Mashup network integration model [201], which can scale to many hundreds of input networks with reasonable computational power. Importantly, we use the memory-efficient version of Mashup that computes integrated gene embedding features with the eigendecomposition in (5.4), rather than the learnable objective that cannot scale to our datasets. We also compare to BIONIC [202], which uses a

graph attention network to produce integrated gene embeddings, for yeast networks as well as human and mouse networks in the STRING collection. Given the large number of networks and size of each network (Table A.8), BIONIC is unable to scale to the mouse and human BioGRID datasets. Finally, we compare to gene embeddings produced from the network collection's average adjacency matrix, $\bar{\boldsymbol{A}} = \frac{1}{M}\sum_{i=1}^{M} \mathbf{A_i}$. Specifically, we consider the principal component analysis (PCA) [103] decomposition of $\bar{\boldsymbol{A}}$, the truncated SVD of $\bar{\boldsymbol{A}}$, and Mashup random walk smoothing and eigendecomposition applied to $\bar{\boldsymbol{A}}$.

5.2.9 Experimental settings

We use $\alpha = 0.5$ for the restart probability in (5.1) and an embedding dimension $d = 400$ for mouse and human, and $d = 200$ for yeast. During Gemini's clustering stage, we use Affinity Propagation [250] with a damping factor of 0.875, which we found to converge empirically. We train the BIONIC embeddings using the model defaults wherever possible with respect to GPU memory; for other methods, hyperparameters shared by multiple models use the same configurations to facilitate method comparison. Further discussion of method hyperparameters is provided in Section A.3.2. For each species' experiment, we use five-fold cross validation, where 80% of the data is seen during training of each fold. We divide that 80% into a further 80% training and 20% validation set to select the best number of training epochs, use a batch size of 64, and terminate training when the validation loss has not achieved a new global minimum for 50 epochs. We use the same protein function prediction model in (5.12) for all network integration models, where $f_\theta(\cdot)$ is implemented as a multi-layer perceptron (MLP) with two hidden layers, with 200- and 100-hidden units respectively, rectified linear unit (ReLU) activation functions, and trained with cross entropy loss.

We use the BIONIC Github repository[3] for the BIONIC experiments; we re-implemented the Mashup algorithm, and built Gemini around the same framework.

5.2.10 Evaluation metrics

We evaluate the models' embedding quality in terms of their test performance on the downstream protein function prediction task. Since this is an imbalanced classification problem, we consider maximum F_1, which is the F_1 score with the optimal probability threshold for the model, macro-AUPRC, and micro-

[3]Github: `https://github.com/bowang-lab/BIONIC`.

AUPRC.

5.3 Results

5.3.1 Gemini improves downstream protein function prediction on BioGRID

We found that Gemini substantially outperformed the comparison approaches [200, 201] on the BP, MF, and CC sub-categories of the GOA annotations for all three species (Figure 5.3) by integrating hundreds of networks from BioGRID. Gemini has the largest improvement on human, where it has an average test F_1 score of 0.46, macro-AUPRC of 0.10, and micro-AUPRC of 0.45, compared to 0.42, 0.06, and 0.39 for the best-performing baseline respectively. The mouse dataset also has a similar improvement, while the yeast improvement is significant but more modest. Since human and mouse networks are larger than yeast networks, this indicates the importance of weighting networks, especially in large network collections. Furthermore, we find that BIONIC and Mashup both struggle on this BioGRID network integration task; BIONIC's Graph Convolutional Network (GCN) framework may struggle with the relatively dense networks (Table A.8), while Mashup uses the network-specific weight in the SVD computation in (5.2) to model the more unique networks, rather than incorporating the information in the gene vertex embeddings. Gemini addresses these challenges by more evenly sampling different types and combinations of networks, thereby incorporating rare network information into the vertex embeddings as well as the network embeddings.

5.3.2 Gemini effectively integrates networks from BioGRID and STRING

We next evaluated Gemini's performance on integrating all networks from BioGRID and STRING. We repeated the protein function classification task using the GOA database, comparing the quality of the embeddings learned from the STRING network collection, the BioGRID network collection, and the union of the STRING and BioGRID network collections. In such a setting, the union of the two network collections is a superset of the individual collection, and we therefore expect it to contain more information than either collection alone without removing any of the information. We would therefore hope that a network integration method would perform better on the combined network collections than either of the collections alone.

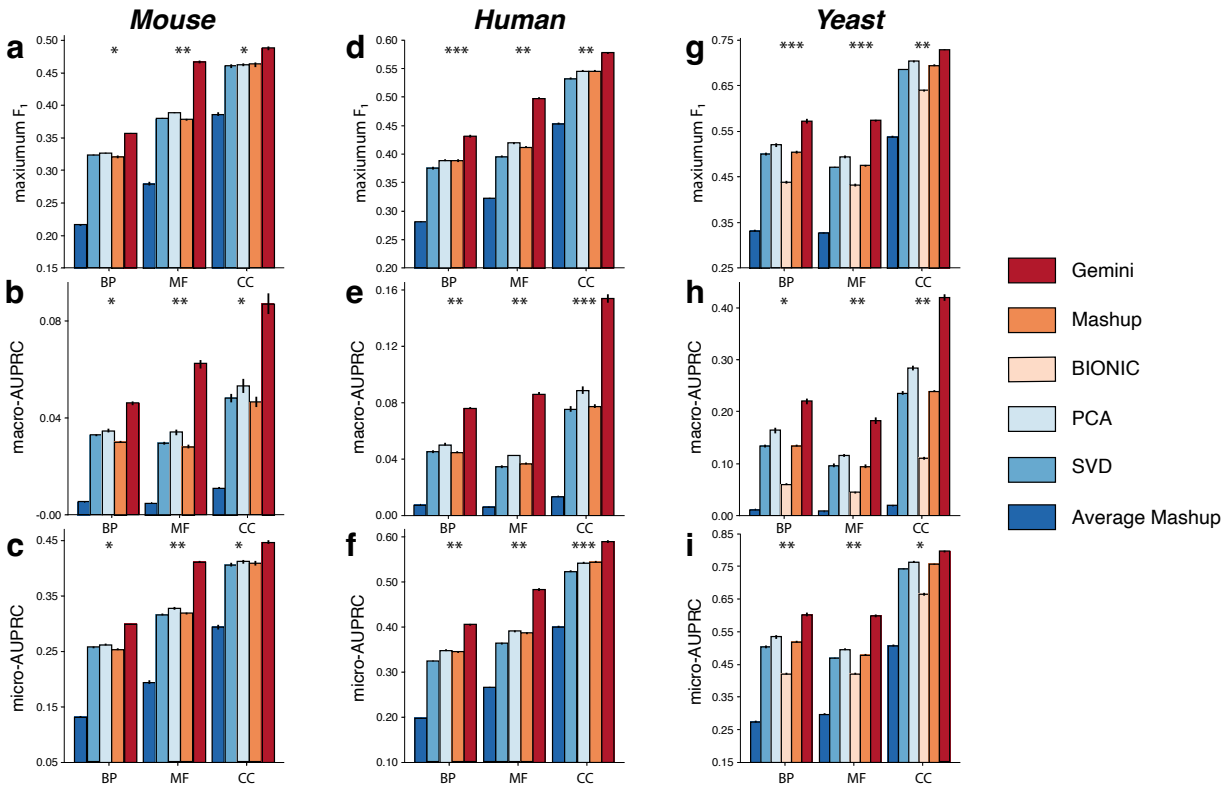


Figure 5.3: **Performance of protein function prediction by integrating networks from BioGRID, stratified by GO sub-ontology. a-i**, Barplot of model performance for protein function prediction task, divided into the BP, MF, and CC sub-ontologies of GOA. Comparison of Mashup, BIONIC, Mashup on $\bar{\boldsymbol{A}}$, PCA on $\bar{\boldsymbol{A}}$, SVD on $\bar{\boldsymbol{A}}$, and Gemini on mouse (**a-c**), human (**d-f**), and yeast (**g-i**) species. We consider F_1 score (**a,d,g**), macro-AUPRC (**b,e,h**), and micro-AUPRC (**c,f,i**), where higher values indicate better performance and error bars indicate standard error over five test folds; one-sided paired t-test comparison with Mashup is also shown ($*$ indicates p-value $< 5\mathrm{e}{-4}$, $**$ indicates p-value $< 5\mathrm{e}{-5}$, and $***$ indicates p-value $< 5\mathrm{e}{-6}$).

We find that Gemini achieves peak performance on the combined dataset, indicating that it effectively makes use of all networks from two different sources (Figures 5.4) and A.17). In particular, Gemini has a maximum F_1 score of 0.62 on the combined yeast dataset, compared to 0.55 for Mashup. Furthermore, this improves compared to both Gemini's F_1 score of 0.56 on the STRING and 0.61 on the BioGRID databases individually. As an important comparison point, we find that Mashup consistently performs best when embeddings are learned only from the STRING networks. We hypothesize that the curated STRING networks are all sufficiently diverse and high-quality that Gemini's high-order pooling for network uniqueness and subsequent network-mixup are not helpful in this setting. However, Mashup's downstream protein function annotation quality degrades when the BioGRID networks are also included. Specifically, while

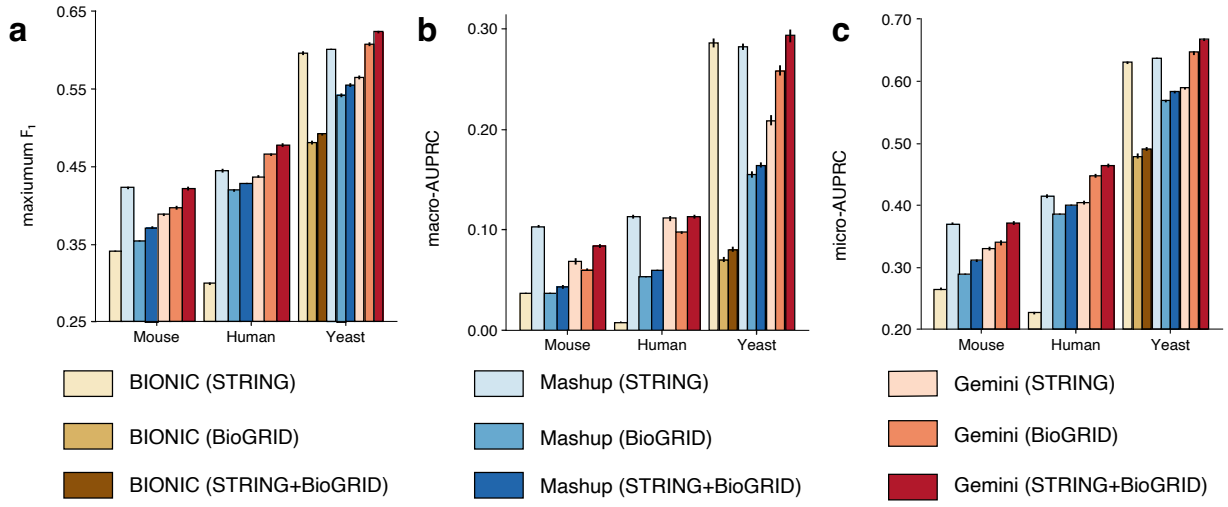


Figure 5.4: **Downstream protein function prediction performance comparison for network integration with the BioGRID and STRING collections. a-c**, Barplots comparing Mashup, BIONIC, and Gemini network integration quality on the STRING, BioGRID, and union of STRING and BioGRID network collections, evaluated based on the average performance of the downstream protein function prediction task on GOA. BIONIC does not have BioGRID or STRING+BioGRID results for mouse or human due to GPU memory constraints. Results for mouse, human, and yeast experiments are shown. Test set performance is measured with the maximum F_1 score (**a**), macro-AUPRC (**b**), and micro-AUPRC (**c**). Higher values indicate better performance; errors bars indicate the standard error over five test folds.

Gemini's macro-AUPRC performance improves by more than 40% when yeast BioGRID networks augment the STRING input (one-sided t-test p-value $< 5\text{e}{-6}$ for improvement over STRING alone, p-value $< 5\text{e}{-3}$ for improvement over BioGRID alone), Mashup's performance decreases by 71% (one-sided t-test p-value $< 5\text{e}{-9}$ compared to STRING alone). Furthermore, Gemini's STRING+BioGRID integrated gene embeddings outperform Mashup's STRING integrated embeddings, indicating the downstream benefit of additional, if noisy, networks. Gemini's strong performance when integrating hundreds of networks from many sources demonstrates Gemini's wide applicability for integrating the accumulating and continually-generated biological networks.

5.3.3 Kurtosis similarity reflects biological similarity

Finally, we sought to further understand the promising performance of Gemini by validating its hypotheses and network representations. Gemini is based on the belief that similar networks reflect biological similarity. We considered the t-Stochastic Neighbor Embedding (t-SNE) [260] of the human BioGRID network's

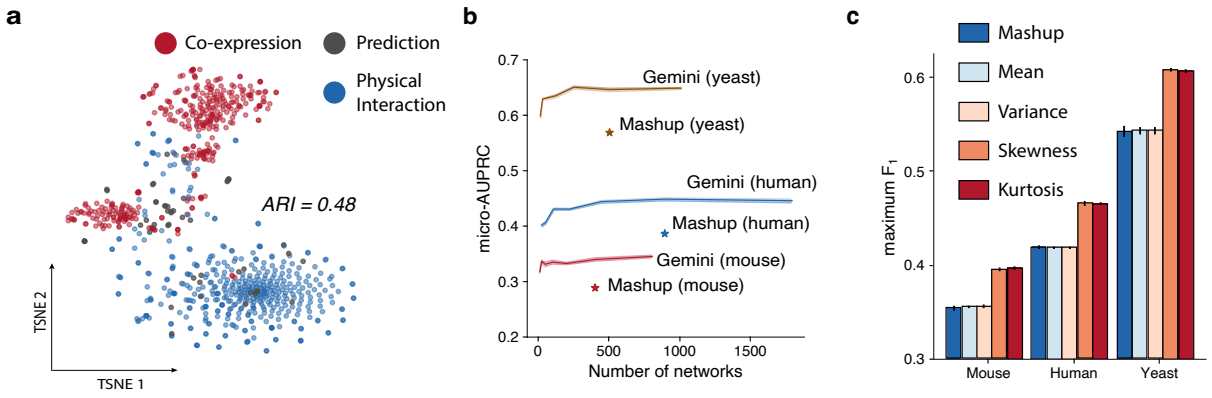


Figure 5.5: **Gemini feature analysis on BioGRID networks**. **a**, t-SNE embeddings of kurtosis vector representations for the BioGRID human dataset, colored by the type of biological evidence used to construct that network, with an adjusted Rand index (ARI) of 0.48. **b**, The average macro-AUPRC (higher is better) with increasing numbers of mixed-up networks K for each species across 5 random seeds, compared to Mashup on the BioGRID dataset. **c**, Barplot of downstream protein prediction F_1 score (higher is better) for mean-, variance-, skewness-, and kurtosis-based network similarity approximations, with Mashup included as a reference, on the BioGRID networks. Error bars indicate standard error over five test folds.

kurtosis representations, and compared these to the evidence type encoded in the GeneMANIA filenames [237, 256]. We found that the t-SNE space largely clustered according to the type of network (Figure 5.5a). We found that the affinity-propagation-based clustering of kurtosis vectors had an adjusted Rand index (ARI) [261] of 0.48 for the human dataset, relative to a random cluster assignment ARI of 0, indicating that similar kurtosis representations corresponded to similar biological evidence. Importantly, Gemini was able to automatically detect these similarities, both enabling networks with common patterns from different types of biological evidence to be grouped together and automatically separating different clusters of networks that may not be manually annotated with biologically-relevant subtype labels.

We also evaluated the impact of the size K of the synthetic, mixed-up dataset of diffusion state matrices that Gemini generates as an intermediate step on the downstream protein function prediction performance. We expected that larger dataset sizes would improve downstream performance, but that smaller values of K would enable users with smaller computational budgets to compute vertex embeddings. If necessary, we can set $K < M$, reducing the number of networks in the SVD computation compared to the input dataset, and making Gemini run potentially faster than Mashup depending on the computational resources available. We varied K from 0.03125 to $2M$ and found that, as expected, larger values of K lead to better downstream

performance for all species (Figure 5.5b), although smaller values of K can still perform relatively well in resource-limited settings. For instance, the Gemini model with $K = 0.03125M$ (28 networks) on the human dataset has an F_1 score 89% of the model trained with $K = M$, and this increases to 95% with $K = 0.0625M$ (56 networks).

Finally, we analyzed the choice of kurtosis pooling by approximating network similarity using different moments. In addition to kurtosis, we compared protein prediction performance based on network similarity approximations using the first moment, with the mean of the diffusion distribution $\vec{r_{ik}}[j]$; the second central moment, with the variance of $\vec{r_{ik}}[j]$; and the third standardized moment, with the skew of $\vec{r_{ik}}[j]$. We found that kurtosis and skewness, the fourth and third standardized moments, consistently performed best on all species (Figure 5.5c), and kurtosis featurization significantly outperforms all other representations (one-sided paired t-test with Bonferroni correction, p-value $< 1e{-}17, 1e{-}13, 1e{-}7$ for mean, variance, and skewness respectively). This validates our intuition that higher-order pooling can better capture both global and local network structure, compared to low-order pooling measures like mean and variance.

5.4 Discussion

In this chapter I have presented Gemini [28], a memory-efficient network integration method for large-scale and heterogeneous biological networks. We construct a kurtosis-based pooling of the diffusion state to cover different types and combinations of biological evidence, and use this vector to effectively weight each network. Although we address the under- and over-representation of different types of evidence, Gemini considers cluster membership, and therefore similarity, at the network-level. Future work could extend this notion to a vertex-based similarity, where two networks could be considered similar for some genes and different for others. Furthermore, as Gemini will weight unique networks more heavily than repetitive networks, outlier detection methods based on Gemini's kurtosis-pooled network representations could be employed to remove particularly noisy or adversarial networks from consideration. Finally, future rigorous comparisons to supervised baselines across a variety of downstream tasks would further demarcate the benefit that Gemini and other unsupervised models can convey for gene network integration.

Method's like Gemini therefore provide some path towards *approximate automatic curation*. By learning to identify and balance different types of information and relationships in a very large and heterogeneous

dataset, Gemini automatically performs a form a curation at very low cost (theoretical and practical computational costs analysis is provided in Section A.3.4). Such tools can also be incorporated into the curation process itself; for instance, the "experimental" network incorporated in the STRING [45] network database is itself based on data from sources including BioGRID [46]. Future curation pipelines could incorporate advice from approximate automatic curation methods like Gemini to serve as a starting point for network integration or to sift through massive amounts of input data, thereby reducing the expert's workload. Gemini can therefore reduce curation costs while maintaining high accuracy in the network integration setting, and suggests possibilities for future work in approximate automatic curation.

Chapter 6

Conclusion

Over the course of this thesis, I have shown how computational methods can simulate high-quality datapoints from low-cost inputs. The three works I discuss here have spanned many biological data types and applications, and acknowledged the varied challenges presented in each area. However, all the biological experiments presented here have been fundamentally limited by the high costs associated with obtaining high-quality biological data. I hope to have shown the reader that these high costs are inherently prohibitive to conducting all of the high-quality experiments that scientists are keen to study, but also convinced them that *in silico* data generation is well poised to assist in biological hypothesis generation and experimental prioritization.

In Chapter 3, I presented the Sagittarius model [26] for temporal and combinatorial extrapolation in genomic time series. Sagittarius mitigates costs both stemming from long-running experimental studies and the untestably large number of possible experimental combinations to measure. I also showed not only that Sagittarius's simulations are accurate, obtaining low error and high correlation with held-out test measurements, but also that these simulated datapoints demonstrate sizeable potential for biological hypothesis generation. Sagittarius was able to extrapolate datapoints that reflected our understanding of organogenesis and senescence, aligning with other, more expensive datasets to specifically study those conditions, even though Sagittarius was never exposed to these data. To further highlight its practical applications as a biological hypothesis generation tool, Sagittarius identified drug repurposing opportunities from untested drug and cell line applications, several of which were supported in the literature. Finally, I demonstrated the flexi-

bility of Sagittarius's framework by applying the model to cancer patient somatic mutation profiles, grouped by cancer type in a time-series formulation. I was then able to use Sagittarius to simulate tumorigenesis, which would require hugely expensive, large-scale longitudinal studies to conduct *in vivo*.

I then moved on to the Capricorn model [27] in Chapter 4, focusing on contact matrix resolution enhancement to simulate high-quality measurements from lower-cost, lower-quality datapoints. Given datapoints from a small number of high-cost, high-quality experiments were were able to train a model that can enhance lower-coverage measurements during inference. Importantly, I again demonstrated that the *in silico* contact matrices were useful for identifying biologically relevant patterns. Here, I focused on whether Capricorn correctly identified and enhanced small-scale structural features from the low-coverage input data, and found that the model was able to identify chromatin loops found in their high-coverage counterparts that could not be directly identified from the low-coverage data with existing loop calling tools, and that these loops were further supported by the DNA sequence and other experimental assays.

Finally, I discussed the Gemini model [28] in Chapter 5. Here, I broadened the view on experimental costs of biological data to include expert curation, specifically focusing on gene network repositories. Gemini balances a largely ideal but very costly solution of total expert curation with a very noisy but cheap solution of minimal-to-no curation, and proposes a notion of automatic approximate curation instead. I show that Gemini's method of identifying unique source of information in a large, imbalanced, and under-annotated set of networks also led to improved gene representations for downstream hypothesis generation tasks like protein function prediction.

I emphasize that none of my findings suggest that experimental data generation and validation are unimportant or unnecessary; on the contrary, I believe that they are the most critical components of the biological analysis pipeline. In an ideal world, there would be no need or use case for the methods developed here. However, these critical components come with high associated costs, and a pragmatist realizes that not every experiment can be run. I view this as one of the most exciting avenues for *in silico* biological generation: a guidepost towards which experimental roads might be the most interesting or beneficial to investigate.

These avenues point to other biological and medical applications that could benefit greatly from *in silico* generation approaches. For instance, high-cost medical application parameters can be optimized using low-cost simulated datasets that make use of already-available resources [262]. Future work can also focus

on multimodal profile integration, such as better representing a patient's rare disease state by capturing available information across clinical imaging, lab tests, and electronic health record data, or by combining different types of disease models (e.g., cell line, organoid, xenograft) to holistically analyze human health and disease.

Furthermore, additional lines of research should improve the robustness of the loop between computational data synthesis and hypothesis generation by investigating which experimentally-generated datapoints would be most useful for future work. By combining principles from active learning [263], mutual information estimation [264] and datapoint influence functions [265] with economic cost-benefit analyses [266], computational scientists could request the most beneficial *in vitro* or *in vivo* data samples to strengthen future *in silico* predictions, thereby tightening and iterating on the biological hypothesis generation loop.

Bibliography

[1] Siang Yong Tan and Yvonne Tatsumura. Alexander fleming (1881-1955): Discoverer of penicillin. *Singapore Med. J.*, 56(7):366–367, July 2015.

[2] Henrietta lacks: science must right a historical wrong. *Nature*, 585(7823):7, September 2020.

[3] Significant research advances enabled by HeLa cells. `https://osp.od.nih.gov/hela-cells/significant-research-advances-enabled-by-hela-cells/`, August 2022. Accessed: 2024-3-20.

[4] C W Young and S Hodas. HYDROXYUREA: INHIBITORY EFFECT ON DNA METABOLISM. *Science*, 146(3648):1172–1174, November 1964.

[5] Takumi Ito, Hideki Ando, Takayuki Suzuki, Toshihiko Ogura, Kentaro Hotta, Yoshimasa Imamura, Yuki Yamaguchi, and Hiroshi Handa. Identification of a primary target of thalidomide teratogenicity. *Science*, 327(5971):1345–1350, March 2010.

[6] T T Puck, P I Marcus, and S J Cieciura. Clonal growth of mammalian cells in vitro; growth characteristics of colonies from single HeLa cells with and without a feeder layer. *J. Exp. Med.*, 103(2): 273–283, February 1956.

[7] P A Furman, J A Fyfe, M H St Clair, K Weinhold, J L Rideout, G A Freeman, S N Lehrman, D P Bolognesi, S Broder, and H Mitsuya. Phosphorylation of 3'-azido-3'-deoxythymidine and selective interaction of the 5'-triphosphate with human immunodeficiency virus reverse transcriptase. *Proc. Natl. Acad. Sci. U. S. A.*, 83(21):8333–8337, November 1986.

[8] M A Fischl, D D Richman, M H Grieco, M S Gottlieb, P A Volberding, O L Laskin, J M Leedom, J E Groopman, D Mildvan, and R T Schooley. The efficacy of azidothymidine (AZT) in the treatment of patients with AIDS and AIDS-related complex. a double-blind, placebo-controlled trial. *N. Engl. J. Med.*, 317(4):185–191, July 1987.

[9] Alexander Schuhmacher, Markus Hinder, Alexander von Stegmann Und Stein, Dominik Hartl, and Oliver Gassmann. Analysis of pharma R&D productivity - a new perspective needed. *Drug Discov. Today*, 28(10):103726, October 2023.

[10] J M Hall, M K Lee, B Newman, J E Morrow, L A Anderson, B Huey, and M C King. Linkage of early-onset familial breast cancer to chromosome 17q21. *Science*, 250(4988):1684–1689, December 1990.

[11] Suhas S P Rao, Miriam H Huntley, Neva C Durand, Elena K Stamenova, Ivan D Bochkov, James T Robinson, Adrian L Sanborn, Ido Machol, Arina D Omer, Eric S Lander, and Erez Lieberman Aiden. A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell*, 159(7):1665–1680, December 2014.

[12] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. March 2023.

[13] Gemini Team et al. Gemini: A family of highly capable multimodal models. December 2023.

[14] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional image generation with CLIP latents. April 2022.

[15] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. DNABERT: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics*, 37 (15):2112–2120, February 2021.

[16] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. DNABERT-2: Efficient foundation model and benchmark for Multi-Species genome. June 2023.

[17] Gonzalo Benegas, Sanjit Singh Batra, and Yun S Song. DNA language models are powerful predictors of genome-wide variant effects. *Proc. Natl. Acad. Sci. U. S. A.*, 120(44):e2311219120, October 2023.

[18] Nadav Brandes, Grant Goldman, Charlotte H Wang, Chun Jimmie Ye, and Vasilis Ntranos. Genome-wide prediction of disease variant effects with a deep protein language model. *Nat. Genet.*, 55(9): 1512–1522, September 2023.

[19] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan Dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, March 2023.

[20] Jinwoo Leem, Laura S Mitchell, James H R Farmery, Justin Barton, and Jacob D Galson. Deciphering the language of antibodies using self-supervised learning. *Patterns (N Y)*, 3(7):100513, July 2022.

[21] Richard J Chen, Chengkuan Chen, Yicong Li, Tiffany Y Chen, Andrew D Trister, Rahul G Krishnan, and Faisal Mahmood. Scaling vision transformers to gigapixel images via hierarchical self-supervised learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022.

[22] Wisdom O Ikezogwo, M S Seyfioglu, Fatemeh Ghezloo, Dylan Stefan Chan Geva, Fatwir Sheikh Mohammed, Pavan Kumar Anand, Ranjay Krishna, and Linda G Shapiro. Quilt-1M: One million image-text pairs for histopathology. *ArXiv*, abs/2306.11207, June 2023.

[23] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[24] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2672–2680, Cambridge, MA, USA, December 2014. MIT Press.

[25] Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R Bowman. Studying large language model generalization with influence functions. August 2023.

[26] Addie Woicik, Mingxin Zhang, Janelle Chan, Jianzhu Ma, and Sheng Wang. Extrapolating heterogeneous time-series gene expression data using sagittarius. *Nature Machine Intelligence*, 5(7):699–713, June 2023.

[27] Tangqi Fang, Yifeng Liu, Addie Woicik, Minsi Lu, Anupama Jha, Xiao Wang, Gang Li, Borislav Hristov, Zixuan Liu, Hanwen Xu, William S Noble, and Sheng Wang. Enhancing Hi-C contact matrices for loop detection with capricorn, a multi-view diffusion model. *Bioinformatics*, 40(40 Suppl 1), June 2024.

[28] Addie Woicik, Mingxin Zhang, Hanwen Xu, Sara Mostafavi, and Sheng Wang. Gemini: memory-efficient integration of hundreds of gene networks with high-order pooling. *Bioinformatics*, 39(39 Suppl 1):i504–i512, June 2023.

[29] Zhong Wang, Mark Gerstein, and Michael Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.*, 10(1):57–63, January 2009.

[30] Meenakshi S Kagda, Bonita Lam, Casey Litton, Corinn Small, Cricket A Sloan, Emma Spragins, Forrest Tanaka, Ian Whaling, Idan Gabdank, Ingrid Youngworth, J Seth Strattan, Jason Hilton, Jennifer Jou, Jessica Au, Jin-Wook Lee, Kalina Andreeva, Keenan Graham, Khine Lin, Matt Simison, Otto Jolanki, Paul Sud, Pedro Assis, Philip Adenekan, Eric Douglas, Mingjie Li, Pedro Assis, Keenan Graham, Paul Sud, Stuart Miyasato, Weiwei Zhong, Yunhai Luo, Zachary Myers, J Michael Cherry, and Benjamin C Hitz. Data navigation on the ENCODE portal. April 2023.

[31] Allison Piovesan, Francesca Antonaros, Lorenza Vitale, Pierluigi Strippoli, Maria Chiara Pelleri, and Maria Caracausi. Human protein-coding genes and gene feature statistics in 2019. *BMC Res. Notes*, 12(1):315, June 2019.

[32] Ivana Bozic, Tibor Antal, Hisashi Ohtsuki, Hannah Carter, Dewey Kim, Sining Chen, Rachel Karchin, Kenneth W Kinzler, Bert Vogelstein, and Martin A Nowak. Accumulation of driver and passenger mutations during tumor progression. *Proc. Natl. Acad. Sci. U. S. A.*, 107(43):18545–18550, October 2010.

[33] Alison Sinclair. Genetics 101: detecting mutations in human genes. *CMAJ*, 167(3):275–279, August 2002.

[34] Michael R Waarts, Aaron J Stonestrom, Young C Park, and Ross L Levine. Targeting mutations in cancer. *J. Clin. Invest.*, 132(8), April 2022.

[35] Zhijun Duan, Mirela Andronescu, Kevin Schutz, Choli Lee, Jay Shendure, Stanley Fields, William S Noble, and C Anthony Blau. A genome-wide 3c-method for characterizing the three-dimensional architectures of genomes. *Methods*, 58(3):277–288, November 2012.

[36] Jinlei Han, Zhiliang Zhang, and Kai Wang. 3C and 3c-based techniques: the powerful tools for spatial genome organization deciphering. *Mol. Cytogenet.*, 11:21, March 2018.

[37] Job Dekker, Karsten Rippe, Martijn Dekker, and Nancy Kleckner. Capturing chromosome conformation. *Science*, 295(5558):1306–1311, February 2002.

[38] Marieke Simonis, Petra Klous, Erik Splinter, Yuri Moshkin, Rob Willemsen, Elzo de Wit, Bas van Steensel, and Wouter de Laat. Nuclear organization of active and inactive chromatin domains uncovered by chromosome conformation capture-on-chip (4c). *Nat. Genet.*, 38(11):1348–1354, November 2006.

[39] Josée Dostie, Todd A Richmond, Ramy A Arnaout, Rebecca R Selzer, William L Lee, Tracey A Honan, Eric D Rubio, Anton Krumm, Justin Lamb, Chad Nusbaum, Roland D Green, and Job Dekker. Chromosome conformation capture carbon copy (5c): a massively parallel solution for mapping interactions between genomic elements. *Genome Res.*, 16(10):1299–1309, October 2006.

[40] Melissa J Fullwood, Mei Hui Liu, You Fu Pan, Jun Liu, Han Xu, Yusoff Bin Mohamed, Yuriy L Orlov, Stoyan Velkov, Andrea Ho, Poh Huay Mei, Elaine G Y Chew, Phillips Yao Hui Huang, Willem-Jan

Welboren, Yuyuan Han, Hong Sain Ooi, Pramila N Ariyaratne, Vinsensius B Vega, Yanquan Luo, Peck Yean Tan, Pei Ye Choy, K D Senali Abayratna Wansa, Bing Zhao, Kar Sian Lim, Shi Chi Leow, Jit Sin Yow, Roy Joseph, Haixia Li, Kartiki V Desai, Jane S Thomsen, Yew Kok Lee, R Krishna Murthy Karuturi, Thoreau Herve, Guillaume Bourque, Hendrik G Stunnenberg, Xiaoan Ruan, Valere Cacheux-Rataboul, Wing-Kin Sung, Edison T Liu, Chia-Lin Wei, Edwin Cheung, and Yijun Ruan. An oestrogen-receptor-alpha-bound human chromatin interactome. *Nature*, 462(7269):58–64, November 2009.

[41] Erez Lieberman-Aiden, Nynke L van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R Lajoie, Peter J Sabo, Michael O Dorschner, Richard Sandstrom, Bradley Bernstein, M A Bender, Mark Groudine, Andreas Gnirke, John Stamatoyannopoulos, Leonid A Mirny, Eric S Lander, and Job Dekker. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, October 2009.

[42] Tsung-Han S Hsieh, Assaf Weiner, Bryan Lajoie, Job Dekker, Nir Friedman, and Oliver J Rando. Mapping nucleosome resolution chromosome folding in yeast by Micro-C. *Cell*, 162(1):108–119, July 2015.

[43] Sarah B Reiff, Andrew J Schroeder, Koray Kırlı, Andrea Cosolo, Clara Bakker, Luisa Mercado, Soohyun Lee, Alexander D Veit, Alexander K Balashov, Carl Vitzthum, William Ronchetti, Kent M Pitman, Jeremy Johnson, Shannon R Ehmsen, Peter Kerpedjiev, Nezar Abdennur, Maxim Imakaev, Serkan Utku Öztürk, Uğur Çamoğlu, Leonid A Mirny, Nils Gehlenborg, Burak H Alver, and Peter J Park. The 4D nucleome data portal as a resource for searching and visualizing curated nucleomics data. *Nat. Commun.*, 13(1):2365, May 2022.

[44] Houda Belaghzal, Job Dekker, and Johan H Gibcus. Hi-C 2.0: An optimized Hi-C procedure for high-resolution genome-wide mapping of chromosome conformation. *Methods*, 123:56–65, July 2017.

[45] Damian Szklarczyk, Annika L Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T Doncheva, John H Morris, Peer Bork, Lars J Jensen, and Christian von Mering. STRING v11: protein-protein association networks with increased coverage,

supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.*, 47(D1): D607–D613, January 2019.

[46] Rose Oughtred et al. The BioGRID database: A comprehensive biomedical resource of curated protein, genetic, and chemical interactions. *Protein Sci.*, 30(1):187–200, January 2021.

[47] Vaswani, Shazeer, Parmar, and others. Attention is all you need. *Adv. Neural Inf. Process. Syst.*, 2017.

[48] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[49] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 2015. PMLR.

[50] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H Larochelle, M Ranzato, R Hadsell, M F Balcan, and H Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[51] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution image synthesis with latent diffusion models. *arXiv [cs.CV]*, 2021.

[52] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image diffusion models with deep language understanding. In Alice H Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[53] Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, David L Lahr, Jodi E Hirschman,

Zihan Liu, Melanie Donahue, Bina Julian, Mariya Khan, David Wadden, Ian C Smith, Daniel Lam, Arthur Liberzon, Courtney Toder, Mukta Bagul, Marek Orzechowski, Oana M Enache, Federica Piccioni, Sarah A Johnson, Nicholas J Lyons, Alice H Berger, Alykhan F Shamji, Angela N Brooks, Anita Vrcic, Corey Flynn, Jacqueline Rosains, David Y Takeda, Roger Hu, Desiree Davison, Justin Lamb, Kristin Ardlie, Larson Hogstrom, Peyton Greenside, Nathanael S Gray, Paul A Clemons, Serena Silver, Xiaoyun Wu, Wen-Ning Zhao, Willis Read-Button, Xiaohua Wu, Stephen J Haggarty, Lucienne V Ronco, Jesse S Boehm, Stuart L Schreiber, John G Doench, Joshua A Bittker, David E Root, Bang Wong, and Todd R Golub. A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles. *Cell*, 171(6):1437–1452.e17, November 2017.

[54] A Einstein. Die grundlage der allgemeinen relativitätstheorie. *Ann. Phys.*, 354(7):769–822, January 1916.

[55] Margarida Cardoso-Moreira, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, Kelly Ascenção, Coralie Rummel, Svetlana Ovchinnikova, Pavel V Mazin, Ioannis Xenarios, Keith Harshman, Matthew Mort, David N Cooper, Carmen Sandi, Michael J Soares, Paula G Ferreira, Sandra Afonso, Miguel Carneiro, James M A Turner, John L VandeBerg, Amir Fallahshahroudi, Per Jensen, Rüdiger Behr, Steven Lisgo, Susan Lindsay, Philipp Khaitovich, Wolfgang Huber, Julie Baker, Simon Anders, Yong E Zhang, and Henrik Kaessmann. Gene expression across mammalian organ development. *Nature*, 571(7766):505–509, July 2019.

[56] Cancer Genome Atlas Research Network, John N Weinstein, Eric A Collisson, Gordon B Mills, Kenna R Mills Shaw, Brad A Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, and Joshua M Stuart. The cancer genome atlas Pan-Cancer analysis project. *Nat. Genet.*, 45(10):1113–1120, October 2013.

[57] Gunsagar S Gulati, Shaheen S Sikandar, Daniel J Wesche, Anoop Manjunath, Anjan Bharadwaj, Mark J Berger, Francisco Ilagan, Angera H Kuo, Robert W Hsieh, Shang Cai, Maider Zabala, Ferenc A Scheeren, Neethan A Lobo, Dalong Qian, Feiqiao B Yu, Frederick M Dirbas, Michael F Clarke, and Aaron M Newman. Single-cell transcriptional diversity is a hallmark of developmental potential. *Science*, 367(6476):405–411, January 2020.

[58] Michelle N Arbeitman, Eileen E M Furlong, Farhad Imam, Eric Johnson, Brian H Null, Bruce S Baker, Mark A Krasnow, Matthew P Scott, Ronald W Davis, and Kevin P White. Gene expression during the life cycle of drosophila melanogaster. *Science*, 297(5590):2270–2275, September 2002.

[59] Liangtao Zheng, Shishang Qin, Wen Si, Anqiang Wang, Baocai Xing, Ranran Gao, Xianwen Ren, Li Wang, Xiaojiang Wu, Ji Zhang, Nan Wu, Ning Zhang, Hong Zheng, Hanqiang Ouyang, Keyuan Chen, Zhaode Bu, Xueda Hu, Jiafu Ji, and Zemin Zhang. Pan-cancer single-cell landscape of tumor-infiltrating T cells. *Science*, 374(6574):abe6474, December 2021.

[60] Allon M Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A Weitz, and Marc W Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, May 2015.

[61] Jeong Seok Lee, June-Young Koh, Kijong Yi, Young-Il Kim, Su-Jin Park, Eun-Ha Kim, Se-Mi Kim, Sung Ho Park, Young Seok Ju, Young Ki Choi, and Su-Hyung Park. Single-cell transcriptome of bronchoalveolar lavage fluid reveals sequential change of macrophages during SARS-CoV-2 infection in ferrets. *Nat. Commun.*, 12(1):4567, July 2021.

[62] Roser Vento-Tormo, Mirjana Efremova, Rachel A Botting, Margherita Y Turco, Miquel Vento-Tormo, Kerstin B Meyer, Jong-Eun Park, Emily Stephenson, Krzysztof Polański, Angela Goncalves, Lucy Gardner, Staffan Holmqvist, Johan Henriksson, Angela Zou, Andrew M Sharkey, Ben Millar, Barbara Innes, Laura Wood, Anna Wilbrey-Clark, Rebecca P Payne, Martin A Ivarsson, Steve Lisgo, Andrew Filby, David H Rowitch, Judith N Bulmer, Gavin J Wright, Michael J T Stubbington, Muzlifah Haniffa, Ashley Moffett, and Sarah A Teichmann. Single-cell reconstruction of the early maternal-fetal interface in humans. *Nature*, 563(7731):347–353, November 2018.

[63] Gilad Almogy, Mark Pratt, Florian Oberstrass, Linda Lee, Dan Mazur, Nate Beckett, Omer Barad, Ilya Soifer, Eddie Perelman, Yoav Etzioni, Martin Sosa, April Jung, Tyson Clark, Eliane Trepagnier, Gila Lithwick-Yanai, Sarah Pollock, Gil Hornung, Maya Levy, Matthew Coole, Tom Howd, Megan Shand, Yossi Farjoun, James Emery, Giles Hall, Samuel Lee, Takuto Sato, Ricky Magner, Sophie Low, Andrew Bernier, Bharathi Gandi, Jack Stohlman, Corey Nolet, Siobhan Donovan, Brendan Blumenstiel, Michelle Cipicchio, Sheila Dodge, Eric Banks, Niall Lennon, Stacey Gabriel, and Doron

Lipson. Cost-efficient whole genome-sequencing using novel mostly natural sequencing-by-synthesis chemistry and open fluidics platform. June 2022.

[64] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, Kaiqin Lao, and M Azim Surani. mRNA-Seq whole-transcriptome analysis of a single cell. *Nat. Methods*, 6(5):377–382, May 2009.

[65] Daniel Ramsköld, Shujun Luo, Yu-Chieh Wang, Robin Li, Qiaolin Deng, Omid R Faridani, Gregory A Daniels, Irina Khrebtukova, Jeanne F Loring, Louise C Laurent, Gary P Schroth, and Rickard Sandberg. Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells. *Nat. Biotechnol.*, 30(8):777–782, August 2012.

[66] Junyue Cao, Wei Zhou, Frank Steemers, Cole Trapnell, and Jay Shendure. Sci-fate characterizes the dynamics of gene expression in single cells. *Nat. Biotechnol.*, 38(8):980–988, August 2020.

[67] Tabula Muris Consortium. A single-cell transcriptomic atlas characterizes ageing tissues in the mouse. *Nature*, 583(7817):590–595, July 2020.

[68] Nicholas Schaum, Benoit Lehallier, Oliver Hahn, Róbert Pálovics, Shayan Hosseinzadeh, Song E Lee, Rene Sit, Davis P Lee, Patricia Morán Losada, Macy E Zardeneta, Tobias Fehlmann, James T Webber, Aaron McGeever, Kruti Calcuttawala, Hui Zhang, Daniela Berdnik, Vidhu Mathur, Weilun Tan, Alexander Zee, Michelle Tan, Tabula Muris Consortium, Angela Oliveira Pisco, Jim Karkanias, Norma F Neff, Andreas Keller, Spyros Darmanis, Stephen R Quake, and Tony Wyss-Coray. Ageing hallmarks exhibit organ-specific temporal signatures. *Nature*, 583(7817):596–602, July 2020.

[69] Wanxin Wang, Felipe Vilella, Pilar Alama, Inmaculada Moreno, Marco Mignardi, Alina Isakova, Wenying Pan, Carlos Simon, and Stephen R Quake. Single-cell transcriptomic atlas of the human endometrium during the menstrual cycle. *Nat. Med.*, 26(10):1644–1653, October 2020.

[70] Susan M Sunkin, Lydia Ng, Chris Lau, Tim Dolbeare, Terri L Gilbert, Carol L Thompson, Michael Hawrylycz, and Chinh Dang. Allen brain atlas: an integrated spatio-temporal portal for exploring the central nervous system. *Nucleic Acids Res.*, 41(Database issue):D996–D1008, January 2013.

[71] A Radovic, Jiawei He, J Ramanan, Marcus A Brubaker, and Andreas M Lehrmann. Agent forecasting at flexible horizons using ODE flows. *ICML*, 2021.

[72] Guangdun Peng, Guizhong Cui, Jincan Ke, and Naihe Jing. Using Single-Cell and spatial transcriptomes to understand stem cell lineage specification during early embryo development. *Annu. Rev. Genomics Hum. Genet.*, 21:163–181, August 2020.

[73] Muzlifah Haniffa, Deanne Taylor, Sten Linnarsson, Bruce J Aronow, Gary D Bader, Roger A Barker, Pablo G Camara, J Gray Camp, Alain Chédotal, Andrew Copp, Heather C Etchevers, Paolo Giacobini, Berthold Göttgens, Guoji Guo, Ania Hupalowska, Kylie R James, Emily Kirby, Arnold Kriegstein, Joakim Lundeberg, John C Marioni, Kerstin B Meyer, Kathy K Niakan, Mats Nilsson, Bayanne Olabi, Dana Pe'er, Aviv Regev, Jennifer Rood, Orit Rozenblatt-Rosen, Rahul Satija, Sarah A Teichmann, Barbara Treutlein, Roser Vento-Tormo, Simone Webb, and Human Cell Atlas Developmental Biological Network. A roadmap for the human developmental cell atlas. *Nature*, 597(7875):196–205, September 2021.

[74] Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L Ibarra, Sanjay R Srivatsan, Mohsen Naghipourfar, Riza M Daza, Beth Martin, Jay Shendure, Jose L McFaline-Figueroa, Pierre Boyeau, F Alexander Wolf, Nafissa Yakubova, Stephan Günnemann, Cole Trapnell, David Lopez-Paz, and Fabian J Theis. Predicting cellular responses to complex perturbations in high-throughput screens. *Mol. Syst. Biol.*, page e11517, May 2023.

[75] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN Encoder–Decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

[76] Ricky T Q Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[77] Ricky T Q Chen, Brandon Amos, and Maximilian Nickel. Learning neural event functions for ordinary differential equations. *International Conference on Learning Representations*, 2021.

[78] Satya Narayan Shukla and Benjamin Marlin. Multi-Time Attention Networks for Irregularly Sampled Time Series. In *International Conference on Learning Representations*, March 2021.

[79] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5301–5310. PMLR, 2019.

[80] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ICLR*, 2015.

[81] Julian de Ruiter. pybiomart: A simple pythonic interface to biomart, 2018.

[82] Fergal J Martin, M Ridwan Amode, Alisha Aneja, Olanrewaju Austine-Orimoloye, Andrey G Azov, If Barnes, Arne Becker, Ruth Bennett, Andrew Berry, Jyothish Bhai, Simarpreet Kaur Bhurji, Alexandra Bignell, Sanjay Boddu, Paulo R Branco Lins, Lucy Brooks, Shashank Budhanuru Ramaraju, Mehrnaz Charkhchi, Alexander Cockburn, Luca Da Rin Fiorretto, Claire Davidson, Kamalkumar Dodiya, Sarah Donaldson, Bilal El Houdaigui, Tamara El Naboulsi, Reham Fatima, Carlos Garcia Giron, Thiago Genez, Gurpreet S Ghattaoraya, Jose Gonzalez Martinez, Cristi Guijarro, Matthew Hardy, Zoe Hollis, Thibaut Hourlier, Toby Hunt, Mike Kay, Vinay Kaykala, Tuan Le, Diana Lemos, Diego Marques-Coelho, José Carlos Marugán, Gabriela Alejandra Merino, Louisse Paola Mirabueno, Aleena Mushtaq, Syed Nakib Hossain, Denye N Ogeh, Manoj Pandian Sakthivel, Anne Parker, Malcolm Perry, Ivana Piližota, Irina Prosovetskaia, José G Pérez-Silva, Ahamed Imran Abdul Salam, Nuno Saraiva-Agostinho, Helen Schuilenburg, Dan Sheppard, Swati Sinha, Botond Sipos, William Stark, Emily Steed, Ranjit Sukumaran, Dulika Sumathipala, Marie-Marthe Suner, Likhitha Surapaneni, Kyösti Sutinen, Michal Szpak, Francesca Floriana Tricomi, David Urbina-Gómez, Andres Veidenberg, Thomas A Walsh, Brandon Walts, Elizabeth Wass, Natalie Willhoft, Jamie Allen, Jorge Alvarez-Jarreta, Marc Chakiachvili, Bethany Flint, Stefano Giorgetti, Leanne Haggerty, Garth R Ilsley, Jane E Loveland, Benjamin Moore, Jonathan M Mudge, John Tate, David Thybert, Stephen J Trevanion, Andrea Winterbottom, Adam Frankish, Sarah E Hunt, Magali Ruffier, Fiona Cunningham,

Sarah Dyer, Robert D Finn, Kevin L Howe, Peter W Harrison, Andrew D Yates, and Paul Flicek. Ensembl 2023. *Nucleic Acids Res.*, 51(D1):D933–D941, January 2023.

[83] Eric Arazo, Diego Ortego, Albert Paul, Noel E O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. *ICML*, 2019.

[84] Junnan Li, Richard Socher, and Steven C H Hoi. DivideMix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, February 2020.

[85] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J van der Walt, Matthew Brett, Joshua Wilson, K Jarrod Millman, Nikolay Mayorov, Andrew R J Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E A Quintero, Charles R Harris, Anne M Archibald, Antônio H Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: fundamental algorithms for scientific computing in python. *Nat. Methods*, 17(3):261–272, March 2020.

[86] Ronald Aylmer Fisher. 014: On the“ probable error” of a coefficient of correlation deduced from a small sample. 1921.

[87] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, High-Performance deep learning library. In H Wallach, H Larochelle, A Beygelzimer, F dAlché-Buc, E Fox, and R Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037. Curran Associates, Inc., 2019.

[88] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. UMAP: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[89] F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, 2011.

[90] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech: Theory Exp.*, 2008(10):P10008, October 2008.

[91] Thomas Aynaud. python-louvain: Louvain community detection.

[92] Francesco Iorio, Theo A Knijnenburg, Daniel J Vis, Graham R Bignell, Michael P Menden, Michael Schubert, Nanne Aben, Emanuel Gonçalves, Syd Barthorpe, Howard Lightfoot, Thomas Cokelaer, Patricia Greninger, Ewald van Dyk, Han Chang, Heshani de Silva, Holger Heyn, Xianming Deng, Regina K Egan, Qingsong Liu, Tatiana Mironenko, Xeni Mitropoulos, Laura Richardson, Jinhua Wang, Tinghu Zhang, Sebastian Moran, Sergi Sayols, Maryam Soleimani, David Tamborero, Nuria Lopez-Bigas, Petra Ross-Macdonald, Manel Esteller, Nathanael S Gray, Daniel A Haber, Michael R Stratton, Cyril H Benes, Lodewyk F A Wessels, Julio Saez-Rodriguez, Ultan McDermott, and Mathew J Garnett. A Landscape of Pharmacogenomic Interactions in Cancer. *Cell*, 166(3):740–754, July 2016.

[93] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Res.*, 13(11):2498–2504, November 2003.

[94] Brinton Seashore-Ludlow, Matthew G Rees, Jaime H Cheah, Murat Cokol, Edmund V Price, Matthew E Coletti, Victor Jones, Nicole E Bodycombe, Christian K Soule, Joshua Gould, Benjamin Alexander, Ava Li, Philip Montgomery, Mathias J Wawer, Nurdan Kuru, Joanne D Kotz, C Suk-Yee Hon, Benito Munoz, Ted Liefeld, Vlado Dančík, Joshua A Bittker, Michelle Palmer, James E Bradner, Alykhan F Shamji, Paul A Clemons, and Stuart L Schreiber. Harnessing Connectivity in a Large-Scale Small-Molecule Sensitivity Dataset. *Cancer Discov.*, 5(11):1210–1223, November 2015.

[95] Loren M Berry and Zhiyang Zhao. An examination of IC50 and IC50-shift experiments in assessing

time-dependent inhibition of CYP3A4, CYP2D6 and CYP2C9 in human liver microsomes. *Drug Metab. Lett.*, 2(1):51–59, January 2008.

[96] Aviad Tsherniak, Francisca Vazquez, Phil G Montgomery, Barbara A Weir, Gregory Kryukov, Glenn S Cowley, Stanley Gill, William F Harrington, Sasha Pantel, John M Krill-Burger, Robin M Meyers, Levi Ali, Amy Goodale, Yenarae Lee, Guozhi Jiang, Jessica Hsiao, William F J Gerath, Sara Howell, Erin Merkel, Mahmoud Ghandi, Levi A Garraway, David E Root, Todd R Golub, Jesse S Boehm, and William C Hahn. Defining a Cancer Dependency Map. *Cell*, 170(3):564–576.e16, July 2017.

[97] Robin M Meyers, Jordan G Bryan, James M McFarland, Barbara A Weir, Ann E Sizemore, Han Xu, Neekesh V Dharia, Phillip G Montgomery, Glenn S Cowley, Sasha Pantel, Amy Goodale, Yenarae Lee, Levi D Ali, Guozhi Jiang, Rakela Lubonja, William F Harrington, Matthew Strickland, Ting Wu, Derek C Hawes, Victor A Zhivich, Meghan R Wyatt, Zohra Kalani, Jaime J Chang, Michael Okamoto, Kimberly Stegmaier, Todd R Golub, Jesse S Boehm, Francisca Vazquez, David E Root, William C Hahn, and Aviad Tsherniak. Computational correction of copy number effect improves specificity of CRISPR–Cas9 essentiality screens in cancer cells. *Nat. Genet.*, 49(12):1779–1784, December 2017.

[98] Steven M Corsello, Joshua A Bittker, Zihan Liu, Joshua Gould, Patrick McCarren, Jodi E Hirschman, Stephen E Johnston, Anita Vrcic, Bang Wong, Mariya Khan, Jacob Asiedu, Rajiv Narayan, Christopher C Mader, Aravind Subramanian, and Todd R Golub. The drug repurposing hub: a next-generation drug library and information resource. *Nat. Med.*, 23(4):405–408, April 2017.

[99] P P Tam and R R Behringer. Mouse gastrulation: the formation of a mammalian body plan. *Mech. Dev.*, 68(1-2):3–25, November 1997.

[100] Blanca Pijuan-Sala, Jonathan A Griffiths, Carolina Guibentif, Tom W Hiscock, Wajid Jawaid, Fernando J Calero-Nieto, Carla Mulas, Ximena Ibarra-Soria, Richard C V Tyser, Debbie Lee Lian Ho, Wolf Reik, Shankar Srinivas, Benjamin D Simons, Jennifer Nichols, John C Marioni, and Berthold Göttgens. A single-cell molecular map of mouse gastrulation and early organogenesis. *Nature*, 566 (7745):490–495, February 2019.

[101] Junyue Cao, Malte Spielmann, Xiaojie Qiu, Xingfan Huang, Daniel M Ibrahim, Andrew J Hill, Fan Zhang, Stefan Mundlos, Lena Christiansen, Frank J Steemers, Cole Trapnell, and Jay Shendure. The single-cell transcriptional landscape of mammalian organogenesis. *Nature*, 566(7745):496–502, February 2019.

[102] Chengxiang Qiu, Junyue Cao, Beth K Martin, Tony Li, Ian C Welsh, Sanjay Srivatsan, Xingfan Huang, Diego Calderon, William Stafford Noble, Christine M Disteche, Stephen A Murray, Malte Spielmann, Cecilia B Moens, Cole Trapnell, and Jay Shendure. Systematic reconstruction of cellular trajectories across mouse embryogenesis. *Nat. Genet.*, 54(3):328–341, March 2022.

[103] H Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.*, 24(6):417–441, September 1933.

[104] Juliane O Viegas, Gajendra Kumar Azad, Yuan Lv, Lior Fishman, Tal Paltiel, Sundararaghavan Pattabiraman, Jung Eun Park, Daniel Kaganovich, Siu Kwan Sze, Michal Rabani, Miguel A Esteban, and Eran Meshorer. RNA degradation eliminates developmental transcripts during murine embryonic stem cell differentiation via CAPRIN1-XRN2. *Dev. Cell*, 57(24):2731–2744.e5, December 2022.

[105] Rafal Tomecki, Pawel J Sikorski, and Monika Zakrzewska-Placzek. Comparison of preribosomal RNA processing pathways in yeast, plant and human cells - focus on coordinated action of endo- and exoribonucleases. *FEBS Lett.*, 591(13):1801–1850, July 2017.

[106] Eriko Watada, Sihan Li, Yutaro Hori, Katsunori Fujiki, Katsuhiko Shirahige, Toshifumi Inada, and Takehiko Kobayashi. Age-Dependent ribosomal DNA variations in mice. *Mol. Cell. Biol.*, 40(22), October 2020.

[107] Keisuke Nimura, Masamichi Yamamoto, Makiko Takeichi, Kotaro Saga, Katsuyoshi Takaoka, Norihiko Kawamura, Hirohisa Nitta, Hiromichi Nagano, Saki Ishino, Tatsuya Tanaka, Robert J Schwartz, Hiroyuki Aburatani, and Yasufumi Kaneda. Regulation of alternative polyadenylation by nkx2-5 and xrn2 during mouse heart development. *Elife*, 5, June 2016.

[108] Saibal Chatterjee and Helge Grosshans. Active turnover modulates mature microRNA activity in caenorhabditis elegans. *Nature*, 461(7263):546–549, September 2009.

[109] Saibal Chatterjee, Monika Fasler, Ingo Büssing, and Helge Grosshans. Target-mediated protection of endogenous microRNAs in c. elegans. *Dev. Cell*, 20(3):388–396, March 2011.

[110] Tamoghna Chowdhury, Aniruddha Samajdar, Moumita Sardar, and Saibal Chatterjee. Dauer quiescence as well as continuity of the life cycle after dauer-exit in caenorhabditis elegans are dependent on the endoribonuclease activity of XRN-2. August 2022.

[111] Masaomi Kato, Alexandre de Lencastre, Zachary Pincus, and Frank J Slack. Dynamic expression of small non-coding RNAs, including novel microRNAs and piRNAs/21U-RNAs, during caenorhabditis elegans development. *Genome Biol.*, 10(5):R54, May 2009.

[112] Guo-Jie Qiao, Liang Chen, Jin-Cai Wu, and Zhou-Ri Li. Identification of an eight-gene signature for survival prediction for patients with hepatocellular carcinoma based on integrated bioinformatics analysis. *PeerJ*, 7:e6548, March 2019.

[113] Hitomi Takada and Akira Kurisaki. Emerging roles of nucleolar and ribosomal proteins in cancer, development, and aging. *Cell. Mol. Life Sci.*, 72(21):4015–4025, November 2015.

[114] Tamizhini Loganathan, Srimathy Ramachandran, Prakash Shankaran, Devipriya Nagarajan, and Suma Mohan S. Host transcriptome-guided drug repurposing for COVID-19 treatment: a meta-analysis based approach. *PeerJ*, 8:e9357, June 2020.

[115] Anastasiya Belyaeva, Louis Cammarata, Adityanarayanan Radhakrishnan, Chandler Squires, Karren Dai Yang, G V Shivashankar, and Caroline Uhler. Causal network models of SARS-CoV-2 expression and aging to identify candidates for drug repurposing. *Nat. Commun.*, 12(1):1024, February 2021.

[116] Makoto Minamiyama, Masahisa Katsuno, Hiroaki Adachi, Hideki Doi, Naohide Kondo, Madoka Iida, Shinsuke Ishigaki, Yusuke Fujioka, Shinjiro Matsumoto, Yu Miyazaki, Fumiaki Tanaka, Hiroki Kurihara, and Gen Sobue. Naratriptan mitigates CGRP1-associated motor neuron degeneration caused by an expanded polyglutamine repeat tract. *Nat. Med.*, 18(10):1531–1538, October 2012.

[117] Chen Yang, Hailin Zhang, Mengnuo Chen, Siying Wang, Ruolan Qian, Linmeng Zhang, Xiaowen Huang, Jun Wang, Zhicheng Liu, Wenxin Qin, Cun Wang, Hualian Hang, and Hui Wang. A survey

of optimal strategy for signature-based drug repositioning and an application to liver cancer. *Elife*, 11, February 2022.

[118] Xi Cheng, Wensi Zhao, Mengdi Zhu, Bo Wang, Xuege Wang, Xiaoyun Yang, Yuqi Huang, Minjia Tan, and Jing Li. Drug repurposing for cancer treatment through global propagation with a greedy algorithm in a multilayer network. *Cancer Biol Med*, April 2021.

[119] Adrian J Folkes, Khatereh Ahmadi, Wendy K Alderton, Sonia Alix, Stewart J Baker, Gary Box, Irina S Chuckowree, Paul A Clarke, Paul Depledge, Suzanne A Eccles, Lori S Friedman, Angela Hayes, Timothy C Hancox, Arumugam Kugendradas, Letitia Lensun, Pauline Moore, Alan G Olivero, Jodie Pang, Sonal Patel, Giles H Pergl-Wilson, Florence I Raynaud, Anthony Robson, Nahid Saghir, Laurent Salphati, Sukhjit Sohal, Mark H Ultsch, Melanie Valenti, Heidi J A Wallweber, Nan Chi Wan, Christian Wiesmann, Paul Workman, Alexander Zhyvoloup, Marketa J Zvelebil, and Stephen J Shuttleworth. The identification of 2-(1h-indazol-4-yl)-6-(4-methanesulfonyl-piperazin-1-ylmethyl)-4-morpholin-4-yl-thieno[3,2-d]pyrimidine (GDC-0941) as a potent, selective, orally bioavailable inhibitor of class I PI3 kinase for the treatment of cancer. *J. Med. Chem.*, 51(18):5522–5532, September 2008.

[120] Gerald J Roth, Rudolf Binder, Florian Colbatzky, Claudia Dallinger, Rozsa Schlenker-Herceg, Frank Hilberg, Stefan-Lutz Wollin, and Rolf Kaiser. Nintedanib: from discovery to the clinic. *J. Med. Chem.*, 58(3):1053–1063, February 2015.

[121] Norihiko Suzuki, Fumio Nakagawa, Kazuaki Matsuoka, and Teiji Takechi. Effect of a novel oral chemotherapeutic agent containing a combination of trifluridine, tipiracil and the novel triple angiokinase inhibitor nintedanib, on human colorectal cancer xenografts. *Oncol. Rep.*, 36(6):3123–3130, December 2016.

[122] Michael P Menden, Dennis Wang, Mike J Mason, Bence Szalai, Krishna C Bulusu, Yuanfang Guan, Thomas Yu, Jaewoo Kang, Minji Jeon, Russ Wolfinger, Tin Nguyen, Mikhail Zaslavskiy, AstraZeneca-Sanger Drug Combination DREAM Consortium, In Sock Jang, Zara Ghazoui, Mehmet Eren Ahsen, Robert Vogel, Elias Chaibub Neto, Thea Norman, Eric K Y Tang, Mathew J Garnett, Giovanni Y Di Veroli, Stephen Fawell, Gustavo Stolovitzky, Justin Guinney, Jonathan R

Dry, and Julio Saez-Rodriguez. Community assessment to advance computational prediction of cancer drug combinations in a pharmacogenomic screen. *Nat. Commun.*, 10(1):2674, June 2019.

[123] Xingdong Chen, Jeffrey Gole, Athurva Gore, Qiye He, Ming Lu, Jun Min, Ziyu Yuan, Xiaorong Yang, Yanfeng Jiang, Tiejun Zhang, Chen Suo, Xiaojie Li, Lei Cheng, Zhenhua Zhang, Hongyu Niu, Zhe Li, Zhen Xie, Han Shi, Xiang Zhang, Min Fan, Xiaofeng Wang, Yajun Yang, Justin Dang, Catie McConnell, Juan Zhang, Jiucun Wang, Shunzhang Yu, Weimin Ye, Yuan Gao, Kun Zhang, Rui Liu, and Li Jin. Non-invasive early detection of cancer four years before conventional diagnosis using a blood test. *Nat. Commun.*, 11(1):3475, July 2020.

[124] Landon C Brown, Matthew D Tucker, Ramy Sedhom, Eric B Schwartz, Jason Zhu, Chester Kao, Matthew K Labriola, Rajan T Gupta, Daniele Marin, Yuan Wu, Santosh Gupta, Tian Zhang, Michael R Harrison, Daniel J George, Ajjai Alva, Emmanuel S Antonarakis, and Andrew J Armstrong. LRP1B mutations are associated with favorable outcomes to immune checkpoint inhibitors across multiple cancer types. *J Immunother Cancer*, 9(3), March 2021.

[125] Nadia Arang and J Silvio Gutkind. G Protein-Coupled receptors and heterotrimeric G proteins as cancer drivers. *FEBS Lett.*, 594(24):4201–4232, December 2020.

[126] Daisuke Ichikawa, Kyoko Yamashita, Yusuke Okuno, Hideki Muramatsu, Norihiro Murakami, Kyogo Suzuki, Daiei Kojima, Shinsuke Kataoka, Motoharu Hamada, Rieko Taniguchi, Eri Nishikawa, Nozomu Kawashima, Atsushi Narita, Nobuhiro Nishio, Asahito Hama, Kenji Kasai, Seiji Mizuno, Yoshie Shimoyama, Masato Nakaguro, Hajime Okita, Seiji Kojima, Atsuko Nakazawa, and Yoshiyuki Takahashi. Integrated diagnosis based on transcriptome analysis in suspected pediatric sarcomas. *NPJ Genom Med*, 6(1):49, June 2021.

[127] Silvia Pietrobono, Sinforosa Gagliardi, and Barbara Stecca. Non-canonical hedgehog signaling pathway in cancer: Activation of GLI transcription factors beyond smoothened. *Front. Genet.*, 10:556, June 2019.

[128] Winnie W Lo, Dushanthi Pinnaduwage, Nalan Gokgoz, Jay S Wunder, and Irene L Andrulis. Aberrant hedgehog signaling and clinical outcome in osteosarcoma. *Sarcoma*, 2014:261804, March 2014.

[129] Sudeep Banerjee, Christopher L Corless, Markku M Miettinen, Sangkyu Noh, Rowan Ustoy, Jessica L Davis, Chih-Min Tang, Mayra Yebra, Adam M Burgoyne, and Jason K Sicklick. Loss of the PTCH1 tumor suppressor defines a new subset of plexiform fibromyxoma. *J. Transl. Med.*, 17(1): 246, July 2019.

[130] Maria Florencia Martinez, Maria Vanesa Romano, Alfredo Pedro Martinez, Abel González, Carolina Muchnik, Fernando Miguel Stengel, Luis Daniel Mazzuoccolo, and Pablo Javier Azurmendi. Nevoid basal cell carcinoma syndrome: PTCH1 mutation profile and expression of genes involved in the hedgehog pathway in argentinian patients. *Cells*, 8(2), February 2019.

[131] Zheng Ge, Xing Guo, Jianyong Li, Melanie Hartman, Yuka Imamura Kawasawa, Sinisa Dovat, and Chunhua Song. Clinical significance of high c-MYC and low MYCBP2 expression and their association with ikaros dysfunction in adult acute lymphoblastic leukemia. *Oncotarget*, 6(39):42300–42311, December 2015.

[132] R Vatapalli, V Sagar, Y Rodriguez, J C Zhao, K Unno, S Pamarthy, B Lysy, J Anker, H Han, Y A Yoo, M Truica, Z R Chalmers, F Giles, J Yu, D Chakravarti, B Carneiro, and S A Abdulkadir. Histone methyltransferase DOT1L coordinates AR and MYC stability in prostate cancer. *Nat. Commun.*, 11 (1):4153, August 2020.

[133] Joon Won Yoon, Marisa Gallant, Marilyn L G Lamm, Stephen Iannaccone, Karl-Frederic Vieux, Maria Proytcheva, Elizabeth Hyjek, Philip Iannaccone, and David Walterhouse. Noncanonical regulation of the hedgehog mediator GLI1 by c-MYC in burkitt lymphoma. *Mol. Cancer Res.*, 11(6): 604–615, June 2013.

[134] Marcella Tazzari, Laura Bergamaschi, Alessandro De Vita, Paola Collini, Marta Barisella, Alessia Bertolotti, Toni Ibrahim, Sandro Pasquali, Chiara Castelli, and Viviana Vallacchi. Molecular determinants of soft tissue sarcoma immunity: Targets for immune intervention. *Int. J. Mol. Sci.*, 22(14), July 2021.

[135] Xiaofeng Wang, Jeffrey R Haswell, and Charles W M Roberts. Molecular pathways: SWI/SNF

(BAF) complexes are frequently mutated in cancer–mechanisms and potential therapeutic insights. *Clin. Cancer Res.*, 20(1):21–27, January 2014.

[136] Xiaoxiao Fan, Hongbin Guo, Binghua Dai, Lifeng He, Daizhan Zhou, and Hui Lin. The association between methylation patterns of DNAH17 and clinicopathological factors in hepatocellular carcinoma. *Cancer Med.*, 8(1):337–350, January 2019.

[137] Nadia B Hassounah, Thomas A Bunch, and Kimberly M McDermott. Molecular pathways: the role of primary cilia in cancer progression and therapeutics with a focus on hedgehog signaling. *Clin. Cancer Res.*, 18(9):2429–2435, May 2012.

[138] Barbara Stecca and Ariel Ruiz i Altaba. Context-dependent regulation of the GLI code in cancer by HEDGEHOG and non-HEDGEHOG signals. *J. Mol. Cell Biol.*, 2(2):84–95, April 2010.

[139] Jillian Brechbiel, Karen Miller-Moslin, and Alex A Adjei. Crosstalk between hedgehog and other signaling pathways as a basis for combination therapies in cancer. *Cancer Treat. Rev.*, 40(6):750–759, July 2014.

[140] Juhui Chen, Jingshi Zhang, Liang Hong, and Yongtao Zhou. EGFLAM correlates with cell proliferation, migration, invasion and poor prognosis in glioblastoma. *Cancer Biomark.*, 24(3):343–350, 2019.

[141] Qian Yu, Xiaojie Wang, Yinghong Yang, Pan Chi, Jianping Huang, Shengliang Qiu, Xin Zheng, and Xiaowen Chen. Upregulated NLGN1 predicts poor survival in colorectal cancer. *BMC Cancer*, 21 (1):884, August 2021.

[142] Yi-Ming Ren, Yuan-Hui Duan, Yun-Bo Sun, Tao Yang, Wen-Jun Zhao, Dong-Liang Zhang, Zheng-Wei Tian, and Meng-Qiang Tian. Exploring the key genes and pathways of side population cells in human osteosarcoma using gene expression array analysis. *J. Orthop. Surg. Res.*, 13(1):153, June 2018.

[143] Colleen Cutcliffe, Donna Kersey, Chiang-Ching Huang, Yong Zeng, David Walterhouse, Elizabeth J Perlman, and Renal Tumor Committee of the Children's Oncology Group. Clear cell sarcoma of

the kidney: up-regulation of neural markers with activation of the sonic hedgehog and akt pathways. *Clin. Cancer Res.*, 11(22):7986–7994, November 2005.

[144] Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. On calibration and Out-of-Domain generalization. In M Ranzato, A Beygelzimer, Y Dauphin, P S Liang, and J Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 2215–2227. Curran Associates, Inc., 2021.

[145] Manu Setty, Vaidotas Kiseliovas, Jacob Levine, Adam Gayoso, Linas Mazutis, and Dana Pe'er. Characterization of cell fate probabilities in single-cell data with palantir. *Nat. Biotechnol.*, 37(4):451–460, April 2019.

[146] Kelly Street, Davide Risso, Russell B Fletcher, Diya Das, John Ngai, Nir Yosef, Elizabeth Purdom, and Sandrine Dudoit. Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics*, 19(1):477, June 2018.

[147] David S Fischer, Anna K Fiedler, Eric M Kernfeld, Ryan M J Genga, Aimée Bastidas-Ponce, Mostafa Bakhti, Heiko Lickert, Jan Hasenauer, Rene Maehr, and Fabian J Theis. Inferring population dynamics from single-cell RNA-sequencing time series data. *Nat. Biotechnol.*, 37(4):461–468, April 2019.

[148] Grace Hui Ting Yeo, Sachit D Saksena, and David K Gifford. Generative modeling of single-cell time series with PRESCIENT enables prediction of cell trajectories with interventions. *Nat. Commun.*, 12 (1):3222, May 2021.

[149] Patricia Jaaks, Elizabeth A Coker, Daniel J Vis, Olivia Edwards, Emma F Carpenter, Simonetta M Leto, Lisa Dwane, Francesco Sassi, Howard Lightfoot, Syd Barthorpe, Dieudonne van der Meer, Wanjuan Yang, Alexandra Beck, Tatiana Mironenko, Caitlin Hall, James Hall, Iman Mali, Laura Richardson, Charlotte Tolley, James Morris, Frances Thomas, Ermira Lleshi, Nanne Aben, Cyril H Benes, Andrea Bertotti, Livio Trusolino, Lodewyk Wessels, and Mathew J Garnett. Effective drug combinations in breast, colon and pancreatic cancer cells. *Nature*, 603(7899):166–173, March 2022.

[150] Boyan Bonev and Giacomo Cavalli. Organization and function of the 3D genome. *Nat. Rev. Genet.*, 17(11):661–678, October 2016.

[151] T Cremer and C Cremer. Chromosome territories, nuclear architecture and gene regulation in mammalian cells. *Nat. Rev. Genet.*, 2(4):292–301, April 2001.

[152] Robert A Beagrie, Antonio Scialdone, Markus Schueler, Dorothee C A Kraemer, Mita Chotalia, Sheila Q Xie, Mariano Barbieri, Inês de Santiago, Liron-Mark Lavitas, Miguel R Branco, James Fraser, Josée Dostie, Laurence Game, Niall Dillon, Paul A W Edwards, Mario Nicodemi, and Ana Pombo. Complex multi-enhancer contacts captured by genome architecture mapping. *Nature*, 543 (7646):519–524, March 2017.

[153] Sofia A Quinodoz, Noah Ollikainen, Barbara Tabak, Ali Palla, Jan Marten Schmidt, Elizabeth Detmar, Mason M Lai, Alexander A Shishkin, Prashant Bhat, Yodai Takei, Vickie Trinh, Erik Aznauryan, Pamela Russell, Christine Cheng, Marko Jovanovic, Amy Chow, Long Cai, Patrick McDonel, Manuel Garber, and Mitchell Guttman. Higher-Order inter-chromosomal hubs shape 3D genome organization in the nucleus. *Cell*, 174(3):744–757.e24, July 2018.

[154] Maxwell R Mumbach, Adam J Rubin, Ryan A Flynn, Chao Dai, Paul A Khavari, William J Greenleaf, and Howard Y Chang. HiChIP: efficient and sensitive analysis of protein-directed genome architecture. *Nat. Methods*, 13(11):919–922, November 2016.

[155] Anthony D Schmitt, Ming Hu, and Bing Ren. Genome-wide mapping and analysis of chromosome architecture. *Nat. Rev. Mol. Cell Biol.*, 17(12):743–755, December 2016.

[156] Zhilan Li and Zhiming Dai. SRHiC: A deep learning model to enhance the resolution of Hi-C data. *Front. Genet.*, 11:353, April 2020.

[157] Tong Liu and Zheng Wang. HiCNN: a very deep convolutional neural network to better enhance the resolution of Hi-C data. *Bioinformatics*, 35(21):4222–4228, November 2019.

[158] Tong Liu and Zheng Wang. HiCNN2: Enhancing the resolution of Hi-C data using an ensemble of convolutional neural networks. *Genes*, 10(11), October 2019.

[159] Yan Zhang, Lin An, Jie Xu, Bo Zhang, W Jim Zheng, Ming Hu, Jijun Tang, and Feng Yue. Enhancing Hi-C data resolution with deep convolutional neural network HiCPlus. *Nat. Commun.*, 9(1):750, February 2018.

[160] Michael Dimmick. *HiCsr: A Hi-C Super-Resolution Framework for Producing Highly Realistic Contact Maps*. PhD thesis, University of Toronto (Canada), 2020.

[161] Parker Hicks and Oluwatosin Oluwadare. HiCARN: resolution enhancement of Hi-C data using cascading residual networks. *Bioinformatics*, 38(9):2414–2421, April 2022.

[162] Max Highsmith and Jianlin Cheng. VEHiCLE: a variationally encoded Hi-C loss enhancement algorithm for improving and generating Hi-C data. *Sci. Rep.*, 11(1):8880, April 2021.

[163] Hao Hong, Shuai Jiang, Hao Li, Guifang Du, Yu Sun, Huan Tao, Cheng Quan, Chenghui Zhao, Ruijiang Li, Wanying Li, Xiaoyao Yin, Yangchen Huang, Cheng Li, Hebing Chen, and Xiaochen Bo. DeepHiC: A generative adversarial network for enhancing Hi-C data resolution. *PLoS Comput. Biol.*, 16(2):e1007287, February 2020.

[164] Qiao Liu, Hairong Lv, and Rui Jiang. hicGAN infers super resolution Hi-C data with generative adversarial networks. *Bioinformatics*, 35(14):i99–i107, July 2019.

[165] Christopher Jf Cameron, Josée Dostie, and Mathieu Blanchette. HIFI: estimating DNA-DNA interaction frequency from Hi-C data at restriction-fragment resolution. *Genome Biol.*, 21(1):11, January 2020.

[166] Abhijit Chakraborty, Jeffrey G Wang, and Ferhat Ay. dcHiC detects differential compartments across multiple Hi-C datasets. *Nat. Commun.*, 13(1):6827, November 2022.

[167] Jean-Philippe Fortin and Kasper D Hansen. Reconstructing A/B compartments as revealed by Hi-C using long-range correlations in epigenetic data. *Genome Biol.*, 16(1):180, August 2015.

[168] Noelle Haddad, Cédric Vaillant, and Daniel Jost. IC-Finder: inferring robustly the hierarchical organization of chromatin folding. *Nucleic Acids Res.*, 45(10):e81, June 2017.

[169] Hannah L Harris, Huiya Gu, Moshe Olshansky, Ailun Wang, Irene Farabella, Yossi Eliaz, Achyuth Kalluchi, Akshay Krishna, Mozes Jacobs, Gesine Cauer, Melanie Pham, Suhas S P Rao, Olga Dudchenko, Arina Omer, Kiana Mohajeri, Sungjae Kim, Michael H Nichols, Eric S Davis, Dimos Gkountaroulis, Devika Udupa, Aviva Presser Aiden, Victor G Corces, Douglas H Phanstiel, William Stafford

Noble, Guy Nir, Michele Di Pierro, Jeong-Sun Seo, Michael E Talkowski, Erez Lieberman Aiden, and M Jordan Rowley. Chromatin alternates between a and B compartments at kilobase scale for subgenic organization. *Nat. Commun.*, 14(1):3303, June 2023.

[170] Emily Crane, Qian Bian, Rachel Patton McCord, Bryan R Lajoie, Bayly S Wheeler, Edward J Ralston, Satoru Uzawa, Job Dekker, and Barbara J Meyer. Condensin-driven remodelling of X chromosome topology during dosage compensation. *Nature*, 523(7559):240–244, July 2015.

[171] Jesse R Dixon, Siddarth Selvaraj, Feng Yue, Audrey Kim, Yan Li, Yin Shen, Ming Hu, Jun S Liu, and Bing Ren. Topological domains in mammalian genomes identified by analysis of chromatin interactions. *Nature*, 485(7398):376–380, April 2012.

[172] Darya Filippova, Rob Patro, Geet Duggal, and Carl Kingsford. Identification of alternative topological domains in chromatin. *Algorithms Mol. Biol.*, 9:14, May 2014.

[173] Fidel Ramírez, Vivek Bhardwaj, Laura Arrigoni, Kin Chung Lam, Björn A Grüning, José Villaveces, Bianca Habermann, Asifa Akhtar, and Thomas Manke. High-resolution TADs reveal DNA sequences underlying genome organization in flies. *Nat. Commun.*, 9(1):189, January 2018.

[174] François Serra, Davide Baù, Mike Goodstadt, David Castillo, Guillaume J Filion, and Marc A Marti-Renom. Automatic analysis and 3d-modelling of Hi-C data using TADbit reveals structural features of the fly chromatin colors. *PLoS Comput. Biol.*, 13(7):e1005665, July 2017.

[175] Abbas Roayaei Ardakany, Halil Tuvan Gezer, Stefano Lonardi, and Ferhat Ay. Mustache: multi-scale detection of chromatin loops from Hi-C and Micro-C maps using scale-space representation. *Genome Biol.*, 21(1):256, September 2020.

[176] M Jordan Rowley, Axel Poulet, Michael H Nichols, Brianna J Bixler, Adrian L Sanborn, Elizabeth A Brouhard, Karen Hermetz, Hannah Linsenbaum, Gyorgyi Csankovszki, Erez Lieberman Aiden, and Victor G Corces. Analysis of Hi-C data using SIP effectively identifies loops in organisms from c. elegans to mammals. *Genome Res.*, 30(3):447–458, March 2020.

[177] Tarik J Salameh, Xiaotao Wang, Fan Song, Bo Zhang, Sage M Wright, Chachrit Khunsriraksakul,

Yijun Ruan, and Feng Yue. A supervised learning framework for chromatin loop detection in genome-wide contact maps. *Nat. Commun.*, 11(1):3428, July 2020.

[178] Joachim Wolff, Rolf Backofen, and Björn Grüning. Loop detection using Hi-C data with HiCExplorer. *Gigascience*, 11, July 2022.

[179] Yangyang Hu and Wenxiu Ma. EnHiC: learning fine-resolution Hi-C contact maps using a generative adversarial framework. *Bioinformatics*, 37(Suppl_1):i272–i279, July 2021.

[180] L Carron, J B Morlot, V Matthys, A Lesne, and J Mozziconacci. Boost-HiC: computational enhancement of long-range contacts in chromosomal contact maps. *Bioinformatics*, 35(16):2724–2729, August 2019.

[181] Yanlin Zhang and Mathieu Blanchette. Reference panel-guided super-resolution inference of Hi-C data. *Bioinformatics*, 39(39 Suppl 1):i386–i393, June 2023.

[182] Geoff Fudenberg, David R Kelley, and Katherine S Pollard. Predicting 3D genome folding from DNA sequence with akita. *Nat. Methods*, 17(11):1111–1117, November 2020.

[183] Rui Yang, Arnav Das, Vianne R Gao, Alireza Karbalayghareh, William S Noble, Jeffery A Bilmes, and Christina S Leslie. Epiphany: predicting Hi-C contact maps from 1D epigenomic signals. *Genome Biol.*, 24(1):134, June 2023.

[184] Jian Zhou. Sequence-based modeling of three-dimensional genome architecture from kilobase to chromosome scale. *Nat. Genet.*, 54(5):725–734, May 2022.

[185] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *J. R. Stat. Soc. Series B Stat. Methodol.*, 57(1):289–300, 1995.

[186] T Tony Cai and Wenguang Sun. Simultaneous testing of grouped hypotheses: Finding needles in multiple haystacks. *J. Am. Stat. Assoc.*, 104(488):1467–1481, December 2009.

[187] Rola Dali and Mathieu Blanchette. A critical assessment of topologically associating domain prediction tools. *Nucleic Acids Res.*, 45(6):2994–3005, April 2017.

[188] Sicheng Gao, Xuhui Liu, Bohan Zeng, Sheng Xu, Yanjing Li, Xiaoyan Luo, Jianzhuang Liu, Xiantong Zhen, and Baochang Zhang. Implicit diffusion models for continuous Super-Resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10021–10030, June 2023.

[189] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. SRDiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479:47–59, March 2022.

[190] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image Super-Resolution via iterative refinement. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(4): 4713–4726, April 2023.

[191] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. February 2021.

[192] Cyril Matthey-Doret, Lyam Baudry, Axel Breuer, Rémi Montagne, Nadège Guiglielmoni, Vittore Scolari, Etienne Jean, Arnaud Campeas, Philippe Henri Chanut, Edgar Oriol, Adrien Méot, Laurent Politis, Antoine Vigouroux, Pierrick Moreau, Romain Koszul, and Axel Cournac. Computer vision for pattern detection in chromosome contact maps. *Nat. Commun.*, 11(1):5795, November 2020.

[193] Tanya Barrett, Stephen E Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F Kim, Maxim Tomashevsky, Kimberly A Marshall, Katherine H Phillippy, Patti M Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data sets–update. *Nucleic Acids Res.*, 41(Database issue):D991–5, January 2013.

[194] Lusy Handoko, Han Xu, Guoliang Li, Chew Yee Ngan, Elaine Chew, Marie Schnapp, Charlie Wah Heng Lee, Chaopeng Ye, Joanne Lim Hui Ping, Fabianus Mulawadi, Eleanor Wong, Jianpeng Sheng, Yubo Zhang, Thompson Poh, Chee Seng Chan, Galih Kunarso, Atif Shahab, Guillaume Bourque, Valere Cacheux-Rataboul, Wing-Kin Sung, Yijun Ruan, and Chia-Lin Wei. CTCF-mediated functional chromatin interactome in pluripotent cells. *Nat. Genet.*, 43(7):630–638, June 2011.

[195] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards photorealistic image generation and editing with Text-Guided diffusion models. December 2021.

[196] Charles E Grant, Timothy L Bailey, and William Stafford Noble. FIMO: scanning for occurrences of a given motif. *Bioinformatics*, 27(7):1017–1018, April 2011.

[197] Dominic Schmidt, Michael D Wilson, Christiana Spyrou, Gordon D Brown, James Hadfield, and Duncan T Odom. ChIP-seq: using high-throughput sequencing to discover protein-DNA interactions. *Methods*, 48(3):240–248, July 2009.

[198] Tsung-Han S Hsieh, Assaf Weiner, Bryan Lajoie, Job Dekker, Nir Friedman, and Oliver J Rando. Mapping nucleosome resolution chromosome folding in yeast by Micro-C. *Cell*, 162(1):108–119, July 2015.

[199] Tsung-Han S Hsieh, Claudia Cattoglio, Elena Slobodyanyuk, Anders S Hansen, Oliver J Rando, Robert Tjian, and Xavier Darzacq. Resolving the 3D landscape of Transcription-Linked mammalian chromatin folding. *Mol. Cell*, 78(3):539–553.e8, May 2020.

[200] Hyunghoon Cho, Bonnie Berger, and Jian Peng. Diffusion component analysis: Unraveling functional topology in biological networks. *Res. Comput. Mol. Biol.*, 9029:62–64, April 2015.

[201] Hyunghoon Cho, Bonnie Berger, and Jian Peng. Compact integration of Multi-Network topology for functional analysis of genes. *Cell Syst*, 3(6):540–548.e5, December 2016.

[202] Duncan T Forster, Sheena C Li, Yoko Yashiroda, Mami Yoshimura, Zhijian Li, Luis Alberto Vega Isuhuaylas, Kaori Itto-Nakama, Daisuke Yamanaka, Yoshikazu Ohya, Hiroyuki Osada, Bo Wang, Gary D Bader, and Charles Boone. BIONIC: biological network integration using convolutions. *Nat. Methods*, 19(10):1250–1261, October 2022.

[203] Bolin Chen, Min Li, Jianxin Wang, Xuequn Shang, and Fang-Xiang Wu. A fast and high performance multiple data integration algorithm for identifying human disease genes. *BMC Med. Genomics*, 8 Suppl 3:S2, September 2015.

[204] Bolin Chen, Min Li, Jianxin Wang, and Fang-Xiang Wu. Disease gene identification by using graph kernels and markov random fields. *Sci. China Life Sci.*, 57(11):1054–1063, November 2014.

[205] Bolin Chen, Jianxin Wang, Min Li, and Fang-Xiang Wu. Identifying disease genes by integrating multiple data sources. *BMC Med. Genomics*, 7 Suppl 2:S2, October 2014.

[206] Jing Chen, Bruce J Aronow, and Anil G Jegga. Disease candidate gene identification and prioritization using protein interaction networks. *BMC Bioinformatics*, 10:73, February 2009.

[207] Insuk Lee, U Martin Blom, Peggy I Wang, Jung Eun Shim, and Edward M Marcotte. Prioritizing candidate disease genes by network-based boosting of genome-wide association data. *Genome Res.*, 21(7):1109–1121, July 2011.

[208] Mark D M Leiserson, Fabio Vandin, Hsin-Ta Wu, Jason R Dobson, Jonathan V Eldridge, Jacob L Thomas, Alexandra Papoutsaki, Younhun Kim, Beifang Niu, Michael McLellan, Michael S Lawrence, Abel Gonzalez-Perez, David Tamborero, Yuwei Cheng, Gregory A Ryslik, Nuria Lopez-Bigas, Gad Getz, Li Ding, and Benjamin J Raphael. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nat. Genet.*, 47(2):106–114, February 2015.

[209] Yongjin Li and Jagdish C Patra. Genome-wide inferring gene-phenotype relationship by walking on the heterogeneous network. *Bioinformatics*, 26(9):1219–1224, May 2010.

[210] Yuansheng Liu, Xiangxiang Zeng, Zengyou He, and Quan Zou. Inferring microRNA-disease associations by random walk on a heterogeneous network with multiple data sources. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 14(4):905–915, July 2017.

[211] Damian Smedley, Sebastian Köhler, Johanna Christina Czeschik, Joanna Amberger, Carol Bocchini, Ada Hamosh, Julian Veldboer, Tomasz Zemojtel, and Peter N Robinson. Walking the interactome for candidate prioritization in exome sequencing studies of mendelian diseases. *Bioinformatics*, 30(22): 3215–3222, November 2014.

[212] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin

Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nat. Methods*, 11(3):333–337, January 2014.

[213] Ping Xuan, Ke Han, Yahong Guo, Jin Li, Xia Li, Yingli Zhong, Zhaogong Zhang, and Jian Ding. Prediction of potential disease-associated microRNAs based on random walk. *Bioinformatics*, 31 (11):1805–1815, June 2015.

[214] Angela Bruex, Raghunandan M Kainkaryam, Yana Wieckowski, Yeon Hee Kang, Christine Bernhardt, Yang Xia, Xiaohua Zheng, Jean Y Wang, Myeong Min Lee, Philip Benfey, Peter J Woolf, and John Schiefelbein. A gene regulatory network for root epidermis cell differentiation in arabidopsis. *PLoS Genet.*, 8(1):e1002446, January 2012.

[215] Yijie Wang, Justin M Fear, Isabelle Berger, Hangnoh Lee, Brian Oliver, and Teresa M Przytycka. Reconstruction of gene regulatory networks by integrating biological model and a recommendation system. In *Research in Computational Molecular Biology*, pages 274–275. Springer International Publishing, 2020.

[216] Elena Zotenko, Julian Mestre, Dianne P O'Leary, and Teresa M Przytycka. Why do hubs in the yeast protein interaction network tend to be essential: reexamining the connection between the network topology and essentiality. *PLoS Comput. Biol.*, 4(8):e1000140, August 2008.

[217] Tewis Bouwmeester et al. A physical and functional map of the human TNF-alpha/NF-kappa B signal transduction pathway. *Nat. Cell Biol.*, 6(2):97–105, February 2004.

[218] Anne-Claude Gavin et al. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415(6868):141–147, January 2002.

[219] Yuen Ho et al. Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. *Nature*, 415(6868):180–183, January 2002.

[220] Peter Uetz et al. A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature*, 403(6770):623–627, February 2000.

[221] Amy Hin Yan Tong et al. Global mapping of the yeast genetic interaction network. *Science*, 303 (5659):808–813, February 2004.

[222] Roktaek Lim, Josephine Jill T Cabatbat, Thomas L P Martin, Haneul Kim, Seunghyeon Kim, Jaeyun Sung, Cheol-Min Ghim, and Pan-Jun Kim. Large-scale metabolic interaction network of the mouse and human gut microbiota. *Sci Data*, 7(1):204, June 2020.

[223] Hongzhong Lu et al. A consensus s. cerevisiae metabolic model yeast8 and its ecosystem for comprehensively probing cellular metabolism. *Nat. Commun.*, 10(1):3586, August 2019.

[224] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nat. Rev. Genet.*, 12(1):56–68, January 2011.

[225] Marinka Žitnik and Blaž Zupan. Gene network inference by fusing data from diverse distributions. *Bioinformatics*, 31(12):i230–9, June 2015.

[226] Koyel Mitra, Anne-Ruxandra Carvunis, Sanath Kumar Ramesh, and Trey Ideker. Integrative approaches for finding modular structure in biological networks. *Nat. Rev. Genet.*, 14(10):719–732, October 2013.

[227] The GTEx Consortium et al. The Genotype-Tissue expression (GTEx) pilot analysis: Multitissue gene regulation in humans. *Science*, 348(6235):648–660, 2015.

[228] Taibo Li, Rasmus Wernersson, Rasmus B Hansen, Heiko Horn, Johnathan Mercer, Greg Slodkowicz, Christopher T Workman, Olga Rigina, Kristoffer Rapacki, Hans H Stærfeldt, Søren Brunak, Thomas S Jensen, and Kasper Lage. A scored human protein-protein interaction network to catalyze genomic interpretation. *Nat. Methods*, 14(1):61–64, January 2017.

[229] Yue Qin et al. A multi-scale map of cell structure fusing protein images and interactions. *Nature*, 600 (7889):536–542, December 2021.

[230] Roded Sharan, Silpa Suthram, Ryan M Kelley, Tanja Kuhn, Scott McCuine, Peter Uetz, Taylor Sittler, Richard M Karp, and Trey Ideker. Conserved patterns of protein interaction in multiple species. *Proc. Natl. Acad. Sci. U. S. A.*, 102(6):1974–1979, February 2005.

[231] Michael E Cusick, Niels Klitgord, Marc Vidal, and David E Hill. Interactome: gateway into systems biology. *Hum. Mol. Genet.*, 14 Spec No. 2:R171–81, October 2005.

[232] Vladimir Gligorijevic, Meet Barot, and Richard Bonneau. deepNF: deep network fusion for protein function prediction. *Bioinformatics*, 34(22):3873–3881, November 2018.

[233] Jiajie Peng, Hansheng Xue, Zhongyu Wei, Idil Tuncali, Jianye Hao, and Xuequn Shang. Integrating multi-network topology for gene function prediction using deep neural networks. *Brief. Bioinform.*, 22(2):2096–2105, March 2021.

[234] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. Scalable multiplex network embedding. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pages 3082–3088. AAAI Press, July 2018.

[235] Gert R G Lanckriet, Tijl De Bie, Nello Cristianini, Michael I Jordan, and William Stafford Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, November 2004.

[236] Koji Tsuda and William Stafford Noble. Learning kernels from biological networks by maximizing entropy. *Bioinformatics*, 20 Suppl 1:i326–33, August 2004.

[237] Sara Mostafavi, Debajyoti Ray, David Warde-Farley, Chris Grouios, and Quaid Morris. GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol.*, 9 Suppl 1:S4, June 2008.

[238] Sara Mostafavi and Quaid Morris. Fast integration of heterogeneous data sources for predicting gene function with limited annotation. *Bioinformatics*, 26(14):1759–1765, July 2010.

[239] Mengfei Cao, Christopher M Pietras, Xian Feng, Kathryn J Doroschak, Thomas Schaffner, Jisoo Park, Hao Zhang, Lenore J Cowen, and Benjamin J Hescott. New directions for diffusion-based network prediction of protein function: incorporating pathways with confidence. *Bioinformatics*, 30 (12):i219–27, June 2014.

[240] Mengfei Cao, Hao Zhang, Jisoo Park, Noah M Daniels, Mark E Crovella, Lenore J Cowen, and

Benjamin Hescott. Going the distance for protein function prediction: a new distance metric for protein interaction networks. *PLoS One*, 8(10):e76339, October 2013.

[241] Sheng Wang, Hyunghoon Cho, Chengxiang Zhai, Bonnie Berger, and Jian Peng. Exploiting ontology graph for predicting sparsely annotated gene function. *Bioinformatics*, 31(12):i357–64, June 2015.

[242] Marc Vidal, Michael E Cusick, and Albert-László Barabási. Interactome networks and human disease. *Cell*, 144(6):986–998, March 2011.

[243] María E Soler-Oliva, José A Guerrero-Martínez, Valentina Bachetti, and José C Reyes. Analysis of the relationship between coexpression domains and chromatin 3D organization. *PLoS Comput. Biol.*, 13(9):e1005708, September 2017.

[244] Hyunju Lee, Zhidong Tu, Minghua Deng, Fengzhu Sun, and Ting Chen. Diffusion kernel-based logistic regression models for protein function prediction. *OMICS*, 10(1):40–55, 2006.

[245] Wei Peng, Min Li, Lu Chen, and Lusheng Wang. Predicting protein functions by using unbalanced random walk algorithm on three biological networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 14(2):360–369, March 2017.

[246] Roded Sharan, Igor Ulitsky, and Ron Shamir. Network-based prediction of protein function. *Mol. Syst. Biol.*, 3:88, March 2007.

[247] Rohit Singh, Kapil Devkota, Samuel Sledzieski, Bonnie Berger, and Lenore Cowen. Topsy-Turvy: integrating a global view into sequence-based PPI prediction. *Bioinformatics*, 38(Suppl 1):i264–i272, June 2022.

[248] Koji Tsuda, Hyunjung Shin, and Bernhard Schölkopf. Fast protein classification with multiple networks. *Bioinformatics*, 21 Suppl 2:ii59–65, September 2005.

[249] Konstantin Voevodski, Shang-Hua Teng, and Yu Xia. Finding local communities in protein networks. *BMC Bioinformatics*, 10:297, September 2009.

[250] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, February 2007.

[251] Paul R Rosenbaum. Model-Based direct adjustment. *J. Am. Stat. Assoc.*, 82(398):387–394, 1987.

[252] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2017.

[253] S Gopal Krishna Patro and Kishore Kumar Sahu. Normalization: A preprocessing stage. March 2015.

[254] B Schwikowski, P Uetz, and S Fields. A network of protein-protein interactions in yeast. *Nat. Biotechnol.*, 18(12):1257–1261, December 2000.

[255] Jörg Menche, Amitabh Sharma, Maksim Kitsak, Susan Dina Ghiassian, Marc Vidal, Joseph Loscalzo, and Albert-László Barabási. Disease networks. uncovering disease-disease relationships through the incomplete interactome. *Science*, 347(6224):1257601, February 2015.

[256] David Warde-Farley et al. The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function. *Nucleic Acids Res.*, 38(Web Server issue):W214–20, July 2010.

[257] Rose Oughtred, Chris Stark, Bobby-Joe Breitkreutz, Jennifer Rust, Lorrie Boucher, Christie Chang, Nadine Kolas, Lara O'Donnell, Genie Leung, Rochelle McAdam, Frederick Zhang, Sonam Dolma, Andrew Willems, Jasmin Coulombe-Huntington, Andrew Chatr-Aryamontri, Kara Dolinski, and Mike Tyers. The BioGRID interaction database: 2019 update. *Nucleic Acids Res.*, 47(D1):D529–D541, January 2019.

[258] Evelyn Camon, Michele Magrane, Daniel Barrell, Vivian Lee, Emily Dimmer, John Maslen, David Binns, Nicola Harte, Rodrigo Lopez, and Rolf Apweiler. The gene ontology annotation (GOA) database: sharing knowledge in uniprot with gene ontology. *Nucleic Acids Res.*, 32(Database issue): D262–6, January 2004.

[259] Rachael P Huntley, Tony Sawford, Prudence Mutowo-Meullenet, Aleksandra Shypitsyna, Carlos Bonilla, Maria J Martin, and Claire O'Donovan. The GOA database: Gene ontology annotation updates for 2015. *Nucleic Acids Res.*, 43(D1):D1057–D1063, November 2014.

[260] L V D Maaten and Geoffrey E Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 2008.

[261] Lawrence Hubert and Phipps Arabie. Comparing partitions. *J. Classification*, 2(1):193–218, December 1985.

[262] Addie Woicik, Zachary R McCaw, Santiago Akle, Ben Dulken, Sanjana Narayanan, and Christopher Probert. In silico optimization of tissue microarray design for machine learning analysis. In *International Symposium on Biomedical Imaging*, 2024.

[263] David Cohn, Atlas, Les, and Richard Ladner. Improving generalization with active learning. *Mach. Learn.*, 15(2):201–221, May 1994.

[264] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, 69(6 Pt 2):066138, June 2004.

[265] Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. On the accuracy of influence functions for measuring group effects. *Adv. Neural Inf. Process. Syst.*, 32, 2019.

[266] Jean Drèze and Nicholas Stern. Chapter 14 the theory of cost-benefit analysis. In *Handbook of Public Economics*, volume 2, pages 909–989. Elsevier, January 1987.

[267] Odette Mariani, Caroline Brennetot, Jean-Michel Coindre, Nadège Gruel, Carine Ganem, Olivier Delattre, Marc-Henri Stern, and Alain Aurias. JUN oncogene amplification and overexpression block adipocytic differentiation in highly aggressive sarcomas. *Cancer Cell*, 11(4):361–374, April 2007.

[268] Jung Yun Bae, Kyung Un Choi, Ahrong Kim, So Jung Lee, Kyungbin Kim, Jee Yeon Kim, In Sook Lee, So Hak Chung, and Jeung Il Kim. Evaluation of immune-biomarker expression in high-grade soft-tissue sarcoma: HLA-DQA1 expression as a prognostic marker. *Exp. Ther. Med.*, 20(5):107, November 2020.

[269] Yingqi Hua, Xiaofang Jia, Mengxiong Sun, Longpo Zheng, Lin Yin, Lijun Zhang, and Zhengdong Cai. Plasma membrane proteomic analysis of human osteosarcoma and osteoblastic cells: revealing NDRG1 as a marker for osteosarcoma. *Tumour Biol.*, 32(5):1013–1021, October 2011.

[270] Xinrui Yan, Mei-Sze Chua, Hongbo Sun, and Samuel So. N-Myc down-regulated gene 1 mediates proliferation, invasion, and apoptosis of hepatocellular carcinoma cells. *Cancer Lett.*, 262(1):133–142, April 2008.

[271] Hongsheng Wang, Wei Sun, Mengxiong Sun, Zeze Fu, Chenghao Zhou, Chongren Wang, Dongqing Zuo, Zifei Zhou, Gangyang Wang, Tao Zhang, Jing Xu, Jian Chen, Zhuoying Wang, Fei Yin, Zhenfeng Duan, Francis J Hornicek, Zhengdong Cai, and Yingqi Hua. HER4 promotes cell survival and chemoresistance in osteosarcoma via interaction with NDRG1. *Biochim. Biophys. Acta Mol. Basis Dis.*, 1864(5 Pt A):1839–1849, May 2018.

[272] Jun Cheng, Hai-Yang Xie, Xiao Xu, Jian Wu, Xuyong Wei, Rong Su, Wu Zhang, Zhen Lv, Shusen Zheng, and Lin Zhou. NDRG1 as a biomarker for metastasis, recurrence and of poor prognosis in hepatocellular carcinoma. *Cancer Lett.*, 310(1):35–45, November 2011.

[273] Saskia Anna Graf, Markus Vincent Heppt, Anja Wessely, Stefan Krebs, Claudia Kammerbauer, Eva Hornig, Annamarie Strieder, Helmut Blum, Anja-Katrin Bosserhoff, and Carola Berking. The myelin protein PMP2 is regulated by SOX10 and drives melanoma cell invasion. *Pigment Cell Melanoma Res.*, 32(3):424–434, May 2019.

[274] Lijun Cheng, Pankita H Pandya, Enze Liu, Pooja Chandra, Limei Wang, Mary E Murray, Jacquelyn Carter, Michael Ferguson, Mohammad Reza Saadatzadeh, Khadijeh Bijangi-Visheshsaraei, Mark Marshall, Lang Li, Karen E Pollok, and Jamie L Renbarger. Integration of genomic copy number variations and chemotherapy-response biomarkers in pediatric sarcoma. *BMC Med. Genomics*, 12 (Suppl 1):23, January 2019.

[275] Qiaoge Guo, Hui Sun, Kunpeng Zheng, Shaojie Yin, and Junjie Niu. Long non-coding RNA DLX6-AS1/miR-141-3p axis regulates osteosarcoma proliferation, migration and invasion through regulating rab10. *RSC Adv.*, 9(58):33823–33833, 2019.

[276] International Cancer Genome Consortium et al. International network of cancer genome projects. *Nature*, 464(7291):993–998, April 2010.

[277] Jeffrey K Mito, Richard F Riedel, Leslie Dodd, Guy Lahat, Alexander J Lazar, Rebecca D Dodd, Lars Stangenberg, William C Eward, Francis J Hornicek, Sam S Yoon, Brian E Brigman, Tyler Jacks, Dina Lev, Sayan Mukherjee, and David G Kirsch. Cross species genomic analysis identifies a mouse model as undifferentiated pleomorphic sarcoma/malignant fibrous histiocytoma. *PLoS One*, 4(11): e8075, November 2009.

[278] Maria Capra, Paolo Giovanni Nuciforo, Stefano Confalonieri, Micaela Quarto, Marco Bianchi, Manuela Nebuloni, Renzo Boldorini, Francesco Pallotti, Giuseppe Viale, Mikhail L Gishizky, Giulio F Draetta, and Pier Paolo Di Fiore. Frequent alterations in the expression of serine/threonine kinases in human cancers. *Cancer Res.*, 66(16):8147–8154, August 2006.

[279] Poomy Pandey, Bailee Sliker, Haley L Peters, Amit Tuli, Jonathan Herskovitz, Kaitlin Smits, Abhilasha Purohit, Rakesh K Singh, Jixin Dong, Surinder K Batra, Donald W Coulter, and Joyce C Solheim. Amyloid precursor protein and amyloid precursor-like protein 2 in cancer. *Oncotarget*, 7 (15):19430–19444, April 2016.

[280] Guoliang Li, Liuyang Cai, Huidan Chang, Ping Hong, Qiangwei Zhou, Ekaterina V Kulakova, Nikolay A Kolchanov, and Yijun Ruan. Chromatin interaction analysis with Paired-End tag (ChIA-PET) sequencing technology and application. *BMC Genomics*, 15 Suppl 12(Suppl 12):S11, December 2014.

Chapter A

Appendix One

A.1 Sagittarius

A.1.1 Comparison with natural language processing transformer architecture

Sagittarius's transformer-based architecture is built around a continuous transformer formulation [78], which is inspired by the traditional transformer architecture [47] used in natural language processing (NLP). In the NLP context, the transformer attends to tokens (words) in the input sentence, comparing token similarity to encode the input (Figure A.1a). Importantly, the query and (key, value) pair are embeddings computed to represent the tokens (words). In the NLP transformer encoder, then, the attention mechanism functions within a sequence-to-sequence framework by comparing key- and query- embeddings representing different words. More similar keys and queries then have a larger contribution for the computed value to be associated with the query; keys that are dissimilar to the query will have a smaller contribution to the query's computed value.

The continuous transformer [78] builds upon this idea by using timepoint embeddings as the keys and queries in the transformer architecture (Figure A.1b). While the value still represents the measurement associated with the key's timepoint, this incorporates a continuous notion of time, including enabling a query during inference for a timepoint not used during model training. Sagittarius further extends the continuous transformer to include timepoint keys and queries that are also conditioned on the environmental variables,

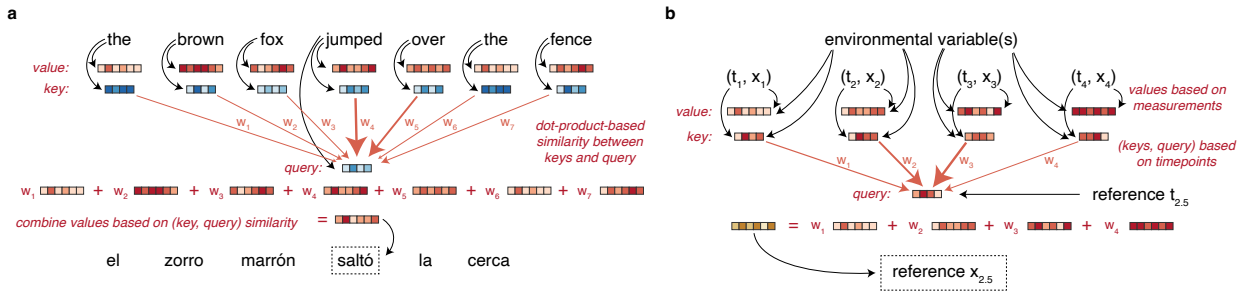


Figure A.1: **Illustration of attention mechanisms in different transformer architectures. a**, Conventional natural language processing transformer with self-attention for a toy machine translation task. The keys, queries, and values of the transformer's attention computation are all based on the words in the input. **b**, Conditional continuous transformer used in Sagittarius, where keys and queries are computed from the timepoints (and conditioned on the environmental or experimental variables), and the value is computed from the measurement and environmental variables. For simplicity, we show the encoder mapping to the reference space only; the decoder would proceed analogously.

enabling implicit time series alignment and accounting for varying temporal rates in the input trajectories.

A.1.2 Comparison approaches

Mean model

The mean model predicts data as $\mathbf{x_i}(t) = \frac{1}{T_i}\sum_{r=1}^{T_i}\mathbf{x_i}[r]$; that is, the predicted expression for each gene at any timepoint of interest t is the average of the gene expression across all measured timepoints.

Linear model

The linear baseline model defines a weight

$$\lambda_i(t) = \begin{cases} 0 & \text{if } t < \min(\mathbf{t_i}) \\ 1 & \text{if } t > \max(\mathbf{t_i}) \\ \max_{t_0 \in \mathbf{t_i}: t_0 \leq t} \min_{t_1 \in \mathbf{t_i}: t_1 \geq t} \left(\frac{t-t_0}{t_1-t_0}\right) & \text{otherwise.} \end{cases} \tag{A.1}$$

Then, the linear model predicts expression at time t as

$$\hat{\mathbf{x}}_\mathbf{i}(t) = \max_{r_0 \in \{1,\ldots,T_i\}: \mathbf{t_i}[r_0] \leq t} (1-\lambda_i(t))\mathbf{x_i}[r_0] + \min_{r_1 \in \{1,\ldots,T_i\}: \mathbf{t_i}[r_1] \geq t} \lambda_i(t)\mathbf{x_i}[r_1]. \tag{A.2}$$

Note that, in the extrapolation setting, the linear baseline therefore predicts a gene expression vector identical to the expression vector of the nearest temporal measurement.

Neural ODE model

We trained a set of neural ordinary differential equation (ODE) models [76] that each model measurements from a single combination of environmental variables. As the environmental conditions $\mathbf{y_i}$ are constant within a single sequence i, we drop the subscript and reduce the task inputs to $(\mathbf{x}, \mathbf{t})$. We compute an ODE

for both the forward and backward direction of the sequence as

$$\tilde{\mathbf{x}_{\rightarrow}}(t) = \max_{r \in \{1,\ldots,T\}:\mathbf{t}[r] \leq t} \mathbf{x}[r] + \int_{\tau=r}^{t} f_\theta(\mathbf{x}[\tau])d\tau \quad \tilde{\mathbf{x}_{\leftarrow}}(t) = \min_{r \in \{1,\ldots,T\}:\mathbf{t}[r] \geq t} \mathbf{x}[r+1] + \int_{\tau=r}^{t} g_\phi(\mathbf{x}[\tau])dt. \tag{A.3}$$

In the case where $\tilde{\mathbf{x}_{\rightarrow}}$ or $\tilde{\mathbf{x}_{\leftarrow}}$ requires extrapolation (i.e., $t < \min(\mathbf{t})$ or $t > \max(\mathbf{t})$), we set $\tilde{\mathbf{x}_{\rightarrow}} = \tilde{\mathbf{x}_{\leftarrow}}$. In order to empirically compute the integrals we used the python `torchdiffeq` package [76, 77], and parameterized $f_\theta(\cdot)$ and $g_\phi(\cdot)$ using multi-layer perceptrons (MLPs). Finally, we combined the forward and backward results to produce the final estimate

$$\hat{\mathbf{x}}(t) = \frac{1}{2}(\tilde{\mathbf{x}_{\rightarrow}}(t) + \tilde{\mathbf{x}_{\leftarrow}}(t)). \tag{A.4}$$

We trained the model using the Adam optimizer [80] and a batch size of 1 time series.

RNN model

We learned a set of single-sequence bidirectional gated recurrent unit (GRU)[75] models to learn the dynamics for a single $(\mathbf{x_i}, \mathbf{y_i}, \mathbf{t_i})$ sequence, again dropping the subscript notation and reducing the problem input to $(\mathbf{x}, \mathbf{y})$. We define a time step $\Delta_t = 1$ between observations of interest. We computed $\mathbf{z_r} = q_\phi(\mathbf{x}[r])$ for an MLP $q_\phi(\cdot)$ as the embedding for each observation in the time series, and computed

$$\begin{aligned}
&\mathbf{h_0}^{\rightarrow} = 0 \\
&\hat{\mathbf{z}_\tau}^{\rightarrow} = q_\phi(p_\theta(\mathbf{h}_\tau{}^{\rightarrow})) \\
&\mathbf{h}_{\tau+\mathbf{\Delta_t}}{}^{\rightarrow} = \begin{cases} g_\xi^{(gru\rightarrow)}(\mathbf{z}_\tau, \mathbf{h}_\tau{}^{\rightarrow}) & \text{if } \tau \in \mathbf{t} \\ g_\xi^{(gru\rightarrow)}(\hat{\mathbf{z}_\tau}, \mathbf{h}_\tau{}^{\rightarrow}) & \text{otherwise} \end{cases}
\end{aligned} \tag{A.5}$$

for an MLP $p_\theta(\cdot)$. Similarly, we define the backward GRU as

$$\begin{aligned}
&\mathbf{h_T}^{\leftarrow} = 0 \\
&\hat{\mathbf{z}_\tau}^{\leftarrow} = q_\phi(p_\theta(\mathbf{h}_\tau{}^{\leftarrow})) \\
&\mathbf{h}_{\tau-\mathbf{\Delta_t}}{}^{\leftarrow} = \begin{cases} g_\xi^{(gru\leftarrow)}(\mathbf{z}_\tau, \mathbf{h}_\tau{}^{\leftarrow}) & \text{if } \tau \in \mathbf{t} \\ g_\xi^{(gru\leftarrow)}(\hat{\mathbf{z}_\tau}, \mathbf{h}_\tau{}^{\leftarrow}) & \text{otherwise.} \end{cases}
\end{aligned} \tag{A.6}$$

Finally, we combine the forward- and backward directions to produce the simulated gene expression profile

$$\hat{\mathbf{x}}(t) = p_\theta(\frac{1}{2}(\mathbf{h_t}^{\rightarrow} + \mathbf{h_t}^{\leftarrow})). \tag{A.7}$$

While this formulation is sufficient for our experiments, note that this does limit interpolation and extrapolation to units of Δ_t. We trained the model end-to-end with the Adam optimizer [80] and a batch size of 1 time series.

mTAN model

We trained a discretized multi-time attention network (mTAN) [78] using the Adam optimizer [80]. As the mTAN module does not handle environmental variables, for each time series $(\mathbf{x_i}, \mathbf{y_i}, \mathbf{t_i})$ the model received the reduced input $(\mathbf{x_i}, \mathbf{t_i})$.

cVAE model

We trained a conditional variational autoencoder (cVAE) [23] to learn $p(\mathbf{x_i}[r]|\mathbf{y_i}[r], \mathbf{t_i}[r])$. We trained the model using the Adam optimizer [80] and a batch size of 512 gene expression measurements. This is the only model that can be applied to multiple continuous variables out-of-the box, so we use it as the comparison method for the LINCS experiments.

CPA model

We trained a compositional perturbation autoencoder (CPA) [74] with the Adam optimizer [80] and a batch size of 128. In the Evo-devo experiments, we considered the time to be independent of the organ label (the

covariate) and dependent on the species label (the perturbation).

A.1.3 Hyperparameter search

Evo-devo

For each deep learning model, we randomly select 20% of the measurements available at training time to use as validation data, which we use for hyperparameter tuning and early training termination, where we stop training when the validation loss has not reached a new minimum for 250 epochs. For each comparison approach, we conducted an approximate hyperparameter grid search on the late-timepoint Evo-devo extrapolation task, selecting 50 experimental configurations per model from a set of possible values per hyperparameter. Given some redundant experimental configurations, this resulted in 49 unique experiments for cVAE, 50 for CPA, 50 for mTAN, 42 for RNN, 22 for Neural ODE, and 50 for PRESCIENT. We selected the best model based on Pearson correlation comparing genes in the validation set. We then used the same hyperparameters for all Evo-devo experiments, including extrapolation to early timepoints. Selected hyperparameters for each method are shown in Table A.1.

We conducted a reduced hyperparameter grid search for the Sagittarius model, again using the validation dataset for both early stopping and hyperparameter selection. We fixed the embedding dimension to $d = 32$, learning rate to $\eta = 1\text{e}{-3}$, autoencoder width to 1024, number of heads to $H = 8$, number of reference points to $S + 1 = 4$, and time embedding dimension to $d_{temp} = 4$. We then searched over an autoencoder depth $D \in \{2, 3\}$; regularization weight $\beta \in \{0, \frac{1}{6}, 1\}$; autoencoder categorical embedding dimension $d_{yae} \in \{2, 8\}$; and transformer categorical embedding dimension $d_{ytr} \in \{4, 8\}$. Based on the validation performance, we selected an autoencoder depth $D = 3$, regularization $\beta = \frac{1}{6}$, and autoencoder and transformer categorical embedding dimensions $d_{yae} = 2$ and $d_{ytr} = 4$.

We then conducted an ablation study for different configuration results to investigate Sagittarius's robustness to various hyperparameter settings. Figures A.2, A.3, and A.4 show that Sagittarius's test performance for the late development extrapolation experiment under different model hyperparameter settings. We emphasize that this study was conducted after hyperparameters were fixed for all of Sagittarius's Evo-devo experiments, and that the test results were never used outside of the ablation. In the ablation study, we tuned one hyperparameter while setting the remainder to our optimal configuration as determined by the validation

Hyperparameter	Tested values	Model	Best value
Embedding dimension d	$\{8, 16, 32, 64, 128\}$	cVAE	128
		CPA	8
		mTAN	64
		RNN	16
Learning rate η	$\{1\text{e}{-4}, 1\text{e}{-3}, 1\text{e}{-2}\}$	cVAE	$1\text{e}{-3}$
		CPA (autoencoder)	$1\text{e}{-4}$
		mTAN	$1\text{e}{-3}$
		RNN	$1\text{e}{-4}$
		Neural ODE	$1\text{e}{-4}$
		PRESCIENT	$1\text{e}{-4}$
Autoencoder width W	$\{512, 1024, 2046\}$	cVAE	1024
		CPA	1024
		RNN	1024
		PRESCIENT	1024
Autoencoder depth D	$\{1, 2, 3\}$	cVAE	1
		CPA	3
		RNN	1
		PRESCIENT	1
Time step Δ_t	$\{0.1, 0.5, 1.0\}$	Neural ODE	0.1
		PRESCIENT	0.1
Regularization weight β	$\{0, \frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}, 1\}$	cVAE	0
Additional learning rate	$\{1\text{e}{-4}, 1\text{e}{-3}\}$	CPA (adversary)	$1\text{e}{-4}$
		CPA (doser)	$1\text{e}{-4}$
Adversary width	$\{128, 256, 512\}$	CPA	256
Adversary depth	$\{1, 2, 3\}$	CPA	1
Adversary steps	$\{2, 4, 8\}$	CPA	2
Doser type	{logsigm, sigm, mlp}	CPA	logsigm
Learnable temporal embedding	{true, false}	mTAN	true
Attention heads H	$\{4, 8, 12\}$	mTAN	4
Time embedding dimension d_{temp}	$\{1, 2, 3\}$	mTAN	2
ODE depth	$\{1, 2, 3\}$	Neural ODE	3
Sinkhorn blur	$\{0.1, 0.5, 1.0\}$	PRESCIENT	0.5
Sinkhorn scaling	$\{0.25, 0.5, 0.75\}$	PRESCIENT	0.75
Pretraining epochs	$\{50, 100, 200\}$	PRESCIENT	200
Train τ	$\{0, 1\text{e}{-6}, 1\text{e}{-4}\}$	PRESCIENT	0
Gradient clipping	$\{0, 0.25, 0.5\}$	PRESCIENT	0

Table A.1: **Evo-devo hyperparameter settings for different comparison approaches.** The values considered during the hyperparameter sweep are shown along with the final hyperparameter value selected based on the best performance on validation data. Not all hyperparameters are relevant for all models. For CPA, we also considered MLP doser width and depth in $\{16, 32, 64\} \times \{1, 2\}$, but did not use these in the final configuration as the logsigm doser was chosen.

set. We found that Sagittarius's improvement over existing approaches was robust to several hyperparameter settings, including the number of attention heads H, the transformer's environmental variable embedding dimension d_{ytr}, and the number of generative datapoints used to train the model.

Unsurprisingly, we found that Sagittarius was sensitive to choice of learning rate with respect to all of the evaluation metrics that we measured. We also found that the regularization weight β impacted model performance; in the ablation study, we considered $\beta \in \{0.0, 0.1667, 1.0\}$, where $0.1667 = \frac{m}{N}$, where $m = 8$ was the batch size and $N = 48$ was the original dataset size. This finding suggests that both under-regularizing and over-regularizing the model leads to worse performance. Sagittarius's performance is also impacted by the choice of autoencoder structure and latent embedding dimension, perhaps reflecting the need for a relatively tight information bottleneck ($d < 64$) given the limited size of the dataset, and the importance of disentangling non-linear species- and organ-specific patterns from the gene expression measurements before reaching the reference space.

LINCS

For Sagittarius and the cVAE model, we randomly partitioned the into an 80% training, 10% validation, and 10% test split. We then further excluded any data included in one of the test settings from the training and validation data. We terminated model training when the validation loss had not decreased for at least 100 epochs and returned the model with lowest validation loss. For the Sagittarius and cVAE models, we set the embedding dimension $d_{cvae} = d_{sag} = 16$, learning rate $\eta_{cvae} = \eta_{sag} = 1\text{e}{-3}$, and the autoencoder width and depth $(W, D)_{cvae} = (W, D)_{sag} = (128, 2)$. We set the regularization weight $\beta_{cvae} = 1.0$ and $\beta_{sag} = 0.25$. For Sagittarius, we further set the number of attention heads $H = 8$, total number of reference points as 16 (spanning both dose and time, as a 4×4 grid in the reference space), time embedding dimension d_{temp} to 8 for dose and 4 for time, and both autoencoder and transformer categorical embedding dimensions as $d_{yae} = d_{ytr} = 8$.

TCGA

For each of the deep learning models, we used 20% of the available data at train time as a validation set for training termination and hyperparameter selection where applicable. We set a latent embedding dimension

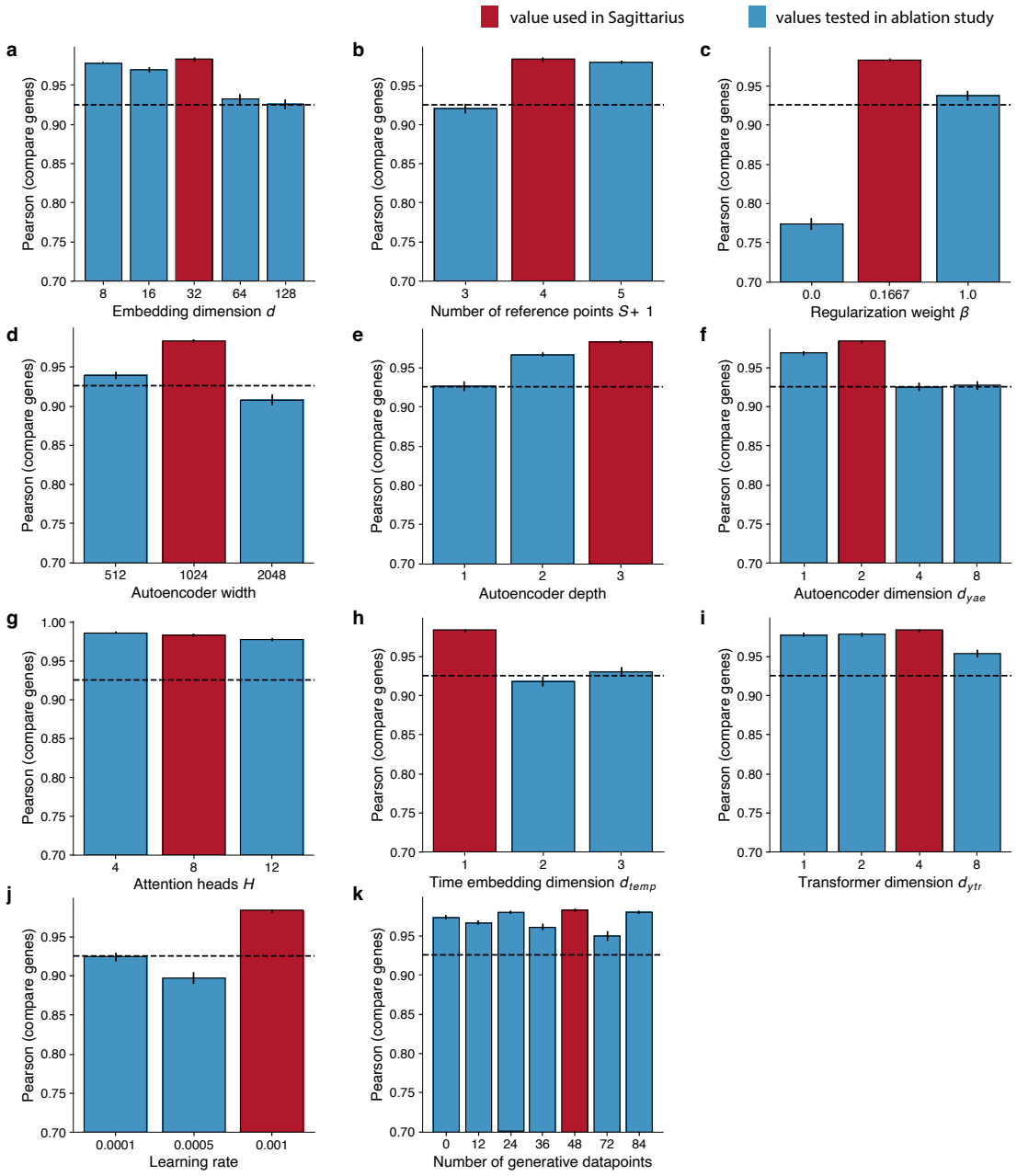


Figure A.2: **Test Pearson correlation comparing genes for different choices of Sagittarius hyperparameters on the Evo-devo late timepoint extrapolation task. a-k**, Bar plot comparing Sagittarius's performance when holding all but one hyperparameter fixed and varying a single hyperparameter. Data are presented as mean values $\pm$ standard error, with $n = 48$ species and organ time series. The dotted line shows the average Pearson correlation comparing timepoints for the best-performing baseline for reference. Higher values indicate better performance.

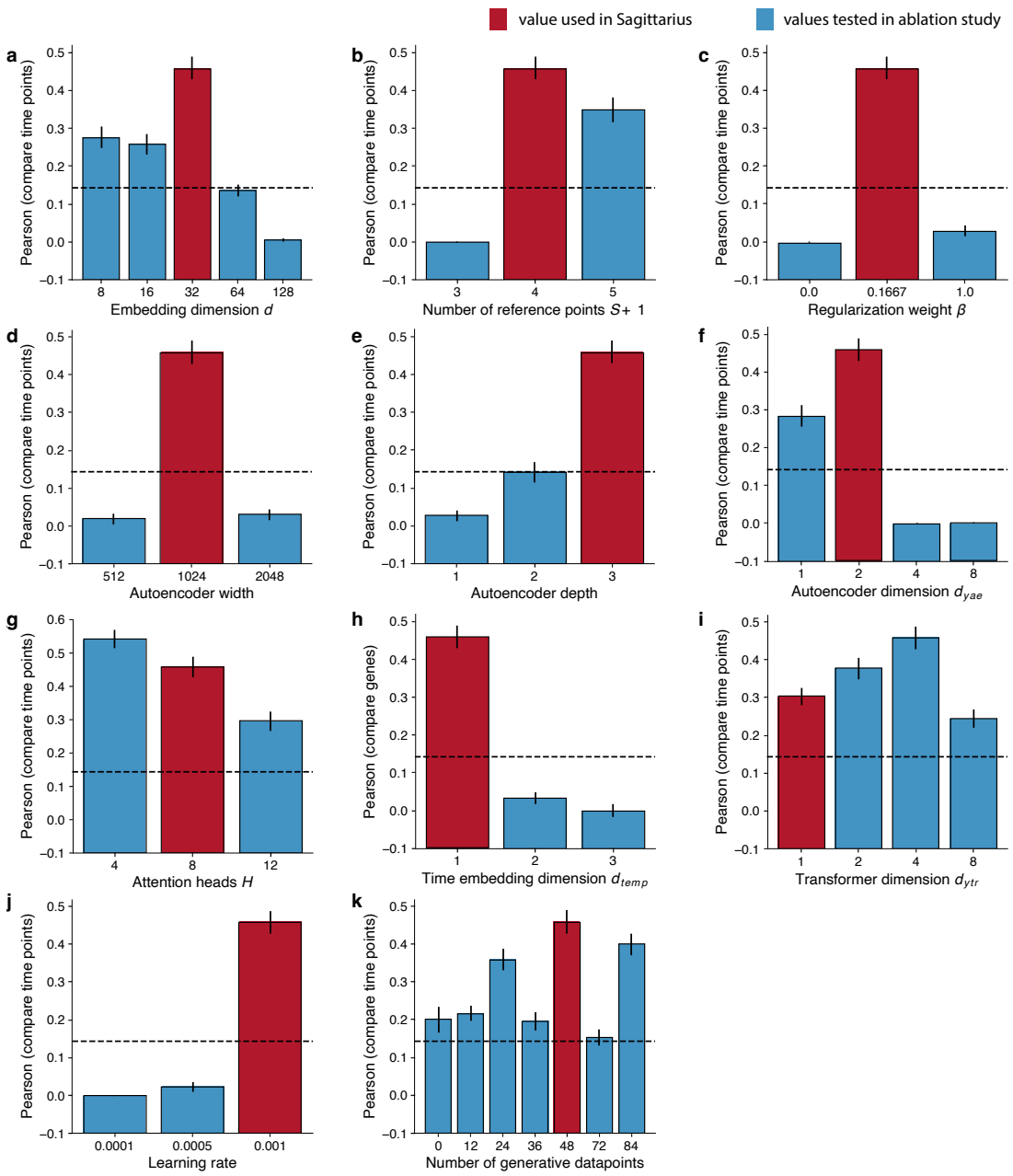


Figure A.3: **Test Pearson correlation comparing timepoints for different choices of Sagittarius hyperparameters on the Evo-devo late timepoint extrapolation task. a-k**, Bar plot comparing Sagittarius's performance when holding all but one hyperparameter fixed and varying a single hyperparameter. Data are presented as mean values $\pm$ standard error, with $n = 48$ species and organ time series. The dotted line shows the average Pearson correlation comparing timepoints for the best-performing baseline for reference. Higher values indicate better performance.

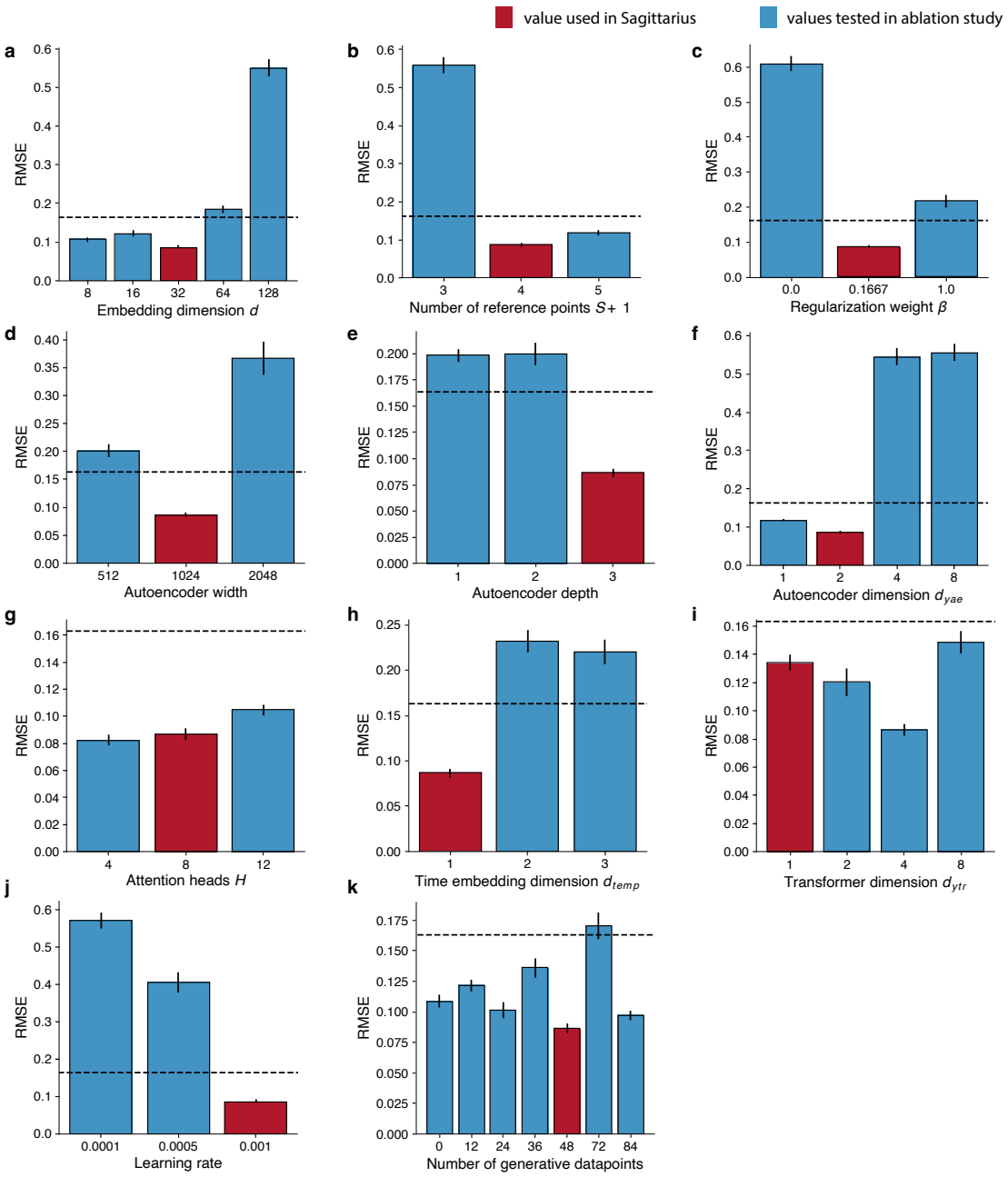


Figure A.4: **Test root mean squared error comparing timepoints for different choices of Sagittarius hyperparameters on the Evo-devo late timepoint extrapolation task. a-k**, Bar plot comparing Sagittarius's performance when holding all but one hyperparameter fixed and varying a single hyperparameter. Data are presented as mean values $\pm$ standard error, with $n = 48$ species and organ time series. The dotted line shows the average Pearson correlation comparing timepoints for the best-performing baseline for reference. Lower values indicate better performance.

$d_{rnn} = d_{sag} = 16$. We set the learning rate $\eta_{rnn} = \eta_{node} = 1\text{e}{-4}$, and $\eta_{sag} = 1\text{e}{-3}$. For the RNN, we used an autoencoder of width of 256 and depth of 1. For the Neural ODE, we set the timestep $\Delta_t = 1$ and ODE depth of 1. For Sagittarius, we set the regularization weight $\beta = 1$, number of attention heads $H = 8$, number of reference points $S + 1 = 4$, temporal embedding dimension $d_{temp} = 2$, and the autoencoder and transformer categorical embedding dimensions $d_{yae} = 8$ and $d_{ytr} = 4$ respectively. We further set the autoencoder width to be 256, and then searched over depths in $\{1, 2, 3\}$, using the validation data to select a depth 3 and depth 2 autoencoders for the THCA and SARC experiments respectively.

A.1.4 Additional experimental results

Here, I include some additional experimental results for the Sagittarius model.

Evo-devo

Aggregated results for extrapolation to early developmental timepoints are shown in Figure A.5. A specific comparison to the single-cell fate prediction method PRESCIENT on late-timepoint extrapolation is shown in Figure A.6. A visualization of simulated mouse organogenesis using PCA low-dimensional projections is shown in Figure A.7.

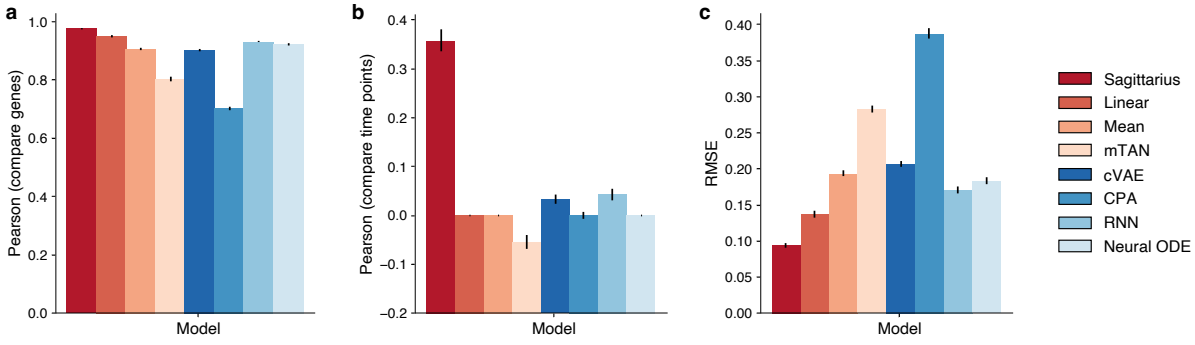


Figure A.5: **Overall gene expression extrapolation performance for Evo-devo extrapolation to early timepoints. a-c**, Bar plot of Pearson correlation comparing genes (**a**), Pearson correlation comparing timepoints (**b**), and RMSE (**c**) of the predicted expression profile and measured expression profile when extrapolating to the first four measured timepoints from each species and organ combination in the Evo-devo dataset for Sagittarius and the comparison approaches. For Pearson correlation, comparing genes or comparing timepoint (**a,b**), higher values indicate better performance; for RMSE (**c**), lower values indicate better performance. Data are presented as mean values $\pm$ standard error, with $n = 48$ species and organ time series.

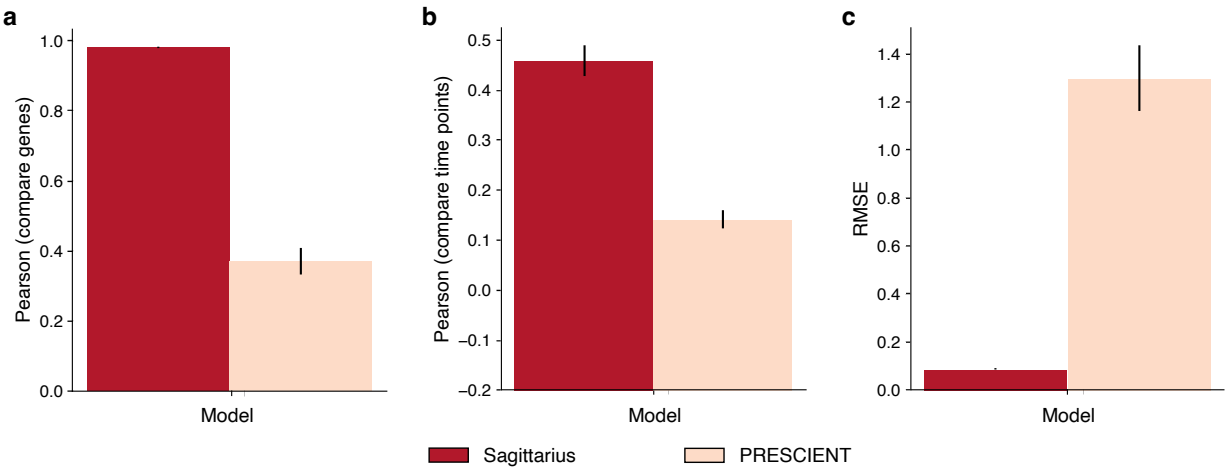


Figure A.6: **Comparison of Sagittarius and PRESCIENT [148] models for late timepoint Evo-devo extrapolation performance. a-c**, Bar plot of Pearson correlation comparing genes (**a**), Pearson correlation comparing timepoints (**b**), and RMSE (**c**) of the predicted expression profile and measured expression profile when extrapolating to the last four measured timepoints from each species and organ combination in the Evo-devo dataset for Sagittarius and the single-cell fate prediction method PRESCIENT. For Pearson correlation (**a,b**), higher values indicate better performance; for RMSE (**c**), lower values indicate better performance. Data are presented as mean values $\pm$ standard error, with $n = 48$ species and organ time series.

LINCS

Additional analyses for the LINCS experiments are included here. Figures A.8 and A.9 show the frequency of the best-performing drugs and cell lines in the Sagittarius-augmented GDSC [92] and CTRP [94] drug sensitivity experiments. Similarly, Figure A.10 shows the frequency of the best- and worst-performing cell lines in the Sagittarius-augmented DepMap [96] cancer gene essentiality experiments.

TCGA

Figure A.11 shows the number of patients retained by our noisy label learning [83, 84] approach towards patients with a censored event time, and Figure A.12 shows the error distribution specifically for the THCA cancer type, where no censored patients are retained for the later analyses. Figure A.13 compares Sagittarius's somatic mutation profile extrapolation performance to other deep learning methods in the quantitative evaluation setting. Figure A.14 shows the distribution of event times in the processed sarcoma dataset and the ten survival times used for extrapolation during the case study of sarcoma tumorigenesis.

Furthermore, we considered cancer patient gene expression extrapolation (Figure A.15). To investigate

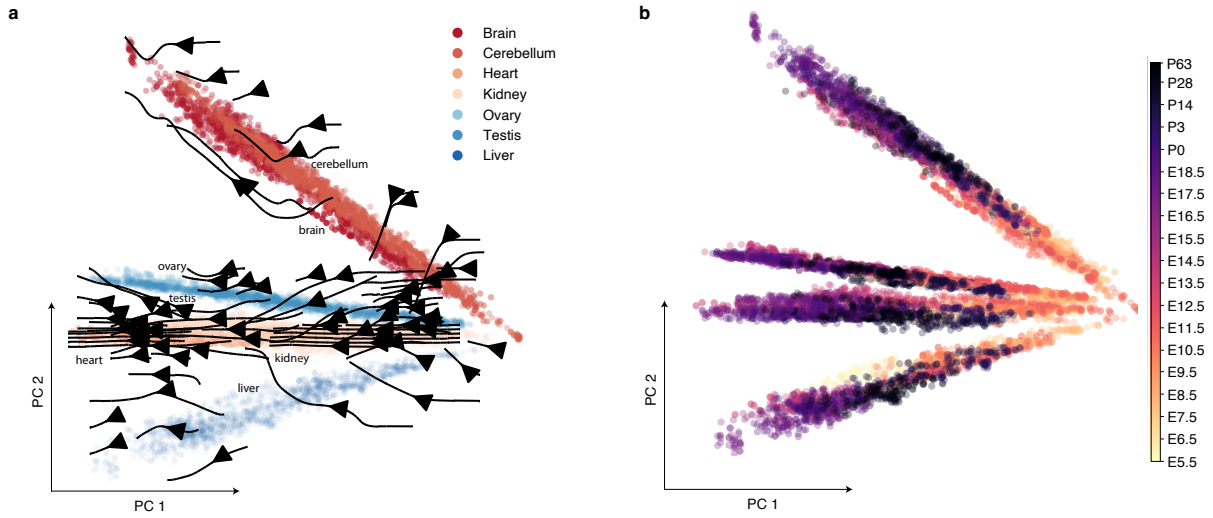


Figure A.7: **Mouse transcriptomic velocity across organs. a,b**, PCA plot showing extrapolated mouse gene expression from E5.5 to P63 for 7 organs, colored by organ (**a**) and time (**b**). The arrows in (**a**) indicate the transcriptomic velocity of each organ. The first PC shows most variation with respect to time, while the second shows most variation with respect to organ. Organ annotations in (**a**) are added to help differentiate between organs, especially in the case of overplotting.

the transcriptional differences between early- and late-stage sarcoma patients, we trained Sagittarius on all available TCGA RNA-seq profiles and then simulated new expression profiles for two synthetic sarcoma patients, one with post-diagnosis survival $t = 0$ and the other with $t = 244.91$, representing extremely late-stage and early-stage tumors respectively. We the identified the $k = 10$ genes that were most overexpressed in the late-stage tumor compared to the early-stage tumor as *PMP2*, *NDRG1*, *JUN*, *SEPT9*, *PHC2*, *NCAM1*, *GFAP*, *APP*, *WNK1*, and *RAB10*. Previous work identified *JUN* [267], *NCAM1* [268], *NDRG1* [269–272], and *PMP2* [273, 274] in aggressive sarcomas with poor prognosis and short time-to-metastasis. Furthermore, other work has found *RAB10* [275], *SEPT9* [276], *PHC2* [277], *WNK1* [278], and *APP* [279] overexpression in sarcomas.

Finally, we provide more literature evidence connecting Sagittarius's extrapolated mutation profile for early stage sarcoma patients to the hedgehog (HH) signaling pathway and *GLI* oncogene.

- *PTCH1*: Tumor suppressor gene in the HH signaling pathway, which has been linked to medulloblastoma [128], plexiform fibromyxoma [129], and basal cell carcinoma [130]. Loss-of-function mutations have been documented to lead to aberrant activation of the HH pathway and subsequent

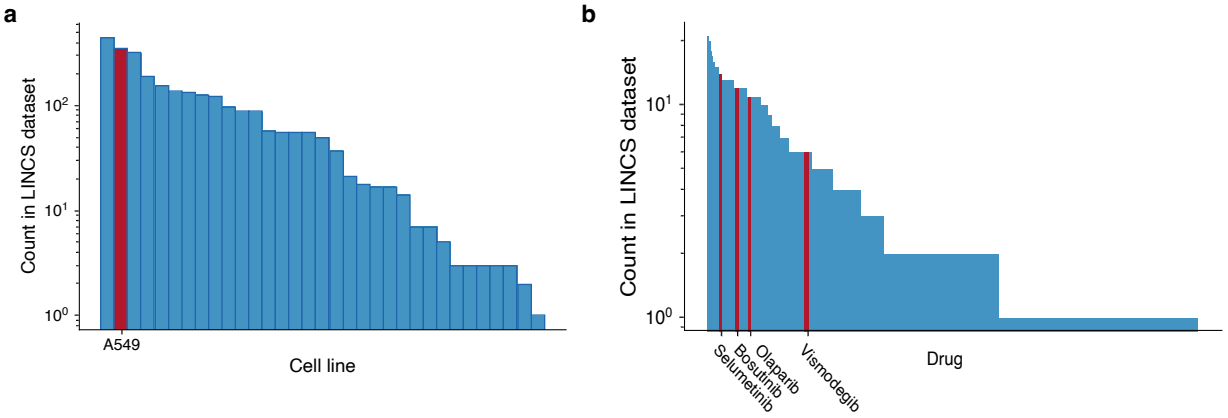


Figure A.8: **LINCS measurements with the best-performing cell line and drugs for the IC_{50} prediction task with the GDSC dataset. a,b**, Histogram of the number of measured drug treatments per cell line (**a**) and cell lines treated per drug (**b**) in the LINCS dataset. The A549 cell line is annotated as the cell line with the most improved predictions from Sagittarius's imputed dataset (**a**); Selumetinib, Bosutinib, Olaparib, and Vismodegib are annotated in (**b**) as the drugs with the most-improved predictions.

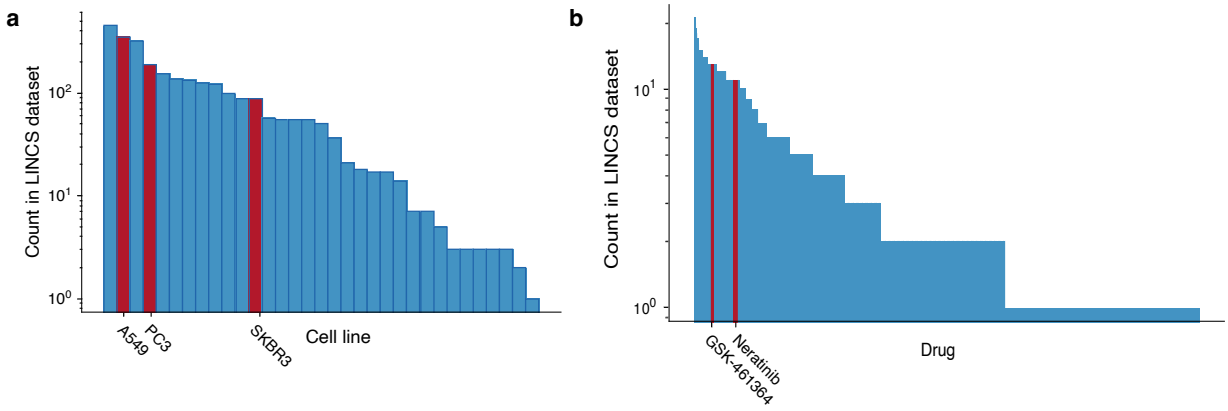


Figure A.9: **LINCS measurements with the best-performing cell line and drugs for the IC_{50} prediction task with the CTRP dataset. a,b**, Histogram of the number of measured drug treatments per cell line (**a**) and cell lines treated per drug (**b**) in the LINCS dataset. The A549 and PC3 cell lines are annotated as the cell lines with the most improved predictions from Sagittarius's imputed dataset; SKBR3, which Sagittarius struggles on, is also annotated (**a**). GSK-461364 and Neratinib are annotated in (**b**) as the most-improved drugs with Sagittarius's imputed dataset.

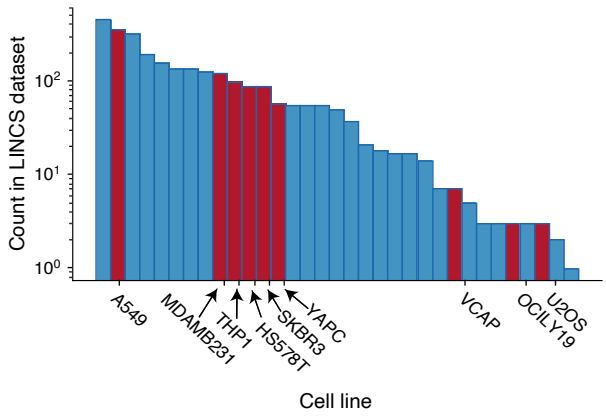


Figure A.10: **LINCS measurements with the best-performing cell lines for the gene essentiality prediction task with the DEMETER and CERES DepMap datasets.** Histogram of the number of measured drug treatments per cell line in the LINCS dataset. Sagittarius's imputed dataset provided the most benefit to the A549, MDAMB231, THP1, HS578T, SKBR3, YAPC, VCAP, OCILY19, and U2OS cell lines, which are annotated.

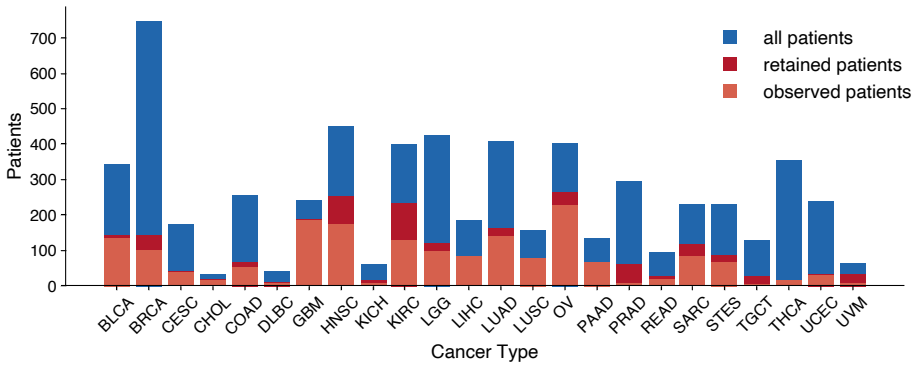


Figure A.11: **Distribution of TCGA patients per cancer type in the processed mutation dataset.** Bar plot comparison of patient counts per cancer type if all patients are included in the analysis, if only patients retained by the noisy label learning model (including all patients with an observed death event and a subset of censored patients) are included in the analysis, and if only patients with an observed death event are included in the analysis.

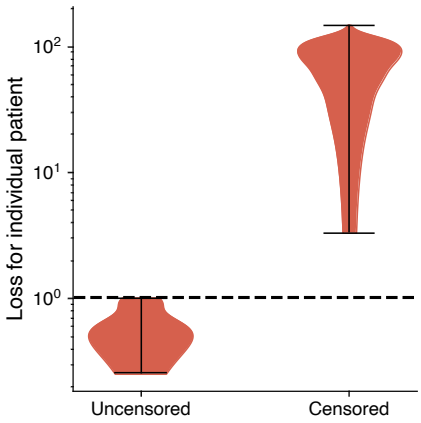


Figure A.12: **THCA censored patient analysis.** Violin plot of the survival regressor's absolute error for each THCA patient, subdivided into an observed group and a censored group.

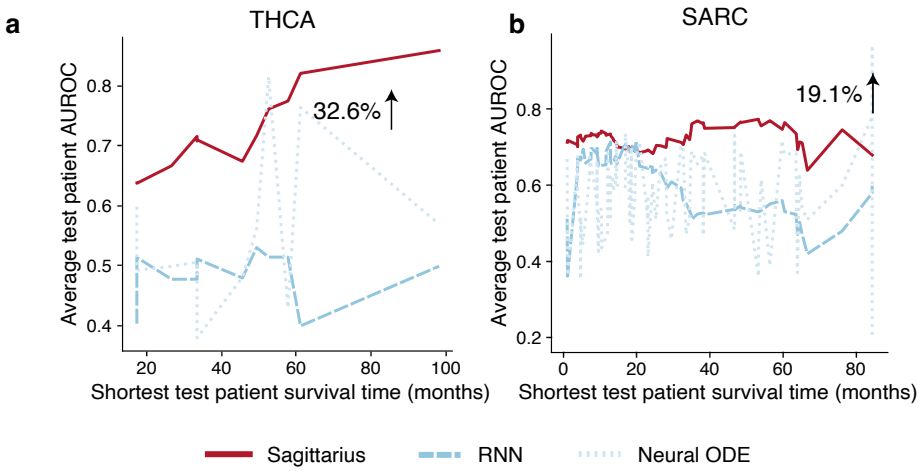


Figure A.13: **Patient somatic mutation profile extrapolation performance for Sagittarius and other deep learning models. a,b**, Line plot comparing the average test patient AUROC for each of the THCA (**a**) and SARC (**b**) cancer type test splits, ordered according to the shortest survival time in that test set. The annotations indicate Sagittarius's percent average AUROC improvement over the best-performing deep learning comparison approach.

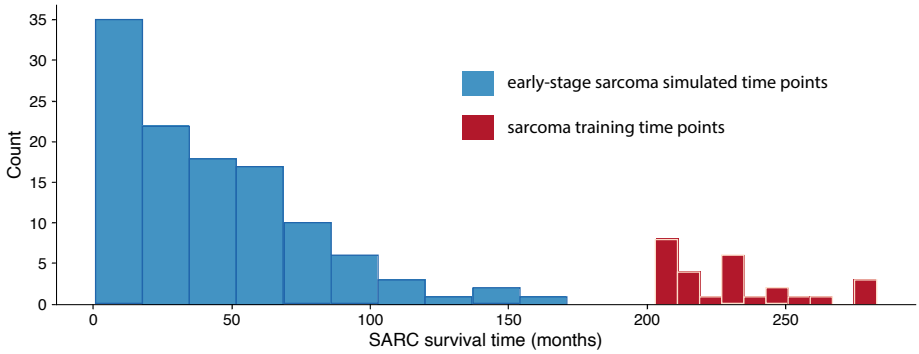


Figure A.14: **SARC training and extrapolation timepoint distributions.** Histogram showing the measured survival time of patients in the SARC time series as the available sarcoma training data and the extrapolation timepoints used to simulate the expression profile of an early-stage sarcoma patient.

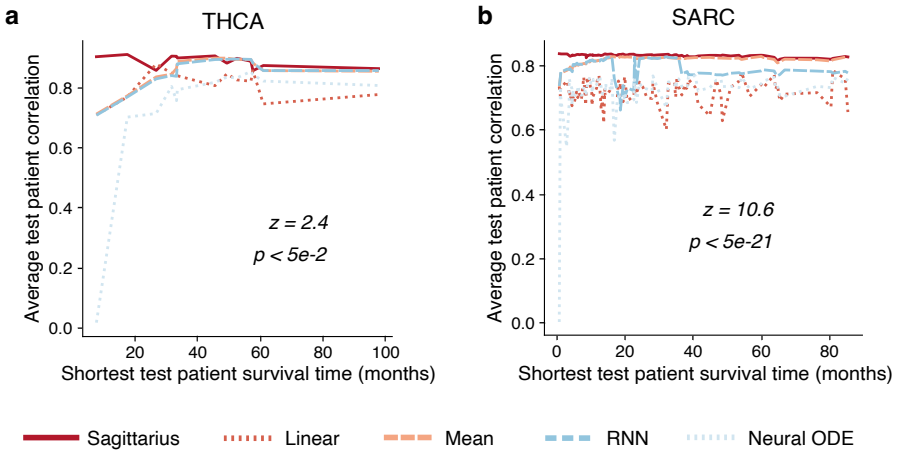


Figure A.15: **Performance comparison for cancer patient gene expression extrapolation at the time of diagnosis. a,b**, Line plot showing average Spearman correlation of predicted expression profile and measured expression profile for THCA (**a**) and SARC (**b**) test splits, ordered according to the shortest survival time in the test set. Annotations indicate the z- and p-value comparing Sagittarius to the best-performing baseline (one-sided Fisher z-transformed test), with p-value < 0.011 for THCA and p-value $< 3.92\text{e}{-21}$ for SARC.

Cell line	Weight vector	$\mathbf{X}_{(v=0)}$	$\mathbf{X}^{(oe)}_{(v=1)}$	$\mathbf{X}^{(tad)}_{(v=2)}$	$\mathbf{X}^{(loop-p)}_{(v=3)}$	$\mathbf{X}^{(loop-r)}_{(v=4)}$
GM12878	ω_0	1.00	7.30e−2	9.83e−3	1.59e−1	3.10e−2
	ω	1.00	7.30e−2	8.44e−3	3.81e−2	1.82e−2
K562	ω_0	1.00	3.58e−2	9.99e−3	1.37e−1	2.29e−2
	ω	1.00	3.58e−2	9.69e−3	4.92e−2	1.39e−2

Table A.2: **Capricorn's computed weights for each view when training on either GM12878 or K562.** The initial, distribution-based weights ω_0 and further refined, difficulty-based weights ω are reported. For training the final model, we use the ω weight vector. The v indicates the channel index in the image-like input for each view.

tumorigenesis [128].

- *MYCBP2*: Encodes a protein that promotes degredation of the *MYC* oncogene [132], which directly regulates *GLI1* in Burkitt lymphoma cell lines [133]. Some studies have found that lymphoblastic leukemia patients had high *c-MYC* expression and low *MYCBP2* expression [131].

- *ARID2*: Encodes a protein that directly interacts with *GLI1* [135] as a core subunit of the SWI/SNF chromatin remodeling complex [134, 135].

- *PREX1*: Member of the PI3K-Akt signaling pathway, which has been associated with *GLI* code regulation [138] and cross-talk with the HH signaling pathway in melanoma [130, 139].

- *DNAH17*: Encodes a protein that makes up a subunit of the primary cilium's basic structure [136]. The primary cilia have been shown to be both positive and negative effectors of the HH signaling pathway [136, 137].

- *EGFLAM*: Induces activation of the PI3K-Akt signaling pathway [140], which includes *PREX1*.

- *NLGN1*: Found to be significantly enriched with the HH pathway in a colorectal carcinoma study [141].

A.2 Capricorn

A.2.1 Multi-view weighting

The exact weights determined by our initial pass on the validation loss are shown in Table A.2.

Cell line	Loop caller	Loops called
GM12878	HiCCUPS	10,176
	Mustache	15,417
	Chromosight	22,648
K562	HiCCUPS	5136
	Mustache	7776
	Chromosight	10,630

Table A.3: **Number of loops called from the original high-coverage data across loop calling methods and cell lines.**

A.2.2 Additional experimental results

Cross-cell-line loop statistics

In addition to the loop F_1 score, here we report the total number of loops called and the precision and recall of the called loops, still treating loops called from the high-coverage data as the ground truth. We report results using the HiCCUPS [11], Mustache [175], and Chromosight [192] loop calling tools for evaluation; in all cases, Capricorn's loop-based views were computed using the HiCCUPS algorithm. For reference, the total number of loops called from the original high-coverage Hi-C matrices is provided in Table A.3. The performance metrics for all three loop calling algorithms given the low-coverage (LC) data and the enhanced matrices produced by Capricorn (ours), HiCNN [157], HiCARN-1 [161], HiCARN-2 [161], and HiCSR [160] are shown in Table A.4.

Capricorn view ablation study

In addition to applying our multi-view framework to the other comparison approaches, we also conducted an ablation study over the views included in Capricorn. Specifically, we considered the 8 different approaches listed in Table A.5, still using Capricorn's diffusion model backbone for resolution enhancement.

We then compared the loop F_1 score and MSE in the cross-cell-line setting for all ablated methods. Results are given in Table A.5.

Cell line	Loop caller	Method	Loops called	F_1 score	Precision	Recall
GM12878	HiCCUPS	LC	7021	0.32	0.41	0.27
		HiCNN	10,625	0.28	0.28	0.28
		HiCARN-1	9035	0.41	0.44	0.39
		HiCARN-2	8793	0.43	0.46	**0.40**
		HiCSR	7608	0.45	0.53	0.40
		Capricorn	5686	**0.50**	**0.70**	0.39
	Mustache	LC	1483	0.18	**0.97**	0.10
		HiCNN	37,218	0.36	0.25	0.64
		HiCARN-1	34,236	0.40	0.29	0.66
		HiCARN-2	36,301	0.40	0.28	**0.68**
		HiCSR	34,782	0.41	0.29	**0.68**
		Capricorn	33,109	**0.42**	0.30	**0.68**
	Chromosight	LC	1495	0.09	**0.68**	0.05
		HiCNN	87,693	0.23	0.15	0.53
		HiCARN-1	100,370	0.22	0.14	0.56
		HiCARN-2	101,953	0.23	0.14	0.57
		HiCSR	100,011	0.22	0.14	0.57
		Capricorn	83,058	**0.28**	0.18	**0.61**
K562	HiCCUPS	LC	4668	0.22	0.28	0.20
		HiCNN	9117	0.17	0.20	0.20
		HiCARN-1	7879	0.20	0.19	0.24
		HiCARN-2	5406	0.24	0.27	0.23
		HiCSR	4675	0.23	0.31	0.21
		Capricorn	2144	**0.35**	**0.60**	**0.25**
	Mustache	LC	17	0.00	**0.36**	0.00
		HiCNN	30,870	0.24	0.15	0.63
		HiCARN-1	28,283	0.25	0.16	**0.63**
		HiCARN-2	25,099	0.28	0.18	0.62
		HiCSR	19,656	0.32	0.22	0.60
		Capricorn	18,080	**0.34**	0.24	0.59
	Chromosight	LC	267	0.01	**0.24**	0.01
		HiCNN	48,557	0.17	0.11	0.43
		HiCARN-1	101,468	0.10	0.06	0.45
		HiCARN-2	95,614	0.10	0.06	0.44
		HiCSR	105,196	0.10	0.06	0.46
		Capricorn	43,390	**0.21**	0.14	**0.52**

Table A.4: **Cross-cell-line resolution enhancement performance based on identifiable loops using different loop calling methods and test cell lines.** "LC" indicates the low-coverage contact map provided as input to the resolution enhancement methods. The best-performing resolution enhancement method for each cell line and loop caller is shown in bold for each metric. Reported performance numbers are the average performance across test chromosomes (23 chromosomes for GM12878; 22 chromosomes for K562).

Ablation study name	$\mathbf{X}$	$\mathbf{X}^{(oe)}$	$(\mathbf{X}^{(loop-p)}, \mathbf{X}^{(loop-r)})$	$\mathbf{X}^{(tad)}$
HiC	✓			
HiC+Distance	✓	✓		
HiC+Loop	✓		✓	
HiC+TAD	✓			✓
Full−Distance	✓		✓	✓
Full−Loop	✓	✓		✓
Full−TAD	✓	✓		✓
Full	✓	✓	✓	✓

Table A.5: **Ablation studies conducted over Capricorn's multi-view framework.** We considered combining the original Hi-C contact matrices with only one additional source of information, as well the the full Capricorn framework missing only one source of information.

Cell line	Study name	Loop F_1 score
GM12878	HiC	0.451^*
	HiC+Distance	0.468^*
	HiC+Loop	0.494
	HiC+TAD	0.462^*
	Full−Distance	0.498
	Full−Loop	0.479^*
	Full−TAD	0.474^*
	Full	$\mathbf{0.500}$
K562	HiC	0.289^*
	HiC+Distance	0.325^*
	HiC+Loop	0.334^*
	HiC+TAD	0.340
	Full−Distance	0.337^*
	Full−Loop	0.312^*
	Full−TAD	0.351
	Full	$\mathbf{0.352}$

Table A.6: **Comparison of different view ablation study results in the Capricorn framework for both the GM12878 and K562 cell lines.** Experiments are conducted in the cross-cell-line setting. The average chromosome loop F_1 score using the HiCCUPS loop calling algorithm is shown. $*$ indicates that the result is significantly worse than the complete Capricorn framework (full; Wilcoxon signed-rank test with Benjamini-Hochberg [185] correction with a controlled FDR of 0.1). The best-performing Capricorn variant is shown in bold.

Method	Loops called	CTCF-validated loops
High-coverage	10,176	8370
Low-coverage	7021	2665
Capricorn	5686	3536

Table A.7: **Loop CTCF validation with CTCF ChIA-PET experimental validation, following the protocol in Roayaei Ardakany et al. [175].**

Secondary CTCF Validation

In addition to the CTCF-motif-based loop validation following the protocol of Rao et al. [11], here we also provide a CTCF-based validation following the steps in Roayaei Ardakany et al. [175] for GM12878. We downloaded a GM12878 CTCF ChIA-PET [280] experiment[1] from the Gene Expression Omnibus (GEO). The ChIA-PET experimental data identified 92,807 total peaks, indicating CTCF binding sites along the genome.

In this analysis, we considered a called loop to be "CTCF validated" if there was a CTCF ChIA-PET peak within 10 kb of the loop locus. Loops from the high-coverage, low-coverage, and Capricorn-enhanced contact matrices were called with HiCCUPS [11]. Results are shown in Table A.7.

A.3 Gemini

A.3.1 Input network statistics

We evaluate Gemini and other unsupervised, scalable network integration methods on two main network collections: the BioGRID [46] and STRING [45] databases. We downloaded the processed BioGRID networks processed from GeneMANIA [237, 256] for the mouse, human, and yeast gene networks. We download the processed STRING networks from Mashup [201], again focusing on the mouse, human, and yeast gene networks. Network statistics are shown in Table A.8. We also show the distribution of gene vertex degrees for each dataset in Figure A.16.

[1] GEO accession code: GSM1872886.

Network Collection	Organism	Genes n	Networks M	Average network size	Average node degree
BioGRID	mouse	$n = 21{,}081$	$M = 403$	10,283	23.7
	human	$n = 19{,}695$	$M = 895$	6802	18.0
	yeast	$n = 6365$	$M = 505$	1985	12.1
STRING	mouse	$n = 21{,}104$	$M = 6$	11,223	201.1
	human	$n = 18{,}362$	$M = 6$	7873	37.1
	yeast	$n = 6311$	$M = 6$	3221	42.1

Table A.8: **Network statistics for the BioGRID and STRING network collections.** The average network size reports the average number of nodes contained in each network, while the number of genes n is the number of unique genes contained across all M networks of the collection. The average node degree is the number of nodes divided by the number of edge, averaged over all networks in the collection.

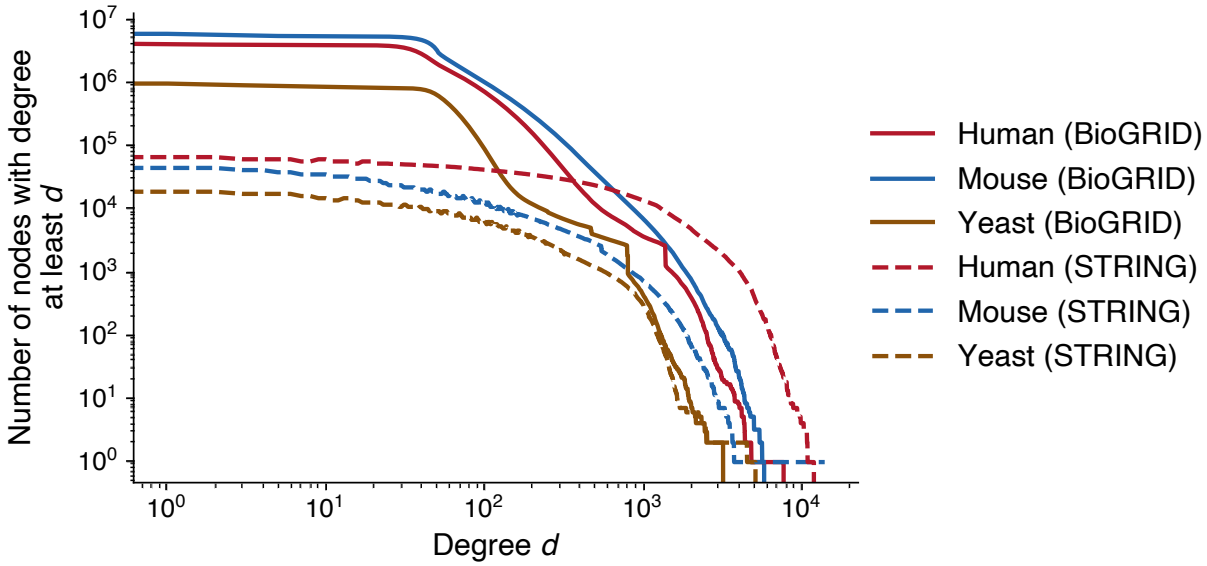


Figure A.16: **Number of nodes with degree at least d for each of the network collections.** The x-axis shows the range of node degrees in the dataset; the y-axis shows the number of nodes with degree at least $d = x$ in the given network collection, where each network's nodes are counted separately within a given dataset. BioGRID network statistics are plotted with solid lines while STRING network statistics are plotted with dashed lines; colors are used to differentiate human, mouse, and yeast datasets.

A.3.2 Comparison approaches

We benchmark Gemini against computational approaches that are able to scale to hundreds of gene networks. Given a collection of M gene networks represented by their adjacency matrices $\{\mathbf{A}_i\}_{i=1}^{M}$, we include the Mashup [201] and BIONIC [202] network integration methods. Furthermore, we compute the average adjacency matrix of each collection as $\bar{\mathbf{A}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{A}_i$ and use this average representation to compute network embeddings with PCA, SVD, and Mashup (denoted Average Mashup).

Mashup

To facilitate a quality comparison, we set Gemini and Mashup [201] to use the same model hyperparameters. Namely, we use the memory-efficient eigendecomposition Mashup framework, diffusion restart probability $\alpha = 0.5$, and gene embedding dimension of 200 for yeast and 400 for human and mouse. We re-implemented Mashup in python, and also used this framework to build Gemini.

BIONIC

We also compare to the BIONIC network integration approach for yeast networks, and for mouse and human STRING networks. For the yeast and human STRING network integration tasks, we used the default BIONIC repository configuration, including 3000 training epochs, a batch size of 2048, learning rate of 5e-4, a 64-dimension, 10 head, and 2-layer graph attention network (GAT), and embedding dimension of 512. For the mouse STRING integration task, we reduced the number of GAT layers to 1 in order to fit in GPU memory during inference, but otherwise retained the default hyperparameters. For all STRING integration tasks, we included all networks in each batch.

To train BIONIC on the BioGRID and STRING+BioGRID network integration collections, we reduced the size of the default architecture to fit in GPU memory and decreased the number of training epochs for a reasonable training speed. For the yeast BioGRID and STRING+BioGRID integrations we trained for 1000 epochs with a batch size of 256 and learning rate of 5e-4. We reduced the GAT architecture to 64 dimension, 10 heads, and 1 layer, with an output embedding size of 200. We included 10 networks per batch. We attempted to train BIONIC on the mouse and human BioGRID collections, but were unable to train even a very minimal BIONIC architecture on an A4000 GPU (e.g., out-of-memory error for batch size

of 128; GAT with 4 dimensions, 1 head, and 1 layer; output embedding dimension 8; 2 networks sampled per batch). These findings are also consistent with the scalability analysis conducted by BIONIC (Extended Data Figure 9 from Forster et al. [202]). We used the BIONIC pip package to train the BIONIC embeddings, providing the model and training details in a configuration file.

PCA

We take the average adjacency matrix $\bar{\mathbf{A}}$ of the input network collection and compute the top k principal components [103] as the output gene embeddings, with $k = 200$ for yeast and $k = 400$ for human and mouse.

SVD

We compute the truncated singular value decomposition of the average adjacency matrix $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \bar{\mathbf{A}}$, with the output gene embedding matrix $\mathbf{V} \in \mathbb{R}^{n \times k}$, for n the number of genes. Then, we take $k = 200$ for yeast and $k = 400$ for human and mouse.

Average Mashup

We also apply the Mashup method [201] with eigendecomposition to the average adjacency matrix $\bar{\mathbf{A}}$. As in the full Mashup method, we use a diffusion restart probability $\alpha = 0.5$ and use a gene embedding dimension of 200 for yeast and 400 for human and mouse.

A.3.3 Evaluation metrics

To evaluate the models, we compare the integrated gene embeddings' performance on a downstream protein function prediction task. We parameterize the downstream one-to-many classifier $f_\theta(\cdot)$ as a multi-layer perceptron with two hidden layers of 200- and 100-units respectively. Using the Gene Ontology Annotation (GOA) [258, 259] as ground-truth protein function labels, we perform 5-fold cross validation with 20% of the data used as the test split.

For each of the five test splits, we can compute the maximum F_1, macro-AUPRC, and micro-AUPRC. The macro- and micro- averaging strategies are both included to more fully capture the downstream task

Organism	BP terms	MF terms	CC terms
mouse	12,306	3070	1250
human	15,418	4408	1829
yeast	4997	2416	974

Table A.9: **Number of GO terms for each organism by sub-ontology.** Biological Process (BP), Molecular Function (MF), and Cellular Component (CC).

performance: macro-AUPRC weights every *class* (in this case, protein function label) equally, while micro-AUPRC weights every *datapoint* (in this case, gene) equally. For instance, a method with high micro-AUPRC but low macro-AUPRC might therefore struggle on rare class labels but perform well on common labels. We compute the average and standard deviation over these five splits for each metric. When comparing Gemini's integrated embeddings for an organism and network collection to an existing approach, we then use a one-sided paired t-test to test whether Gemini outperforms the existing approach with respect to a given metric on the five test splits.

In some settings we further stratify the results by GO sub-ontology (Figures 5.3 and A.17). To evaluate sub-ontology performance, we compute the maximum F_1, macro-AUPRC, and micro-AUPRC over all test splits when the output predictions are restricted to protein functions within the relevant sub-ontology. To compare different methods' performance on a given sub-ontology, we again use a one-sided paired t-test for each metric on the five test splits when restricted to the sub-ontology of interest. Table A.9 shows the number of GO terms associated with each organism from each sub-ontology.

A.3.4 Computational Complexity

Gemini space and time complexity

First, we conduct a theoretical study of Gemini's time and space complexity. We divide Gemini into five main steps: diffusion, pooling, clustering, mixup, and decomposition.

In the diffusion step, we perform random walk with restart (RWR) for each network. The RWR steady state can be solved for using matrix inversion, which dominates the computational complexity of this step. For M networks, each with n genes, this step therefore takes $O(Mn^3)$ time and $O(Mn^2)$ space. In the pooling step, we compute the kurtosis over each node in each network. Kurtosis-pooling for each node requires computing the mean and standard deviation over a vector of length n. Therefore, we can complete

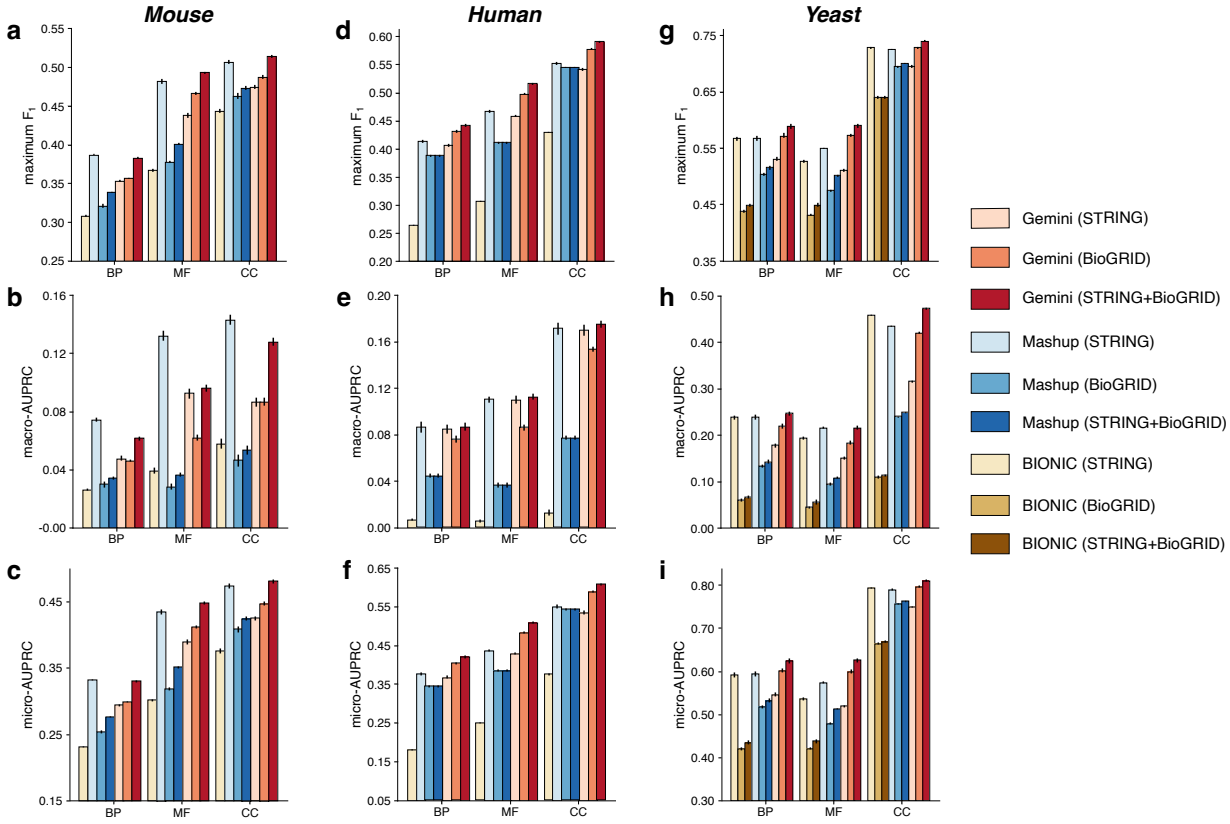


Figure A.17: **Comparison of network integration performance over different input network collections, stratified by GO sub-ontology.** Performance of Gemini, Mashup, and BIONIC when applied to the STRING, BioGRID, and combined STRING+BioGRID network collection datasets for Mouse (**a-c**), Human (**d-f**), and Yeast (**g-i**) on a downstream protein function prediction task. Performance is measured by maximum F_1 score (**a,d,g**), macro-AUPRC (**b,e,h**), and micro-AUPRC(**c,f,i**). BIONIC has no results for mouse and human on BioGRID and STRING+BioGRID inputs due to GPU memory constraints. Higher values indicate better performance on all metrics; error bars denote standard error over the five test splits.

the kurtosis-pooling step in $O(Mn^2)$ time and $O(Mn^2)$ space. In the clustering step, we use Affinity Propagation over the set of all networks as implemented in `sklearn` [89]. This has a time complexity of $O(M^2T)$, where $T = 200$ is the number of clustering iterations, and a space complexity $O(M^2)$. In the mixup step, we sample pairs of networks and then take computed weighted combination of the two diffusion state matrices. We repeat this mixup procedure K times, requiring $O(Kn^2)$ time and $O(Kn^2)$ space. Finally, in the eigendecomposition step, we take the sum over the K mixed-up networks transpose themselves, and then compute the eigenvectors, requiring $O(Kn^3)$ time and $O(n^2)$ space.

Overall, we can combine these steps to compute Gemini's overall time complexity as

$$O(Mn^3 + M^2T + Kn^3) = O(Mn^3 + Mn^2 + M^2T + Kn^2 + Kn^3) \tag{A.8}$$

and Gemini's overall space complexity as

$$O(Mn^2 + M^2 + Kn^2) = O(Mn^2 + Mn^2 + M^2 + Kn^2 + n^2). \tag{A.9}$$

In practice, we have $n \gg M$ for our networks and choose $K \leq M$ for our quantitative experiments. Using the default $T = 200$ also results in $T < M \ll n$, so Gemini's computational complexity reduces to $O(Mn^3)$ time and $O(Mn^2)$ space in these settings.

Ablation study comparison

To compare Gemini's time and memory usage in practice to other methods, we also conducted an ablation study in the number of networks. Here, we used the BioGRID yeast network collection [46] and varied the number of input networks as $M \in [5, 10, 50, 100, 250, 505]$, where 505 is the full BioGRID dataset. We then compared the runtime of Gemini, Mashup, and BIONIC in these settings, keeping the model hyperparameters fixed to the BioGRID experiment hyperparameters. We also compared the cpu and gpu peak utilization of each method, as measured by `tracemalloc` and `torch.cuda.max_memory_allocated`. Ablation results are shown in Figure A.18. Due to time constraints, we trained BIONIC for a reduced number of epochs for the ablation and extrapolated the training time to the 1000 epochs that we would have trained BIONIC for; model inference time was measured separately and accounted for in the results.

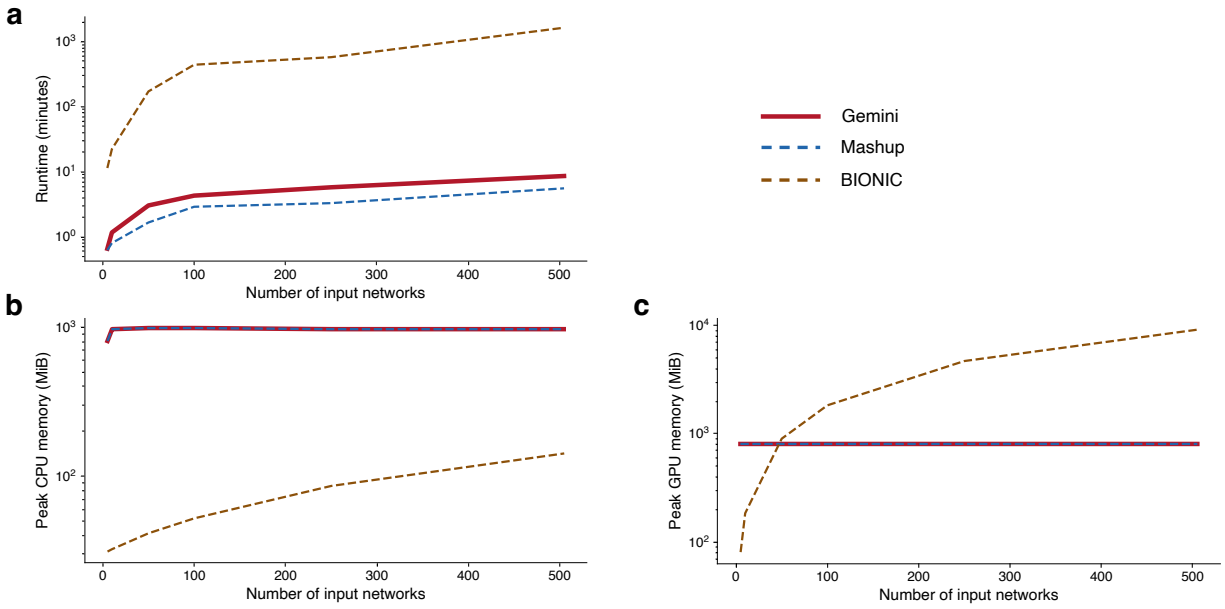


Figure A.18: **Computational requirements for network integration methods over different numbers of input gene networks.** We compare runtime (**a**), peak CPU memory utilization (MiB; **b**) and peak GPU memory utilization (MiB; **c**) of Gemini, Mashup, and BIONIC. For CPU and GPU utilization, Gemini and Mashup are practically indistinguishable.